

1 Tvorba dynamických dokumentů

1.1 Základní informace o dynamických dokumentech

Velmi užitečným nástrojem pro vytváření efektivních reportů analýzy dat je funkce Sweave. Tato funkce, implementovaná v programovacím jazyce \mathbb{R} umožňuje integraci \mathbb{R} kódu do \LaTeX ových dokumentů. Zmíněná integrace nám umožňuje vytvářet tzv. *dynamické dokumenty*. Dynamický dokument je klasický \LaTeX ový dokument sestávající z *pevné* a *dynamické* složky.

Pevná složka se ničím neliší od zdrojového kódu klasického \LaTeX ového dokumentu. Sestává ze dvou částí, a sice *hlavičky* a *těla* dokumentu. Hlavička dokumentu determinuje vzhled a nastavení celého dokumentu (jazyk vysázeného textu, načítání balíčků, rozložení stránek, nastavení čítačů, definování nových prostředí a příkazů, nastavení indexů a rejstříků, apod.). Tělo dokumentu potom slouží jako prostor pro sazbu a formátování klasického či matematického textu, vytváření seznamů, vkládání obrázků a částí kódu, tvorbu tabulek, apod. Stejně jako v klasickém \LaTeX ovém dokumentu je obsah těla dokumentu pevný a neměnný. Například každý obrázek je pouze načtením a zobrazením externího obrázku, přičemž změna obsahu obrázku je možná pouze prostřednictvím změny v externím obrázku. Stejně tak tabulka vysázená v těle dokumentu je přesně determinována vysázenými hodnotami a jakákoli změna v tabulce je možná pouze ručním přepisem hodnot.

Dynamická složka umožňuje vznik dynamického obsahu dokumentu. Vytváříme ji v rámci těla dokumentu definováním tzv. *dynamických prostředí* (angl. *chunks*¹), což jsou prostředí obsahující \mathbb{R} kód. \mathbb{R} kód umístěný v dynamickém prostředí se chová jako klasický kód vytvořený například v \mathbb{R} skriptu. Nestanovíme-li jinak, dochází při každé kompilaci zdrojového kódu k vyhodnocení \mathbb{R} kódu umístěného v dynamickém prostředí a k zobrazení jeho výstupů ve zkompilovaném dokumentu. Forma výstupů zcela závisí na volbě uživatele. Výstupy tak mohou být zobrazovány například jako číselné hodnoty, obrázky, tabulky, animace, atd. Umístění výstupů ve zkompilovaném dokumentu bývá zpravidla ve stejné části textu, v jaké se nachází dynamické prostředí v těle dokumentu. V případě nedostatku místa na stránce jsou výstupy umístěny na nejbližší možné pozici. Oproti pevné složce má dynamická složka tu výhodu, že jakákoli změna v provedeném kódu či jeho výstupech se po kompilaci souboru automaticky promítne do zkompilovaného dokumentu.

1.2 Nastavení R Sweave

1.2.1 Instalace softwaru

Abychom mohli plně využívat komfortu dynamických dokumentů, musíme si pro jejich tvorbu nejprve přichystat potřebný software. Předně je nezbytné mít v počítači staženou a nainstalovanou nejlépe aktuální verzi programovacího jazyku \mathbb{R} . Nastavení R Sweave, stejně jako následnou tvorbu dynamických dokumentů výrazně usnadňuje a zpříjemňuje nadstavbový software RStudio, proto silně doporučujeme také instalaci aktuální verze tohoto softwaru. Dále je třeba mít v počítači nainstalovaný software na sazbu a kompilaci \TeX u a \LaTeX u. Jako vhodný sázeční systém doporučujeme zvolit software MikTeX, který se v souvislosti s tvorbou dynamických dokumentů prostřednictvím R Sweave osvědčil. Tento software bývá v souvislosti s R Sweave a tvorbou dynamických dokumentů také nejčastěji doporučovaný komunitou uživatelů \mathbb{R} . Software MikTeX je volně dostupný ke stažení na stránce <https://miktex.org/> pro všechny v Česku a na Slovensku nejčastěji využívané platformy, jako jsou Windows, Mac, Linux a Docker. Uživatelům Macu bývá případně komunitou uživatelů \mathbb{R} také doporučován sázeční systém MacTeX, volně dostupný ke stažení na stránce <https://rpubs.com/YaRrr/SweaveIntro>. Nejen uživatelům, kteří se necítí být silní ve znalostech \LaTeX u, důrazně doporučujeme také instalaci vhodného \LaTeX ového editoru, který se při tvorbě dynamických dokumentů záhy ukáže jako neocenitelný pomocník při hledání chyb ve zdrojovém kódu. Z neznámějších editorů uvádíme pro příklad editory TeXmaker, TeXnicCenter nebo TeX Live. Z uvedených editorů doporučujeme praxí prověřený editor TeXmaker v kombinaci se sázečním systémem MikTeX, nebo editor TeX Live v kombinaci se sázečním systémem MacTeX.

¹Anglický pojem *chunk* bývá v českých textech buď ponecháván v nezměněné podobě, nebo překládán formou fonetického přepisu (*čank*). Protože však ani jeden přístup není správný, v tomto textu se podobnému názvosloví vyhneme a nahradíme jej právě pojmem *dynamické prostředí*.

1.2.2 Instalace balíčků knitr a xtable

Po dokončení instalace potřebného softwaru zbývá doladit několik detailů týkajících se programovacího jazyka \mathbb{R} a nastavení RStudia. Nejprve je třeba stáhnout a nainstalovat balíčky `knitr` a `xtable`. Instalaci balíčků provedeme v RStudiu buď ručně prostřednictvím záložky `Packages` v multifunkčním okně, nebo zadáním následujících příkazů do konzole.

```
1 install.packages('knitr')
2 install.packages('xtable')
```

