KAPITOLA 2

Príprava a spracovanie údajov, úvod do {R}

Ciele kapitoly
Kroky pri spracovaní údajov
Prostredie {R}
Objekty
Jednoduché operácie s údajmi a premenné
Štruktúry údajov
Čítanie údajov zo súborov
Pomocné programy
Súhrn
Literatúra

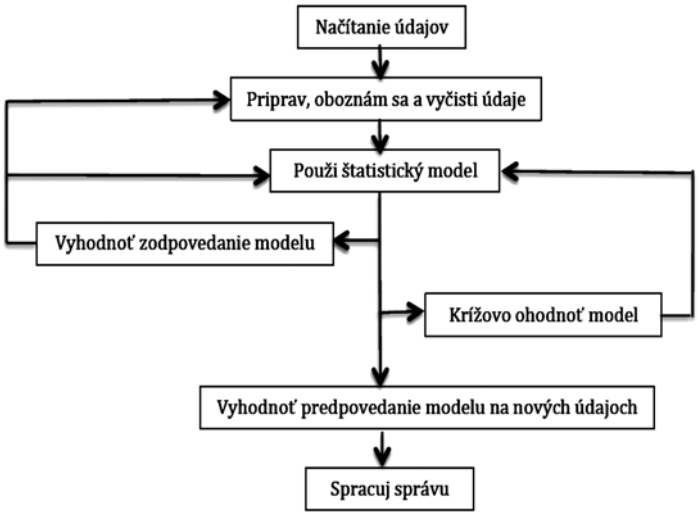
Ciele kapitoly

V tejto kapitole popíšeme základné kroky pri príprave údajov a tiež použitie nástrojov prostredia {R}, Excel, a databázové funkcie. V úvode uvádza spôsoby spracovania údajov, ktoré sú časté pri práci s dátami, charakterizujú stav zdravia populácie či služieb a s ním súvisia. Kroky pri zbere údajov, ako aj často používanú dotazníkovú metódu sme podrobne popísali v publikácii o bioštatistike. [1] Tu poukážeme najmä na otázky, ktoré súvisia so spracovaním údajov z oficiálnych zdrojov, najmä národných či medzinárodných štatistík.

Cieľom tejto kapitoly je poskytnúť čitateľovi prehľad nástrojov prostredia {R} projektu, ktoré mu umožnia narábať s jednoduchými, ale aj zložitejšími dátovými štruktúrami účelne, pri využití čo najširších možností, ktoré {R} ponúka. Ich využitie predpokladá trochu predstavivosti, keďže usporiadanie dát je najlepšie vizualizovať na ploche či v priestore. Aj keď autori majú snahu vyhýbať sa formálnym matematickým zápisom, dvojrozmerné a viacrozmerné dátové štruktúry narábajú s indexmi a matematický zápis je zvyčajne najvhodnejší na zaznamenanie návodu na prácu.

Kroky pri spracovaní údajov

Postupnosť krokov pri spracovaní údajov sa líši v závislosti od ich zdroja. V tejto publikácii sa zameriavame najmä na tie, ktoré boli zozbierané a spracované niektorou z inštitúcií, ktoré sa zaoberajú hlavne dokumentovaním vývoja v niektorej z oblastí zdravia. Z tohto dôvodu nekladieme dôraz na prípravu prvotných údajov, keďže našimi primárnymi dátami sú čísla, ktoré pochádzajú zo súborov rutinného zberu štatistických údajov. Predpokladáme, že čistenie údajov už prebehlo a údaje sú predspracované, overené a validované. Prvým krokom potom býva načítanie údajov z existujúcich súborov (dnes najčastejšie dostupných z internetu) (Obrázok 1).



Obrázok 1 Kroky pri spracovaní údajov podľa [2]

Načítanie

Pri práci v prostredí {R} projektu je načítanie údajov zo súboru uľahčené funkciami. Podrobne sa im venujeme v tejto kapitole po oboznámení sa so základmi {R} a s dátovými štruktúrami. Tu si iba zopakujeme, že súbory údajov môžu mať rôzny formát a medzi najčastejšie patria databázové súbory či Excel alebo textové súbory, kde sú použité oddeľovače, napríklad čiarka alebo tabelátor.

Transformácia

Transformácia býva potrebná najčastejšie pri práci s údajmi, ktoré majú rozmer. Teda napríklad prevody medzi jednotkami z inej ako metrickej sústavy či prepočty mien alebo kombinácia dvoch a viacerých premenných na novú, napríklad výpo-

čet Body Mass Index. Tu sa využijú postupy narábania s údajmi, ktoré sú uvedené nižšie. Špecifickým príkladom transformácie údajov je práca s dátumom. Často sa snažíme získať z dátumu narodenia vek či vypočítať dĺžku trvania deja v rokoch, mesiacoch, týždňoch či dňoch. Základy uvedených postupov si priblížime v ďalších podkapitolách.

Prehľad

Analýza by mala začínať dôsledným štúdiom údajov. Keď budeme predpokladať, že vstupné údaje pochádzajú z verifikovaných zdrojov, nemusíme sa zaoberať ich kvalitou. Vždy je však dobré si uvedomiť, ako boli údaje získané a pri interpretácii si byť vedomí obmedzení, vyplývajúcich zo zdroja. Napríklad pri práci s dátami o hlásených ochoreniach (infekčných či neinfekčných) je potrebné si uvedomiť, že z rôznych dôvodov neboli všetky prípady zachytené. Napríklad kvapavka sa často nehlási, pretože si to pacient nepraje zo spoločenských dôvodov. Mnohé chrípkové ochorenia sa nezachytia, pretože sa chorý lieči sám a nežiada o práceneschopnosť. Bolo opakovane dokázané, že práceneschopnosť je silne ovplyvnená príjmom pracujúceho a ľudia s vyššími príjmami majú tendenciu prekonať ochorenie bez využitia práceneschopnosti.

