Úvod do softwaru R

Charakteristika:

- nekomerční, volně šiřitelný statistický software,
- analogie S+, objektově orientovaný jazyk,
- ovládání z příkazové řádky, otevřený zdrojový kód,
- možnost tvorby vlastních procedur,
- pod Unix i Windows,
- ke stažení na adrese: http://cran.at.r-project.org/ (zde zdrojový i zkompilovaný kód)
- dodatečné knihovny také na adrese: http://www.bioconductor.org/

Vybrané knihovny

- **vegan** -balík pro modelování v ekologii
- **epi, epitools** - nástroje pro epidemiologii
- **survival** -balík pro analýzu přežití
- **tree** - klasifikační a regresní stromy
- **ts** - časové řady
- **mgcv** - GAM - zobecněné aditivní modely
- **bioconductor** - soubor balíků pro analýzu Microarrays

Instalace:

- klasicky (setup.exe),
- doinstalování dalších knihoven: zkompilovanou knihovnu nakopírovat do adresáře .../library, nebo přidat přes Packages → Instal package(s) from CRAN

Přivolání knihovny (seznam knihoven: help → Packages):

- knihovnu už musíme mít nainstalovanou
- `library(nazevknihovny)`
  např. `library(MASS)`
- nebo volíme: packages → Load Package

Nápověda:

- Help → Html Help → Packages + Search Engine & Keywords, Search Engine vyžaduje Javu!

- Spuštění hypertextového helpu z příkazové řádky: `help.start()`

- informace o konkrétní proceduře - příkazový řádek: např: `help(median)`

- vypsání zdrojového kódu procedury: pouze název procedury (např: `median`)

- kompletní manuály ke stažení na http://www.r-project.org/

- Peter Dalgaar: Introductory Statistics with R, Springer 2002

Začínáme s prací

- Nastavení pracovního adresáře (zde se ukládají a vyhledávají soubory dat a výsledků): File → Change directory, nebo z příkazového řádku `setwd('......')`.

- Zjištění aktuálního nastavení pracovního adresáře: `getwd('......')`

- Načtení vlastní procedury ve formátu procedura.r : File → Source R code nebo z příkazového řádku `source(file)`

- Načtení dat:

  a datový formát soubor.Rdata: File → Load WorkSpace
  nebo z příkazového řádku `load(file, envir = parent.frame())`

  ```
  load("D:/Rko/mammals2.Rdata")
  ```

  b textový soubor soubor.txt,
  data ve sloupcích, sloupce oddělené mezerou, sloupce nemají jména:

  ```
  read.table(file, header = FALSE, sep = "", quote = "\"'", dec = "."...,)
  ```

  Příklad: data ve sloupcích, sloupce oddělené mezerou, sloupce mají jména:

  ```
  read.table("mammals1.txt",header=TRUE,sep=" ",dec=".")
  ```

  c textový soubor, další předdefinované formáty:

  ```
  read.csv(file, header = TRUE, sep = ",", quote="\"", dec=".", ...)
  read.csv2(file, header = TRUE, sep = ";", quote="\"", dec=",", ...)
  read.delim(file, header = TRUE, sep = "\t", quote="\"", dec=".",  ...)
  read.delim2(file, header = TRUE, sep = "\t", quote="\"", dec=",",  ...)
  ```

- výpis objektů, které máme momentálně v pracovním prostoru: `ls()`

- ukončení práce s programem: `q()`

Uložení dat:

- ve formátu .Rdata: `save(data, file="data.Rdata")`
- jako textový soubor: `write.table()`

Tvorba objektů:

- názvy proměnných - rozlišují se velká a malá písmena (X,x - dva různé objekty)
- přiřadíme názvu proměnné hodnotu: `x<-4`
- přiřadíme názvu proměnné hodnotu: `z<-x` nebo také

```
z<-x^2+abs(z)/cos(2)+pi-exp(1)*log(8)+log10(8)/sqrt(10)
```

- vytvoření vektoru: `v<-c(1,2,3,5,6,9)`
- k-tý prvek vektoru: `v[k]`
- vytvoření matice 3×4, která má všechny prvky nulové: `matice<-matrix(0,3,4)`
- prvek v prvním řádku a druhém sloupci je překvapivě nula: `m[1,2]`
- posloupnost od jedne do deseti, skok=0.5: `posloupnost<-seq(1,10,by=0.5)`
- uchop objekt: `attach(objekt)`
- vymaž objekt: `rm(objekt)`
- vypiš seznam existujících objektů : `ls()`
- vypiš seznam existujících objektů + některé další informace : `ls.str()`

