

STRUČNÝ ÚVOD DO R

Webová stránka pro přednášku Statistika NSTP097:

<http://www.karlin.mff.cuni.cz/~kulich/vyuka/statfpm/index.html>

Charakteristiky R:

- Interaktivní programovatelné „výpočetní prostředí“ pro statistické výpočty a grafiku
- Dialekt jazyka S volně šiřitelný pod GNU GPL
- Funguje pod Windows, Unixem, MacOS, všemi versemi Linuxu
- Řádkový interpret, objektově orientovaný
- Snadno rozšiřovatelný (cca 1500 rozšiřujících knihoven na WWW)
- Současná verze 2.8.0 (k 1.12.2008)

Reference:

- *Hlavní webová stránka projektu:* <http://www.r-project.org/> (informace, manuály, FAQ)
- *Hlavní depositář:* <http://cran.r-project.org/> (instalace, zdrojový kód, rozšiřující knihovny ke stažení)

Technika práce s R:

- *Nastartování:* Ve Windows XP/2000 například z menu Start/Všechny programy/R/R 2.8.0 (závisí na instalaci a OS)
- *Konzolové okno [R Console]:* Slouží pro vstup příkazů z klávesnice a pro textový výstup. Předchozí příkazy lze vyvolat pomocí klávesy ↑.
- *Grafické okno [R Graphics]:* Slouží pro grafický výstup, otevře se samo po zadání jakéhokoli grafického příkazu. Graf lze vytisknout nebo uložit na disk ve zvoleném formátu skrze menu File/Save as. . .
- *Editace příkazů [R editor]:* Sem lze zaznamenávat příkazy, editovat je a odesílat je do konzolového okna. Kopírování funguje přes vyznačení myší a standardní klávesové zkratky Ctrl+C, Ctrl+X, Ctrl+V. K provedení jednoho nebo více příkazů stačí vyznačit myší požadovanou část kódu a zmáčknout Ctrl+R. Editovaný program lze uložit do souboru pomocí menu File/Save as. . . (programy v R mají koncovku `.r`).
- *Volba pracovního direktoráře:* V pracovním direktoráři se vyhledávají a ukládají soubory dat a výsledků. Pod Windows lze nastavit pomocí menu File/Change dir. . . Taktéž lze použít funkci `setwd()`. Nastavení pracovního direktoráře zobrazí funkce `getwd()`.
- *Uvádění cest (paths):* Pokud je třeba uvést se jménem souboru i cestu, použijeme jako oddělovač buď obvyčejné lomítko, nebo dvě zpětná lomítka. Píšeme tedy `c:/data` nebo `c:\\data`, nikoli `c:\data`.
- *Ukončení práce:* Práce s R se ukončuje příkazem `q()`. Následuje otázka, zdali se mají všechny existující objekty (nikoli však programový kód) uložit na disk. Odpovíte-li „Y“, vše se uloží do souboru `.RData` ve vašem pracovním direktoráři.

¹Michal Kulich, KPMS MFF UK, 1.12.2008

Nápověda:

- *Nápověda pro daný příkaz/funkci:* Vyvolá se v konsoli funkcí `help`, například `help(q)` pro ukončovací funkci `q()`.
- *Hypertextová nápověda s vyhledáváním:* Vyvolá se v konsoli pomocí `help.start()`. Nastartuje webový prohlížeč na úvodní stránce nápovědy. Přístup k nápovědě pro jednotlivé funkce, tříděné podle knihoven, jest skrze *Packages* a výběrem knihovny na následující stránce. Běžné příkazy a funkce se nacházejí v knihovnách *Base* (operace s daty), *Stats* (statistické funkce), *Utils* (ovládání a export/import dat) a *Graphics* (grafika). Navíc je zde k dispozici šest manuálů v angličtině a řada odkazů.

Vytváření, vypisování a mazání objektů:

R je objektově orientovaný jazyk: proměnné, data, výsledky, funkce, příkazy i jazykové konstrukce jsou objekty. Jedna funkce může vykonávat značně rozdílnou činnost v závislosti na typu objektů, které má zpracovat.

Jako názvy objektů jsou přípustné řetězce obsahující písmena, číslice a znak „.” (tečka). Velká a malá písmena se rozlišují, tj. `barel` a `Barel` označují dva různé objekty.

Objekty se vytvářejí přiřazením. Základní přiřazovací operátor je `=`; lze jej také psát jako `<-`, tj. posloupnost dvou znaků „`<`” a „`-`”.

Jména existujících objektů lze vypsat funkcí `ls()`. Jména plus některé další informace vypisuje i funkce `ls.str()`. Existující objekty lze smazat funkcí `rm()`.

