Štatistika a programovanie v R

Aleš Kozubík
Žilinská univerzita v Žiline

Project: Innovative Open Source Courses for Computer Science

31. 5. 2021

Innovative Open Source Courses for Computer Science

This teaching material was written as one of the outputs of the project
"Innovative Open Source Courses for Computer Science",
funded by the Erasmus+ grant no. 2019-1-PL01-KA203-065564.
The project is coordinated by West Pomeranian University of Technology in Szczecin (Poland)
and is implemented in partnership with Mendel University in Brno (Czech Republic)
and University of Žilina (Slovak Republic).
The project implementation timeline is September 2019 to December 2022.

Innovative Open Source Courses for Computer Science

Project was implemented under the Erasmus+.
Project name: "Innovative Open Source courses for Computer Science curriculum"
Project no.: 2019-1-PL01-KA203-065564
Key Action: KA2 – Cooperation for innovation and the exchange of good practices
Action Type: KA203 – Strategic Partnerships for higher education

Consortium: Zachodniopomorski uniwersytet technologiczny w Szczecinie
Mendelova univerzita v Brně
Žilinská univerzita v Žiline

Erasmus+ Disclaimer: This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Štatistika a programovanie v R

I. Úvod do prostredia R

Charakteristika jazyka R

- je to jazyk a slobodné softvérové prostredie špecializované na štatistické výpočty a vizualizáciu dát,
- je dostupný pre všetky bežné operačné systémy: UNIX-ové platformy, Win alebo MacOS,
- je alternatívou ku komerčnému nástroju S resp. S-plus (jazyk a prostredie), ktorý vyvinula spoločnosť AT&T.

Prečo R

- je bezplatný, väčšina platforiem štatistického softvéru stojí tisíce dolárov,
- ku programu je dostupné veľké množstvo rozširujúcich balíčkov,
- R dokáže ľahko importovať údaje z rôznych zdrojov,
- R má implementovaných mnoho pokročilých štatistických nástrojov,
- R poskytuje interaktívnu platformu na analýzu údajov,
- prostredie R ponúka vizualizáciu dát vo forme veľmi kvalitných a estetických grafov,
- je platformovo nezávislé, kompatibilné s väčšinou najrozšírenejších operačných systémov,
- je kompatibilný s programovacími jazykmi ako C,C++, Python, Java.

Inštalácia R

Je voľne dostupný zo zdroja Comprehensive R Archive Network (skratka CRAN).

Na internete je umiestnený na adrese https://cran.r-project.org.

K dispozícii sú predkompilované binárne súbory pre bežné platformy Linux, Mac OS a Windows.

Pre stiahnutie inštalačného balíčka si môžeme vybrať vhodné zrkadlo.

Inštalácia balíčkov R

Ku jadru R existuje bohatý súbor balíčkov rozširujúcich jeho funkcie.

Balíčky zvyšujú výkonnosť R.

Na inštaláciu balíčkov používame funkciu `install.packages()`

R Prvé spustenie

Ak sme nainštalovali R, môžeme overiť jeho funkčnosť.

Prostredie R spustíme jednoducho z príkazového riadku zadaním:

```
username@host:~$ R
```

Zobrazí sa krátka úvodná poznámka, ktorá je nasledovaná znakom

>

Tento symbol je znakom príkazového riadku prostredia R.

Opustenie prostredia R

Prostredie R je teraz pripravené na prácu.

Pre ukončeniu práce v prostredí R jednoducho zadáme

```
> q()
```

R reaguje otázkou:

```
Save workspace image? [y/n/c]:
```

Ak zvolíme `y`, záznam celej histórie vykonaných príkazov sa uloží do súboru `.Rhistory`, ktorý sa zapisuje do pracovného adresára.

Pracovná plocha a navigácia

Všetky príkazy zadávame interaktívne na príkazovom riadku.

V histórii príkazov sa pohybujeme s použitím kurzorových klávesov, šípok smerom nahor a nadol.

To umožňuje vrátiť sa ku starším príkazom bez nutnosti ich prepisovania. Iba si vyberieme požadovaný príkaz a opätovne ho odošleme klávesou Enter.

Ak si pri odchode z prostredia uložíme históriu, môžeme sa vrátiť aj ku príkazom z predchádzajúcej relácie.

Komunikácia s OS

Predvolený pracovný adresár je adresár, v ktorom bol spustený program R. V tomto aktuálnom pracovnom adresári R číta a ukladá súbory a výsledky. Aktuálny pracovný adresár zistíme pomocou funkcie `getwd()`.

Aktuálny pracovný adresár môžeme zmeniť pomocou funkcie `setwd()`.

Na spustenie príkazov operačného systému použijeme funkciu `system()`.

Nový adresár vytvoríme príkazom

```
> system("mkdir new")
```

Získanie nápovedy

Funkcia na získanie nápovedy má vo všeobecnosti jednoduchý tvar `help()` alebo skrátene pomocou operátora `?`.

Ak chceme získať informácie o rozširujúcich balíčkoch, použijeme

```
> help(package="meno balíčka")
```

Niektoré balíčky obsahujú aj ukážky kódu, ktoré spustíme pomocou funkcie `demo()`

```
> demo(package="stats")
```

R ako kalkulátor

Konzola príkazového riadku umožňuje interaktívny výpočet výsledkov operácií a funkcií

```
> 5+3
[1] 8
```

Ak nevidíme nový znak príkazového riadku, môže to byť spôsobené tým, že sme zadali neúplný príkaz

```
> 5-
+
```

Musíme napísať zvyšok príkazu a potom stlačiť Enter alebo zrušiť príkaz stlačením tlačidla Esc.

Aritmetické operácie v R

+	Sčítanie.
-	Odčítanie.
*	Násobenie.
/	Delenie.
^	Umocňovanie.
%%	Modul (Zvyšok celočíselného delenia).
%/%	Celočíselný podiel.

Relačné operátory v R

<	Menšie
>	Väčšie
<=	Menšie alebo rovné
>=	Väčšie alebo rovné
==	Rovná sa
!=	Nerovná sa

Často používané matematické funkcie

`exp()`	Exponenciálna e^x	`sqrt()`	Druhá odmocnina
`log()`	Logaritmus (implicitne prirodzený)	`abs()`	Absolútna hodnota
`log10()`	Logaritmus so základom 10		
`sin()`	Sínus	`asin()`	Arkussínus
`cos()`	Kosínus	`acos()`	Arkuskosínus
`tan()`	Tangens	`atan()`	Arkustangens
`round()`	Zaokrúhľovanie (implicitne celé číslo)		

Objekty

R je objektovo orientovaný jazyk

V R je všetko objektom a predstavuje nejaké údaje, ktoré boli uložené v pamäti

Objekty môžu mať ľubovoľné meno, musia sa však rešpektovať tieto pravidlá:

- názov pozostáva len z malých alebo veľkých písmen, číslic, podčiarkovníkov a bodiek,
- názov začína veľkým alebo malým písmenom,
- R rozlišuje veľkosť písmen (to znamená, že A a a sú dva rôzne objekty),
- názov nesmie byť žiadne z rezervovaných slov R (ich zoznam zobrazíme po zadaní `help(reserved)`),

Vytváranie objektov

Nový objekt vytvoríme jednoducho pomocou operátora priradenia.

Operátor priradenia má dve možné podoby: <- alebo =.

Odporúča sa používať <-, pretože = môže niekedy viesť k chybám:

```
> log(x=25,base=5)
[1] 2
> x
Error: object 'x' not found
> log(x<-25,base=5)
[1] 2
> x
[1] 25
```

Zoznam a odstraňovanie objektov

Zoznam všetkých existujúcich objektov získame ako výstup funkcie `ls()`.

Objekty, ktoré v budúcnosti nebudeme používať, môžeme z pamäte odstrániť pomocou funkcie `rm()`.

Štatistika a programovanie v R

II. Dátové štruktúry v R

Typy dát a dátové štruktúry

Existuje 5 základných typov údajov

- numeric,
- integer,
- complex,
- logical,
- character.

Môžu byť agregované do dátových štruktúr:

- vector,
- matrix,
- list,
- frame.

Dáta typu `numeric`

Dátový typ `numeric` predstavuje reálne desatinné čísla .

Je to implicitný typ každého nového objektu.

Vznikne ak priradíme ľubovoľnej premennej reálne číslo.

Typ akéhokoľvek objektu overíme pomocou funkcie `class()`

Dáta typu numeric – príklad

Pozrime sa na príklad.

```
1 > x<-12.35
2 > class(x)
3 [1] "numeric"
```

Poznámka

Číslo je reprezentované ako vektor s dĺžkou 1. Znak [1] na začiatku riadku znamená prvú pozíciu v tomto vektore.

Dáta typu integer

Na vytvorenie objektu typu `integer` použijeme funkciu `as.integer()`

Príklad

```
> a <- as.integer(12)
> a
[1] 12
> class(a)
[1] "integer"
> is.integer(a)
[1] TRUE
```

Pretypovanie premenných

Pri akýchkoľvek výpočtoch je potrebné si uvedomovať, že môže dôjsť ku zmene typu premennej.

Príklad

```
> x<-as.integer(20)
> class(x)
[1] "integer"
> x<-x/3+1
> x
[1] 7.666667
> class(x)
[1] "numeric"
```

Dáta typu `complex`

Prostredie R umožňuje aj prácu s komplexnými číslami.

Komplexná hodnota je v R definovaná prostredníctvom imaginárnej jednotky `i`

Príklad

```
> z<-1+2i
> class(z)
[1] "complex"
```

Dáta typu `logical`

Môžu nadobúdať dve logické hodnoty `TRUE` alebo `FALSE`

Často sa vytvára prostredníctvom porovnania medzi premennými

```
1 > x<-10;y<-20
2 > z<-x<y
3 > z
4 [1] TRUE
5 > class(z)
6 [1] "logical"
```

Dáta typu `logical`

Sú pre ne definované všetky štandardné logické operácie

`&`	Logické `AND`
`|`	Logické `OR`
`!`	Negácia

Dáta typu character

Používa sa na ukladanie znakových reťazcov, reťazce sa zadávajú pomocou úvodzoviek

```
1  > x<-"facina"
2  > class(x)
3  [1] "character"
4  #Ale aj
5  > x<-as.character(3.1415926)
6  > x
7  [1] "3.1415926"
8  > class(x)
9  [1] "characcter"
```

Dáta typu character

Znakové reťazce je možné spájať pomocou funkcie `paste()`

```
1  > name<-"Donald"
2  > surname<-"Knuth"
3  > paste(name,surname)
4  [1] "Donald␣Knuth"
5  # Ak chceme
6  > paste(name,surname,sep=",")
7  [1] "Donald,Knuth"
```

Vektory

Vektor je najjednoduchšia dátová štruktúra.

Môžeme ju ju charakterizovať ako postupnosť prvkov rovnakého typu dát.

Jednotlivé hodnoty obsiahnuté vo vektore sa označujú ako komponenty.

Počet komponentov vektora sa označuje ako jeho dĺžka.

Vektory

Vektor `v` vzniká použitím funkcie `c()`.

Jeho dĺžku zistíme pomocou funkcie `length()`

```
1 > v<-c(1,3,5,7,9)
2 > length(v)
3 [1] 5
```

Vektory – aritmetika

Vektorová aritmetika je implementovaná po komponentoch.

Aritmetické operácie sa vykonávajú komponent po komponente.

- `+` pripočítanie čísla ku všetkým komponentom alebo sčítanie vektorov komponent po komponentoch
- `-` odčítanie čísla od všetkých zložiek alebo odčítanie vektorov po jednotlivých zložkách,
- `*` násobenie všetkých zložiek číslom alebo násobenie vektorov po zložkách,
- `/` delenie všetkých zložiek číslom alebo delenie vektorov zložku po zložke.

