

XIII Semana de Engenharia Química UFBA

Introdução à Exploração e Visualização de Dados em R aplicada à Engenharia Química

Profa Carolina Paraíba e Prof Gilberto Sassi

Departamento de Estatística
Instituto de Matemática e Estatística
Universidade Federal da Bahia

Outubro de 2026

Preparando o ambiente

- Você precisa de um computador para acompanhar as aulas.
- Usaremos nas aulas: RStudio Cloud.
- No seu dia-a-dia, recomendamos instalar o R com versão pelo menos 4.1: cran.r-project.org.
- **IDE** recomendada: *RStudio* e
- Neste curso, usaremos o *framework tidyverse*:
 - Instale o framework a partir do repositório CRAN:
`install.packages("tidyverse")`
- Outras linguagens interessantes: `python` e `julia`.
 - `python`: linguagem interpretada de propósito geral, contemporânea do R, simples e fácil de aprender.
 - `julia`: linguagem interpretada para análise de dados, lançada em 2012, promete simplicidade e velocidade.

A linguagem R: uma introdução

O precursor do R: S.

- R é uma linguagem derivada do S.
- S foi desenvolvido em `fortran` por **John Chambers** em 1976 no **Bell Labs**.
- S foi desenvolvido para ser um ambiente de análise estatística.
- Filosofia do S: permitir que usuários possam analisar dados usando Estatística com pouco conhecimento de programação.

História do R

- Em 1991, **Ross Ihaka** e **Robert Gentleman** criaram o R na **Nova Zelândia**.
- Em 1996, **Ross** e **Robert** liberam o R sob a licença “GNU General License”, o que tornou o R um software livre.
- Em 1997, **The Core Group** é criado para melhorar e controlar o código fonte do R.

Porque usar o R

- Constante melhoramento e atualização.
- Portabilidade (disponível em praticamente todos os sistemas operacionais).
- Grande comunidade de desenvolvedores que adicionam novas capacidades ao R através de pacotes.
- Produz gráficos de maneira relativamente simples.
- Interatividade.
- Grande comunidade de usuários (especialmente útil para resolução de problemas).

Onde estudar fora da aula?

Livros

- **Nível *cheguei agora aqui*:** zen do R.
- **Nível Iniciante:** R Tutorial na W3Schools.
- **Nível Iniciante:** Hands-On Programming with R.
- **Nível Intermediário:** R for Data Science.
- **Nível Avançado:** Advanced R.

Em pt-br

- curso-r: material.curso-r.com.

O que você pode fazer quando estiver em apuros?

- consultar a documentação do R:

```
help(mean)  
?mean
```

- Peça ajuda a um programador mais experiente.
- Consulte o pt.stackoverflow.com.
- Use ferramentas de busca como o google e duckduckgo.com.

```
log("G")
```

- Na ferramenta de busca, pesquise por `Error in log("G"): non-numeric argument to mathematical function`

- **Tipo de dados:** `character` (character), número real (`double`), número inteiro (`integer`), número complexo (`complex`) e lógico (`logical`).
- **Estrutura de dados:** `atomic vector` (a estrutura de dados mais básica no R), `matrix`, `array`, `list` e `data.frame` (`tibble` no `tidyverse`).
- **Estrutura de dados Homogênea:** `vector`, `matrix` e `array`.
- **Estrutura de dados Heterôgenea:** `list` e `data.frame` (`tibble` no `tidyverse`).

Valores especiais no R

Valores especiais	Descrição	Função para identificar
NA (Not Available)	Valor faltante.	<code>is.na()</code>
NaN (Not a Number)	Resultado do cálculo indefinido.	<code>is.nan()</code>
Inf (Infinito)	Valor que excede o valor máximo que sua máquina aguenta.	<code>is.inf()</code>
NULL (Nulo)	Valor indefinido de expressões e funções (diferente de NaN e NA)	<code>is.null()</code>

O operador pipe |>

O valor resultante da expressão do lado esquerdo vira primeiro argumento da função do lado direito.

Principal vantagem: simplifica a leitura e a documentação de funções compostas.

```
f(x, y)
```

é exatamente a mesma coisa que executar

```
x |> f(y)
```

```
log(sqrt(sum(x^2)))
```

é exatamente a mesma coisa que executar

```
x^2 |> sum() |> sqrt() |> log()
```

Parênteses 1: guia de estilo no R

- O nome de um objeto precisa ter um *significado*.
- O nome deve indicar e deixar claro o que este objeto é ou faz: qualquer pessoa precisa entender o que este objeto é ou faz.

