LNCC
Laboratório Nacional de Computação Científica

Jornada de Ciência de Dados
11 anos!

CEFET/RJ

DATA VISUALIZATION

Eduardo Ogasawara
eduardo.ogasawara@cefet-rj.br
https://eic.cefet-rj.br/~eogasawara

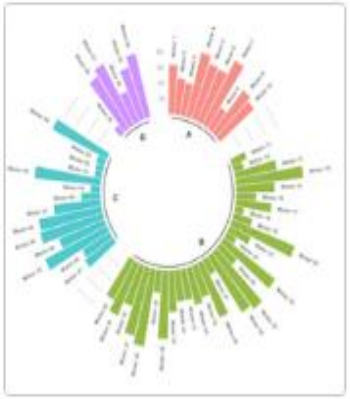
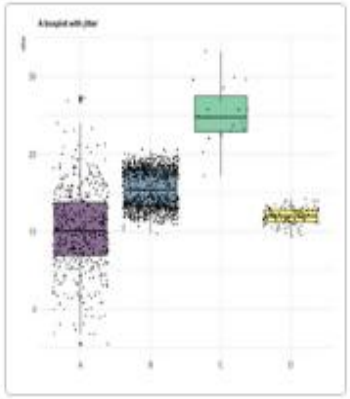
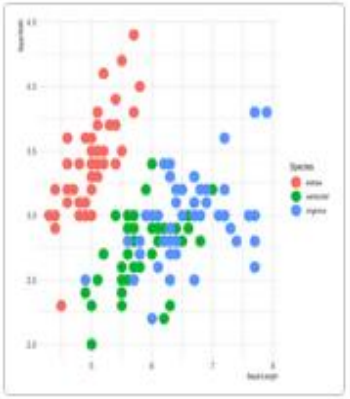
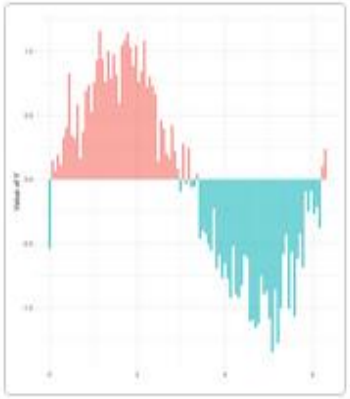
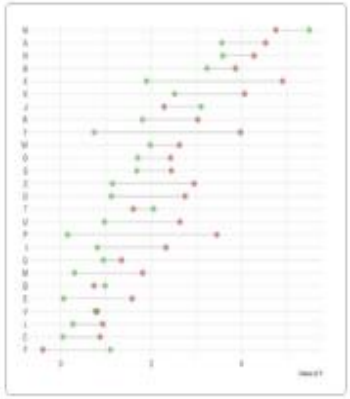
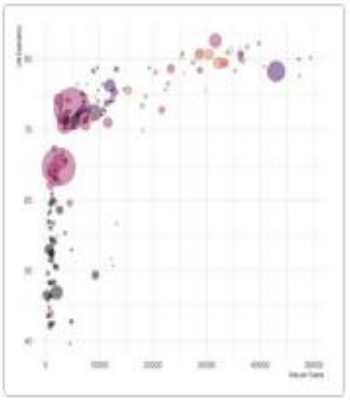
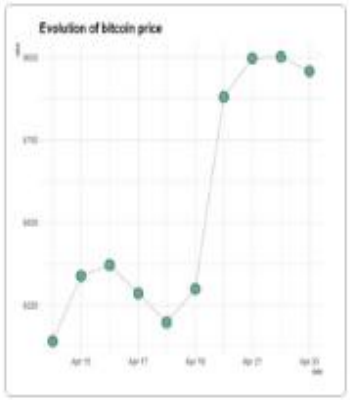
Plotting charts

