Introdução à Análise Gráfica com R base

Janio Lima

08 novembro, 2022

Contents

1 Introdução

A análise gráfica (ou visual) é uma técnica fundamental para diversas etapas do processo de análise de dados. Na análise exploratória, sua importância é relacionada ao entedimento do conjunto de dados com o qual se trabalha. Na construção de modelos, a visualização é importante tanto para validar intuições sobre a validação de modelos quanto para análise dos resultados finais.

Há diversas ferramentas para análise gráfica, desde planilhas eletrônicas tracionais, ferramentas de Business Intelligence (BI) e linguagens de programção. Na linguagem R, há duas principais abordagens para esta análise: (1) **R base**: funções nativas de construção de gráficos; (2) **Tidyverse**: usando funções do pacote `ggplot2`.

Ambas alternativas possuem vantagens e desvantagens. Por exemplo, as funções nativas apresentam uma sintaxe mais simples e rodam em qualquer ambiente onde o R estiver instalado. O pacote ggplot2, apesar de ter uma sintaxe um pouco mais complexa, permite a construção de gráficos mais avançados e é muito bem integrado com outros pacotes do *universo* `tidyverse`.

Para exemplificar, construiremos um gráfico simples sobre o conjunto de dados de consumo de combustíveis de carros "mtcars" usando as funções básicas do R e sua versão equivalente no ggplot2. O conjunto de dados `mtcars` é um conjunto pequeno, com informações sobre 32 modelos de carros, com uma variável relativa ao desempenho de milhas por galão de combustível, que pode ser analisada como alvo. Há também 10 atributos dos veículos que podem ser analisadas como variáveis explicativas do consumo. Para analisar os detalhes do dataset pode ser usada a instrução `?mtcars` no console do R.

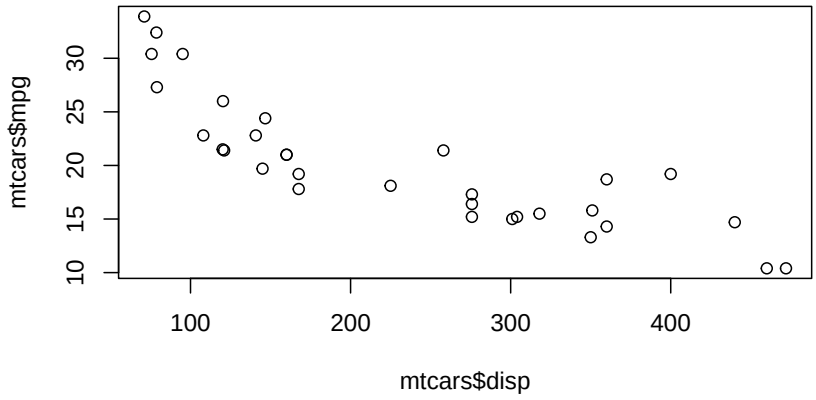
A Tabela "Dataset mtcars" exibe as primeiras observações do conjunto de dados em análise. O primeiro código a seguir cria uma visualização simples da dispersão entre o tamanho do motor (disp) e o consumo

de combustível em milhas por galão (mpg). Na sequência é demonstrada a criação da mesma visualização usando o pacote `ggplot2`.