Parênteses 1: guia de estilo no R

- Use a convenção do R:
 - Use apenas letras minúsculas, números e *underscore* (comece sempre com letras minúsculas).
 - Nomes de objetos precisam ser substantivos e precisam descrever o que este objeto é ou faz (seja conciso, direto e significativo).
 - Evite ao máximo os nomes que já são usados (*buit-in*) no R.
 - Coloque espaço depois da vírgula.
 - Não coloque espaço antes nem depois de parênteses. Exceção: coloque um espaço () antes e depois de `if`, `for` ou `while`, e coloque um espaço depois de () .
 - Coloque espaço entre operadores básicos: `+`, `-`, `*`, `==` e outros. Exceção: `^`.
- Para mais detalhes, consulte: guia de estilo do `tidyverse`.

Parênteses 2: estrutura de diretórios

- Mantenha uma estrutura (organização) consistente de diretórios em seus projetos.
- Sugestão de estrutura:
 - dados: diretório para armazenar seus conjuntos de dados.
 - brutos: dados brutos.
 - processados: dados processados.
 - codigos: código fonte do seu projeto.
 - figuras: figuras criadas no seu projeto.
 - resultados: outros arquivos que não são figuras.
 - antigo: arquivos da versão anterior do projeto.
 - notas: notas de reuniões e afins.
 - relatorio (ou artigo): documento final de seu projeto.
 - 'referencias': livros, artigos e qualquer coisa que são referências em seu projeto.
- Para mais detalhes, consulte esse guia do curso-r: diretórios e .Rproj.

Leitura de arquivos no formato `xlsx` ou `xls`

- **Pacote:** `readxl` do `tidyverse` (instale com o comando `install.packages('readxl')`)
- Parâmetros das funções `read_xls` (para ler arquivos `.xls`) e `read_xlsx` (para ler arquivos `.xlsx`):
 - `path`: caminho até o arquivo.
 - `sheet`: especifica a planilha do arquivo que será lida.
 - `range`: especifica uma área de uma planilha para leitura. Por exemplo: `B3:E15`.
 - `col_names`: Argumento lógico com valor padrão igual a `TRUE`. Indica se a primeira linha tem o nome das variáveis.
- Para mais detalhes, consulte a documentação oficial do *tidyverse*: documentação de `read_xl`.

Leitura de arquivos no formato `xlsx` ou `xls`

```
library(readxl)
library(tidyverse)

dados <- read_xlsx(
  path = "../dados/brutos/nome.xlsx")
```

Salvar no formato `.xlsx`

- **Pacote:** `writexl` do `tidyverse` (instale com o comando `install.packages('writexl')`)
- Parâmetros da função `write_xlsx` (para ler arquivos `.xlsx`):
 - `path`: caminho até o arquivo.
 - `col_names`: Argumento lógico com valor padrão igual a `TRUE`. Indica se a primeira linha tem o nome das variáveis.
 - `format_headers`: Argumento lógico com valor padrão igual a `TRUE`. Indica que os nomes das colunas no arquivo `.xlsx` estarão centralizados e em negrito.
- Para mais detalhes, consulte a documentação oficial do *writexl*: documentação de `writexl`.

Salvando dados no R

Salvar no formato **.xlsx**

```
library(writexl)

write_xlsx(dados_selecionado,
           path = "dados/processados/nome.xlsx")
```

Conceitos Básicos

- **População:** todos os elementos ou indivíduos alvo do estudo.
- **Amostra:** parte da população.
- **Parâmetro:** característica numérica da população. Usamos letras gregas para denotar parâmetros populacionais.
- **Estatística:** característica numérica da amostra. Em geral, usamos uma estatística para estimar o parâmetro populacional.
- **Variável:** *característica mensurável comum a todos os elementos da população.* Usamos letras maiúsculas do alfabeto latino para representar uma variável e letras minúsculas do alfabeto latino para representar o valor observado da variável em um elemento da amostra.

Exemplo:

- **População:** Todos os residentes da cidade de Salvador com 25 anos ou mais.
- **Amostra:** 5 residentes da cidade de Salvador com 25 anos ou mais *selecionados segundo um plano de amostragem probabilística*.
- **Variável:** salário em R\$ (denotado pela letra X).
- **Parâmetro:** *salário médio* da população de residentes da cidade de Salvador com 25 anos ou mais (denotado pela letra grega μ).
- **Estatística:** *salário médio* da amostra de 20 residentes da cidade de Salvador com 25 anos ou mais.

Exemplo (continuação):

Suponha que foi selecionada uma amostra de $n = 5$ residentes da cidade de Salvador com 25 anos ou mais para os quais foi observada a variável salário em R\$.

Tabela 1: Salário em R\$ de uma amostra de 5 residentes da cidade de Salvador com 25 anos ou mais.

