Apresentação de Resultados Experimentais para Processamento de Alto Desempenho em R

Vinícius Garcia Pinto, Lucas Leandro Nesi, Lucas Mello Schnorr

20 de abril de 2022

ERAD RS 2022

Considerações Iniciais

Quem somos nós

Vinícius Garcia Pinto

Professor
FURG
Orientador PPGComp
vinicius.pinto@furg.br

Lucas Leandro Nesi

Doutorando PPGC
UFRGS
llnesi@inf.ufrgs.br

Lucas Mello Schnorr

Professor
UFRGS
Orientador PPGC
schnorr@inf.ufrgs.br

Preparando o ambiente para o minicurso

Acesse `https://exp-hpc.gitlab.io/erad-2022/`

Computador de uso pessoal (caso não tenha o R e Rstudio instalados)

1. Obter e instalar o R e o Rstudio

```sh
sudo apt install r-base libclang-dev libxkbcommon-x11-0 libpq5
    libcurl4-openssl-dev libssl-dev
wget https://download1.rstudio.org/desktop/bionic/amd64/rstudio
    -2022.02.1-461-amd64.deb
sudo dpkg -i rstudio-2022.02.1-461-amd64.deb
```

SH

PC do laboratório

1. abrir o Rstudio

Pacotes R (no Rstudio ou em um console R)

2. tentar executar os comandos abaixo

```R
library("magrittr")
library("tidyr")
library("tibble")
library("readr")
```

```R
library("dplyr")
library("ggplot2")
library("patchwork")
library("plotly")
```

3. em caso de falha, executar:

```R
install.packages(
    c("magrittr", "tidyr", "tibble", "readr", "dplyr", "ggplot2",
      "patchwork", "plotly"))
```

Dados para os exemplos

4. baixar os arquivos exemplo.csv e TOP500history.csv

```sh
wget https://exp-hpc.gitlab.io/erad-2022/exemplo.csv
wget https://raw.githubusercontent.com/dlenski/top500/master/
    TOP500_history.csv
```

SH

ou

```r
download.file("https://exp-hpc.gitlab.io/erad-2022/exemplo.csv")
download.file("https://raw.githubusercontent.com/dlenski/top500/master
    /TOP500_history.csv", destfile = "TOP500_history.csv")
```

R

Dados para os exemplos

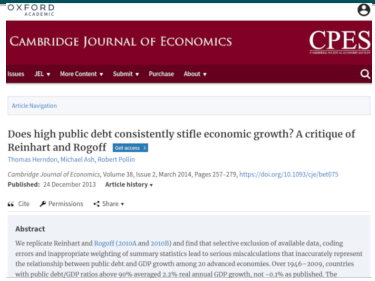
5. leitura dos arquivos exemplo.csv e TOP500history.csv

```
read_csv("exemplo.csv",
         col_types = list(Maquina = col_character(),
                          A       = col_integer(),
                          B       = col_integer(),
                          Tempo   = col_double()),
         progress = FALSE) -> df_ex

library(lubridate)
read_csv("TOP500_history.csv", show_col_types = FALSE) %>%
    filter(!is.na(Region)) %>%
    mutate(Data = ymd(paste(Year, Month, Day))) %>%
    select(Data, Rank, RPeak, Region, Nmax) -> df.simple
```

Introdução

- Processo metodológico para experimentos em PAD
  1. coleta dos dados
  2. análise dos resultados
- Reprodutibilidade da análise dos resultados é aspecto chave do processo científico
  - frequentemente subestimada
  - conduzida com ferramentas inapropriadas (pouca rastreabilidade)
  - diversos relatos de problemas, equívocos ou retratações
    - falhas metodológicas
    - falhas de manipulação e interpretação dos dados



Introdução

- Processo metodológico para experimentos em PAD
  1. coleta dos dados
  2. análise dos resultados
- Reprodutibilidade da análise dos resultados é aspecto chave do processo científico
  - frequentemente subestimada
  - conduzida com ferramentas inapropriadas (pouca rastreabilidade)
  - diversos relatos de problemas, equívocos ou retratações
    - falhas metodológicas
    - falhas de manipulação e interpretação dos dados