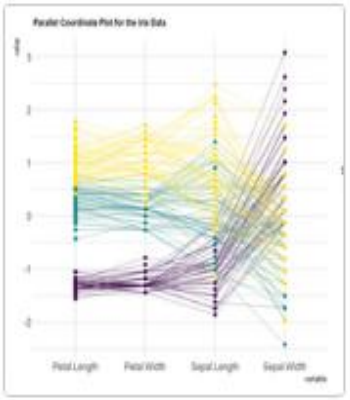
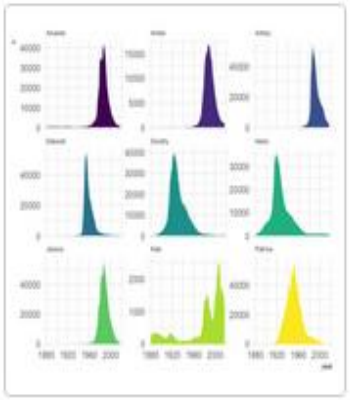
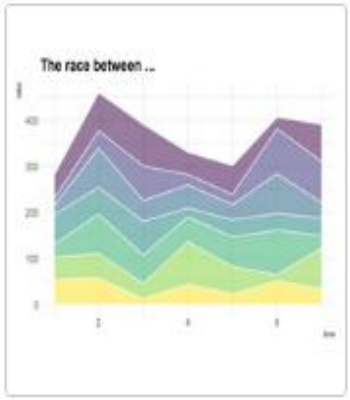
- R comes with basic plot functions
  - They are easy to operate for simple charts, but they are limited
- ggplot2 is a system for declaratively creating graphics, based on The Grammar of Graphics*.
- R for Data Science** is designed to give you a comprehensive introduction to ggplot2
  - The Data Visualization and Graphics for communication chapters

```
# The easiest way to get ggplot2 is to install the whole tidyverse
#install.packages("tidyverse")
#install.packages("ggplot2")
```

(*) https://www.amazon.com/Grammar-Graphics-Statistics-Computing/dp/0387245448
(**) https://r4ds.had.co.nz/

Examples of ggplot2 charts

https://www.r-graph-gallery.com/ggplot2-package.html

ggplot2 cheat sheet

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>) +                                   required
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
 stat = <STAT>, position = <POSITION>) +                   Not required, sensible defaults supplied
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>
```

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.

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian")
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly()
x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5)
x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy))
x, y, alpha, color, fill, linetype, size, weight

TWO VARIABLES both continuous

e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl")
x, y, alpha, color, linetype, size

e + geom_smooth(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

one discrete, one continuous

f <- ggplot(mpg, aes(class, hwy))

f + geom_col()
x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot()
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center")
x, y, alpha, color, fill, group

f + geom_violin(scale = "area")
x, y, alpha, color, fill, group, linetype, size, weight

https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf

ggplot easy encapsulation through daltoolbox

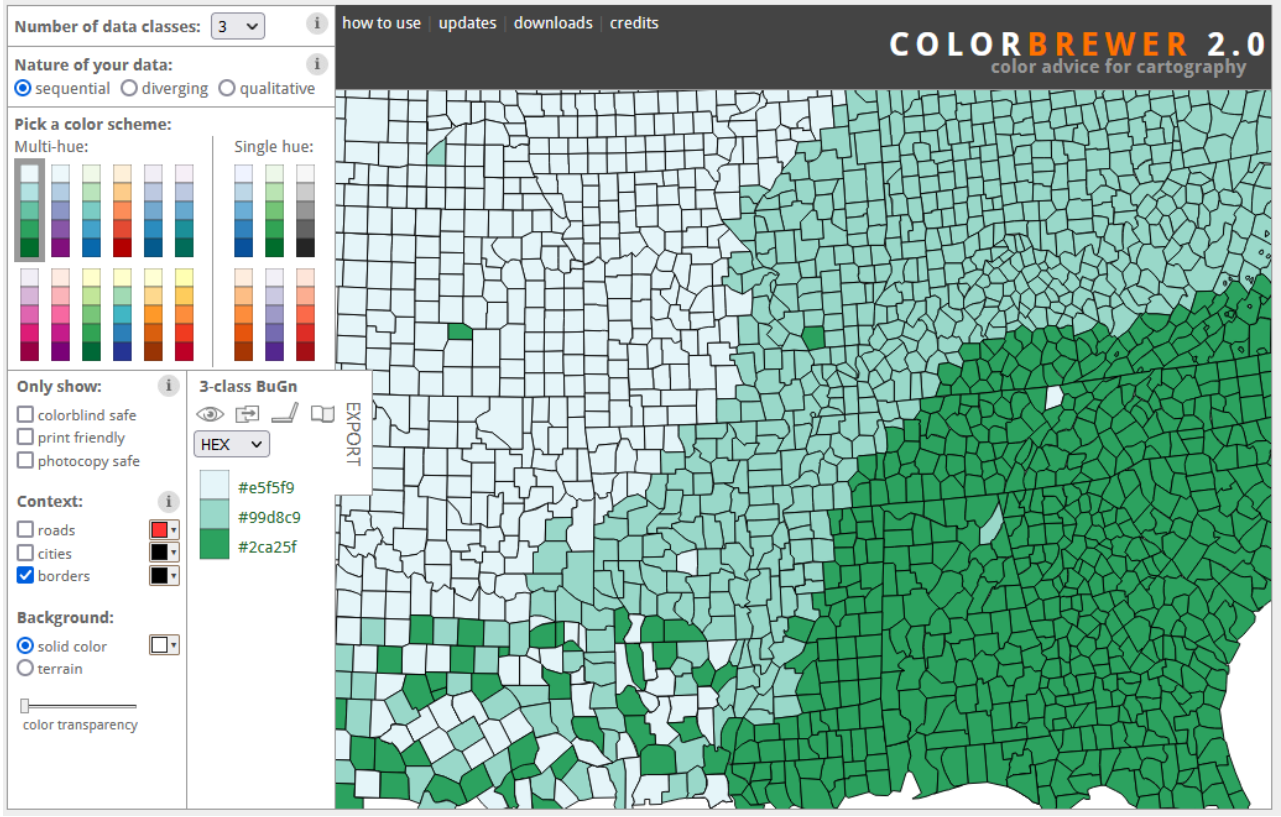
- The DAL Toolbox enables plotting charts encapsulating ggplot2
- It enables an easy startup while learning how to use ggplot2
- Most functions require a data.frame with two basic parameters
  - The first parameter is usually associated with the x-axis: x
  - The second parameter is related to the y-axis: value
  - Sometimes, the third parameter might be a group variable definition
- The DAL Toolbox is loaded using library function