Vektory – aritmetika

```
> v<-c(1,3,5,7,9)
> u<-c(10,20,30,40,50)
> u+v
[1] 11 23 35 47 59
> u-v
[1] 9 17 25 33 41
> 5*v
[1] 5 15 25 35 45
> u*v
[1] 10 60 150 280 450
> u/5
[1] 2 4 6 8 10
> u/v
[1] 10.000000 6.666667 6.000000 5.714286 5.555556
```

Matica

Matica je dvojrozmerná tabuľka údajov rovnakého typu usporiadaná do obdĺžnikovej schémy.

Vytvoríme ju pomocou funkcie `matrix()` s nasledujúcimi argumentmi

`vector` obsahuje prvky matice,

`nrow` je celočíselná hodnota, určuje počet riadkov v matici,

`ncol` je celočíselná hodnota, určuje počet stĺpcov v matici,

`byrow` je logická hodnota, udáva, či sa má matica vyplniť po riadkoch (`byrows=TRUE`) alebo po stĺpcoch (`byrows=FALSE`), jej predvolená hodnota je `FALSE`,

`dimnames` je zoznam vektorov typu `character`, ktoré obsahujú voliteľné označenia riadkov a stĺpcov.

Matica – príklad zadavania

vfill

```
> A<-matrix(3:8,nrow=3,ncol=2,byrow=TRUE)
> A
     [,1] [,2]
[1,]    3    4
[2,]    5    6
[3,]    7    8
> B<-matrix(3:8,nrow=3,ncol=2,byrow=FALSE)
> B
     [,1] [,2]
[1,]    3    6
[2,]    4    7
[3,]    5    8
```

Matica – extrakcia submatíc

Riadky a stĺpce definujeme pomocou funkcie `c()`.

```
> C<-matrix(1:12,nrow=3)
> C
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> C[c(1,3),c(2,4)]
     [,1] [,2]
[1,]    4   10
[2,]    6   12
```

Matica – priradenie názvov

Riadkom a stĺpcom priradíme názvy pomocou funkcií `dimnames()` a `list()`

```
> dimnames(A)<-list(c("row1","row2","row3"),
+ c("col1","col2"))
> A
     col1 col2
row1    3    4
row2    5    6
row3    7    8

> A["row2","col1"]
[1] 5
```

Matica – transpozícia

Maticu môžeme transponovať pomocou funkcie `t()`

```
> B<-matrix(3:8,nrow=3,ncol=2,byrow=FALSE)
> t(B)
     [,1] [,2] [,3]
[1,]    3    4    5
[2,]    6    7    8
```

Ďalšie funkcie sú definované v balíčku `matlib`.

Matica – operácie

Definované sú po komponentoch

Je dôležité pri násobení matíc. Bežnáá operácia $*$ znamená násobenie prvkov na rovnakých pozíciách.

Štandardné násobenie matíc z lineárnej algebry je definované ako operácia %*%.

Array

Polia `array` sú zovšeobecnením maticovej dátovej štruktúry.

Ide vlastne o viac ako dvojrozmerné matice

Pole môžeme vytvoriť pomocou funkcie `array()`.

Syntax tejto funkcie je

```
name<-array(vector, dimensions,dimnames)
```

Array – vytváranie

Ilustrujme si vytvorenie poľa s rozmermi $3 \times 4 \times 3$.

Pre lepšiu orientáciu v poli si najskôr vytvoríme názvy jednotlivých dimenzií.

```
1 > dim1<-c("A1", "A2", "A3")
2 > dim2<-c("B1", "B2", "B3", "B4")
3 > dim3<-c("C1", "C2", "C3")
```

Štruktúra data frame

Data frame je najbežnejšia štruktúra na ukladanie údajov.

Umožňuje ukladať stĺpcové vektory rôznych typov údajov.

Data framy sa vytvárajú pomocou funkcie `data.frame()`, jej všeobecná syntax

```
1 > name<-data.frame(col1,col2,col3, ...)
```

Data frame – vytvorenie

Vytvoríme krátky data frame obsahujúci údaje o streľbe basketbalistov

```
1  > playerID<-c(1,2,3,4)
2  > position<-c("forward","guard","forward","center")
3  > attempted<-c(12,6,10,15)
4  > made<-c(7,4,6,12)
5  > players<-data.frame(playerID,position,attempted,made)
6  > players
7    playerID position attempted made
8  1        1  forward        12    7
9  2        2    guard         6    4
10 3        3  forward        10    6
11 4        4   center        15   12
```

Data frame – zlučovanie dát

Často potrebujeme zlúčiť údaje z dvoch alebo viacerých súborov údajov.

Používame funkciu `merge()`.

Argumenty sú názvy dvoch data framov, ktoré sa majú zlúčiť.

Tretí argument `by='''column_name'` definuje určujúcu premennú pre spojenie údajov.

Data frame – zlučovanie dát

Na demonštráciu zlučovania vytvoríme najprv nový data frame `rebounds`.

```
1  > offensive<-c(5,2,3,10)
2  > defensive<-c(6,3,8,12)
3  > rebounds<-data.frame(playerID,defensive,offensive)
4  > row.names(rebounds)<-c("Player1","Player2","Player3",
5  + "Player4")
```

Zoznam - `List`

Zoznamy predstavujú najzložitejšiu dátovú štruktúru.

Zoznamy predstavujú usporiadané kolekcie objektov.

Na vytvorenie zoznamu používame funkciu `list()`. Jej syntax je jednoduchá:

```
\list(object1,object2,...)
```

Jej argumentmi sú názvy existujúcich objektov

Zoznamy

Voliteľnou možnosťou je pomenovanie objektov vo vytvorenom zozname:

```
\list(name1=object1,name2=object2,...)
```

Vkladanie údajov z klávesnice

Najjednoduchšia metóda (ale aj časovo najnáročnejšia pre veľké vzorky)

Pracujeme v dvoch krokoch

- Vytvorte prázdny data frame s názvami a typmi premenných, ktoré chceme ukladať.
- Otvortee jednoduchý editor údajov pomocou funkcie `edit()`, ktorej argumentom je názov data framu, ktorý chceme upraviť.

Vkladanie údajov z klávesnice

Vytvoríme prázdny data frame s názvom `mydata` so štyrmi premennými: `name`, ktorá má typ `character` a tri číselné premenné `age`, `height` a `weight`.

```
1 > mydata<-data.frame(name=character(0),age=numeric(0),
2 + height=numeric(0),weight=numeric(0))
3 > mydata<-edit(mydata)
```

Poznámka

Všimnime si, že priradenie ako `numeric(0)` a `character(0)` vytvorí premennú daného typu, ale bez údajov.

Vstup údajov zo súboru `.csv`

Voliteľné argumenty funkcie `read.csv()`.

- `header` logická hodnota, udáva, či vstupný súbor obsahuje názvy premenných ako prvý riadok, predvolená hodnota `TRUE`.
- `sep` definuje znak oddeľujúci položky, predvolená hodnota je čiarka,
- `dec` definuje znak použitý v súbore pre desatinné miesta, predvolená hodnota je `.`, spomeňme tiež funkciu `read.csv2()`, ktorá používa čiarku pre desatinné čísla a bodkočiarku ako oddeľovač.
- `skip=n` určuje počet riadkov, ktoré sa majú preskočiť pred začatím čítania údajov. Táto možnosť je užitočná pre dátové tabuľky s prázdnymi riadkami alebo textovými popismi na začiatku súborov.
- `stringsAsFactors` čo je logická hodnota, ktorá udáva, či sa reťazce konvertujú na faktory, pre zabránenie konverzie, ju nastavíme na `FALSE`.
- `row.names` vektor názvov riadkov.

Zápis údajov do súboru .csv

R dokáže vytvoriť súbor csv z existujúceho data framu.

Použijeme funkciu `write.csv()`, prípadne funkciu `write.csv2()`, ktorá používa čiarku na desatinnú čiarku a bodkočiarku ako oddeľovač.

Bežná syntax

```
write.csv(object,file="file_name",...options)
```

`object` je povinný argument obsahujúci názov data framu, ktorý chceme uložiť, a `file_name` je názov (alebo úplná cesta) súboru

Zápis údajov do súboru `.csv`

Vybrané možnosti funkcie `write.csv()`

- `append` čo je logická hodnota, ktorá označuje, či sa výstup pripojí na koniec súboru. Predvolená hodnota je `FALSE` a akýkoľvek existujúci súbor s daným názvom sa prepíše.
- `sep` definuje znak oddeľovača položiek. Hodnoty v každom riadku `object` sú oddelené týmto znakom.
- `dec` reťazec, ktorý sa použije pre desatinné čiarky v číselných alebo zložených stĺpcoch, musí to byť jeden znak. Predvolená hodnota je desatinná bodka.
- `row.names` logická hodnota určujúca, či sa majú zapísať názvy riadkov `object`.

Vstup údajov zo súborov Excelu

Existuje viacero balíčkov, ktoré nám umožňujú importovať údaje priamo zo súborov Excel. Uveďme niektoré z nich:

- `xlsx`,
- `XLconnect`
- `readxl`

Excel 2007 a novšie verzie používajú formát `xlsx`, preto tu spomenieme balíček `xlsx`.

Vstup údajov zo súborov Excelu

Balíček nainštalujeme obvyklým príkazom:

```
install.packages("xlsx")
```

Ak ho chceme použiť v aktuálnom pracovnom priestore, načítame ho štandardným spôsobom:

```
library("xlsx")
```

Vstup údajov zo súborov Excelu

Tento balíček poskytuje dve funkcie pre načítanie obsahu pracovného hárku Excelu do R data.frame: `read.xlsx()` a `read.xlsx2()`.

Rozdiel medzi týmito dvoma funkciami je:

- `read.xlsx()` zachováva typ údajov, typ premennej zodpovedá každému stĺpcu v pracovnom hárku, ale je pomalá pre veľké súbory údajov (pracovný hárok s viac ako 100 000 bunkami).
- `read.xlsx2()` je rýchlejší pri veľkých súboroch.

Vstup údajov zo súborov Excelu

Obe funkcie majú podobnú syntax:

```
read.xlsx(file, sheetIndex, header=TRUE, colClasses=NA)
read.xlsx2(file, sheetIndex, header=TRUE, colClasses="character")
```

Ich argumenty majú nasledujúci význam:

- `file` je názov súboru, ktorý obsahuje tabuľku. Ak sa súbor nenachádza v pracovnom adresári, musí byť zadaný s úplnou cestou.
- `sheetIndex` číslo označujúce index listu, ktorý sa má načítať. Môžeme ho nahradiť argumentom `sheetname` zadaným ako reťazec znakov s názvom listu.
- `header` logická hodnota. Ak `header=TRUE`, použije sa prvý riadok ako pomenovanie premenných.
- `colClasses` znakový vektor, ktorý predstavuje triedu každého stĺpca.
- `startRow`, `endRow` čísla určujúce index počiatočného riadku a posledného riadku, ktorý sa má načítať.

Zápis údajov do súborov Excelu

Balíček `xlsx` poskytuje dve funkcie na zápis `write.xlsx()` a `write.xlsx2()`

Všeobecná syntax

```
write.xlsx(x, file, sheetName="Sheet1", col.names=TRUE,
row.names=TRUE, append=FALSE)

write.xlsx2(x, file, sheetName="Sheet1",col.names=TRUE,
row.names=TRUE, append=FALSE)
```

Zápis údajov do súborov Excelu

Ich argumenty majú nasledujúci význam:

- `x` data frame, ktorý sa má zapísať do zošita.
- `file` názov (resp. cesta) výstupného súboru.
- `sheetName` reťazec znakov s názvom listu.
- `col.names` logická hodnota, udáva, či sa majú do súboru zapísať názvy stĺpcov `x`.
- `row.names` logická hodnota, udáva, či sa majú do súboru zapísať názvy riadkov `x`.
- `append` logická hodnota, udáva, či sa má `x` pripojiť k existujúcemu súboru, ak je `FALSE`, prepíše existujúci súbor s rovnakou cestou.

Čítanie údajov zo súborov JSON

JSON (JavaScript Object Notation) je jednoduchý formát na výmenu údajov

Ak chceme načítať súbory JSON do R, musíme najprv nainštalovať alebo načítať balíček `rjson`.