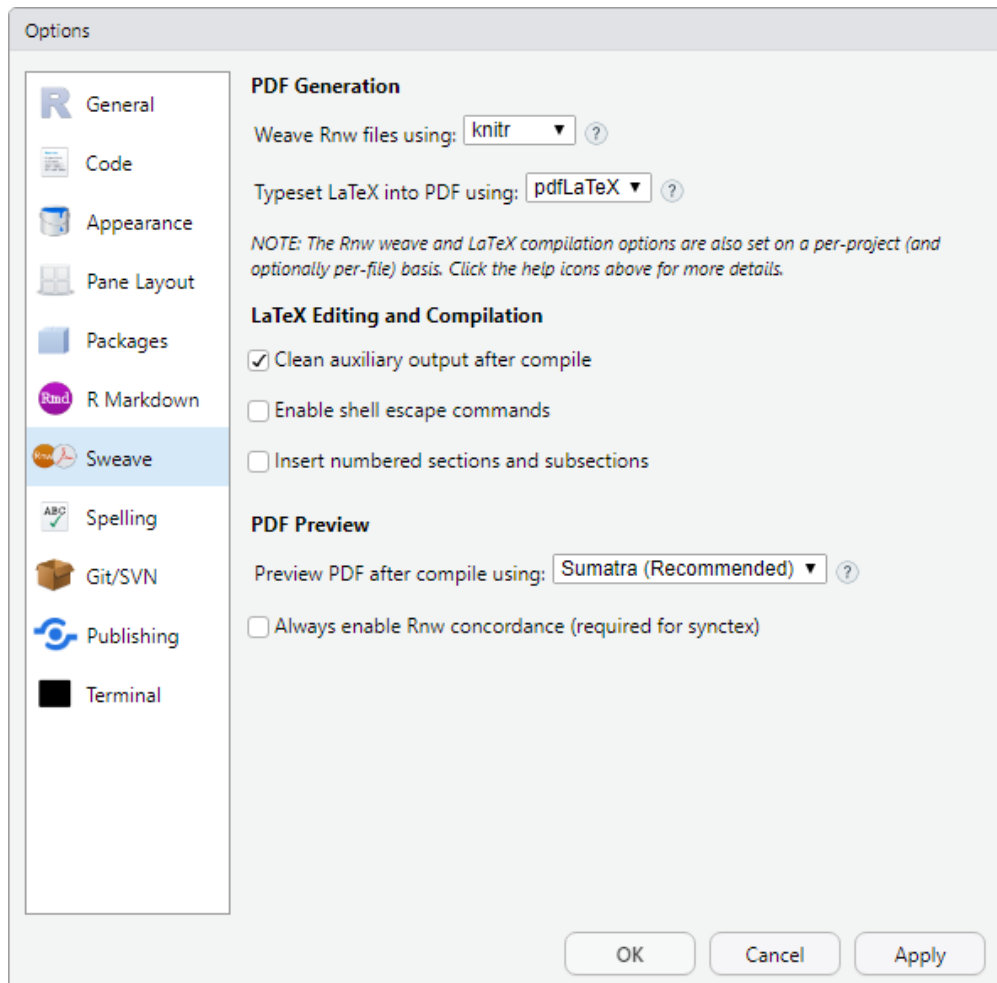
Poznámka: Balíček `knitr` vznikl spojením několika balíčků a funkcí, které průběžně vznikaly za účelem pohodlné tvorby dynamických dokumentů prostřednictvím funkce `Sweave()`. Součástí balíčku `knitr` jsou například samotná funkce `Sweave()`, balíčky `cacheSweave` a `weaver` umožňující ukládání výsledků \mathbb{R} kódu do mezipaměti, čímž dochází k významnému zkrácení výpočetní doby \mathbb{R} kódu, a tedy i doby kompilace celého zdrojového kódu, balíček `pgfSweave`, který umožňuje ukládání grafiky do mezipaměti (v návaznosti na balíček `cacheSweave`) a zajišťuje barevné zvýraznění \mathbb{R} kódu zobrazeného jako výstup ve zkompilevaném dokumentu, funkce `saveLatex()` z balíčku `animation` umožňující tvorbu animací, apod. Balíček `knitr` dále disponuje nově vytvořenými funkcemi i různými modifikacemi a nadstavbami funkcí z výše uvedených balíčků, které společně zjednodušují tvorbu výstupů (například umožňují přímé vykreslení grafů bez nutnosti použití funkce `print()`) a zvyšují transparentnost výstupů \mathbb{R} kódu (například poskytují volbu mezi výpisem a potlačením výpisu varovné hlášky, je-li tato hláška součástí výstupu \mathbb{R} funkce). S touto bohatou výbavou poskytuje balíček `knitr` kompletní soubor funkcí potřebných pro co nejpohodlnější tvorbu \mathbb{R} kódu v dynamickém prostředí, efektivní spuštění kódu, hladké vyhodocení jednotlivých příkazů a zařazení výstupů kódu ve správném tvaru a na správnou pozici do zkompilevaného dokumentu.

Balíček `xtable` slouží ke konverzi tabulek získaných jako výstupy \mathbb{R} kódu do podoby \LaTeX ové tabulky. Prostřednictvím funkce `xtable()` umožňuje konverzi \mathbb{R} objektů různých datových typů (např. `numeric`, `integer`, `data.frame`, `tibble`, apod.) na objekty typu `xtable`. Objekt typu `xtable` má tvar tabulky, a to podle nastavení buď v podobě \LaTeX ového kódu, nebo v podobě HTML kódu. V kombinaci s funkcí `print()` pak můžeme libovolný datový objekt převést na objekt typu `xtable` a následně jej v rámci kompilace zdrojového kódu automaticky vysázat jako \LaTeX ovou tabulku.

1.2.3 Nastavení RStudia pro tvorbu dynamických dokumentů

Po instalaci balíčků `knitr` a `xtable` zbývá doladit nastavení RStudia. V hlavní nabídce vybereme panel `Tools` → `Global Options...` V okně `Options` vybereme v nalevo umístěném vertikálním panelu záložku `Sweave`. Tato záložka obsahuje veškerá nastavení RStudia související s tvorbou dynamických dokumentů prostřednictvím `Sweave`.

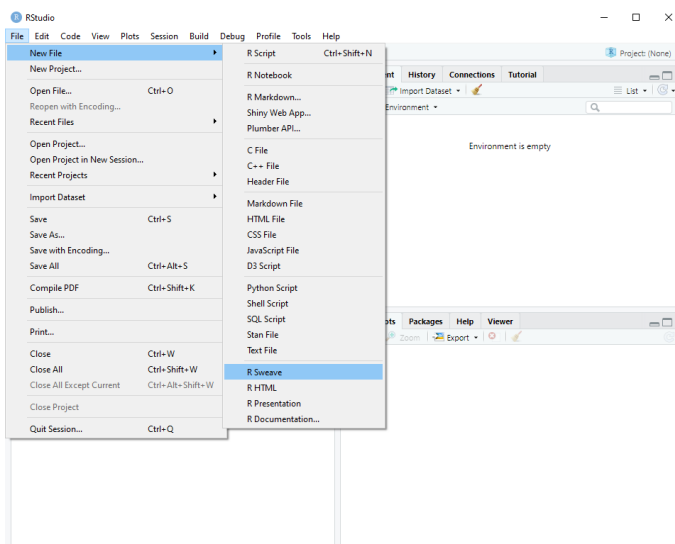
Ukázka optimálního nastavení RStudia pro tvorbu dynamických dokumentů prostřednictvím záložky `Sweave` v panelu `Tools` → `Global Options...` je zobrazena na obrázku 1. Závěrem mějme paměti, že k tomu, aby kompilace zdrojového kódu proběhla úspěšně na různých počítačích, je třeba, aby nastavení v panelu `Tools` → `Global Options...` → `Sweave` bylo na těchto počítačích (s výjimkou volby prohlížeče) shodné.



