

# Pacotes em R

Diego Moraes

# Pacotes

- Agrupam funções com um tema em comum
  - Manipulação de arquivos
  - Manipulação de grafos
- Disponíveis no CRAN, Bioconductor e Github
  - Dependendo do repositório o modo de instalação pode variar

# Criação de pacotes

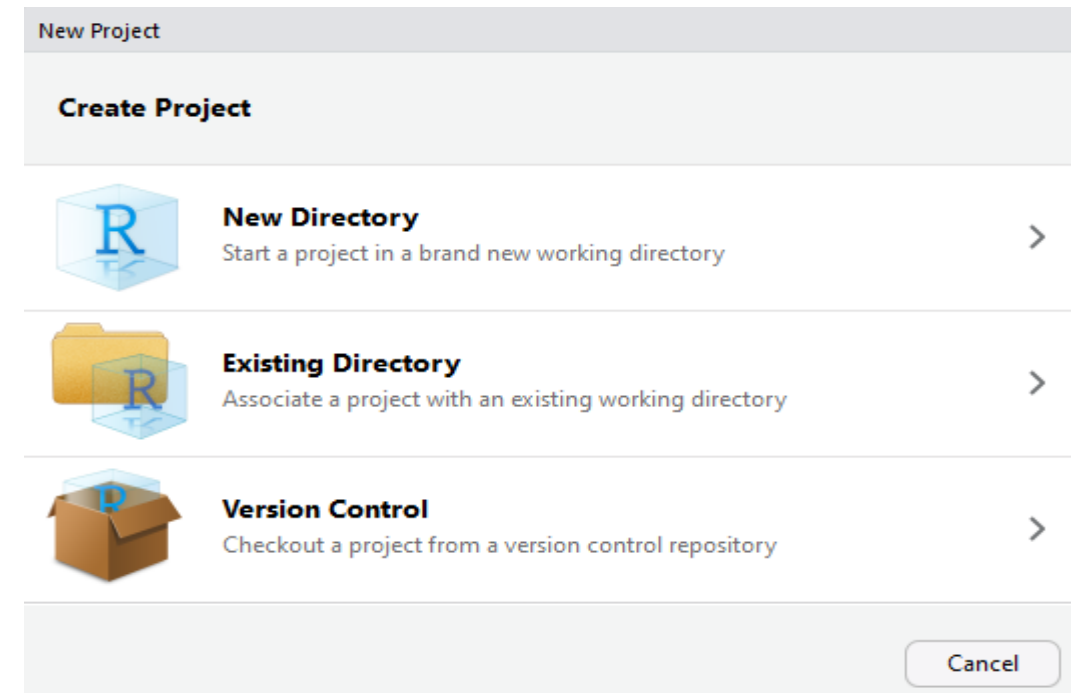
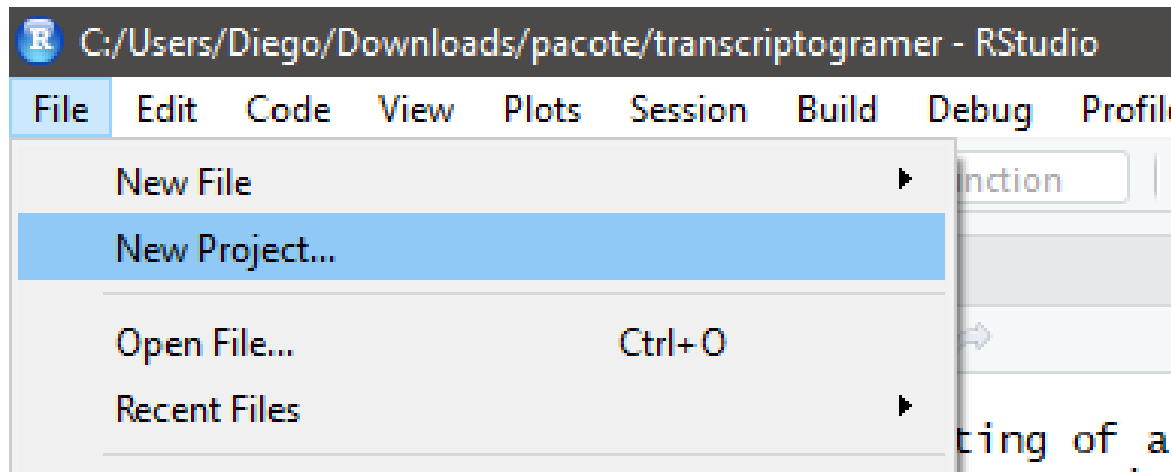
- Ferramentas necessárias para a criação de um pacote:
  - Pacote devtools
  - Pacote roxygen2
  - Rstudio
  - R
- No Windows é necessário instalar também o Rtools
  - <https://cran.r-project.org/bin/windows/Rtools/>
- No Linux algumas outras dependências também são necessárias
  - libssl-dev, libcurl4-openssl-dev e libxml2-dev
  - O nome pode variar dependendo da distribuição utilizada
  - No Fedora os nomes são libcurl-devel, libssl-devel e libxml2-devel

