

International Seminar
on Statistics with R

Visualização de dados
através do pacote
dubois

por Ícaro Bernardes
Maio de 2022



Bacharelado em
Eng. Química

Mestrando em
Eng. Industrial

Co-fundador e
cientista de dados

UFBA

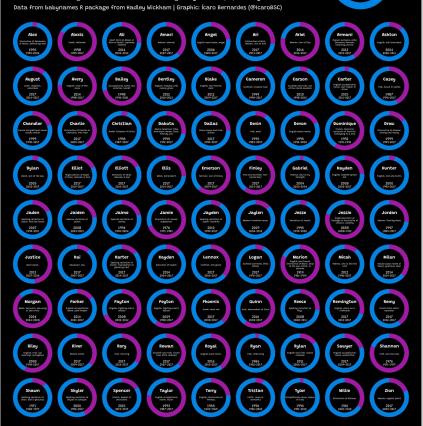
UFBA

BIT::Analytics

What is in a name?

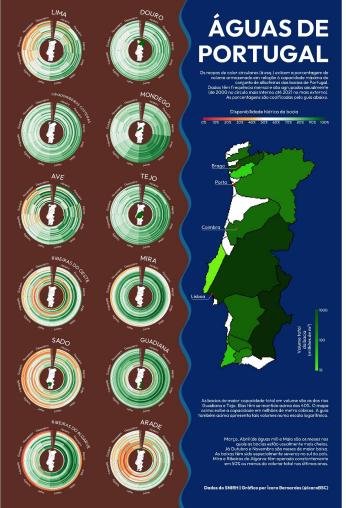
Of the myriad of names given to babies in the US between 1880 and 2017, most were unique. But some names were so popular they became neutral names that were in the top 70% for both sexes for at least two years. The sections represent the percentage of all males and females born that were given a certain name during its gender's popularity.

Data from [Nameberry](#) (percentage from [nameberry.com](#)) and [Babynamerank](#) ([@Babnamerank](#))



TidyTuesday

30DayChartChallenge



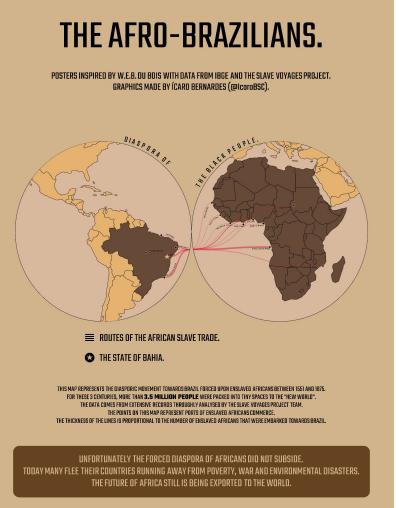
DuBoisChallenge

Afiliação e trabalhos em dataviz



THE AFRO-BRASILIANS.

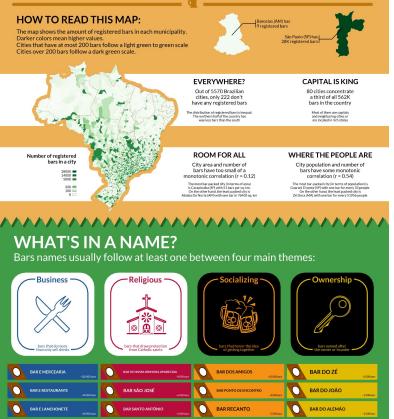
POSTERS INSPIRED BY W.E.B. DU BOIS WITH DATA FROM IBGE AND THE SLAVE VOYAGES PROJECT.
GRAPHICS MADE BY IACO BERNARDES (@icacode95).



