aula04

ME315-2S2025

brotto

2025-09-08

Sumário

§ Cat

```
cat(1)
```

```
## 1
```

§ Natureza

Atividade de BD para 09/09/25 – Laboratório Especial.

Tabela 1: Os dois primeiros registros do CD.

YEAR	MONTH	DAY	DAY_OF_WEEK	AIRLINE	FLIGHT_NUMBER
2015	1	1	4	AS	98
2015	1	1	4	AA	2336

TAIL_NUMBER	ORIGIN_AIRPORT	DESTINATION_AIRPORT
N407AS	ANC	SEA
N3KUAA	LAX	PBI

SCHEDULED_DEPARTURE	DEPARTURE_TIME	DEPARTURE_DELAY	TAXI_OUT
0005	2354	-11	21
0010	0002	-8	12

WHEELS_OFF	SCHEDULED_TIME	ELAPSED_TIME	AIR_TIME	DISTANCE
0015	205	194	169	1448
0014	280	279	263	2330

WHEELS_ON	TAXI_IN	SCHEDULED_ARRIVAL	ARRIVAL_TIME	ARRIVAL_DELAY
0404	4	0430	0408	-22
0737	4	0750	0741	-9

DIVERTED	CANCELLED	CANCELLATION_REASON	AIR_SYSTEM_DELAY
0	0	NA	NA
0	0	NA	NA

SECURITY_DELAY	AIRLINE_DELAY	LATE_AIRCRAFT_DELAY	WEATHER_DELAY
NA	NA	NA	NA
NA	NA	NA	NA

§ Conjunto de Dados (CD): `flights.csv.zip`

```
## Amostra do CD os 2 primeiros registros
   a = read_csv("flights.csv.zip", n_max = 2)
   lista_partes <- split.default(a, cut(1:ncol(a), breaks = seq(0, ncol(a), by = 1)))
```

```
## Trata Tabela 1
   u = (lista_partes %>% kable())[1]
   v = substring(u, 14, nchar(u))
   ss2 = "\\caption{Os dois primeiros registros do CD.}"; ss1 = "\\begin{table}"
   paste0(ss1,ss2,v) %>% cat()
```

Laboratório Especial

Benilton Carvalho & Carlos Trucios

§ Parte 1:

Crie uma função que:

a. Receba um valor de `TAIL_NUMBER` (por exemplo, `N431WN`);

b. Produza uma tabela (tidy) com todos os trajetos realizados pela aeronave (ordenadas por data e hora, contendo todas as colunas do arquivo `flights.csv.zip`);

c. Produza um mapa que apresente todo o trajeto voado pela aeronave ao longo de todo o ano; o trajeto deve ser apresentado de maneira linear no tempo (i.e., segue a sequência do tempo, como no exemplo hipotético dado acima);

d. O mapa deve ser decorado com estatísticas do seu interesse (por exemplo, velocidade média do vôo como espessura da linha que conecta os aeroportos envolvidos no trajeto);

Função:

`analisa_aeronave` $= f \circ g : \text{String} \to \mathbb{M}$

$f : \text{String} \to F(\text{String} \times \mathbb{M}; \mathbb{M})$

$g : F(\text{String} \times \mathbb{M}; \mathbb{M}) \to \mathbb{M}$

```
analisa_aeronave = function(tail_number){
  function(arquivo, pos){
    arquivo %>%
      filter(TAIL_NUMBER == tail_number) %>%
      filter(!is.na(ORIGIN_AIRPORT), !is.na(DESTINATION_AIRPORT)) %>%
      group_by(DAY, MONTH, TAIL_NUMBER)
  }
}
```

