# Další datové struktury a práce s nimi, matice, datové tabulky, seznamy

—

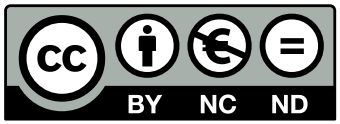
17VSADR – Skriptování a analýza dat v jazyce R

Lubomír Štěpánek[1, 2]

[1]Oddělení biomedicínské statistiky & výpočetní techniky
Ústav biofyziky a informatiky
1. lékařská fakulta
Univerzita Karlova v Praze

ČVUT FBMI

[2]Katedra biomedicínské informatiky
Fakulta biomedicínského inženýrství
České vysoké učení technické v Praze



# Obsah

# Datové struktury

- vektor (`vector`)
- faktor (`factor`)
- matice (`matrix`)
- pole (`array`)
- tabulka dat (`data.frame`)
- seznam (`list`)

# Tvorba matic a základní příkazy

- matice (`matrix`) je dvojrozměrným polem, které obsahuje prvky stejného datového typu
- všechny sloupce matice mají shodnou délku, podobně všechny řádky matice mají shodnou délku
- ať je

$$\boldsymbol{A} = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \qquad \boldsymbol{B} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

- v R matici $\boldsymbol{A}$ a $\boldsymbol{B}$ získáme příkazy

```
A <- matrix(c(1, 2, 3, 4), nrow = 2,
            ncol = 2)
B <- matrix(c(1, 3, 2, 4), nrow = 2,
            ncol = 2)
B <- matrix(c(1, 2, 3, 4), nrow = 2,
            ncol = 2, byrow = TRUE)
# vždy jeden z argumentů "nrow" či "ncol" je zbytný
```

# Manipulace s maticemi

- ať je

$$\boldsymbol{C} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

- v R pak pomocí

```
C <- matrix(letters[1:12], nrow = 3,
            byrow = T)
```

- některé užitečné příkazy

```
is.matrix(C)    # TRUE
class(C)        # "matrix"
mode(C)         # "character"; datový typ pruků
str(C)          # chr [1:3, 1:4] "a" "e" "i" ...
dim(C)          # c(3, 4); rozměry matice C
```

# Manipulace s maticemi

- ať je

$$\boldsymbol{C} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

- další užitečné příkazy

```
colnames(C) <- c("c1", "c2", "c3", "c4")
rownames(C) <- c("r1", "r2", "r3")
              # přidá jmenovky sloupcům i řádkům
C <- unname(C)
              # zbaví sloupce i řádky jmenovek
dimnames(C) <- list(
                    c("r1", "r2", "r3"),
                    c("c1", "c2", "c3", "c4")
                  )
              # opět přidá jmenovky sloupcům i řádkům
```

# Manipulace s maticemi

- stále mějme

$$\boldsymbol{C} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

- další užitečné příkazy

```
rbind(C, c("x", "x", "x", "x"))
                # přidání řádku c("x", "x", "x", "x")
                # k matici C
cbind(C, c("x", "x", "x"))
                # přidání sloupce c("x", "x", "x")
                # k matici C
C[-1, ]         # odebrání 1. řádku matici C
C[, -2]         # odebrání 2. sloupce matici C
```

# Submatice, indexování, adresace

- ať je

$$\boldsymbol{C} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

- v R pomocí

```
C <- matrix(letters[1:12], nrow = 3,
            byrow = T, dimnames = list(
            c("r1", "r2", "r3"),
            c("c1", "c2", "c3", "c4")))
```

- adresace

```
C[2, 3]          # "g"; prvek 2. řádku, 3. sloupce
C["r2", "c3"]    # "g"; prvek 2. řádku, 3. sloupce
C[1, ]           # c("a", "b", "c", "d");
                 # tedy vektor 1. řádku matice C
                 # s popisky
```

# Submatice, indexování, adresace

- stále mějme

$$\boldsymbol{C} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

- adresace

```
C[, 3]              # c("c", "g", "k");
                    # tedy vektor 3. sloupce matice C
                    # s popisky
C[c(1, 3), c(2, 4)]
                    # matrix(c("b", "j", "d", "l"), 2)
                    # submatice 1. a 3. řádku,
                    # 2. a 4. sloupce matice C
                    # s popisky
C["r2", ]           # c("e", "f", "g", "h");
                    # tedy vektor 2. řádku matice C
                    # s popisky
```