DuBoisChallenge

BARS ALL AROUND

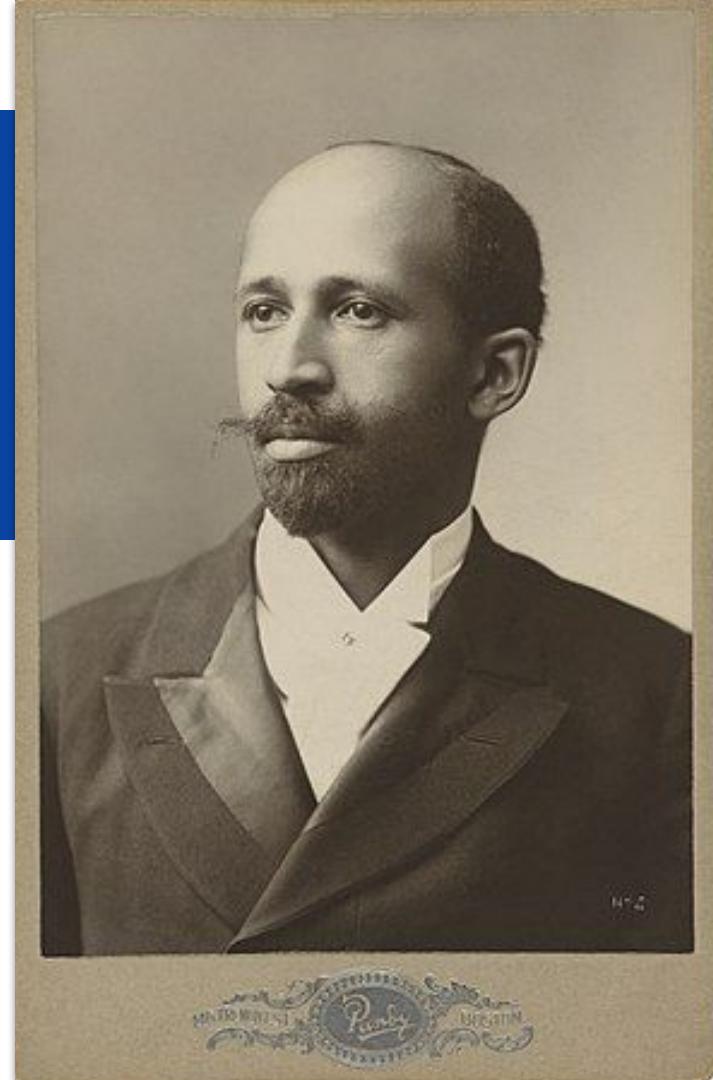
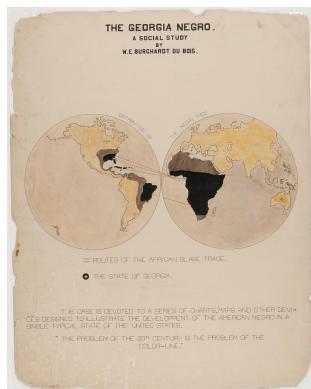
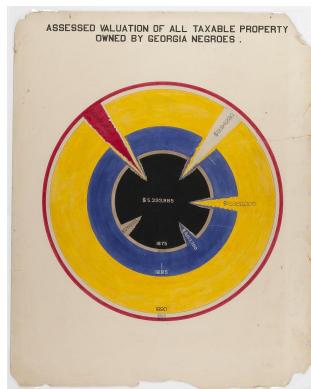
Bar is a gathering place for many Brazilians and are extremely linked to cultural symbols like Carnaval and Capoeira. However they are not solely Carnaval and Leisure.



Extras

Du Bois

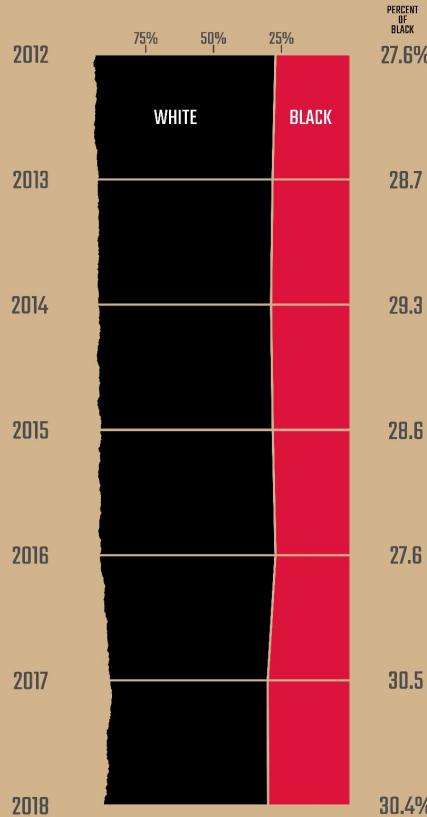
Sociólogo americano e
gênio do *dataviz*



Entendendo a função dubois::db_area

PARTICIPATION IN MANAGERIAL POSITIONS BY RACE IN BRAZIL.

