UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA

BAKALÁŘSKÁ PRÁCE

Webová aplikace pro podporu výuky statistiky

Katedra matematické analýzy a aplikací matematiky
Vedoucí práce: **Mgr. Kamila Fačevicová Ph.D.**
Vypracoval(a): **Josef Šumpík**
Studijní program: Aplikovaná matematika
Specializace: Data Science
Forma studia: prezenční
Rok odevzdání: 2023

BIBLIOGRAFICKÁ IDENTIFIKACE

Autor: Josef Šumpík

Název práce: Webová aplikace pro podporu výuky statistiky

Typ práce: Bakalářská práce

Pracoviště: Katedra matematické analýzy a aplikací matematiky

Vedoucí práce: Mgr. Kamila Fačevicová Ph.D.

Rok obhajoby práce: 2023

Abstrakt: Bakalářská práce se zabývá interaktivní vizualizací základních statistických pojmů v aplikaci vyvinuté autorem práce v programovacím jazyku R pomocí knihovny shiny. Výsledek práce by měl být užitečným příspěvkem v základním statistickém kurzu na Přírodovědecké fakultě v Olomouci. V neposlední řadě budou v práci vymezeny matematické pojmy potřebné k dosáhnutí cíle. Práce seznamuje s vývojem aplikace a s prostředky, které k tomu byly zapotřebí, a nabízí její detailní matematický popis.

Klíčová slova: Kurz statistiky, shiny webová aplikace, vizualizace, statistická inference

Počet stran: 54

Počet příloh: 0

Jazyk: český

BIBLIOGRAPHICAL IDENTIFICATION

Author: Josef Šumpík

Title: An Interactive App as a Supporting Material for the Statistics Course

Type of thesis: Bachelor's

Department: Department of Mathematical Analysis and Applications of Mathematics

Supervisor: Mgr. Kamila Fačevicová Ph.D.

The year of presentation: 2023

Abstract: The goal of the thesis is to interactively show basic statistics concepts through an application developed in software R using the shiny library. As as result, it should serve the user to visually demonstrate the concepts particularly among participants of basic statistics course. Last but not least, all the mathematical concepts needed to accomplish the task will be layed out. The thesis zooms in the process of development of the application, discusses means needed to accomplish the goal and offers detailed insight into mathematical backround.

Key words: Statistics course, shiny web application, visualization, statistical inference

Number of pages: 54

Number of appendices: 0

Language: czech

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně pod vedením paní Mgr. Kamila Fačevicové, Ph.D. a všechny použité zdroje jsem uvedl v seznamu literatury.

V Olomouci dne .....

Obsah

Seznam obrázků

Poděkování

Rád bych poděkoval své vedoucí práce Mgr. Kamile Façevicové Ph.D. za vedení k nápadům a trpělivost při konzultacích a také své rodině a své milé za podporu při psaní práce.

Úvod

Člověk se dobře učí na příkladech a poznává věci, které dokáže vnímat svými smysly. Na jeden konkrétní smysl - *zrak* - se zaměříme v této práci a podpoříme vyúku statistiky působením na zrak prostřednictvím vizualizací. Toho chceme dosáhnout skrze webovou aplikaci. Ta je vytvořena autorem ve vývojovém prostředí *RStudio* a má za cíl interaktivně přiblížit teoretické statistické pojmy. Aplikaci může spustit jakýkoli zájemce o statistiku, zejména pak ti, kteří navštěvují základní statistický kurz na Přírodovědecké fakultě v Olomouci. Pro velmi široký rozsah probíraného učiva vybíráme právě ty pojmy, které úzce souvisí se statistickou inferencí a s jejími odnožemi - testováním hypotéz a odhadováním parametrů. Ostatně to jsou také jedny z hlavních motivů statistické části skript, které jsou základní literaturou zmíněného kurzu (viz [2], kapitola 5 - 8). Ovšem kromě učiva, o kterém se pojednává v těchto skriptech, se budeme zabývat i koncepty, které v nich jsou zmíněné jen nepřímo nebo vůbec, protože napomohou k uchopení statistické inference jako celku. Dovolíme si to, protože z hlediska teoretického je vše připraveno pro to, aby byly uchopeny. Práce je rozdělena do 3 kapitol.

V první kapitole se seznámíme se základními koncepty programovacího jazyku *R*; zvláštní pozornost věnujeme balíčku *Shiny*, který je hlavní prostředek pro vývoj aplikace.

Ve druhé kapitole vymezíme matematické pojmy, které jsou potřebné k uchopení konkrétních vizualizací. Budeme se věnovat popisu nezbytných základů teorie pravděpodobnosti a matematické statistiky.

Stěžejní třetí kapitola nám bude střídavě formou uživatelského manuálu a matematického popisu dopodrobna komentovat úvodní záložku *vítejte!* a 5 hlavních částí shiny aplikace. V nich se teoreticky zabýváme příklady demonstrující pojmy *p-hodnota*, *hladina testu* α, *nestrannost a konzistence odhadu parametru* a *síla testu*. V poslední záložce *hod mincí (velikost testu)* provádíme interaktivní praktickou ilustraci pojednané teorie.