Komentované příklady:

```
> X = 14                přiřazení objektu X
> x = sqrt(8)          x ≠ X
> z = x+X
> z                    výpis hodnoty z
[1] 16.82843            výsledek
> z+x^3                spočti jednoduchý výraz
[1] 39.45584
> ls()                 výpis jmen existujících objektů
[1] "x" "X" "z"
> ls.str()             výpis s více podrobnostmi
x : num 2.83
X : num 14
z : num 16.8
> zajic = 22.21
> bazant = 14.72
> ls()
[1] "bazant" "x" "X" "z" "zajic"
> ls(pattern="^z")    výpis objektů začínajících na „z”
[1] "z" "zajic"
> ls(pattern="z")     výpis objektů obsahujících „z”
[1] "bazant" "z" "zajic"
rm(x)                 smaž objekt x
rm(list=ls(pattern="^z")) smaž všechny objekty začínající na „z”
rm(list=ls())         smaž všechny objekty (opatrně!)
```

Volání funkcí

Většina funkcí v R má mnoho argumentů, ale zadávají se jen některé. Argumenty funkcí v R totiž mohou mít předdefinované hodnoty nebo mohou zůstat zcela nespécifikované. Pokud se volá funkce bez argumentů, je třeba uvést prázdné závorky; jinak se pouze vypíše definice funkce (zkuste `ls` versus `ls()`). Argument je možné identifikovat jménem, tak jako `ls(pattern="z")`,

anebo pořadím. Např. `ls(2)` je totéž jako `ls(name=2)`, neboť `name` je první argument funkce `ls`. Jaké má určitá funkce argumenty, jaké jsou jejich předdefinované hodnoty a jaké mají argumenty význam, to vše je vysvětleno v nápovědě. Stačí například zadat `help(ls)`.

Datové typy

R rozlišuje čtyři základní datové typy (*modes*): numerický, znakový, komplexní a logický. Typ objektu zjistíme funkcí `mode()`. Chybějící hodnoty jsou representovány kódem `NA` (*Not Available*). Speciální numerické hodnoty jsou `Inf`, `-Inf` a `NaN` ($+\infty$, $-\infty$, *Not A Number*). Znakové objekty se skládají z řetězců, které je nutno zadávat v dvojítech uvozovkách ("`Toto je retezec`"). Logické objekty obsahují konstanty `TRUE` a `FALSE` (psáno samými velkými písmeny!).

Datové struktury

Základní datové struktury v R jsou mj. vektor (*vector*), matice (*matrix*), pole (*array*), datová tabulka (*data frame*) a seznam (*list*).

Vektory:

<code>> b = c(1,5,8,1,5)</code>	Vytvoření vektoru z čísel
<code>> b</code>	Výpis
<code>[1] 1 5 8 1 5</code>	
<code>> b[4]</code>	Hodnota daného prvku
<code>[1] 1</code>	
<code>> b = 5:14</code>	Aritmetická posloupnost s krokem 1
<code>> b</code>	
<code>[1] 5 6 7 8 9 10 11 12 13 14</code>	
<code>> b[3:6]</code>	Subvektor
<code>[1] 7 8 9 10</code>	
<code>> b<7</code>	Logická operace na vektor
<code>[1] TRUE TRUE FALSE FALSE FALSE FALSE FALSE</code>	
<code>FALSE FALSE FALSE</code>	
<code>> b[b>9]</code>	Hodnoty splňující podmínku
<code>[1] 10 11 12 13 14</code>	
<code>> z = rep("+",4)</code>	Vytvoření vektoru identických prvků (zde znakových)
<code>> z = c(z,rep("-",4),rep("+",2))</code>	Ukázka lepení vektorů
<code>> z</code>	
<code>[1] "+" "+" "+" "+" "-" "-" "-" "-" "+" "+"</code>	
<code>> b[z=="-"]</code>	Prvky jednoho vektoru na místech, kde druhý vektor splňuje podmínku
<code>[1] 9 10 11 12</code>	
<code>> q = seq(1,8,by=0.05)</code>	Aritmetická posl. se zvoleným krokem
<code>> length(q)</code>	Délka vektoru

Matice:

<code>> mat = cbind(b,rep(1,10))</code>	Vytvoření matice ze sloupců
<code>> mat = rbind(b,rep(1,10))</code>	Vytvoření matice z řádků
<code>> mat = matrix(c(1:6),c(11:16),ncol=3)</code>	Vytvoření matice z prvků po sloupcích
<code>> mat = matrix(1:12,ncol=3,byrow=T)</code>	Vytvoření matice z prvků po řádcích
<code>> dim(mat)</code>	Dimenze matice
<code>[1] 4 3</code>	
<code>> mat[1,3]</code>	Prvek matice
<code>> mat[3,]</code>	Řádek matice
<code>> mat[,2]</code>	Sloupec matice
<code>> mat[1:3,2:3]</code>	Submatice

Seznam (*list*):

Seznam je datová struktura, jejíž složky mohou mít různý typ i různou délku (rozměr). Každá složka seznamu má své jméno. Výsledky většiny statistických analýz včetně regresních modelů jsou objekty typu seznam.

```
> l = list(cislo=15,vektor=q,logic=(b<7),
+ znaky=z)
> l$cislo
[1] 15
> l[[1]]
[1] 15
> l$logic[6:8]
[1] T F F
> l[[2]][3]
[1] 1.100000
> names(l)
```