Introdução

- Processo metodológico para experimentos em PAD
  1. coleta dos dados
  2. análise dos resultados
- Reprodutibilidade da análise dos resultados é aspecto chave do processo científico
  - frequentemente subestimada
  - conduzida com ferramentas inapropriadas (pouca rastreabilidade)
  - diversos relatos de problemas, equívocos ou retratações
    - falhas metodológicas
    - falhas de manipulação e interpretação dos dados
- Solução é investir em automatização de processos (*scripting*), rastreabilidade, versionamento e apresentação

Organização do minicurso

Objetivos

- Apresentar uma abordagem baseada na linguagem `R` abrangendo desde o processamento dos dados brutos até a representação visual.
- Utilizar ferramentas modernas para permitir a documentação e a reprodutibilidade de todas as transformações e manipulações aplicadas aos dados

Organização

1. Introdução à linguagem R
   - comandos essenciais
   - biblioteca base
2. Processamento de dados brutos

   `readr` `tidyr` `magrittr` `dplyr` `tibble`
3. Visualização

   `ggplot2` `patchwork` `plotly`
4. Prática

Introdução à linguagem R

ERAD RS 2022

A linguagem R

- Linguagem interpretada
- Multi-plataforma
- Código aberto (GPL v2)
- Foco em cálculos estatísticos, gráficos e análise de dados
- Extenso conjunto de pacotes
  - CRAN disponibiliza mais de 19000 pacotes (abril de 2022)
  - Várias maneiras de fazer a "mesma" coisa
- Extensa documentação
- Comunidade ampla e ativa

Comandos básicos

comandos de atribuição `<- e ->`

aritmética básica `-`, `+`, `*`, `/` e `^`

operações relacionais `<`, `>`, `==`, `!=`, `>=` e `<=`

comentários de linha `#`

```r
x <- 100              # x recebe 100
100/3 + 2^2 -> y      # y recebe 37.33333...
z <- "escola regional de alto desempenho"
```

R

Funções

construção `function`

- um nome
- argumentos (opcional)
  - *default values* também são possíveis
- retorno explícito com `return` (opcional)
  - na ausência de `return`, retorna o valor da última linha de código avaliada

```R
aceleracao <- function(tseq, tpar){
    return(tseq/tpar)
}
```

```R
raiz <- function(x, i = 2){
    x^(1/i)
}
```

Pacotes

instalação de pacote(s) `install.packages(` *nome(s) do(s) pacote(s)* `)`[1]
carregamento de um pacote `library(` *nome do pacote* `)`
atualização dos pacotes instalados `update.packages()`
remoção de pacotes `remove.packages(` *nome(s) do(s) pacote(s)* `)`
documentação de pacote (ou função) `?pacote` (`?funcao`)

```r
install.packages("ggplot2")
library(ggplot2)

ggplot(mtcars, aes(x = mpg, y = cyl, color = gear)) +
    geom_point()
```

R

[1]Por padrão, utiliza o CRAN (*Comprehensive R Archive Network* - `https://cran.r-project.org/`) que é o repositório oficial da linguagem R. Alternativamente, também pode ser usado para instalação a partir de arquivos locais.

Tipos básicos

Vetores

- tipo mais simples de objeto `R`
- armazenam valores de um mesmo tipo básico (e.g., `logical`, `integer`, `numeric`, `complex`, `character` e `raw` [R C21])
- podem ser de tamanho 1
- indexados de 1 até N

Matrizes

- vetores com mais de uma dimensão

Lista

- similar a um vetor porém admite valores de diferentes tipos básicos

Tipos básicos

Definição de vetores

```r
d <- c(1, 5, 3)   # d recebe 1 3 5
e <- 2:6          # e recebe 2 3 4 5 6
f <- seq(9, 5)    # f recebe 9 8 7 6 5
```

R

Elementos nomeados

Listas e vetores podem ter seus valores internos nomeados (útil para indexação)

```r
z <- c(prim = 1, seg = 5, terc = 3)
z["seg"] # acessa o elemento de nome 'seg'
z[c("prim", "terc")] # acessa os elementos 'prim' e 'terc'
```