```
library(daltoolbox)
library(ggplot2)
```

https://github.com/cefet-rj-dal/daltoolbox/tree/main/graphics

But before continuing, lets talk about colors…

- One thing very relevant while plotting charts is clear pass the idea also preserving visual identity
  - The color brewer is an excellent tool to set up colors for your graphics using appropriate colors

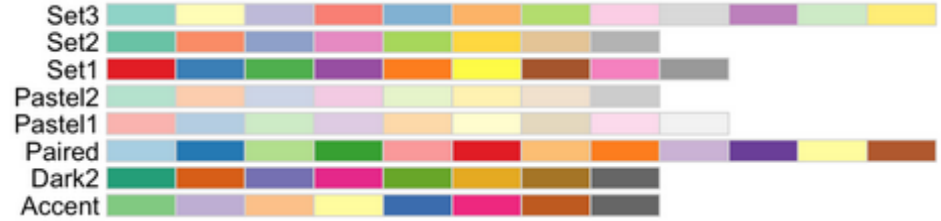


Color Brewer R Package

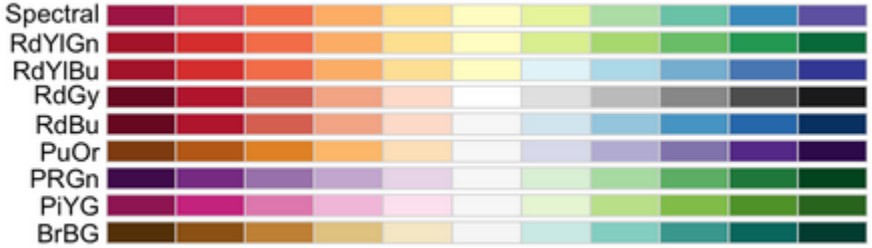
```
#install.packages("RColorBrewer")
library(RColorBrewer)
colors <- brewer.pal(4, 'Set1')
```

Sequential

Qualitative



Diverging



https://rdrr.io/cran/RColorBrewer/man/ColorBrewer.html

Basic setup for examples

Iris datasets