---

§ Execução

Leitura em blocos:

```
mycols = cols_only(DAY='i', MONTH='i', YEAR='i', AIRLINE='c', TAIL_NUMBER='c',
↪  ORIGIN_AIRPORT='c', DESTINATION_AIRPORT='c')
t0 = Sys.time()
in2 = read_csv_chunked("flights.csv.zip",
                       callback = DataFrameCallback$new(analisa_aeronave(tail_number =
                       ↪  "N651DL")),
```

```
                          chunk_size = 1e5,
                          col_types = mycols)
  t1 = Sys.time()
```

$$\Delta t = 24.3804819583893\ s$$

```
  in2 %>% head(10) %>% kable(format = "latex") %>% kable_styling(latex_options =
↪  c("hold_position","scale_down"))
```

YEAR	MONTH	DAY	AIRLINE	TAIL_NUMBER	ORIGIN_AIRPORT	DESTINATION_AIRPORT
2015	1	1	DL	N651DL	SEA	MSP
2015	1	1	DL	N651DL	MSP	SEA
2015	1	1	DL	N651DL	SEA	DTW
2015	1	1	DL	N651DL	DTW	MCO
2015	1	2	DL	N651DL	MCO	ATL
2015	1	2	DL	N651DL	ATL	SEA
2015	1	2	DL	N651DL	SEA	ATL
2015	1	3	DL	N651DL	ATL	SFO
2015	1	3	DL	N651DL	SFO	DTW
2015	1	3	DL	N651DL	DTW	LAX

```
  airports = read_csv("airports.csv")
  airports %>% head(10) %>% kable(format = "latex") %>% kable_styling(latex_options =
↪  c("hold_position","scale_down"))
```

IATA_CODE	AIRPORT	CITY	STATE	COUNTRY	LATITUDE	LONGITUDE
ABE	Lehigh Valley International Airport	Allentown	PA	USA	40.65236	-75.44040
ABI	Abilene Regional Airport	Abilene	TX	USA	32.41132	-99.68190
ABQ	Albuquerque International Sunport	Albuquerque	NM	USA	35.04022	-106.60919
ABR	Aberdeen Regional Airport	Aberdeen	SD	USA	45.44906	-98.42183
ABY	Southwest Georgia Regional Airport	Albany	GA	USA	31.53552	-84.19447
ACK	Nantucket Memorial Airport	Nantucket	MA	USA	41.25305	-70.06018
ACT	Waco Regional Airport	Waco	TX	USA	31.61129	-97.23052
ACV	Arcata Airport	Arcata/Eureka	CA	USA	40.97812	-124.10862
ACY	Atlantic City International Airport	Atlantic City	NJ	USA	39.45758	-74.57717
ADK	Adak Airport	Adak	AK	USA	51.87796	-176.64603

```
  airports$COUNTRY %>% unique()
```

