Základy pravdepodobnosti a matematickej štatistiky

Martin Kalina
Tomáš Bacigál
Anna Schiesslová

ISBN 978-80-227-3273-4

Slovenská technická univerzita v Bratislave
2010

Obsah

Kapitola 11

Príklady v R

Štatistické výpočty sa už dávno nevykonávajú hlavou a perom na papieri. Počnúc vedeckými kalkulačkami s niekoľkými štatistickými funkciami (výpočet priemeru či štandardnej odchýlky), obľúbenným nástrojom hlavne vo výučbe na vysokých školách sa stali najmä všeobecne zamerané výpočtové programy ako MS Excel, Mathsoft Mathcad alebo ešte univerzálnejší Wolfram Mathematica. Táto ich prednosť sa však stáva prekážkou, ak vyžadujeme pokročilejšie nástroje matematickej štatistiky. Vtedy odborníci siahajú po špecializovaných a prevažne komerčných softvérových riešeniach ako SAS, S-PLUS, SPSS, STATGRAPHICS a pod. Z tých nekomerčných si v odbornej komunite získal rešpekt projekt známy pod názvom R, postavený na programovacom jazyku S (ten je základom aj vyššie spomínaného softvérového produktu S-PLUS). Svojim „čarom“ si získal aj autorov týchto skrípt a jeho schopnostiam venujú celú 11. kapitolu. Tá je členená do troch častí, v prvej si R predstavíme ako univerzálny výpočtový systém, druhá osloví riešenými príkladmi základného kurzu matematickej štatistiky a konečne v tretej časti si „na svoje“ prídu záujemci o pokročilejšie štatistické metódy.

Keďže R používa bodku (nie čiarku) ako oddelovač desatinných miest, v záujme lepšej čitateľnosti kapitoly prispôsobíme tejto konvencii aj zápis reálnych čísel v bežnom texte.

11.1 Úvod do R

R je voľne šíriteľný softvérový nástroj pre štatistické výpočty a grafickú vizualizáciu. Týmto jediným písmenom sa označuje prostredie i programovací jazyk zároveň. Príkazy, napísané v ľubovoľnom editore, sú spracovávané v príkazovom riadku podobne ako v operačnom systéme Unix, niektoré nadstavby umožňujú výber z ponuky príkazov a ovlá-

danie myšou. Hoci „eRko“ nie je obmedzené len na štatistiku, to čo z neho robí silný štatistický nástroj, je rozsiahla knižnica podporných programov vyvíjaných odbornou verejnosťou – dobrovoľníkmi – na celom svete. Verný filozofii voľne šíritelného softvéru, R je dostupný na platformách Windows, Linux, MacOS a ďalších. Na oficiálnej webstránke cran.r-project.org sa nachádza dokumentácia, inštalačné súbory, komunitné fóra na zdieľanie poznatkov a daľšie informácie.

V nasledujúcich podkapitolách si na jednoduchých príkladoch opíšeme syntax jazyka R. Vstup z klávesnice je indikovaný znakom „>“ zo začiatku príkazového riadku, kratší komentár umiestňujeme priamo do kódu za znak „#“ (za ním sa text už nevyhodnocuje). Záujemcom o zvládnutie základov odporúčame si príklady aj sami vyskúšať, v systéme Windows najlepšie písaním príkazov do vstavaného editora (v menu zvoliť *File/New script*). Text z editora sa do príkazového riadku odosiela buď po vyznačených častiach alebo po riadkoch, stlačením kombinácie kláves *Ctrl-R*.

11.1.1 R ako lepšia kalkulačka

R primárne pracuje s poľami, z ktorých najjednoduchší je vektor. Aj osamotené reálne číslo je reprezentované ako jednoprvkový vektor, čo nám už v nasledujúcich príkladoch pripomína index prvku `[1]` na začiatku výstupného riadku.

Základné matematické funkcie

```
> 2+3
[1] 5
> 3/2
[1] 1.5
> 2^3
[1] 8
> 4 ^ 2 - 3 * 2
[1] 10
> (56-14)/6 - $4*7*10/(5^2-5)$
[1] -7
> sqrt(2)
[1] 1.414214
> abs(2-4)       #|2-4|
[1] 2
> cos(4*pi)      #ďalšie sú sin(), tan(), atan(), atan2() ...
[1] 1
> log(0)         #nie je definované
[1] -Inf
> exp(1)
[1] 2.718282
> factorial(6)      #6!
```

```
[1] 720
> choose(52,5)       #kombinačné číslo 52!/(47!*5!)
[1] 2598960
```

Priradenie hodnoty premennej

```
> n <- 5
> n
[1] 5
> 15 -> n
> n
[1] 15
> assign("n", 25)
> n = 35        #radšej nepoužívať, (esteticky) vyhradené pre zadávanie argumentov funkcie
> (n <- 0)      #dva v jednom -- priradenie a výpis
[1] 0
```

Vektor

```
> x <- c(1,2,3,4)      #kombinovanie prvkov do vektora
> y <- c(5,6,7,8)
> x*y; y/x; y-x; x^y
[1]  5 12 21 32
[1] 5.000000 3.000000 2.333333 2.000000
[1] 4 4 4 4
[1]     1    64  2187 65536
> cos(x*pi) + cos(y*pi)
[1] -2  2 -2  2
> s <- c(1,1,3,4,7,11)
> length(s)            #dĺžka vektora
[1] 6
> sum(s)               #1+1+3+4+7+11
[1] 27
> prod(s)              #1*1*3*4*7*11
[1] 924
> cumsum(s)
[1]  1  2  5  9 16 27
> diff(s)              #1-1, 3-1, 4-3, 7-4, 11-7
[1] 0 2 1 3 4
> diff(s, lag = 2)     #3-1, 4-1, 7-3, 11-4
[1] 2 3 4 7
```

Matica a pole

```
> a <- c(1,2,3,4,5,6,7,8,9,10)
> A <- matrix(a, nrow = 5, ncol = 2)   #napĺňanie matice sa štandardne deje po stĺpcoch
> A
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
```

```
[4,]    4    9
[5,]    5   10
> B <- matrix(a, nrow = 5, ncol = 2, byrow = TRUE)      #napĺňanie po riadkoch
> B
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
[5,]    9   10
> C <- a
> dim(C) <- c(5,2)      #matica je vektor, ktorý dostal dva rozmery
> C <- t(C)             #transponovanie
> C
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    6    7    8    9   10
> B %*% C               #násobenie
     [,1] [,2] [,3] [,4] [,5]
[1,]   13   16   19   22   25
[2,]   27   34   41   48   55
[3,]   41   52   63   74   85
[4,]   55   70   85  100  115
[5,]   69   88  107  126  145
>
> D <- C %*% B
> D
     [,1] [,2]
[1,]   95  110
[2,]  220  260
>
> det(D)
[1] 500
>
> solve(D)        #inverzia matice (riesenie D*x=I, kde I je jednotkova m.)
      [,1]  [,2]
[1,]  0.52 -0.22
[2,] -0.44  0.19
>
> cbind(x,y)      #prilepenie vektorov do stlpcov, resp. do riadkov matice
     x y
[1,] 1 5
[2,] 2 6
[3,] 3 7
[4,] 4 8
> rbind(x,y)
  [,1] [,2] [,3] [,4]
x    1    2    3    4
y    5    6    7    8
>
>
> E <- c(letters,LETTERS)       #zabudovaná databáza písmen v abecednom poradí
> dim(E) <- c(1,26,2)           #z vektora trojrozmerné pole
> E
, , 1
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
[1,] "a"  "b"  "c"  "d"  "e"  "f"  "g"  "h"  "i"  "j"   "k"   "l"   "m"   "n"
     [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26]
[1,] "o"   "p"   "q"   "r"   "s"   "t"   "u"   "v"   "w"   "x"   "y"   "z"

, , 2

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
[1,] "A"  "B"  "C"  "D"  "E"  "F"  "G"  "H"  "I"  "J"   "K"   "L"   "M"   "N"
     [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26]
[1,] "O"   "P"   "Q"   "R"   "S"   "T"   "U"   "V"   "W"   "X"   "Y"   "Z"
```

Vstup údajov.

```
> ovocie <- c("jablko", "hruška", "pomaranč")      #priame zadanie kombináciou
> 1:9           #sekvencia s krokom 1
[1] 1 2 3 4 5 6 7 8 9
> 1.5:10        #nedostaneme sa až po koniec, no ten môžeme pridať
[1] 1.5 2.5 3.5 4.5 5.5 6.5 7.5 8.5 9.5
> c(1.5:10,10)
 [1]  1.5  2.5  3.5  4.5  5.5  6.5  7.5  8.5  9.5 10.0
> seq(-2,5)     #to isté ako -2:5, ale seq() dokáže i viac...
[1] -2 -1  0  1  2  3  4  5
> seq(-2,5,by=.5)       #prírastok o 0.5
 [1] -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0
> seq(-2,5,length=4)  #dĺžka sekvencie bude práve 4
[1] -2.0000000  0.3333333  2.6666667  5.0000000
> rep(9,5)      #hodnotu 9 opakuj päťkrát; to isté ako rep(9,times=5)
[1] 9 9 9 9 9
> rep(1:4,2)
[1] 1 2 3 4 1 2 3 4
> rep(1:4, each = 2)
[1] 1 1 2 2 3 3 4 4
> rep(1:4, each=2, times=3)
 [1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4
> rep(1:4,1:4)
 [1] 1 2 2 3 3 3 4 4 4 4
> matrix(rep(c(1,rep(0,4)),4),nrow=4,ncol=4)   #i takto sa dá vytvoriť jednotková matica
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
> diag(1, nrow=4)       #no takto je to predsalen jednoduchšie :)
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
> vstup <- scan()       #postupne zadávanie z klavesnice (na konci Enter naprázdno)
1: 10
2: 101
3: 1010
4:
```

```
Read 3 items
> vstup
[1]  10 101 1010
>      #načítanie zo súboru, kde čísla sú oddelené medzerou alebo novým riadkom
> vstup <- scan("d:/data/udaje.txt", sep=" ")
Read 9 items
> vstup
[1]   10  101 1010   20  202 2010   30  303 3010
>      #ak si chceme súbor vyhľadať (prvé 2 riadky vynechá)
> vstup <- scan( file.choose(), skip=2)
Read 3 items
> vstup
[1]   30  303 3010
```

Manipulácia s prvkami poľa

```
> z <- c(y,x); z
[1] 5 6 7 8 1 2 3 4
> z[1]
[1] 5
> z[5:8]           #výpis konkrétnych prvkov
[1] 1 2 3 4
> z[c(5,8)]
[1] 1 4
> z[-(2:8)]        #vyberie všetky okrem prvkov zadaných záporným indexom
[1] 5
> z[-c(5,8)]
[1] 5 6 7 8 2 3
> z[8] <- 10       #ôsmemu prvku je priradená iná hodnota
> z > 5            #výsledkom porovnania je vektor logických hodnôt
[1] FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE
> z[z>5]           #výber všetkých prvkov väčších ako 5
[1]  6  7  8 10
> (1:20)[c(TRUE,FALSE)]    #všetky nepárne (vektor logických hodnôt sa replikoval)
 [1]  1  3  5  7  9 11 13 15 17 19
> A
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10
> A[1,2]           #prvok z 1. riadku a 2. stĺpca matice A
[1] 6
> A[1,]            #prvý riadok
[1] 1 6
> A[,2]            #druhý stĺpec
[1]  6  7  8  9 10
```

Operátory V tabuľke 11.1.1 sú operátory zoradené hierarchicky podľa prednosti pri vyhodnocovaní. Zoznam je prevzatý z nápovedy, dostupný príkazom `?Syntax`.

```
> x <- 2
> y <- z <- 1:3
> LO <- c(F,F,F)
> 0 < x & x < 1      #skráteny zápis 0 < x < 1 NEfunguje
[1] FALSE
> x < y | LO
[1] FALSE FALSE  TRUE
> x < y || LO        #zdvojený logický operátor si všíma iba prvý prvok vektorov
[1] FALSE
>
> y == z             #porovnanie po prvkoch
[1] TRUE TRUE TRUE
> identical(y, z)    #porovnanie objektov ako celku
[1] TRUE
> all.equal(y, z)
[1] TRUE
>
> 0.9 == 1.1 - 0.2             #porovnanie numerických hodnôt; prekvapujúce?
[1] FALSE
> identical(0.9, 1.1 - 0.2)
[1] FALSE
> all.equal(0.9, 1.1 - 0.2)
[1] TRUE
> all.equal(0.9, 1.1 - 0.2, tolerance = 1e-16)    #hodnoty sa skutočne drobne líšia
[1] "Mean relative difference: 1.233581e-16"
```

Grafický výstup sa nerealizuje v príklazovom riadku, ale je presmerovaný do zobrazovacieho *zariadenia* (graphic device), ktorým je — štandardne — nové okno, prípadne súbor formátu bitmap, jpeg, png, alebo postscript. Jeho plocha sa dá ďalej členiť na oblasti, z ktorej každá zobrazuje iný graf. Vykreslovať možno jednak grafy funkcií daných explicitným vťahom, ale hlavnou silou R je práca s dátami, teda v prípade funkcie jej diskrétnymi bodmi.