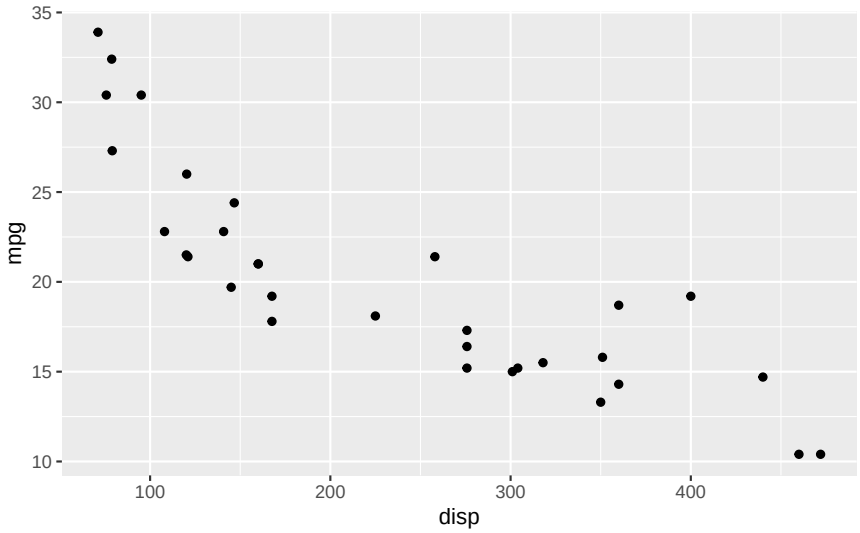
Table 1: Dataset mtcars: primeiras observações

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

```r
#Gráfico de dispersão usando R base
plot(x = mtcars$disp, y = mtcars$mpg)
```




```r
#Gráfico de dispersão usando ggplot2
library(ggplot2)
ggplot(data = mtcars) +
  geom_point(mapping = aes(x = disp, y = mpg))
```



O gráfico apresentado é bem simples e, em ambos exemplos, foi usada a forma mais básica de construção. Isto é, não foram realizadas quaisquer configurações adicionais na visualização além da indicação da origem de dados e o tipo de gráfico. Percebe-se que ambos os gráficos, apesar de terem detalhes visuais diferentes, transmitem a mesma informação acerca da mudança de consumo à medida que o tamanho do motor é menor ou maior. Também é possível notar que a primeira opção usa um pouco menos de código, contudo o resultado do `ggplot2` é um pouco mais atrativo mesmo na configuração mais básica.

Este trabalho tem objetivo de apresentar uma introdução à análise visual usando o R base. Esta escolha não configura qualquer predileção desta abordagem em detrimento do uso de `ggplot2`, mas apenas a capacitação do leitor na construção de gráficos úteis para suas análises sem dependência de qualquer pacote adicional à instalação básica do R. Além desta introdução, este trabalho apresenta no Capítulo 2 os tipos básicos de gráficos, em seguida o Capítulo 3 discute personalização de elementos complementares para melhor entendimento dos gráficos e o Capítulo 4 aborda visuais com mais de um gráfico gerado ao mesmo tempo e painéis com múltiplos gráficos. Por fim, as considerações finais destacam os elementos principais do estudo e alternativas de aprofundamento das técnicas de visualização.

2 Tipos básicos de gráficos

```
library(tidyverse, warn.conflicts = FALSE, verbose = FALSE, quietly = TRUE)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

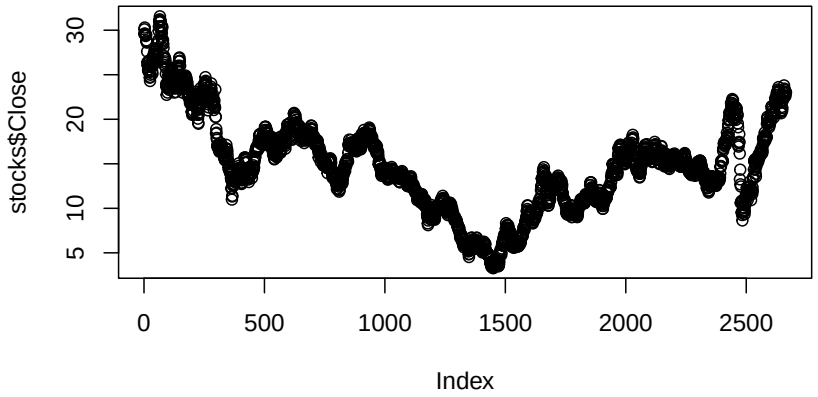
```
arquivo <- "https://raw.githubusercontent.com/janiosl/python.ds/master/data/yahoo_stock_12-12-2020.csv"
stocks <- read_csv(arquivo)
```