Môžeme použiť funkciu `fromJSON()`

Použitie závisí od umiestnenia súboru `.json`

```
data<-fromJSON(file = "filename.json")
data<-fromJSON(file ="URL na súbor json")
```

V oboch prípadoch je objekt `data` uložený ako zoznam. Na ďalšiu analýzu môžeme údaje konvertovať pomocou funkcie `as.data.frame()`.

Zápis údajov do súborov JSONI

Musí sa vykonať v dvoch krokoch.

V prvom kroku musíme pripraviť objekt JSON a v druhom kroku ho zapíšeme do súboru.

Na vytvorenie objektu JSON použijeme funkciu `toJSON()`:

```
dataJSON<-toJSON(data)
```

Potom použijeme funkciu `write()`

```
write(dataJSON, "filename.json")
```

Štatistika a programovanie v R

III. Rozdelenia pravdepodobnosti v R

Implementované funkcie

R umožňuje pracovať s veľkým množstvom rozdelení pravdepodobnosti

Pre každé z rozdelení, s ktorými R pracuje, sú implementované štyri funkcie

Ich názvy sú zložené z koreňového názvu a predpony:

- `p` pre distribučnú funkciu,
- `d` pre hustotu alebo pravdepodobnostnú funkciu,
- `q` pre kvantilovú funkciu, inverznú ku distribučnej funkcii,
- `r` náhodná premenná s určeným rozdelením (generátor náhodných hodnôt).

Diskrétne rozdelenie

Pravdepodobnosti sú určené zoznamom pravdepodobností diskrétnych výsledkov, známym ako pravdepodobnostná funkcia

Ak označíme množinu všetkých možných hodnôt diskrétnej náhodnej premennej X ako H , môžeme zaviesť pravdepodobnostnú funkciu $p(x)$ podľa vzorca

$$p(x) = \mathbb{P}(X = x), \; x \in H. \tag{1}$$

Diskrétne rozdelenie

Niektoré z nich spomenieme:

- Bernoulliho rozdelenie,
- binomické rozdelenie,
- geometrické rozdelenie,
- hypergeometrické rozdelenie,
- negatívne binomické rozdelenie,
- Poissonovo rozdelenie.

Bernoulliho rozdelenie

Najjednoduchšie rozdelenie pravdepodobnosti

Ide o rozdelenie, ktoré má len dve možné hodnoty.

Možno ho interpretovať ako indikátorovú premennú, či nejaká náhodná udalosť nastane alebo nie.

Nech A je náhodná udalosť, $\mathbb{P}(A) = p$ a náhodná premenná $X = 1$ ak nastane A a $X = 0$ v opačnom prípade. Potom má pravdepodobnostná funkcia tvar

$$p(x) = p^x(1-p)^{1-x} \text{ pre } x = 0 \text{ alebo } 1$$

Je implementovaná v balíčku `Rlab` ako `bern` s parametrom `prob`.

Bernoulliho rozdelenie

Máme k dispozícii štyri funkcie:

- `rbern(n,prob)`, kde `n` je počet pozorovaní a `prob` je pravdepodobnosť výskytu náhodnej udalosti A (úspech v pokuse). Generuje vektor `0` a `1` vybraný z Bernoulliho rozdelenia s danou pravdepodobnosťou.
- `pbern(q, prob, lower.tail = TRUE, log.p = FALSE)`
- `dbern(x, prob, log = FALSE)`
- `qbern(p, prob, lower.tail = TRUE, log.p = FALSE)`

Binomické rozdelenie

Binomické rozdelenie je diskrétne rozdelenie, ktoré opisuje počet úspechov v sérii pokusov s dvoma možnými výsledkami

Formálne nech p označuje pravdepodobnosť úspechu v jednom pokuse, n počet nezávislých pokusov a x počet úspechov v postupnosti n nezávislých pokusov. Náhodná premenná X sa riadi binomickým rozdelením, ak jej pravdepodobnostná funkcia má tvar:

$$\mathbb{P}(X = x) = \binom{n}{x} p^x q^{n-x}, \quad x = 0, 1, 2, \ldots, n, \tag{2}$$

kde $q + p = 1$.

Binomické rozdelenie

Na prácu s binomickým rozdelením sú implementované 4 funkcie:

- `rbinom(n,prob)`, kde `n` je počet pozorovaní, `p` je pravdepodobnosť úspechu. Táto funkcia generuje n náhodných premenných s danou pravdepodobnosťou.
- `pbinom(x, n, k)`, kde `n` je celkový počet pokusov, `p` je pravdepodobnosť úspechu, `x` je hodnota, pre ktorú je potrebné zistiť pravdepodobnosť.
- `dbinom(x, n, p)`, kde `n` je celkový počet pokusov, `p` je pravdepodobnosť úspechu, `x` je hodnota, pre ktorú sa má určiť pravdepodobnosť.
- `qbinom(prob, n, p)`, kde `prob` je pravdepodobnosť, `n` je celkový počet pokusov a `p` je pravdepodobnosť úspechu v jednom pokuse. Táto funkcia sa používa na určenie n-tého kvantilu, t. j. ak je dané $P(X \leq k)$, určí k.

Hypergeometrické rozdelenie

Hypergeometrické rozdelenie je diskrétne rozdelenie pravdepodobnosti, ktoré opisuje pravdepodobnosť x úspechov pri n výberoch bez náhrady z konečnej populácie veľkosti N, ktorá obsahuje presne K objektov s touto vlastnosťou.

Označme ako N veľkosť populácie, K počet úspešných stavov v populácii, n počet výberov a x počet pozorovaných úspešných výsledkov. Náhodná premenná X sa riadi hypergeometrickým rozdelením, ak jej pravdepodobnostná funkcia má tvar

$$\mathbb{P}(X = x) = \frac{\binom{K}{x}\binom{N-K}{n-x}}{\binom{N}{n}} \quad x = 0, 1, 2, \ldots, n. \tag{3}$$

Hypergeometrické rozdelenie

Štyri funkcie na prácu s hypergeometrickým rozdelením v R:

- `rhyper(N, m, n, k)`, vo všeobecnosti sa vzťahuje na funkciu generovania náhodných čísel pri zadaní parametrov a veľkosti vzorky,
- `phyper(x, m, n, k)` definuje distribučnú funkciu hypergeometrického rozdelenia,
- `dhyper(x, m, n, k)` definuje pravdepodobnostnú funkciu hypergeometrického rozdelenia,
- `qhyper(N, m, n, k)` je kvantilová funkcia hypergeometrického rozdelenia, ktorá sa používa na určenie postupnosti pravdepodobností medzi 0 a 1.

Tu `x` predstavuje súbor hodnôt, `m` veľkosť populácie, `n` počet vybraných vzoriek, `k` počet položiek v populácii a `N` hypergeometricky rozdelené hodnoty.

Negatívne binomické rozdelenie

Negatívne binomické rozdelenie je diskrétne rozdelenie pravdepodobnosti, ktoré modeluje počet úspechov v postupnosti nezávislých a identicky rozdelených Bernoulliho pokusov pred výskytom určitého (nenáhodného) počtu zlyhaní (označených n).

Ak označíme počet úspechov x a pravdepodobnosť úspechu ako p, pravdepodobnostná funkcia negatívneho binomického rozdelenia má tvar:

$$\mathbb{P}(X = x) = \binom{n + x - 1}{n - 1} p^x q^n, \quad x = 0, 1, 2, \ldots \tag{4}$$

kde $q + p = 1$, $p > 0, q > 0$.

Negatívne binomické rozdelenie

Štyri funkcie na prácu so záporným binomickým rozdelením v R:

- `rnbinom(N,n,prob)`, kde `n` je počet pokusov, `N` je veľkosť vzorky, `prob` je pravdepodobnosť úspechu. Táto funkcia generuje N náhodných premenných s danou pravdepodobnosťou.
- `pnbinom(x, n, p)`, sa používa na výpočet hodnoty distribučnej funkcie negatívneho binomického rozdelenia. Tu `x` je počet zlyhaní pred `n`-tým úspechom a `p` je pravdepodobnosť úspechu.
- `dnbinom(x, n, p)`, je pravdepodobnosť `x` zlyhaní pred `n`-tým úspechom (všimnite si rozdiel), keď pravdepodobnosť úspechu je `p`.
- `qnbinom(x, n, p)` sa používa na výpočet hodnoty kvantilovej funkcie negatívneho binomického rozdelenia. Tu `x` je vektor požadovaných hladín kvantilov, `n` je celkový počet pokusov a `p` je pravdepodobnosť úspechu v jednom pokuse.

Poissonovo rozdelenie

Poissonovo rozdelenie je diskrétne rozdelenie pravdepodobnosti, ktoré vyjadruje pravdepodobnosť výskytu daného počtu udalostí v pevne stanovenom časovom alebo priestorovom intervale, ak sa tieto udalosti vyskytujú so známou konštantnou strednou rýchlosťou a nezávisle od času, ktorý uplynul od poslednej udalosti.

Ak sa náhodná premenná X riadi Poissonovým rozdelením s parametrom $\lambda > 0$ (priemerný počet udalostí), jej pravdepodobnostná funkcia má tvar

$$\mathbb{P}(X = x) = \frac{\mathrm{e}^{-\lambda}\lambda^x}{x!}, \quad x = 0, 1, 2, \ldots \tag{5}$$

Poissonovo rozdelenie

Štyri funkcie na prácu s Poissonovým rozdelením v R:

- `dpois(x,l)` vypočíta hodnotu pravdepodobnostnej funkcie $\mathbb{P}(X = x)$ Poissonovho rozdelenia s parametrom λ implementovaným ako argument `l`.
- `ppois(x,l)` vypočíta distribučnú funkciu náhodnej premennej, ktorá sa riadi Poissonovým rozdelením. Určuje pravdepodobnosť $\mathbb{P}(X \leq x)$, argument `l` je parameter rozdelenia. Uvedením ďalšieho argumentu `lower.tail=FALSE` dostaneme pravdepodobnosť $\mathbb{P}(X > x)$.
- `rpois(k,l)` sa používa na generovanie náhodných čísel z daného Poissonovho rozdelenia, `k` je počet potrebných náhodných čísel a `l` je parameter rozdelenia.
- `qpois(q,l)` sa používa na generovanie kvantilov daného Poissonovho rozdelenia, `q` je vektor potrebných kvantilových hladín a `l` je parameter rozdelenia.

Spojité rozdelenie

Je to rozdelenie pravdepodobnosti, ktorého nosičom je nespočítateľná množina.

Rozdelenie je jednoznačne charakterizované kumulatívnou distribučnou funkciou

$$F(x) = \mathbb{P}(X \leq x) = \int_{-\infty}^{x} f(t)\,\mathrm{d}t.$$

Spojité rozdelenie

Uveďme niektoré z nich:

- rovnomerné rozdelenie,
- exponenciálne rozdelenie,
- normálne rozdelenie,
- Studentovo rozdelenie t,
- Chi kvadrát rozdelenie,
- Fisherovo F rozdelenie.

V R je implementovaných mnoho iných.

Spojité rozdelenie

Uveďme niektoré z nich:

- rovnomerné rozdelenie,
- exponenciálne rozdelenie,
- normálne rozdelenie,
- Studentovo rozdelenie t,
- Chi kvadrát rozdelenie,
- Fisherovo F rozdelenie.

V R je implementovaných mnoho iných.

Budeme sa zaoberať, tromi „modrými“ rozdeleniami.

Rovnomerné rozdelenie

Spojité rovnomerné rozdelenie opisuje experiment, v ktorom je možný ľubovoľný výsledok, ktorý leží medzi určitými hranicami.