Vytvoření seznamu
Přístup ke složkám jménem
Přístup ke složkám pořadím
Přístup k prvkům složek
Příklad řetězení odkazů
Jména složek seznamu

Datová tabulka (*data frame*):

Datová tabulka je speciální případ seznamu, jehož složky jsou vektory různého typu, ale stejné délky. Datovou tabulku lze považovat za zobecněnou matici, jejíž řádky odpovídají pozorováním a sloupce veličinám.

```
> a = data.frame(x1=c(rep(1,8),rep(0,8)),
+ pohl=rep(c("Muz","Zena"),c(8,8)))
> names(a)
> names(a)[1] = "Skupina"
> names(a)
> a$pohl
> attach(a)
> search()

> Skupina[3]
```

Vytvoření datové tabulky
Výpis jmen veličin
Změna jména veličiny
Přístup k veličinám jménem
Zavedení datové tabulky k aktivní práci
Seznam aktivních datových tabulek a knihoven
Nyní je možný přímý přístup k veličinám (bez \$)

Faktor:

Faktor je speciální forma znakového vektoru. Má atribut `levels` (úrovně) a používá se pro diskrétní veličiny. Pro faktor je možné předem nastavit kódování parametrů použité později při zařazení faktoru do regresního modelu.

```
> fac = factor(rep(c(0,1,9),c(8,3,2)))
> levels(fac)
> levels(fac) = c("Chlap","Zenska","???"")
> is.factor(a$pohl)
> levels(a$pohl)
```

Vytvoření (konverse) faktoru
Výpis úrovní
Změna úrovní
Veličina `pohl` je již faktor
Výpis úrovní

Operátory

Operátory aplikované na vektory, matice nebo pole se provádějí po složkách. Výsledkem je opět vektor, matice nebo pole.

	Aritmetické		Relační		Logické
+	Sčítání	<	menší	!	negace
-	Odečítání	>	větší	&	a zároveň
*	Násobení	<=	menší nebo rovno		nebo
/	Dělení	>=	větší nebo rovno		
^	Mocnění	==	rovná se		
%%	Modulo	!=	nerovná se		
%%/	Celočíselné dělení				

Elementární funkce

Skalární: Přijímají za argument skalár, vektor, matici. Provádějí se po složkách. Příklady: `sqrt`, `exp`, `log`, `log10`, `gamma`, `abs`, `round`, `trunc`.

```
> mat = matrix(seq(1,by=0.5,length=12),ncol=4,byrow=T)
> sqrt(mat)
> round(exp(-mat),4)
> round(mat*mat)
```

Vektorové: Přijímají za argument vektor nebo matici. Matice se transformuje na vektor po sloupcích. Příklady: `length`, `sum`, `prod`, `min`, `max`, `range`, `mean`, `median`, `quantile`, `var`, `cor`, `cumsum`, `cumprod`.

```
> v = seq(1,by=0.5,length=12)
> sum(v)                Součet
> min(v)               Minimum
> range(v)             Vektor minima a maxima
> mean(v)              Průměr
> var(v)               Výběrový rozptyl
> quantile(v,c(0.3,0.5)) Výběrové kvantily
> cumsum(v)            Kumulativní součty
```

Maticové: Přijímají za argument matici.

```
> mat1 = matrix(c(5,0,-1,4),nrow=2)
> mat2 = matrix(c(2,-2,0,1),nrow=2)
> t(mat1)              Transpozice
> diag(mat1)          Diagonála
> mat1%*%mat2         Maticový součin
> solve(mat1)         Inverse
> colSums(mat1)       Vektor sloupcových součtů
> rowMeans(mat1)     Vektor řádkových průměrů
```

Kontrolní příkazy a smyčky

```
if (cond){expressions}
if (cond){exp1} else {exp2}
while (cond) {expr}
for (variable in range) {expr}
ifelse(vec.cond,vec.expr.1,vec.expr.2)
```

Příkazu `for` se lze většinou vyhnout. R má řadu funkcí, které zpracovávají celé objekty najednou bez nutnosti chodit po složkách. Velmi užitečné jsou funkce `apply()`, `tapply()`, `sapply()` a `lapply()`.

<i>Funkce:</i>	<i>Vstup:</i>	<i>Činnost:</i>
<code>apply()</code>	Matice, pole	Proveď danou funkci na daný rozměr/sekci matice/pole <i>Př:</i> <code>apply(x,2,sum)</code> vektor sloupcových součtů matice <code>x</code> (totéž jako <code>colSums(x)</code>)
<code>tapply()</code>	Vektor	Rozděl vektor na skupiny a spočítej funkci pro každou skupinu <i>Př:</i> <code>tapply(y,list(pohl,rasa),mean)</code> vytvoř tabulku průměrů veličiny <code>y</code> pro každou kombinaci pohlaví a rasy
<code>lapply()</code>	Seznam, data frame	Proveď danou funkci na každou položku seznamu <i>Př:</i> <code>lapply(a,summary)</code> spočítej základní popisné statistiky ze všech veličin datové tabulky <code>a</code> a udělej z nich seznam