Aplikaci lze spustit prostřednictvím odkazu `https://sumpikpepa.shinyapps.io/Statistika_v_kostce/`, kde ji lze omezeně dlouho používat. Návod jak sdílet aplikaci lze najít zde [1].

1 Programovací jazyk *R*

R je programovací jazyk sloužící ke statistickým výpočtům a vykreslování grafů. Ke zkoumání tohoto jazyka a příslušných knihoven nás zavede zdroj [3]. Řada knihoven otevírá mnoho dalších možností pro práci s grafy, statistickými funkcemi a dalšími matematickými prostředky. V následujících sekcích se v jednoduchosti zaměříme na některé aspekty jazyka a také na důležité knihovny, které byly při práci použity. Práce není informatická, pobavíme se tedy zvláště o praktickém použití některých užitečných funkcí, které knihovny nabízí.

1.1 Vykreslování grafů

Velmi často je výsledkem práce s *R* vizualizace dat. Vykreslovat data spolu s odhady jejich rozdělení můžeme například pomocí histogramů, krabicových grafů, violin plotů, boxplotů nebo bodových diagramů. V *R* najdeme více způsobů, jak s pomocí i bez pomoci knihoven data vizualizovat. Pro účely aplikace použijeme kvůli jejímu teoretickému zaměření především bodové grafy a histogramy.

1.1.1 GGplot2

Mezi nejpopulárnější volby grafické vizualizace patří `ggplot2` (*Grammar of graphics plot 2*). Knihovna není zabudována v *R* při instalaci, ale lze v krátkém čase nainstalovat. Umožňuje podrobně modifikovat vzhled celého grafu, jeho tvary, popisky os i konkrétních bodů a automaticky přidávat statistické ukazatele (např. vykreslení regresního pásu kolem přímky). V neposlední řadě umožňuje měnit souřadnicový systém a volit náměty vzhledu grafu (tzv. themes). Složité zacházení s knihovnou ulehčuje tzv. Cheat sheet [4], který uceleně graficky shrnuje hlavní funkce knihovny.

V mém případě byl `ggplot2` první volbou pro splnění cíle mé práce nicméně pro větší nároky na výkonnost, které začaly být zjevné při delším používání knihovny, a protože pro mé potřeby nemá knihovna přednosti před základní knihovnou `graphics`, rozhodl jsem se knihovnu `ggplot2` nepoužít.

1.1.2 Graphics

Tato knihovna je zabudovaná v *R* při instalaci (je tzv. *built-in*), proto lze použít její funkce bez jakékoli další instalace balíčků. Ústřední pro aplikaci je funkce `plot()`, která umožňuje vykreslit dvourozměrný graf zadáním x-ových a y-ových souřadnic. Volbou parametrů lze specifikovat typ grafu

(např. zda-li dostaneme křivku nebo pouze body), názvy os, název celého grafu či většího počtu grafů. V případě vizualizace bodů dále tvar, velikost nebo barvu; v případě křivky pak zda-li je zobrazena plně, tečkovaně či čárkovaně, její tloušťku a barvu. Nesmíme opomenout také specifikace zobrazení os, kdy lze rozhodnout, které části os x a y budou zobrazeny nebo zda-li vůbec budou zobrazeny, a kde budou ležet označení osových hodnot. Osy lze upravovat také zpětně funkcí `axis()`.

Pro vykreslení více přímek do jednoho grafu nabízí balíček funkci `lines()`. Pro 1-dimenzionální zobrazení hodnot zadáme `rug()`. Funkcí `polygon()` znázorníme plochu v námi určené oblasti, což je vhodné pro vykreslení plochy mezi grafem a osou x. To lze uplatnit například při vizualizaci hodnoty distribuční funkce spojitého rozdělení pravděpodobnosti pomocí hustoty.

Abychom rozuměli značení v grafu, nelze opomenout funkci `legend()`. S její pomocí lze k bodům, křivkám a barvám přiřadit název či vysvětlení. Stejně jako funkci `axis()` nemohu `legend()` použít bez předešlého zavolání funkce `plot()`; v případě potřeby lze ovšem vykreslit prázdný graf a následným zavoláním `legend()` mohu získat samostatnou legendu. Tento trik je užitečný, pokud je legenda obsáhlá a její velikost by výrazně narušovala vzhled celého grafu.

Nakonec bych se chtěl zmínit o funkci `par()`, která má mnoho využití. V kontextu shiny aplikace bych zdůraznil zejména možnost zmenšit prázdný prostor mezi grafem a okolním uživatelským prostředím. Toho lze dosáhnou vhodnou volbou hodnot vektorového parametru *mar*.

1.1.3 Plotrix

Funkce `abline()` z balíčku `graphics` vykreslí lineární funkci přes celý graf. Balíček `plotrix` zmiňuji kvůli funkci `ablineclip()`, pomocí níž lze zobrazit lineární funkci na omezeném definičním oboru. Toho mohu využít např. při zvýrazňování kritického oboru hustoty rozdělení testové statistiky. Pro použití je třeba knihovnu nainstalovat.

1.2 Stats

Knihovna poskytuje funkce pro statistické výpočty a generování náhodných čísel. Čísla mohou být generována z různých pravděpodobnostních rozdělení. Balíček tedy umožňuje simulovat náhodný výběr, provádět testy, vyčíslit kvantily z různých rozdělení nebo vytvořit model lineární regrese. Využíváme ji také při konstrukci jádrové hustoty. Knihovna je *built-in*.