R

Tipos básicos

Data frames

- estruturas mais usadas para armazenamento de dados experimentais
  - casos (linhas)
  - variáveis (colunas)
- implementação
  - uma lista de vetores de mesmo tamanho
  - cada vetor desta lista possui um nome e representa uma variável
  - *Tibbles* são uma implementação moderna dos *Data frames*

```
  Maquina A B Tempo
1    Mac1 0 0    20
2    Mac1 0 1    18
3    Mac1 1 0    16
4    Mac1 1 1    12
5    Mac2 0 0    25
6    Mac2 0 1    20
7    Mac2 1 0    25
8    Mac2 1 1    18
```

```
'data.frame': 8 obs. of  4 variables:
 $ Maquina: chr  "Mac1" "Mac1" "Mac1" "Mac1" ...
 $ A      : int  0 0 1 1 0 0 1 1
 $ B      : int  0 1 0 1 0 1 0 1
 $ Tempo  : num  20 18 16 12 25 20 25 18
```

Aritmética com vetores

- vetores podem ser operandos em expressões aritméticas → operação vetorial

```r
a <- 1:5     # a recebe 1  2  3  4   5
b <- a ^ 3   # b recebe 1  8 27 64 125
c <- a + b   # c recebe 2 10 30 68 130
```

R

$$a = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \end{bmatrix}$$

$$b = a^3 \begin{bmatrix} 1 & 8 & 27 & 64 & 125 \end{bmatrix}$$

$$c = a + b \begin{bmatrix} 2 & 10 & 30 & 68 & 130 \end{bmatrix}$$

- atenção com vetores de tamanhos diferentes
  - resultado terá o tamanho do maior vetor
  - operandos menores são *reciclados*
  - *warnings* são emitidos apenas quando o tamanho do maior não é múltiplo do menor

Scripts R

- muito útil para disponibilização do código da análise dos resultados e geração dos gráficos
  - facilita reexecuções
  - rastreamento integral → melhor reprodutibilidade
- exemplo: `https://exp-hpc.gitlab.io/erad-2022/companion.R`

```r
#!/usr/bin/Rscript
args <- commandArgs(trailingOnly = TRUE)

print(paste("primeiro argumento:", args[1]))
print(paste("segundo argumento:",  args[2]))
print(paste("terceiro argumento:", args[3]))
```

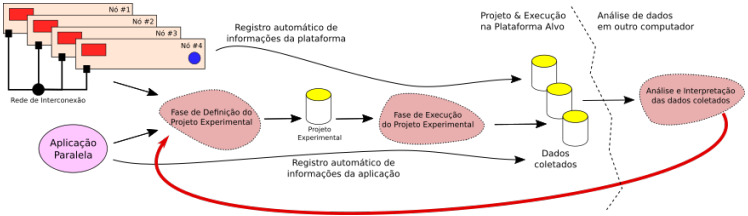
R

```sh
./meuscript.R arg1 arg2 arg3
```

SH

Manipulação de resultados em texto puro

Processo experimental



Análise dos resultados

- resultados em formato texto (e.g., tempo, memória, energia, ...)
- pré-processamento, filtragem, transformação, interpretação e apresentação

Leitura

- Resultados de experimentos de PAD usualmente são do tipo configuração vs métrica
  - exemplos de configurações: máquinas, rede, número de núcleos, versão sw
  - exemplos de métricas: tempo, memória, banda, energia
- *Data frame* (observações vs variáveis)
  - `read_csv` do pacote `readr` faz a leitura de dados "retangulares"

```
Maquina,A,B,Tempo
Mac1,0,0,20.0
Mac1,0,1,18.0
Mac1,1,0,16.0
Mac1,1,1,12.0
Mac2,0,0,25.0
Mac2,0,1,20.0
Mac2,1,0,25.0
Mac2,1,1,18.0
```

```r
read_csv("exemplo.csv",
         col_types = list(Maquina = col_character(),
                          A       = col_integer(),
                          B       = col_integer(),
                          Tempo   = col_double()),
         progress = FALSE) -> df_ex
```

Encadeamento

- operador %>% (*pipe*) do pacote `magrittr`
  - estilo convencional `fun(x, y)`
  - com encadeamento `x %>% fun(y)`
- melhora a legibilidade do código → as operações aparecem na ordem que acontecem de fato
- versão R 4.1.0 inclui um comando de *pipe* nativo |>