Presnejšie povedané, náhodná premenná X sa riadi rovnomerným rozdelením s parametrami a a b, $a < b$, ak má jej funkcia hustoty tvar:

$$f(x) = \begin{cases} \frac{1}{b-a} & x \in \langle a; b \rangle \\ 0 & \text{inde} \end{cases} \tag{6}$$

Rovnomerné rozdelenie

Štyri funkcie na prácu s rovnomerným rozdelením v R:

- `dunif()`, ktorá definuje funkciu hustoty, jej argumenty sú vektor `x` a parametre `min` a `max` rozdelenia,
- `punif()`, ktorá definuje distribučnú funkciu, jeho argumenty sú vektor `x` a parametre `min` a `max` rozdelenia,
- `qunif()`, ktorá poskytuje kvantilovú funkciu, jej argumenty sú kvantily `q` a parametre `min` a `max` rozdelenia,
- `runif()`, ktorá generuje náhodné hodnoty premennej, jej argumenty sú veľkosť vzorky `n` a parametre `min` a `max` rozdelenia.

Exponenciálne rozdelenie

Exponenciálne rozdelenie opisuje čakaciu dobu medzi udalosťami v Poissonovom procese, t. j. procese, v ktorom sa udalosti vyskytujú nepretržite a nezávisle s konštantnou priemernou frekvenciou.

Náhodná premenná X sa riadi exponenciálnym rozdelením s parametrom $\lambda > 0$, (zvyčajne sa uvádza ako frekvencia), ak má jej funkcia hustoty tvar:

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{7}$$

Keď sa parameter λ interpretuje ako frekvencia, priemerný čas čakania je $\frac{1}{\lambda}$.

Exponenciálne rozdelenie

Štyri funkcie na prácu s exponenciálnym rozdelením v R:

- `dexp()`, ktorá poskytuje funkciu hustoty, jej argumenty sú vektor `x` a parameter `rate` rozdelenia,
- `pexp()`, ktorá predstavuje distribučnú funkciu, jej argumenty sú vektor `x` a parameter `rate` rozdelenia,
- `qexp()`, ktorá určuje kvantilovú funkciu, jej argumenty sú kvantily `q` a parameter `rate` rozdelenia,
- `rexp()`, ktorá generuje náhodné hodnoty premennej, jej argumenty sú veľkosť vzorky `n` a parameter `rate` rozdelenia.

Normálne rozdelenie

Normálne (alebo Gaussovo) rozdelenie je typ spojitého rozdelenia pravdepodobnosti, jeho funkcia hustoty je

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}. \tag{8}$$

Parameter μ je stredná hodnota rozdelenia (a tiež jeho medián a modus), parameter σ je jeho štandardná odchýlka.

Normálne rozdelenie je dôležité kvôli centrálnej limitnej vete, ktorá hovorí, že populácia všetkých možných vzoriek veľkosti n z populácie so strednou hodnotou μ a rozptylom σ^2 sa blíži k normálnemu rozdeleniu so strednou hodnotou μ a $\frac{\sigma^2}{n}$, keď sa n blíži k nekonečnu.

Normálne rozdelenie

Štyri funkcie na prácu s normálnym rozdelením v R:

- `dnormf()`, ktorá predstavuje funkciu hustoty, jej argumenty sú vektor `x` a parametre `mean` a `sd` rozdelenia,
- `pnorm()`, ktorá predstavuje distribučnú funkciu, jej argumenty sú vektor `x` a parametre `mean` a `sd` rozdelenia,
- `qnorm()`, ktorá predtavuje kvantilovú funkciu, jej argumenty sú kvantily `q` a parametre `mean` a `sd` rozdelenia,
- `rnorm()`, ktorá generuje náhodné hodnoty premennej, jej argumenty sú veľkosť vzorky `n` a parametre `mean` a `sd` rozdelenia.

Štatistika a programovanie v R

IV. Programovanie v R

Funkcie

Takmer všetky činnosti v R sa vykonávajú prostredníctvom funkcií.

Je v ňom implementovaní bohatá škála vstavaných funkcií

Ďalšie funkcie môže definovať používateľ

Vstavané funkcie môžeme rozdeliť na

- matematické funkcie,
- reťazcové funkcie,
- špecializované štatistické a pravdepodobnostné funkcie,
- iné užitočné funkcie.

Matematické funkcie

Niektoré z nich sme už spomenuli v lekcii 1.

Tu uvedieme niektoré ďalšie podrobnosti

Logaritmická funkcia `log()` vypočíta ako predvolenú hodnotu prirodzený logaritmus.

Ak chceme získať logaritmus s ľubovoľným základom, musíme deklarovať argument `base` funkcie `log()`.

```
1 > log(4)
2 [1] 1.386294
3 > log(4,base=2)
4 [1] 2
```

Reťazcové funkcie

Funkcia `nchar()` určuje veľkosť jednotlivých prvkov vektora znakov

```
1  > z<-c("yellow","black","white")
2  > nchar(z)
3  [1] 6 5 5
4  > str<-"This␣is␣a␣long␣string"
5  > nchar(str)
6  [1] 21
```

Elementárne štatistické funkcie

- `mean()` Priemer vzorky.
- `median()` Medián vzorky.
- `sd()` Štandardná odchýlka.
- `var()` Výberový rozptyl .
- `mad()` Absolútna odchýlka mediánu.
- `quantile()` Výberové kvantily, implicitne kvartily.
- `range()` Rozsah hodnôt.
- `sum()` Súčet prvkov vektora.
- `min()` Minimum.
- `max()` Maximum.

Elementárne štatistické funkcie – `mean()` voliteľné argumenty

`trim`, ktorý udáva percento najvyšších a najnižších hodnôt, ktoré sa vynechajú z výpočtu, a tak vracia orezaný priemer.

Druhý nepovinný argument `na.rm` je logická hodnota, ktorá udáva, či sa majú hodnoty `NA` pred pokračovaním výpočtu odstrániť.

```
1  > x<-c(1,3,5,10,12)
2  > mean(x)
3  [1] 6.2
4  > mean(x,trim=0.2)
5  [1] 6
```

```
1  > x<-c(1,5,2,12,NA,3,6)
2  > mean(x)
3  [1] NA
4  > mean(x,na.rm=TRUE)
5  [1] 4.833333
6  > mean(x,na.rm=TRUE,trim=0.17)
7  [1] 4
```

Elementárne štatistické funkcie – `mad()`

Absolútna mediánová odchýlka je robustná miera variability jednorozmernej vzorky kvantitatívnych údajov.

Pre vzorku $X_1, \ldots, X_n$ je definovaná vzorcom:

$$\mathrm{MAD}(X) = \mathrm{median}\{|X_i - \overline{X}|\}$$

```
1  > mad(delay)
2  [1] 13.3434
```

Užitočné funkcie – seq()

Funkcia seq() generuje postupnosť čísel začínajúcu hodnotou from a končí hodnotou to. Posledný argument by definuje krok postupnosti.

```
> seq(10)
 [1] 1 2 3 4 5 6 7 8 9 10
> seq(5,15)
 [1] 5 6 7 8 9 10 11 12 13 14 15
> seq(5,15,2)
[1] 5 7 9 11 13 15
```

Užitočné funkcie – rep()

Funkcia rep() má dva argumenty, vektor x, ktorý sa má opakovať, a počet n cyklov opakovania

```
> rep(1,10)
 [1] 1 1 1 1 1 1 1 1 1 1
> rep(c(1,3),4)
[1] 1 3 1 3 1 3 1 3
> rep("hello",3)
[1] "hello" "hello" "hello"
```

Užitočné funkcie – `sort()` a `order()`

Funkcie `sort()` a `order()` sú spojené s usporiadaním prvkov vektorax

`sort()` poskytuje vzostupne zoradené hodnoty, zatiaľ čo `order()` poskytuje indexy zoradených zložiek v pôvodnom vektore.

```
1  > x<-c(5,2,10,3,7,8)
2  > sort(x)
3  [1]  2  3  5  7  8 10
4  > order(x)
5  [1]  2  4  1  5  6  3
```

Užitočné funkcie – `rev()`

Dáva vektor `x` v opačnom poradí

```
> rev(x)
[1]  8  7  3 10  2  5
> rev(sort(x))
[1] 10  8  7  5  3  2
```

Podmienené príkazy – príkaz `if`

`if()` príkaz vykonáva operácie na základe jednoduchej podmienky

`if (podmienka) {príkaz, ktorý sa vykoná, ak podmienka platí}`

Viac ako jeden príkaz musí byť v zátvorkách

```
1  > x<-5
2  > if(x%%2){print("Odd number")}
3  [1] "Odd number"
4  > x<-6
5  > if(x%%2){print("Odd number")}
6  >
```

Podmienené príkazy – príkaz `ifelse`

Na získanie očakávaného výsledku musíme použiť príkaz `ifelse` so všeobecnou syntaxou:

```
ifelse(podmienka, výraz1, výraz2)
```

```
> ifelse(x>3,2*x,x)
[1] 10  8  3  2  1
```

Podmienené príkazy – switch

`switch()` testuje výraz voči prvkom zoznamu. Každá hodnota v zozname sa nazýva case

Syntax funkcie `switch()`:

```
switch (expression, list)
```

```
1 > x<-10
2 > switch(x%%2+1, "even", "odd")
3 [1] "even"
4 > x<-9
5 > switch(x%%2+1, "even", "odd")
6 [1] "odd"
```

Podmienené príkazy – `switch`

Ak je výraz reťazec znakov, switch() vráti hodnotu na základe názvu prvku.

```
> x <- "a"
> switch(x, "a"="apple", "b"="banana", "c"="cherry")
[1] "apple"
> x <- "c"
> switch(x, "a"="apple", "b"="banana", "c"="cherry")
[1] "cherry"
```

Cyklus – `for`

`for` cyklus nám umožňuje pevný počet opakovaní príkazu alebo bloku príkazov

Všeobecná syntax cyklu `for` je takáto:

```
for (val in sequence)
{
príkaz
}
```

kde `sequence` je vektor a `val` nadobúda počas cyklu každú z jej hodnôt.

Užívateľom definovaná funkcia

Všeobecná štruktúra funkcie je

```
myfunction_name <- function(arg1, arg2, ... ){
príkazy
return(objekt)
}
```

Užívateľom definovaná funkcia

Jednotlivé zložky funkcie sú:

- **Názov funkcie**, čo je skutočný názov funkcie. Je uložený v prostredí R ako objekt s týmto názvom.
- **Argumenty**, ktoré sú zástupné znaky. Pri volaní funkcie odovzdávame hodnoty argumentov. Argumenty sú nepovinné, to znamená, že funkcia nemusí obsahovať žiadne argumenty. Aj argumenty môžu mať predvolené hodnoty.
- **Telo funkcie**, ktoré obsahuje súbor príkazov, ktoré definujú, čo funkcia robí. Telo funkcie sa nachádza vo vnútri zložených zátvoriek {}.
- **Návratová hodnota**, ktorá je posledným výrazom v tele funkcie, ktorý sa má vyhodnotiť.

Spustenie skriptov v R

Skript R je jednoducho textový súbor obsahujúci (takmer) rovnaké príkazy, aké by ste zadali

Môžeme ho vytvoriť v akomkoľvek jednoduchom textovom editore a uložiť s príponou `.R`

Na spustenie skriptu v Linuxe existujú v podstate dva príkazy

```
Rscript filename.R
```

ktorý je uprednostňovaný. Starší príkaz je

```
R CMD BATCH filename.R
```

Štatistika a programovanie v R

V. Základy grafiky R

Bodové grafy

Bodový (tiež korelačný) graf zobrazuje hodnoty dvoch rôznych číselných premenných.

Poloha každého bodu zodpovedá jednotlivým dátovým hodnotám na vodorovnej a zvislej osi.

Najčastejšie aplikácie a použitia bodových grafov sú:

1. Demonštrácia vzťahu medzi dvoma premennými.
2. Identifikácia korelačných vzťahov.
3. Identifikácia dátových modelov.

Bodové grafy

Vytvoríme ho jednoducho pomocou funkcie `plot()`.

Pri najjednoduchšom použití má funkcia dva argumenty `x` a `y`

Tieto premenné sú vektory, ktoré obsahujú hodnoty, ktoré chceme vykresliť.