K prehľadu o údajoch sa zvykne pristupovať ich tabeláciou, či zobrazením pomocou bodového diagramu (scatterplot), histogramu či inou formou vizualizácie. V zhode s Rothmanom et al. [3] by sme chceli upozorniť na často zavádzajúce štatistické parametre, ktoré automaticky ponúkajú niektoré štatistické programy (p-hodnota, intervaly istoty). Tie v štádiu, keď sa študujúci orientuje a rozhoduje o ďalšom postupe môžu nesprávne ovplyvniť rozhodnutie. Zároveň podrobné štúdium pôvodných údajov môže naznačiť veľa o metódach trendovej analýzy či sezonalite, alebo viesť ku hypotézam, ktoré môžeme neskôr potvrdiť či vyvrátiť. V neposlednom rade takéto štúdium môže odhaliť niektoré chyby, ktoré unikli spracovateľom pôvodných údajov, aj keď sa to stáva zriedka pri údajoch z profesionálnych zdrojov.

Štatistická analýza

Štatistické testovanie hypotéz a odhady je potrebné vykonávať s rozvahou a s vedomím možných systematických chýb (bias). Závery je dôležité robiť po odstránení čo najviac potenciálne možných zdrojov skreslenia, používajúc štandardizáciu údajov či iný z univariatných alebo multivariatných postupov. Dobrá znalosť zvoleného postupu štatistickej inferencie (odvodzovania) či zvoleného modelu je predpokladom formulácie správneho záveru. Opatrnosť je potrebná pri úsudkoch o súvislostiach a o vplyvoch premenných medzi sebou. Uvedomenie si možnej chyby (I. alebo II. typu) vedie skúmajúceho k pravdepodobnostnej interpretácii a nedovoľuje kategorické výroky o prítomnosti či neprítomnosti javu.

Správa

Hoci nie je cieľom tejto publikácie zaoberať sa písaním správ o zistených faktoch, považujeme za potrebné uviesť niekoľko poznámok. V prvom rade chceme zdôrazniť potrebu formulácie správ o výsledkoch výskumu formou čo najjednoduchšou a čo najzrozumiteľnejšou. Jasná definícia cieľov výskumu, založená na poznaní stavu poznania vo svete, je základnou podmienkou úspešnej správy (za úspešnú považujeme takú správu, ktorá prinesie nový poznatok). Takáto definícia cieľov potom vedie aj k správnemu použitiu metód výskumu, kvalitnej prezentácii výsledkov a v konečnom dôsledku k vecnej diskusii. Na Slovensku rozšírené členenie správ na teoretickú a praktickú časť jednoznačne neodporúčame, pretože narúša logický sled štruktúrovanej správy a zvádza k prázdnemu a neodôvodnenému hromadeniu referencií (v lepšom prípade, v horšom k plagiátorstvu).

Prostredie {R}

Terminológia

Vzhľadom na to, že všetky príklady sú vykonané pomocou volaní funkcií napísaných pre prostredie {R}, považujeme za potrebné uviesť všeobecné postupy vlastné tomuto prostrediu. V prvom rade je nevyhnutné si vyjasniť základné slovné vyjadrenia, teda terminológiu.

Mnohí používatelia považujú {R} za štatistický program. Hoci to nie je chybou, predsa by sme chceli upozorniť, že {R} je programové prostredie (je v ňom možno písať programy), v ktorom sa používajú, ale aj vyvíjajú funkcie, prevažne na použitie v štatistike. [4] Preto obsahuje aj balíky s údajmi, umožňuje vykresľovať údaje či výsledky niektorých riešení, vytvárať tabuľky i spolupracovať s inými programovými celkami, napr. Excel. Môžeme vyhlásiť, že toto prostredie môže v dostatočnej miere nahrádzať historicky staršie, roky vyvíjané profesionálne štatistické programy, akými sú SPSS, SAS a iné.

Funkcie

Funkciou sa rozumie program, ktorý je volaný z hlavného programu, v tomto prípade z prostredia {R}. Funkcie spracúvajú parametre a vracajú výsledky ich spracovania vo forme hodnôt. Všeobecný formát volania funkcie sa skladá z jej názvu s parametrami uzavretými v zátvorke *funkcia (parameter, ...)*. Funkcie sa pre lepšiu prehľadnosť a účelnosť združujú do knižníc. Knižnica obsahuje viacero funkcií, ktoré sa naraz inštalujú do programového prostredia a následne možno jednu funkciu po druhej volať. Knižnice sú voľne dostupné vo forme balíkov (package) v CRAN. Za touto skratkou sa skrýva sieť serverov, ktoré obsahujú rovnakú sadu zdrojov (programov a balíkov) prostredia {R} (http://cran.r-project.org). Prehľad základných funkcií a definície ich štruktúry sú dostupné v príručke. [5]

Premenná

Premenná je výraz, ktorý sa často vyskytuje v štatistike, ale aj v práci s číslami. Pri používaní kalkulačky väčšinou zadáme číslo, potom operátor, za ním ďalšie číslo a po stlačení „rovná sa" dostaneme výsledok. Po vypnutí kalkulačky, po zadaní iného výpočtu sa však k nemu nemôžeme vrátiť. Tu pomôže premenná. Ak nazveme tento „výsledok" menom, nie však v kalkulačke, ale v počítači, napríklad v programe {R}, program si pod týmto názvom bude program pamätať číslo až do jeho ukončenia, alebo do uloženia na disk. Takto sa s výhodou používa *premenná* (variable), názov vyplývajúci zo skutočnosti, že pod menom jednej premennej sa môžu skrývať rôzne, variabilné údaje. Obsahom premennej môže byť jedno alebo viac čísel. Tie získame najčastejšie meraním na jedincovi, a preto sa aj zvyknú nazývať *merania*. Typickým objektom štatistického skúmania je preto súbor premenných, ktoré obsahujú rôzne merania. Napríklad pri zisťovaní vplyvu výživy na rast dieťaťa budeme mať premenné, ktoré obsahujú merania u detí, a to vek, pohlavie, množstvo prijatých kalórií denne, množstvo tuku, bielkovín a cukrov v prijatej potrave a podobne. Všetky údaje zaznamenané napríklad v Excel uložíme na disk počítača v súbore s názvom *dieta*, a tak uchováme premenné na použitie neskôr.