Elemento da amostra	Salário
1	843.95
2	876.98
3	1055.87
4	907.05
5	912.93

Exemplo (continuação):

Para este exemplo, temos que:

- Variável: X : salário em R\$.
- Valores observados de X : x_i : valor observado da variável no i -ésimo elemento da amostra, $i = 1, 2, 3, 4, 5$: 843.95; 876.98; 1055.87; 907.05; 912.93
- Parâmetro: μ : salário médio dos residentes da cidade de Salvador com 25 anos ou mais.
- Estatística: média amostral: $\bar{x} = \frac{x_1 + x_2 + x_3 + x_4 + x_5}{n}$.

Exemplo (continuação):

Média amostral:

$$\begin{aligned}\bar{x} &= \frac{x_1 + x_2 + x_3 + x_4 + x_5}{n} \\ &= \frac{843.95 + 876.98 + 1055.87 + 907.05 + 912.93}{5} \\ &= 919.356.\end{aligned}$$

Classificação de Variáveis

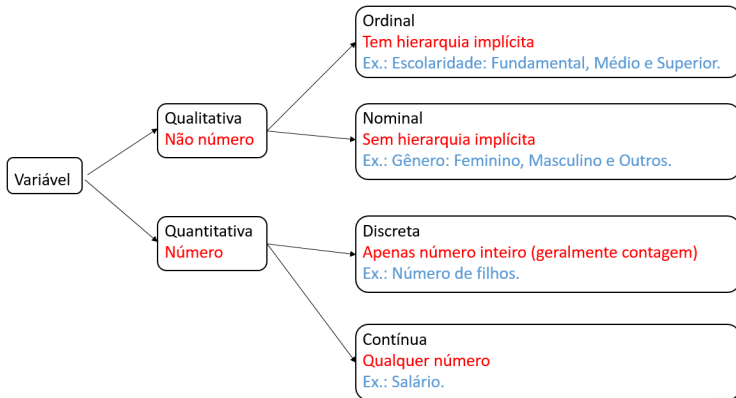


Figura 1: Classificação de variáveis.

Pacotes

```
library(AppliedPredictiveModeling)
library(tidyverse) # Inclui dplyr, tidyr e ggplot2
library(corrplot)  # Para a matriz de correlação
library(naniar)     # Para visualização de NAs
library(knitr)
```

Dados

```
data(ChemicalManufacturingProcess)

df <- ChemicalManufacturingProcess

# Visão geral da estrutura

glimpse(df)
```

Dados

```
# Encurtar nomes

nomes <- names(df)
nomes <- gsub("ManufacturingProcess", "mp_", nomes)
nomes <- gsub("BiologicalMaterial", "bio_", nomes)

names(df) <- nomes

glimpse(df)

df <- df |>
  rename_with(tolower)

glimpse(df)
```

Diagnóstico e Tratamento de NAs

```
gg_miss_var(df) +  
  labs(title = "Nro de Valores Ausentes por Variável",  
        subtitle = "Processo de Manufatura Química",  
        x = "Variável",  
        y = "Número de NAs") +  
  theme_minimal()  
  
tabela_na <- miss_var_summary(df)  
  
tabela_na |>  
  filter(n_miss > 0)
```

Diagnóstico e Tratamento de NAs

Mapa de ausência por observação e variável

```
vis_miss(df) +  
  labs(  
    title = "Mapa de valores ausentes",  
    subtitle = "Cada célula representa uma  
    observação de uma variável") +  
  theme_minimal()
```

Diagnóstico e Tratamento de NAs

```
# Combinações de padrões de ausência
```

```
gg_miss_upset (df)
```

Diagnóstico e Tratamento de NAs

```
# Resumo geral de ausentes
```

```
total_na <- sum(is.na(df))
```

```
percentual_na <- 100 * mean(is.na(as.matrix(df)))
```

```
tibble(  
  total_celulas = length(as.matrix(df)),  
  total_ausentes = total_na,  
  percentual_ausentes = percentual_na) |>  
kable(digits = 2)
```

Diagnóstico e Tratamento de NAs

```
df_clean <- df |>  
  mutate(across(everything(),  
    ~replace_na(., median(., na.rm = TRUE))))
```

Análise Exploratória Univariada

```
# Histograma (com frequências absolutas)

k <- ceiling(1 + log2(nrow(df_clean)))

ggplot(df_clean, aes(x = yield)) +
  geom_histogram(fill = "grey",
                 color = "white",
                 bins = k) +
  geom_vline(aes(xintercept = mean(yield)),
             color = "red",
             linetype = "dashed", size = 1) +
  labs(title = "Distribuição do Rendimento (yield)",
       subtitle = "Linha vermelha: média do processo",
       x = "Rendimento",
       y = "Frequência (Nro de Lotes)") +
  theme_minimal()
```

Análise Exploratória Univariada

```
# Boxplot

ggplot(df_clean, aes(y = yield)) +
  geom_boxplot(fill = "steelblue") +
  labs(
    title = "Boxplot do rendimento",
    y = "Rendimento (yield)",
    x = NULL) +
  theme_minimal() +
  coord_flip()
```