```
head(iris, 3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
##   Species
## 1  setosa
## 2  setosa
## 3  setosa
```

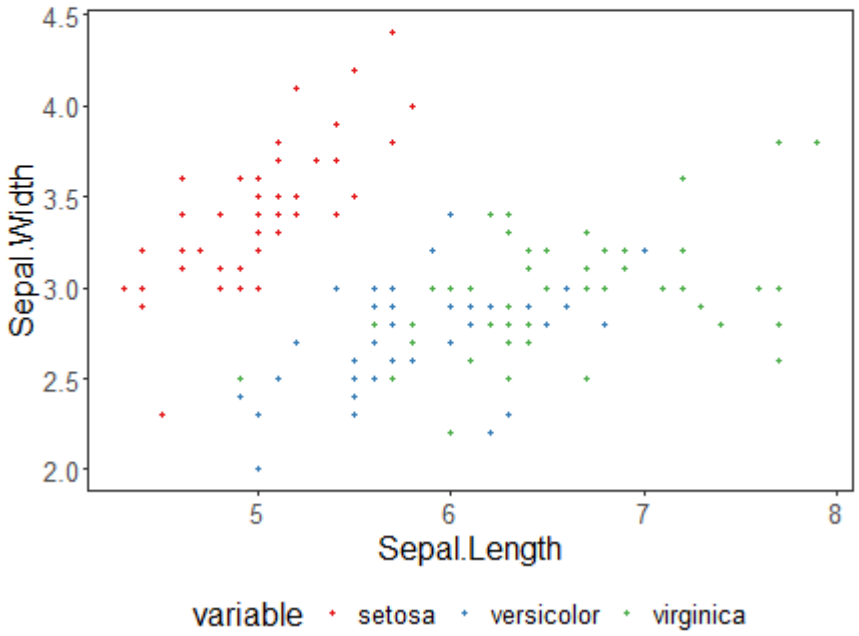
Options from graphics: colors and font size

```
colors <- brewer.pal(4, 'Set1')
font <- theme(text = element_text(size=16))
```

Scatter plot

- Display values for the typical relationship between variables

```
library(dplyr)
data <- iris |> select(x = Sepal.Length, value = Sepal.Width, variable = Species)
#head(data)
grf <- plot_scatter(data, label_x = "Sepal.Length", label_y = "Sepal.Width", colors=colors[1:3]) + font
plot(grf)
```

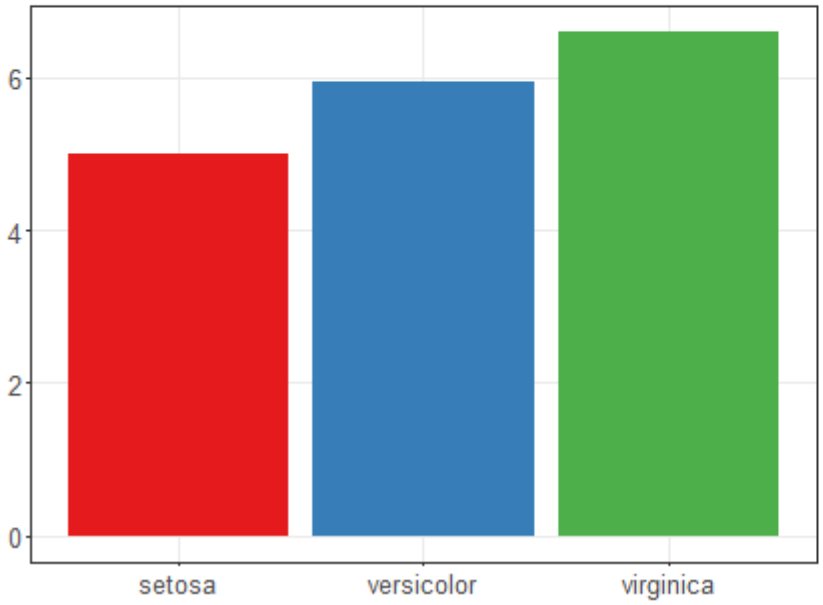


https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_scatter.md

Bar graph

- Presents categorical data with rectangular bars proportional to values

```
library(dplyr)
data <- iris |> group_by(Species) |> summarize(Sepal.Length=mean(Sepal.Length))
#head(data)
grf <- plot_bar(data, colors=colors[1:3]) + font
plot(grf)
```

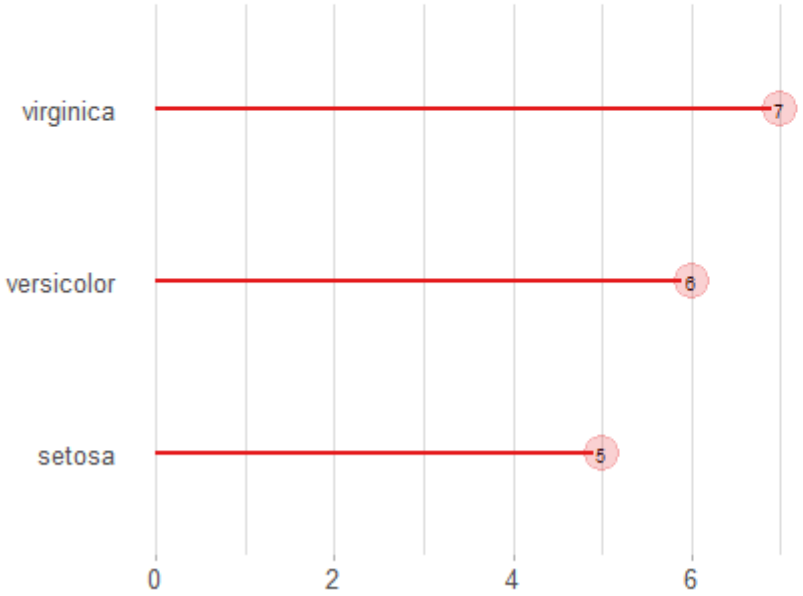


https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_bar.md

Lollipop plot

- The lollipop graph has the same goal as a horizontal bar graph