# Preparando o ambiente

- Instale as dependências específicas do sistema operacional
- Instale o R e o Rstudio
- Execute o Rstudio como administrador
- Instale o devtools e o roxygen2 via `install.packages`

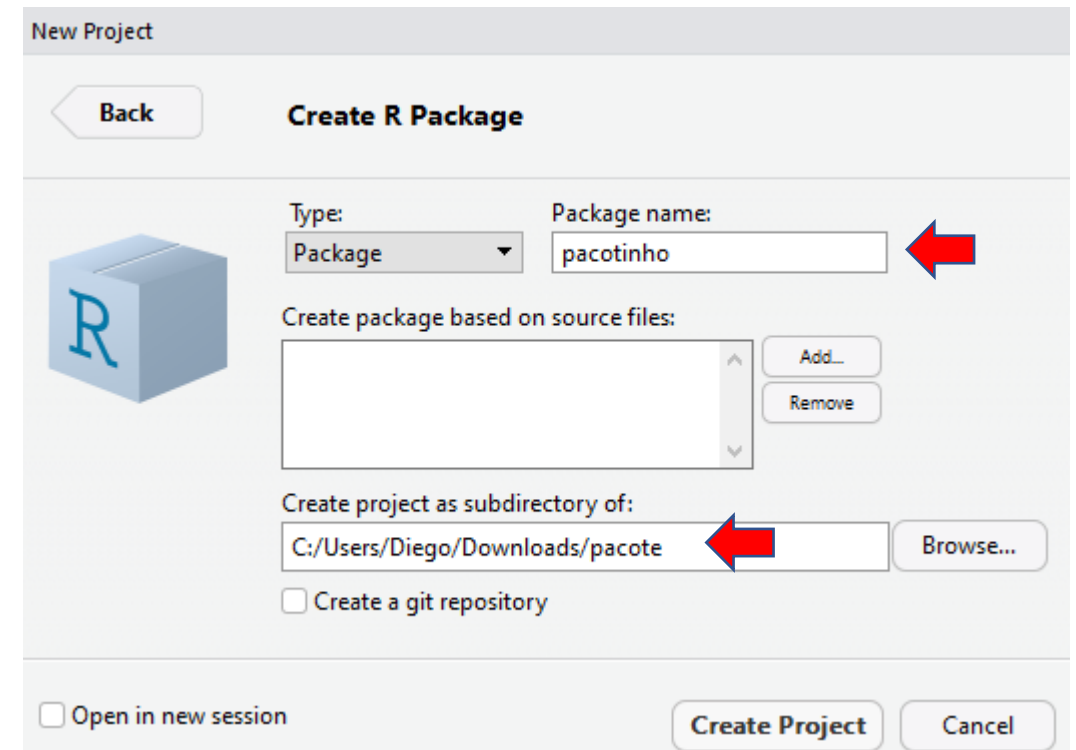
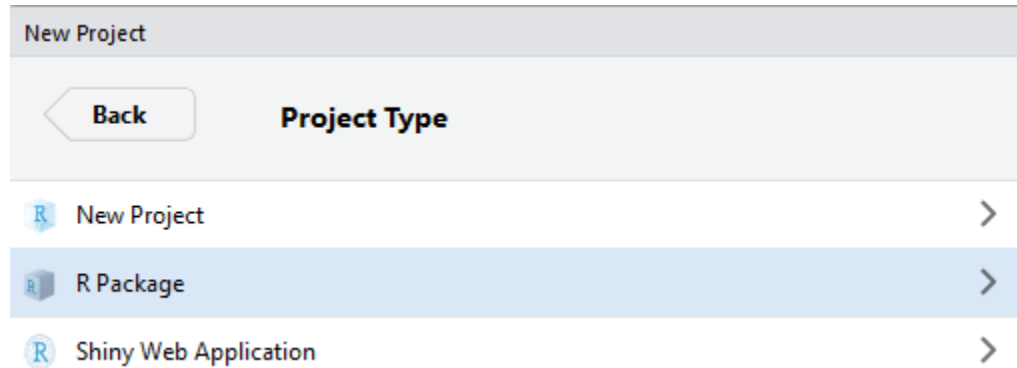
# Criando um pacote