1.3 Shiny a příbuzné knihovny

Ústřední knihovnou práce je *Shiny* (verze 1.7.3). Programátor a analytik Christ Beeley ve své knize pojednávající o *Shiny* říká:

"*Shiny* je perfektní společník *R* umožňující rychlé a jednoduché předávání analýz a grafických výstupů z *R*, se kterými mohou uživatelé interagovat, a do kterých mohou zadávat dotazy přes webové prostředí."
(Volně přeloženo autorem z [5], str. 21)

S jeho výrokem nemohu než souhlasit. Naprogramovat jednoduché interaktivní rozhraní nezabere mnoho času, zvláště pokud má vývojář dostatek zkušeností. Knihovna nabízí novou dimenzi přiblížení datových analýz a statistických výpočtů oproti pouhým *nehybným* grafům a obrázkům.

Každá *Shiny* aplikace má dvě základní struktury (komponenty). První je *ui* nebo-li *Uživatelské rozhraní* (anglicky *User interface*), ve kterém definujeme vzhled aplikace a to, jakým způsobem budeme s aplikací komunikovat. Ve druhé struktuře *server* definujeme fungování aplikace a provádění výpočtů. Obě struktury mohou být buď jako dva objekty v jednom *R* skriptu pojmenovaném *app.R* anebo dvě funkce ve dvou různých skriptech s názvy *ui.R* a *server.R*. Na pojmenování skriptu nezáleží, ale použití ustálených názvů je vhodné pro přehlednost. Kvůli obsáhlosti aplikace a pro větší orientaci v kódu jsem aplikaci vyvíjel ve dvou různých skriptech.

V následujících kapitolách pojednáme o způsobech, kterými můžeme interaktivní vizualizace dosáhnout. Nejprve pokryjeme způsoby vytváření uživatelského prostředí, dále zmíníme některé aspekty *server* komponenty jako je například princip reaktivního programování. V neposlední řadě se neopomeneme zaměřit na některé knihovny či celé programovací jazyky otevírající nové způsoby práce s *Shiny*. Důraz bude kladen zejména na funkcionality, které byly použity při vytváření aplikace.

Během vývoje aplikace mi bylo nápomocné užití taháku, který zájemce nalezne ve zdroji číslo [6]. Pro získání povědomí o základních principech knihovny doporučuji online knihu Hadley Wickhama [7].

Knihovnu i všechny příbuzné knihovny je třeba nainstalovat pomocí příkazu `install.packages`, který je v *R* automaticky zabudován.

1.3.1 Uživatelské prostředí Shiny

Účelem uživatelského prostředí (dále jen *ui*) je budování vzhledu aplikace, vytváření tzv. *vstupů* (anglicky *input*), a následné zpracování a zobrazování tzv. *výstupů* (anglicky *outputs*).

Vstupy

Vstupy určují nejen to k jakým procesům dojde ve skriptu *server*, ale i jaké hodnoty budou v těchto procesech figurovat. V *server* skriptu se na ně odkazujeme pomocí námi zvolených identifikací, což jsou označení, které mohou být tvořeny jak čísly, tak i písmeny. Výsledný kód následně putuje zpět do *ui* jako výstup. Takto je zajištěna komunikace mezi námi a aplikací. Způsobů jakými můžeme navolit hodnotu vstupů je několik.

`NumericInput()` nám umožnuje zadat číselnou hodnotu. Pomocí šipek lze hodnotu zvyšovat nebo snižovat. Přípustné hodnoty lze omezit maximem a minimem. Nevýhodou je, že můžeme pomocí klávesnice zadat i číslo mimo povolený interval. To lze vyřešit pomocí knihovny *ShinyFeedback* a funkce `req()` případně `validate()`.

Pro mé účely byl nejvýhodnějším číselným vstupem `SliderInput()`. Umožnuje posuvníkem zvolit hodnotu na omezeném intervalu, ze kterého nelze žádnou kličkou vystoupit. Argumenty vstupu lze navíc upravit tak, že si nevybereme pouze jednu hodnotu, ale celý interval. Přídáním argumentu `Animate = TRUE` lze `SliderInput()` využít pro animaci. Zašrtnutím tlačítka *play* se automaticky mění hodnota vstupu o jednu jednotku, kterou lze navolit pomocí parametru `step`. Modifikovat lze i frekvenci těchto změn. Použití bylo uplatněno např. při volbě hodnot parametrů rozdělení za platnosti nulové hypotézy anebo při volbě velikosti náhodného výběru. Animace pak dostala své místo při simulaci konzistence odhadu s rostoucím rozsahem výběru.

`TextInput()` je vstup, do kterého píšeme text. V aplikaci tento vstup není. V článku [8] z webu *rstudio.com* je varování, že pokud dojde k vložení textového vstupu do funkce `HTML()` (příkaz označí text jako součást jazyku HTML), může zlomyslný uživatel zneužít nebo poničit aplikaci, proto je třeba dbát zvýšené opatrnosti.