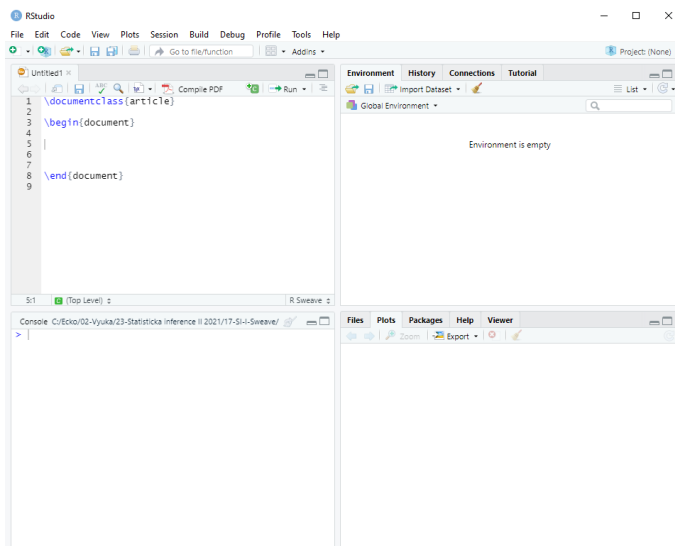
Obrázek 1: Ukázka optimálního nastavení RStudio pro tvorbu dynamických dokumentů

1.3 Vytvoření nového .Rnw souboru

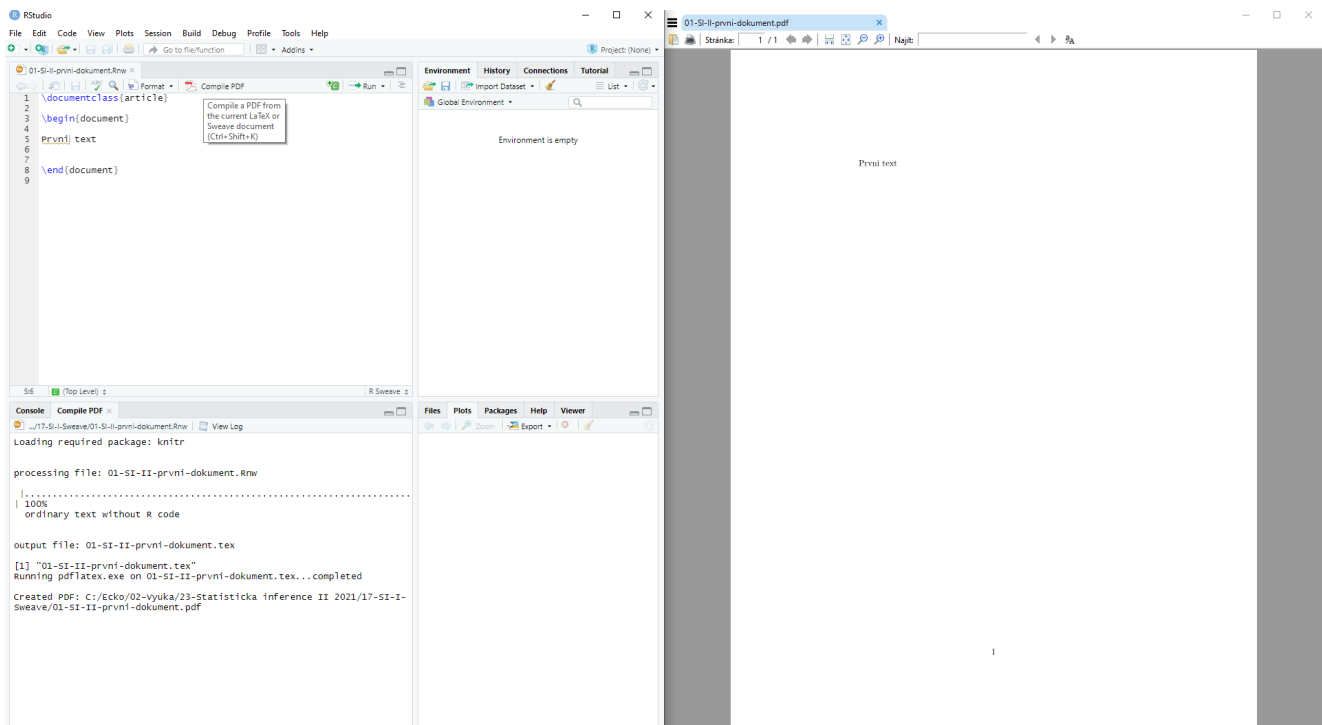
Po nastavení RStudio pro tvorbu dynamických dokumentů můžeme přistoupit k vytvoření našeho prvního dynamického dokumentu. Práci na novém dynamickém dokumentu započneme vytvořením .Rnw souboru pomocí panelu File → New File → R Sweave (viz obrázek 2). Po zvolení uvedené kombinace se vytvoří .Rnw soubor (viz obrázek 3). Kliknutím na ikonu diskety, nebo pomocí panelu File → Save As .Rnw soubor pojmenujeme a uložíme do naší pracovní složky. Název souboru volíme rozumně, **bez mezer a bez diakritiky**, jinak soubor nebude pracovat správně. Po uložení napíšeme do těla dokumentu libovolný text (pro začátek bez diakritiky) a klikneme na ikonu Compile PDF (viz obrázek 4 vlevo). Uvedený .Rnw soubor se zkompiluje a získáme .pdf soubor obsahující námi zapsaný text (viz obrázek 4 vpravo).