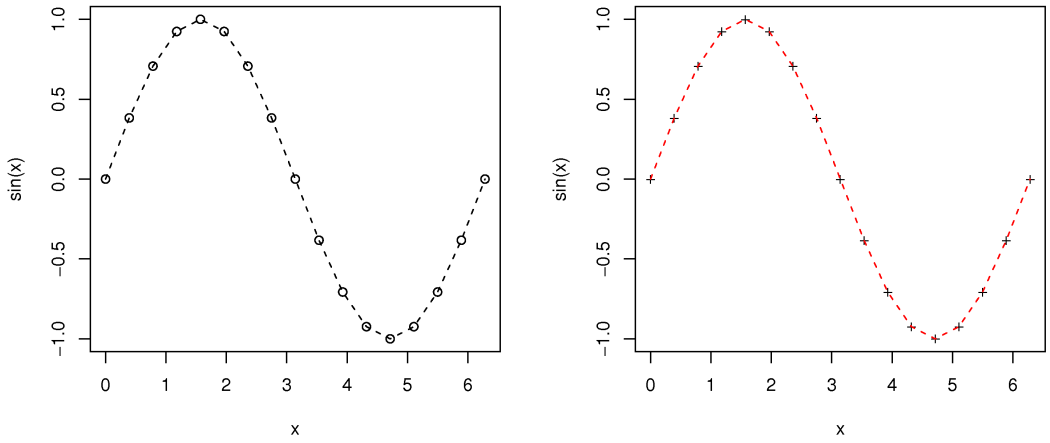
```
> plot(sin, from=0, to=2*pi)      #vykreslenie spojitej funkcie (pozn.: výstup neuvádzame)
> par(mfrow=c(1,2))               #rozdelenie výstupu do tabuľky s 1 riadkom a 2 stĺpcami
> x <- seq(0, 2*pi, length=16+1)          #diskrétne body
> plot(x,sin(x),type="o", lty="dashed") #zobrazenie bodov a (čiarkovaných) spojníc naraz
>                                          #alebo postupne:
> plot(x,sin(x), type="p", pch="+")          #typ 'p' predstavuje body; označ. krížikom
> lines(x,sin(x), col="red", lty="dashed")   # spojnice sú 'prikreslené' červenou
> par(mfrow=c(1,1))               #obnovenie štandardného nastavenia
```

Aby sme výstup presmerovali do súboru, musíme najprv *otvoriť* príslušné grafické zariadenie a po vykonaní grafických príkazov ho znova zavrieť, napr.

```
> jpeg("d:/vystup/graf.jpg", width = 960, height = 480)
  (...)
> dev.off()
null device
          1
```

Tabuľka 11.1.1: Hierarchia operátorov, priorita klesá zhora nadol. Vyhodnotenie sa deje zľava doprava, opačné prípady sú označené (PL).

Operátor	Význam
`[ [[`	indexovanie
`:: :::`	prístup k premenným v externom balíku
`$ @`	výber zložky objektu
`^`	exponovanie (PL)
`- +`	unárne mínus a plus
`:`	operátor postupnosti
`%operator%`	špeciálne operátory
`* /`	násobenie, delenie
`+ -`	binárny operátor súčtu a rozdielu
`< > <= >= == !=`	porovnávanie
`!`	negácia
`& &&`	logické „a“
`| ||`	logické „alebo“
`~`	oddelovač v objekte typu formula
`-> ->>`	pravosmerné priradenie
`=`	priradenie (PL)
`<- <<-`	priradenie (PL)
`?`	nápoveda (unárny aj binárny operátor)



Obr. 11.1.1: Rozdelenie grafického výstupu.

kde (...) bolo nahradené predošlým príkladom s rozdelením výstupu do dvoch stĺpcov. Výsledok je zobrazený na obrázku 11.1.1.

Demonštračné ukážky grafických možností R možno získať príkazmi `demo(graphics)` a `demo(persp)`.

11.1.2 Dátové objekty

R rozlišuje nasledujúce dátové objekty: vektor (*vector*), faktor (*factor*), pole (*array*), matica (*matrix*), dátový rámec (*data frame*), časový rad (*ts*), zoznam (*list*). Sú charakterizované menom, obsahom ale aj atribútmi, ktoré špecifikujú typ obsahu. Základné atribúty sú dĺžka (length) a mód (mode). Módy poznáme: číselný (numeric), znakový (chracter), komplexné číslo (complex), logický (logical), funkcia (function), výraz (expression), a ďašie. Iba data frame a list môžu obsahovať viac ako jeden mód.

Vector. V nasledujúcich príkladoch vytvoríme nový vektor zadaním dĺžky, módu a konkrétneho prvku. Nezadané hodnoty sú doplnené prednastavenými, napr. dĺžka rovná nule alebo ostatné prvky rovné FALSE (ekvivalentom v numerickom móde je nula), príadne nie sú definované vôbec (NA vo význame „not available“).

```
> v <- vector(mode="logical", length=0); v
logical(0)
> v <- logical(); v[2] <- TRUE; v
[1]   NA TRUE
```

```
> v <- logical(3); v[2] <- TRUE; v
[1] FALSE  TRUE FALSE
```

Factor reprezentuje kategoriálnu premennú a okrem samotných realizácií archivuje aj zoznam kategórií (levels).

```
> factor(c(1,2,2,3,2,3))
[1] 1 3 2 2 3 2
Levels: 1 2 3
```

Matrix.

```
> A <- 1:12
> dim(A) <- c(6,2)      #ďalší atribút, nepatrí medzi základné (non-intrinsic)
> t(A)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    2    3    4    5    6
[2,]    7    8    9   10   11   12
>
> matrix(1:12,nrow=2)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    7    9   11
[2,]    2    4    6    8   10   12
```

Data frame. Do dátového rámca zadáme merania k experimentu, v ktorom sa sledovala obsadenosť náhodne vybraných osobných automobilov, pripútanosť pasažierov a pôvod (podľa značky mesta na ŠPZ).

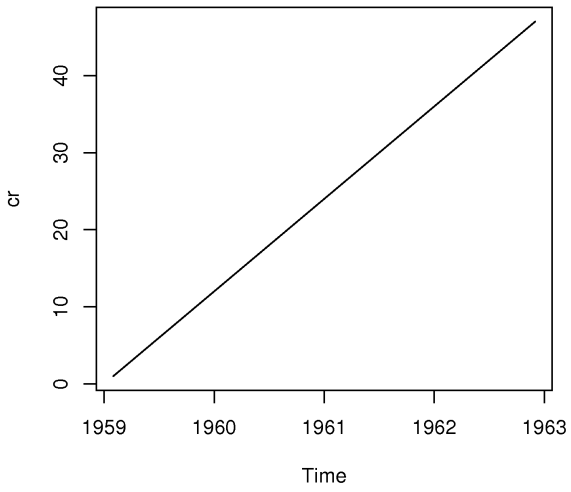
```
> pocet_pasazierov <- c(1,3,2,5,2,2,1,1,2,1)
> priputani <- c(T,T,F,T,F,F,T,F,F,T)
> auta <- data.frame(pocet_pasazierov,priputani) #uloženie do dátového rámca
> auta <- edit(auta)    #realizácie tretej premennej 'mesta' zadáme pomocou editora
> auta
   pocet_pasazierov priputani mesta
1                 1       TRUE    BA
2                 3       TRUE    TT
3                 2      FALSE    TT
4                 5       TRUE    GA
5                 2      FALSE    BA
6                 2      FALSE    BA
7                 1       TRUE    SE
8                 1      FALSE    BA
9                 2      FALSE    BA
10                1       TRUE     W
> str(auta)                                          #vypíše štruktúru objektu
'data.frame':   10 obs. of  3 variables:
 $ pocet_pasazierov: num  1 3 2 5 2 2 1 1 2 1
 $ priputani       : logi   TRUE  TRUE FALSE  TRUE FALSE FALSE ...
 $ mesta           : chr  "BA" "TT" "TT" "GA" ...
```

R obsahuje mnoho vlastných súborov dát, ich zoznam sa zobrazí príkazom `data()`. Načítame napríklad údaje o hrúbke a dĺžke stromov zo súboru „trees“. Rozsiahlejší výpis krátime.

```
> data(trees)
> trees
   Girth Height Volume
1    8.3     70   10.3
2    8.6     65   10.3
3    8.8     63   10.2
  (...)
29  18.0     80   51.5
30  18.0     80   51.0
31  20.6     87   77.0
```

Extrakcia náhodných premenných (stĺpcov) — *subsetting* — je možná niekoľkými spôsobmi:

```
> trees["Girth"]     #názvom
   Girth
1    8.3
2    8.6
3    8.8
  (...)
29  18.0
30  18.0
31  20.6
> trees[1]           #poradím stĺpca
   Girth
1    8.3
2    8.6
3    8.8
  (...)
29  18.0
30  18.0
31  20.6
> trees[[1]]         #poradím v zozname vektorov
 [1]  8.3  8.6  8.8 10.5 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4 11.4 11.7
[15] 12.0 12.9 12.9 13.3 13.7 13.8 14.0 14.2 14.5 16.0 16.3 17.3 17.5 17.9
[29] 18.0 18.0 20.6
> trees$Girth        #pomocou operátora $
 [1]  8.3  8.6  8.8 10.5 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4 11.4 11.7
[15] 12.0 12.9 12.9 13.3 13.7 13.8 14.0 14.2 14.5 16.0 16.3 17.3 17.5 17.9
[29] 18.0 18.0 20.6
> Girth              #premennú ‘Girth’ bez kontextu súboru ‘trees’ systém nepozná,
Error: object "Girth" not found
> attach(trees)      #kým ho explicitne nepripojíme
> Girth
 [1]  8.3  8.6  8.8 10.5 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4 11.4 11.7
[15] 12.0 12.9 12.9 13.3 13.7 13.8 14.0 14.2 14.5 16.0 16.3 17.3 17.5 17.9
[29] 18.0 18.0 20.6
> detach(trees)          #odpojenie databázy
```



Obr. 11.1.2: Časový rad 47 mesačných záznamov.

List nemusí mať rovnaký počet riadkov/stĺpcov (na rozdiel od predošlých), subsetting funguje podobne ako pri data frame.

```
> zoznam <- list(polozka1=c(1,2,3),polozka2=c("hruska","jablko"),polozka3=F)
> zoznam
$polozka1
[1] 1 2 3

$polozka2
[1] "hruska" "jablko"

$polozka3
[1] FALSE
```

Time series umiestňuje vektor dát do časového rámca.

```
> ts(1:10, start=1959)
Time Series:
Start = 1959
End = 1968
Frequency = 1
 [1]  1  2  3  4  5  6  7  8  9 10
> cr <- ts(1:47, frequency = 12, start=c(1959,2))
> ts.plot(cr)
```

Expression umožňuje symbolickú manipuláciu s matematickými výrazmi.

```
> x <- 3; y <- 2.5; z <- 1
> výraz <- expression(x/(y+exp(z)))
> výraz
expression(x/(y + exp(z)))
> eval(výraz)                    #vyhodnotenie výrazu
[1] 0.5749019
> D(výraz, "y")                  #derivácia
-(x/(y + exp(z))^2)
```

Konverzia medzi dátovými objektmi.

```
> as.numeric(c(TRUE,FALSE))
[1] 1 0
> as.numeric("4")
[1] 4
> as.numeric(c("A","Z"))       #iné ako numerické znaky nemajú svoj ekvivalent v číslach
[1] NA NA
Warning message:
NAs introduced by coercion
> as.logical(c(-1,0,1,2))
[1]  TRUE FALSE  TRUE  TRUE
> as.logical(c("FALSE","F"))
[1] FALSE FALSE
> as.logical("A")
[1] NA
> as.character(1)
[1] "1"
> as.character(TRUE)
[1] "TRUE"
> as.numeric( factor(c(0,100,36.7,100)) )      #výsledkom je poradie zodp. kategórií
[1] 1 3 2 3
> as.numeric(as.character( factor(c(0,100,36.7,100)) ))        #sprostredkovane
[1]   0.0 100.0  36.7 100.0
```

11.1.3 Programovanie

Cykly a podmienky. Riadenie sledu príkazov (tzv. control flow) sa v R zabezpečuje pomocou: *if, else, ifelse, for, while, repeat, break, next*. Nasledujúci cyklus vypíše každú hodnotu premennej „a“, ktorá je menšia alebo rovná 10 prípadne rovná 144, pri ktorej prvom výskyte sa cyklus zastaví. Premenná „a“ je generovaná v každej iterácii.

```
> a <- 0                                          #inicializácia
> for (i in 1:20) {                               #stanovenie počtu cyklov
+   a <- i^2                                      #priraď hodnotu pre aktuálne kolo
+   if(a <= 10 ) {                                #testovanie
+     cat('a = ', a, '(<= 10)'); cat('\n')        #výpis
+     next                                        #pokračuj v ďalším kolom
+   }
+   if(a == 144) {
+     cat('a = ', a); cat('\n')
```

```
+     break                                          #ak a=144, ukonči cyklus
+   }
+ }
a =  1 (<= 10)
a =  4 (<= 10)
a =  9 (<= 10)
a =  144
> i              #hodnota iteračnej premennej nie je po skončení cyklu vymazaná
[1] 12
```

Monte Carlo simulácia hodnoty Ludolfovho čísla π.

```
> eps <- 1; s <- 0; n <- 0          #inicializačné hodnoty
> while(eps > .001) {               #opakuj kým 'eps' nebude menšie ako 0.001
+   n <- n + 1                      #počet generovaných bodov
+   x <- runif(1,-1,1)              #súradnice náhodného bodu vo štvorci
+   y <- runif(1,-1,1)                #[-1,-1][1,-1][1,1][-1,1]
+   if(x^2 + y^2 < 1) s <- s + 1    #počet bodov ležiacich vnútri kruhu s polomerom 1
+   pihat <- 4*s/n                  #odhad 'pi'
+   eps = abs(pihat - pi)           #s toleranciou 'eps'
+ }
> pihat        #odhad
[1] 3.140831
> n            #počet iterácií
[1] 987
```

Generovanie náhodných čísel z N(0,1) rozdelenia, až kým číslo nevypadne z intervalu (-2.0,2.0).

```
> repeat {a <- rnorm(1); if (abs(a) > 2.0) break; cat(a); cat("\n")}
-1.438424
1.079813
-0.5798427
-1.418716
-0.1287706
0.2254817
0.004220227
1.391046
0.7339264
-1.585988
1.133099
```