INSPIRED BY: W.E. DU BOIS | DATA FROM: IBGE | GRAPHIC BY: ICARO BERNARDES



0. Argumentos da função (linhas 33-154)

1. Uso de {showtext} (linhas 156-170)

2. Manejo dos dados (linhas 172-302)

3. Produção do gráfico (linhas 304-374)

IN THE SERIES, USUALLY WHITES OCCUPY SLIGHTLY LESS GENERAL WORK POSITIONS.
HOWEVER WHITES OCCUPY WAY MORE MANAGERIAL POSITIONS THAN BLACKS

```

# Carrega o pacote e outros auxiliares
library(dubois)

# Busca os dados sobre trabalhadores
# em cargos de comando contido no pacote
data <- dubois::managers

```

total_bosses_per_1000	total_workers_per_1000	race	year	pct_bosses_total	pct_workers_total	cv_pct_bosses
3035	92333	white	2018	68.6	45.2	1.5
3035	92333	black	2018	29.9	53.7	3.3
3294	91073	white	2017	68.0	45.9	1.4
3294	91073	black	2017	29.8	53.1	3.4
3516	90776	white	2016	71.3	46.6	1.6
3516	90776	black	2016	27.2	52.5	4.0
3650	92163	white	2015	70.3	47.2	1.2
3650	92163	black	2015	28.2	52.1	3.1
3600	91945	white	2014	69.4	47.9	1.2
3600	91945	black	2014	28.7	51.3	2.7
3450	90715	white	2013	70.1	48.1	1.1
3450	90715	black	2013	28.2	51.1	2.8
3516	90776	white	2012	71.3	46.6	1.6
3516	90776	black	2012	27.2	52.5	4.0

```

# Carrega o pacote e outros auxiliares
library(dubois)

# Busca os dados sobre trabalhadores
# em cargos de comando contido no pacote
data <- dubois::managers

# Faz pequenas manipulações nos dados
data <- data %>%
  dplyr::select(race, year, pct_bosses_total) %>%
  tidyr::pivot_wider(names_from = "race",
                     values_from = "pct_bosses_total")

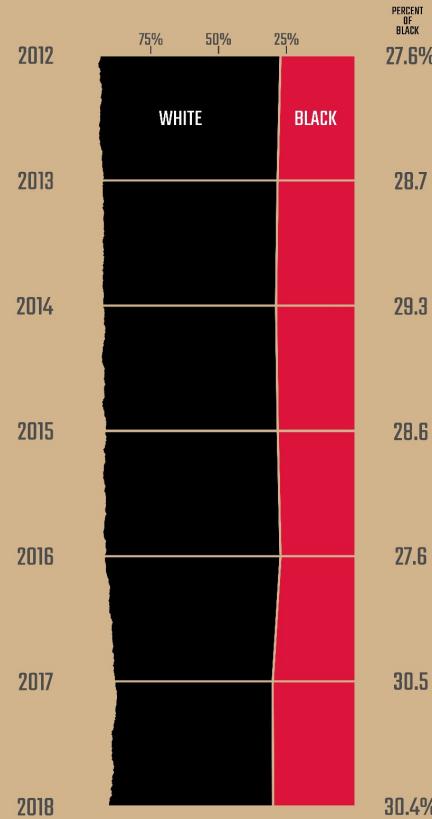
# Define título, subtítulo e mensagem
title <- "PARTICIPATION IN MANAGERIAL POSITIONS BY RACE IN BRAZIL."
subtitle <- "INSPIRED BY: W.E.B. DU BOIS | DATA FROM: IBGE | GRAPHIC BY: ICARO BERNARDES"
message <- "IN THE SERIES, USUALLY WHITES OCCUPY SLIGHTLY LESS GENERAL WORK POSITIONS. HOWEVER WHITES OCCUPY WAY MORE MANAGERIAL POSITIONS THAN BLACKS"

# Faz uso da função. A figura é
# salva no working directory
dubois::db_area(data = data, order = "year",
                 cat1 = "black", cat2 = "white",
                 limits = c(-3,4), # limites para o random walk
                 filename = "managers.png",
                 title = title, subtitle = subtitle,
                 message = message)

```

PARTICIPATION IN MANAGERIAL POSITIONS BY RACE IN BRAZIL.

INSPIRED BY: W.E.B. DU BOIS | DATA FROM: IBGE | GRAPHIC BY: ICARO BERNARDES



IN THE SERIES, USUALLY WHITES OCCUPY SLIGHTLY LESS GENERAL WORK POSITIONS.
HOWEVER WHITES OCCUPY WAY MORE MANAGERIAL POSITIONS THAN BLACKS

Entendendo a função dubois::db_area

0. Argumentos da função (linhas 63-154)

1. Uso de {showtext} (linhas 156-170)

2. Manejo dos dados (linhas 172-302)

3. Produção do gráfico (linhas 304-374)



```
# 0. Verificação inicial dos argumentos
## Confirma a classe dos argumentos
verify_class_fun <- function(arg, class) {
  sym = rlang::sym(arg)
  if (!(class %in% class(rlang::eval_tidy(sym)))) {
    stop(glue::glue("{arg} has to be a {class} or similar"), call. = FALSE)
  }
}
verify_class_data <- tibble::tibble(
  arg = c("data", "order", "cat1", "cat2",
         "dpi", "seed", "res_step", "limits",
         "names", "title", "subtitle", "message", "path", "filename"),
  class = c("data.frame", rep("character", 3), rep("numeric", 4), rep("character", 6)))
)
purrr::pwalk(verify_class_data, ~verify_class_fun(.x, .y))
```