Datové typy

- typ numerický (speciální numerické hodnoty: Inf, -Inf, NaN = Not A Number)
- typ znakový (”Toto je řetězec”)
- typ komplexní
- typ logický (TRUE, FALSE)
- typ objektu zjistíme pomocí funkce: `mode()`
- chybějící hodnoty jsou reprezentovány kódem: **NA = Not Available**

Datové struktury

- vektor (vector)
- matice (matrix)

- pole (array)
- datová tabulka (data frame)
- seznam (list)
- faktor (factor)

Počítání s vektory

- s vektory lze počítat, jako by to byla obyčejná čísla (mají-li vektory stejnou délku!)
- příklad BMI:

```
weight<-c(60,72,57,90,95,72)
height<-c(1.75,1.8,1.65,1.9,1.74,1.91)
bmi<-weight/height^2
bmi
```

- zjištění délky vektoru: `length(weight)`

Elementární vektorové funkce

- `sum(weight)`
- `min(weight), max(weight)`
- `mean(weight), median(weight)`
- `var(weight), quantile(weight,c(0.3,0.5))`
- `cumsum(weight)`

Práce s maticemi

- tvorba matice z vektoru: např. `matice<-matrix(c(1,2,3,4,5,6),2,3)`
- matice lze transponovat: `maticetr<-t(matice)`
- matice lze násobit (pozor na rozměry!): `matice % * % maticetr`
- sčítat a násobit skalárem...etc.
- submatice 2×2: `sub<-matice[1:2,1:2]`
- spojování matic - "slepujeme sloupce": `cbind(a,b)`
- spojování matic - "slepujeme řádky": `rbind(a,b)`

Příklad: Násobení matic

- `matice1<-matrix(2,2,2)`

- `matice1<-matrix(3,2,2)`

- `matice1 * matice2`

- `matice1 % * % matice2`

- Jak se liší tyto dva druhy násobení?

Pole

- 3 rozměrné pole: `pole<-array(1:24,c(3,2,4))`

- jaká je dimenze pole? `dim(pole)`

- prvek pole (analogicky jako v případě vektoru a matice): `pole[1,2,3]`

- vrstva pole (analogicky jako v případě matice): `pole[1,,]`

Seznam

- vytvoření seznamu:
  `pacient<-list(vek=45,mereni=c(1,1,2),ok=c(TRUE,TRUE,FALSE),znaky=c(rep("+",3)))`

- přístup ke složkám jménem: `pacient$vek, pacient$ok`

- přístup ke složkám pořadím: `pacient[[1]]`

- přístup k prvkům složek: `pacient$ok[3], pacient[[3]][3]`

- jména složek seznamu: `names(pacient)`

Datová tabulka

- speciální případ seznamu,

- složky jsou vektory různého typu, ale stejné délky

- zobecněná matice, řádky = pozorování, sloupce = veličiny

- vytvoření datové tabulky:
  `clovek<-data.frame(x1=c(rep(1,8),rep(0,8)),pohl=rep(c("Muz","Zena"),c(8,8)))`

- názvy položek datové tabulky: `names(clovek)`

- přístup k jednotlivým položkám: `clovek$pohl, clovek$pohl[3]`

- tato datová tabulka je nyní aktivní: `attach(clovek)`

- přístup k jednotlivým položkám (nyní nepotřebujeme $):`pohl,pohl[3]`

- ukončení práce s datovou tabulkou: `detach(clovek)`

- přejmenování: `names(clovek)[1]<-"pohl1"`

- oprava dat: `clovek$pohl1[3]<-1`

- je datová tabulka: `is.data.frame(clovek)`

Faktor

- speciální případ znakového vektoru
- má atribut "úroveň": `levels`
- vytvoření faktoru: `vek_kat<- factor(rep(c(0,1,2),c(4,5,6)))`
- výpis úrovní: `levels(vek_kat)`
- změna úrovní: `levels(vek_kat)<-c("dítě","dospělý","důchodce")`
- je faktor?: `is.factor(vek_kat)`

Rozdělení

- Normální rozdělení:
  a. hustota, x - kvantil: `dnorm(x, mean=0, sd=1, log = FALSE)`
  b. disribuční funkce,q - kvantil: `pnorm(q, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)`
  c. kvantily, p-pravděp.:`qnorm(p, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)`
  d. generování náhodných výběrů, n - rozsah výběru: `rnorm(n, mean=0, sd=1)`
- Studentovo rozdělení: df - stupně volnosti

```
dt(x, df, ncp=0, log = FALSE)
pt(q, df, ncp=0, lower.tail = TRUE, log.p = FALSE)
qt(p, df,        lower.tail = TRUE, log.p = FALSE)
rt(n, df)
```