Vlastné funkcie. Až vďaka možnosti definovať funkcie podľa vlastných predstáv sa naplno prejaví potenciál výpočtových systémov ako je R. Premenné definované vo vnútornom prostredí funkcie nie sú viditeľné v nadradenom, tzv. rodičovskom prostredí, z ktorého je funkcia volaná. To bráni vzniku omylov pri väčšom počte používaných premenných. Nasledujúca funkcia 'fun1' vráti maximum dvoch skalárnych čísel (argumenty funkcie) alebo hlášku o ich rovnosti.

```
> fun1 <- function(a, b) {        #'a','b' sú argumenty, medzi { } sa nachádza telo funkcie
+   if(is.numeric(c(a,b))) {
+   if(a < b) return(b)
+   if(a > b) return(a)
+     else print("Hodnoty sú rovnaké")
+   }
+   else print("Akceptujem iba čísla.")
+ }
> fun1(4,7)
[1] 7
> fun1(0,exp(log(0)))
[1] "Hodnoty sú rovnaké"
> fun1("Adam","Eva")      #ak argumenty nie sú z definičného oboru, program na to upozorní
[1] "Akceptujem iba čísla."
```

V prípade, že nemáme potrebu definovať novú funkciu, no trváme na použití lokálnych premenných, môžme postupnosť príkazov uzavrieť do prostredia `local()`.

```
> a <- 0
> local({
+   b <- a
+   a <- 8       #hodnota premennej je prepísaná iba lokálne
+   a + b
+ })
[1] 8
> a
[1] 0
```

Vektorizácia ponúka spôsob, ako sa vyhnúť programovaniu cyklov. V nasledujúcom príklade chceme vypočítať stĺpcové priemery matice M

```
> M <- cbind(rnorm(20,0,1),rnorm(20,-5,1))
```

Klasickým uvažovaním by sme navrhli dva vnorené cykly s postupnou kumuláciou prvkov.

```
> suma <- numeric(m <- NCOL(M)); n <- NROW(M);
> for(i in 1:n) {
+   for(j in 1:m) {
+     suma[j] <- suma[j] + M[i,j]
+   }
+ }
> suma/n
[1] -0.01575158 -5.34533873
```

No efektívnejšie je využiť zameranie systému R na prácu s vektorovými dátami a použiť funkciu `apply()` (v tomto príklade alternatívne aj `colMeans()`).

```
> apply(M, MARGIN=2, FUN=mean)        #pri MARGIN=1 by FUN bola aplikovaná na riadky
[1] -0.01575158 -5.34533873
```

alebo niektorú z jej modifikácií: `lapply()`, `sapply()`, `replicate()`, `mapply()`, `tapply()`. Tiež namiesto vstavanej možno použiť vlastnú (aj nepomenovanú funkciu)

```
> sapply(c(-1,0,5,-5,9), function(x) if(x>0) x else 0)
[1] 0 0 5 0 9
```

alebo danú funkciu pomenovať a explicitne vektorizovať

```
> f2 <- function(prvy, druhy) {
+   if(prvy > druhy) prvy else if(druhy>prvy) druhy else NA
+ }
> f2(prvy=c(-1,2,10),druhy=c(2,2,2))    #testuje iba prvý prvok z oboch argumentov
[1] 2 2 2
Warning messages:
1: In if (prvy > druhy) prvy else if (druhy > prvy) druhy else NA :
  the condition has length > 1 and only the first element will be used
2: In if (druhy > prvy) druhy else NA :
  the condition has length > 1 and only the first element will be used
> f2 <- Vectorize(f2)              #ak ju však vektorizujeme,
> f2(dru=c(2,2,2),pr=c(-1,2,10))    #porovnanie sa vykoná paralelne po všetkých prvkoch
[1]  2 NA 10
```

Všimnime si, že argumenty netreba uvádzať celým menom, ak je ich skratka unikátna. Vektorizovanou náhradou za `if(cond) expr1 else expr2` je `ifelse(cond,expr1,expr2)`, v nasledujúcom príklade má rovnaký efekt aj `pmax()`.

```
> a <- c(-1,2,10); b <- c(2,2,2)
> ifelse(a > b, a, b)
[1]  2  2 10
```

Ďalšie tipy. Ak uložíme definície funkcií a premenných do súboru, povedzme *mojeprogramy.r*, môžme ich kedykoľvek použiť v inom skriptovom súbore pripojením pomocou `source("mojeprogramy.r")`.
Užitočná vec pri hľadaní chýb vo vlastnej funkcii (tzv. debugging) je umiestniť `browser()` do tela funkcie. Po zavolaní našej funkcie nám sprístupní jej prostredie so všetkými lokálnymi premennými.

11.1.4 Ďalšie poznámky

Pracovná plocha. Pokiaľ používateľ preferuje priamu komunikáciu so systémom pomocou klávesnice, prehľad základných príkazov príde vhod. Príkaz `ls()` vypíše zoznam názvov používateľom definovaných a načítaných objektov, `ls.str()` k nim zobrazí aj detaily, pomocou `rm(x)` sa zmaže hodnota premennej „x“, `rm(list=ls())` vymaže všetky

definované objekty alebo `rm(list=ls(part="^m"))` len tie začínajúce znakom „m“. Nápovedu napr. k funkcii „sum“ zavolá `help(fun)` prípadne `?sum`, hľadanie v prehliadači systému nápoveda aktivuje `help()` pričom priamy príkaz na hľadanie všetkých výskytov slova „sum“ v nápovede je `help.search("sum")`. Zoznam nainštalovaných programových knižníc nám poskytne `library()`, pripojenie jednej z nich `library(nazovkniznice)`, naopak jej odpojenie sa vykoná pomocou `detach()`. Prostredie R ukončíme príkazom `q()`, následne nám systém ponúkne možnosť uložiť pracovné prostredie so všetkými definovanými objektami — hibernáciu systému.

Web. Mnoho zaujímavého o R sa dá dozvedieť z elektronických publikácií dostupných na stránke cran.r-project.org v oddelení *Documentation/Contributed*, z nich odporúčame najmä tie uvádzané aj v zozname literatúry k týmto skriptám [3, 4, 6, 7]. Na stránke sa okrem toho nachádza zbierka programových balíkov rozširujúcich základné výpočtové možnosti R, ale aj programy pre jednoduchšiu komunikáciu R s používateľom, ako napr. editor Tinn-R so zvýrazňovaním syntaxe, vlastnou nápovedou, evidenciou objektov a ďalšími podpornými funkciami.

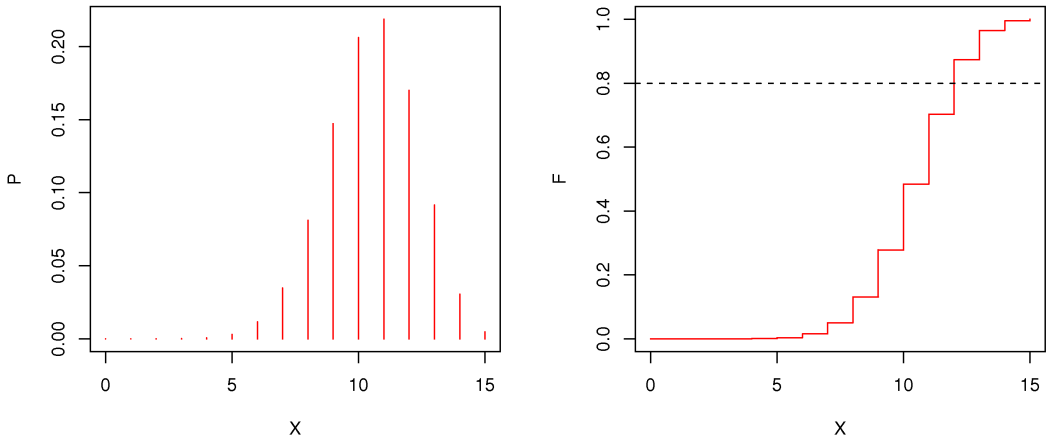
11.2 Základy štatistiky

11.2.1 Náhodná premenná

Binomické rozdelenie. Basketbalista má úspešnosť zásahu 70 %.
a) Nájdite rozdelenie pravdepodobnosti náhodnej premennej X, ktorou je počet úspešných zásahov pri 15 násobnom opakovaní pokusu. Vypočítajte strednú hodnotu a disperziu.
b) S akou pravdepodobnosťou sa basketbalista trafí aspoň 11-krát?
c) Koľkokrát musí trafiť, aby dosiahol 80 % úspešnosť?
d) Nech pri tom istom basketbalistovi je náhodná premenná Y daná súčtom bodov, ktoré získa počas 15 hodov, ak za kôš získa 10 a za minutie koša stratí 5 bodov. Nájdite jej strednú hodnotu.

Riešenie:
Zapíšme všetky vstupné informácie do premenných:

```
X <- 0:15       #všetky hodnoty, ktoré náhodna premenná môže nadobúdať
p <- 0.70       #úspešnosť zásahu
n <- 15       #počet opakovaní pokusu
```



Obr. 11.2.1: Pravdepodobnostná a distribučná funkcia Bi(n,p)

a) Rozdelenie pravdepodobnosti náhodnej premennej X môže byť dané tabuľkou pravdepodobností, alebo predpisom. Kedže zo zadania X je zrejmé, že bude mať binomické rozdelenie, jej pravdepodobnostná funkcia sa získa

```
> P <- choose(n,X)*p^X*(1-p)^(n-X)        #zo vzťahu, alebo
> P <- dbinom(X,n,p)                      #vstavanou funkciou
```

Distribučná funkcia potom bude funkcia kumulatívnych pravdepodobností, teda

```
F <- cumsum(P)          #kumulatívna sumácia, alebo
F <- pbinom(X,n,p)      #distribučná funkcia binomického rozdelenia
```

Namiesto tabuľky si vykreslíme graf pravdepodobnostnej a distribučnej funkcie (Obrázok 11.2.1)

```
par(mfrow=c(1,2))       #nastaví vykreslovanie grafov do mriežky 1x2
plot(X, P, type="h", col="red")
plot(X, F, type="s", col="red"); abline(h=0.8, lty=2)
```

Stredná hodnota a disperzia

```
c(E = sum(P*X), D = sum(P*(X-E)^2))
```

b) P(X > 10) = 1 – P(X ? 10) =

```
> 1-pbinom(10,15,0.7)
[1] 0.5154911
```

c) Zisťujeme 80 % kvantil:

```
> qbinom(0.80,n,p)
[1] 12
```

d)

```
> Y <- 10*X-5*(n-X); Y
[1] -75 -60 -45 -30 -15   0  15  30  45  60  75  90 105 120 135 150
```

Keďže pravdepodobnosti budú rovnaké ako pre príslušné X, potom stredná hodnota bude

```
> sum(P*Y)        #to isté pomocou E(X) z výrazu 15*sum(P*X) - 5*n
[1] 82.5
```

Poissonovo rozdelenie. Náhodná premenná X predstavuje počet pokazených kusov v sérii 400 novovyrobených televízorov. Vieme, že kazovosť prevádzky je 8 kusov na 1000 výrobkov, takisto vieme, že X má Poissonovo rozdelenie pravdepodobnosti. Aká je pravdepodobnosť, že v spomenutej sérii sa vyskytnú a) najviac 2, b) práve 3, c) aspoň 4 poruchové televízory?

Riešenie:
Poznáme

```
> p <- 8/1000; n <- 400
```

a vieme určiť parameter Poissonovho rozdelenia (ktorý je zároveň strednou hodnotou a disperziou)

```
> (lambda <- n*p)
[1] 3.2
```

Potom pravdepodobnostná funkcia Poisonovho rozdelenia je definovaná

```
P <- function(i) lambda^i*exp(-lambda)/factorial(i)    #vzťahom
P <- function(i) dpois(i,lambda)                       #vstavanou funkciou
```

Odpoveď na otázku teda bude

```
> P(0)+P(1)+P(2)        #a) alternatívne ppois(3,lambda)
[1] 0.3799037
> P(3)                  #b)
[1] 0.222616
> 1 - ppois(3,lamda)    #c) alebo ppois(3,lambda,lower.tail=FALSE)
[1] 0.3974803
```

Vzťahom predpísané rozdelenie. Funkcia hustoty pravdepodobnosti f(x) spojitej náhodnej premennej X nadobúda hodnoty $Ksin(2x-2)$ pre $x \in [1, 1+\pi/2]$, mimo tohto intervalu je nulová.

a) Určite konštantu K tak, aby f(x) spĺňala podmienku pre funkciu hustoty,

b) nájdite distribučnú funkciu F(x) a obe funkcie vykreslite,

c) vypočítajte strednú hodnotu, disperziu a 95% kvantil náhodnej premennej X,

d) zistite pravdepodobnosť $P(2 < X < 2.3)$

Riešenie:

a) Celková plocha pod hustotou pravdepodobnosti sa musí rovnať jednej.

```
> fun <- function(k)
+ integrate(function(x) k*sin(2*x-2), lower=1, upper=1+pi/2)$value - 1
> (K <- uniroot(fun, interval=c(0,10))$root)
[1] 1
> f <- function(x) ifelse(1<=x & x<=1+pi/2, K*sin(2*x-2), 0)
```

b) Ručne vypočítame neurčitý integrál (R nemá nástroje na symbolické výpočty) $\int \sin(2x-2)dx = -\cos(2x-2)/2$ a hodnotu takejto funkcie v počiatočnom bode $x = 1$, $-\cos(2*1-2)/2 = -1/2$, potom

```
> F <- function(x) ifelse(x<1, 0, ifelse(x<=1+pi/2,-cos(2*x-2)/2-(-1/2),1))
```

c) stredná hodnota a disperzia:

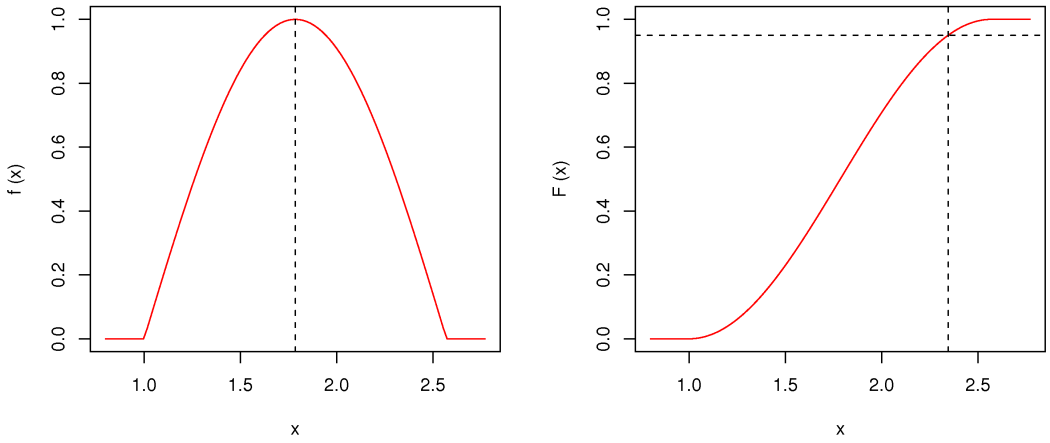
```
> E <- integrate(function(t) t*f(t), lower=-Inf, upper=Inf)$value
> D <- integrate(function(t) (t-E)^2*f(t), lower=-Inf, upper=Inf)$value
> round(c(E,D), digits=4)      #výpis zaokrúhlený na 4 des. miesta
[1] 1.7854 0.1169
```

95 % kvantil q sa určí z riešenia rovnice $F(q) = 0.95$ numerickou motódou

```
> (q <- uniroot(function(x) F(x)-0.95, interval=c(1,3))$root)
[1] 2.345282
```

Graficky to celé vyzerá asi takto (Obrázok 11.2.2)

```
> split.screen(c(1,2))       #predtým treba obnoviť nastavenie par(mfrow=c(1,1))
[1] 1 2
> screen(1); curve(f, from=0.8, to=1.2+pi/2, type="l", col="red");
+ abline(v=E,lty=2)
> screen(2); curve(F, from=0.8, to=1.2+pi/2, type="l", col="red");
+ abline(h=0.95,v=q,lty=2)
> close.screen(all=T)
```



Obr. 11.2.2: Pravdepodobnostná a distribučná funkcia Bi(n,p)

d)

```
> F(2.3)-F(2)
[1] 0.2203710
```

Normálne rozdelenie. Náhodná premenná X, ktorou je % chybných tehiel má normálne rozdelenie pravdepodobnosti so strednou hodnotou $\mu = 19$ a disperziou $\sigma^2 = 9$, teda X $\sim$ N(19,9). a) Vypočítajte pravdepodobnosť toho, že v dodávke tehiel ich bude viac ako 15 % a menej ako 25 % chybných. b) Za aké najnižšie percento chybných tehiel sa môžeme zaručiť s pravdepodobnosťou 0.95?

Riešenie.
Úlohu tohto typu možno riešiť za pomoci vstavaných funkcií R rovnako ako s použitím tabuliek. Ukážeme si obidva spôsoby. Vstupné údaje:

```
> mu <- 19                  #stredná hodnota
> sigma <- sqrt(9)          #štandardná odchýlka
```

a) $P(15 < X < 25) = P(\frac{15-\mu}{\sigma} < Z < \frac{25-\mu}{\sigma}) = P(\frac{15-19}{3} < Z < \frac{25-19}{3}) = \Phi(2) - \Phi(\frac{-4}{3})$, kde $Z = \frac{X-\mu}{\sigma}$ je náhodná premenná s normovaným normálnym rozdelením pravdepodobnosti, teda Z $\sim$ N(0,1), Φ je distribučná funkcia normovaného normálneho rozdelenia a jej hodnoty sú zostavené do tabuliek, ktoré možno nájsť v mnohých publikáciách o matematickej štatistike. V systéme R sa distribučná funkcia normálneho rozdelenia nachádza pod označením 'pnorm' a platí $\Phi(x) = pnorm(x, 0, 1)$. A tak môžme úlohu vyriešiť i bez použitia tabuliek: $P(15 < X < 25) = P(-4/3 < Z < 2)$, teda

```
> pnorm(2,0,1) - pnorm(-4/3,0,1)
[1] 0.8860386
```

alebo ešte jednoduchšie

```
> pnorm(25,mu,sigma) - pnorm(15,mu,sigma)
[1] 0.8860386
```

b) Riešime rovnicu $P(X < q) = 0.95$, kde q nazývame 95 % kvantil rozdelenia pravdepodobnosti. Ak nemáme možnosť vypočítať distribučnú funkciu nenormovaného normálneho rozdelenia pravdepobnosti (napr. pomocou R), musíme opäť prejsť z $\mathrm{N}(\mu, \sigma^2)$ na N(0,1), ktorého hodnoty distribučnej funkcie sú tabelované, teda $P(X < q) = P(-\infty < \frac{X-\mu}{\sigma} < \frac{q-\mu}{\sigma}) = \Phi(\frac{q-\mu}{\sigma}) = 0.95$, a v tabuľke nájdeme hodnotu argumentu x pri hodnote funkcie $\Phi(x) = 0.95$. Bude to presne hodnota 1.645, ktorej sa má rovnať výraz $(q - \mu)/\sigma$. Z toho zistíme, že $q = 1.645\sigma + \mu = 23.935$.
Pomocou R:

```
> qnorm(0.95,mu,sigma)
[1] 23.93456
```

Aritmetický priemer. Nech X $\sim$ N(2,4). a) Vypočítajte pravdepodobnosť toho, že aritmetický priemer realizácií náhodnej premennej X padne do intervalu (1.8, 2.2) v prípade, že sme urobili 100 meraní. b) Koľko meraní musíme vykonať, aby aritmetický priemer padol do tohto intervalu s pravdepodobnosťou 0.95?

Riešenie.
Vstupné údaje:

```
> mu <- 2;                  #stredná hodnota
> sigma <- sqrt(4)          #štandardná odchýlka
> n <- 100                  #počet meraní
```

Aritmetický priemer A je náhodná premenná, $A = \frac{1}{n}\sum_{i=1}^{n} X_i$, ktorej rozdelenie pravdepodobnosti je $\mathrm{N}(\mu, \sigma^2/\mathrm{n})$ za predpokladu, že X má rozdelenie $\mathrm{N}(\mu, \sigma^2)$.

a) $P(1.8 < A < 2.2) = P(\frac{1.8-2}{\sqrt{4/100}} < \frac{A-2}{\sqrt{4/100}} < \frac{2.2-2}{\sqrt{4/100}}) = \Phi(1) - \Phi(-1)$

```
> pnorm(2.2,mu,sigma/sqrt(n)) - pnorm(1.8,mu,sigma/sqrt(n))
[1] 0.6826895
```

b) $0.95 = P(\frac{1.8-2}{\sqrt{4/n}} < \frac{A-2}{\sqrt{4/n}} < \frac{2.2-2}{\sqrt{4/n}}) = \Phi(0.1\sqrt{n}) - \Phi(-0.1\sqrt{n}) = 2\Phi(0.1\sqrt{n}) - 1$, riešime teda rovnicu $\Phi(0.1\sqrt{n}) = 1.95/2$. V tabuľkách či v R možno nájsť 0.975 kvantil N(0,1) rozdelenia:

```
> qnorm(1.95/2,0,1)
[1] 1.959964
```

Zároveň $q = 0.1\sqrt{n}$ a z toho $n = (10q)^2$, teda $n = 384.1$ Alternatívne riešenie počíta koreň rovnice v tvare $f(n) = 0$

```
> uniroot(
+  function(n) pnorm(2.2,mu,sigma/sqrt(n))-pnorm(1.8,mu,sigma/sqrt(n))-0.95,
+  interval=c(1,1000)
+  )$root
[1] 384.1459
```

Musíme urobiť aspoň 385 meraní, aby sa nám aritmetický priemer so spoľahlivosťou 95 % zmestil do intervalu (1.8, 2.2)

11.2.2 Popisná štatistika

Netriedený súbor. Počas 50 týždňov bola sledovaná kazovosť vo výrobe panelov. a) Vypočítajte aritmetický priemer, modus, medián, rozptyl, smerodajnú odchýlku, variačný koeficient a variačný rozsah. b) Zostavte tabuľku absolútnej početnosti (frekvenčná tabuľka) a zobrazte ju ako histogram.

Riešenie:
Počty nekvalitných panelov v jednotlivých týždňoch sa načítajú z textového súboru (sekvencia čísel oddelených medzerou)

```
> panel <- scan("panel.txt"); panel
Read 50 items
 [1] 14 16 11 10  8 13 12 14 16 12 15 13 12 10 16 12 17  9 12 12 14 18 15 13
[25] 17  9 13 11 11 12 15 13 14 13 13  8 10 15 11 11 14 14 11  9 13 10 16 15
[49] 13 12
```

a)Aby sa skrátil výpis všetkých charakteristík, uložíme ich hodnoty do jediného dátového objektu, zaokrúhlime výpis a dočasne sprístupníme lokálne premenné.

```
> info <-  data.frame(
+   min = min(panel),                          #najmenej a
+   max = max(panel),                          #najviac zlých panelov v jednom týždni
+   v.rozsah = max(panel) - min(panel),        #variačný rozsah
+   a = mean(panel),                           #aritmetický priemer
+   median = median(panel),                    #medián
+   rozptyl = var(panel),                      #odhad rozptylu
+   s = sd(panel),                             #a smerodajnej odchýlky
+   v.koef = sd(panel)/mean(panel)             #variačný koeficient
+   )
```

```
> round(info,dig=2)
  min max v.rozsah     a median rozptyl   s v.koef
1   8  18       10 12.74     13    5.75 2.4   0.19
> attach(info)
```

Modus určíme z tabuľky absolútnych početností v úlohe b) ako hodnotu štatistického znaku „panel“ s najvyššou početnosťou.

b) Frekvenčná tabuľka

```
> table(panel)
panel
 8  9 10 11 12 13 14 15 16 17 18
 2  3  4  6  8  9  6  5  4  2  1
```

a histogram spolu s teoretickým rozdelením pravdepodobnosti (škálovaným na absolútne početnosti, Obrázok 11.2.3)

```
> hist(panel,
+  breaks=seq(min-0.5,max+0.5,1),       #hranice tried
+  main="", xlab="pocet chybných panelov", border="red")     #popis a farba
> x <- seq(min-1, max+1, 0.1)
> y <- dnorm(x, mean=a, sd=s)*length(panel)
> lines(x,y,col="blue")      #vykreslí diskrétne body a pospája úsečkami
> detach(info)       #deaktivácia priameho prístupu k premenným v 'info'
```

Z grafu je zjavné, že modus je 13.

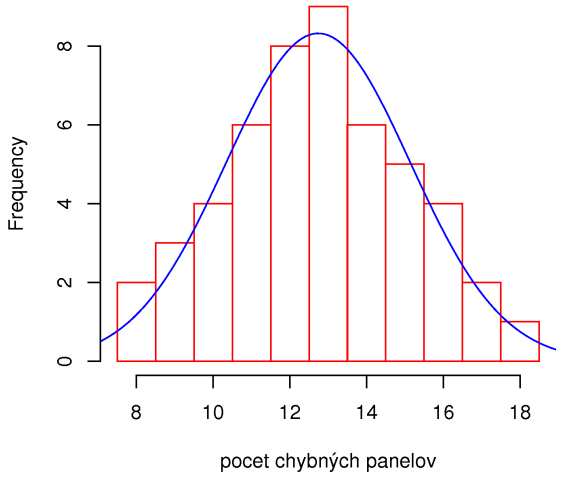
Roztriedený súbor. Majme znova prípad nekvalitných panelov z predošlého príkladu, tentokrát je však namiesto pôvodnej neroztriedenej tabuľky panel daná tabuľka triednych početností. Vypočítajte aritmetický priemer, disperziu a medián.

Vstupné údaje:

```
x <- 8:18                              #triedny znak
p <- c(2,3,4,6,8,9,6,5,4,2,1)          #triedna početnosť
```

Riešenie:

```
> n <- sum(p)                          #celkový počet
> sum(x*p)/n                           #aritmetický priemer (alternatívne: a <- x %*% p / n)
[1] 12.74
> sum((x-a)^2*p)/n                     #disperzia
[1] 5.6324
```



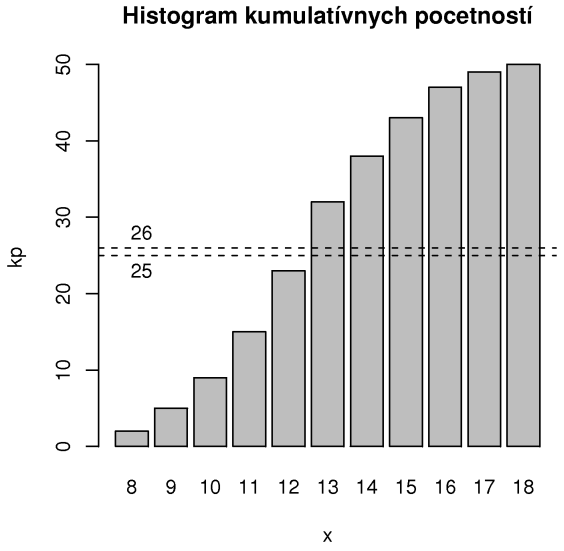
Obr. 11.2.3: Histogram spolu s teoretickým rozdelením

Medián je definovaný ako prostredná hodnota štatistického znaku, ak sú hodnoty znaku usporiadané podľa veľkosti (x'), teda ak $n = 2k+1$ ($k \in N$, n je nepárne), tak $m = x'_{k+1}$, a ak $n = 2k$ (n je párne), tak $m = (x'_k + x'_{k+1})/2$.
Medián môžeme zistiť odčítaním z histogramu kumulatívnych početností (Obrázok 11.2.4),

```
> kp <- cumsum(p)       #kumulatívne početnosti
> barplot(kp, names.arg=x,
+   main="histogram kumulatívnych početností", xlab="x", ylab="kp")
> abline(h=ceiling((n+1)/2), lty=2); text(x=1, y=28, labels="26")
> abline(h=floor((n+1)/2), lty=2); text(x=1, y=23, labels="25")
```

alebo replikovaním triedneho znaku podľa príslušnej početnosti do (usporiadaného) vektora meraní.

```
> rep(x,p)
[1]  8  8  9  9  9 10 10 10 10 11 11 11 11 11 11 12 12 12 12 12 12 12 12 13
[25] 13 13 13 13 13 13 13 13 14 14 14 14 14 14 15 15 15 15 15 16 16 16 16 17
[49] 17 18
> median(rep(x,p))
[1] 13
```



Obr. 11.2.4: Histogram kumulatívnych početností s hladinou pre odčítanie mediánu.