# Submatice, indexování, adresace

- stále mějme

$$\boldsymbol{C} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

- adresace

```
C[dim(C)[1], dim(C)[2]]
                # "l"; obecná adresace prvku
                # vpravo dole (např. neznáme-li
                # číselné rozměry matice)
C[5]            # "f"; major-column ordering
C[c(8, 9)]      # c("g", "k")
C[13]           # NA
diag(C)         # c("a", "f", "k"); hlavní diagonála
diag(C[, dim(C)[2]:1])
                # c("d", "g", "j"); vedlejší diagonála
```

# Intermezzo

- mějme matici

$$\boldsymbol{X} = \begin{pmatrix} 1 & 3 & 9 & 6 \\ -5 & 0 & 1 & 2 \\ -8 & 4 & 7 & 3 \end{pmatrix}$$

- pomocí R
  - (i) založme matici $\boldsymbol{X}$
  - (ii) najděme prvek třetího řádku druhého sloupce matice $\boldsymbol{X}$
  - (iii) jaký řádkový a sloupcový index mají prvky $0$, $7$ matice $\boldsymbol{X}$
  - (iv) najděme minimum matice $\boldsymbol{X}$ a řádkový a sloupcový index této hodnoty
  - (v) najděme stopu matice $\boldsymbol{X}$

# Intermezzo

- mějme matici

$$\boldsymbol{X} = \begin{pmatrix} 1 & 3 & 9 & 6 \\ -5 & 0 & 1 & 2 \\ -8 & 4 & 7 & 3 \end{pmatrix}$$

- řešení

```
# (i)
X <- matrix(c(1, -5, -8, 3, 0, 4, 9, 1,
              7, 6, 2, 3), nrow = 3)
# (ii)
X[3, 2]       # 4
# (iii)
c((which(X == 0) - 1) %% dim(X)[1] + 1,
  ceiling(which(X == 0) / dim(X)[1]))
c((which(X == 7) - 1) %% dim(X)[1] + 1,
  ceiling(which(X == 7) / dim(X)[1]))
```

# Intermezzo

- mějme matici

$$\boldsymbol{X} = \begin{pmatrix} 1 & 3 & 9 & 6 \\ -5 & 0 & 1 & 2 \\ -8 & 4 & 7 & 3 \end{pmatrix}$$

- řešení

```
# (iii) lépe
which(X == 0, ind.arr = TRUE)
which(X == 7, ind.arr = TRUE)
          # which(X == 1, ind.arr = TRUE)
          # vrátí c(1, 2) i c(2, 3)
# (iv)
min(X); which(X == min(X), arr.ind = T)
          # -8; c(3, 1)
# (v)
sum(diag(X))
          # 8 := 1 + 0 + 7
```

# Maticová algebra

- buďte

$$\boldsymbol{A} = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \qquad \boldsymbol{B} = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix} \qquad \boldsymbol{b} = (1, 0)^T$$

- v R

```
A <- matrix(c(1, 2, 3, 4), nrow = 2)
B <- matrix(c(5, 6, 7, 8), nrow = 2)
b <- c(1, 0)
```

- Hadamardův součin (*element-wise*, *pairwise*) $\boldsymbol{A} \circ \boldsymbol{B} = \begin{pmatrix} 5 & 21 \\ 12 & 32 \end{pmatrix}$

```
A * B       # matrix(c(5, 12, 21, 32), 2)
```

- maticový součin $\boldsymbol{A} \cdot \boldsymbol{B} = \begin{pmatrix} 23 & 31 \\ 34 & 46 \end{pmatrix}$

```
A %*% B     # matrix(c(23, 34, 31, 46), 2)
```

# Maticová algebra

- buďte

$$\boldsymbol{A} = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \qquad \boldsymbol{B} = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix} \qquad \boldsymbol{b} = (1, 0)^T$$

- transpozice $\boldsymbol{A}^T = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

```
t(A)          # matrix(c(1, 3, 2, 4), 2)
```

- jednotková matice $\boldsymbol{I}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

```
diag(2)       # matrix(c(1, 0, 0, 1), 2)
```