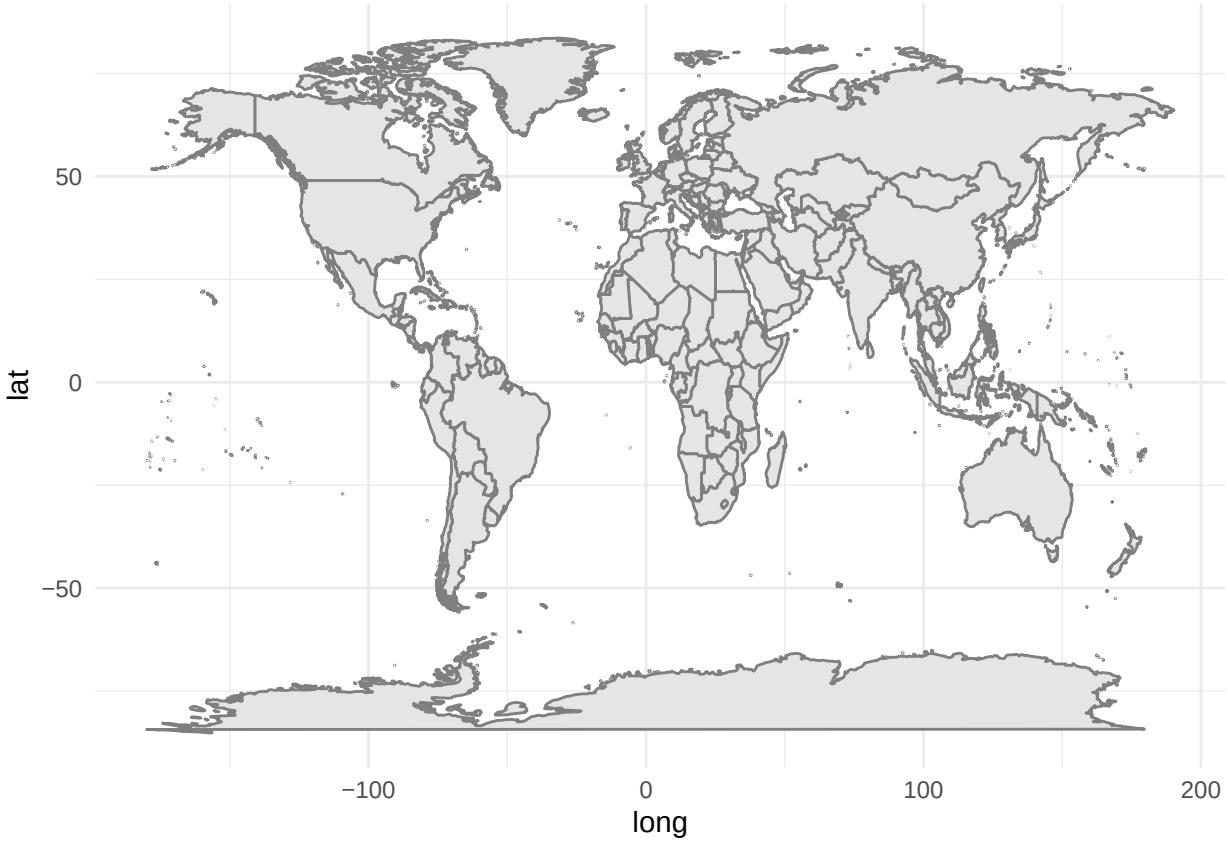
```
## [1] "USA"
```

```
install.packages("farver")
install.packages("ggplot2")
install.packages("maps")
install.packages("mapdata")
```

```
library(farver)
library(ggplot2)
library(maps)
library(mapdata)
```

```
  g1 = ggplot() +
    borders("world", colour = "gray50", fill = "gray90") +
```

```
  theme_minimal()

g1
```