Como é feita a verificação:

1. Uma função é criada para verificar alguma condição e parar a operação se preciso
2. Uma tibble guarda as variáveis e valores esperados para a condição ser falsa
3. purrr::pwalk aplica a função na tibble

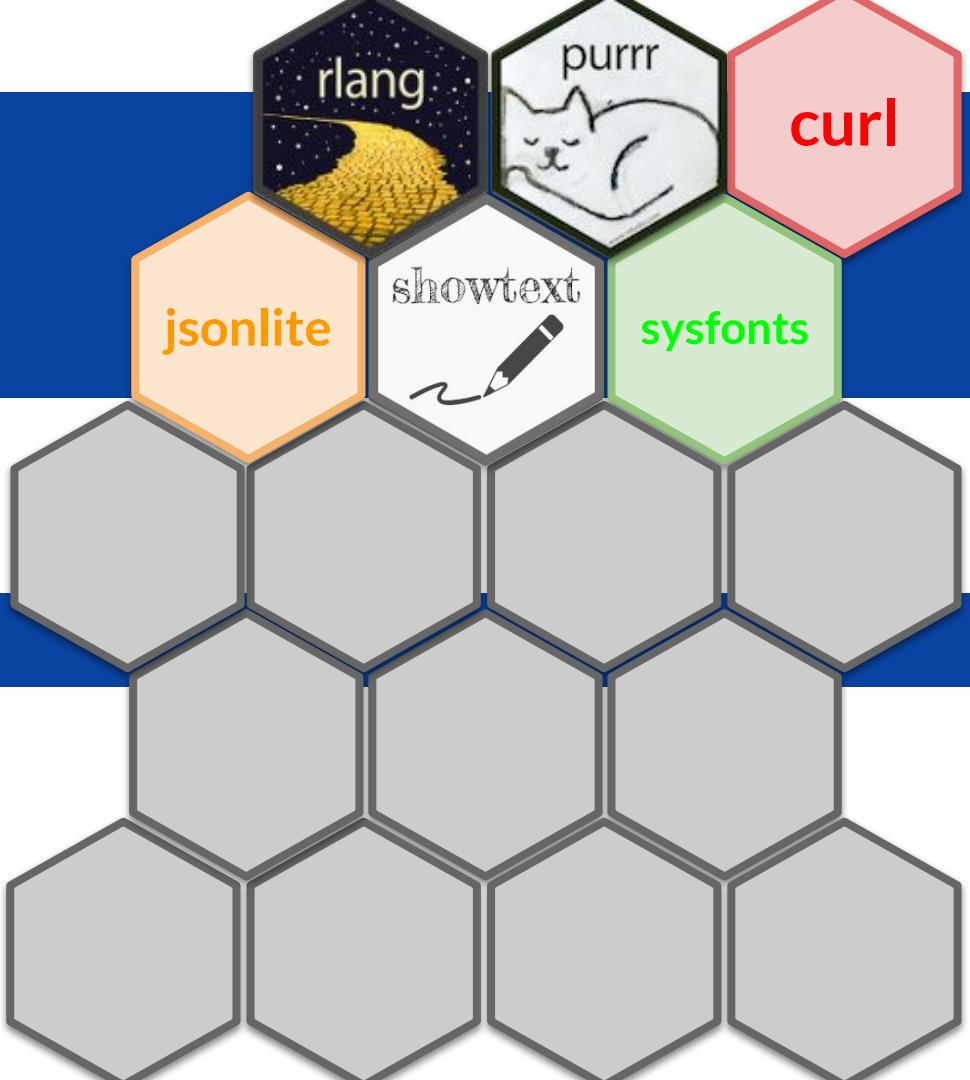
Entendendo a função dubois::db_area

0. Argumentos da função (linhas 63-154)

1. Uso de {showtext} (linhas 156-170)

2. Manejo dos dados (linhas 172-302)

3. Produção do gráfico (linhas 304-374)



```
# 1. Cuida da impressão do texto na imagem
## Verifica se a fonte "Teko" está disponível para ser
## usada por showtext e baixa ela se estiver ausente
if (!("Teko" %in% sysfonts::font_families())) {
  sysfonts::font_add_google(name = "Teko")
}

## Define a resolução de textos impressos por showtext nos dispositivos gráficos
showtext::showtext_opts(dpi = dpi)

## Ativa o controle da impressão do texto por parte de showtext.
## ATENÇÃO! showtext tem problemas na interalação com alguns plots
## (notadamente do pacote circlize), então é melhor criar eles
## primeiro e depois usar esse pacote
showtext::showtext_auto()
```