```
## Rows: 2665 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Ticker
## dbl  (6): High, Low, Open, Close, Volume, Adj Close
## date (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Na comparação entre gráficos criados com R base e `ggplot2`, já foi vista a função `plot`, uma das mais comuns para construção de gráficos. No exemplo anterior a função foi usada para criar um gráfico de dispersão, porém com pequenos ajustes podemos criar gráficos diferentes. Por exemplo, se o analista fornecer apenas um argumento de dados, ao invés da dispersão são exibidos os dados ao longo do tempo. O mesmo ocorre se o primeiro argumento for um campo data.

Na sequência de códigos abaixo temos três formas diferentes de exibir os mesmos dados. Inicialmente, será chamada a função `plot` passando apenas uma variável com o preço de fechamentos de uma ação. Como resultado é gerado um gráfico com os preços no eixo y e a sequência ordenada em que aparecem no dataset no eixo x.

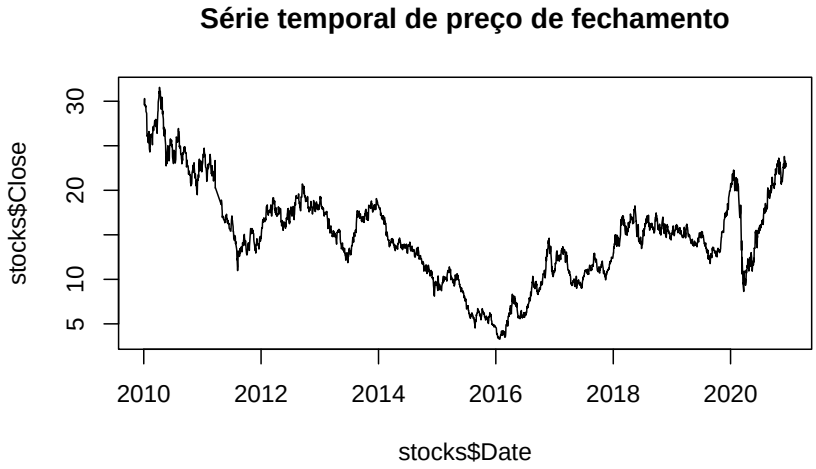
```
plot(stocks$Close
     )
```



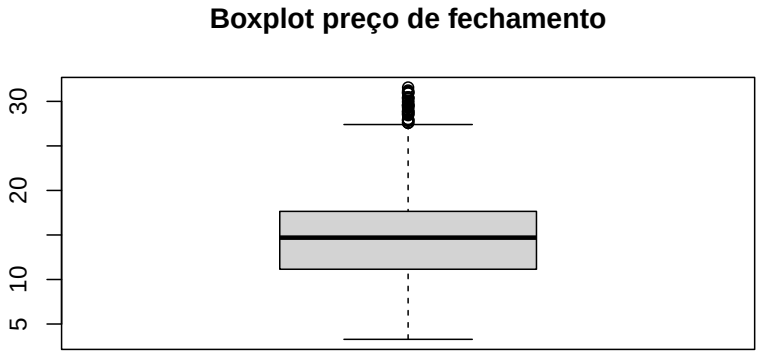
Como estamos tratando de mudanças em uma única variável ao longo do tempo, podemos mudar o tipo de gráfico para um gráfico de linha, adicionado o parâmetro `type = "l"`. Com esta pequena mudança já temos um gráfico mais adequado para análise de uma série temporal (dados numéricos organizados cronologicamente). No gráfico seguinte adicionamos a variável `Date`, assim fica mais explícito que estamos com uma série de tempo e acrescentamos um título ao gráfico com o parâmetro `main = "Título do gráfico"`. Outros parâmetros serão estudados no próximo capítulo, mas antes serão apresentados outros tipos básicos de gráficos.

```
plot(stocks$Close,
     type = "l"
     )
```

