Untitled

Luca Misi 204326 , Felipe Basilio 238550

```r
analisa_aeronave <- function(tail_number, arquivo = "flights.csv", arquivo_aeroportos = "air
  library(tidyverse)
  library(leaflet)
  library(lubridate)
  library(readr)
  library(scales)

  # a. Ler dados do arquivo flights.csv filtrando pelo tail_number
  cat("Lendo dados para a aeronave:", tail_number, "\n")

  # Verificar se os arquivos existem
  if (!file.exists(arquivo)) {
    stop("Arquivo de voos não encontrado: ", arquivo)
  }
  if (!file.exists(arquivo_aeroportos)) {
    stop("Arquivo de aeroportos não encontrado: ", arquivo_aeroportos)
  }

  # Ler dados dos voos
  dados_aeronave <- read_csv(arquivo,
                             col_types = cols(
                               .default = col_character(),
                               YEAR = col_integer(),
                               MONTH = col_integer(),
                               DAY = col_integer(),
                               DEPARTURE_TIME = col_integer(),
                               DEPARTURE_DELAY = col_double(),
                               ARRIVAL_TIME = col_integer(),
                               ARRIVAL_DELAY = col_double(),
                               AIR_TIME = col_double(),
                               DISTANCE = col_double(),
                               WHEELS_ON = col_integer(),
```

```
                              WHEELS_OFF = col_integer(),
                              TAXI_IN = col_double(),
                              TAXI_OUT = col_double()
                            )) %>%
  filter(TAIL_NUMBER == tail_number) %>%
  arrange(YEAR, MONTH, DAY, DEPARTURE_TIME) %>%
  mutate(
    DATE = make_date(YEAR, MONTH, DAY),
    DEPARTURE_DATETIME = make_datetime(YEAR, MONTH, DAY,
                                       DEPARTURE_TIME %/% 100,
                                       DEPARTURE_TIME %% 100)
  )

if (nrow(dados_aeronave) == 0) {
  stop("Nenhum dado encontrado para a aeronave: ", tail_number)
}

cat("Encontrados", nrow(dados_aeronave), "voos para a aeronave", tail_number, "\n")

# b. Produzir tabela tidy com todos os trajetos
tabela_trajetos <- dados_aeronave %>%
  select(TAIL_NUMBER, AIRLINE, ORIGIN_AIRPORT, DESTINATION_AIRPORT,
         DATE, DEPARTURE_DATETIME, DEPARTURE_DELAY, ARRIVAL_DELAY,
         AIR_TIME, DISTANCE, everything())

# Carregar dados dos aeroportos
aeroportos <- read_csv(arquivo_aeroportos, show_col_types = FALSE) %>%
  select(IATA_CODE, LATITUDE, LONGITUDE) %>%
  filter(!is.na(LATITUDE), !is.na(LONGITUDE),
         between(LATITUDE, -90, 90), between(LONGITUDE, -180, 180))

# Adicionar coordenadas de origem e destino
dados_mapa <- tabela_trajetos %>%
  left_join(aeroportos, by = c("ORIGIN_AIRPORT" = "IATA_CODE")) %>%
  rename(ORIGIN_LAT = LATITUDE, ORIGIN_LON = LONGITUDE) %>%
  left_join(aeroportos, by = c("DESTINATION_AIRPORT" = "IATA_CODE")) %>%
  rename(DEST_LAT = LATITUDE, DEST_LON = LONGITUDE) %>%
  filter(!is.na(ORIGIN_LAT), !is.na(ORIGIN_LON),
         !is.na(DEST_LAT), !is.na(DEST_LON),
         between(ORIGIN_LAT, -90, 90), between(ORIGIN_LON, -180, 180),
         between(DEST_LAT, -90, 90), between(DEST_LON, -180, 180))
```

```r
if (nrow(dados_mapa) == 0) {
  warning("Nenhum dado válido com coordenadas para criar o mapa")
  return(list(
    tabela = tabela_trajetos,
    mapa = NULL
  ))
}

# Calcular velocidade média
dados_mapa <- dados_mapa %>%
  mutate(
    AVG_SPEED = ifelse(AIR_TIME > 0, (DISTANCE / AIR_TIME) * 60, NA)
  )

# Função para rescale personalizado (linhas MAIS FINAS)
my_rescale <- function(x, to = c(0, 1), from = range(x, na.rm = TRUE)) {
  (x - from[1]) / (from[2] - from[1]) * (to[2] - to[1]) + to[1]
}

# Calcular espessura da linha - LINHAS MAIS FINAS (0.8 a 3 pixels)
if (all(is.na(dados_mapa$AVG_SPEED))) {
  dados_mapa$LINE_WIDTH <- 1.5  # Espessura padrão mais fina
} else {
  dados_mapa <- dados_mapa %>%
    mutate(
      # LINHAS MAIS FINAS: de 0.8 a 3 pixels (ao invés de 2-8)
      LINE_WIDTH = ifelse(is.na(AVG_SPEED), 1.5,
                          my_rescale(AVG_SPEED, to = c(0.8, 3)))
    )
}

# Criar tooltip informativo
dados_mapa <- dados_mapa %>%
  mutate(
    TOOLTIP = paste0("Voo: ", ORIGIN_AIRPORT, " → ", DESTINATION_AIRPORT, "<br>",