- O pacote é um projeto do Rstudio
  - É possível criar um novo projeto
  - Ou a partir do diretório de um pacote baixado caso ele não tenha um arquivo de extensão .Rproj (caso tenha, é só clicar duas vezes neste arquivo que o projeto será aberto no Rstudio)



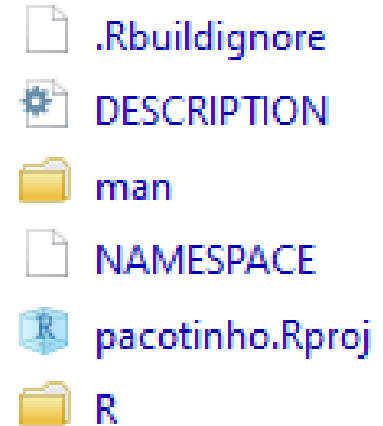
# Criando um pacote

- Defina um nome para o pacote e o local onde ele será criado



# Estrutura de um pacote

- Estrutura mínima
  - DESCRIPTION
  - NAMESPACE
  - man
  - R











# Estrutura de um pacote

- **DESCRIPTION:** Um arquivo de texto a ser editado com as informações do pacote, tais como versão, pacotes requeridos, autores e descrição. As informações deste arquivo são usadas para verificar o conteúdo do pacote e para instalá-lo. A regra de versionamento sugere que a versão do pacote seja composta por `X.Y.Z`, sendo o `Z` incrementado a cada alteração, o `Y` sendo incrementado a cada lançamento ou adição de funcionalidades, e o `X` sendo incrementado em casos raros de mudanças bruscas ou grandes alterações.
- **NAMESPACE:** Este arquivo descreve tudo que é importado e exportado pelo pacote, e `nunca deve ser editado manualmente`. Este arquivo é essencial e deve ser consistente com o que o pacote utiliza e o que se deseja disponibilizar para os usuários. Para descrever isto é necessário o uso de `comentários roxygen`, interpretados pelo pacote `roxygen2` do CRAN para gerar a documentação das funções.
- **man:** Um diretório que armazena a documentação das funções. Seu conteúdo é gerado e atualizado pela função `document()` do pacote `devtools`, que gera arquivos `.Rd` a partir de `comentários roxygen`. A função `document()` também atualiza o conteúdo do arquivo `NAMESPACE`.
- **R:** Todos os arquivos `.R`, contendo códigos referentes às funcionalidades, devem ser armazenados neste diretório.