Dĺžka vektorov musí byť rovnaká.

Ako uložiť obrázok

Alternatívne môžeme presmerovať výstup z obrazovky do súboru.

Môžeme použiť funkcie

`pdf()`	Vector `pdf` formát, najlepšia voľba pri použití s `pdflatex`
`svg()`	Vektorový `svg` formát, ľahko sa mení veľkosť.
`postscript()`	Vektorový formát postscript `ps`, ľahko meniteľná veľkosť.
`png()`	Bitmapový formát s vysokým rozlíšením, veľkosť sa nemení bezstrát.
`jpeg()`	Komprimovaný bitmapový formát, nemení bezstratovo veľkosť.
`bmp()`	Bitmapový formát s vysokým rozlíšením, nemení bezstratovo veľkosť.
`tiff()`	Bitmapový formát s vysokým rozlíšením, nemení bezstratovo veľkosť.

Možnosti uloženia grafov

`filename`	Názov uloženého súboru, v prípade potreby s úplnou cestou.
`width`	Šírka výsledného grafu, predvolená hodnota 7 in.
`height`	Výška výsledného grafu, predvolená hodnota 7 in.
`res`	rozlíšenie obrázka, platí pre bitmapové formáty, predvolené 72 dpi.
`units`	Merné jednotky.
`bg`	Farba pozadia.
`fg`	Farba popredia.
`family`	Použité písmo (predvolené Helvetica).

Modifikácia grafu – značky bodov

Značka bodov je daná hodnotou argumentu `pch` funkcie `plot()`

Možné hodnoty

□ pch=0	○ pch=1	△ pch=2	+ pch=3	× pch=4
◇ pch=5	▽ pch=6	⊠ pch=7	✳ pch=8	⌖ pch=9
⊕ pch=10	✡ pch=11	⊞ pch=12	⊗ pch=13	◰ pch=14
■ pch=15	● pch=16	▲ pch=17	◆ pch=18	● pch=19
• pch=20	○ pch=21	□ pch=22	◇ pch=23	△ pch=24

Modifikácia grafu – typ spojnice

Typ spojnice sa nastavuje pomocou argumentu `type` funkcie `plot()`

Možné hodnoty

`p`	Bodový graf, predvolená hodnota.
`l`	Spojitá čiara.
`b`	Spojitá čiara s bodmi.
`c`	Časti spojitých čiar s vynechanými bodmi.
`o`	Časti súvislých čiar s prekreslenými bodmi.
`h`	Graf podobný histogramu.
`s`	Schodovitý graf.

Modifikácia grafu – štýl spojnice

Štýl čiary sa nastavuje pomocou argumentu `lty` funkcie `plot()`

Možné hodnoty

1	Plná čiara (predvolené).	2	Čiarkovaná čiara.
3	Bodkovaná čiara.	4	Bodko-čiarkovaná čiara.
5	Dlhé čiarky.	6	Dlhá a krátka dvojitá prerušovaná čiara.

Šírka čiary sa nastavuje pomocou argumentu `lwd` funkcie `plot()`

Modifikácia grafu – farbenie

Nastavenie farieb môžeme vykonať pomocou

- mena farby, napríklad `col=red`
- číslom farby, napríklad `col=636`
- podľa hexadecimálneho kódu (v režime RGB), napríklad `col="#FFCC00"`

Zoznam dostupných farieb získame ako odpoveď funkcie `colors()`.

Modifikácia Grafu – farbenie

Ďalšie možnosti vyfarbenia sú

`col.axis`	Farba popisu osí.	`col.lab`	Farba označení osí.
`col.main`	Farba hlavného nadpisu.	`col.sub`	Farba podnadpisu.
`bg`	Farba výplne znakov.	`fg`	Farba podkladu.

Modifikácia grafu – farbenie

Farby ako vektory

Hodnotu argumentu `col` môžeme nastaviť ako vektor.

Farby z vektora sa pravidelne menia

Môžeme tiež použiť funkciu `rainbow()` s preddefinovanou postupnosťou farieb.

Modifikácia grafu – názvy a titulky

Základné kresliace funkcie v R obsahujú argument s názvom `main`, ktorý umožňuje pridať do grafu nadpis.

Pomocou argumentu `sub` je možné pridať aj podnadpis, ktorý bude umiestnený pod grafom.

Alternatívnym spôsobom, ako pridať nadpis a podnadpis do grafu, je použitie funkcie `title()`.

Modifikácia grafu – pridanie textu do grafu

Do vykresleného grafu môžeme pridať akékoľvek texty pomocou funkcií `text()` a `mtext()`.

Funkcia `text()` umiestni daný text na ľubovoľné miesto v kresliacej oblasti , funkcia `mtext()` umiestni text na okraje.

Funkcia `text()` má dva ďalšie argumenty:

- `location` definuje súradnice `x` a `y`, kde bude text umiestnený. Súradnice musia byť uvedené ako prvé dva argumenty funkcie.
- `pos` definuje pozíciu podľa aktuálneho miesta, 1=dole, 2=vľavo, 3=hore a 4=vpravo. Definovanie pozície ako `locator(1)` umožňuje umiestnenie textu pomocou myši.

Modifikácia grafu – prispôsobenie osí

Ak chceme odstrániť rámček grafu, nastavíme vo vnútri kresliacej funkcie možnosť `axes=FALSE`.

Nové osi pridáme pomocou funkcie `axes()`.

Argument funkcie `axis()` definuje stranu grafu, na ktorú bude pridaná os.

Ako zvyčajne, čísla definujú strany 1=spodná, 2=ľavá, 3=horná a 4=pravá.

Vyskúšajme

```
1 > plot(sort(x),y[order(x)],pch=17,type="b",col=30,axes=FALSE)
2 > axis(1)
3 > axis(2)
```

Ďalšie prispôsobenie osí

Môžeme tiež:

- nastaviť počet značiek so zadanými hodnotami začiatku a konca,
- upraviť dĺžku a orientáciu značiek,
- otáčať popisy značiek,
- prispôsobiť popisy značiek,
- odstrániť značky,
- pridať menšie značky pomocou `Hmisc` .

Ďalšie prispôsobenie osí – dĺžka a orientácia značiek

Argument `tck` umožňuje upraviť dĺžku a orientáciu deliacich značiek.

Jeho kladná hodnota nastavuje značky do vnútai kresliacej oblasti, kým záporné hodnoty definujú značky von z kresliacej oblasti. Čím väčšia je absolútna hodnota, tým dlhšie sú značky. predvolená hodnota je `tck=-0,05`.

Otáčanie je povolené pomocou argumentu`las`, ktorý môže nadobúdať jednu zo štyroch hodnôt:

- `las=0` štítky sú rovnobežné s osou (predvolené),
- `las=1` všetky štítky sú vodorovné,
- `las=2` štítky sú kolmé na os,
- `las=3` všetky štítky sú vertikálne.

Rozsah osí a prispôsobenie

Rozsah hodnôt pre osi môžeme definovať pomocou voliteľných argumentov `xlim` a `ylim` funkcie `plot()`

Hranice sa zadávajú ako vektory v tvare `c(start,end)`

Osi môžeme tiež transformovať do logaritmickej mierky nastavením argumentu `log` tak, aby sa rovnal osi, ktorú plánujeme prispôsobiť.

`log="x "` nastaví logaritmickú stupnicu na os `x`,
`log="y "` nastaví logaritmickú mierku na os `y` a
`log="xy "` transformuje obe osi do logaritmickej stupnice.

Dve duálne vertikálne osi

Príklad.
Do toho istého grafu chceme zakresliť dve charakteristiky zdravotného stavu pacientov, teplotu a krvný tlak.

Dve duálne vertikálne osi

Príklad.
Do toho istého grafu chceme zakresliť dve charakteristiky zdravotného stavu pacientov, teplotu a krvný tlak.

Máme údaje o 100 pacientoch uložené v premenných `y` a `z`, pričom premenná `x` obsahuje postupnosť identifikátorov pacientov, čísla od 1 do 100.

Kreslenie kriviek

Jednou z mnohých praktických funkcií v R je `curve()`.

Je to malá šikovná funkcia, ktorá umožňuje vykresľovanie kriviek, napr. grafy funkcií.

Funkcia `curve()` prijíma ako prvý argument výraz v syntaxi R.

Napríklad

```
curve(x^2)
curve(x^2,xlim=c(-2,2),col="red",lwd=2)
```

Zobrazenie dvoch alebo viacerých kriviek do jedného grafu

Používame funkciu `curve()` s argumentom `add=TRUE`.

Napríklad

```
curve(x^2)
curve(sqrt(x),col="red",lwd=2,add=TRUE)
```

Pridanie legendy

Funkcia `legend()` umožňuje pridať legendu ku grafom v R.

Niektoré z argumentov:

- `x,y` pozícia v kresliacej oblasti definovaná súradnicami v grafe,
- `legend` vektor reťazcov pre popis v legende,
- `col` vektor farieb použitých v grafe,
- `pch` vektor tvarov značiek použitých v grafe,
- `lty` vektor typov čiar použitých v grafe,
- `ncol` počet stĺpcov použitých v legende, predvolená hodnota je jeden stĺpec.

Stĺpcové grafy

Stĺpcový graf zobrazuje kategoriálne údaje pomocou obdĺžnikových stĺpcov s výškou alebo dĺžkou úmernou hodnotám, ktoré predstavujú.

Na vytvorenie stĺpcových grafov používa R funkciu

```
barplot(H,xlab,ylab,title, names.arg,col)
```

Parametre použité vo funkcii sú nasledovné:

- `H` je vektor alebo matica obsahujúca číselné hodnoty použité v stĺpcovom grafe,
- `xlab` je označenie osi x,
- `ylabj` e označenie osi y,
- `title` je nadpis stĺpcového grafu,
- `names.arg` je vektor popisov, ktoré sa zobrazujú pod každým stĺpcom,
- `col` sa používa na priradenie farieb stĺpcom v grafe.

Stĺpcové grafy – vyfarbenie a označenia stĺpcov

Na priradenie názvov stĺpcom použijeme parameter `names.arg` stĺpcového grafu

Ďalej definujeme hodnoty parametrov

- `xlab` a `ylab` pre názvy osí,
- `col` a `border` na vyfarbenie stĺpcov a
- `main` na definovanie nadpisu grafu

Je to podobné ako v prípade funkcie `plot()`.

Histogramy

Histogram je znázornenie približného rozdelenia číselných údajov.

Zobrazujú frekvencie hodnôt premennej rozdelené do intervalov.

Histogram je podobný stĺpcovému grafu, ale s tým rozdielom, že agreguje hodnoty do súvislých intervalov.

Histogramy poskytujú približný obraz o hustote základného rozdelenia údajov

Histogramy

Histogram možno vytvoriť pomocou funkcie `hist()` v R

```
barplot(H,xlab,ylab,title, names.arg,col)
```

Parametre použité vo funkcii sú nasledovné:

- `data` je vektor obsahujúci číselné hodnoty použité v histograme,
- `main` označuje názov grafu,
- `col` sa používa na nastavenie farby stĺpcov,
- `border` sa používa na nastavenie farby okraja každého stĺpca,
- `xlab` sa používa na popis osi x,
- `xlim` sa používa na zadanie rozsahu hodnôt na osi x,
- `ylim` sa používa na určenie rozsahu hodnôt na osi y,
- `breaks` sa používa na uvedenie šírky jednotlivých stĺpcov.

Koláčové grafy

Koláčový graf je graf pre jednu kategoriálnu premennú a je alternatívou stĺpcového grafu.

Koláčový graf (alebo kruhový graf) je diagram v tvare kruhu, ktorý je rozdelený na výseky zodpovedajúcich pomerom zobrazovaných hodnôt.

V koláčovom grafe je dĺžka oblúka každého výseku (a teda aj jeho stredový uhol a plocha) úmerná veličine, ktorú predstavuje.