11.2.3 Intervaly spoľahlivosti a testovanie hypotéz o parametroch normálneho rozdelenia

Interval spoľahlivosti. Preverovala sa zdatnosť študentov v skoku do výšky. Výsledky sú zhrnuté v tabuľke početností:

výška: 120 130 140 150 160 170 180 190 200 210

počet: 3 5 7 11 12 6 2 2 1 1

a)Urobte bodový odhad strednej hodnoty a disperzie.

b)Nech má náhodná premenná X (dosiahnutá výška skoku) rozdelenie $N(\mu, \sigma^2)$. Nájdite 95 % obojstranný aj ľavostranný interval spoľahlivosti pre μ

b1) pri známom rozptyle $\sigma^2 = 400$,

b2) pri neznámom σ^2, a určite hneď aj 95 % obojstranný interval spoľ. pre varianciu.

Riešenie:

```
x <- seq(120,210,10)              #dosiahnutá výška (triedny znak)
p <- c(3,5,7,11,12,6,2,2,1,1)     #počet študentov (triedna početnosť)
alpha <- 0.05                     #hladina významnosti
sigma <- 20                       #známa smerodajná odchýlka
n <- sum(p)                       #celkový počet
a <- sum(x*p)/n                   #bodový odhad strednej hodnoty
s2 <- sum((x-a)^2*p)/(n-1)        #bodový odhad disperzie
s <- sqrt(s2)                     #bodový odhad smerodajnej odchýlky
```

a)

```
> c(a = a, s2 = s2)
       a       s2
154.6000 388.6122
```

b1) Obojstranný interval:
$0.95 = 1 - \alpha = P(D < \mu < H) = P(-q < \frac{a-\mu}{\sigma/\sqrt{n}} < q)$, kde q je kvantil N(0,1) rozdelenia na hladine významnosti α. Vypočítame ju

```
> q <- qnorm(1-alpha/2,0,1); q
[1] 1.959964
```

alebo nájdeme v tabuľkách pod označením $u_{1-\alpha/2}$ (kvantil) prípadne ako kritickú hodnotu pre $\alpha/2$. Potom 95 % interval spoľahlivosti je

```
> D <- a - q*sigma/sqrt(n)       #alebo priamo: qnorm(alpha/2,a,sigma/sqrt(n))
> H <- a + q*sigma/sqrt(n)       #podobne qnorm(1-alpha/2,a,sigma/sqrt(n))
> cat("<",D,",",H,">\n")
[ 149.0564 , 160.1436 ]
```

Ľavostranný interval:
$0.95 = 1 - \alpha = P(D < \mu < \infty) = P(-q < \frac{a-\mu}{\sigma/\sqrt{n}} < \infty)$, rozdiel je v tom, že teraz sa celé α presunie pod ľavý chvost gaussovej krivky.

```
> q <- qnorm(1-2*alpha/2,0,1); q
[1] 1.644854
> D <- a - q*sigma/sqrt(n)       #alebo qnorm(alpha,a,sigma/sqrt(n))
> cat("[",D,", Inf )\n")
[ 149.9477 , Inf )
```

b2) Obojstranný interval
$1 - \alpha = P(D < \mu < H) = P(-q < \frac{a-\mu}{s/\sqrt{n}} < q)$, kde q je kritická hodnota Studentovho t-rozdelenia na hladine významnosti $\alpha/2$ alebo kvantil

```
> q <- qt(1-alpha/2,n-1); q
[1] 2.009575
```

Všimnime si, že parametrami t-rozdelenia nie je stredná hodnota ani rozptyl, ale tzv. stupne voľnosti, v našom prípade $n-1$. Z nerovníc podobne ako v príklade b1) dostaneme dolnú a hornú hranicu 95 % intervalu spoľahlivosti

```
> D <- a - q*s/sqrt(n)
> H <- a + q*s/sqrt(n)
> cat("[",D,",",H,"]\n")
[ 148.9976 , 160.2024 ]
```

Ľavostranný interval:

```
> q <- qt(1-alpha,n-1)
> D <- a - q*s/sqrt(n)
> cat("[",D,", Inf )\n")
[ 149.926 , Inf )
```

Upozorňujeme, že namiesto smerodajnej odchýľky σ sme použili jej bodový odhad s.

Intervalový odhad disperzie: Ak $X \sim \mathrm{N}(\mu, \sigma^2)$, potom platí $\frac{(n-1)s^2}{\sigma^2} \sim \chi^2(n-1)$. Hľadáme také D a H, aby platilo $P(D < \sigma < H) = 1 - \alpha$. Úpravou $P(D < \sigma < H) = P(q_1 < \frac{(n-1)s^2}{\sigma^2} < q_2) = P(\frac{1}{q_1} > \frac{\sigma^2}{(n-1)s^2} > \frac{1}{q_2}) = P(\frac{(n-1)s^2}{q_1} > \sigma^2 > \frac{(n-1)s^2}{q_2})$, kde q_1 a q_2 sú kvantily $\chi^2(n-1)$ rozdelenia.

```
> q1 <- qchisq(alpha/2,n-1)
> q2 <- qchisq(1-alpha/2,n-1)
> c(q1,q2)
[1] 31.55492 70.22241
> D <- (n-1)*s^2/q2
> H <- (n-1)*s^2/q1
> cat("[",D,",",H,"]\n")
[ 271.167 , 603.4559 ]
```

Test hypotézy o strednej hodnote. V meste LM je známa fabrika na výrobu alkoholu. V náhodnom výbere 500 domácností tohto mesta sa sledovala spotreba alkoholických nápojov v priebehu roka. Z náhodného výberu sa výpočítal aritmetický priemer 18.9 litra a smerodajná odchýlka 8.5 litra. Celoštátna priemerná ročná spotreba alkoholu na jednu domácnosť je 17.8 litra. Na hladine významnosti 5 % testujte hypotézu, že prítomnosť fabriky nemá vplyv na vyšší alkoholizmus obyvateľov mesta a teda spotreba v meste sa nelíši od celoštátneho priemeru.

Riešenie:

```
n <- 500
a <- 18.9
s <- 8.5
mu0 <- 17.8
alpha <- 0.05
```

Nulová hypotéza H0: $a = \mu 0$ Alternatívna hypotéza H1: $a > \mu 0$ (alternatíva je jednostranná!)

Postup riešenia úlohy je podobný ako pri určovaní intervalu spoľahlivosti, zmenila sa iba „filozofia“ zadania úlohy. Musíme vypočítať testovaciu štatistiku — označme ju TS —

a zistiť, či padne do intervalu ohraničeného (v našom prípade jednou) kritickou hodnotou KH, teda TS $<$ KH. Ak nie, teda ak TS $>$ KH, potom nulovú hypotézu H0 zamietame. Testovacou štatistikou je normovaný aritmetický priemer (porovnaj s úlohou b2 v predošlom príklade, pravostranný interval spoľahlivosti pre strednú hodnotu) a kritickou hodnotou je $t_{\alpha,n-1}$ (kvantil t-rozdelenia).

```
> TS <- (a-mu0)*sqrt(n)/s       #testovacia štatistika
> KH <- qt(1-alpha,n-1)         #kritická hodnota
> c(TS,KH)
[1] 2.893735 1.647913
```

Odpoveď:

```
> if(TS > KH)
+  cat("TS > KH, zamietame H0\n") else
+  cat("TS < KH, zamietame H1\n")
TS > KH, zamietame H0
```

Na hladine významnosti 5% zamietame hypotézu o rovnosti stredných hodnôt, teda prítomnosť fabriky pravdepodobne zvyšuje spotrebu alkoholu v meste.

11.2.4 Testovanie odľahlých hodnôt

Grubbsov a Dixonov test. Zisťovala sa hmotnosť porobetónových tvárnic. Vo výsledkoch (v kg)

`5.83 5.80 5.85 5.88 5.84 5.83 5.98 5.78 5.82 5.81 5.86 5.82`

vzbudila hodnota 5.98 podozrenie, že ide o hrubú chybu merania. Zistite

a) Grubsovým T-testom,

b) Dixonovým Q-testom

na hladine významnosti 0.01, či hodnotu treba zo súboru vylúčiť.

Riešenie:

```
> y <- c(5.83,5.80,5.85,5.88,5.84,5.83,5.98,5.78,5.82,5.81,5.86,5.82)
> alpha <- 0.01
> n <- length(y)
> a <- mean(y)
> s <- sd(y)
```

Usporiadajme si vektor meraní vzostupne (od najmenšieho po najväčší prvok)

```
> z <- sort(y); z
[1] 5.78 5.80 5.81 5.82 5.82 5.83 5.83 5.84 5.85 5.86 5.88 5.98
```

a) V prípade najväčšej hodnoty z_n testovaciu štatistiku $G = \frac{z_n - a}{s}$

```
> G <- (z[n]-a)/s; G
[1] 2.705069
```

porovnávame s kritickou hodnotou, ktorú možno nájsť v tabuľkách pre Grubbsov test (kG = 2.551), alebo približne vypočítať (pre jednostranný test nasledovne)

```
> kG <- (n-1)/sqrt(n) * sqrt(qt(alpha/n,n-2)^2/(n-2+qt(alpha/n,n-2)^2)); kG
[1] 2.549417
```

b) Pri Dixonovom teste netreba a ani s, jeho sila je však menšia. Testovaciu štatistiku $Q = \frac{z_n - z_{n-1}}{z_n - z_1}$

```
> Q <- (z[n]-z[n-1])/(z[n]-z[1]); Q
[1] 0.5
```

porovnávame s tabelovanou kritickou hodnotou kQ = 0.482 (test je iba obojstranný).

V oboch testoch vyšla testovacia štatistika väčšia ako kritická hodnota, preto hodnotu $z_n = 5.98$ vylúčime so súboru meraní.

Poznámka k riešeniu:
V R existuje knižnica funkcií (treba ju doinštalovať) pre testovanie odľahlých hodnôt,

```
> library(outliers)
```

a tá obsahuje aj obidva uvedené testy. Kritické hodnoty získame príkazmi

```
> qgrubbs(1-alpha, n, type=10)
```

```
> qdixon(alpha, n, type=10)
```

Celý test vykonajú funkcie

```
> grubbs.test(y, type = 10)

        Grubbs test for one outlier

data:  y
G = 2.7051, U = 0.2743, p-value = 0.002613
alternative hypothesis: highest value 5.98 is an outlier
```

a

```
> dixon.test(y, type = 10)

        Dixon test for outliers

data:  y
Q = 0.5, p-value = 0.0144
alternative hypothesis: highest value 5.98 is an outlier
```

Argument `type=10` spôsobí testovanie iba jednej hodnoty. Či pôjde o maximum alebo minimum, o tom rozhodne program sám, prepínačom `opposite = TRUE` môžeme toto rozhodnutie zmeniť. Výstupom funkcií je okrem testovacej štatistiky aj p-hodnota (*p-value*) a znenie alternatívnej hypotézy. Tú „prijímame“, keď je p-hodnota väčšia ako vopred stanovená hladina významnosti α.

11.3 Príklady pre pokročilých

11.3.1 Komplexný príklad

Test dobrej zhody, jedno- a dvojvýberový test. Výrobca automobilov A testoval spotrebu na 100 km pri jednom type jeho automobilov. Bolo testovaných celkom 50 automobilov. Namerané spotreby sú

```
> a <- c(7.7,6.8,5.0,9.8,7.4, 8.7,6.3,8.0,8.6,6.4, 8.5,7.7,7.7,7.9,7.7,
+ 9.3,6.0,7.0,6.3,6.7, 6.3,6.5,8.7,6.7,7.6, 6.7,7.5,9.6,7.2,6.6,
+ 7.3,7.2,6.3,6.6,5.6, 6.4,8.4,7.7,7.3,7.4, 6.6,8.8,9.2,9.8,7.6,
+ 7.5,8.1,8.6,6.8,8.8)
```

a) Aká bola priemerná spotreba a jej rozptyl?

```
> cat(mu <- mean(a), sigma <- sd(a), "\n")
7.498 1.104257
```

b) Chi-kvadrát testom testujte hypotézu H0, že spotreba má Normálne rozdelenie s parametrami z bodu a).

```
> histogram <- hist(a, breaks=5, plot=FALSE)
> hranice <- histogram$breaks
> poc_skutocne <- histogram$counts
> cat(hranice,"\n",poc_skutocne,"\n")      #hranice tried a skutočné početnosti
5 6 7 8 9 10
 3 16 17 9 5
> m <- length(hranice) - 1; n <- length(a)      #počet tried; rozsah súboru
> hranice[1] <- -Inf; hranice[m+1] <- Inf
```

```
> poc_teoreticke <- diff(pnorm(hranice, mean=mu, sd=sigma))*n
> cat(hranice,"\n",poc_teoreticke,"\n")      #hranice tried a teoretické početnosti
-Inf 6 7 8 9 Inf
 4.372961 11.92710 17.46509 11.89061 4.344241
> stat <- sum((poc_skutocne-poc_teoreticke)^2/poc_teoreticke)    #chi^2 štatistika
> 1 - pchisq(stat, df=m-2-1)      #p-hodnota > 5 %, preto H0 nezamietame
[1] 0.2676748
> chisq.test(poc_skutocne, p=poc_teoreticke/n)$p.value
[1] 0.6204655
```