Obrázek 2: Ukázka vytvoření nového .Rnw souboru



Obrázek 3: Ukázka nově vytvořeného .Rnw souboru



Obrázek 4: Ukázka kompilace .Rnw souboru (vlevo) a výsledek kompilace (vpravo)

1.4 Využití šablony s přednastavenou hlavičkou dynamického dokumentu

Pokud se vám povedlo svůj první .Rnw soubor zkompilevat, můžeme se posunout dál. Ve studijních materiálech je k dispozici soubor 01-SI-II-Sablona.Rnw. Tento soubor typu .Rnw (tedy stejného typu, jako právě vytvořený první .Rnw soubor) obsahuje šablonu pro .Rnw soubor. Šablona má nadefinovanou hlavičku pevné složky, v rámci které je zdefinován soubor základních L^AT_EXbalíčků a dále nastavení vlastního prostředí pro \mathbb{R} -kód a výstupy \mathbb{R} kódu. V těle dokumentu je potom uvedena prvotní ukáзка dynamické složky, jejíž prostředí je zahájeno symboly $\langle\langle\rangle\rangle=$ a zakončeno symbolem @ (ukáзка šablony viz obrázek 5).

Vaším úkolem nyní je uvedenou šablonu stáhnout, kompletně nakopírovat do vašeho prvního .Rnw souboru a soubor zkompilevat, abyste se přesvědčili, že se šablona na vašem počítači překompiluje bez problémů. Po úspěšném zkompilevání si šablonu detailně prostudujte, abyste jí rozuměli. V případě, že se vám přednastavené prostředí \mathbb{R} kódu a jeho výstupů bude líbit a že se vám s šablonou bude dobře pracovat, můžete tuto šablonu ze začátku používat jako výchozí šablonu pro práci s .Rnw soubory. Věřím, že časem si najdete vlastní styl, který vám vyhovuje a šablonu si modifikujete k obrazu svému :).

K pochopení šablony se vám budou hodit informace o univerzálním sázení diakritiky uvedené v sekci 1.5 a dále popisy vybraných argumentů dynamických prostředí uvedené v sekci 1.6.1.

1.5 Univerzální sázení diakritiky

Při vytváření .Rnw souborů s českou či slovenskou diakritikou musíme počítat, s různým kódováním počítačů. Nekompatibilní kódování vede k tomu, že na jednom počítači se text souboru překompiluje bez chyby a na jiném počítači bude výstupem kompilace „rozsypaný čaj“. Nejjednodušší způsob, jak zajistit, aby námi vytvořený zdrojový kód Rnw souboru byl kompatibilní pro každý počítač, je sázet písmena s diakritikou pomocí následujících L^AT_EXpříkazů.

- v textu:
 - čárka: \’a ... á
 - háček: \v{ r } ... ř;
 - kroužek: \r{ u } ... ů;
- v těle dynamického prostředí:
 - čárka: \\’a ... á;
 - háček: \\v{ r } ... ř;
 - kroužek: \\r{ u } ... ů.

```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
01-SI-II-Sablona.Rnw
Format Compile PDF
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2
3 % O 1 - R N W S O U B O R - S A B L O N A
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5
6
7 \documentclass{article}
8
9 %----- J A Z Y K O V E B A L I C K Y -----
10
11 \usepackage{czech}[babel]
12 \usepackage{cp1250}[inputenc] % pro iso8859-2
13 %\usepackage{utf8}[inputenc] % pro unicode UTF-8
14
15 %----- O S T A T N I B A L I C K Y -----
16
17 \usepackage{geometry}
18 \geometry {verbose, tmargin = 2cm, bmargin = 3cm, lmargin = 2cm, rmargin = 2cm}
19 \usepackage{url, fancyhdr, enumerate, comment}
20 \usepackage{amsthm, amsmath, amssymb, thmtools}
21 \usepackage{booktabs, longtable, multirow, multicol, caption, bigstrut}
22 \usepackage{xcolor, animate}
23 \usepackage{float, subfig, hyperref}
24
25 %----- D E F I N I C E P R I K A Z U -----
26
27 \newcommand{\mbr}{\mathbb{R}}
28 \newcommand{\RR}{\includegraphics[angle = 0,width = 0.025\textwidth]{0-Rlogo.jpg} }
29
30 %----- N A S T A V E N I C I T A C U -----
31
32 \setcounter{secnumdepth}{3}
33 \setcounter{tocdepth}{2}
34 \numberwithin{equation}{section}
35
36 %----- D E F I N I C E B A R E V -----
37
38 \definecolor{dgray} {gray}{0.35} % barva textu komentaru
39 \definecolor{lgray} {gray}{0.92} % barva pozadi R-kodu
40 \definecolor{l1gray} {gray}{0.96} % barva pozadi R-vystupu
41 \definecolor{dred} {rgb}{0.40, 0, 0}
42 \definecolor{ddred} {rgb}{0.35, 0, 0}
43
44 %----- R P R O S T R E D I K O D U -----
45
46 \lstdefinestyle {Rstyle}{
47   language = R, % nastaveni vzhledu
48   basicstyle = \ttfamily \small, % nastaveni jazyka R
49   backgroundcolor = \color{lgray}, % typ a velikost pisma
50   commentstyle = \ttfamily \small \itshape \color{dgray}, % barva pozadi
51   showstringspaces = false, % barva komentare k funkcim
52   numbers = left, % zakaz zvyraznovani mezer
53   numberstyle = \ttfamily \small, % cislovani vlevo
54   stepnumber = 1, % typ pisma a velikost cislovani
55   firstnumber = last, % cislovani po kroku jedna
56   breaklines = T, % kumulativni cislovani radku
57   aboveskip = 1\baselineskip,
58   belowskip = 1\baselineskip,
59   keywords = {smooth} % vymaze seznam klucovych slov
60 }
61
62 %----- R P R O S T R E D I V Y S T U P U -----
63
64 \lstdefinestyle {Routstyle}{
65   language = R, % nastaveni vzhledu R-vystupu
66   basicstyle = \ttfamily \footnotesize \color{dred}, % nastaveni jazyka R
67   commentstyle = \ttfamily \small \itshape \color{dgray}, % typ a velikost pisma
68   rulecolor = \color{ddred}, % barva ramecku okolo vystupu
69   backgroundcolor = \color{l1gray}, % barva pozadi
70   showstringspaces = true, % barva ramecku okolo vystupu
71   numbers = right, % zakaz zvyraznovani mezer
72   numberstyle = \ttfamily \small, % cislovani vpravo
73   firstnumber = last, % typ pisma a velikost cislovani
74   frame = single, % kumulativni cislovani radku
75   framexleftmargin = -0.1em, % ramecek okolo outputu
76   framexrightmargin = -0.2em,
77   framesep = 1.25pt,
78   framesule = 0.25pt,
79   aboveskip = 0.1cm,
80   belowskip = 0.2cm,
81   keywords = {smooth}, % vymaze seznam klucovych slov
82   keywordstyle = \color{red}, % barva klucovych slov - cerven
83   otherkeywords = {Error, error, warning, warning, warnings, warnings} % zvyrazneni klucovych
84   slov jako warning
85 }
86 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
87
88 \begin{document}
89 << setup, echo = F >>=
90 # set global chunk options
91 opts_chunk\set{root.dir = '', fig.path = '01-Figure/figure-',
92 fig.show = 'hold', fig.align = 'center',
93 fig.pos = 'ht', highlight = F}
94 render_listings() #
95 @
96
97 << echo = F, message = F, warning = F >>=
98 \library{knitr} # nacteni balicku knitr
99 \library{xtable} # nacteni balicku xtable
100 @
101
102 \noindent
103 Toto je z\'akladn\'i \v{s}ablona pro tvorbu dynamick\'ych dokument\{r}{u}.
104
105 << echo = T >>=
106 x <- rnorm(10) # generovani pseudonahodnych cisel
107 tab <- data.frame(m = mean(x), var = var(x), sd = sd(x)) # souhrnna tabulka vysledku
108 round(tab, 4) # vypis tabulky
109 @
110
111 \end{document}
93:39 Chunk 1: setup z 7 R Sweave z