Ve vstupu `radioButtons()` si můžeme vybrat *jednu* z předem definovaných možností. V aplikaci vstup využívám např. při volbě druhu alternativní hypotézy; tzn. výběr jedné ze tří možností.

Velmi podobný je `selectInput()` nebo `selectizeInput()`. Při prvním vstupu si stejně jako v případě `radioButtons()` vybíráme pouze *jednu* možnost, ale možnosti volby se nám zobrazí až po kliknutí na panel vstupu. Výhodou je přehlednost a skladnost tlačítka v *ui*. Vhodné využití nalezneme v kombinaci s `conditionalPanel()` (viz následující kapitola o budování uživatelského prostředí). V druhém případě můžeme vybrat *více* možností současně.

`CheckBoxInput()` a `CheckBoxGroupInput()` jsou vstupy, které umožňují vybrat libovolný počet z jedné nebo více možností. Hodnoty, ze kterých vy-

bíráme, jsou předem dané; jsou to `TRUE` v případě zakšrtnutého políčka nebo `FALSE`, pokud políčko zašktrnuté není.

Stěžejní je tlačítko, nebo-li `actionButton()`. Po stisknutí se zvyší hodnota vstupu o 1, přičemž před stisknutím je rovna 0. Počáteční hodnota zaručuje, že při inicializaci aplikace nedojde k provedení kódu v reaktivním prostředí, které má na prvním řádku `req(input$actionButton)`. Vyplníme-li vhodně argument `class`, bude vstup zobrazen v námi zvoleném stylu (např. zeleně pro tlačítko úspěchu). Tlačítko je důležité při práci s reaktivitou. S jeho pomocí lze zamezit chaotické změně výstupů, které jsou závislé na několika vstupech. Více se na problém podíváme v kapitole o reaktivitě.

Výstupy

Výstupy chápame v *Shiny* jako kód vytvářející grafy, texty, tabulky, obrázky, HTML nebo celé uživatelské prostředí zpracované v *server* skriptu. V *ui* na ně odkazujeme jejich identifikací v příslušné funkci (např. `textOutput()` nebo `uiOutput()`). Pro vzhled aplikace je důležité správně navolit hodnotu boolean parametru `inline`, kterým rozhodneme, zda bude výstup vložen na nový řádek či nikoliv. Každý výstup můžeme použít pouze jednou! Nelze vykreslit dva stejné grafy anebo dvě textové pole pomocí stejné identifikace.

Do `plotOutput()` můžeme vložit argumentů více. Rozhodneme-li se, můžeme přidat interaktivitu grafu, např. přibližování, přidávání bodů, mazání bodů, a to pomocí obdélníkové selekce části grafu, kliknutí, dvojitého kliknutí nebo jen pouhého přejíždění kurzoru myši. Tyto možnosti mě velmi nadchly, nicméně výsledné grafy byly příliš překombinované, proto jsem od nich po kratším zkoumání upustil. Podrobnějšího poučení načerpáme např. zde: [9].

Budování uživatelského prostředí

Vytvořené vstupy je důležité do *ui* přehledně umístit a popsat, abychom dokázali rozlišit, co aplikace dokáže, a kde najdeme co potřebujeme. *Shiny* nabízí mnoho způsobů, jak *ui* zkonstruovat. Pro komplexnější aplikace vyzdvihneme zvláště knihovnu *shinydashboard*, která přináší nesčetné možnosti vytváření struktury stránek shiny aplikace pomocí tzv. *dashboardů*. V aplikaci tato knihovna použita není, protože si myslím, že funkce *Shiny* knihovny otevírají dostatek potenciálu při vytváření přehledných struktur.

Pro vytvoření základní stránky, do které se budou následně vkládat další struktury se vstupy a grafy, se nejčastěji používá funkce `fluidPage()`. Škáluje svůj obsah, aby vyplnil veškerou šířku prohlížeče. Alternativa `fixedPage()` má pevně danou šířku v pixelech a neumožňuje použití struktury `sideBarLayout()`. Rozložením `navbarPage()` vytvoříme stránku s navigačním panelem v horní části stránky. Zadáním argumentu `fluid` určíme, zda se bude tato stránka chovat jako prostředí `fluidPage()` nebo

`fixedPage()`. Pro časté používání rozložení `sideBarLayout()` jsem si zvolil funkci `fluidPage()`. Důležitým argumentem funkce je `theme`, kde si můžeme pomocí balíčku *bslib* vybrat z široké škály vhodných témat aplikace a grafů (barvu tlačítek, vstupů, pozadí a jiné) nebo si pomocí funkce `bs_theme_preview` vytvořit své vlastní téma, což si ze své zkušenosti troufám tvrdit, že není dobrý nápad. Najít pěkné vyvážení barev je obtížné. Témata jsou vytvořeny pomocí programátorské sady nástrojů Bootstrap (neplést se statistickou metodou bootstrap). Podrobný popis vytváření a používání stylů lze nalézt zde: [10].

Podle volby prostředí základní stránky se dále používají funkce `fluidRow()` nebo `fixedRow()`, které umožňují vkládat text, vstupy nebo výstupy do jednoho řádku. Funkcí `column()` uvnitř `fluidRow()` lze řádek rozdělit na sloupce různé šířky; součet šířek nesmí přesáhnout hodnotu šířky řádku, která je vždy 12.