Como usar o {showtext}:

- 1.** Obtenha a fonte, caso não esteja instalada em sua máquina
- 2.** Ateve o {showtext} para que ele maneje o dispositivo gráfico
- 3.** Use o nome da família da fonte no argumento “family” de geom_text e afins!

```
sysfonts::font_add_google(name = "Teko")
```

```
showtext::showtext_auto()
```

```
geom_text(family = "Teko")
```

Entendendo a função dubois::db_area

0. Argumentos da função (linhas 63-154)

1. Uso de {showtext} (linhas 156-170)

2. Manejo dos dados (linhas 172-302)

3. Produção do gráfico (linhas 304-374)



```
# 2. Maneja os dados
## Elimina linhas com dados ausentes
data <- data %>%
  dplyr::filter(dplyr::if_all(.fns = ~ !is.na(.)))

## Obtém o número total de observações
n_obs <- dim(data)[1]

## Mantém apenas os dados da ordem das observações e os
## valores das duas categorias. Também as renomeia
data <- data %>%
  dplyr::select(order, cat1, cat2) %>%
  dplyr::rename(
    "order" = order,
    "cat1" = cat1,
    "cat2" = cat2
)
```

```

## Verifica se a soma do par de categorias é igual a 100%.
## Caso não seja, converte as categorias a porcentagens que o são.
check <- data %>%
  dplyr::mutate(
    pair = cat1 + cat2,
    pair == 100
  ) %>%
  dplyr::summarise(pair = sum(pair)) %>%
  dplyr::mutate(pair = (pair == n_obs)) %>%
  dplyr::pull(pair)

```

```

if (!check) {
  data <- data %>%
    dplyr::mutate(
      total = cat1 + cat2,
      cat1 = 100 * cat1 / total,
      cat2 = 100 * cat2 / total
    ) %>%
    dplyr::select(-total)
}

```

```

## Garante que a variável de ordem é ordenada. Se é um character,
## converte a factor e toma os níveis na ordem que eles aparecem
if (!is.numeric(data$order)) {
  if (!is.factor(data$order)) {
    data <- data %>%
      dplyr::mutate(order = factor(order, levels = unique(order)))
  }
}
data <- data %>% dplyr::arrange(order)

```



Ordem de elementos discretos

Pôr uma variável de classe *character* como *factor* permite escolher o ordenamento dos valores da variável na exibição no gráfico

```
## Cria os rótulos para a categoria destacada
highlight <- data %>%
  dplyr::mutate(
    cat1 = round(cat1, digits = 1),
    cat1 = ifelse(dplyr::row_number(order) == 1L | dplyr::row_number(order) == dplyr::n(),
                  paste0(cat1, "%"),
                  cat1
    )
  ) %>%
  dplyr::pull(cat1)

## Obtém o primeiro e último itens entre as
## categorias da ordem e converte elas a números
ord1 <- data %>%
  dplyr::slice(1L) %>%
  dplyr::mutate(order = as.numeric(order)) %>%
  dplyr::pull(order)
ord2 <- data %>%
  dplyr::slice(dplyr::n()) %>%
  dplyr::mutate(order = as.numeric(order)) %>%
  dplyr::pull(order)
```

```
## Calcula uma área à esquerda do gráfico
## usando algo como um bounded random walk
withr::local_seed(seed)
lft_area <- tibble::tibble(
  y = seq(ord1, ord2, res_step * (ord2 - ord1))
) %>%
  dplyr::rowwise() %>%
  dplyr::mutate(r = stats::rnorm(n = 1)) %>%
  dplyr::ungroup() %>%
  dplyr::mutate(
    z = cumsum(r),
    z = scales::rescale(z, to = c(limits[1], limits[2])))
) %>%
  dplyr::mutate(
    x = z + 2 * mean(data$cat1) / 3 + mean(data$cat2),
    x = ifelse(x > 100, 100, x)
)
```

```

## Define os nomes das categorias
categ_names <- data %>%
  dplyr::slice(1L:2L) %>%
  dplyr::summarise(
    x1 = mean(cat1) / 2,
    x2 = mean(cat2) / 2 + mean(cat1),
    y = mean(as.numeric(order)))
) %>%
tidyverse::pivot_longer(
  cols = c("x1", "x2"),
  names_to = "varname",
  values_to = "x")
) %>%
dplyr::select(-varname) %>%
dplyr::mutate(label = toupper(names))

## Define o título, título do eixo secundário e mensagem do gráfico
title <- paste0(toupper(title), "<br><span style='font-size:60px; '>", toupper(subtitle), "</span>")
sectitle <- paste0("PERCENT<br>OF<br>", toupper(names[1]))
message <- toupper(message) %>%
  stringr::str_wrap() %>%
  stringr::str_replace_all(pattern = "\n", "<br>")