```

Obrázek 5: Předdefinovaná šablona dynamického dokumentu

1.6 Vybrané argumenty dynamického prostředí

V tomto odstavci si uvedeme přehled základních argumentů, které můžeme definovat v záhlaví každého dynamického prostředí a se kterými se setkáte při vytváření vašeho prvního Rnw souboru. Rozšířený přehled (nejen zde uvedených) argumentů dynamického prostředí je k dispozici na webové adrese <https://yihui.name/knitr/options/>.

Uvedené argumenty je možné vkládat buď do dynamického prostředí `<< setup >>= ...@` jako argumenty funkce `opts_chunk$set()`, nebo přímo do hlaviček dynamických prostředí. Argumenty definované v prostředí `<< setup >>= opts_chunk$set(...)` @ fungují jako defaultní nastavení každého dynamického prostředí, není-li v hlavičce dynamického prostředí tento argument vyložene definován jinak. Např. pokud bychom zapsali `opts_chunk$set(echo = T)`, znamená to, že kód uvedený v každém dynamickém prostředí `<<>>= ...@` se po překompilování automaticky vypíše ve výsledném .pdf souboru, aniž bychom v hlavičce každého dynamického prostředí museli zapisovat `<< echo = T >>=`. Pokud bychom v některém dynamickém prostředí chtěli vypsání kódu potlačit, musíme v hlavičce dynamického prostředí explicitně zapsat `<< echo = F >>=`.

1.6.1 Argumenty nastavené defaultně pro všechna dynamická prostředí

- platnost těchto argumentů se vztahuje ke všem dynamickým prostředí v .Rnw souboru, není-li v hlavičce dynamického prostředí tento argument definován jinak
- jde o argumenty, které v naší šabloně figurují v prostředí `<< setup >>= ...@` jako argumenty funkce `opts_chunk$set()`; tyto argumenty tedy specifikují automatická nastavení pro všechna dynamická prostředí; poznamenejme, že do prostředí `<< setup >>= opts_chunk$set()` @ můžete podle vlastního uvážení libovolné argumenty přidávat, nebo odebírat.


Seznam argumentů

- `setup ...` dynamické prostředí s tímto záhlavím umožňuje vytvořit defaultní nastavení všech dynamických prostředí;
- `root.dir = " "` ... automatické nastavení složky, ve které je .Rnw soubor uložený, jako aktuálního adresáře;
- `fig.path = "01-Figure/figure-"` ... v kombinaci s příkazem `root.dir = " "` dochází k tomu, že v aktuálním adresáři se vytvoří složka 01-Figure a do této složky se budou vkládat všechny obrázky, které se během kompilace .Rnw souboru vytvoří; název každého obrázku bude začínat slovy `figure-`;
- `fig.show = "hold"` ... vygenerované obrázky se vykreslí pod sebe, všechny obrázky budou mít dohromady jeden společný popisek shodný s popisem v argumentu `fig.cap = "..."`;
- `fig.align = "center"` ... vygenerované obrázky se automaticky zarovnájí na střed (tj. uprostřed mezi pravým a levým okrajem stránky);
- `fig.pos = "ht"` ... je-li to možné, vygenerovaný obrázek se umístí v textu na stejnou pozici, jako ve zdrojovém kódu (plovoucí umístění obrázku); není-li to možné, umístí se na začátek stávající, nebo následující stránky;
- `highlight = T` ... kód zobrazený ve zkompilevaném .pdf souboru je automaticky zbarvován barevně v souladu s barevným nastavením kódu podle protokolu R Sweave (platí pouze v případě, že za příkazem `opts_chunk$set()` nenásleduje silnější příkaz `render_listings()`, který přebíjí defaultní nastavení vlastním nastavením definovaným v hlavičce dokumentu)
- `highlight = F` ... kód zobrazený ve zkompilevaném .pdf souboru je automaticky zbarvován černobíle v souladu s černobílým nastavením kódu podle protokolu R Sweave (platí pouze v případě, že za příkazem `opts_chunk$set()` nenásleduje silnější příkaz `render_listings()`, který přebíjí defaultní nastavení vlastním nastavením definovaným v hlavičce dokumentu)


1.6.2 Argumenty, se kterými se v šabloně setkáte v hlavičkách jednotlivých dynamických prostředí

- (platnost těchto argumentů se vztahuje pouze k jednomu dynamickému prostředí)

Seznam argumentů:

- `echo = T` ... vypiš kód; `echo = F` ... nevypisuj kód;
- `eval = T` ... vyhodnoť příkazy v dynamickém prostředí; `eval = F` ... nevyhodnocuj příkazy v dynamickém prostředí (příkazy se nevyhodnotí; klasické výsledky příkazů, obrázky, animace nebo tabulky se nevypíší);
- `results = "asis"` ...  výstup vysáže jako \LaTeX tabulku;
- `warning = F` ... varování (warnings), která provází výstup funkce, se nevypíší; `warning = T` ... varování, která provází výstup funkce se vypíší;
- `message = F` ... zprávy, které provází výstup funkce, se nevypíší; `message = T` ... zprávy, které provází výstup funkce, se vypíší;
- `fig.height = 5` ... absolutní výška obrázku (nejčastěji volíme hodnotu mezi 3–15, lze ale volit vyšší i nižší čísla; čím je hodnota `fig.height` vyšší, tím je obrázek **větší**, a popisky v grafu **menší**);
- `fig.width = 5` ... absolutní šířka obrázku (nejčastěji volíme hodnotu mezi 3–15, lze ale volit vyšší i nižší čísla; čím je hodnota `fig.height` vyšší, tím je obrázek **větší**, a popisky v grafu **menší**; pokud `fig.height = fig.width`, je obrázek čtvercový);
- `out.width = "0.44\linewidth"` ... relativní velikost obrázku vzhledem k délce jednoho řádku (`1\linewidth` ... graf je v rozsahu od levého okraje textu k pravému okraji textu; `0.5\linewidth` ... graf má velikost poloviny šířky rozsahu textu, apod.);
- `fig.cap = "\label{Malajsie} Popisek obrázku"` ... vytváří popisek obrázku
- `fig.show = "animate"` ... vygenerované obrázky se nevykreslí pod sebe, ale automaticky se poskládají do animace (efektivní a elegantní tvorba animace);
- `fig.show = "asis"` ... vygenerované obrázky se vykreslí pod sebe, každý obrázek bude mít popisek shodný s popiskem v argumentu `fig.cap = "..."`;
- `fig.show = "hold"` ... vygenerované obrázky se vykreslí pod sebe, všechny obrázky budou mít dohromady jeden společný popisek shodný s popiskem v argumentu `fig.cap = "..."`.

1.7 Argumenty příkazů `xtable()` a `print.xtable()`

Praktickou vlastností dynamických dokumentů je, že tabulkový výstup  funkcí dokážeme automaticky přetransformovat na \LaTeX tabulku. K tomu používáme kombinaci příkazů `xtable()` (viz nápověda funkce `xtable`) a `print()` (viz nápověda funkce `print.xtable`). Zde si uvedeme přehled vybraných argumentů obou funkcí, se kterými se v rámci vypracování samostatné práce setkáte.

Vybrané argumenty funkce `xtable()`

- `align = "l|rrrrr"` ... definuje zarovnání sloupců tabulky, jako to známe z vytváření tabulek v \LaTeX u;
- `digits = rep(4, 6)`, ... definuje zaokrouhlení čísel ve sloupcích tabulky;
- `caption = "Popisek tabulky"` ... definuje znění popisku tabulky;
- `label = "Odkaz"` ... definuje název odkazu tabulky; na tabulku se potom můžeme v textu odkazovat klasicky pomocí příkazu `ref()`.

Vybrané argumenty funkce `print.xtable()`

- `caption.placement = "top"` ... popisek tabulky se umístí nad tabulku;
- `hline.after = c(0, 2)` ... definujeme, pod kterými řádky tabulky se vykreslí vodorovná čára (`hline.after = c(0, 2)` značí, že se vykreslí dvě čáry, jedna pod nultým řádkem, tj. pod hlavičkou tabulky, a jedna pod druhým řádkem tabulky);
- `include.rownames = T` ... do \LaTeX tabulky se zahrnou názvy řádků původní tabulky typu `data.frame`;
- `include.rownames = F` ... do \LaTeX tabulky se nezahrnou názvy řádků původní tabulky typu `data.frame`;
- `include.colnames = T` ... do \LaTeX tabulky se zahrnou názvy sloupců původní tabulky typu `data.frame`;
- `include.colnames = F` ... do \LaTeX tabulky se nezahrnou názvy sloupců původní tabulky typu `data.frame`;
- `add.to.row = addtorow` ... do \LaTeX tabulky se doplní řádky podle seznamu uvedeného v proměnné `addtorow`; je-li v proměnné `addtorow` zdefinován řádek na pozici 0, je tím myšlena hlavička tabulky;
- `sanitize.text.function = function(x) x` ... je-li v původní tabulce typu `data.frame` v názvu řádku vložen \LaTeX vzorec (např. `"\sum"`), argument `sanitize.text.function = function(x) x` zajistí, že se vzorec přeloží a symbol \sum se po kompilaci vysází správně;
- `table.placement = "H"` ... je-li to možné, tabulka se umístí na pozici v textu, kde je tabulka umístěna ve zdrojovém kódu (analogie argumentu `fig.pos = "h"` u umístění obrázků).

2 Řešení samostatné práce

Nyní již máte k dispozici veškeré informace o dynamických dokumentech a argumentech dynamických prostředí, které potřebujete k vypracování samostatné práce. Níže máte k dispozici kód, pomocí kterého můžete svůj první dynamický dokument vytvořit.

Při vytváření dynamického dokumentu postupně zpracovávejte jednotlivé části uvedeného kódu a po dokončení každého dynamického prostředí si .pdf soubor zkompilujte, abyste měli jistotu, že pracujete bez chyb. Při přepisování kódu se soustřeďte na jednotlivé části kódu a uvědomujte si, k čemu které argumenty a příkazy slouží. Snažte se kód co nejlépe pochopit a klidně si zkoušejte části kódu různě modifikovat a zkoumejte, co modifikace způsobují :).

```
88 \begin{document}
89 << setup, echo = F >>=
90 # set global chunk options
91 opts_chunk$set(root.dir = '', fig.path = '01-Figure/figure-',
92               fig.show = 'hold', fig.align = 'center',
93               fig.pos = 'ht', highlight = F )
94 render_listings() #
95 @
96
97 << echo = F, message = F, warning = F >>=
98 library(knitr) # nacteni balicku knitr
99 library(xtable) # nacteni balicku xtable
100 library(car)
101 library(dplyr)
102 @
103
104 \section{Průzkumová analýza}
105 \subsection{Návtěn dat}
106 \noindent
107 {\bf Ukázka vypsání pseudokódu}
108 << echo = T, eval = F >>=
109 data <- read.delim("...", stringsAsFactors = F) # načti data
110 data <- na.omit(...) # odstran NA hodnoty
111 data$upface_cat <- car::recode(data$upface.H, "lo:70 = 1; 70.001:hi = 2")
112 head(...) # vypsání hlavičky proměnné data
113 dim(...) # vypočítej dimenzi tabulky data
114 @
115
116 \noindent
117 {\bf Ukázka vypsání kódu s výstupy}
118 << echo = T, eval = T >>=
119 data <- read.delim("15-anova-means-skull.txt", stringsAsFactors = F)
120 data <- na.omit(data)
121 data$upface_cat <- car::recode(data$upface.H, " lo:70 = 1; 70.001:hi = 2")
122 head(data)
123 dim(data)
124 @
125
126 \cleardoublepage
127 \subsection{Tabulky}
128
129 << echo = T >>=
130 upface <- data$upface.H
131 sex <- data$sex
132 pop <- data$pop
133 mal.m <- upface[pop == "mal" & sex == "m"]
134 per.m <- upface[pop == "per" & sex == "m"]
135 tab <- data.frame(m = c(mean(mal.m), mean(per.m)),
136                 s = c(sd(mal.m), sd(per.m)),
137                 med = c(quantile(mal.m, 0.50), quantile(per.m, 0.50)),
138                 q1 = c(quantile(mal.m, 0.25), quantile(per.m, 0.25)),
139                 q3 = c(quantile(mal.m, 0.75), quantile(per.m, 0.75)),
140                 row.names = c("malajska", "peruanska"))
141 tab <- round(tab, 4)
142 @
143
```