- inverzní matice $\boldsymbol{A}^{-1} = \begin{pmatrix} -2 & 3/2 \\ 1 & -1/2 \end{pmatrix}$

```
solve(A)      # matrix(c(-2, 1, 1.5, -0.5), 2)
```

# Maticová algebra

- buďte

$$\boldsymbol{A} = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \qquad \boldsymbol{B} = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix} \qquad \boldsymbol{b} = (1, 0)^T$$

- vlastní čísla matice $\boldsymbol{A}$ jsou $\boldsymbol{\lambda} = (\frac{1}{2}(5 + \sqrt{33}), \frac{1}{2}(5 - \sqrt{33}))^T$

```
eigen(A)$values
```

- vlastní vektory matice $\boldsymbol{A}$ jsou například
  $\boldsymbol{v} = [(\frac{1}{4}(-3 + \sqrt{33}), 1)^T, (\frac{1}{4}(-3 - \sqrt{33}), 1)^T]$

```
eigen(A)$vectors    # R vrací ortonormální
                    # vlastní vektory
```

- řešení $\boldsymbol{x}$ ze soustavy $\boldsymbol{Ax} = \boldsymbol{b}$ je $\boldsymbol{x} = (-2, 1)^T$

```
solve(A, b)         # c(-2, 1)
```

# Intermezzo

- uvažujme markovský proces s počátečním stavovým vektorem $\boldsymbol{p}^0 = (1, 0, 0)^T$ a přechodovou maticí

$$\boldsymbol{S} = \begin{pmatrix} 1/4 & 1/2 & 1/4 \\ 2/3 & 0 & 1/3 \\ 1/2 & 1/2 & 0 \end{pmatrix},$$

  kde v $i$-tém řádku jsou vždy po řadě pravděpodobnosti přechodu z $i$-tého stavu do $j$-tého stavu pro $\forall i, j \in \{1, 2, 3\}$

- pomocí R
  - (i) ověřme, že matice $\boldsymbol{S}$ je stochastická
  - (ii) určeme pravděpodobnosti, s jakými proces dospěl po pěti krocích do prvního, druhého či třetího stavu
  - (iii) určeme, zda existuje dynamicky rovnovážný stav procesu

# Intermezzo

- uvažujme markovský proces s počátečním stavovým vektorem $\boldsymbol{p}^0 = (1, 0, 0)^T$ a přechodovou maticí

$$\boldsymbol{S} = \begin{pmatrix} 1/4 & 1/2 & 1/4 \\ 2/3 & 0 & 1/3 \\ 1/2 & 1/2 & 0 \end{pmatrix},$$

  kde v $i$-tém řádku jsou vždy po řadě pravděpodobnosti přechodu z $i$-tého stavu do $j$-tého stavu pro $\forall i, j \in \{1, 2, 3\}$

- řešení

```
S <- matrix(c(1/4, 1/2, 1/4,
              2/3, 0, 1/3,
              1/2, 1/2, 0), nrow = 3,
              byrow = TRUE)
p <- c(1, 0, 0)
```

# Intermezzo

- uvažujme markovský proces s počátečním stavovým vektorem $\boldsymbol{p}^0 = (1, 0, 0)^T$ a přechodovou maticí

$$\boldsymbol{S} = \begin{pmatrix} 1/4 & 1/2 & 1/4 \\ 2/3 & 0 & 1/3 \\ 1/2 & 1/2 & 0 \end{pmatrix},$$

  kde v $i$-tém řádku jsou vždy po řadě pravděpodobnosti přechodu z $i$-tého stavu do $j$-tého stavu pro $\forall i, j \in \{1, 2, 3\}$

- řešení

```
# (i)
rowSums(S)          # c(1, 1, 1)
```

# Intermezzo

- uvažujme markovský proces s počátečním stavovým vektorem $\boldsymbol{p}^0 = (1, 0, 0)^T$ a přechodovou maticí

$$\boldsymbol{S} = \begin{pmatrix} 1/4 & 1/2 & 1/4 \\ 2/3 & 0 & 1/3 \\ 1/2 & 1/2 & 0 \end{pmatrix},$$

  kde v $i$-tém řádku jsou vždy po řadě pravděpodobnosti přechodu z $i$-tého stavu do $j$-tého stavu pro $\forall i, j \in \{1, 2, 3\}$

- řešení

```
# (ii)
install.packages("expm", dependency = T)
library("expm")
p %*% (S %^% 5)    # c(0.437, 0.344, 0.219)
```