Transformações

seleção de colunas comando `select` (pacote `dplyr`)

```r
df_ex %>% select(Maquina, Tempo)
```

R

```
# A tibble: 8 × 2
  Maquina Tempo
  <chr>   <dbl>
1 Mac1       20
2 Mac1       18
3 Mac1       16
4 Mac1       12
5 Mac2       25
6 Mac2       20
7 Mac2       25
8 Mac2       18
```

ordenação das linhas comando arrange (pacote dplyr)

```r
df_ex %>% arrange(A, B)
```

```
# A tibble: 8 × 4
  Maquina     A     B Tempo
  <chr>   <int> <int> <dbl>
1 Mac1        0     0    20
2 Mac2        0     0    25
3 Mac1        0     1    18
4 Mac2        0     1    20
5 Mac1        1     0    16
6 Mac2        1     0    25
7 Mac1        1     1    12
8 Mac2        1     1    18
```

filtragem de linhas comando `filter` (pacote `dplyr`)

```R
df_ex %>% filter(Maquina %in% c("Mac1", "Mac2") & Tempo > 18)
```

```
# A tibble: 4 × 4
  Maquina     A     B Tempo
  <chr>   <int> <int> <dbl>
1 Mac1        0     0    20
2 Mac2        0     0    25
3 Mac2        0     1    20
4 Mac2        1     0    25
```

fatiamento por linhas comando `slice` (pacote `dplyr`)

```r
df_ex %>% slice(2:4)
```

R

```
# A tibble: 3 × 4
  Maquina      A     B Tempo
  <chr>    <int> <int> <dbl>
1 Mac1         0     1    18
2 Mac1         1     0    16
3 Mac1         1     1    12
```

Pivotagem

comando `pivot_longer` do pacote `tidyr` transforma para uma organização mais longa (mais linhas) enquanto `pivot_wider` para mais larga (mais colunas)

```R
df_wid <- tibble(Nome = c("Alpha", "Beta"),
                 Caso_A = c(1, 4),
                 Caso_B = c(2, 5),
                 Caso_C = c(3, 6))
df_wid
```

```
# A tibble: 2 × 4
  Nome  Caso_A Caso_B Caso_C
  <chr>  <dbl>  <dbl>  <dbl>
1 Alpha      1      2      3
2 Beta       4      5      6
```

Pivotagem

comando pivot_longer do pacote tidyr transforma para uma organização mais longa (mais linhas) enquanto pivot_wider para mais larga (mais colunas)

```r
df_wid <- tibble(Nome = c("Alpha", "Beta"),
                 Caso_A = c(1, 4),
                 Caso_B = c(2, 5),
                 Caso_C = c(3, 6))
df_wid
```

R

```
# A tibble: 2 × 4
  Nome  Caso_A Caso_B Caso_C
  <chr>  <dbl>  <dbl>  <dbl>
1 Alpha      1      2      3
2 Beta       4      5      6
```

```r
pivot_longer(df_wid,
             c(Caso_A, Caso_B, Caso_C),
             names_to = "Variavel_1",
             values_to = "Valor") -> df_lon
df_lon
```

R

```
# A tibble: 6 × 3
  Nome  Variavel_1 Valor
  <chr> <chr>      <dbl>
1 Alpha Caso_A         1
2 Alpha Caso_B         2
3 Alpha Caso_C         3
4 Beta  Caso_A         4
5 Beta  Caso_B         5
6 Beta  Caso_C         6
```

Pivotagem

```r
df_longer %>% pivot_wider(names_from = `Variavel_1`,
                          values_from = Valor)
```

R

```
# A tibble: 2 × 4
  Nome  Caso_A Caso_B Caso_C
  <chr>  <dbl>  <dbl>  <dbl>
1 Alpha      1      2      3
2 Beta       4      5      6
```

Derivações

criação de nova coluna comando `mutate` (pacote `dplyr`) utilizando outras colunas ou valores externos

```r
df_ex %>% mutate(D = A + B)
```