Obrázek 6: Řešení zadání samostatné práce 1/4

```

144 << echo = F, results = "asis" >>=
145 print(xtable(tab, align = "l|rrrrr", digits = rep(4, 6),
146           caption = paste("Tabuľka z\\'akladn\\'ich \\v{c}\\'iseIn\\'ych charakteristik"),
147           label = "Tab_zakl_char"), caption.placement = "top",
148           hline.after = c(0, 0), include.rownames = T, include.colnames = T) # add.to.row = addtorow,
149 @
150 \noindent
151 Z tabuľky \ref{Tab_zakl_char} vid\\'ime, \\v{z}e
v\\'y\\v{s}ka horn\\'i \\v{c}\\'asti tv\\'a\\v{r}e malajsk\\'e populace je m\\'irn\\v{e} ni\\v{z}\\v{s}\\'i ne\\v{z}
v\\'y\\v{s}ka horn\\'i \\v{c}\\'asti tv\\'a\\v{r}e peru\\'ansk\\'e populace.
152
153 << echo = F, warning = F, results = "asis" >>=
154 tab <- round(tab, 4)
155 addtorow <- list()
156 addtorow$pos <- list(0)
157 addtorow$command <- paste0(" & \\$\\bar{x}$ & \\$\\mathrm{s}$ & \\$x_{0.5}$ & \\$x_{0.25}$ & \\$x_{0.75}$ \\\\")
158
159 print(xtable(tab, align = "l|rrrrr", digits = rep(4, 6),
160           caption = paste("Tabuľka z\\'akladn\\'ich charakteristik"), label = "Tab_char"),
161           caption.placement = "top", hline.after = c(0, 0), include.rownames = T,
162           include.colnames = F, add.to.row = addtorow)
163 @
164 \noindent
165 Z tabuľky \ref{Tab_char} vid\\'ime, \\v{z}e sm\\v{e}rodatn\\'a odchyľka v\\'y\\v{s}ky horn\\'i \\v{c}\\'asti
tv\\'a\\v{r}e malajsk\\'e populace je m\\'irn\\v{e} vy\\v{s}\\v{s}\\'i ne\\v{z} sm\\v{e}rodatn\\'a odchyľka
v\\'y\\v{s}ky horn\\'i \\v{c}\\'asti tv\\'a\\v{r}e peru\\'ansk\\'e populace.
166
167 << echo = F, warning = F, results = "asis" >>=
168 tab <- table(data[data$pop %in% c("mal", "per"), ]$pop,
169           data[data$pop %in% c("mal", "per"), ]$upface_cat)
170 tab <- cbind(tab, suma = apply(tab, 1, sum))
171 tab <- rbind(tab, suma = apply(tab, 2, sum))
172 row.names(tab) <- c("malajska", "peruanska", "\\$\\sum$")
173
174 addtorow <- list()
175 addtorow$pos <- list(0)
176 addtorow$command <- paste0("\\\\multirow{2}*{populace} & \\multicolumn{2}{c}{kategorie} & \\\\
177           & \\$\\leq 70$ & \\$> 70$ & \\$\\sum$ \\\\")
178 print(xtable(tab, align = "l|rr|r", digits = rep(0, 4),
179           caption = paste("kontingen\\'c\\'n\\'i tabuľka absolutn\\'ich \\v{c}etnost\\'i"),
180           label = "Tab_charakteristik"),
181           caption.placement = "top", sanitize.text.function = function(x) x,
182           hline.after = c(0, 2), include.rownames = T, add.to.row = addtorow, include.colnames = F)
183 @
184
185 \cleardoublepage
186 \subsection{Grafy, obr\\'azky}
187 << echo = T, fig.height = 5, fig.width = 5, out.width = ".42\\linewidth", fig.cap = "\\label{Malajsie}
(a) Histogram (vlevo); (b) krabicov\\'y diagram (vpravo) v\\'y\\v{s}ky horn\\'i \\v{c}\\'asti
tv\\'a\\v{r}e mu\\'v{z}\\'r{u} malajsk\\'e populace" >>=
188 hist(mal.m, prob = T, col = "darkolivegreen1", main = "", ylab = 'hustota',
189       xlab = "vyska horni casti tvare (v mm)", las = 1)
190 box(bty = 'o')
191 boxplot(mal.m, col = "darkolivegreen1", main = "",
192         ylab = "vyska horni casti tvare (v mm)", las = 1)
193 mtext("malajska populace", side = 1, line = 2.1)
194 @
195