Koláčové grafy

Základná syntax pre vytvorenie koláčového grafu v prostredí R je:

```
pie(data, labels, radius, main, col, clockwise)
```

Význam argumentov:

- `data` je vektor obsahujúci číselné hodnoty použité v koláčovom grafe,
- `labels` sa používa na popis výsekov,
- `radius` označuje polomer kruhu koláčového grafu (hodnota medzi -1 a $+1$),
- `main` označuje názov grafu,
- `col` označuje paletu farieb,
- `clockwise` je logická hodnota, ktorá udáva, či sa výseky kreslia v smere alebo proti smeru hodinových ručičiek.

Koláčový graf – úprava farieb

Na zmenu farieb v grafe použijeme funkciu `rainbow()`, ktorá definuje paletu farieb.
Jej argumenty sú:

- `n` počet farieb (≥ 1), ktoré majú byť v palete,
- `s,v` „sýtosť“ a „hodnota“, ktoré sa majú použiť na doplnenie popisu ku farbám
- `start` (opravený) odtieň v $\langle 0; 1\rangle$, pri ktorom začína zvolená dúha,
- `end` (opravený) odtieň v $\langle 0; 1\rangle$, pri ktorom dúha končí,
- `gamma` korekcia gama pre každú farbu, `(r,g,b)` v priestore RGB (so všetkými hodnotami v $\langle 0; 1\rangle$), výsledná farba zodpovedá $(r^\gamma, g^\gamma, b^\gamma)$,
- `alpha` priehľadnosť, číslo v $\langle 0; 1\rangle$, (0 znamená priehľadný a 1 znamená nepriehľadný).

Koláčový graf–použitie rainbow()

```
1 description<-paste(labels,"\n",data,sep="")
2 pie(data,description,main="Monthly␣expenses",
3 col=rainbow(length(data)))
```

Poznámka

Zmenili sme aj popisky. K ich názvom sme pridali aj číselné hodnoty.

Vejárový graf

Užitočnou alternatívou ku koláčovým grafom je `fun.plot()` definovaný v balíku `plotrix`.

Umožňuje vizuálne porovnať koláčové sektory grafu.

Vejárový graf môžeme prispôsobiť nastavením ďalších argumentov:

- `max.span` uhol maximálneho sektora v radiánoch. Predvolené nastavenie je škálovanie `data` tak, aby sa súčet rovnal 2π.
- `ticks` počet políčok, ktoré by sa objavili, keby boli sektory na koláčovom grafe. Predvolené nastavenie je žiadne políčka.

Krabicový graf

V popisnej štatistike je krabicový graf alebo boxplot typ grafu, ktorý sa často používa pri vysvetľovaní analýzy údajov.

Krabicové grafy vizualizujú rozdelenie číselných údajov a šikmosť prostredníctvom zobrazenia kvartilov (alebo percentilov) a priemerov údajov.

Je tiež užitočný pri porovnávaní rozdelenia údajov v rôznych súboroch údajov tak, že sa pre každý z nich nakreslí krabicový graf.

Krabicový graf

Boxplot sa skladá z dvoch častí, boxu a sady „whiskerov“.

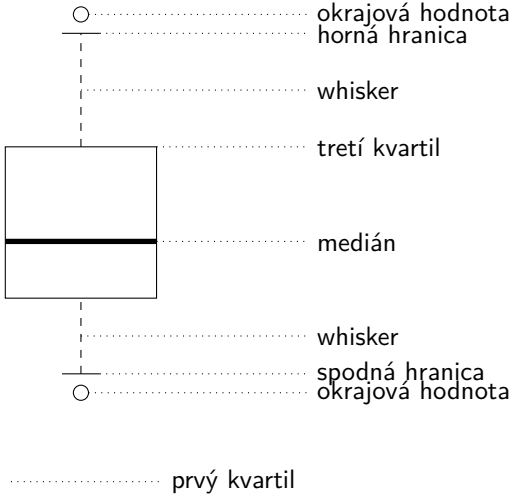
Box je zobrazuje rozsah od dolného kvartilu po horný kvartil s vodorovnou čiarou nakreslenou vo vnútri boxu, ktorá označuje medián.

Whiskery môžu alternatívne znázorňovať niekoľko okrajových hodnôt spomedzi pozorovaných údajov:

- minimum a maximum všetkých údajov,
- jedna štandardná odchýlka nad a pod priemerom údajov,
- 9. percentil a 91. percentil,
- 2. percentil a 98. percentil.

Všetky údaje, ktoré nie sú zahrnuté medzi whiskery, by sa mali vykresliť ako odľahlé hodnoty vyznačené bodkou, malým krúžkom alebo hviezdičkou, občas sa to však nevykonáva.

Krabicový graf



Q-Q graf

Kvantilový graf (alebo skrátene Q-Q graf) je grafický nástroj, ktorý nám pomáha posúdiť, či súbor údajov vierohodne pochádza z nejakého teoretického rozdelenia, napríklad normálneho alebo exponenciálneho.

Ak napríklad vykonávame štatistickú analýzu, ktorá predpokladá, že naša závislá premenná je normálne rozdelená, môžeme na overenie tohto predpokladu použiť normálny Q-Q graf.

Je to len vizuálna kontrola, nie exaktný dôkaz, ale umožňuje nám na prvý pohľad vidieť, či je náš predpoklad hodnoverný, a ak nie, ako je predpoklad porušený a ktoré dátové body prispievajú k porušeniu.

Q-Q graf

Q-Q graf je v podstate bodový graf vytvorený vykreslením dvoch súborov kvantilov proti sebe.

Ak obe sady kvantilov pochádzajú z rovnakého rozdelenia, body tvoria približne priamku.

Q-Q grafy vezmú naše výberové vzorky, zoradia ich vzostupne a potom ich vykreslia oproti kvantilom navrhovaného teoretického rozdelenia.

Počet kvantilov je zvolený tak, aby zodpovedal veľkosti našej vzorky údajov.

Viac grafov v jednom obrázku

V prostredí R môžeme graf kombinovať s grafickými parametrami `mfrow` a `mfcol`.

Stačí zadať vektor, ktorý určuje počet riadkov a počet stĺpcov, ktoré plánujeme vytvoriť.

Rozhodnutie, ktorý grafický parameter použijeme, závisí od toho, ako chceme mať usporiadané naše grafy:

- `mfrow` budú grafy usporiadané podľa riadkov,
- `mfcol` grafy budú usporiadané podľa stĺpcov.

Toto nastavenie sa používa ako argument funkcie `par()`, ktorá definuje parametre grafického zariadenia

Štatistika a programovanie v R

VI. Výberové charakteristiky

Priemer

Priemer danej množiny hodnôt $x_1, x_2, \ldots, x_n$ je určený vzťahom:

$$\overline{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{\sum_{i=1}^{n} x_i}{n}. \tag{9}$$

V jazyku R je implementovaný ako funkcia `mean()`.

Jej použitie je veľmi jednoduché.

Priemer

Na internetovej stránke slovenskej centrálnej banky `https://www.nbs.sk/en/monetary-policy/macroeconomic-database` nájdeme makroekonomickú databázu.

Môžeme si stiahnuť napríklad súbor `.csv`, ktorý obsahuje počty registrácií nových osobných automobilov vo vybranom časovom období.

Tento súbor je implicitne uložený ako `makrostat.csv`.

Z dôvodu použitia čiarky ako oddeľovača desatinných čísiel čítame údaje pomocou funkcie `read.csv2()`.

Priemer

Na výpočet priemeru musíme použiť funkciu `as.numeric()`, pretože `cars[1,]` dáva hodnoty vo formáte zoznamu.

Takže na získanie priemernej hodnoty nových osobných automobilov registrovaných za mesiac (v tisícoch) v rokoch 2017-18 použijeme kód:

```
1 cars<-read.csv2("macrostat.csv",header=FALSE,sep=";")
2 mean(as.numeric(cars[1,]))
3 [1] 8.090125
```

Priemer

Často sa stáva, že hodnoty štatistického znaku, ktorý nás zaujíma, sú zoradené v postupnosti absolútnych početností.

V tomto prípade upravíme vzťah (9) na výpočet priemernej hodnoty na tvar:

$$\overline{x} = \frac{x_1 \cdot n_1 + x_2 \cdot n_2 + \cdots + x_k \cdot n_k}{n_1 + n_2 + \cdots + n_k} = \frac{\sum_{i=1}^{k} x_i \cdot n_i}{\sum_{i=1}^{k} n_i}, \tag{10}$$

kde x_i označujú hodnoty premennej a n_i ich absolútne početnosti.

Priemer

V tomto prípade musíme definovať vlastnú funkciu na výpočet strednej hodnoty.

Ako vstupné hodnoty zadáme dva vektory. Prvý vektor obsahuje hodnoty, ktoré nadobúda náhodná premenná, a druhý je vektor ich početností.

Pred vykonaním výpočtu podľa vzťahu (10) je potrebné overiť, či majú oba vektory rovnakú dĺžku.

Medián

Medián môžeme charakterizovať ako strednú hodnotu, ak sú pozorované hodnoty zoradené od najmenšej po najväčšiu.

Formálne môžeme povedať, že pravdepodobnosť, že hodnota premennej je väčšia ako medián sa rovná pravdepodobnosti, že jej hodnota je menšia ako medián, a teda táto pravdepodobnosť sa rovná $\frac{1}{2}$.

Ak máme vzorku hodnôt $x_1, x_2, \ldots, x_n$, medián $\tilde{x}$ je daný vzťahmi

$$\tilde{x} = \begin{cases} \left(\frac{n+1}{2}\right)\text{-tá usporiadaná hodnota} & n \text{ je nepárne} \\ \frac{1}{2}\left(\left(\frac{n}{2}\right)\text{-tá} + \left(\frac{n}{2}\right)\text{-tá usporiadaná}\right) & n \text{ je párne} \end{cases} \tag{11}$$

Kvantily

Na zistenie kvantilov je v jazyku R implementovaná funkcia `quantile()`. Bez zadania voliteľných parametrov je výstupom minimum vzorky, prvý kvartil, medián, tretí kvartil a maximm vzorky.

Môžeme si to ilustrovať na údajoch o COVID-19, stiahnutých z oficiálnej webovej stránky slovenskej vlády https://korona.gov.sk.

```
1  data<-read.csv("https://mapa.covid.chat/export/csv",
2     header=T,sep=";")
3  > quantile(data[,4])
4     0%    25%    50%    75%   100%
5      0     30    232   1737  15278
6  >
```

Variačné rozpätie

Variačné rozpätie závisí len od dvoch hodnôt vo vzorke – najväčšej a najmenšej hodnoty.

Je definované ako rozdiel

$$R = \max\{x_1, \ldots, x_n\} - \min\{x_1, \ldots, x_n\} \tag{12}$$

Je dobre použiteľné na rýchlu orientáciu v rozsahu variability danej vzorky.

Variačné rozpätie

Výstupom funkcie `range()` v prostredí jazyka R je variačné rozpätie.

Jej výstupom sú dve hodnoty - najväčšia a najmenšia hodnota vo vzorke.

Aby sme mohli vyjadriť rozsah odchýlky ako jednu hodnotu podľa definície (12), použijeme funkcie `max()` a `min()`.

Medzikvartilové rozpätie

Medzikvartilové rozpätie je mierou štatistického rozptylu.

Je definované ako rozdiel medzi horným a dolným kvartilom vzorky.

Priradením týchto kvartilov ako Q_3 a Q_1 môžeme medzikvartilové rozpätie IQR vyjadriť ako

$$IQR = Q_3 - Q_1. \tag{13}$$

Stredná absolútna odchýlka

Priemerná absolútna odchýlka *MAD* výberovej vzorky je priemerná vzdialenosť medzi každým údajom a priemerom.

Pre vzorku $x_1, \ldots, x_n$ je definovaná vzorcom

$$MAD = \frac{1}{n} \sum_{i=1}^{n} |x_i - \overline{x}| \,. \tag{14}$$

Rozptyl a smerodajná odchýlka

Rozptyl je najpopulárnejšou a najčastejšie používanou mierou variability vzorky.