R

```
# A tibble: 8 × 5
  Maquina     A     B Tempo     D
  <chr>   <int> <int> <dbl> <int>
1 Mac1        0     0    20     0
2 Mac1        0     1    18     1
3 Mac1        1     0    16     1
4 Mac1        1     1    12     2
5 Mac2        0     0    25     0
6 Mac2        0     1    20     1
7 Mac2        1     0    25     1
8 Mac2        1     1    18     2
```

sumarizar uma coluna comando `summarize` (pacote `dplyr`)

```R
df_ex %>% summarize(media_t = mean(Tempo))
```

```
# A tibble: 1 × 1
  media_t
    <dbl>
1    19.2
```

Combinando Tabelas

combinação de conjuntos de dados distintos comando `inner_join` (e outros comandos `*_join` providos pelo pacote `dplyr`)

```
# A tibble: 8 × 4
  Maquina     A     B Tempo
  <chr>   <int> <int> <dbl>
1 Mac1        0     0    20
2 Mac1        0     1    18
3 Mac1        1     0    16
4 Mac1        1     1    12
5 Mac2        0     0    25
6 Mac2        0     1    20
7 Mac2        1     0    25
8 Mac2        1     1    18
```

```
# A tibble: 2 × 2
  Nome  Cores
  <chr> <dbl>
1 Mac1     20
2 Mac2     24
```

Combinando Tabelas

combinação de conjuntos de dados distintos comando `inner_join` (e outros comandos `*_join` providos pelo pacote dplyr)

```
# A tibble: 8 × 4
  Maquina     A     B Tempo
  <chr>   <int> <int> <dbl>
1 Mac1        0     0    20
2 Mac1        0     1    18
3 Mac1        1     0    16
4 Mac1        1     1    12
5 Mac2        0     0    25
6 Mac2        0     1    20
7 Mac2        1     0    25
8 Mac2        1     1    18
```

```
# A tibble: 2 × 2
  Nome  Cores
  <chr> <dbl>
1 Mac1     20
2 Mac2     24
```

```r
inner_join(df_ex, maquinas,
           by = c("Maquina" = "Nome"))
```

R

```
# A tibble: 8 × 5
  Maquina     A     B Tempo Cores
  <chr>   <int> <int> <dbl> <dbl>
1 Mac1        0     0    20    20
2 Mac1        0     1    18    20
3 Mac1        1     0    16    20
4 Mac1        1     1    12    20
5 Mac2        0     0    25    24
6 Mac2        0     1    20    24
7 Mac2        1     0    25    24
8 Mac2        1     1    18    24
```

Grupos

agrupamento de observações que compartilham uma mesma chave comando `group_by` (pacote `dplyr`)

```R
df_ex %>% group_by(Maquina) %>% slice(1)
```

```
# A tibble: 2 × 4
# Groups:   Maquina [2]
  Maquina     A     B Tempo
  <chr>   <int> <int> <dbl>
1 Mac1        0     0    20
2 Mac2        0     0    25
```

Construção de gráficos

Gramática de Gráficos

- O gráfico é descrito como uma combinação de dados, mapeamento e elementos geométricos

Pacote `ggplot2`

```R
ggplot(data = mtcars,
       mapping = aes(x = mpg, y = cyl, color = gear)) +
    geom_point()
```

Gramática de Gráficos

- O gráfico é descrito como uma combinação de dados, mapeamento e elementos geométricos

Pacote `ggplot2`

```R
ggplot(data = mtcars,
       mapping = aes(x = mpg, y = cyl, color = gear)) +
    geom_point()
```

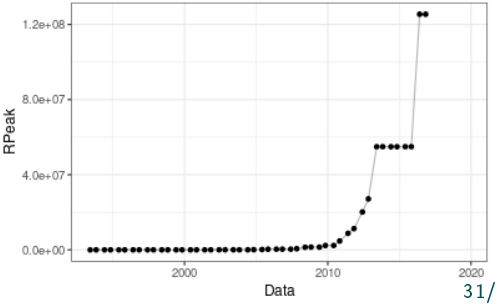
- construção incremental com uso do operador +

```R
ggplot(data = mtcars,
       mapping = aes(x = mpg, y = cyl, color = gear)) +
    geom_point() + geom_line() +
    theme_light() + xlim(c(0, NA)) + ylim(c(0, NA))
```