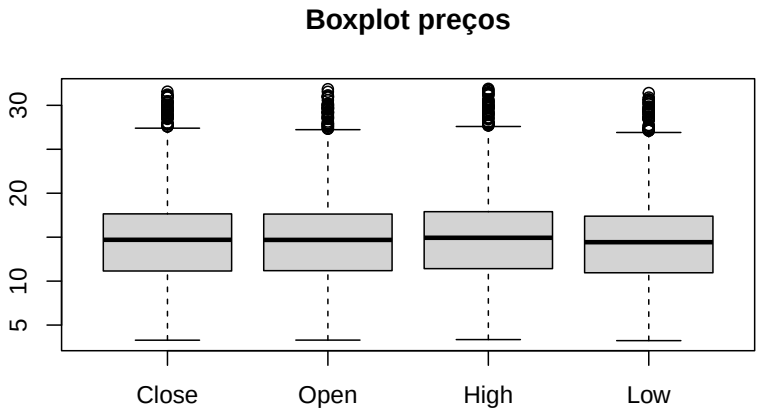

```
plot(stocks$Date, stocks$Close,
     type = "l",
     main = "Série temporal de preço de fechamento"
     )
```



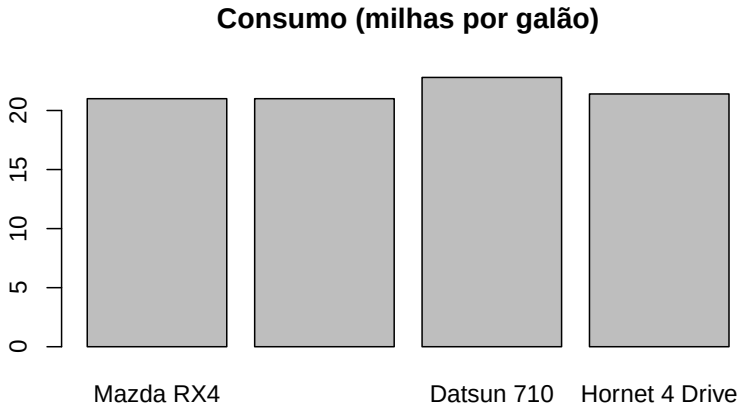

```
boxplot(stocks$Close,
        main = "Boxplot preço de fechamento")
```



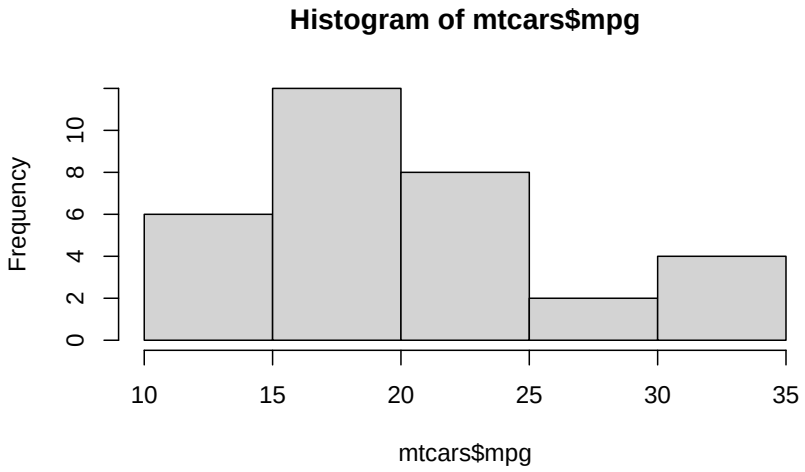

```
boxplot(stocks$Close,
        stocks$Open,
        stocks$High,
        stocks$Low,
        names = c("Close", "Open", "High", "Low"),
        main = "Boxplot preços")
```




```
barplot(mtcars[1:4,]$mpg,
        width = 0.7,
        names = row.names(mtcars[1:4,]),
        main = "Consumo (milhas por galão)"
        )
```




```
hist(mtcars$mpg)
```



3 Personalização de elementos complementares

4 Gráficos compostos e painéis

5 Considerações Finais

6 Referências