`SideBarLayout()` je struktura vhodná pro vykreslování grafů. Rozděluje prostředí na dvě části, z nichž první nazývaná `sidebarPanel()` (boční panel), je vhodná pro vstupy a její obsah je uzavřen v šedém obdélníku se zaoblenými rohy, a druhá `mainPanel()` (hlavní panel) se používá pro výstupy. Šířka obou části je volitelná uživatelem s defaultním nastavením poměru velikostí 1:2.

Použijeme-li funkci `wellPanel()`, bude podobně jako v případě struktury `sidebarPanel()` její obsah zaobalen v šedém obdélníku. Můžeme tak rozlišit důležité od nedůležitého nebo vstupy od výstupů.

Navolíme-li u výše zmíněných funkcí (kromě `navbarPage()`) hodnotu parametru `align = "center"`, získáme vycentrovaný nejen text, ale celý obsah rozložení.

Funkcí `tabsetPanel()` vytvoříme lištu záložek, ve kterých mohou být další struktury. V rámci rozložení můžeme přidávat funkce `tabPanel()`, která každá přidá jednu záložku se zvoleným názvem. Struktura je ideální pro přechod mezi různými tématy, proto je ústředním rozložením naší aplikace.

Struktura `conditionalPanel()` je speciální tím, že podobně jako kód ve skriptu *server* pracuje s hodnotou vstupu v *ui*. Zadáním argumentu `condition` si můžeme zvolit, za jakých podmínek bude obsah panelu zobrazen. Odvážíme-li se kombinovat prostředí s funkcí `selectInput()`, dokážeme vytvořit uživatelské prostředí, které je přívětivé, usporné a přehledné. Např. můžeme zvolením tohoto vstupu měnit vzhled celé stránky. Vytvoříme tak tzv. *Dynamické uživatelské prostředí* nebo-li *dynamické ui*. V aplikaci využíváme `conditionalPanel()` při volbě odhadovaného parametru rozdělení zkoumané veličiny nebo při výběru vhodné odhadové statistiky.

Ukazatele průběhu

Tzv. *ukazatele průběhu* (anglicky progress bar), sloužící k informování uživatele o načítání, aplikace můžeme do *Shiny* zabudovat mnoha způsoby. V aplikaci je použit progress bar knihovny *waiter* pro inicializační načítání aplikace vytvořený funkcí `waiterPreloader()`. Informace o možném uplatnění funkce spolu s využitím celého balíčku nalezneme v rozsáhle dokumentaci *Comprehensive R Archive Network*. Odkaz je v literatuře [11]. Informace o funkci `waiterPreloader()` pak na str. 25, 27. Průběh vykreslování grafů ukazuje progress bar funkce `withSpinner()` balíčku *shinycssloaders*. Oba balíčky disponují svými výhodami i nevýhodami; ale také širokou škálou možností zobrazení ukazatele.

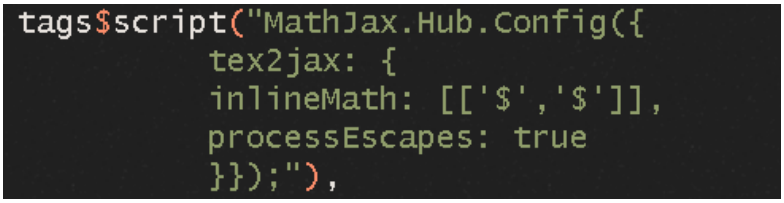
Informovat uživatele o probíhajících výpočtech např. `for()` cyklu můžeme pomocí zabudovaného progress baru `withProgress()`. Při každém dokončeném cyklu zvýšíme hodnotu ukazatele zavoláním `incProgress()`, který jako argument přijme číslo mezi 0 a 1 vyjadřující procentuální navýšení ukazatele.

Využití HTML při práci s Shiny

Při zkoumání knihovny *Shiny* dříve nebo později zjistíme, že je provázána s některými dalšími jazyky. Není třeba panikařit, protože to neznamená, že musíme nutně s těmito jazyky umět zacházet. Pro úpravu uživatelského prostředí je užitečný zejména značkovací jazyk *HTML* (Hyper Text Markup Language), protože skript *ui* je zamaskovaný HTML dokument. HTML kód můžeme vytvářet buď pomocí funkcí zabudovaných v *Shiny*, tzv. *helper* funkcí, anebo psát čistě řetězec v HTML, který následně vložíme do funkce `HTML()`. Pro uvedení do obrazu budu znovu citovat Christa Beleeyho:

"Shiny je velmi vstřícný a zjistíte, že celkem bez problému zpracuje Shiny kód smíchaný s HTML kódem, který vytvoříte pomocí Shiny *helper* funkcí a nezpracovaného HTML."
(Volně přeloženo autorem z [5], str. 129)

Pokud chceme mít v aplikaci kontrolu nad vzhledem uživatelského prostředí, je užitečné se s HTML více seznámit. Všechny pochybnosti o užitečnosti práce s jazykem snad vyvrátí jednoduchý kód, kde integrujeme LaTeX s naší aplikací a tak si umožníme psát matematické definice a výpočty pomocí LaTeX syntaxe. O této integraci je dopodrobna pojednáno v dokumentaci [12] v kapitole *TeX and LaTeX math delimiters*. Abychom toho dosáhli, je třeba do *server* funkce anebo do základní stránky (např. `fluidPage()`) v *ui* zadat kód, který je uveden na obrázku 1.