Gráficos Básicos

- comando ggplot do pacote ggplot2
  - conjunto de dados (e.g., *Data frame*)
  - estética básica com aes() (e.g., eixo x, eixo y, cores, formas, tamanho)
- comandos geom_* definem o estilo do gráfico
  - ex: geom_point para gráfico de pontos
  - cada estilo possui um conjunto de estéticas obrigatório

```R
df.simple %>%
    filter(Rank == 1) %>%
    ggplot(aes(x = Data, y = RPeak)) +
    geom_line(alpha = 0.3) +
    geom_point() +
    theme_bw(base_size = 12)
```



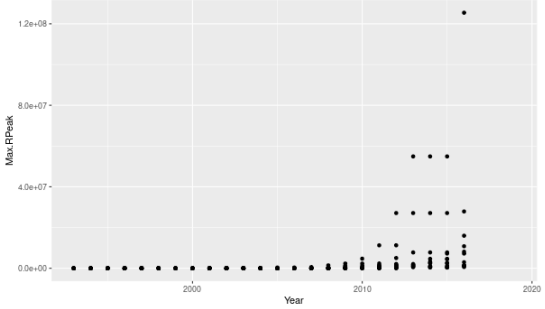
Comparações com Facetas

- Como adicionar novas variáveis no gráfico?
  - Adição de elementos estéticos (aes) como cores, tamanhos e formas
  - Uso de facetas
- Facetas são úteis para visualização de variáveis que representam categorias, isto é, variáveis discretas
  - dividem um gráfico em vários sub-gráficos similares menores
  - `facet_wrap` cria sub-gráficos por linhas (uma variável)
  - `facet_grid` cria sub-gráficos por linhas e colunas (duas variáveis)

Comparações com Facetas

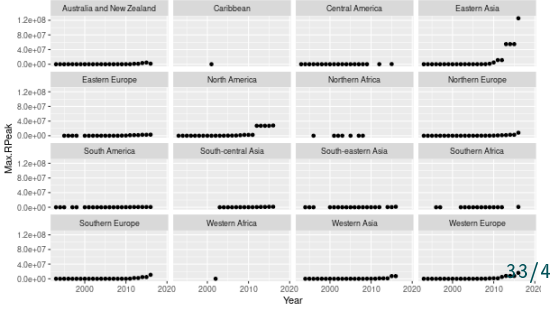
```r
df.simple %>%
  mutate(Year = year(Data)) %>%
  group_by(Year, Region) %>%
  summarize(Max.RPeak = max(RPeak)) %>%
  ggplot(aes(x = Year, y = Max.RPeak)) +
  geom_point()
```

R



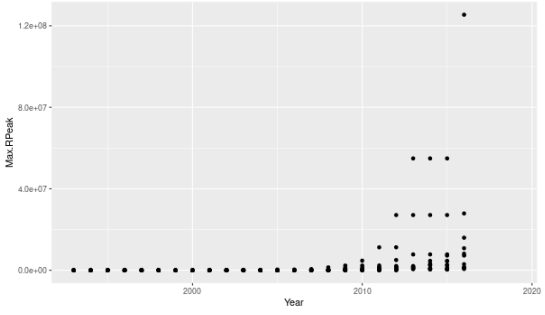

```r
df.simple %>%
  mutate(Year = year(Data)) %>%
  group_by(Year, Region) %>%
  summarize(Max.RPeak = max(RPeak)) %>%
  ggplot(aes(x = Year, y = Max.RPeak)) +
  geom_point() +
  facet_wrap(~Region)
```

R

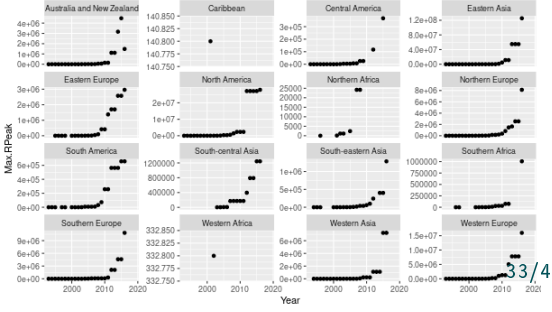


Comparações com Facetas

```R
df.simple %>%
  mutate(Year = year(Data)) %>%
  group_by(Year, Region) %>%
  summarize(Max.RPeak = max(RPeak)) %>%
  ggplot(aes(x = Year, y = Max.RPeak)) +
  geom_point()
```




```R
df.simple %>%
  mutate(Year = year(Data)) %>%
  group_by(Year, Region) %>%
  summarize(Max.RPeak = max(RPeak)) %>%
  ggplot(aes(x = Year, y = Max.RPeak)) +
  geom_point() +
  facet_wrap(~Region, scales = "free_y")
```