Análise Exploratória - Correlações

Gráfico Q-Q para avaliar aproximação à normalidade

```
ggplot(df_clean, aes(sample = yield)) +  
  stat_qq(alpha = 0.7) +  
  stat_qq_line(color = "red", linewidth = 1) +  
  labs(  
    title = "Gráfico Q-Q do rendimento",  
    x = "Quantis teóricos normais",  
    y = "Quantis amostrais") +  
  theme_minimal()
```

Análise Exploratória - Correlações

```
# Correlações

cor_matrix <- cor(df_clean)

# Correlações com o Yield

cor_yield <- cor_matrix[, "yield"]

# Top 12 variáveis com maior correlação com yield

top_vars <- names(sort(abs(cor_yield),
                        decreasing = TRUE)[1:13])

cor_matrix_top <- cor(df_clean[, top_vars])
```

Análise Exploratória - Correlações

```
# Correlograma

corrplot(cor_matrix_top,
         method = "color",
         type = "lower",
         tl.col = "black",
         tl.cex = 0.8,
         addCoef.col = "black",
         number.cex = 0.7,
         title = "12 Variáveis com Maior Impacto
no Rendimento",
         mar = c(0,0,1,0),
         col = colorRampPalette(
           c("#BB4444", "#FFFFFF", "#4477AA"))(200))
```

Resumo Descritivo

```
variaveis_interesse <- c("yield", "mp_32", "mp_36",  
                          "bio_02", "bio_06")  
  
tabela_descritiva <- df_clean |>  
  select(all_of(variaveis_interesse)) |>  
  # Calcula as métricas ignorando NAs (na.rm = TRUE é essencial aqui)  
  summarise(across(everything(), list(  
    media = ~mean(., na.rm = TRUE),  
    dp = ~sd(., na.rm = TRUE),  
    min = ~min(., na.rm = TRUE),  
    mediana = ~median(., na.rm = TRUE),  
    max = ~max(., na.rm = TRUE))) |>  
  # Reorganiza os dados de "largo" para "longo", separando variável e métrica  
  pivot_longer(everything(),  
    names_to = c("variavel", ".value"),  
    names_pattern = "(.*)_(.*)") |>  
  # Arredonda as casas decimais para facilitar a leitura  
  mutate(across(where(is.numeric), ~round(., 2)))
```

Resumo Descritivo

```
# Visualização com kable

tabela_descritiva |>
  kable(col.names = c("Variável", "Média",
                      "Desvio Padrão", "Mínimo",
                      "Mediana", "Máximo"),
        align = c("l", "c", "c", "c", "c", "c"))
```

Resumo Descritivo - Dados Agrupados

```
# Criando uma categoria baseada na mediana do Rendimento (Yield)
# Se o yield for maior que a mediana, é "Alto", senão "Baixo"

df_agrupado <- df_clean |>
  mutate(performance_lote = ifelse(yield > median(yield, na.rm = TRUE),
                                   "Alto Rendimento",
                                   "Baixo Rendimento"))

# Comparando as médias das variáveis de processo entre bons e maus lotes

df_agrupado |>
  group_by(performance_lote) |>
  summarise(
    media_mp_32 = round(mean(mp_32, na.rm = TRUE), 2),
    media_bio_02 = round(mean(bio_02, na.rm = TRUE), 2))
```

Análise Exploratória - Dados Agrupados

```
# Boxplots para comparar as distribuições entre grupos

df_grupos_long <- df_agrupado |>
  select(performance_lote, all_of(variaveis_interesse))
  pivot_longer(
    cols = all_of(variaveis_interesse),
    names_to = "variavel",
    values_to = "valor")

ggplot(df_grupos_long,
  aes(x = performance_lote, y = valor,
      color = performance_lote)) +
  geom_boxplot(fill = "steelblue", alpha = 0.65) +
  facet_wrap(~ variavel, scales = "free_y") +
  labs(title = "Distribuição das variáveis
    por grupo de rendimento",
    x = "Grupo de rendimento", y = "Valor observado") +
  guides(color = "none") +
  theme_minimal()
```

Análise Exploratória Bivariada

```
ggplot(df_clean, aes(x = mp_32, y = yield)) +  
  geom_point(color = "#2980b9", alpha = 0.7, size = 3) +  
  geom_smooth(method = "lm",  
              color = "#e74c3c", se = TRUE) +  
  labs(title = "Variável de Processo 32 vs Rendimento",  
        x = "Manufacturing Process 32",  
        y = "Rendimento (yield)") +  
  theme_minimal()
```

Visualização com ggpairs

```
library(GGally)

# Seleciona um subconjunto de variáveis

df_subset <- df_clean |>
  select(yield, mp_32, mp_11, bio_07)

ggpairs(df_subset)
```