Inštalácia programového prostredia {R}

Zdrojom programu {R} je stránka projektu http://www.R-project.org/. Hoci je program dostupný pre rôzne operačné systémy, my sa zameriame na Windows. Pokiaľ niekto používa iný systém, napríklad Linux alebo MAC OS, musí si stiahnuť príslušnú verziu z tej istej stránky. Postup stiahnutia a inštalácie dokumentujeme v tabuľke 1.

- Choďte na stránku projektu http://www.R-project.org/
- Kliknite na odkaz CRAN v časti Dowlnload v ľavom stĺpci stránky
- Vyberte si stránku, z ktorej budete sťahovať program.
- V časti Download and Install R kliknite na link Windows
- Na nasledujúcej stránke vyberte kliknutím možnosť base
- Aktivujte link Download R 2.11.1 for Windows (33 megabytes, 32 bits)
- Po stiahnutí súboru, spustite jeho rozbalenie a inštaláciu programu. Úspešná inštalácia sa ukončí zobrazením modrého písmena R na ploche obrazovky.

Tabuľka 1 Postup pri inštalácii programu {R}

Objekty

Všetko, s čím pracujeme v prostredí {R}, sa nazýva *objektom*, teda čísla, znaky, funkcie atď. Pri práci s objektmi používame im priradené názvy, ktorými sa označuje hodnota objektu. Poznáme viacero druhov objektov, klasifikácia nazýva druhy objektov ako *triedy (class)*.

- čísla (numeric)
  - celé (integer) – 2, 3, 4
  - „reálne“ (double) – .5, 3.8, 4.3 (pozor na desatinnú bodku a nie čiarku)
  - komplexné (complex) – 4+2i, 25-3i, i;
- textové reťazce (character) – “test”, “Ahoj”;
- dátum (date) – zadanie je zložitejšie, ale objekt vyzerá nasledovne “2006-03-15”;
- logické hodnoty (logical) – TRUE, FALSE;
- vektory (vector) – rady čísel, reťazcov alebo iných dát;
- faktor (factor) – špeciálny typ vektoru s nominálnymi nebo ordinálnymi údajmi;
- matica (matrix) – klasická matematická matica nebo matica reťazcov a iných dát;
- pole (array) – vo svojej podstate viacrozmerná matica;
- zoznamy (list) – špecifické objekty združujúce rôzne iné objekty rôznych typov;
- tabuľky dát (data.frame) – klasické databázové tabuľky;
- funkcie (function) – príkazy, ktorými spúšťame určitú operáciu s objektami;
- výrazy (expression), rovnice (formula), špeciálne objekty (výsledky testov, analýz).

Obrázok 2 Triedy objektov podľa [6]

Pri priraďovaní názvov objektom je dobré a niekedy nevyhnutné dodržať niektoré pravidlá. Nie je možné používať názvy funkcií a niektorých hodnôt, ktoré sú rezervované, napr. označiť premennú ako *F*, keďže je to označenie pre *FALSE*. Pri nedodržaní tohto pravidla vás systém upozorní na chybu. Rovnako je potrebné prísne rozlišovať malé a veľké písmená. Objekty s názvom *prvy* a *Prvy* nie sú totožné, ale označujú dva objekty, ktorým môže byť priradený rôzny obsah. Je potrebné používať desatinnú bodku a nie čiarku. Dve a viacslovné označenia nesmú obsahovať medzeru alebo rozdeľovník, ale slová sa môžu spojiť pomocou bodky či podčiarnika, napríklad *krstne meno* je ako názov objektu nesprávne, ale *krstne.meno* či *krstne_meno* je v poriadku.

```
> krstne meno <- c(„František“) # systém ohlási chybu
Error: unexpected symbol in “krstne meno”

> krstne_meno <- c(“František”)
> krstne_meno
[1] “František”

> krstne.meno <- c(“František”)
> krstne.meno
[1] “František”
```

Príklad 1 Správne a nesprávne pomenovania objektov

Niektoré funkcie vyžadujú presné dodržanie triedy objektu. Pokiaľ si nie sme istí, napríklad pri prečítaní externých údajov, môžeme na jej určenie použiť niektorú z funkcií *class*(), *mode*(), *typeof()*. Vo väčšine prípadov si vystačíme s prvou z nich. Druhé dve poskytujú menej podrobnú klasifikáciu.

Prácu s objektmi uľahčujú niektoré funkcie. Tu si uvedieme len tie najčastejšie používané. Potrebujeme napríklad zistiť, aké objekty máme momentálne k dispozícii. Funkcia *ls()* vypíše všetky objekty, ktoré sa momentálne nachádzajú v pracovnom prostredí (workspace). Podobne sa správa funkcia *objects()*.

Na odstránenie objektu alebo objektov z pracovného prostredia použijeme funkciu *rm()*. Ako argument funkcie uvedieme do zátvorky názvy objektov, ktoré chceme odstrániť. Štruktúru objektu zobrazí funkcia *str()*. Podrobnejšie sa s uvedenými funkciami stretneme pri ich praktickom využití.

Jednoduché operácie s údajmi a premennými

V predchádzajúcom texte sme uviedli pojem premennej na príklade obmedzenia kalkulačky. {R} môžeme použiť ako kalkulačku veľmi jednoducho, príklad 2 predstavuje použitie základných operácií. Všimnite si, že používame desatinnú bodku a nie desatinnú čiarku. Znamienko # označuje komentár, ktorý program {R} ignoruje, ale umožňuje si aj neskôr pripomenúť, čo bolo zámerom určitých krokov.

```
> 3+4 # súčet
[1] 7
> 5.6-3.2 # rozdiel
[1] 2.4
> 4.5*3 # násobenie
[1] 13.5
> 9.6/4 # delenie
[1] 2.4
```