Composição de Gráficos

- Por vezes, gostaríamos de combinar dois gráficos distintos em uma mesma figura
- Pacote `patchwork` permite compor gráficos criados com ggplot
  - operador / (empilhados)
  - operador + (lado a lado)
  - comando `wrap_plots`

```R
df.simple %>%
  mutate(Year = year(Data)) %>%
  group_by(Year, Region) %>%
  summarize(Max.RPeak = max(RPeak)) %>%
  ggplot(aes(x = Year, y = Max.RPeak)) +
  geom_point() -> p1
```

```R
p1 +
  facet_wrap(~Region,
             scales = "free_y") -> p2
```

Composição de Gráficos

- Por vezes, gostaríamos de combinar dois gráficos distintos em uma mesma figura
- Pacote `patchwork` permite compor gráficos criados com ggplot
  - operador / (empilhados)
  - operador + (lado a lado)
  - comando `wrap_plots`

```r
df.simple %>%
  mutate(Year = year(Data)) %>%
  group_by(Year, Region) %>%
  summarize(Max.RPeak = max(RPeak)) %>%
  ggplot(aes(x = Year, y = Max.RPeak)) +
  geom_point() -> p1
```

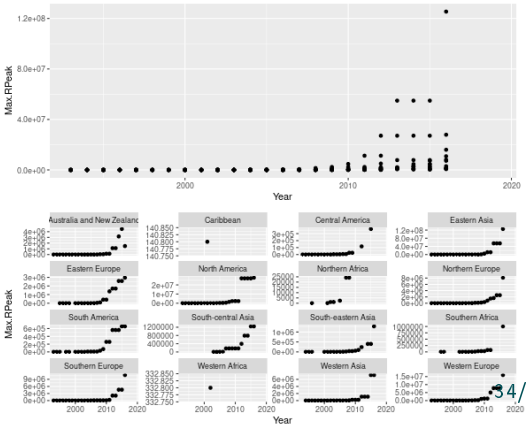
R

```r
p1 +
  facet_wrap(~Region,
             scales = "free_y") -> p2
```

R

```r
p1 / p2
```

R



Gráficos Interativos

- Gráficos interativos permitem ir além da visão geral
  - navegar pelos valores
  - aproximar regiões do gráfico
  - desativar categorias
- Comando `ggplotly` do pacote `plotly` traduz um gráfico ggplot2 para plotly

```R
df.simple %>%
    filter(Rank == 1) %>%
    ggplot(aes(x = Data, y = RPeak)) +l
    geom_point() -> p3
ggplotly(p3)
```

- Exemplos:
  - http://exp-hpc.gitlab.io/erad-2022/plotly-1.html
  - http://exp-hpc.gitlab.io/erad-2022/plotly-2.html

Prática

Exercício 1

Contabilize o número de sistemas de cada região em cada edição da lista do TOP500, ordenando a partir da lista mais recente.

```
# A tibble: 610 × 3
# Groups:   Data, Region [610]
   Data       Region                    Total
   <date>     <chr>                     <int>
 1 2019-11-01 Australia and New Zealand     3
 2 2019-11-01 Eastern Asia                263
 3 2019-11-01 Eastern Europe                5
 4 2019-11-01 North America               126
 5 2019-11-01 Northern Europe              30
 6 2019-11-01 South America                 3
 7 2019-11-01 South-central Asia            2
 8 2019-11-01 South-eastern Asia            4
 9 2019-11-01 Southern Europe               7
10 2019-11-01 Western Asia                  5
# ... with 600 more rows
```