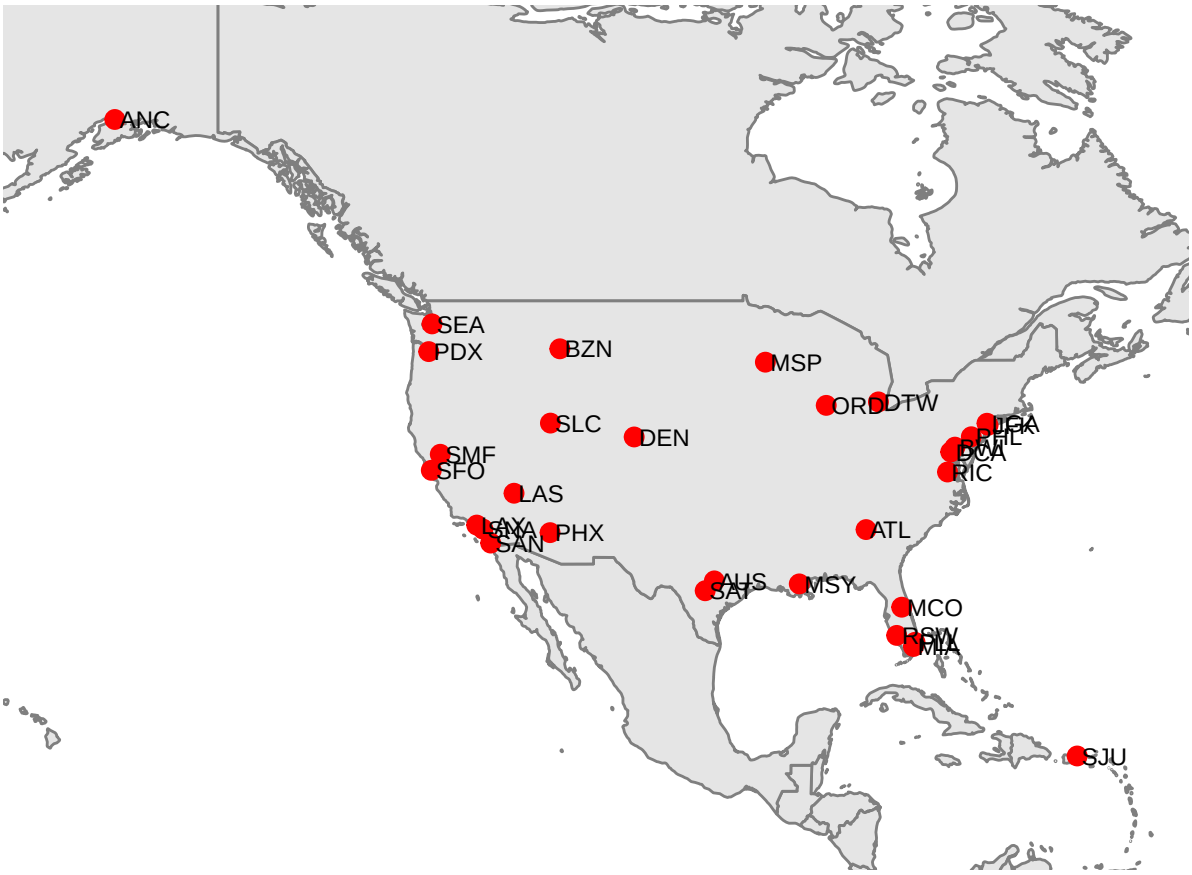

```
airports
```

```
## # A tibble: 322 x 7
##    IATA_CODE AIRPORT                       CITY  STATE COUNTRY LATITUDE LONGITUDE
##    <chr>     <chr>                         <chr> <chr> <chr>      <dbl>     <dbl>
##  1 ABE       Lehigh Valley International~ Alle~ PA    USA         40.7     -75.4
##  2 ABI       Abilene Regional Airport      Abil~ TX    USA         32.4     -99.7
##  3 ABQ       Albuquerque International S~ Albu~ NM    USA         35.0    -107.
##  4 ABR       Aberdeen Regional Airport     Aber~ SD    USA         45.4     -98.4
##  5 ABY       Southwest Georgia Regional ~ Alba~ GA    USA         31.5     -84.2
##  6 ACK       Nantucket Memorial Airport    Nant~ MA    USA         41.3     -70.1
##  7 ACT       Waco Regional Airport         Waco  TX    USA         31.6     -97.2
##  8 ACV       Arcata Airport                Arca~ CA    USA         41.0    -124.
##  9 ACY       Atlantic City International~ Atla~ NJ    USA         39.5     -74.6
## 10 ADK       Adak Airport                  Adak  AK    USA         51.9    -177.
## # i 312 more rows
```

```
a1 = airports[c("IATA_CODE","LATITUDE","LONGITUDE")]
local_airports = c(in2$ORIGIN_AIRPORT, in2$DESTINATION_AIRPORT) %>% unique()
```

```
seus_dados = a1
seus_dados = seus_dados %>% filter(IATA_CODE %in% local_airports)
```

```r
lat_range <- range(seus_dados$LATITUDE) + c(-5, 5)
long_range <- range(seus_dados$LONGITUDE) + c(-5, 5)

ggplot() +
  borders("world", colour = "gray50", fill = "gray90") +
  geom_point(data = seus_dados, aes(x = LONGITUDE, y = LATITUDE),
             color = "red", size = 3) +
  geom_text(data = seus_dados, aes(x = LONGITUDE, y = LATITUDE, label = IATA_CODE),
            hjust = -0.1, vjust = 0.5, size = 3) +
  coord_fixed(xlim = long_range, ylim = lat_range, ratio = 1.3) +
  theme_void()
```




```r
seus_dados = a1
seus_dados = seus_dados %>% filter(IATA_CODE %in% local_airports)

# Esta solução funciona sempre!
plotar_mapa_com_pontos <- function(dados, margem = 0.1) {
  # Calcular limites
  lat_range <- range(dados$LATITUDE, na.rm = TRUE)
  long_range <- range(dados$LONGITUDE, na.rm = TRUE)

  # Adicionar margem
  lat_margem <- diff(lat_range) * margem
  long_margem <- diff(long_range) * margem
```