```
library(dplyr)
data <- iris |> group_by(Species) |> summarize(Sepal.Length=mean(Sepal.Length))
#head(data)
grf <- plot_lollipop(data, colors=colors[1], max_value_gap=0.2) + font + coord_flip()
plot(grf)
```

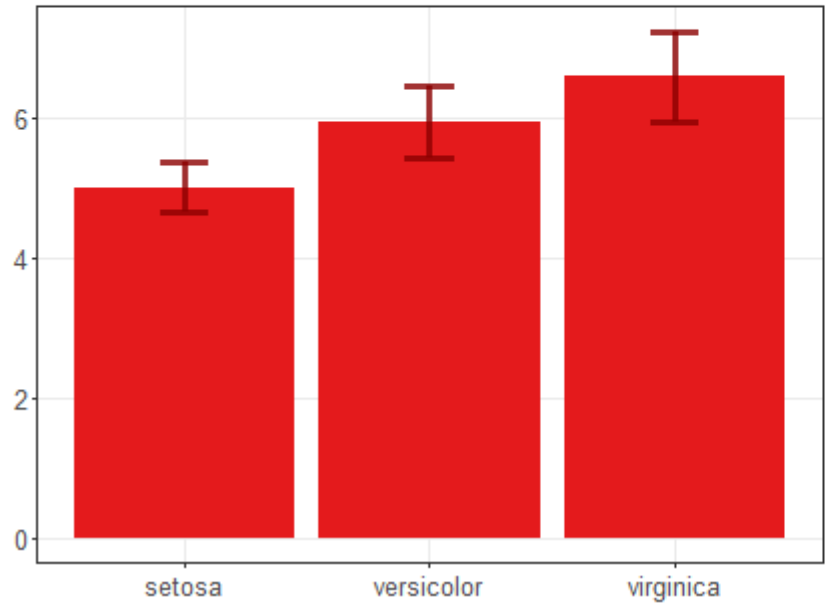


https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_lollipop.md

Bar graph with error bars

- Average behavior with dispersion

```
library(dplyr)
data <- iris |> group_by(Species) |> summarize(mean=mean(Sepal.Length), sd=sd(Sepal.Length))
#head(data)
grf <- plot_bar(data, colors=colors[1], alpha=1) + font
grf <- grf + geom_errorbar(aes(x=Species, ymin=mean-sd, ymax=mean+sd),
                           width=0.2, colour="darkred", alpha=0.8, size=1.1)
plot(grf)
```

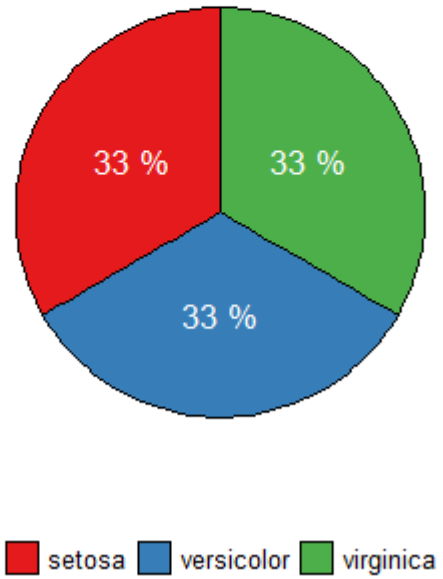


https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_bar_error.md

Pie chart

- Circular statistical graphic to illustrate numerical proportion