                     "Data: ", DATE, "<br>",
                     "Distância: ", round(DISTANCE, 1), " miles<br>",
                     "Tempo de voo: ", round(AIR_TIME, 1), " min<br>",
                     "Velocidade média: ", round(AVG_SPEED, 1), " mph<br>",
                     "Espessura da linha: ", round(LINE_WIDTH, 1), " px")
  )
```

```r
# Criar mapa leaflet com linhas mais finas
mapa <- leaflet(dados_mapa) %>%
  addTiles() %>%
  addProviderTiles(providers$CartoDB.Positron)  # Mapa mais claro para melhor visualização

# Adicionar trajetos sequenciais com LINHAS MAIS FINAS
for (i in 1:nrow(dados_mapa)) {
  mapa <- mapa %>%
    addPolylines(
      lng = c(dados_mapa$ORIGIN_LON[i], dados_mapa$DEST_LON[i]),
      lat = c(dados_mapa$ORIGIN_LAT[i], dados_mapa$DEST_LAT[i]),
      weight = dados_mapa$LINE_WIDTH[i],  # LINHAS MAIS FINAS
      color = "blue",
      opacity = 0.6,  # Opacidade ligeiramente reduzida
      popup = dados_mapa$TOOLTIP[i],
      group = "Trajetos"
    )
}

# Adicionar marcadores para aeroportos únicos (também menores)
aeroportos_unicos <- bind_rows(
  dados_mapa %>%
    select(AIRPORT = ORIGIN_AIRPORT, LAT = ORIGIN_LAT, LON = ORIGIN_LON),
  dados_mapa %>%
    select(AIRPORT = DESTINATION_AIRPORT, LAT = DEST_LAT, LON = DEST_LON)
) %>%
  distinct(AIRPORT, .keep_all = TRUE)

# Adicionar marcadores dos aeroportos (também menores)
mapa <- mapa %>%
  addCircleMarkers(
    data = aeroportos_unicos,
    lng = ~LON,
    lat = ~LAT,
    radius = 3,  # Marcadores menores (era 5)
    color = "darkgreen",
    fillColor = "green",
    fillOpacity = 0.7,
    popup = ~paste("Aeroporto:", AIRPORT),
    group = "Aeroportos"
  )
```

```r
  # Adicionar legenda e controles
  mapa <- mapa %>%
    addLegend(
      position = "bottomright",
      colors = c("green", "blue"),
      labels = c("Aeroportos", "Trajetos do Voo"),
      opacity = 0.8
    ) %>%
    addLayersControl(
      overlayGroups = c("Trajetos", "Aeroportos"),
      options = layersControlOptions(collapsed = FALSE)
    ) %>%
    addScaleBar(position = "bottomleft")

  return(list(
    tabela = tabela_trajetos,
    mapa = mapa,
    dados_validos_mapa = nrow(dados_mapa),
    dados_totais = nrow(tabela_trajetos),
    espessura_min = min(dados_mapa$LINE_WIDTH, na.rm = TRUE),
    espessura_max = max(dados_mapa$LINE_WIDTH, na.rm = TRUE)
  ))
}
analisa_aeronave("N431WN")
```

```
Warning: pacote 'lubridate' foi compilado no R versão 4.4.3
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
Warning: pacote 'leaflet' foi compilado no R versão 4.4.3
```

```
Anexando pacote: 'scales'

O seguinte objeto é mascarado por 'package:purrr':

    discard

O seguinte objeto é mascarado por 'package:readr':

    col_factor

Lendo dados para a aeronave: N431WN
Encontrados 1938 voos para a aeronave N431WN


$tabela
# A tibble: 1,938 x 33
   TAIL_NUMBER AIRLINE ORIGIN_AIRPORT DESTINATION_AIRPORT DATE
   <chr>       <chr>   <chr>          <chr>               <date>
 1 N431WN      WN      DCA            HOU                 2015-01-01
 2 N431WN      WN      HOU            BOS                 2015-01-01
 3 N431WN      WN      BOS            BWI                 2015-01-01
 4 N431WN      WN      BWI            MCI                 2015-01-01
 5 N431WN      WN      MCI            LAX                 2015-01-02
 6 N431WN      WN      LAX            LAS                 2015-01-02
 7 N431WN      WN      LAS            ICT                 2015-01-02
 8 N431WN      WN      ICT            DAL                 2015-01-02
 9 N431WN      WN      DAL            MDW                 2015-01-02
10 N431WN      WN      MDW            ATL                 2015-01-02
# i 1,928 more rows
# i 28 more variables: DEPARTURE_DATETIME <dttm>, DEPARTURE_DELAY <dbl>,
#   ARRIVAL_DELAY <dbl>, AIR_TIME <dbl>, DISTANCE <dbl>, YEAR <int>,
#   MONTH <int>, DAY <int>, DAY_OF_WEEK <chr>, FLIGHT_NUMBER <chr>,
#   SCHEDULED_DEPARTURE <chr>, DEPARTURE_TIME <int>, TAXI_OUT <dbl>,
#   WHEELS_OFF <int>, SCHEDULED_TIME <chr>, ELAPSED_TIME <chr>,
#   WHEELS_ON <int>, TAXI_IN <dbl>, SCHEDULED_ARRIVAL <chr>, ...

$mapa

$dados_validos_mapa
[1] 1780

$dados_totais
[1] 1938
```

```
$espessura_min
[1] 0.8

$espessura_max
[1] 3
```