Obrázky a grafy:

- kreslíme graf: příkaz `plot`, má velké množství parametrů pro nastavení: barvy, bodů, popisu os... etc.
  Příklad:

```
s<-seq(-3,3,by=0.1)
d<-dnorm(s)
plot(s,d, main="Hustoty",col="red", type="p", xlab="kvantily")
```

- do stávajícího grafu dokreslujeme další funkce:

```
t<-dt(s,5)
lines(s,t,type="l",col="blue")
```

- uložení obrázku: file → save as...

- pro listování mezi obrázky je nutné průběžné ukládání: history → recording

- histogram

```
x<-rnorm(1000,0,1)
 hist(x)
```

- box-ploty

```
boxplot(x, notch=TRUE)
y<-rnorm(1000,1,1)
r<-c(x,y)
nula<- matrix(0,1,1000)
jedna<-matrix(1,1,1000)
group<-c(nula,jedna)
#group<-c(rep(0,1000),rep(1,1000))
boxplot(r~group, col="yellow")
```

Testy

- Příklad: t-test - X_n je náhodný výběr z N(0,1) nezávislý s Y_n, náhodným výběrem z N(1,1). Testujte H_0 o shodnosti středních hodnot.

- `ttest<-t.test(x, y , alternative = c("two.sided", "less", "greater"), mu = 0, paired = FALSE, var.equal = FALSE,conf.level = 0.95, ...)`

- výpis pouze jednoho výstupu: `ttest$p.value, ttest$statistic` atd.

Tvorba vlastních procedur:

- v externím textovém editoru (Notepad, WinEdit,....), přípona.r

- File → New script (open script)

- struktura:

```
mojeprocedura<-function(vstup1, vstup2)
                {
                  příkaz;
                  příkaz;
                  příkaz;
                  vysledek<-příkaz;
                  return(vysledek);
                  print(vysledek);
                 }
```

- načtení do prostředí R: viz. načtení vlastní procedury přes file → Source R Code nebo z příkazové řádky pomocí příkazu `source("D:/Rko/mojeprocedura.r")`

- Výstup z procedury(funkce):

- Volání procedury: `mojeprocedura(1,2)`

- Komentáře: # bla bla bla, co je od křížku až do konce řádku, to zůstává komentářem

- Příklad: `secti<-function(x,y=2) {s<-x+y, return(s)}` , volání procedury např.: `soucet1<-secti(3), soucet2<-secti(3,2)`

Kontrolní příkazy a smyčky

- `if` (condifion) expression

- `if` (condifion) expression1 `else` expression2

- `while` (condifion) expression

- `for` (condifion) expression

- Příklad: Kódování proměnné:

```
n<-length(x)
xx<-x
for (i in 1:n) { if (x[i]>=0)  {xx[i]<-1 } else {xx[i]<- 0  }  }
```

- Poznámka: Operátory

  == rovná se

  != nerovná se

  & a zároveň

Chybějící pozorování

- Chybějící pozorování jsou reprezentovány kódem `NA`.

- ```
  tNA<-read.table("trees_NA.txt", header=TRUE, sep=" ", dec=".")
  attach(tNA)
  is.na(Height)
  sum(Height)
  sum(Height,na.rm=TRUE)
  ```

Lineární modely:

- `model1<-lm(zavisle~nezavisle1+nezavisle2*nezavisle3)`

- podrobnější výpis: `summary(model1)`

- regresní diagnostiky: `plot(model1, ask=TRUE)`
  ```

- `model2<-update(model, .~.-nezavisle2 )`
- testování podmodelu: `anova(model1,model2)`
- zobecněný lineární model:

  `glm(zavisle~nezavisle, family="binomial")`
- analýza rozptylu - jednoduché třídění :

  `anova(lm(zavisle~kategorie))`

Příklad: Lineární modely

```
library(MASS)
data(cats)
summary(cats)
plot(cats)
attach(cats)
cor(Bwt, Hwt, method = "spearman")
linmodel<-lm(Bwt~Hwt*Sex)
summary(linmodel)
plot(resid(linmodel)~Hwt,data=cats)
abline(h=0)
coef(linmodel)
glmmodel<-glm(Sex~Bwt, family="binomial")
summary(glmmodel)
```