```
library(dplyr)
data <- iris |> group_by(Species) |> summarize(n = n())
#head(data)
grf <- plot_pieplot(data, colors=colors[1:3]) + font
plot(grf)
```



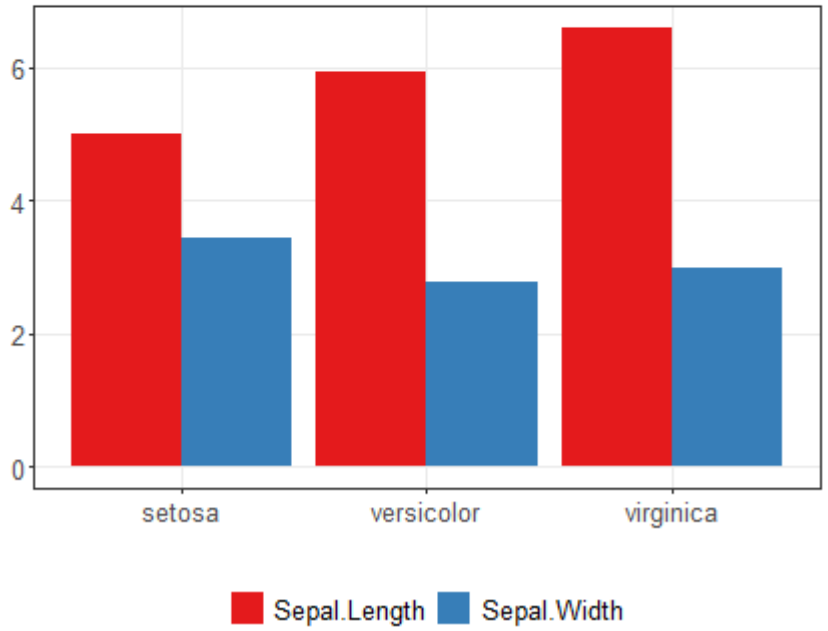
https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_pie.md

Grouped bar

- Organize data into groups for each category

```
library(dplyr)
data <- iris |> group_by(Species) |> summarize(Sepal.Length=mean(Sepal.Length), Sepal.Width=mean(Sepal.Widt

#head(data)
grf <- plot_groupedbar(data, colors=colors[1:2]) + font
plot(grf)
```



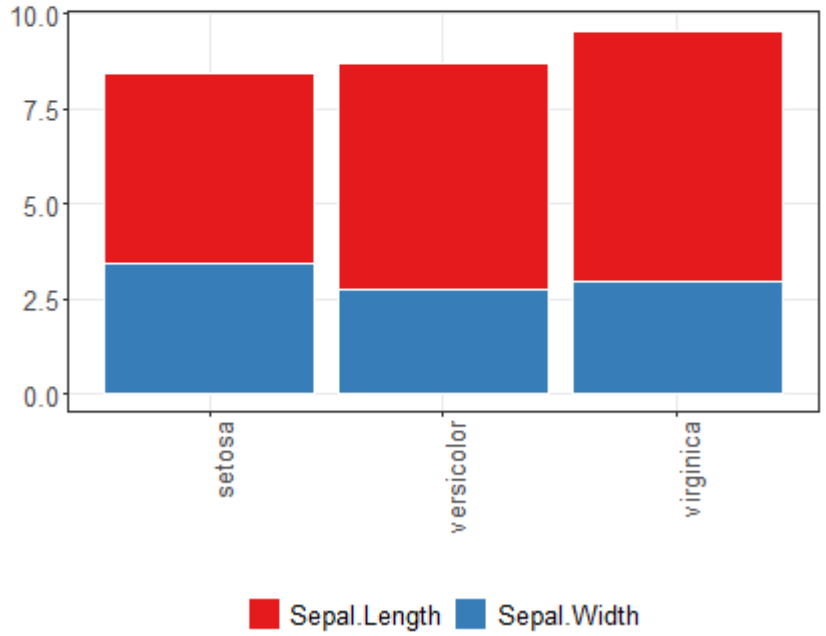
https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_grouped_bar.md

Stacked-bars

- The height of the bar shows the combined result of the groups

```
library(dplyr)
data <- iris |> group_by(Species) |> summarize(Sepal.Length=mean(Sepal.Length), Sepal.Width=mean(Sepal.Widt

#head(data)
grf <- plot_stackedbar(data, colors=colors[1:2]) + font
grf <- grf + theme(axis.text.x = element_text(angle=90, hjust=1))
plot(grf)
```



https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_stacked_bar.md

Data Distribution Analysis

Examples using data distribution

- The following examples use random variables so that different data distribution can be better viewed

```
example <- data.frame(exponencial = rexp(10000, rate = 1),
                      uniform = runif(10000, min=2.5, max = 3.5),
                      normal = rnorm(10000, mean = 5))
head(example)
```