Príklad 2 Základné operácie s číslami

Ak by sme však chceli výsledok niektorej z operácií zachovať, musíme ho uložiť do premennej. Najjednoduchším spôsobom je použiť znamienko rovná sa = (Príklad 3).

```
> a = 3*4 # vynásobili sme dve čísla a výsledok uložili do premennej a
> a  # výsledok sa vypíše zadaním názvu premennej
[1] 12
> b=6^2 # operácia umocnenia na druhú
> b # výsledok sa vypíše zadaním názvu premennej
[1] 36
> ls()  # funkcia, ktorá vypíše názvy premenných, ktoré sa nachádzajú v pamäti programu {R}
[1] "a"                    "b"
> a*b # s premennými môžeme narábať rovnako ako s číslami
[1] 432
> b-a
[1] 24
> vysledok<-a+b # výsledok môžeme uložiť do novej premennej
> vysledok
[1] 48
```

Príklad 3 Použitie premennej na uloženie výsledkov operácií

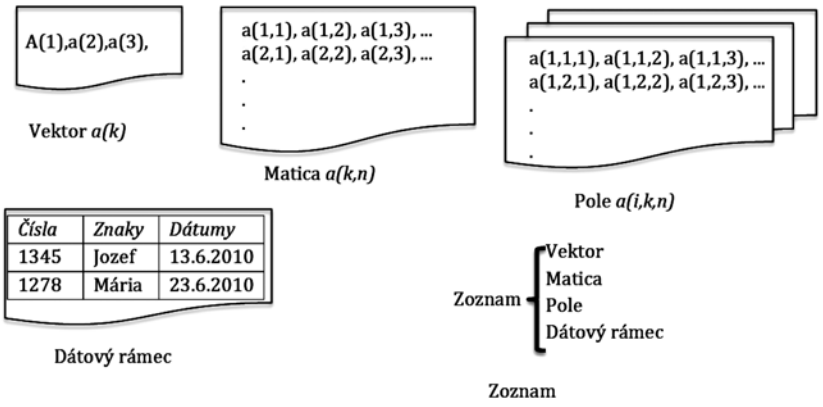
Premenné môžu obsahovať celé i desatinné čísla, ale aj písmená a znaky. Okrem toho premenné môžu obsahovať logické hodnoty *YES* (áno) a *NO* (nie). Špeciálnym prípadom je chýbajúca hodnota NA, ktorá má v štatistike významné postavenie. Písmená a texty označíme úvodzovkami a môžeme ich ukladať do premenných. Logické premenné musíme buď vypísať veľkými písmenami, alebo stačí použiť len prvé písmeno Y a N. Chýbajúcu hodnotu označíme kombináciou NA (Príklad 4).

```
# priradenie textov do premennej
> texty <- c("Martin", "Juraj", "Zdeno", "Karol")
> texty
[1] "Martin" "Juraj"  "Zdeno"  "Karol"
> texty <- c("Martin", "Juraj", NA, "Zdeno", "Karol") # chýbajúca hodnota
> texty
[1] "Martin" "Juraj"  NA       "Zdeno"  "Karol"
# logické premenné
> ano <- T
> ano
[1] TRUE
> ano = TRUE
> ano
[1] TRUE
> nie <- FALSE
> nie
[1] FALSE
> nie <- F
> nie
[1] FALSE
```

Príklad 4 Vytvorenie premenných s rôznym obsahom

Štruktúry údajov

Okrem premenných, ktoré obsahujú jediné číslo, používame komplikovanejšie premenné, akými sú vektory, matice, polia, dátový rámec a zoznamy (Obrázok 3). Líšia sa v rôznosti typov údajov, ktoré môžu obsahovať, v zložitosti štruktúry, v spôsobe dostupnosti jednotlivých položiek.



Obrázok 3 Rôzne druhy štruktúr údajov

Postupne sa zoznámime s jednotlivými z nich.

Vektory

V štatistike často používame vektory. Obsahujú merania na rôznych jedincoch z jednej premennej. Zadali sme premennú vek, ktorá obsahuje vek detí. Na zadanie hodnôt sme použili funkciu *c()* , ktorá kombinuje (combine), alebo priraďuje hodnoty v jej argumente a uloží ich do jednej premennej vo forme vektora (Príklad 5).

```
> vek <- c(1, 2.5, 2, 4.5, 3, 6)
> vek
[1] 1.0 2.5 2.0 4.5 3.0 6.0
```

Príklad 5 Vytvorenie vektora meraní a ich uloženie do premennej

Výsledok je rovnaký, či použijeme < – alebo = pre priradenie. Názvy premenných nie sú obmedzené, len je potrebné dávať pozor na malé a veľké písmená. Taktiež je dobré sa vyvarovať písmen s diakritikou, hoci mnohé prostredie {R} prijme, nie vždy sa tak musí stať.

Matice

Zložitejšie je vytváranie matíc, ktoré sa najčastejšie používajú pri riešení štatistických problémov. Matica je vlastne tabuľka, ktorá obsahuje viaceré premenné a každá má viaceré merania. Jej veľkosť definuje počet stĺpcov a počet riadkov, čo sa zvykne nazývať aj *dimenziou*. Vytvorenie matice z vektora hodnôt vyžaduje pridanie dimenzie matice, teda určenie počtu riadkov a stĺpcov pomocou funkcie *dim()*. Jej použitie si ukážeme na vektore, ktorý sme vyššie vytvorili (Príklad 6).

```
> vek
[1] 1.0 2.5 2.0 4.5 3.0 6.0
> dim(vek) <- c(2,3) # z vektora so šiestimi meraniami sme vytvorili maticu s dvoma riadkami
a troma stĺpcami
> vek
     [,1] [,2] [,3]
[1,]  1.0  2.0    3
[2,]  2.5  4.5    6
> dim(vek) <- c(3,2) # z vektora so šiestimi meraniami sme vytvorili maticu s troma riadkami
a dvoma stĺpcami
> vek
     [,1] [,2]
[1,]  1.0  4.5
[2,]  2.5  3.0
[3,]  2.0  6.0
```

Príklad 6 Vytvorenie matice z vektora čísel

Elegantnejšie a zároveň obsiahle riešenie je použitie funkcie *matrix()*. Tá poskytuje možnosti vytvorenia matice údajov zadaním vektora hodnôt, ktoré sa potom upravia do stĺpcov a riadkov, pričom je možné špecifikovať názvy riadkov a stĺpcov. Formát zadania funkcie je:

matica < – matrix(vector, nrow=number_of_rows, ncol=number_of_columns, byrow=logical_value, dimnames=list(char_vector_rownames, char_vector_colnames))