# Estrutura de um pacote

- Estrutura opcional e o que armazena
  - data
    - Arquivos .Rdata contendo dados
  - inst
    - CITATION, NEWS, doc e script com testes unitários
  - tests
    - Script que executa os testes unitários
  - vignettes
    - Arquivo rmarkdown usado para gerar a vinheta

 R
 data
 inst
 man
 tests
 vignettes
 DESCRIPTION
 NAMESPACE

# Exemplo de pacote

- <https://github.com/arthurvinx/transcriptograder>
- Para instalar um pacote a partir do github
  - `devtools::install_github("user/repositorio")`
  - `devtools::install_github("arthurvinx/transcriptograder")`
  - Note que é necessário instalar as dependências primeiro
    - Você pode criar um código utilizando a função `require` para instalar dependências necessárias e disponibilizá-lo num arquivo README.md no github
    - Lembre-se que pacotes do bioconductor não são instalados com `install.packages()`

```
1 # tenta carregar o pacote
2 require(ggplot2)
3
4 # instala o pacote caso ele não esteja instalado
5 pacote <- "ggplot2"
6 if(!require(pacote, character.only = TRUE)){
7   install.packages(pacote)
8 }
```

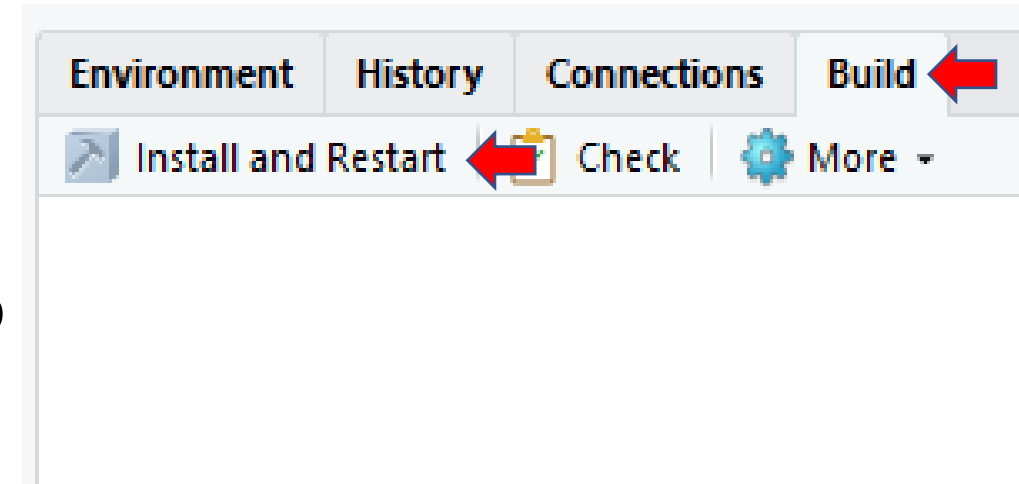
# Prática

- Delete o script hello.R e a documentação hello.Rd
- Crie um script soma.R e escreva o seguinte código

```
1 #' Soma
2 #'
3 #' Soma dois numeros
4 #'
5 #' @param x Um numero inteiro
6 #'
7 #' @param y Um numero inteiro
8 #'
9 #' @return A soma de \code{x} e \code{y}
10 #'
11 #' @example
12 #' soma(2, 3)
13 #'
14 #' @export
15
16 soma <- function(x, y){
17   return(x+y)
18 }
```

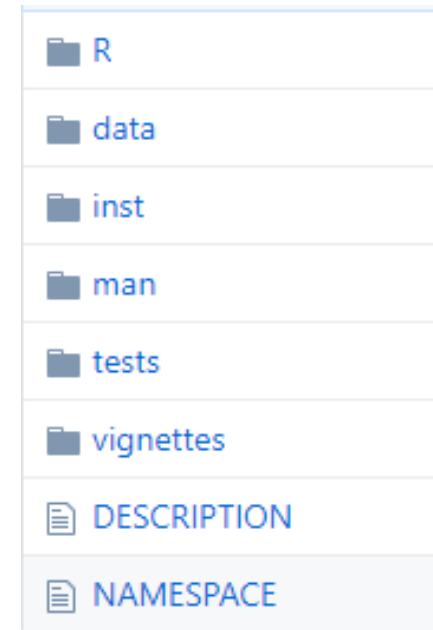
# Prática

- No console digite
  - `devtools::document()`
- Esta função varrerá o conteúdo do projeto e transformará os comentários do roxygen em documentação
- Instale o pacote
- Carregue o pacote
  - `library(pacotinho)`
- Teste a função e consulte a documentação