```
##   exponencial  uniform   normal
## 1  0.65926515 2.577149 3.380406
## 2  0.96877025 2.937234 5.967816
## 3  0.05059315 2.758117 3.307216
## 4  0.12027304 2.565390 4.780961
## 5  5.31988220 2.641392 4.713785
## 6  2.17472525 2.591428 3.539582
```

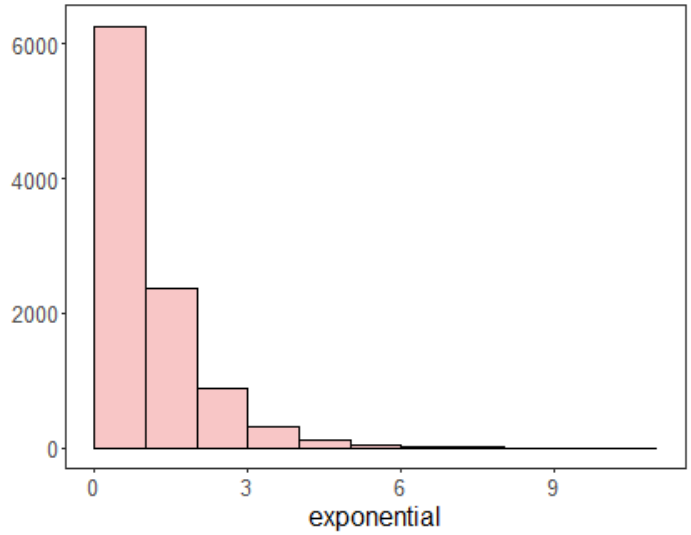
Histogram

- Counting of a single continuous variable organized into bins

```
library(dplyr)
data <- example |> select(exponential)
#head(data)
grf <- plot_hist(data, label_x = "exponential", color=colors[1]) + font
```

```
## Using  as id variables
```

```
plot(grf)
```

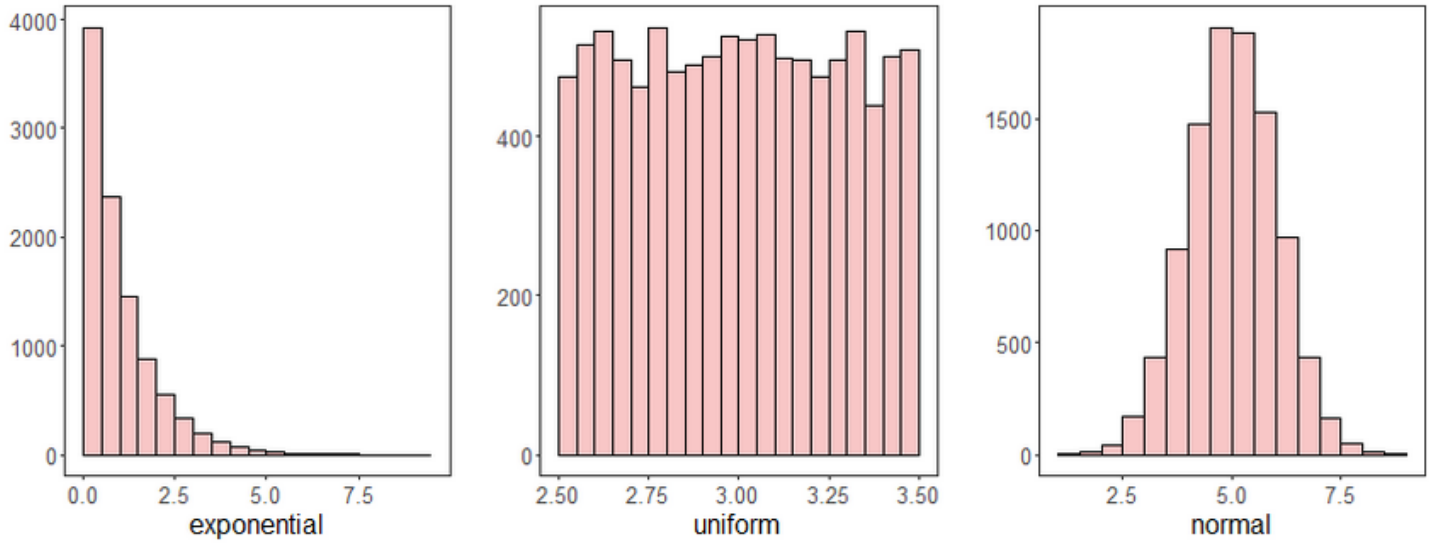


https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_histogram.md

Multiple histograms