# Intermezzo

- uvažujme markovský proces s počátečním stavovým vektorem $\boldsymbol{p}^0 = (1, 0, 0)^T$ a přechodovou maticí

$$\boldsymbol{S} = \begin{pmatrix} 1/4 & 1/2 & 1/4 \\ 2/3 & 0 & 1/3 \\ 1/2 & 1/2 & 0 \end{pmatrix},$$

  kde v $i$-tém řádku jsou vždy po řadě pravděpodobnosti přechodu z $i$-tého stavu do $j$-tého stavu pro $\forall i, j \in \{1, 2, 3\}$

- řešení

```
# (iii)
p %*% (S %^% 10)   # c(0.444, 0.333, 0.222)
p %*% (S %^% 20)   # c(0.444, 0.333, 0.222)
p %*% (S %^% 30)   # c(0.444, 0.333, 0.222)
p %*% (S %^% 40)   # c(0.444, 0.333, 0.222)
```

# Tvorba polí a práce s nimi

- pole je objekt s vlastnostmi matice, ale libovolným počtem dimenzí
- např.

```
(X <- array(c(1:24), dim = c(4, 3, 2)))
# , , 1
# [,1] [,2] [,3]
# [1,] 1 5 9
# [2,] 2 6 10
# [3,] 3 7 11
# [4,] 4 8 12
# , , 2
# [,1] [,2] [,3]
# [1,] 13 17 21
# [2,] 14 18 22
# [3,] 15 19 23
# [4,] 16 20 24
```

# Manipulace s poli

- navigace v poli

```
X[, , 1]      # první vrstva
X[, , 2]      # druhá vrstva
X[3, , 1]     # třetí řádek první vrstvy
X[, 2, ]      # druhé sloupce všech vrstev
```

- pole se příliš nepoužívají, protože vstupem pro analytické funkce jazyka R je obvykle matice či datová tabulka
- pole libovolných rozměrů lze převést na dvourozměrnou matici pomocí dummy proměnné „vrstva“

# Tvorba datových tabulek a základní příkazy

- datová tabulka (`data.frame`) je dvojrozměrným polem, které obsahuje v každém sloupci prvky stejného datového typu, ale jednotlivé sloupce se mohou datovým typem lišit
- všechny sloupce datové tabulky mají shodnou délku, podobně všechny řádky datové tabulky mají shodnou délku
- v R je řada vestavěných datových tabulek

```
mtcars
str(mtcars)
class(mtcars)                # "data.frame"
mode(mtcars)                 # "list"
is.data.frame(mtcars)        # TRUE
str(iris)
     # 'data.frame': 150 obs. of 5 variables
     # ...
dim(iris)                    # c(150, 5)
```

# Manipulace s datovými tabulkami

- užitečné příkazy

```
data <- mtcars
colnames(data)
colnames(data) <- paste("c",
                        1:dim(data)[2],
                        sep = "_")
rownames(data) <- paste("r",
                        1:dim(data)[1],
                        sep = "_")
                  # změní jmenovky sloupcům i řádkům
head(data)        # náhled na prvních 6 řádků
head(data, 10)
                  # náhled na prvních 10 řádků
tail(data)        # náhled na posledních 6 řádků
tail(data, 10)
                  # náhled na posledních 10 řádků
```

# Manipulace s datovými tabulkami

- další užitečné příkazy

```
rbind(data, rep(0, dim(data)[2]))
                # přidání řádku c(0, 0, ..., 0)
                # k data.frameu "data"
cbind(data, rep(0, dim(data)[1]))
                # přidání sloupce c(0, 0, ..., 0)
                # k data.frameu "data"
data.frame(data,
           "ahoj" = rep(0, dim(data)[1]))
                # přidání sloupce c(0, 0, ..., 0)
                # se jménem "ahoj" k data.frameu
                # "data"
data[-1, ]      # odebrání 1. řádku data.frameu
                # "data"
data[, -1]      # odebrání 1. sloupce data.frameu
                # "data"
```