```

Obrázek 7: Řešení zadání samostatné práce 2/4

```

196 << echo = F, fig.height = 5, fig.width = 10, out.width = ".84\\linewidth", fig.cap = "\\label{Peruan}
(a) Histogram (vlevo); (b) krabicov\\y diagram (vpravo) v\\y\\v{s}ky horn\\i \\v{c}\\'asti
tv\\'a\\v{r}e mu\\v{z}\\r{u} peru\\'ansk\\'e populace" >>=
197 par(mfrow = c(1, 2))
198 hist(per.m, prob = T, col = "darkolivegreen1", main = "", ylab = 'hustota',
199 xlab = "vyska horni casti tvare (v mm)", las = 1)
200 box(bty = 'o')
201 boxplot(per.m, col = "darkolivegreen1", main = "",
202 ylab = "vyska horni casti tvare (v mm)", las = 1)
203 mtext("peruanska populace", side = 1, line = 2.1)
204 @
205 \\noindent
206 Z obr\\'azk\\r{u} \\ref{Malajsie} a \\ref{Peruan} je vid\\v{e}t, \\v{z}e rozd\\v{e}len\\'i v\\'ysky horn\\'i
\\v{c}\\'asti tv\\'a\\v{r}e mu\\v{z}\\r{u} malajsk\\'e populace je bl\\'i\\v{z}e norm\\'aln\\'imu rozd\\v{e}len\\'i
ne\\v{z} rozd\\v{e}len\\'i v\\'y\\v{s}ky horn\\'i \\v{c}\\'asti tv\\'a\\v{r}e mu\\v{z}\\r{u} peru\\'ansk\\'e
populace.
207
208 \\cleardoublepage
209 \\subsection{Animace}
210 << echo = T, message = F, fig.width = 5, fig.height = 5, out.width = ".49\\linewidth", fig.show =
"animate", fig.cap = "\\label{animace} 3D-diagram j\\'adrov\\'eho odhadu hustoty v\\'y\\v{s}ky nosu a
\\v{s}\\'i\\v{r}ky nosu mu\\v{z}\\r{u} peru\\'ansk\\'e populace" >>=
211 data19 <- read.delim('19-more-samples-correlations-skull.txt')
212 data.P <- data19[data19$pop == 'per', c('nose.H', 'nose.B')]
213 data.P <- na.omit(data.P)
214 library(MASS)
215 n <- 50
216 Z <- kde2d(data.P$nose.H, data.P$nose.B, n = n, lim = c(40, 60, 15, 30))
217 #-----
218 k <- 12
219 nrz <- nrow(Z$z) # pocet radku matice Z
220 ncz <- ncol(Z$z) # pocet sloupcu matice Z
221 color <- terrain.colors(k)
222 stredy <- (Z$z[-1, -1] + Z$z[-1, -ncz] + Z$z[-nrz, -1] + Z$z[-nrz, -ncz]) / 4
223 stredy.col <- cut(stredy, k)
224
225 theta <- seq(0, 360, by = 15)
226 for(i in 1:length(theta)) {
227 persp(Z$x, Z$y, Z$z, main = '', xlab = 'vyska nosu (v mm)', ylab = 'sirka nosu (v mm)',
228 zlab = 'f(x, y)', asp = F, col = color[stredy.col], phi = 30, theta = theta[i])
229 }
230 @
231
232
233 \\noindent
234 Na obr\\'azku \\ref{animace} je zachycen 3D-diagram j\\'adrov\\'eho odhadu hustoty v\\'y\\v{s}ky nosu a
\\v{s}\\'i\\v{r}ky nosu mu\\v{z}\\r{u} peru\\'ansk\\'e populace.
235

```

Obrázek 8: Řešení zadání samostatné práce 3/4

```

236 \cleardoublepage
237 \subsection{Mnohon'asobn'e generov'an'i tabulek}
238 << echo = F, warning = F, results = "asis" >>=
239 loop.tab <- function(pop1, pop2, lab1, lab2) {
240   tab <- table(data[data$pop %in% c(pop1, pop2), ]$pop,
241               data[data$pop %in% c(pop1, pop2), ]$upface_cat)
242   tab <- cbind(tab, suma = apply(tab, 1, sum))
243   tab <- rbind(tab, suma = apply(tab, 2, sum))
244   row.names(tab) <- c(lab1, lab2, "$\\sum$")
245
246   addtorow <- list()
247   addtorow$pos <- list(0)
248   addtorow$command <- paste0("\\multirow{2}*{populace} & \\multicolumn{2}{c}{kategorie} & \\\\
249                               & $\\leq 70$ & $> 70$ & $\\sum$ \\\\")
250   print(xtable(tab, align = "|rr|r", digits = rep(0, 4),
251               caption = paste("kontingen'v{c}n'i tabulka absolutn'ich \\v{c}etnosti -- ",
252                               lab1, "a", lab2, "populace"),
253                               label = paste("Tab_cetnosti-", pop1, '-', pop2, sep = '')),
254         caption.placement = "top", sanitize.text.function = function(x) x,
255         hline.after = c(0, 2), include.rownames = T, add.to.row = addtorow,
256         include.colnames = F, table.placement = "H")
257 }
258
259 pop1 <- c("mal", "per")
260 nam1 <- c("malajsk'a", "peru'ansk'a")
261 pop2 <- c("nem", "ban", "cin")
262 nam2 <- c("n\\v{e}meck'a", "\\v{c}insk'a", "bantusk'a")
263 for(i in 1:2) {
264   for(j in 1:3) {
265     loop.tab(pop1[i], pop2[j], nam1[i], nam2[j])
266   }
267 }
268 @
269
270 \cleardoublepage
271 \subsection{Mnohon'asobn'e generov'an'i graf{r}{u}}
272 << echo = F, fig.width = 10, fig.height = 5, out.width = ".75\\linewidth", fig.show = "asis", fig.cap
273 = "(a) Histogram (vlevo); (b) krabicov'y diagram (vpravo) v\\y\\v{s}ky horn'i \\v{c}'astitv\\'a\\v{r}je u mu\\v{z}'r{u}" >>=
274 loop.plot <- function(pop, lab) {
275   par(mfrow = c(1, 2))
276   hist(data$upface.H[data$pop == pop], prob = T, col = "darkolivegreen1",
277        main = "", xlab = "", las = 1, ylab = "hustota")
278   box(bty = "o")
279   mtext("vyska horni casti tvare (v mm)", side = 1, line = 2.2)
280   mtext(paste(lab, "populace"), side = 1, line = 3.5)
281   boxplot(data$upface.H[data$pop == pop], col = "darkolivegreen1",
282           main = "", ylab = paste("vyska horni casti tvare (v mm)", las = 1)
283           mtext(paste(lab, "populace"), side = 1, line = 2.2)
284 }
285
286 pop <- c("mal", "per", "nem", "ban", "cin")
287 nam <- c("malajska", "peruanska", "nemecka", "cinska", "bantuska")
288 for(i in 1:5) {
289   loop.plot(pop[i], nam[i])
290 }
291 @
292 \end{document}

```

Obrázek 9: Řešení zadání samostatné práce 4/4