```



Funções do {ggtext}

Alternativa ao geom_text que permite pôr cor de fundo com e sem contorno, aceita algumas tags HTML e hjust e vjust como argumento dentro de aes()

```
## Define algumas constantes de layout
lnhgt <- 0.8
bgcolor <- "#d2b48c"
l_marg <- 750 - 10 * max(stringr::str_length(data$order))
lb_sz <- 60 - 2 * max(stringr::str_length(data$order))
if (lb_sz < 30) {
  lb_sz <- 30
}

## Reorganiza os dados
data <- data %>%
  tidyr::pivot_longer(
    cols = c("cat1", "cat2"),
    names_to = "categ",
    values_to = "pct"
  )

## Cria uma nova variável numérica com base na ordem
data <- data %>%
  dplyr::mutate(num_order = as.numeric(order))
```



Fonte responsiva

Um truque para criar um texto cuja fonte escala com tamanho do conteúdo é escolher famílias que sejam monoespaçadas e fazer o *font-size* inversamente proporcional ao comprimento da *string*

Entendendo a função dubois::db_area

0. Argumentos da função (linhas 63-154)
1. Uso de {showtext} (linhas 156-170)
2. Manejo dos dados (linhas 172-302)
3. Produção do gráfico (linhas 304-374)



```

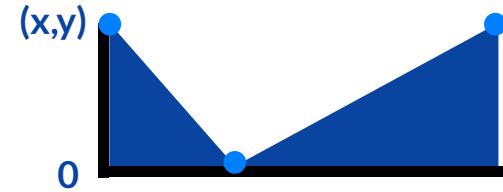
# 3. Produção do gráfico
## Cria o gráfico
p <- data %>%
  ggplot2::ggplot() +
  #### Insere o par de áreas
  ggplot2::geom_area(ggplot2::aes(x = pct, y = num_order,
                                    fill = categ),
                     orientation = "y", size = 4, color = bgcolor,
                     position = ggplot2::position_stack(reverse =
TRUE))
) +
  #### Insere o efeito de "rasgo" à esquerda
  ggplot2::geom_ribbon(ggplot2::aes(xmin = x, xmax = 100, y = y),
                       fill = bgcolor, data = lft_area
)

```



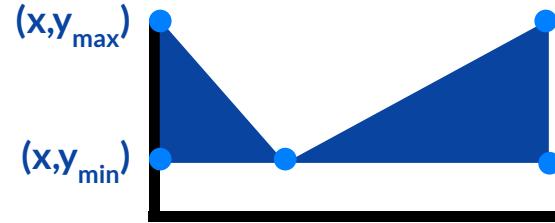
geom_area

área definida entre o eixo x e valores de y (ou entre o eixo y e valores de x)



geom_ribbon

área definida entre valores máximo e mínimo de y (ou máximo e mínimo de x)



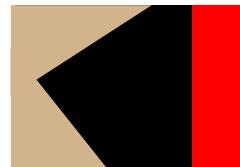
```

# 3. Produção do gráfico
## Cria o gráfico
p <- data %>%
  ggplot2::ggplot() +
  #### Insere o par de áreas
  ggplot2::geom_area(ggplot2::aes(x = pct, y = num_order,
                                    fill = categ),
                     orientation = "y", size = 4, color = bgcolor,
                     position = ggplot2::position_stack(reverse =
TRUE))
) +
  #### Insere o efeito de "rasgo" à esquerda
  ggplot2::geom_ribbon(ggplot2::aes(xmin = x, xmax = 100, y = y),
                       fill = bgcolor, data = lft_area
)

```



`geom_area() +
geom_ribbon()`



Ordem das camadas

A ordem como as camadas (geometrias, escalas, etc) são chamadas no gráfico afeta a visualização. A ordem de “prioridade” na exibição é inversa a como as camadas são chamadas