Pozn.: Vstavaná funkcia `chisq.test` uvažuje iba s $m-1$ stupňami voľnosti (akoby parametre rozdelenia boli známe), preto p-hodnotu vypočíta nesprávne.

c) Prijali sme hypotézu H0 (namerané hodnoty spotreby majú Normálne rozdelenie). Určite neznáme parametre Normálneho rozdelenia a ich intervaly spoľahlivosti.

```
>      #bodový odhad strednej hodnoty a s.odchýlky
> cat(mu, sigma,"\n")
7.498 1.104257
>      #intervalový odhad
> cat("mu: [",mu+c(-1,1)*qt(1-0.05/2,n-1)*sigma/sqrt(n-1),"]\n")
mu: [ 7.180987 7.815013 ]
> cat("sigma: [",n*sigma^2/qchisq(c(1-0.05/2,0.05/2),n-1),"]\n")
sigma: [ 0.8682297 1.932161 ]
```

d) Výrobca automobilov B uviedol na trh nový automobil rovnakej triedy ako výrobca A a tvrdí, že jeho automobil má nižšiu spotrebu. Za rovnakých podmienok boli pri 30 automobiloch výrobcu B namerané spotreby:

```
> b <- c(8.9,6.1,7.6,6.5,8.4, 6.7,7.1,6.1,10.1,5.1, 6.6,5.8,7.7,6.1,7.5,
+ 5.6,9.4,11.1,7.4,10.0, 5.3,7.3,6.7,3.8,6.7, 8.6,6.9,7.9,7.5,7.1)
```

Predpokladajme (a prípadne Shapirovým-Wilkovým testom normality skontrolujte), že aj táto spotreba má normálne rozdelenie pravdepodobnosti. Má výrobca B pravdu?

```
> shapiro.test(b)$p.value      #test normality (nevyžaduje triedenie), nezamietame H0
[1] 0.6573306
> var.test(a, b)$p.value              #test rovnosti rozptylov, zamietame H0
[1] 0.02445645
>      #test rovnosti strednych hodnot, zamietame H1: mu(a)>mu(b)
> t.test(a, b, alternative="greater", var.equal=(.Last.value>0.05))$p.value
[1] 0.2305746
```

Výrobca B s 95% pravdepodobnosťou nemá pravdu.

e) Výrobca A sa rozhodol vybaviť svoje automobily novým typom karburátora, ktorý by mal významne znížiť spotrebu paliva. Testoval ho na prvých 10 automobiloch s výsledkami:

```
> a1 <- c(7.4,6.5,5.1,9.6,7.3, 8.5,6.3,8.0,8.4,6.3)
```

Má skutočne nový karburátor vplyv na zníženie spotreby?

```
>       #párový t-test, prijimame H1: mu(a')>mu(a1)
> t.test(a[1:10], a1, alternative="greater", paired=TRUE)

        Paired t-test

data:  a[1:10] and a1
t = 3.0736, df = 9, p-value = 0.006638
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 0.05246801        Inf
sample estimates:
mean of the differences
                   0.13
```

Nový karburátor má vplyv na zníženie spotreby.

11.3.2 Testy dobrej zhody — jednovýberové a párové

Kolmogorovov-Smirnovov test. Študent informatiky mal naprogramovať generátor náhodných čísel. Jeho algoritmus je však príliš pomalý, k dispozícii je preto iba prvých desať vygenerovaných hodnôt:

```
> a <- c(0.42, 0.93, 0.30, 0.89, 0.24, 1.00, 0.80, 0.66, 0.79, 0.90)
```

Rozhodnite, či výber je na hladine významnosti 10 % naozaj z R(0,1), teda rovnomerného rozdelenia na intervale [0,1].

```
> ks.test(a,"punif",min=0,max=1)       #punif - distribučná fcia rovn. (uniform) rozdelenia

        One-sample Kolmogorov-Smirnov test

data:  a
D = 0.39, p-value = 0.07018
alternative hypothesis: two-sided
```

Algoritmus možno na hladine 10 % významnosti považovať za generátor náhodných čísel s R(0,1) rozdelením (no na zvyčajnej 5 % hladine by už testom neprešiel).

χ^2 test. Bolo urobených 120 pokusov s klasickou hracou kockou, pozorované početnosti boli zapísané do tabuľky (prvý riadok tvoria hodnoty premennej „x“, druhý riadok početnosť „poc“). Na hladine významnosti 0.01 overte podozrenie, že kocka je falošná.

```
> x <- c(1,2,3,4,5,6)
> poc <- c(13,17,22,13,13,42)
```

Budeme testovať hypotézu H0, že rozdelenie pravdepodobnosti je *rovnomerné* s pravdepodobnosťami 1/6, oproti alternatíve, že nie je.

```
> chisq.test(poc, p=rep(1/6,6))     #zamietame H0 a teda kocka je falošná

        Chi-squared test for given probabilities

data:  poc
X-squared = 32.2, df = 5, p-value = 5.423e-06
```

Pozn.: Ak sú z merania k dispozícii iba „surové“ dáta, riešenie zahŕňa aj vytvorenie tabuľky početností pomocou table(), teda celý zápis bude (zjednodušene)

```
chisq.test(table( c(1,6,6,2,3,1,4,...) ), p=rep(1/6,6))
```

Wilcoxonov test. V podniku uvažujú o zavedení nového databázového systému. Aby rozhodli medzi dvoma kandidátmi A a B, je zostavená séria desiatich najbežnejších požiadaviek na systém a meria sa čas ich vybavenia (v sek.):

```
a <- c(0.18,2.45,5.32,1.38,0.47,0.83,0.57,0.55,1.04,1.94)
b <- c(0.17,2.65,5.56,1.47,0.46,0.86,0.53,0.56,1.15,2.11)
```

Rozhodnite, ktorý databázový systém je pre potreby podniku efektívnejší.

```
> shapiro.test(b)$p.value      #test normality
[1] 0.00883098
> wilcox.test(a-b, conf.int=T)      #to iste ako: wilcox.test(a,b,paired=T,conf.int=T)

        Wilcoxon signed rank test

data:  a - b
V = 8, p-value = 0.04883
alternative hypothesis: true location is not equal to 0
95 percent confidence interval:
 -1.550000e-01 -1.387779e-17
sample estimates:
(pseudo)median
         -0.08
```

Pozn.: Problém s výpočtom presnej p-hodnoty nastane, ak sa vyskytnú nulové alebo rovnaké rozdiely $(a - b)$. Vtedy nevypočíta ani interval spoľahlivosti.

Na základe zamietnutia hypotézy o normálnom rozdelení súboru sme použili neparametrický, jednovýberový Wilcoxonov test. Z významne záporného stredného rozdielu $a - b$ vyplýva, že databázový systém A je efektívnejší.

11.3.3 Testy dobrej zhody — dvojvýberové

t-test, Wilcoxon, Kolmogorov-Smirnov. Na porovnanie presnosti dvoch diaľkomerných prístrojov sa rovnaká vzdialenosť merala 15-krát prvým a 20-krát druhým prístrojom, pričom odchýlky od 100m boli tieto (v cm):

```
> a <- c(-0.13,-0.13,0.15,-0.03,-0.27,-0.22,0.30,0.24,-0.28,0.14,
+ 0.14,-0.34,0.12,0.35,0.47)
> b <- c(-0.17,-0.36,-0.03,0.31,-0.09,0.21,-0.06,-0.38,0.30,-0.39,
+ -0.34,-0.31,-0.36,0.23,-0.09,0.05,-0.17,-0.34,-0.18,0.12)
```

Otestujte, či obidva prístroje merajú s rovnakým rozptylom (vnútorná presnosť prístroja, *precision*). Je pravda, že prvý prístroj meria presnejšie (vonkajšia presnosť, *accuracy*)?

```
> cat("a: ", mean(a),sd(a), " b: ", mean(b), sd(b),"\n")      #bodové odhady
a:  0.034 0.2535125  b:  -0.1025 0.2388321
```

a) Test na posúdenie vnútornej presnosti

```
> var.test(a, b)     #test rovnosti rozptylov

        F test to compare two variances

data:  a and b
F = 1.1267, num df = 14, denom df = 19, p-value = 0.7934
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.425668 3.223211
sample estimates:
ratio of variances
          1.126713
```

Rozptyly oboch súborov sa zjavne rovnajú, teda diaľkomery majú rovnakú vnútornú presnosť. Tento fakt využijeme v nasledujúcom parametrickom teste rovnosti stredných hodnôt dvoch výberov (t-test).

b) Testy na posúdenie vonkajšej presnosti (Pre porovnanie odchýliek je potrebné zjednotiť smer vzdialenosti od nuly):

```
> a <- a*sign(mean(a)); b <- b*sign(mean(b))      #znamienkové škálovanie
> t.test(a, b, var.equal=T, alternative="less")

        Two Sample t-test

data:  a and b
t = 1.63, df = 33, p-value = 0.9437
alternative hypothesis: true difference in means is less than 0
```

```
95 percent confidence interval:
      -Inf 0.2782194
sample estimates:
mean of x mean of y
   0.0340   -0.1025

> wilcox.test(a, b, alternative="less")     #'Mann-Whitney' test

        Wilcoxon rank sum test with continuity correction

data:  a and b
W = 199.5, p-value = 0.9523
alternative hypothesis: true location shift is less than 0

Warning message:
In wilcox.test.default(a, b, alternative = "less") :
  cannot compute exact p-value with ties
> ks.test(a, b, alternative="greater")

        Two-sample Kolmogorov-Smirnov test

data:  a and b
D^+ = 0, p-value = 1
alternative hypothesis: the CDF of x lies above that of y

Warning message:
In ks.test(a, b, alternative = "greater") :
  cannot compute correct p-values with ties
```

Pozn.: Alternatíva „less“ v prvých dvoch testoch znamená nerovnosť $H1 : \mu(a) < \mu(b)$. V poslednom teste to zodpovedá hodnote „greater“, čo znammená, že (graf) distribučná funkcia rozdelenia pravdepodobnosti prvého súboru leží nad distr. funkciou druhého.

Všetky testy na obvyklých hladinách významnosti (5 % aj 10 %) zamietajú alternatívnu hypotézu, teda nemožno povedať, že by prvý prístroj bol presnejší.

11.3.4 Analýza rozptylu (ANOVA), viacvýberové testy

1-faktorová ANOVA, Kruskall-Wallis, Tukey Bola testovaná účinnosť troch druhov insekticídov (A,B,C), výsledkom je počet kusov usmrteného hmyzu v každom pokuse.

```
A <- c(7,7,4,5); B <- c(9,10,14,11); C <- c(6,4,2,4)
```

a) Na hladine 0.05 rozhodnite, či všetky tri druhy majú rovnakú účinnosť.

```
    #vektor meraní účinností a ich prísluchajúce triedy (typy) insekticídov
> Ucinnost <- c(A,B,C)
> Insekticid <- factor( c(rep("A",4),rep("B",4),rep("C",4)) )
> ANOVA <- aov(Ucinnost ~ Insekticid)      #1. riešenie pomocou analýzy rozptylu
```

```
> summary(ANOVA)
            Df  Sum Sq Mean Sq F value    Pr(>F)
Insekticid   2 106.167  53.083  16.617 0.0009519 ***
Residuals    9  28.750   3.194
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>kruskal.test(Ucinnost,Insekticid)      #2. riešenie pomocou Kruskal-Wallis testu

        Kruskal-Wallis rank sum test

data:  Ucinnost and Insekticid
Kruskal-Wallis chi-squared = 8.4947, df = 2, p-value = 0.01430
```

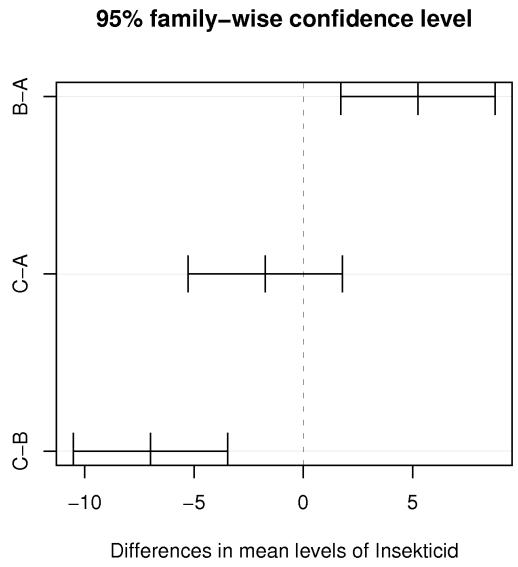
V obidvoch riešeniach zamietame hypotézu H0 o rovnosti stredných hodnôt, teda dané druhy insekticídov nemajú rovnakú účinnosť.

b1) Pomocou Tukeyho metody rozhodnite, ktoré triedy sa odlišujú.

```
> tapply(Ucinnost,Insekticid,mean)      #triedne priemery
    A     B     C
 5.75 11.00  4.00
> TukeyHSD(ANOVA)      #insekticíd typu B je výrazne účinnejší než ostatné dva
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Ucinnost ~ Insekticid)

$Insekticid
     diff        lwr       upr     p adj
B-A  5.25   1.721428  8.778572 0.0062697
C-A -1.75  -5.278572  1.778572 0.3882270
C-B -7.00 -10.528572 -3.471428 0.0009429

> plot(TukeyHSD(ANOVA))       #grafické znázornenie konfidenčných intervalov z Tukeyho testu
```

b2) O odlišnosti konkrétnych tried sa presvedčte aj pomocou t-testu porovnaním po dvojiciach.

```
      test homogenity (rovnosti disperzií)
> bartlett.test(list(A,B,C))     #alebo: bartlett.test(Ucinnost ~ Insekticid)

        Bartlett test of homogeneity of variances

data:  list(A, B, C)
Bartlett's K-squared = 0.3973, df = 2, p-value = 0.8199

> pairwise.t.test(Ucinnost, Insekticid,
+ p.adjust.method = "bonferroni", var.equal = TRUE)

        Pairwise comparisons using t tests with pooled SD

data:  Ucinnost and Insekticid
```



Obr. 11.3.1: konfidenčné intervaly rozdielov stredných hodnôt

```
  A      B
B 0.0074 -
C 0.5985 0.0011

P value adjustment method: bonferroni
```

Pozn.: p-hodnotu je treba prispôsobiť väčšiemu počtu výberov - použili sme Bonferroniho metódu

Rovnosť stredných hodnôt je v oboch testoch zamietnutá pre dvojice A-B, B-C. Z toho vyplýva, že insekticíd typu B je účinnejší ako ostatné dva, ktoré sú na tom s účinnosťou rovnako.