Ak máme vzorku $x_1, \ldots, x_n$, definujeme výberový rozptyl s^2 vzťahom

$$s^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \overline{x})^2 . \tag{15}$$

Rozptyl a smerodajná odchýlka

V prípade údajov usporiadaných v postupnosti absolútnych početností $n_1, \ldots, n_k$, $\sum_{i=1}^{k} n_i = n$, jednotlivých hodnôt $x_1, \ldots, x_k$, upravíme vzorec (15) do tvaru

$$s^2 = \frac{1}{n} \sum_{i=1}^{k} n_i \left(x_i - \overline{x}\right)^2 . \tag{16}$$

Rozptyl a smerodajná odchýlka

Funkcie `var()` a `sd()` musíme používať opatrne.

Ich výsledkom sú nevychýlené odhady rozptylu a smerodajnej odchýlky celej populácie.

Ak chceme vypočítať výberový rozptyl podľa vzťahu (15), musíme definovať vlastnú funkciu, ktorú ilustrujeme v nasledujúcom zdrojovom kóde.

Rozptyl a smerodajná odchýlka

```
> variance<-function(x) sum((x-mean(x))^2)/length(x)
> stdev<-function(x) sqrt(variance(x))
> variance(x)
[1] 14.40816
> stdev(x)
[1] 3.795809
> var(x) # porovnajte vysledky
[1] 16.80952
> sd(x)
[1] 4.099942
```

Variačný koeficient

Variačný koeficient je štatistická miera relatívneho rozptylu dátových údajov v pomere ku strednej hodnote.

Variačný koeficient CV je definovaný ako pomer štandardnej odchýlky s k priemeru $\overline{x}$

$$CV = \frac{s}{\overline{x}}. \tag{17}$$

Variačný koeficient sa často vyjadruje v percentách.

Variačný koeficient

Variačný koeficient nie je v jazyku R implementovaný ako funkcia

Môžeme ho vyčísliť pomocou známych funkcií alebo si definovať funkciu vlastnú

```
1 > cv<-function(x) variance(x)/mean(x) * 100
2 > cv(x)
3 [1] 157.5893
```

Šikmosť

Šikmosť je mierou asymetrie rozdelenia alebo súboru údajov.

Šikmosť γ_1 definujeme ako

$$\gamma_1 = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^3}{s^3}. \tag{18}$$

Šikmosť

V závislosti od hodnoty γ_1 existujú 3 typy šikmosti.

Ak je $\gamma_1 > 0$, hovoríme o pravostrannej šikmosti. Znamená to, že väčšina údajov je menšia ako priemer.

Na druhej strane, keď $\gamma_1 < 0$ hovoríme o ľavostrannej šikmosti. V tomto prípade je väčšina hodnôt v súbore údajov väčšia ako priemer.

A napokon nulová šikmosť $\gamma_1 = 0$ predstavuje symetrické rozdelenie údajov.

Špicatosť

Špicatosť meria ostrosť vrcholu v rozdelení údajov.

špicatosť γ_2 definujeme ako

$$\gamma_2 = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^4}{s^4}. \tag{19}$$

Špicatosť

Podobne ako v prípade šikmosti existujú tri typy špicatosti.

Ak $\gamma_2 = 3$, hovoríme o normálnej špicatosti rozdelenia. Hodnota γ_2 sa porovnáva s hodnotou 3, pretože špicatosť normálneho rozdelenia je rovná 3. Porovnávame vrchol vzorky s Gaussovou krivkou.

V prípade $\gamma_2 < 3$ je rozdelenie plochejšie ako normálne rozdelenie.

V opačnom prípade, keď $\gamma_2 > 3$, analyzované rozdelenie má ostrejší vrchol než normálne rozdelenie.

Štatistika a programovanie v R

VII. Odhady parametrov

Odhady parametrov

Jedným z cieľov štatistickej analýzy je odhadnúť parametre pôvodného rozdelenia, z ktorého pochádza náhodný výber.

Rozlišujeme dva typy odhadov:

- bodové odhady, ktoré poskytujú odhad presných hodnôt parametrov,
- intervaly spoľahlivosti, ktoré predstavujú intervaly, ktoré obsahujú hodnotu parametra s danou pravdepodobnosťou (uvádza sa ako hladina spoľahlivosti).

Bodové odhady

Pre odhad hodnoty parametra sa snažíme vybrať takú charakteristiku, ktorá najlepšie aproximuje parameter Θ.

Kvalitu odhadu určujú jeho vlastnosti:

- Nevychýlený odhad parametra θ je taký odhad T, že platí rovnosť $\mathbb{E}(T) = \theta$.
- Konzistentný odhad môže byť charakterizovaný rastúcou presnosťou so zvyšujúcou sa veľkosťou vzorky. Formálne môžeme písať $\lim_{n\to\infty} T_n = \Theta$, kde indexy predstavujú veľkosť vzorky použitej pri odhade.
- Efektívny odhad možno interpretovať ako najlepší možný odhad. Nepresnosť sa meria strednou kvadratickou chybou T, t.j. hodnotou $MSE(T) = \mathbb{E}\left((T - \Theta)^2\right)$. Efektívny odhad túto hodnotu minimalizuje.

Bodové odhady

Tieto vlastnosti ovplyvňujú implementovanie charakteristík vzoriek v jazyku R.

Pozrime sa na priemer vzorky. Ak máme náhodnú vzorku $X_1, \ldots, X_n$ z rozdelenia so strednou hodnotou μ, môžeme vypočítať:

$$\mathbb{E}(\overline{x}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}(X_i) = \frac{1}{n} n\mu = \mu.$$

Takže sme ukázali, že priemer je nevychýlený odhad priemeru vzorky.

Bodové odhady

Teraz sa pozrime na výberový rozptyl. Použijeme nerovnosti

$$\sum_{i=1}^{n}(X_i - \overline{X})^2 = \sum_{i=1}^{n}(X_i - \mu + \mu - \overline{X})^2 = \sum_{i=1}^{n}(X_i - \mu)^2 - n(\overline{X} - \mu)^2,$$

dostaneme

$$\begin{aligned}\mathbb{E}\left(\sum_{i=1}^{n}(X_i - \overline{X})^2\right) &= \mathbb{E}\left(\sum_{i=1}^{n}(X_i - \mu)^2 - n(\overline{X} - \mu)^2\right) \\ &= \sum_{i=1}^{n}\mathbb{D}\left(\overline{X_i}\right) - n\mathbb{D}\left(\overline{X}\right) \\ &= n\sigma^2 - \frac{n\sigma^2}{n} = (n-1)\sigma^2,\end{aligned}$$

Bodové odhady

Teraz sa pozrime na výberový rozptyl. Použijeme nerovnosti

$$\sum_{i=1}^{n}(X_i - \overline{X})^2 = \sum_{i=1}^{n}(X_i - \mu + \mu - \overline{X})^2 = \sum_{i=1}^{n}(X_i - \mu)^2 - n(\overline{X} - \mu)^2,$$

a preto

$$\mathbb{E}\left(s^2\right) = \mathbb{E}\left(\frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^2\right) = \frac{n-1}{n}\sigma^2.$$

Vidíme teda, že výberový rozptyl nie je nevychýlený odhad náhodného rozptylu.

Bodové odhady

Teraz sa pozrime na výberový rozptyl. Použijeme nerovnosti

$$\sum_{i=1}^{n}(X_i - \overline{X})^2 = \sum_{i=1}^{n}(X_i - \mu + \mu - \overline{X})^2 = \sum_{i=1}^{n}(X_i - \mu)^2 - n(\overline{X} - \mu)^2,$$

Aby sme získali nevychýlený odhad, musíme vynásobiť výberový rozptyl $\frac{n}{n-1}$. Získame tak nevychýlený odhad rozptylu

$$s_{n-1}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})^2.$$

Tento výsledok vysvetľuje, prečo je funkcia `var()` implementovaná odlišne ako výberový rozptyl. Predstavuje nevychýlený odhad pôvodnej náhodnej premennej.

Bodové odhady

Metódy

V tomto kurze predstavíme dve metódy konštrukcie bodových odhadov:

- metóda momentov,
- metóda maximálnej pravdepodobnosti.

Predpokladajme, že máme vzorku $X_1, \ldots, X_n$ z rozdelenia, ktorá závisí od vektora parametrov $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_m)$.

Metóda momentov

Predpokladajme ďalej, že existujú všetky momenty $\nu_k = \mathbb{E}\left(X_i^k\right)$, $k = 1, \ldots, m$.

Momenty vzorky v_k sú definované ako $v_k = \frac{1}{n}\sum_{i=1}^{n} X_i^k$ pre $k = 1, \ldots, m$.

Princípom metódy momentov je rovnosť teoretického momentu a momentu vzorky.

Metóda momentov

Znamená to odhadnúť momenty pre $\theta_1, \theta_2, \ldots, \theta_k$, označenie $\widehat{\theta}_1, \widehat{\theta}_2, \ldots, \widehat{\theta}_k$, ktoré sú definované ako riešenie (pokiaľ existuje) rovníc:

$$\nu_k = v_k, \quad k = 1, \ldots, m.$$

Alternatívne namiesto ľubovoľného r-teho momentu môžeme použiť r-tý centrálny moment definovaný ako $\mu_r = \mathbb{E}\left((X - \mathbb{E}(X))^r\right)$ a r-tý centrálny moment vzorky $m_r = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^r$.

Metóda momentov

Metódu ilustrujeme na prípade rovnomerného rozdelenia s parametrami a a b. Je známe, že momenty rovnomerne rozloženej náhodnej premennej X sú

$$\mathbb{E}(X) = \frac{a+b}{2} \quad \text{a} \quad \mathbb{D}(X) = \frac{(b-a)^2}{12}.$$

Určiť odhady parametrov a a b predstavuje vyriešiť rovnice:

$$\overline{x} = \frac{a+b}{2},$$
$$s^2 = \frac{(b-a)^2}{12}.$$

Metóda momentov

Riešením týchto rovníc dostaneme:

$$a = \overline{x} - \sqrt{3}s,$$
$$b = \overline{x} + \sqrt{3}s.$$

Teraz sme pripravení implementovať tieto odhady v jazyku R. Použijeme vopred definované funkcie `variation()` resp. `stdev()`.

Metóda momentov

```
1 > x<-runif(1000,1,3) #generating the random sample
2 > a<-mean(x)-sqrt(3)*stdev(x)
3 > b<-mean(x)+sqrt(3)*stdev(x)
4 > a
5 [1] 1.018323 #can differ for other samples
6 > b
7 [1] 3.033395 #can differ for other samples
```

Metóda maximálnej pravdepodobnosti

Táto metóda je založená na maximalizácii pravdepodobnostnej funkcie.

Bod v priestore parametrov, ktorý maximalizuje funkciu pravdepodobnosti, sa považuje za odhad maximálnej pravdepodobnosti.

Formálne predpokladajme, že $X_1, \ldots, X_n$ sú nezávislé náhodné premenné s rovnakým rozdelením s hustotou $f(x, \boldsymbol{\theta})$. Zjednotená hustota je súčinom týchto jednorozmerných funkcií hustoty:

$$L(x_1, \ldots, x_n; \boldsymbol{\theta}) = \prod_{i=1}^{n} f(x, \boldsymbol{\theta}).$$

Metóda maximálnej pravdepodobnosti

Práve predstavená funkcia $L(x_1, \ldots, x_n; \boldsymbol{\theta})$ sa nazýva pravdepodobnostná funkcia.

Aby sme získali odhady parametrov, maximalizujeme túto funkciu štandardným procesom známym z matematickej analýzy.

Pre uľahčenie práce maximalizujeme namiesto funkcie $L(x_1, \ldots, x_n; \boldsymbol{\theta})$ jej logaritmus $\ln L(x_1, \ldots, x_n; \boldsymbol{\theta})$.

Prirodzený logaritmus je rastúca funkcia, preto má extrémy a navyše prevádza súčin na súčet funkcií.