`geom_ribbon() +
geom_area()`



```

#### Insere o nome das categorias
ggplot2::geom_text(ggplot2::aes(x = x, y = y, label = label),
                   color = "white", size = 15,
                   family = "Teko", data = categ_names
) +


#### Insere títulos e mensagens
ggplot2::labs(title = title, subtitle = sectitle, caption =
message, x = NULL, y = NULL) +

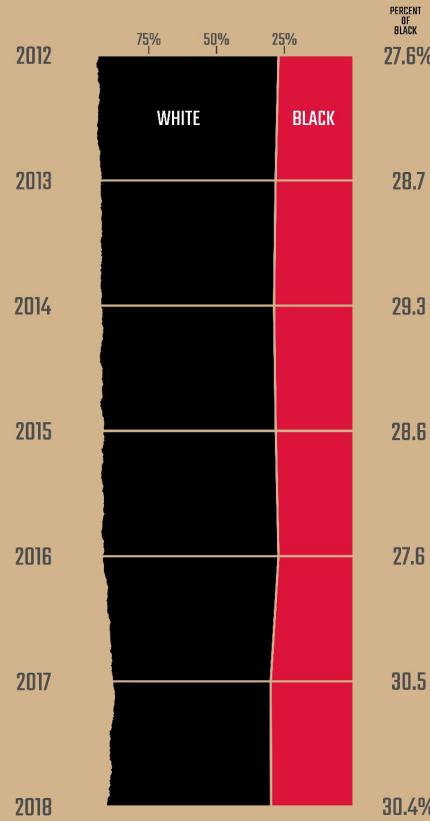

#### Reverte as escalas dos eixos e controla elementos
# das
ggplot2::scale_x_reverse(
  expand = ggplot2::expansion(0, 10), breaks = seq(25, 75, 25),
  label = scales::label_percent(scale = 1), position = "top"
) +
ggplot2::scale_y_reverse(
  expand = ggplot2::expansion(0, 0.01),
  breaks = unique(data$num_order),
  labels = unique(data$order),
  sec.axis = ggplot2::dup_axis(
    name = NULL,
    breaks = unique(data$num_order),
    labels = highlight
)
) +


#### Aplica cores definidas para as áreas
ggplot2::scale_fill_manual(
  values = c("#dc143c", "black"),
  guide = "none"
) +

```

PARTICIPATION IN MANAGERIAL POSITIONS BY RACE IN BRAZIL.

INSPIRED BY: W.E.B. DU BOIS | DATA FROM: IBGE | GRAPHIC BY: ICARO BERNARDES



IN THE SERIES, USUALLY WHITES OCCUPY SLIGHTLY LESS GENERAL WORK POSITIONS.
HOWEVER WHITES OCCUPY WAY MORE MANAGERIAL POSITIONS THAN BLACKS

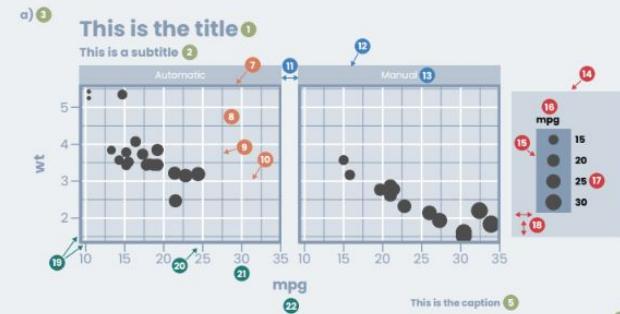
Customiza elementos do gráfico

```
ggplot2::theme(
  text = ggplot2::element_text(family = "Teko"),
  plot.margin = ggplot2::margin(t = 60, r = 400, b = 50, l = 400, unit =
"pt"),
  plot.background = ggplot2::element_rect(fill = bgcolor, color = NA),
  plot.title = ggtext::element_textbox_simple(
    size = 80, halign = 0.5, valign = 0.5, width = 3, lineheight = lnht,
    margin = ggplot2::margin(t = 0, r = 0, b = 20, l = 0, unit = "pt")
  ),
  plot.subtitle = ggtext::element_textbox_simple(
    size = 23, halign = 0.5, valign = 0.5, width = 1.5, lineheight = lnht,
    padding = ggplot2::margin(t = 0, r = 0, b = -10, l = l_marg, unit =
"pt")
  ),
  plot.caption = ggtext::element_textbox_simple(
    size = 45, fill = "#654321", color = bgcolor,
    halign = 0.5, valign = 0.5, width = 3,
    padding = ggplot2::margin(t = 40, r = 0, b = 40, l = 0, unit = "pt"),
    margin = ggplot2::margin(t = 80, r = 0, b = 20, l = 0, unit = "pt")
  ),
  panel.background = ggplot2::element_blank(),
  panel.grid.minor = ggplot2::element_blank(),
  panel.grid.major.x = ggplot2::element_blank(),
  panel.grid.major.y = ggplot2::element_line(color = bgcolor, size = 2),
  panel.on top = TRUE,
  axis.text.x = ggplot2::element_text(size = 35),
  axis.text.y = ggplot2::element_text(size = lb_sz),
  axis.text.y.right = ggplot2::element_text(hjust = 0.5),
  axis.ticks.length.x = ggplot2::unit(15, "pt"),
  axis.ticks.x = ggplot2::element_line(size = 1),
  axis.ticks.y = ggplot2::element_blank()
)
```



ggplot2 Theme System Cheatsheet

Knoppik of the most commonly used theme elements in ggplot2



Plot elements

- 1 plot.title element_rect()
- 2 plot.subtitle element_rect()
- 3 plot.tag element_rect()
- 4 plot.background element_rect()
- 5 plot.caption element_rect()
- 6 plot.margin margin()