Obrázek 1: Kód integrace LaTeXu a *Shiny*

V hranatých závorkách definujeme, jaké oddělovače aktivují LaTeXový matematický inline mód; v případě naší aplikace se tak stane ohraničením textu pomocí symbolů *$*. Argument `processEscapes: true` umožňuje psát *$* bez vstupu do matematického módu pomocí lomítka. Pro použití LaTeXové syntaxe je třeba kdekoli v rámci základní stránky anebo *server* funkce přidat příkaz `withMathJax()` zabudovaný v *Shiny* knihovně.

Použitím `tag$` získáváme přístup k 110 tagům (značkám) HTML otevírající mnoho možností, o kterých se můžeme dočíst např. v [13]. V rámci helper funkcí nalezneme ty nejpoužívanější tagy, mezi které patří *h1,...,h6* umožňující hierarchické formátování textu. Značku `p()` najdeme v aplikaci pro běžný text. Helper funkcí `a()` vytvoříme použitelné URL odkazy a zadaním `br()` se přesuneme na nový řádek. Použijeme-li funkci `span()`, můžeme si zvolit vlastní font, velikost, barvu, zarovnání, odsazení a mnoho dalšího. V aplikaci má tato značka velmi široké uplatnění při úpravě textu jak čistě v *ui*, tak při zpracovávání výstupů ze skriptu *server*. Chceme-li, můžeme vytvořit list značek funkcí `tagList()`.

Ač pro renderování obrázků nalezneme v knihovně *Shiny* funkci `renderImage()`, použil jsem místo toho značku `img()` v kombinaci s `renderUI()`. Syntaxe i proces zobrazení je jednodušší a vše pracuje bez problémů. V argumentech helper funkce lze zadat požadovanou šířku a výšku obrázku.

1.3.2 Výpočetní prostředí Shiny

V této kapitole se zaměříme na koncepty aplikace související především se skriptem *server*. Podíváme se na zpracování vstupů, generování výstupů a jakou roli v tom všem hraje reaktivní programování. V neposlední řadě se dotkneme možností práce s dalšími knihovnami.

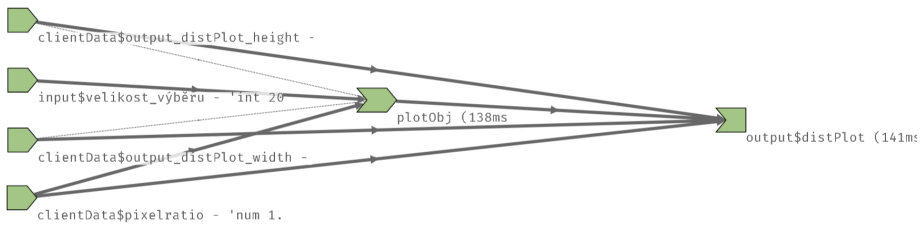
Reaktivní programování

Nejdříve se podíváme na specialitu, která knihovnu odlišuje od většiny ostatních programovacích jazyků; je to tzv. *reaktivní programování*. Hlavním principem je provázanost *reaktivních* objektů, které na sebe automaticky reagují. V článku o reaktivitě z již zmiňovaného webu shiny.rstudio.com [14]

rozlišujeme 3 základní objekty reaktivního programování: *Reaktivní zdroj*, *Reaktivní vodič* a *Reaktivní konec* (volně přeloženo autorem).

Za reaktivní zdroj považujeme zpravidla uživatelský vstup, jehož hodnotu získáme pomocí kódu `input$identifikace`. Není závislý na žádném objektu, ale jiné objekty jsou na něm závislé. Oproti tomu reaktivní konec je zpravidla závislý na nějakém objektu. Nejčastěji ho můžeme ztotožnit s výstupem. Značíme `output$identifikace` a zpracováváme v něm hodnotu (tabulku, graf, text), kterou pomocí identifikace získáme ve skriptu *ui*. Jako reaktivní vodič chápeme reaktivní objekt, který umožňuje provést mezikroky mezi reaktivním zdrojem a reaktivním koncem. Je závislý na objektu a zároveň jsou na něm závislé jiné objekty.

Fungování reaktivních proměnných můžeme vizualizovat pomocí balíčku *reactlog*, zadáme-li `reactlog_enable()` do funkce *server* skriptu. Po spuštěním aplikace dostaneme reaktivní graf stisknutím *CTRL + F3*, který může vypadat např. jako na obrázku 2:



Obrázek 2: Jednoduchý reaktivní graf

Objekty v levé části obrázku jsou reaktivní zdroje (zde parametry grafu), objekt uprostřed je reaktivním vodičem (zde proměnná umožňující vykreslit graf) a v pravé části grafu vidíme reaktivní konec, který nese kód výstupu (výsledný graf). Důležité je si povšimnout šipek, které znázorňují závislost mezi reaktivními objekty. Na obrázku nalezneme i délku vykonávání kódu v milisekundách. Vykreslení reaktivního grafu v naší aplikaci je pro množství proměnných zbytečné a chaotické.