Výber *byrow=TRUE* určuje naplnenie matice po riadkoch, *byrow=FALSE* po stĺpcoch. Predvolená hodnota je po stĺpcoch. Niekoľko operácií si ukážeme na príklade vektora, ktorý sme použili vyššie.

```
> vek < – c(1, 2.5, 2, 4.5, 3, 6) # vytvorenie vektora čísel
> vek1 < – matrix(vek, nrow=3, ncol=2) # vektor transformujeme na maticu s 3 riadkami a 2
stĺpcami, všimnite si, že bez určenia parametra byraw funkcia použila predvolenú hodnotu
byrow=FALSE a spracovala požiadavku po stĺpcoch.
> vek1 # výsledok
     [,1] [,2]
[1,] 1.0 4.5
```

[2,] 2.5 3.0
[3,] 2.0 6.0

> *vek2 < – matrix(vek, nrow=2, ncol=3)* # pri dvoch riadkoch a troch stĺpcoch, ale zachovaní postupnosti po stĺpcoch
> *vek2* # výsledok
  [,1] [,2] [,3]
[1,] 1.0 2.0 3
[2,] 2.5 4.5 6

> *vek1 < – matrix(vek, nrow=3, ncol=2, dimnames = list(c(„A“,“B“,“C“), c(„X“,“Z“)))* # dodali sme názvy stĺpcov a riadkov
> *vek1* # výsledok
  X Z
A 1.0 4.5
B 2.5 3.0
C 2.0 6.0

> *vek2 < – matrix(vek, nrow=2, ncol=3, byrow=TRUE)* # zmena postupu načítania zo stĺpcov na riadky
> *vek2* # výsledok
  [,1] [,2] [,3]
[1,] 1.0 2.5 2
[2,] 4.5 3.0 6

> *vek3 < – matrix(vek, nrow=2, ncol=3, byrow=FALSE)* # prednastavené
> *vek3* # výsledok
  [,1] [,2] [,3]
[1,] 1.0 2.0 3
[2,] 2.5 4.5 6

Príklad 7 Použitie funkcie *matrix* ()

Matice sú užitočným spôsobom narábania s údajmi, potrebujeme však poznať správnu identifikáciu jednotlivých hodnôt. Postupujeme obdobne ako v akýchkoľvek tabuľkách, priesečník stĺpca a riadku jednoznačne udáva polohu čísla. Značíme to pomocou indexov, napríklad *A [i,]* označuje *i*-tý riadok matice A, kým *A [,j]* zodpovedá *j*-tému stĺpcu tej istej matice. Keď to združíme v jednom zápise, potom *A [i, j]* zodpovedá postupne *ij*-tému číslu. Tieto indexy *i* a *j* môžu byť rovnako vo forme vektorov tak, aby umožnili vyberať viaceré riadky či stĺpce.

> *vek <- c(1, 2.5, 2, 4.5, 3, 6)*
> *vek2 <- matrix(vek, nrow=2, ncol=3)* # vytvoríme maticu z údajov vyššie s dvoma riadkami a 3 stĺpcami

```
> vek2 # výsledok
   [,1] [,2] [,3]
[1,]  1.0  2.0    3
[2,]  2.5  4.5    6

> vek2[2,] # vyberieme len druhý riadok
[1] 2.5 4.5 6.0

> vek2[,2] # vyberieme len druhý stĺpec
[1] 2.0 4.5

> vek2[,c(2,3)] # vyberieme len druhý a tretí stĺpec
   [,1] [,2]
[1,]  2.0    3
[2,]  4.5    6
```

Príklad 8 Výber čísel z matice

Matice majú dva rozmery a rovnako ako vektory môžu obsahovať výhradne jeden typ dát. Pokiaľ máme viac ako dve dimenzie, použijeme polia. Keď potrebujeme kombinovať viaceré typy údajov, použijeme rámce údajov.

Polia

Polia sú viac ako dvojrozmerné matice. Môžeme si ich predstaviť ako matice uložené jedna za druhou. Uvádzame ich, hoci sa bežne nepoužívajú pri spôsoboch popisovaných v tejto publikácii. Aj preto použijeme príklad, ktorý uvádza Kabacoff [2], využívajúc funkciu vytvorenia poľa údajov *array(vector, dimensions, dimnames)*. Premenná *vector* obsahuje údaje vo forme vektoru, *dimensions* je ďalší vektor, ktorý určuje najväčší index každej z dimenzií a *dimnames* je voliteľný zoznam popisov dimenzií. Nasledujúci obrázok ilustruje vytvorenie trojrozmerného poľa (2x3x4) čísel.

```
> dim1 <- c(„A1“, „A2“)
> dim2 <- c(“B1”, “B2”, “B3”)
> dim3 <- c(“C1”, “C2”, “C3”, “C4”)
> z <- array(1:24, c(2, 3, 4), dimnames=list(dim1, dim2, dim3)) # vektor vytvoríme z čísel od
1 do 24, čísla rozdelíme do 2 riadkov, po 3 stĺpcoch a v 4 rozmeroch; písmenom A označíme
riadky, písmenom B stĺpce a písmenom C každý z rozmerov
> z # vypíšeme každý z rozmerov
, , C1
   B1 B2 B3
A1  1  3  5
A2  2  4  6
```

```
, , C2
  B1 B2 B3
A1  7  9 11
A2  8 10 12

, , C3
  B1 B2 B3
A1 13 15 17
A2 14 16 18

, , C4
  B1 B2 B3
A1 19 21 23
A2 20 22 24
```

Príklad 9 Príklad vytvorenia poľa údajov funkciou array() podľa [2]

Tabuľky údajov

Najčastejšie používanou štruktúrou pre prácu s údajmi sú tabuľky údajov (data frame). Tie si môžeme predstaviť ako matice, kde každý stĺpec môže obsahovať iný formát údajov. Analogicky k predchádzajúcim postupom na vytvorenie tabuľky (vynecháme slovo „údajov" pre zjednodušenie) použijeme funkciu *data.frame(col1, col2, col3,…)*.