# Indexování, adresace

- adresace

```
data[2, 3]      # 160; prvek 2. řádku, 3. sloupce
data["r_2", "c_3"]
                # 160; prvek pod danými popisky
data[1, ]       # c(21, 6, 160, 110, ...);
                # tedy vektor 1. řádku data.frameu
                # "data" s popisky
data[, 2]       # c(6, 6, 4, 6, ...);
                # tedy vektor 2. sloupce
                # data.frameu "data" s popisky
data$c_5        # c(3.90, 3.90, 3.85, 3.08, ...);
                # tedy vektor 2. sloupce
                # data.frameu "data" s popisky
data$c_5[1]     # 3.9;
                # tedy první prvek 2. sloupce
                # data.frameu "data"
```

# Indexování, adresace

- adresace

```
data[dim(data)[1], dim(data)[2]]
                # 2; obecná adresace pruku
                # vpravo dole (např. neznáme-li
                # číselné rozměry tabulky)
data[5]         # 5. sloupec, nikoliv major-column
                # ordering
```

# Sloupcové přehledy

- může mít smysl znát agregovaný ukazatel nad všemi sloupci

```
colSums(data)
                # součty všech sloupců
apply(data, 2, sum)
                # totéž
colMeans(data)
                # průměry všech sloupců
apply(data, 2, mean)
                # totéž
data <- rbind(data, rep(NA, dim(data)[2]))
                # přidán řádek c(NA, NA, ...)
colMeans(data)
                # c(NA, NA, ...); pro získání průměrů
                # nutné přidat argument na.rm = TRUE
colMeans(data, na.rm = TRUE)
apply(data, 2, mean, na.rm = TRUE)
                # už funguje
```

# Tvorba seznamů a základní příkazy

- seznam (`list`) je výčtem prvků, které mohou být různorodého datového typu, včetně listu
- délky jednotlivých prvků seznamu se mohou lišit

```
my_list <- list("a" = c(1:10),
                "b" = mtcars,
                "c" = matrix(1:8, 2),
                "z" = "ahoj")
str(my_list)
class(my_list)          # "list"
mode(my_list)           # "list"
is.list(my_list)        # TRUE
```

# Manipulace se seznamy

- užitečné příkazy

```
names(my_list)
names(my_list) <- LETTERS[
    1:length(my_list)
]                   # přejmenovávám prvky listu

my_list[[length(my_list) + 1]] <- c(T, F)
names(length(my_list)) <- "XY"
                    # přidání vektoru c(T, F)
                    # k listu "my_list" pod jménem
                    # "XY"
```

# Indexování, adresace

- adresace

```
my_list[[2]]   # 2. prvek listu
my_list[["B"]]
               # prvek listu pod jmenovkou "B"
               # jde o původní datový typ
               # (data.frame)
my_list["B"]   # prvek listu pod jmenovkou "B"
               # jde (vždy) o list
my_list[c(2, 4)]
               # 2. a 4. prvek listu
my_list$C      # prvek listu pod jmenovkou "C"
my_list[[1]][2]
               # 2; 2. prvek 1. prvku listu
my_list[[2]][3, 5]
               # 3.85; z 2. prvku listu vybírám
               # prvek o souřadnicích (3, 5)
```

# Indexování, adresace pomocí funkce `lapply()`

- adresace

```
set.seed(1)
my_long_list <- lapply(
    sample(c(80:120), 100, TRUE),
    function(x) sample(
        c(50:150), x, replace = TRUE
    )
)    # list vektorů náhodné délky
    # generovaných z náhodných čísel

lapply(my_long_list, "[[", 14)
    # z každého prvku listu (vektoru)
    # vybírám jen jeho 14. prvek
```

# Prvkové přehledy

- může mít smysl znát agregovaný ukazatel nad všemi prvky seznamu

```
lapply(my_long_list, mean)
    # pro každý prvek listu (vektor)
    # vracím jeho průměr

lapply(my_long_list, length)
    # pro každý prvek listu (vektor)
    # vracím jeho délku
```

# Literatura


Karel Zvára. *Základy statistiky v prostředí R*. Praha, Česká republika: Karolinum, 2013. ISBN: 978-80-246-2245-3.


Hadley Wickham. *Advanced R*. Boca Raton, FL: CRC Press, 2015. ISBN: 978-1466586963.

## Děkuji za pozornost!

lubomir.stepanek@lf1.cuni.cz
lubomir.stepanek@fbmi.cvut.cz

github.com/LStepanek/17VSADR_Skriptovani_a_analyza_dat_v_jazyce_R