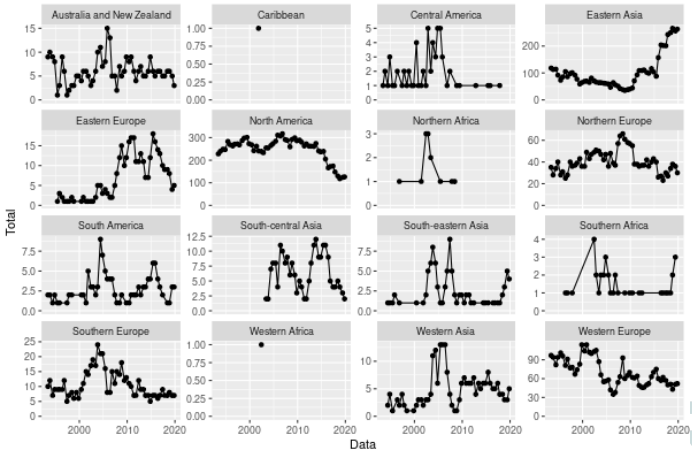
Exercício 2

Contabilize em qual edição da lista cada uma das regiões teve sua maior representatividade, ordenando por ordem decrescente do total de sistemas.

```
# A tibble: 22 × 3
# Groups:   Region [16]
   Data       Region                   Total
   <date>     <chr>                    <int>
 1 2006-11-01 North America              317
 2 2018-11-01 Eastern Asia               266
 3 1999-11-01 Western Europe             114
 4 2000-11-01 Western Europe             114
 5 2008-06-01 Northern Europe             66
 6 2003-11-01 Southern Europe             24
 7 2015-06-01 Eastern Europe              18
 8 2005-11-01 Australia and New Zealand   15
 9 2005-06-01 Western Asia                13
10 2005-11-01 Western Asia                13
# ... with 12 more rows
```

Exercício 3

Faça um gráfico mostrando a quantidade de sistemas classificados ao longo do tempo com facetas por região do globo.



Considerações Finais

Conclusão

- Abordagem em R para apresentação de resultados experimentais para PAD
  - Processamento dos dados brutos até a construção dos gráficos
  - Documentável e rastreável
- Assuntos correlatos em outras edições da ERAD
  - "Are you root? Experimentos Reprodutíveis em Espaço de Usuário" [DPNS21]
  - "Boas Práticas para Experimentos Computacionais de Alto Desempenho" [PNS20]
  - "Boas Práticas para Experimentos" [SP19]
- Entrem em contato para sugestões, colaborações, etc

Obrigado por participar e pela atenção!

Perguntas?

Contato

vinicius.pinto@furg.br
lucas.nesi@inf.ufrgs.br
schnorr@inf.ufrgs.br

Material do minicurso (slides, código, capítulo)

`https://exp-hpc.gitlab.io/erad-2022/`

ERAD RS 2022

Referências

Jessica Imlau Dagostini, Vinícius Garcia Pinto, Lucas Leandro Nesi, and Lucas Mello Schnorr.
Are you root? Experimentos Reprodutíveis em Espaço de Usuário.
In Andrea Charão and Matheus Serpa, editors, *Minicursos da XXI Escola Regional de Alto Desempenho da Região Sul*, chapter 3, pages 70–87. Sociedade Brasileira de Computação - SBC, Porto Alegre, 04 2021.

Garrett Grolemund.
Hands-On Programming with R.
O'Reilly Media, 1 edition, 2014.

Referências ii

Vinícius Garcia Pinto, Lucas Leandro Nesi, and Lucas Mello Schnorr.
Boas Práticas para Experimentos Computacionais de Alto Desempenho.
In André Rauber du Bois and Márcio Bastos Castro, editors, *Minicursos da XX Escola Regional de Alto Desempenho da Região Sul*, chapter 1, pages 1–19. Sociedade Brasileira de Computação - SBC, 04 2020.

R Core Team.
***R Language Definition*, 2021.**

Lucas Mello Schnorr and Vinícius Garcia Pinto.
Boas Práticas para Experimentos.
In *Anais da XIX Escola Regional de Alto Desempenho da Região Sul*, pages 45–64. Sociedade Brasileira de Computação - SBC, apr 2019.

Hadley Wickham and Garrett Grolemund.
R for Data Science.
O'Reilly Media, 1 edition, 2017.