```
{
library(gridExtra)
grfe <- plot_hist(example |> select(exponential), label_x = "exponential", color=colors[1]) + font
grfu <- plot_hist(example |> select(uniform), label_x = "uniform", color=colors[1]) + font
grfn <- plot_hist(example |> select(normal), label_x = "normal", color=colors[1]) + font
grid.arrange(grfe, grfu, grfn, ncol=3)
}
```

```
## Using  as id variables
## Using  as id variables
## Using  as id variables
```



https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_histogram.md

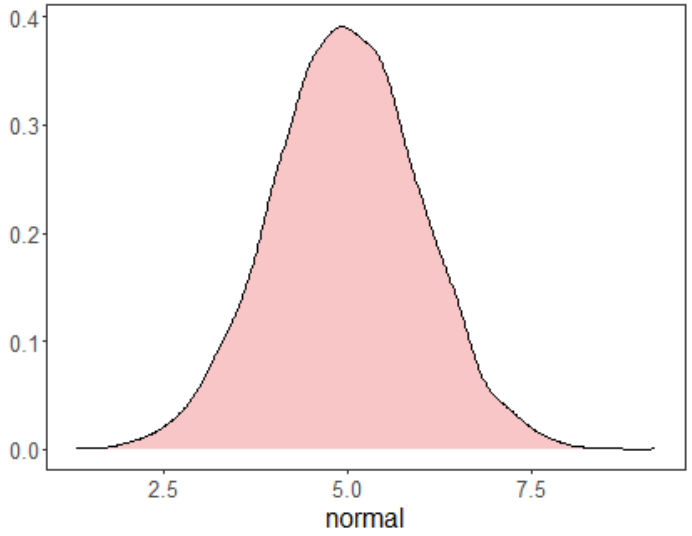
Density plot

- Draws kernel density estimate

```
library(dplyr)
data <- example |> select(normal)
#head(data)
grf <- plot_density(data, label_x = "normal", color=colors[1]) + font
```

```
## Using  as id variables
```

```
plot(grf)
```



https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_density.md

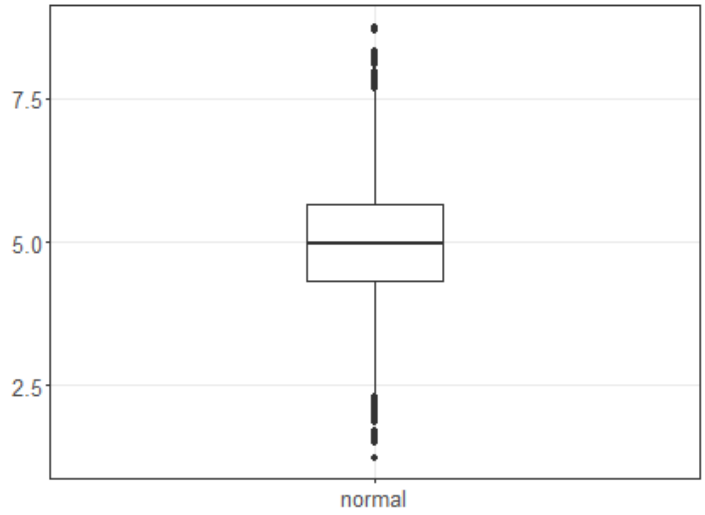
Box-plot

- Groups numerical data through their quartiles

```
library(dplyr)
data <- example |> select(normal)
#head(data)
grf <- plot_boxplot(data, color="white") + font
```

```
## Using  as id variables
```

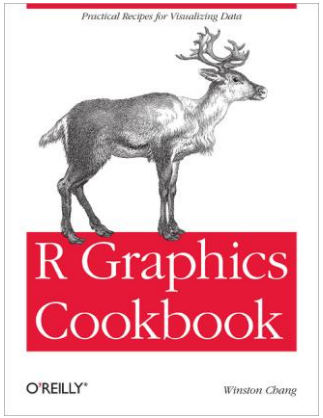
```
plot(grf)
```



https://github.com/cefet-rj-dal/daltoolbox/blob/main/graphics/grf_boxplot.md

Main References

Código: https://github.com/eogasawara/analise-dados/blob/main/examples/3-DataVisualization.md



Further reading: R Graphics Cookbook, Chapters 1-6, 12

Slides e vídeos em: https://eic.cefet-rj.br/~eogasawara/analise-de-dados/

- https://rstudio-education.github.io/hopr/index.html
- https://r-graphics.org