# Disponibilizando dados

- A pasta data é utilizada para armazenar dados
- Basta salvar uma variável como um arquivo .Rdata e colocar na pasta data
  - `save(GSE9988, file = "GSE9988.RData", compress = "xz")`
- Escreva a documentação do dado num arquivo .R
  - Arquivos .R ficam dentro da pasta R
- O arquivo DESCRIPTION deve possuir a linha `LazyData = true` para que o dado seja carregado junto com o pacote



# Disponibilizando dados

- Exemplo da documentação do data.frame GSE9988

```
1 #' Dataset containing expression values
2 #'
3 #' Expression values, obtained by microarray, of 3 cases and 3 controls
4 #' referring to the Gene Expression Omnibus accession number GSE9988.
5 #' The data.frame has 6 columns, each one contains expression values of a
6 #' sample, the first 3 columns are case samples, and the last 3 are control
7 #' samples. Each row contains expression values obtained by the probe mentioned
8 #' in its respective rowname. The expression values were normalized using the
9 #' \pkg{affy} package and, to reduce the required storage space,
10 #' this data.frame contains only 6 samples (GSM252443, GSM252444,
11 #' GSM252445, GSM252465, GSM252466, GSM252467). The rows of each sample
12 #' are composed only by probes mapped, by the GPL570 dictionary,
13 #' to proteins, from STRINGdb release 11,
14 #' of combined score greater than or equal to 900.
15 #'
16 #' @examples
17 #' GSE9988
18 #'
19 #' @seealso
20 #' \link[transcriptogrator]{GPL570}
21 #'
22 #' @author
23 #' Diego Morais
24 #'
25 #' @source \href{https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE9988}{GSE9988}
26
27 "GSE9988"
```

# Boas práticas

- Não carregue suas dependências utilizando library
  - Chame as funções necessários usando o namespace da dependência dentro do seu código
    - pacote::função(args)
- O repositório pode exigir detalhes adicionais e disponibilizar uma bateria de testes
  - <http://bioconductor.org/developers/package-guidelines/#correctness>
  - <http://bioconductor.org/developers/package-guidelines/#rcode>
- Exporte apenas as funções que o usuário precise executar

# Resumo

- Para criar um pacote você precisa
  - Instalar as dependências necessárias
  - Conhecer e utilizar os diretórios necessários
  - Utilizar roxygen para escrever a documentação
    - <https://cran.r-project.org/web/packages/roxygen2/vignettes/rd.html>
  - Escrever o código do seu pacote
- Detalhes opcionais
  - Disponibilizar dados de exemplo
  - Utilizar rmarkdown para escrever uma vinheta
  - Escrever testes unitários



# Exercício

- O dado abaixo possui 332 arquivos .csv
  - <https://d396qusza40orc.cloudfront.net/rprog%2Fdata%2Fspecdata.zip>
- Estes arquivos possuem informações sobre partículas presentes na atmosfera, como sulfato e nitrato (medidos em microgramas por metro cúbico)
- Cada arquivo corresponde a um local diferente dos EUA, sendo cada local monitorado por um sensor identificado por um número
- Crie um arquivo .RData contendo a informação destes 332 arquivos
  - Utilize um laço, a função `read.csv()`, `rbind()` e a função `save()`