2-faktorová ANOVA s interakciami Skúmala sa adaptabilita potkanov v laboratórnom bludisku v závislosti od genetickej dispozície a motivácie. Tri druhy potkanov (A,B,C), dva druhy návnad (zemiak a syr), výsledok každého merania predstavuje počet pokusov (skóre) potrebných na zvládnutie cesty až k návnade.

```
> adaptabilita <- data.frame(
+     Skore = c(4,4,2,3,6,4,6,8,5,4,8,11),
+     Navnada = c(rep("zemiak",6),rep("syr",6)),
+     Rasa = c("A","A","B","B","C","C","A","A","B","B","C","C")
+     )      #namerané údaje je výhodné uložiť do 'data.frame' objektu
```

a) Ktoré faktory majú vplyv na rýchlejšie učenie potkanov?

b) Majú jednotlivé druhy odlišné preferencie návnad?

```
> with(adaptabilita, tapply(Skore,list(Navnada,Rasa),mean))     #triedne priemery
       A   B   C
syr    7 4.5 9.5
zemiak 4 2.5 5.0
> ANOVA <- aov(Skore ~ Navnada * Rasa, data = adaptabilita)
> summary(ANOVA)
             Df  Sum Sq Mean Sq F value   Pr(>F)
Navnada       1 30.0833 30.0833 19.0000 0.004776 **
Rasa          2 28.1667 14.0833  8.8947 0.016044 *
Navnada:Rasa  2  3.1667  1.5833  1.0000 0.421875
Residuals     6  9.5000  1.5833
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Na efektivitu učenia má vplyv okrem návnady aj genetika (v rámci skúmaných druhov). Interakcia medzi týmito dvoma faktormi sa nepotvrdila.

11.3.5 Kontingenčná tabuľka

χ^2 **test nezávislosti.** Farmaceutická firma sa dostala k dotazníku 32 pacientov, do ktorého sa okrem odpovedí zapisovali aj výsledky zdravotníckych vyšetrení. Záznamy boli zostavené do stĺpcov tabuľky, kde každý riadok zodpovedá jednému pacientovi. Pre marketing firmy je zaujímavý najmä súvis názoru na jednu konkrétnu otázku so systolickým tlakom. Stĺpec zodpovedajúci názoru obsahuje skratky „s“ (súhlasí), „v“ (váha), „n“ (nesúhlasí). Normálny systolický tlak je v rozmedzí 101 – 140.

```
> názor <-  c("v","s","n","s","n","s","n","v","s","v","v","s","n","v","s",
+ "s","n","n","s","v","s","n","n","v","n","s","n","s","v","n","s","v")
> tlak <- c(180,158,159,134,70,150,113,118,79,138,140,133,100,132,83,
+ 149,73,99,87,122,166,85,135,71,115,155,173,146,172,84,152,139)
```

Môžme pri hladine významnosti 0.05 tvrdiť, že súvis existuje?

```
> tlak <- cut(tlak,
+   breaks=c(0,100,140,max(tlak)),
+   labels=c("nízky","normál","vysoký"))      #triedenie kvantit. premennej
> názor <- factor(názor,
+   levels=c("s","v","n"),
+   labels=c("súhlas","váhanie","nesúhlas"))
> table(názor,tlak)     #kontingenčná tabuľka
          tlak
názor      nízky normál vysoký
  súhlas       3      2      7
  váhanie      1      6      2
  nesúhlas     6      3      2
```

Kontingenčná tabuľka dáva tušiť, že početnosti sú príliš malé (<5) pre spoľahlivosť nasledujúceho chi-kvadrát testu nezávislosti, použijeme ho aspoň ilustračne.

```
> chisq.test(.Last.value)      #.Last.value odkazuje na kontingenčnú tabuľku

        Pearson's Chi-squared test

data:  .Last.value
X-squared = 10.4441, df = 4, p-value = 0.03358

Warning message:
In chisq.test(.Last.value) : Chi-squared approximation may be incorrect
```

Zamietame H0 o nezávislosti, takže názor na danú tému súvisí s krvným tlakom – žeby politika? :).

11.3.6 Korelácia a regresia

Test významnosti Spearmanovho korelačného koeficientu. Bolo sledovaných 10 žiakov, na základe psychologického vyšetrenia boli títo žiaci zoradení podľa psychickej lability (čím labilnejší, tým vyššie poradie). Okrem toho dostali poradie aj na základe svojich výsledkov v matematike (najlepší dostal poradie 1).

```
> labilita <- c(1,2,3,4,5,6,7,8,9,10)
> matematika <- c(9,3,8,5,4,2,10,1,7,6)
```

Testom významnosti Spearmanovho korelačného koeficientu otestujte závislosť medzi labilitou a matematickým myslením.

```
> cor.test(labilita, matematika, method = "spearman")

        Spearman's rank correlation rho

data:  labilita and matematika
S = 186, p-value = 0.7329
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho
-0.1272727
```

Pozn.: Vstupné dáta nemusia byť zadané vo forme poradí, algoritmus výpočtu Spearmanovho korelačného koeficientu túto konverziu implicitne zahŕňa.

Funkcia je použiteľná aj pre testovanie Pearsonovho či Kendallovho koeficientu korelácie.

Súvis medzi psychickou labilitou a matematickým nadaním sa nepotvrdil.

Lineárna a nelineárna regresia, predpoveď. Vedenie podniku zamýšľa investovať do nákupu najmodernejších atrakcií s cieľom zvýšiť svoje zisky. K rozhodnutiu však pristúpi až po vypracovaní analýzy, ktorej cieľom je zistiť, či zavedenie nových atrakcií priláka postačujúci počet návštevníkov. Z 11 ročnej histórie zábavného parku sú známe údaje o počte atrakcií a zodpovedajúcom priemernom týždennom počte návštevníkov (v tisícoch):

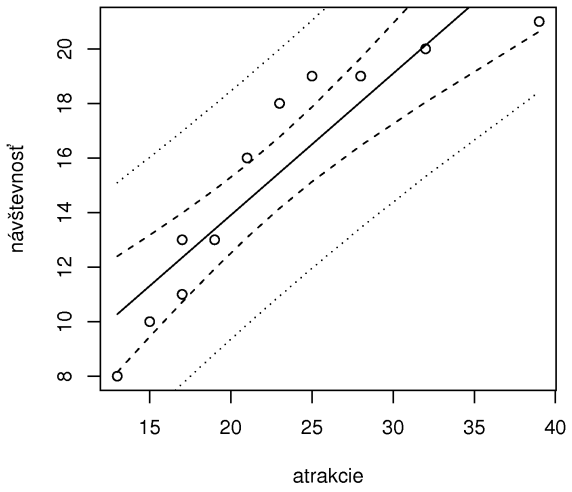
```
> atrakcie <- c(13,15,17,17,19,21,23,25,28,32,39)
> návštevnost' <- c(8,10,11,13,13,16,18,19,19,20,21)
```

a) Odhadnite parametre lineárnej regresnej funkcie. S rizikom 5 % overte, či počet atrakcií štatisticky významne ovplyvňuje návštevnosť (na základe sklonu regresnej priamky).

```
> m.lin <- lm(návštevnost' ~ atrakcie)
> summary(m.lin)

Call:
lm(formula = návštevnost' ~ atrakcie)

Residuals:
    Min      1Q  Median      3Q     Max
-2.7662 -1.3282 -0.1329  1.2600  2.5385

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.52344    1.83337   1.922   0.0868 .
atrakcie     0.51904    0.07687   6.752 8.34e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.915 on 9 degrees of freedom
Multiple R-squared: 0.8351,     Adjusted R-squared: 0.8168
F-statistic: 45.59 on 1 and 9 DF,  p-value: 8.345e-05
```

Sklon regresnej priamky (0.51904) je štatisticky významný (p-hodnota=8.34e-05 $<$ 0.05).

```
> cbind(bod.odhad=coef(m.lin),confint(m.lin))      #bodový a int. odhad parametrov
            bod.odhad      2.5 %    97.5 %
(Intercept) 3.5234398 -0.6239312 7.6708108
atrakcie    0.5190448  0.3451502 0.6929394
> plot(atrakcie,návštevnost')
>      #interval spoľahlivosti pre regresnú priamku
> matlines(atrakcie,predict(m.lin,interval="confidence"),lty=c(1,2,2),col=1)
>      #interval spoľahlivosti okolo regresnej priamky
> matlines(atrakcie,predict(m.lin, interval="prediction")[,-1],
+   lty=c(3,3),col=1)
Warning message:
In predict.lm(m.lin, interval = "prediction") :
  Predictions on current data refer to _future_ responses
```



Obr. 11.3.2: Pásy spoľahlivosti „pre“ (· · ·) a „okolo“ (- - -) regresnej priamky

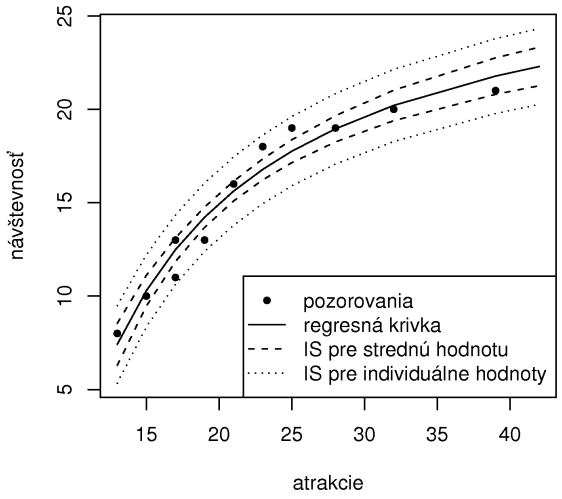
b) Spomedzi bežne používaných nelineárnnych regresných funkcií (exponenciálna, reciproká, logaritmická, mocninová, ...) zvoľte takú, ktorá najlepšie vystihuje pozorované údaje.

```
> vzt'ah <- eval(substitute(
+ c(y~x, log(y)~x, y~I(1/x), I(1/y)~x, log(y)~log(x), y~log(x)),
+ list(y=quote(návštevnost'),x=quote(atrakcie))
+ )) #zásobník výrazov
> sapply(vzt'ah, function(x) summary.lm(lm(x))$r.squared)
[1] 0.8351389 0.7689090 0.9590315 0.6849710 0.8813736 0.9232924
> m.nelin <- lm(vzt'ah[[which.max(.Last.value)]])
```

Na základe indexu determinácie (R^2) najvhodnejšia je reciproká funkcia $y = a + b * 1/x$.

c) Akú návštevnosť s pravdepodobnosťou 0.9 môže vedenie parku v priemere očakávať, ak zvýši počet atrakcií na 42?

```
> predict(m.nelin,newdata=data.frame(atrakcie=42),interval="prediction",level=0.9)
> atrakcieP <- append(atrakcie,42)
> matplot(atrakcieP,
+  cbind(
+   predict(m.nelin,
+       newdata=data.frame(atrakcie=atrakcieP),
+       interval=c("confidence"),
+       level=0.9),
+   predict(m.nelin,
+       newdata=data.frame(atrakcie=atrakcieP),
+       interval=c("prediction"),
+       level=0.9)[,-1]),
```



Obr. 11.3.3: Nelineárna regresia — pásy spoľahlivosti a predpoveď

```
+  type="l",lty=c(1,2,2,3,3),col=1,lwd=1,
+  xlab="atrakcie",ylab="návštevnost'")
> points(atrakcie,návštevnost',pch=20)
> legend("bottomright",
+   c("pozorovania","regresná krivka",
+   "IS pre strednú hodnotu","IS pre individuálne hodnoty"),
+   pch = c(20,-1,-1,-1),lty=c(-1,1,2,3))
```

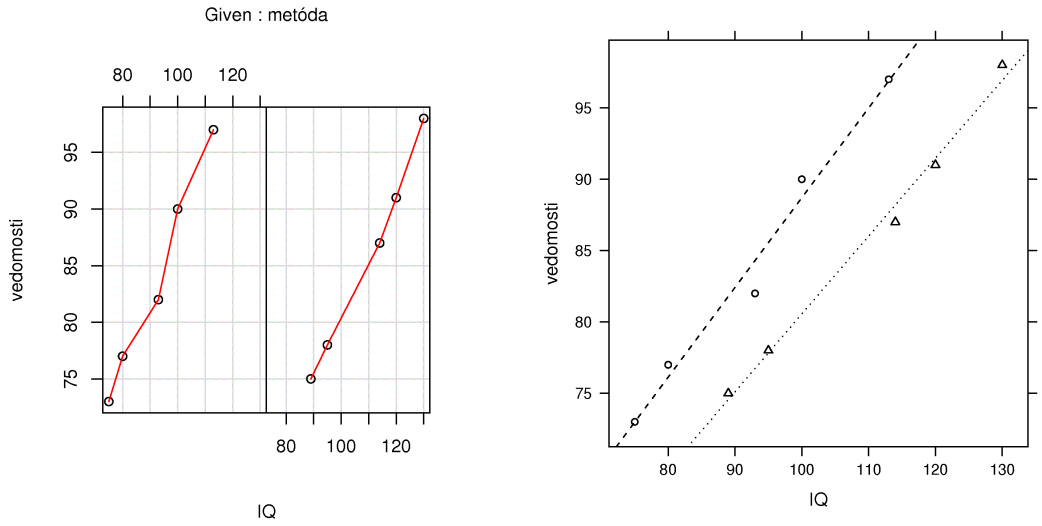
Po zvýšení počtu atrakcií na 42 možno s pravdepodobnosťou 90 % očakávať 20.3 až 24.3 tisíc návštevníkov za týždeň, očakávaná návštevnosť je 22.3 tisíc.

11.3.7 Analýza kovariancie

Skúmala sa efektivita dvoch učebných metód A a B. Desať študetov bolo náhodne rozdelených do dvoch skupín, v každej bola aplikovaná iná metóda. Nakoniec každý študent absolvoval vedomostný test vyhodnotený v percentách. Ešte pred samotným experimentom sa študenti podrobili IQ testu.