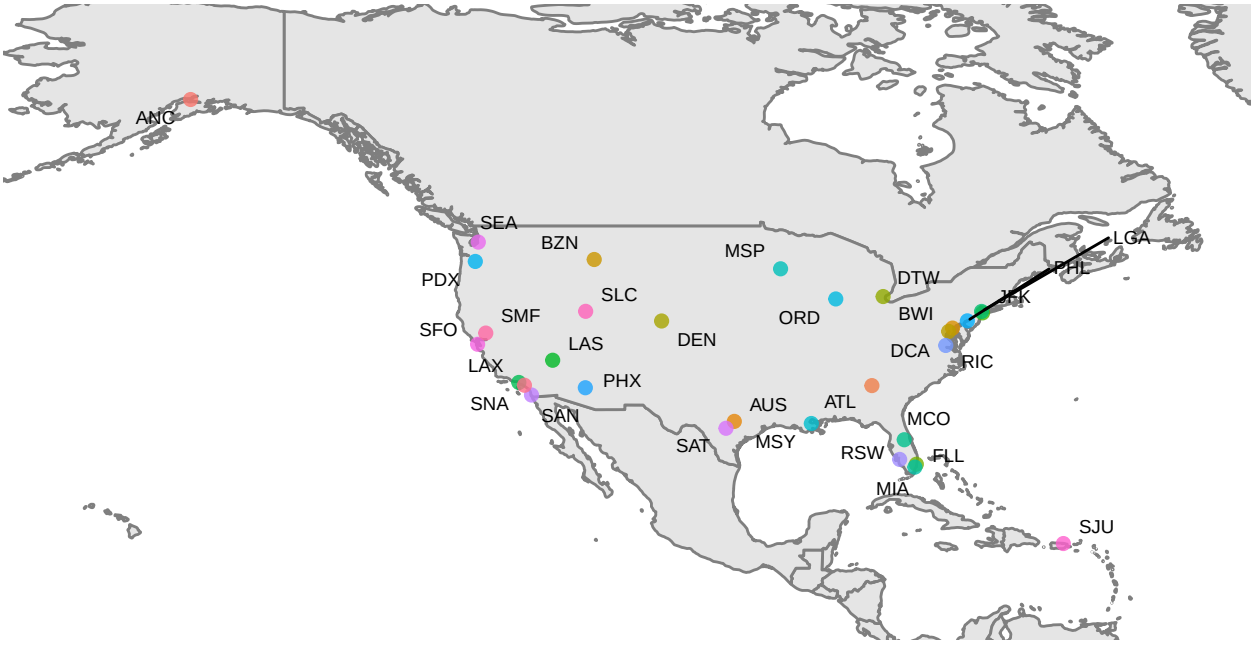
```
  ggplot() +
    borders("world", colour = "gray50", fill = "gray90") +
    geom_point(data = dados, aes(x = LONGITUDE, y = LATITUDE),
               color = "red", size = 1, alpha = 0.6) +
    coord_sf(xlim = c(long_range[1] - long_margem, long_range[2] + long_margem),
             ylim = c(lat_range[1] - lat_margem, lat_range[2] + lat_margem)) +
    theme_void()
}

# Usar a função
plotar_mapa_com_pontos(seus_dados, margem = 0.15)
```

```
install.packages("ggrepel")
```

```
library(ggrepel)
```

```
seus_dados = a1
seus_dados = seus_dados %>% filter(IATA_CODE %in% local_airports)

plotar_pontos_ggrepel <- function(dados, margem = 0.15) {
  # Calcular limites
  lat_range <- range(dados$LATITUDE, na.rm = TRUE)
  long_range <- range(dados$LONGITUDE, na.rm = TRUE)
```

```r
  # Adicionar margem
  lat_margem <- diff(lat_range) * margem
  long_margem <- diff(long_range) * margem

  ggplot() +
    borders("world", colour = "gray50", fill = "gray90") +

    geom_point(data = dados, aes(x = LONGITUDE, y = LATITUDE, color = IATA_CODE),
               size = 2, alpha = 0.8) +

    # ggrepel evita sobreposição de labels
    geom_text_repel(data = dados, aes(x = LONGITUDE, y = LATITUDE, label = IATA_CODE),
                    size = 2.5, color = "black",
                    box.padding = 0.3, max.overlaps = 20) +

    coord_sf(xlim = c(long_range[1] - long_margem, long_range[2] + long_margem),
             ylim = c(lat_range[1] - lat_margem, lat_range[2] + lat_margem)) +
    theme_void() +
    theme(legend.position = "none") # Remove a legenda de cores
}

plotar_pontos_ggrepel(seus_dados)
```




```r
seus_dados = a1
seus_dados = seus_dados %>% filter(IATA_CODE %in% local_airports)
```

```r
search = function(string){
  function(x){
