Programação no R - Aula 9

Disciplina: Lógica da programação (de computadores) e análise de dados no R

Prof. Maurício Garcia de Camargo. IO-FURG.

2025-11-13

Funções vetorizadas

Vocês aprenderam a fazer na unha a transformação de um vetor de abundância em vetor de presência-ausência.

```
 1  pres_aus = c()  # cria vetor vazio
 2
 3  x = c(23, 0, 5, 0, 12)
 4
 5  for (i in 1:length(x)) {
 6    if (x[i] > 0)
 7      pres_aus[i] = 1
 8     else
 9      pres_aus[i] = 0
10  }
11
12  pres_aus
```

```
[1] 1 0 1 0 1
```

Funções vetorizadas

Isso pode ser feito de maneira muito mais simples com a função vetorizada `ifelse`.

```
1 x = c(23, 0, 5, 0, 12)
2 pres_aus = ifelse(x > 0, 1, 0)
3 pres_aus
```

```
[1] 1 0 1 0 1
```

Uma **função vetorizada** avalia todas as posições de um vetor lógico de uma só vez e devolve um vetor do mesmo comprimento, escolhendo elemento a elemento entre os valores de ***Verdadeiro*** ou ***falso***.

Funções vetorizadas

Outro exemplo:

```
1 x = c(-2, 0, 3, 0)
2 ifelse(x == 0, "sim", "não")
```

```
[1] "não" "sim" "não" "sim"
```

`ifelse(x == 0, "sim", "não")`

deve ser lido como: para cada valor do vetor `x`, se for igual a zero substitui por “sim”, caso contrário, por “não”.

Exercício. Funções vetorizadas

- Baixe a planilha `Plan1.xlsx` da página da disciplina.
- Abra o arquivo e armazene no dataframe chamado de `dat`.
- Crie um novo dataframe para o registro de presença-ausência das três espécies usando `ifelse`.

Rever aggregate() da aula passada

Breve aula sobre SQL

- *SQL = Structured Query Language*
- Nasceu na IBM, em 1970.
- A ideia é guardar informações em tabelas (linhas e colunas) e depois relacioná-las através de uma linguagem parecida com o inglês.
- SQL virou padrão universal como linguagem de banco de dados.

Breve aula sobre SQL

- *SQL é capaz de fazer tudo com banco de dados*
- Criar estruturas de dados: criar, alterar e apagar bancos, tabelas, índices.
- Consultar dados: o clássico SELECT … FROM … WHERE ….
- Inserir, atualizar e excluir dados (INSERT, UPDATE, DELETE).
- Criar relatórios e análises: agregações (SUM, AVG, COUNT), agrupamentos (GROUP BY), filtros, junções entre tabelas (JOIN).

Breve aula sobre SQL

Vamos começar criando um dataframe rudimentar chamado `tab`.

```
1  set.seed(12)
2  col1 = c(rep('A',4),rep('B',4))
3  col2 = trunc(runif(8,0,4))
4  col3 = rnorm(8,10,4)
5  tab = data.frame(col1,col2,col3)
6  tab
```

```
  col1 col2      col3
1    A    0  2.009432
2    A    3  8.910816
3    A    3  8.738605
4    A    1  7.486979
5    B    0  9.574144
6    B    0 11.712059
7    B    0  6.889122
8    B    2  4.824471
```

Breve aula sobre SQL

A função sqldf() cria um novo dataframe baseado na seleção feita pela linguagem SQL, que é muito simples.

Exemplos:

```
1 library(sqldf)
2 # Seleciona todas as colunas (*) e todas as linhas
3 sqldf('select * from tab')
```

```
  col1 col2      col3
1    A    0  2.009432
2    A    3  8.910816
3    A    3  8.738605
4    A    1  7.486979
5    B    0  9.574144
6    B    0 11.712059
7    B    0  6.889122
8    B    2  4.824471
```

Breve aula sobre SQL

```
1 # Seleciona apenas a primeira colunas e todas as linhas
2 sqldf('select col1 from tab')
```

```
  col1
1    A
2    A
3    A
4    A
5    B
6    B
7    B
8    B
```

```
1 # Seleciona as duas primeira colunas e todas as linhas
2 sqldf('select col1,col2 from tab')
```

```
  col1 col2
1    A    0
2    A    3
3    A    3
4    A    1
5    B    0
6    B    0
7    B    0
8    B    2
```

Breve aula sobre SQL

```
1 # Filtro para as linhas (usando where)
2 sqldf('select * from tab where col1="A" ')
```

```
  col1 col2     col3
1    A    0 2.009432
2    A    3 8.910816
3    A    3 8.738605
4    A    1 7.486979
```

```
1 # Filtro para as linhas com operador booleano (and / or)
2 sqldf('select * from tab where col1="A" and col2=3')
```

```
  col1 col2     col3
1    A    3 8.910816
2    A    3 8.738605
```

```
1 sqldf('select * from tab where col1="A" or col2=3')
```

```
  col1 col2     col3
1    A    0 2.009432
2    A    3 8.910816
3    A    3 8.738605
4    A    1 7.486979
```

Breve aula sobre SQL

```
1 # Cálculos de médias (avg), soma (sum), contagem (count) etc.
2 sqldf('select avg(col3) from tab')
```

```
  avg(col3)
1  7.518203
```

```
1 # Cálculos por grupos (usando group by)
2 sqldf('select avg(col3) from tab group by col1')
```

```
  avg(col3)
1  6.786458
2  8.249949
```

```
1 # Adicionando col1
2 sqldf('select col1, avg(col3) from tab group by col1')
```

```
  col1 avg(col3)
1    A  6.786458
2    B  8.249949
```

Breve aula sobre SQL

A sentença abaixo cria mesma tabela que usamos com `aggregate()` na aula passada.

```
sqldf('select setor, estacao, avg(abund)
from dad group by setor,estacao')
```

Vejam o arquivo R de exercícios desta aula.