Predstavme si ako príklad vytvorenie súboru, v ktorom budeme mať jednotlivé oddelenia v nemocnici a pri každom z nich bude počet postelí, počet lekárov a stav akreditácie. Postup vytvorenia takéhoto súboru ilustruje Príklad 10.

```
> oddelenie <- c(„interné", „neurológia", „chirurgia", „pediatria") # premenná s textom
> postele <- c(80, 25, 60, 50) # premenná s číslami
> akreditacia <- c(TRUE, FALSE, FALSE, TRUE) # premenná s logickými hodnotami
> nemocnica <- data.frame(oddelenie, postele, akreditacia)
> nemocnica

  oddelenie postele akreditacia
1 interné       80        TRUE
2 neurológia    25       FALSE
3 chirurgia     60       FALSE
4 pediatria     50        TRUE
```

Príklad 10 Vytvorenie rámca údajov

Vytvorenie tabuľky teda nie je nijak zložité. Ako sa však adresujú jednotlivé bunky? Najjednoduchším spôsobom je zápis pomocou indexov, ako sme to robili aj

v prípade matíc. Tento spôsob je síce jednoduchý, ale zdĺhavý a môže viesť ľahko k chybám (Príklad 11).

```
> nemocnica[1,3] # určenie bunky na základe čísla riadku a stĺpca
[1] TRUE

> nemocnica[,2] # všetky hodnoty zo stĺpca 2
[1] 80 25 60 50

> nemocnica[2,] # všetky hodnoty z riadku 2
  oddelenie postele akreditacia
2 neurológia      25      FALSE

> nemocnica[2:3] # všetky hodnoty stĺpca 2 a 3 (predvolený je výber stĺpca)
        postele     akreditacia
1         80          TRUE
2         25          FALSE
3         60          FALSE
4         50          TRUE
```

Príklad 11 Výber elementu z rámca dát prostredníctvom uvedenia priamej adresy

Iným spôsobom je určenie stĺpca na základe jeho názvu. Znak *$* oddeľuje názov tabuľky od názvu stĺpca, v prípade stĺpca 2 s názvom *postele* bude teda určený pomocou *nemocnica$postele*. V prípade, že by sme chceli vybrať len tretiu položku v tomto stĺpci doplníme riadok *nemocnica$postele [3]* a dostaneme číslo 60.

Ani tento spôsob nie je ideálny, pri každej operácii písať celú kombináciu názvu tabuľky a stĺpca nie je príliš praktické. Prostredie {R} ponúka viaceré funkcie, ktoré značne zjednodušujú tieto operácie: *attach()*, *detach()* a tiež *with()*. Prvá z nich, *attach()*, doplní názvy premenných do vyhľadávania prostredia {R}, tým sa javia stĺpce ako samostatné premenné a je možné s nimi narábať pomocou ich názvov.

```
> attach(nemocnica)
The following object(s) are masked _by_ '.GlobalEnv':

  akreditacia, oddelenie, postele

> postele
[1] 80 25 60 50
```

Príklad 12 Použitie funkcie *attach()*

Zrušenie účinku tejto funkcie, a teda odstránenie názvov premenných z vyhľadávania prostredia {R} dosiahneme funkciou *detach()*. Tým sa uvoľnia názvy pre-

menných na použitie v iných zostavách. Aj keď väčšinou nepoužívame toľko premenných, aby sa to dalo nevyhnutné vykonať, odporúča sa použitie tejto funkcie pri ukončení narábania s premennými z jedného poľa údajov.

Namiesto písania *štruktúra$premenná* môžeme použiť funkciu *with()*. Napríklad uvedenie štruktúry a premennej v argumente funkcie zobrazí hodnoty premennej (Príklad 13).

```
> with(nemocnica, postele)
[1] 80 25 60 50
```

Príklad 13 Použitie funkcie *with()*

Aj keď sa nezdá toto použitie ako nevyhnutné, v prípadoch jednorazového použitia premenných v zložitejších funkciách často uľahčí prácu.

Zoznam je špecifickou štruktúrou dát, ktorá umožňuje kombinovať pod jedným názvom rôzne štruktúry dát, napríklad vektory s maticami a tabuľkami dát. Najčastejšie sa používa pri návrate hodnôt z volanej funkcie.

Dátum a čas

Dátum a čas predstavujú zvláštny druh premennej, ktorá sa v zdraví verejnosti vyskytuje pomerne často. Priraďujeme dátum a čas v záznamoch, či už ako dátum, keď sa odohrala nejaká udalosť, či dátum narodenia alebo smrti. Narábanie s premennými, ktoré obsahujú čas, nie je triviálne, preto sa mu venujeme v tejto časti, hoci pri práci s údajmi, ktoré už boli predspracované, sa zvyčajne s časom ako premennou nenarába.

Začneme dnešným dátumom, ktorý {R} načíta z dátumu, ktorý sleduje operačný systém počítača. Teda, pokiaľ dostanete iný dátum, ako je dnešný, neviňte hneď {R}, ale pozrite sa, ako máte nastavený systémový čas vo vašom počítači.

```
> date()
[1] "Sun Jan  6 13:15:41 2013"
> Sys.time()
[1] "2013-01-06 17:24:46 CET"
> Sys.Date()
[1] "2013-01-06"
```

Príklad 14 Čas používaného systému volaním troch funkcií

Dátum a čas môžeme priradiť do premennej typu text priamo, ale budeme s ňou môcť narábať len obmedzene. Priradenie dátumu sa robí pomocou funkcie priradenia *c()*. Na ďalšiu činnosť s dátumom a časom sa však potrebujeme bližšie zoznámiť s dvoma triedami *POSIXct* a *POSIXlt*. [5] Prvá z nich *POSIXct* predstavuje počet se-

kúnd (so znamienkom), ktoré ubehli od začiatku roku 1970 vo forme vektora číslic. Druhá z nich, *POSIXlt*, je zoznamom vektorov, ktoré predstavujú jednotlivé súčasti špecifikácie času a dátumu (Tabuľka 2).

sec	Sekundy 0–61
min	Minúty 0–59
hour	Hodiny 0–23
mday	Deň v mesiaci 1–31
mon	Mesiace po prvom v roku 0–11
year	Počet rokov od roku 1900
wday	Deň v týždni, s nedeľou ako prvým dňom 0–6
yday	Deň v roku 0–365
isdst	Návestie pre letný/zimný čas. Kladné, pokiaľ sa používa, nula, ak sa nepoužíva a záporné, pokiaľ nie je známe

Tabuľka 2 Špecifikácia času a dátumu v triede *POSIXlt.*

Aby sme mohli použiť tento zoznam vektorov, potrebujeme funkciu, ktorá ho dokáže vložiť do premennej. Tou je funkcia *as.POSIXlt()*. Jej zavolaním a použitím systémového času v jej argumente môžeme rozložiť tento zoznam na jednotlivé jeho položky, pri použití názvov vektorov zoznamu podľa vyššie uvedenej tabuľky (Tabuľka 2).