Vstupy v *Shiny* rozpoznáváme jako *reaktivní proměnné*. Reaktivní proměnné si můžeme definovat i vlastní; zadáním `x~=~reactiveValues()` vytvoříme reaktivní proměnnou, která může nabývat více reaktivních hodnot. Oproti tomu `x~=~reactiveVal()` nabývá pouze jedné hodnoty, kterou získáme zavoláním funkce `x()`. Volat reaktivní proměnné můžeme jen v *reaktivním prostředí*, které je zaobaleno do složených závorek a automaticky vytvářeno čímkoli, co umožňuje pracovat s reaktivními proměnnými.

Práci s reaktivitou nám usnadňují *observers*; volněji přeloženo *pozorovatelé*. Sami žádnou hodnotu nepřiřazují, ale vytvářejí reaktivní prostředí,

ve kterém se modifikují reaktivní proměnné nebo provádějí jiné činnosti aplikace. Funkce `observe()` provede kód zadaný v rámci reaktivního prostředí při změně libovolné reaktivní proměnné, kterou prostředí obsahuje. V aplikaci ji využíváme, když při incializaci načítáme grafy pomocí `click()` (více viz kapitolu Java Script a CSS) anebo v uživatelské zpětné vazbě. Stěžejní je funkce `observeEvent()`, která provede kód uvnitř *právě tehdy, když* se změní hodnoty reaktivní proměnné (vstupu) v prvním argumentu funkce. Díky tomu můžeme mít reaktivitu více pod kontrolou a nedojde k neřízeným změnám výstupů nebo reaktivních proměnných při změně vstupů. Uplatnění v aplikaci nalézáme při nastavování tlačítek. Pozorovatelé mají své protějšky ve funkcích `reactive()` a `eventReactive()`, které použijeme, když chceme výslednou hodnotu (graf, tabulku) vložit do reaktivní proměnné buď při změně jakékoli proměnné nebo jen při změně některých námi zvolených proměnných.

Použitím `isolate()` odstraníme reaktivitu výstupu kódu zaobaleného uvnitř reaktivního prostředí. Funkci v kódu aplikace nenajdeme, protože je pro naše potřeby zastupitelná pozorovateli.

Princip reaktivity má velmi mnoho aspektů a může působit zmatečně. Jednoduché pochopení bez přílišného zavalení pojmy lze nabýt v českém článku z webu Karlovy univerzity zde: [15].

Generování výstupů

Jak už bylo řečeno v kapitole o uživatelském prostředí, vstupy putují do *server* skriptu, kde se podílejí na vytváření výstupů buď přímo anebo prostřednictvím reaktivních proměnných. Výstupy můžeme generovat rozdílnými renderovacími (zobrazovacími) funkcemi, které podobně jako pozorovatelé provádějí kód v reaktivním prostředí. Na výstupy se v *ui* odkazujeme odpovídajícími funkcemi (např. výstup vytvořený pomocí `RenderTable()` získáme zadáním `tableOutput()`). Zobrazovací funkce rozlišujeme podle jejich účelu a činnosti.

`RenderText()` nám umožňuje zobrazit v aplikaci text v podobě řetězce. Pomocí funkce `paste()` případně `paste0()` můžeme k řetězci připojit jednu nebo více číselných hodnot. Bohužel nelze tímto výstupem renderovat text vygenerovaný pomocí `withMathJax()`, který umožnuje generování matematických symbolů syntaxí LaTeXu (více viz Využití HTML při práci s Shiny).

Do `renderPlot()` vložíme graf, který chceme vygenerovat. Argumenty `width` a `height` můžeme navolit šířku či výšku grafu v pixelech, což je nutné v případě, že ve výstupu `plotOutput()` máme zadaný argument `inline~=~TRUE`. Ten při jejich nezadání zapříčiní zkrat celého *Rstudia*. V aplikaci používáme `renderPlot()` při vykreslování grafů a legend.

Problém při generování LaTeXu vyřešíme pomocí `renderUI()`, který vy-

kreslí matematické vzorce. Chceme-li, zobrazíme celá uživatelská prostředí nebo obrázky. Funkci v aplikaci používáme při vykreslování všech obrázků a proměnlivého LaTeXu.

Dynamické uživatelské prostředí

V *server* skriptu je nám dovoleno upravovat uživatelské prostředí nejen prostřednictvím zmíněného `renderUI()` a `conditionalPanel()`, ale i speciálními funkcemi `updateInput()`. Nimi změníme nastavení hodnot parametrů libovolného vstupu. Uplatnění nalezneme zejména při filtrování dat, kdy se mohou měnit např. maxima, minima či současné hodnoty `sliderInput()`, možnosti `selectInput()` případně počet křížků `checkBoxGroupInput()` v závislosti na zvolených datech. Před použitím `updateInput()` je vhodné zavolat `freezeReactiveValue()`, jenž zamezí přístup uživatele k zmraženému vstupu. Vyhneme se tak nechtěnému výběru hodnoty vstupu v nevhodnou dobu.