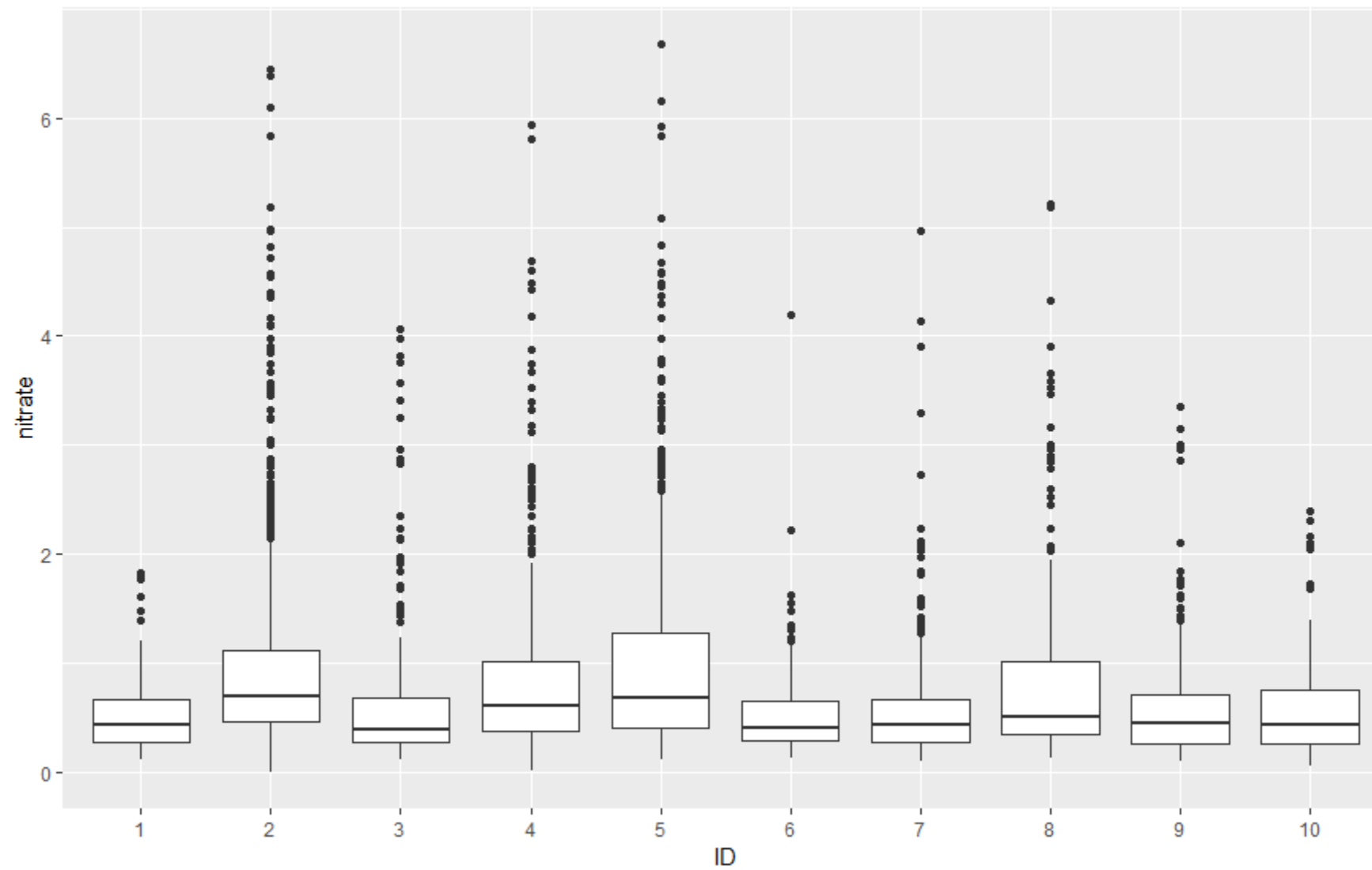
# Exercício

- Crie um pacote e coloque o arquivo `.RData` na pasta `data`
- Escreva a documentação do dado
- Escreva uma função que calcula a média, de sulfato ou nitrato, de um dado número de sensores
  - Uma chamada como `pollutantMean(data, "nitrate", 1:10)` deve retornar `0.7976266`
  - Os argumentos são respectivamente o `data.frame`, o nome da coluna desejada, e os identificadores dos sensores desejados
  - Escreva também a documentação da função

# Exercício

- Escreva uma função que gera um boxplot com os níveis de nitrato de um dado número de sensores
  - Utilize o ggplot2 como dependência
  - Uma chamada como `nitrateBoxplot(data, 1:10)` deve retornar a imagem do próximo slide
  - Os argumentos são respectivamente o `data.frame`, e os identificadores dos sensores desejados
  - Escreva também a documentação da função

# Exercício



# Exercício

- Gere a documentação do pacote
- Comprima seu projeto num arquivo .zip e envie pelo SIGAA