Príklady na použitie funkcie *as.POSIXlt(Sys.time()* pre prácu s časom sú uvedené v nasledujúcom príklade (Príklad 15).

```
> datum<- as.POSIXlt(Sys.time()) # do premennej sme vložili zoznam s momentálnym časom
systému
> datum$sec # sekunda
[1] 22.47269
> datum$min # uplynulé minúty z aktuálnej hodiny
[1] 11
> datum$hour # uplynulé hodiny aktuálneho dňa
[1] 18
> datum$mday # uplynulé dni aktuálneho roku
[1] 7
> datum$mon # poradové číslo aktuálneho mesiaca
[1] 0
> datum$year # poradové číslo aktuálneho roku, počítajúc od roku 1900
[1] 113
> datum$wday # poradové číslo dňa v týždni
[1] 1
> datum$yday # poradové číslo dňa v roku
[1] 6
> datum$isdst # letný/zimný čas
[1] 0
```

Príklad 15 Rozklad položiek zoznamu POSIXlt

Častou operáciou je zistenie času, ktorý uplynul medzi dvoma časovými bodmi. Túto operáciu uľahčuje funkcia *difftime(time1, time2, tz = „“, units = c(„auto“, „secs“, „mins“, „hours“, „days“, „weeks“))*. Prvé dve premenné sú oba časové okamihy, či sú to dátumy, alebo hodinové údaje. Návestie *units* určuje výber formátu požadovaných výsledkov (možno použiť aj skratky). Pri použití tejto funkcie je nevyhnutné konvertovať výsledok na číslo, najmä ak chceme následne vykonať niektorý z aritmetických výkonov, napríklad rozdiel v rokoch, teda vydeliť počtom dní v roku (Príklad 16).

```
> dnes<- as.POSIXct(Sys.time()) #vloženie momentálneho času systému
> tu <- as.POSIXct("1992-03-25") # dátum otvorenia obnovenej Trnavskej univerzity
> diff <- difftime(dnes, tu, units = c("d")) # počet dní, ktoré ubehli od otvorenia
> diff
Time difference of 7596.618 days
> class(diff) # rozdiel je stále v triede diff
[1] "difftime"
> diff <- as.numeric(diff) # predstavenie rozdielu vo forme čísla
> year <- diff/365 # počet rokov
> year
[1] 20.81265
```

Príklad 16 Rozdiel dvoch dátumov

Problematika spracovania času v štatistických systémoch nie je zas taká častá, aby nás to oprávňovalo sa jej venovať v ďalších podrobnostiach. Pokiaľ sa čitateľ stretne s riešením problému času, tak mu odporúčame hľadať riešenie buď na internete, alebo použiť niektorú z kníh o {R} [2, 7].

Čítanie údajov zo súborov

Najjednoduchším spôsobom čítania súborov je ich predstavenie napríklad v programe Excel alebo Word, alebo pomocou Adobe. Potom vyberieme požadované údaje do bloku a skopírujeme CTRL-C alebo príkazom z lišty. Na ich prečítanie z pamäte (clipboard) použijeme funkciu *read.table()*. V prvom kroku si pripravíme súbor údajov, ktoré chceme načítať. Na demonštráciu použijeme hotový súbor v Exceli (čitateľ si to môže vyskúšať na svojich údajoch), ktorý sme získali ako voľne prístupný súbor údajov zo štúdie rizikových faktorov ochorení srdca v Los Angeles.[3] Údaje sú priamo vo formáte Excel a uložíme ich do priečinku vo formáte, kde

[3] http://www.umass.edu/statdata/statdata/data/laheart.txt

sú údaje oddelené čiarkou (*.csv*). Je to spôsobené tým, že existuje viacero formátov Excel súborov a pre vstup údajov je potrebný čo najjednoduchší spôsob ich prečítania. Na uľahčenie používania údajov v tomto priečinku spresníme ho ako pracovný volaním funkcie *setwd()*. Jej argumentom bude cesta, ktorá určí položenie priečinku na disku. Na overenie správnosti použijeme funkciu *getwd()* a zoznam súborov nachádzajúcich sa v ňom zobrazíme pomocou funkcie *list.files()*.

Volanie funkcie *read.table()* umožní takto pripravený súbor prečítať a výsledok vložiť do premennej. Výsledok je premenná vo forme matice, keď chceme pracovať s údajmi v jednotlivých stĺpcoch musíme ich nazývať pomocou znaku $ v spojení *matica$stĺpec*, teda v našom prípade, napríklad adresovanie stĺpca s ID účastníkov šetrenia musíme použiť konštrukciu *srdce$ID*. Volaním funkcie *attach()* a udaním názvu premennej môžeme potom používať názvy stĺpcov ako samostatné premenné. Celý postup ilustruje Príklad 17.

```
> setwd(„C:/Users/Martin Rusnak/Documents/R/Datafiles/Cardio“) # určenie pracovného priečinku

> getwd() # overenie správnosti
[1] “C:/Users/Martin Rusnak/Documents/R/Datafiles/Cardio”

> list.files() # výpis obsahu priečinku
[1] “~$A. HEART DATA (LAHEART.DAT).docx” “L.A. HEART DATA (LAHEART.
DAT).docx” “laheart.xls” “laheart.csv”

> srdce <- read.table(“laheart.csv”,header=T, sep = “,”) # prečítanie údajov oddelených čiarkou,
v prvom riadku sú názvy stĺpcov

> srdce
   ID AGE_50 MD_50 SBP_50 DBP_50 HT_50 WT_50 CHOL_50 SES CL_STATUS
MD_62 SBP_62 DBP_62 CHOL_62 WT_62 IHD_DX DEATH_YR DEATH
1   1   42   1   110   65   64   147   291   2      8   4   120   78   271   146   2   68   1
2   2   53   1   130   72   69   167   278   1      6   2   122   68   250   165   9   67   1
.....
>srdce$ID # vypíše obsah stĺpca ID
>attach(srdce) # umožní aby názov stĺpca bol zároveň názvom premennej
```