Metóda maximálnej pravdepodobnosti

Metódu ilustrujeme na probléme odhadu pravdepodobnosti p nejakej náhodnej udalosti A.

Túto situáciu môžeme interpretovať ako výsledok alternatívnej náhodnej premennej, ktorá je indikátorom náhodnej udalosti A.

Potom máme $\mathbb{P}(X = 1) = p$ a $\mathbb{P}(X = 0) = 1 - p$.

Vykonávame n náhodných pokusov a pozorujeme výskyt udalosti A.

Takže dostaneme vzorku $X_1, \ldots, X_n$ náhodných premenných s hustotami $f(x_i, p) = p^{x_i}(1-p)^{1-x_i}$, kde $x_i \in \{0, 1\}$.

Metóda maximálnej pravdepodobnosti

Zodpovedajúca pravdepodobnostná funkcia má tvar

$$L(x,p) = \prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i} = p^{\sum_{i=1}^{n} x_i}(1-p)^{n-\sum_{i=1}^{n} x_i}.$$

Aby sme mohli určiť odhad p, maximalizujeme funkciu $L(x,p)$ vzhľadom na parameter p. Aby sme to dosiahli, používame prirodzený logaritmus pravdepodobnostnej funkcie

$$\ln(L(x,p)) = \sum_{i=1}^{n} x_i \ln p + \left(n - \sum_{i=1}^{n} x_i\right) \ln(1-p),$$

Metóda maximálnej pravdepodobnosti

Jeho deriváciu vzhľadom na p položíme rovnú 0:

$$\frac{\mathrm{d} \ln L(x,p)}{\mathrm{d}x} = \frac{\sum_{i=1}^{n} x_i}{p} - \frac{n - \sum_{i=1}^{n} x_i}{1-p} = 0.$$

Vynásobením rovnice $\frac{1}{n}$ dostaneme

$$\overline{x} \cdot \frac{1}{p} - (1 - \overline{x}) \cdot \frac{1}{1-p} = 0,$$

a riešením je

$$p = \overline{x}.$$

Metóda maximálnej pravdepodobnosti

Aby sme potvrdili, že $p = \overline{x}$ skutočne maximalizuje funkciu pravdepodobnosti, musíme overiť deriváciu druhého rádu.

$$\frac{\mathrm{d}^2 \ln L(x,p)}{\mathrm{d}x^2} = -\frac{\overline{x}}{p} + (1-\overline{x}) \cdot \frac{1}{(1-p)^2}.$$

Nahradením $p = \overline{x}$ dostaneme

$$\left(\frac{\mathrm{d}^2 \ln L(x,p)}{\mathrm{d}x^2} \right)_{p=\overline{x}} = -\frac{1}{1-\overline{x}} < 0.$$

Potom máme najpravdepodobnejší odhad $p = \overline{x}$.

Metóda maximálnej pravdepodobnosti

Tento prístup môžeme použiť na určenie toho, do akej miery je hádzanie mincou spravodlivé.

Najprv vygenerujeme vzorku 0 a 1, čo znamená hod rub alebo líc. Aby sme získali vzorku neférovej mince, deklarujeme vektor pravdepodobností `prob`, ako môžeme vidieť na zdrojovom kóde:

```
1 > x<-sample(c(0,1),1000,replace=TRUE,prob=c(2/3,1/3))
2 > mean(x)
3 [1] 0.304
```

Intervaly spoľahlivosti

Interval spoľahlivosti môže byť definovaný ako rozsah odhadov pre neznámy parameter, ktorý obsahuje skutočnú hodnotu parametra s danou pravdepodobnosťou.

Táto pravdepodobnosť, že sa parameter nachádza v danom intervale, sa uvádza ako hladina spoľahlivosti.

Najbežnejšia hladina spoľahlivosti používaná v praxi je 95 %, ale často sa používajú aj iné úrovne (napríklad 90 % alebo 99 %).

Intervaly spoľahlivosti

Formálne nech $\mathbf{X} = (X_1, \ldots, X_n)$ je náhodná vzorka z rozdelením, ktoré závisí od neznámeho parametra θ.

Interval spoľahlivosti pre parameter θ s hladinou spoľahlivosti α je interval s náhodnými koncovými bodmi $(u(\mathbf{X}); v(\mathbf{X}))$ určený dvojicou náhodných premenných $u(\mathbf{X})$ a $v(\mathbf{X})$ s vlastnosťou:

$$\mathbb{P}\left(u(\mathbf{X}) < \theta < v(\mathbf{X})\right) = \alpha.$$

Intervaly spoľahlivosti

Odvodenie intervalu spoľahlivosti ilustrujeme na normálnom rozdelení.

Predpokladajme najprv, že $X_1, \ldots, X_n$ je náhodná vzorka z normálneho rozdelenia $N(\mu, \sigma^2)$, ktorej štandardná odchýlka σ je známa.

Chceme nájsť interval spoľahlivosti pre strednú hodnotu μ. Pretože $\overline{X}$ má normálne rozdelenie $N\left(\mu, \frac{\sigma^2}{n}\right)$, máme

$$\mathbb{P}\left(\left|\frac{\overline{X} - \mu}{\frac{\sigma}{\sqrt{n}}}\right| < c\right) = 2\Phi(c) - 1, \text{ pre všetky } c > 0,$$

Intervaly spoľahlivosti

Je to ekvivalentné

$$\mathbb{P}\left(\overline{X} - c\frac{\sigma}{\sqrt{n}} < \mu < \overline{X} + c\frac{\sigma}{\sqrt{n}}\right) = 2\Phi(c) - 1, \text{ pre všetky } c > 0.$$

Z posledného vzťahu vyplýva, že interval spoľahlivosti pre priemer s hladinou spoľahlivosti $\alpha = 2\Phi(c) - 1$ má tvar

$$\left(\overline{X} - c\frac{\sigma}{\sqrt{n}}; \overline{X} + c\frac{\sigma}{\sqrt{n}}\right).$$

Ak vezmeme do úvahy, že $c = \Phi^{-1}\left(\frac{\alpha+1}{2}\right)$, môžeme interval spoľahlivosti zapísať v tvare

$$\left(\overline{X} - \Phi^{-1}\left(\frac{\alpha+1}{2}\right)\frac{\sigma}{\sqrt{n}}; \overline{X} + \Phi^{-1}\left(\frac{\alpha+1}{2}\right)\frac{\sigma}{\sqrt{n}}\right).$$

Intervaly spoľahlivosti

Na zistenie hraníc intervalu spoľahlivosti používame kvantilovú funkciu `qnorm()`.

```
x<-rnorm(100,1,2) # we generate some sample
> n<-length(x)
> sample.mean<-mean(x)
> sample.sd<-2 #  standard deviation is known
> alpha<-0.95 # seting the confidence level 95\%
> c<-qnorm((alpha+1)/2,0,1)
> margin<-c*sample.sd/sqrt(n)
> lower.bound<-sample.mean-margin
> upper.bound<-sample.mean+margin
> print(c(lower.bound,upper.bound))
[1] 0.8149884 1.5989740
```

Intervaly spoľahlivosti

Teraz sa pozrime, ako sa zmení interval spoľahlivosti, ak štandardná odchýlka nie je známa.

Potom musíme použiť nevychýlený odhad štandardnej odchýlky $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\overline{X} - X_i)^2$, $s = \sqrt{s^2}$. Potom náhodná premenná

$$T = \frac{\overline{X} - \mu}{s}\sqrt{n},$$

má Studentovo t-rozdelenie s $n-1$ stupňami voľnosti. Potom dostaneme uvedený interval spoľahlivosti

$$\left(\overline{X} - c\frac{s}{\sqrt{n}}; \overline{X} + c\frac{s}{\sqrt{n}}\right).$$

Intervaly spoľahlivosti

Hodnota c je zodpovedajúci kvantil Studentovho rozdelenia, preto v R použijeme funkciu `qt()`.

```
> n<-length(x) # we use previously generated sample
> sample.mean<-mean(x)
> sample.sd<-sd(x) # estimate of the standard deviation
> alpha<-0.95
> c<-qt((alpha+1)/2,df=n-1)
> margin<-c*sample.sd/sqrt(n)
> lower.bound<-sample.mean-margin
> upper.bound<-sample.mean+margin
> print(c(lower.bound,upper.bound))
[1] 0.8118763 1.6020861
```

Intervaly spoľahlivosti

Teraz sa pozrime na interval spoľahlivosti pre rozptyl normálneho rozdelenia.

Ak s^2 je nevychýlený odhad rozptylu, potom náhodná premenná

$$Y = \frac{(n-1)s^2}{\sigma^2} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{\sigma^2} = \sum_{i=1}^{n}\left(\frac{x_i - \overline{x}}{\sigma}\right)^2$$

sa riadi rozdelením $\chi^2(n-1)$.

Intervaly spoľahlivosti

Pre interval spoľahlivosti dostaneme

$$\mathbb{P}(c_1 < \chi^2 < c_2) = \alpha$$
$$\mathbb{P}\left(c_1 < \frac{(n-1)s^2}{\sigma^2} < c_2\right) = \alpha$$
$$\mathbb{P}\left(\sigma \in \left(\frac{(n-1)s^2}{c_2}; \frac{(n-1)s^2}{c_1}\right)\right) = \alpha,$$

kde c_1 a c_2 sú kritické hodnoty rozdelenia $\chi^2(n-1)$.

Intervaly spoľahlivosti

Pri výpočte musíme rešpektovať, že rozdelenie χ^2 nie je symetrické, čo je dôležité pre kritické hodnoty c_1 a c_2.

```
> n<-length(x) # the sample already generated sooner
> sample.var<-var(x)
> c1<-qchisq(1-(alpha+1)/2,df=n-1)#consequent of asymmetry
> c2<-qchisq((alpha+1)/2,df=n-1)#consequent of asymmetry
> lower.bound<-sample.var*(n-1)/c2
> upper.bound<-sample.var*(n-1)/c1
> print(c(lower.bound,upper.bound))
[1] 3.056626 5.350767
```

Intervaly spoľahlivosti

V poslednom príklade si ukážeme interval spoľahlivosti pre pravdepodobnosť náhodnej udalosti.

Nevychýlený odhad pravdepodobnosti p výskytu náhodnej udalosti je $\hat{p} = \frac{m}{n}$, kde m je počet výskytov pozorovanej udalosti v rade n náhodných pokusov.

Vieme, že $\mathbb{E}(\hat{p}) = p$ a $\mathbb{D}(\hat{p}) = \frac{pq}{n}$, kde $q = 1 - p$.

Pre veľké n platí približná rovnosť $\frac{pq}{n} \approx \frac{\hat{p}\hat{q}}{n}$.

Intervaly spoľahlivosti

Potom náhodná premenná

$$Z = \frac{\frac{m}{n} - p}{\sqrt{\frac{\hat{p}\hat{q}}{n}}}$$

sa riadi štandardizovaným normálnym rozdelením $N(0, 1)$. Potom interval spoľahlivosti pre pravdepodobnosť p s hladinou spoľahlivosti α môžeme zapísať ako

$$\left(\hat{p} - \Phi^{-1}\left(\frac{\alpha+1}{2}\right) \cdot \sqrt{\frac{\hat{p}\hat{q}}{n}}; \hat{p} + \Phi^{-1}\left(\frac{\alpha+1}{2}\right) \cdot \sqrt{\frac{\hat{p}\hat{q}}{n}} \right).$$

Intervaly spoľahlivosti

Example

Predpokladajme, že sa uskutočnil prieskum na 250 náhodne vybraných ľuďoch, aby sa zistilo či vlastnia tablet. Z 250 opýtaných 98 uviedlo, že vlastnia tablet. Pomocou 95 % hladiny spoľahlivosti vypočítajte odhad intervalu spoľahlivosti pre skutočný podiel ľudí, ktorí vlastnia tablet.

Ďakujem za pozornosť.

Štatistika a programovanie v R

Aleš Kozubík