```
> metóda <- factor( rep(c("A","B"), each=5) )
> vedomosti <- c(73,77,82,90,97, 75,78,87,91,98)
> IQ <- c(75,80,93,100,113, 89,95,114,120,130)
```

Sú metódy rovnako úspešné?



a) b)

Obr. 11.3.4: Závislosť v skupinách a) štandardnou funkciou a b) z balíka 'lattice'

```
> tapply(vedomosti,metóda,mean)
   A    B
83.8 85.8
> summary(aov(vedomosti ~ metóda)) #alebo anova(lm(vedomosti ~ metóda))
            Df Sum Sq Mean Sq F value Pr(>F)
metóda       1   10.0    10.0  0.1091 0.7497
Residuals    8  733.6    91.7
> coplot(vedomosti ~ IQ | metóda, panel=panel.smooth, show.given=F)
> lattice::xyplot(vedomosti ~ IQ, groups = metóda,
+ type=c("p","r"), lty=c(2,3), pch=c(1,2), col=1)
```

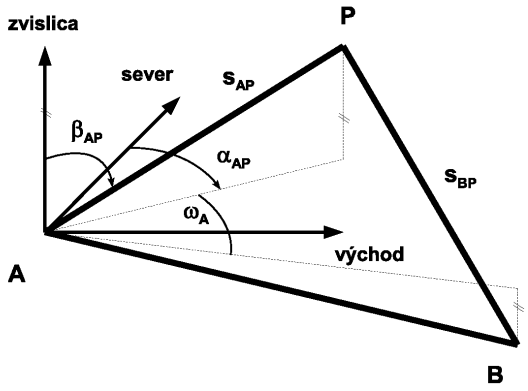
Zo suchého porovnania priemerov by sme dedukovali, že druhá metóda je „ochlp" lepšia, pre ANOVA však už rozdiely medzi metódami nie sú štatisticky významné. Ako vidieť z grafov závislostí (obrázok 11.3.4) medzi získanými vedomosťami a IQ je v každej skupine viditeľný lineárny vzťah, čo značí, že nestačí iba porovnávať skupinové stredné hodnoty.

V analýze kovariancie budeme potrebovať naledujúce 3 modely:

```
> mREG <- lm(vedomosti ~ IQ)       #klasický regresný model bez vplyvu faktora (metódy)
> mANCOVA <- lm(vedomosti ~ metóda + IQ)       #jednoduchý model ANCOVA
> mINTER <- lm(vedomosti ~ metóda * IQ)   #model s interakciami
```

Pozn.: Alternatívny zápis modelu s interakciami: lm(vedomosti metóda + IQ + metóda:IQ)

Testovanie hypotézy rovnobežnosti a sklonu regresných priamok v skupinách (predpoklady ANCOVA).



Obr. 11.3.5: Náčrt zamerania bodu P

```
> anova(mINTER)    #H0: b1 = b2 = b (t.j. zmiešaný člen je v modeli nevýznamný)  prijímame
> anova(mANCOVA)   #H0: b = 0 (t.j. nie je závislosť medzi vedomosťami a IQ) zamietame
```

Test zhody skupinových priemerov (hlavný test ANCOVA)

```
> anova(mREG,mANCOVA)    #H0: a1 = a2 = 0 zamietame, teda metódy nie su rovnako účinné
> coefficients(lm(vedomosti ~ metóda + IQ - 1)) #metóda A má vyššiu úspešnosť
```

Uvážením vplyvu faktora „IQ“ bolo možné odhaliť vplyv faktora „metódy“ na vedomosti študentov.

11.3.8 Vyrovnanie sprostredkujúcich meraní a elipsoid spoľahlivosti

Vyrovnanie sprostredkujúcich meraní a elipsoid spoľahlivosti Z dvoch bodov, ktorých poloha bola určená metódou GPS (a súradnice transformované do lokálneho horizontálneho topocentrického súr. systému), sa meral horizontálny uhol, zenitový uhol a dve vzdialenosti (podľa Obrázku 11.3.5) na bod P. Uhly sú dané v grádovej uhlovej miere (100 g = pravý uhol)

```
> A <- c(100,100,100)                          #súradnice bodu A a B
> B <- c(65.780,212.521,74.669)
> kovA <- matrix(c(
+   10,5,17,
+   5,10,17,
+   17,17,100),ncol=3)*10^6   #kovariančná matica určenia polohy bodu A a B
> kovB <- matrix(c(
+   10,5,17,
+   5,10,17,
```

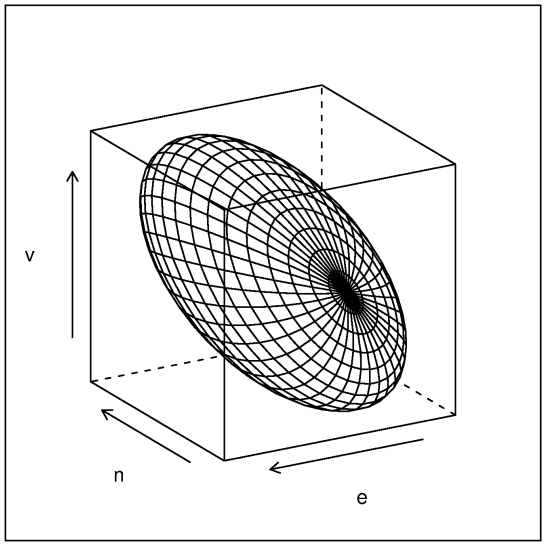
```
+   17,17,100),ncol=3)*10^6
>      #merané veličiny so strednými chybami:
> oA <- 36.9849*pi/200; moA <- 0.0003*pi/200      #horizontálny uhol omegaA
> bAP <- 69.9614*pi/200; mbAP <- 0.0005*pi/200      #zenitový uhol betaAP
> sAP <- 98.495; msAP <- 0.002      #vzdialenosť z A na P
> sBP <- 95.875; msBP <- 0.002      #vzdialenosť z B na P
> merane <- c(oA,bAP,sAP,sBP); mmerane <- c(moA,mbAP,msAP,msBP)
```

Metódou najmenších štvorcov určite súradnice bodu P a elipsoid spoľahlivosti pre jeho polohu v danom súradnicovom systéme.

```
      #Približné súradnice bodu P.
> P0 <- A + c(
+   sAP*sin(bAP)*cos( atan2(B[2]-A[2],B[1]-A[1]) - oA ),
+   sAP*sin(bAP)*sin( atan2(B[2]-A[2],B[1]-A[1]) - oA ),
+   sAP*cos(bAP)
+   )
>      #Merané veličiny ako funkcie neznámych parametrov v symbolickom tvare.
> merane.expr <- c(
+   oA = expression(atan((eB-eA)/(nB-nA))-atan((eP-eA)/(nP-nA))),
+   bAP = expression(atan(sqrt((nP-nA)^2+(eP-eA)^2)/(vP-vA))),
+   sAP = expression(sqrt((nP-nA)^2+(eP-eA)^2+(vP-vA)^2)),
+   sBP = expression(sqrt((nP-nB)^2+(eP-eB)^2+(vP-vB)^2))
+   )
>      #Lokálne priradenie hodnôt symbolom, klasifikácia neznámych a daných parametrov.
> hodnoty <- data.frame(nP=P0[1],eP=P0[2],vP=P0[3],
+ nA=A[1],eA=A[2],vA=A[3],nB=B[1],eB=B[2],vB=B[3])
> nezname.symb <- c("nP","eP","vP")
> dane.symb <- c("nA","eA","vA","nB","eB","vB")
>      #Merané veličiny výpočtom z približných hodnôt neznámych parametrov.
> merane.0 <- c(
+   oA = evalq(atan2(eB-eA,nB-nA)-atan2(eP-eA,nP-nA),hodnoty),
+   bAP = evalq(atan2(sqrt((nP-nA)^2+(eP-eA)^2),vP-vA),hodnoty),
+   sAP = eval(sAP.expr,hodnoty)
+   sBP = eval(sBP.expr,hodnoty) )
> #Pozn.: Funkcia atan2() rieši uhly v kvadrantoch, no nie je derivovateľná.
>      #Funkcia pre vytvorenie matice funkčných hodnôt 'fun(xi,yj)' pre všetky kombinácie indexov (i,j)
> kombinacie <- function(x,y,fun) {
+   komb <- expand.grid(x,y)
+   matrix( mapply(fun,komb[,1],komb[,2]), nrow=length(x))
+   }
>      #Matica plánu (regresná matica) - derivácie funkčných vzťahov podľa neznámych parametrov.
> X <- kombinacie(
+   1:length(merane.expr),
+   1:length(nezname.symb),
+   function(i,j) eval(D(merane.expr[i],nezname.symb[j]),hodnoty)
+   )
>      #Vektor absolútnych členov.
> y <- merane.0 - merane
>      #Kovariančná matica meraných parametrov rozšírená o neistotu v určení daných parametrov
> Sm <- diag(mmerane^2)
> Sd <- rbind(cbind(kovA,diag(c(0,0,0))), cbind(diag(c(0,0,0)),kovB))
> Z <- kombinacie(
+   1:length(merane.expr),
+   1:length(dane.symb),
```

```
+   function(i,j) eval(D(merane.expr[i],dane.symb[j]),hodnoty)
+   )      #derivácie funkčných vzťahov podľa daných parametrov
> S <- Sm + Z %*% Sd %*% t(Z)
> W <- solve(S)       #plná váhová matica (inverzná ku kovariančnej matici)
>      #Odhad prírastkov neznámych parametrov metódou najmenších štvorcov.
> dP <- solve(t(X) %*% W %*% X) %*% t(X) %*% W %*% y
>      #Bodový odhad súradníc bodu P
> P <- P0 + dP
> cbind(P0,dP,P)
           P0
[1,] 124.7276 -0.0001093047 124.7275
[2,] 184.1757  0.0002155401 184.1759
[3,] 144.7690  0.0009881253 144.7700
>      #Elipsoid spoľahlivosti.
> v <- X %*% dP + y      #opravy meraných veličín
> s0 <- c(t(v) %*% W %*% v)/(length(merane)-length(P))      #jednotkový rozptyl
> Sn <- s0 * solve(t(X) %*% W %*% X)   #kovariančná matica neznámych parametrov
> Sn * 10^6      #v mm^2
             [,1]         [,2]     [,3]
[1,]  0.967758397 -0.006001412 1.009065
[2,] -0.006001412  2.048901669 1.678885
[3,]  1.009065008  1.678885234 7.272656
> polosi <- sqrt(eigen(Sn)$values); polosi
[1] 0.002810818 0.001286822 0.000855985
> smery <- eigen(Sn)$vectors       #vlastné vektory tvoria stĺpce matice
>      #Vizualizácia.
> phi <- seq(-100,100,10)*pi/200; lambda <- seq(0,400,10)*pi/200
> n.el <- kombinacie(phi,lambda,function(šírka,dĺžka) c(smery[1,]%*%
+ (polosi*c(cos(šírka)*cos(dĺžka),-cos(šírka)*sin(dĺžka),sin(šírka)))))
> e.el <- kombinacie(phi,lambda,function(šírka,dĺžka) c(smery[2,]%*%
+ (polosi*c(cos(šírka)*cos(dĺžka),-cos(šírka)*sin(dĺžka),sin(šírka)))))
> v.el <- kombinacie(phi,lambda,function(šírka,dĺžka) c(smery[3,]%*%
+ (polosi*c(cos(šírka)*cos(dĺžka),-cos(šírka)*sin(dĺžka),sin(šírka)))))
> lattice::wireframe(v.el~n.el*e.el,
+   screen = list(z = 120, x= -70),
+   perspective=F,
+   xlab="n",ylab="e",zlab="v",
+   # alpha=0.7, scales = list(arrows = FALSE)
+   )
```

Elipsoid spoľahlivosti je zobrazený na Obrázku 11.3.6.



Obr. 11.3.6: Elipsoid spoľahlivosti v súr. sústave (n,e,v)

Literatúra

[1] Anděl, J.: Matematická statistika. SNTL,Alfa, Praha, 1985.

[2] Dallosová, A., Mesiar, R.: Pravdepodobnosť a matematická štatistika. SVŠT, Stavebná fakulta, 1987.

[3] Farnsworth, G.V.: Econometrics in R. cran.r-project.org, 2006.

[4] Owen, W.J.: The R Guide. cran.r-project.org, 2007.

[5] Pacáková, V. a kol.: Štatistika pre ekonómov: Zbierka príkladov, Iura Edition, Bratislava, 2005.

[6] Paradis, E.: R for beginners. cran.r-project.org, 2007.

[7] Verzani, J.: Simple R. www.math.csi.cuny.edu/Statistics/R/simpleR/, 2008.

[8] Tichý, Z., Škrášek, J.: Základy aplikované matematiky III, SNTL, Praha, 1990.

[9] Zvára, K., Štepán, J.: Pravděpodobnost a matematická statistika, Matfyzpress, Praha, 1997.

Svoj názor a pripomienky prosím zašlite na tomas.bacigal[ZAVINAC]stuba.sk.