    seus_dados[[which(seus_dados$IATA_CODE == x),string]]
  }
}
```

```r
# Exemplo de estrutura
voos <- data.frame(
  decolagem_lat = unlist(lapply(in2$ORIGIN_AIRPORT,search("LATITUDE"))),
  decolagem_long = unlist(lapply(in2$ORIGIN_AIRPORT,search("LONGITUDE"))),
  aterrissagem_lat = unlist(lapply(in2$DESTINATION_AIRPORT,search("LATITUDE"))),
  aterrissagem_long = unlist(lapply(in2$DESTINATION_AIRPORT,search("LONGITUDE"))),
  tail_number = in2$TAIL_NUMBER
)
```

```r
plotar_rotas_curvas <- function(dados_voos, margem = 0.2) {
  todas_lats <- c(dados_voos$decolagem_lat, dados_voos$aterrissagem_lat)
  todas_longs <- c(dados_voos$decolagem_long, dados_voos$aterrissagem_long)

  lat_range <- range(todas_lats, na.rm = TRUE)
  long_range <- range(todas_longs, na.rm = TRUE)

  lat_margem <- diff(lat_range) * margem
  long_margem <- diff(long_range) * margem

  ggplot() +
    borders("world", colour = "gray50", fill = "gray90") +

    # Curvas para rotas
    geom_curve(data = dados_voos,
               aes(x = decolagem_long, y = decolagem_lat,
                   xend = aterrissagem_long, yend = aterrissagem_lat,
                   color = tail_number),
               curvature = 0.2, # Curvatura da linha
               arrow = arrow(length = unit(0.15, "cm")),
               linewidth = 0.8, alpha = 0.1) +

    # Pontos
    geom_point(data = dados_voos,
               aes(x = decolagem_long, y = decolagem_lat),
               color = "darkred", size = 2) +

    coord_sf(xlim = c(long_range[1] - long_margem, long_range[2] + long_margem),
             ylim = c(lat_range[1] - lat_margem, lat_range[2] + lat_margem)) +
    theme_void() +
    ggtitle("Trajetória da Nave: N651DL") +
    theme(legend.position = "none")
}

plotar_rotas_curvas(voos)
```

Trajetória da Nave: N651DL