Panel elements

- 7 panel.border element_rect()
- 8 panel.background element_rect()
- 9 panel.grid.minor element_rect()
- 10 panel.grid.major element_rect()
- 11 aspect.ratio numeric

Facet elements

- 11 panel.spacing unit()
- 12 strip.background element_rect()
- 13 strip.text element_rect()

Legend elements

- 14 legend.background element_rect()
- 15 legend.key element_rect()
- 16 legend.title element_rect()
- 17 legend.text element_rect()
- 18 legend.text.align numeric
- 19 legend.title.align numeric
- 20 legend.title.hjust numeric
- 21 legend.margin margin()
- 22 legend.position character

Axis elements

- 23 axis.line element_line()
- 24 axis.ticks element_line()
- 25 axis.ticks.length unit()
- 26 axis.text element_text()
- 27 axis.title element_text()

Global

- These affect all elements of some type in the plot. Used to define defaults.
- text element_rect()
- line element_rect()
- rect element_rect()
- title element_rect()

Element functions
element_text()
(font.family (font.face (font.colour (font.size (in points) hjust [0..1] (left, right) vjust [0..1] (bottom, top) angle (in degrees) lineheight (as ratio of fontsize) margin margin (t, r, b, l) #remember trouble
element_line()
(line.colour size (width of line) line.type line.width (in pixels) line.name ("bold", "solid", "dashed", "dotted", "dotteddash", "longdash", "twodash") lineend arrow dnarrow specification arrow()
element_rect()
fill colour size (width of border) line.type (of border) (see element_line)
element_blank()
Eliminates element. Doesn't take parameters

Note.
Of those elements that have two components, the way to access is by appending _x or _y at the end.
e.g. axis.line will change only the "y" axis line, item with "x" if nothing is specified (e.g. axis.line), both elements (x and y) will be changed.

Customizar usando theme()

O Data Ink Lab fez uma excelente *cheat sheet* que indica os principais argumentos da função theme():

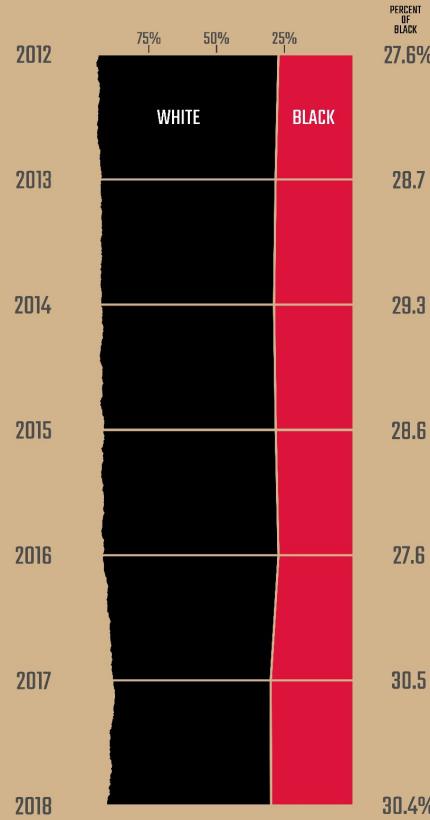
https://github.com/claragranell/ggplot2/blob/main/ggplot_theme_system_cheatsheet.pdf

```
## Salva o gráfico
```

```
file <- paste0(path, "/", filename)
ggplot2::ggsave(file,
  plot = p, dpi = dpi,
  width = 22, height = 28
)
```

PARTICIPATION IN MANAGERIAL POSITIONS BY RACE IN BRAZIL.

INSPIRED BY: W.E.B. DU BOIS | DATA FROM: IBGE | GRAPHIC BY: ICARO BERNARDES



IN THE SERIES, USUALLY WHITES OCCUPY SLIGHTLY LESS GENERAL WORK POSITIONS.
HOWEVER WHITES OCCUPY WAY MORE MANAGERIAL POSITIONS THAN BLACKS

Muito obrigado!

 @IcaroBSC

 <https://github.com/IcaroBernardes>

 icaro@bitanalytics.dev.br

 <https://www.linkedin.com/in/icarobsc>