Príklad 17 Prečítanie súboru exportovaného z Excel

Pokiaľ sa to čitateľovi zdá komplikované, alebo pokiaľ chce pracovať len s časťou tabuľky s dátami (či už v Excel alebo inom programe), riešením je použitie volania rovnakej funkcie, ale namiesto názvu súboru použijeme špecifikáciu *„clipboard“*. Postupujeme tak, že zo súboru vyberieme časť údajov do bloku aj s prvým riadkom, v ktorom sú názvy stĺpcov a stlačením *CTRL-C* ich skopírujeme do pamäte. Potom

volaním funkcie *read.table(„clipboard", header=T)* s argumentmi *„clipboard"*, ktorý hovorí, že údaje sa nachádzajú v pamäti a *header=T*, ktorý hovorí, že v prvom riadku sa nachádzajú názvy stĺpcov, prečítame dáta do premennej v {R} (Príklad 18).

```
> srdce1 <- read.table(„clipboard",header=T) # prečítanie súboru z clipboard
> srdce1
```

	AGE_50	MD_50	SBP_50	DBP_50	HT_50	WT_50	CHOL_50
1	42	1	110	65	64	147	291
2	53	1	130	72	69	167	278
3	53	2	120	90	70	222	342
4	48	4	120	80	72	229	239
5	53	3	118	74	66	134	243
6	58	2	122	72	69	135	210
7	48	4	130	90	67	165	219
8	60	1	124	80	74	235	203
9	59	4	160	100	72	206	269
10	40	3	120	80	69	148	185
11	56	3	115	80	64	147	260
12	58	3	140	90	63	121	312
13	64	2	135	85	64	189	185
14	57	2	110	78	70	173	282

Príklad 18 Čítanie údajov zo súboru v clipboard

V kapitolách o konkrétnych údajoch zo súborov si uvedieme o niečo komplikovanejšie prístupy.

Pomocné programy

Na zjednodušenie práce s programovým prostredím {R} je k dispozícii viacero pomocných programov, ktoré uľahčujú narábanie pomocou výberu z ponuky. Jedným z najjednoduchších je knižnica Rcmdr. Tá sa musí najprv inštalovať z lišty programu {R} *Packages-Install package(s)...* Následne možno túto knižnicu volať pomocou funkcie *library()*

```
> library(Rcmdr)
```

Prostredie R Commander je veľmi jednoduché a ponúka zjednodušenie práce. Podrobný návod je súčasťou programu.

Trochu viac možností ponúka program Tinn-R[4], ktorý je tiež voľne dostupný.

[4] http://www.sciviews.org/Tinn-R/

Jeho inštalácia je jednoduchá a okrem sprostredkovania prístupu k {R} ponúka aj textový editor.

Na narábanie s väčšími súbormi údajov je často užitočný doplnok k Excel, ktorý umožňuje priamo volať funkcie {R}. Odpadá potom čítanie údajov z Excel a volanie funkcií je tiež jednoduché priamo z lišty v Excel. Tento doplnok k Excel je možno rovnako voľne stiahnuť.[5]

Súhrn

Cieľom tejto kapitoly bolo naznačiť spektrum možností, ktoré poskytuje prostredie {R} pre základnú prácu s údajmi. Predstavili sme samotné prostredie {R} a uviedli terminológiu, ktorá sa v ňom používa. Popísali sme druhy objektov a niektoré funkcie, ktoré umožňujú či uľahčujú prácu s nimi. Rovnako sme spomenuli niektoré programy, ktoré uľahčujú prácu v prostredí {R} a prepájajú ho s bežným prostredím Windows či iného operačného systému.

Na záver tejto kapitoly by sme chceli všetkým záujemcom odporučiť niekoľko publikácií, ktoré priblížia viac podrobností k prostrediu {R}, ako sme to mohli urobiť my. Všetky sú dostupné na internete. Prvou je príručka v českom jazyku od autora Pavla Drozda: *Cvičení z biostatistiky Základy práce se softwarem {R}* [6]. Je písaná pre študentov a poskytuje hutné informácie s množstvom praktických príkladov. Druhá od autorky Kateřiny Konečnej, taktiež v českom jazyku, poskytuje návod na základné postupy v tomto prostredí. [8] Ďalšia je v anglickom jazyku a je veľmi inštruktívna. Autor Kabacoff [2] zvolil síce jednoduché, ale obsažné podanie mnohých základov práce v prostredí {R}. V neposlednom rade odporúčame stiahnuť si zoznam základných funkcií prostredia {R} [5] a konzultovať ho čo najčastejšie, najmä v začiatočných fázach zoznamovania sa s prostredím {R}.

Literatúra

1. Rusnák, M., Rusnáková, V., Majdan, M. *Bioštatistika pre študentov verejného zdravotníctva*. I. ed, 2010, Trnava: Typi Universitatis Tyrnaviensis, Vydavateľstvo Trnavskej univerzity. 216.
2. Kabacoff, R. I. *R in Action: Data analysis and graphics with R*. 2011: MANNING, Shelter Island.
3. Rothman, K. J., Greenland S and Lash TL. *Modern Epidemiology, 3rd Edition*. 3rd ed. 2008: Lippincott Williams & Wilkins. 738.
4. R Development Core Team. *R: A language and environment for statistical computing*. 2008, R Foundation for Statistical Computing: Vienna, Austria.

[5] http://cran.r-project.org/web/packages/RExcelInstaller/index.html