Jistou dynamiku můžeme uživatelskému prostředí přidat s knihovnou *ShinyFeedback*, se kterou přidáme ke vstupům barevná upozornění, varování nebo oznámení úspěchu. Jak už bylo zmíněno v odstavcích o vstupech, zpětnou vazbu zobrazíme uživateli, pokud například zadá nechtěnou hodnotu v `numericInput()`. Informaci o tom, zda-li je hodnota námi chtěná, zjistíme pomocí výrazu s hodnotou `TRUE` nebo `FALSE` (tzv. booleanu) zaobaleného funkci `req()` nebo `validate()`.

1.3.3 Java Script a CSS v Shiny

Shiny můžeme provázat také s *JavaScriptem* (zkráceně JS), který přidá aplikaci interaktivitu. Podobně jako v případě HTML se nemusíme trápit psaním kódu v tomto jazyku, protože knihovna *shinyjs* importuje potřebné funkce. Jako nejužitečnější vyzdvihneme např. `click()`, která v *serveru* stiskne námi zvolené tlačítko. V aplikaci tak zavoláním funkce před inicializací uživatelského prostředí automaticky načteme grafy a nemusíme se bát, že by bylo v aplikaci neužité prázdné místo, které čeká na graf. Použitím `enable(),~disable()` můžeme dle libosti aktivovat nebo deaktivovat námi zvolený vstup. Funkce jsou užitečné, pokud chceme zamezit uživateli volit hodnoty vstupů, ale zároveň nechceme, aby vstupy zmizely.

Potřebujeme-li si více pohrát s formátováním a mnohými dalšími dekorativními aspekty aplikace, je vhodné užití jazyku *Kaskádových stylů* (zkráceně CSS; *Cascade style sheets*), který můžeme stejně jako JS psát dvěma způsoby: přímo do aplikace anebo vytvořit skript, který si nahrajeme do *R*.

1.4 (Ne)užitečné typy pro práci v R

Na závěr pojednání o R bych chtěl zhodnotit práci v programovacím jazyku *R* verze 4.2.2 pomocí vývojového prostředí *Rstudio* s nejaktuálnější verzí k 01.03.2023. Věřím, že i z pohledu matematika, tedy neodborníka na informační technologie, mohu pomoct užitečnými typy některým uživatelům.

V první řadě vyzvedneme vhodnost kolekce funkcí `apply()`. Funkce jsou vhodné jako náhrada za `for` cyklus, protože je jimi provedený výpočet rychlejší. Konkrétně funkce `lapply()` vezme jako argument objekt `list` a na všechny prvky v něm obsažené aplikuje funkci v dalším argumentu. Výsledkem je `list`, který můžeme pomocí funkce `unlist()` transfromovat na vektor.

Pro větší přehlednost kódu lze použít funkci `switch()`, která na jednom řádku přehledně umožní dosáhnout stejného výsledku jako několik `if,~else` klauzulí. Funkci ale nemůžeme s klauzulemi zaměnit pokaždé.

Nevíme-li, co se skrývá za některou funkcí, můžeme si o ní přečíst podrobný popis spolu s příklady užití, zadáme-li před funkci otazník, např. `?apply()`. V okně *Pomoc* (anglicky *Help*) v pravém dolním rohu nalezneme podrobný popis spolu s ilustračními příklady.

Pokud chceme okomentovat více řádků kódu současně (například protože potřebujeme vyzkoušet, jak se bude shiny aplikace chovat bez některého grafu či výpočtu), můžeme vybrat příslušné řádky a následně stisknout *CTRL + SHIFT + C*. Pokud je řádků 50, významně si ulehčíme práci. Stejným způsobem můžeme řádky kódu vrátit do původního stavu. Pokud potřebujeme ušetřit čas spouštením aplikace posouváním myši k tlačítku *Spustit aplikaci* (anglicky *Run App*), jednoduše na klávesnici stiskneme *CTRL + SHIFT + ENTER*.

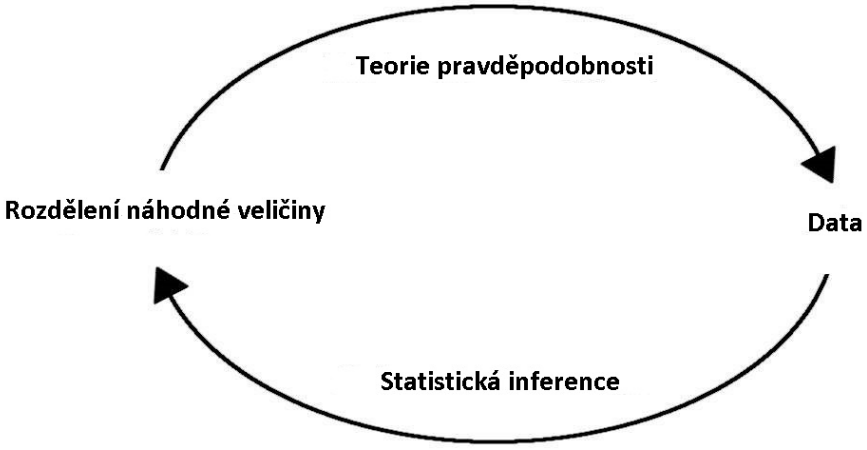
O těchto tricích a mnohých dalších se lze dočíst zde: [16].

2 Matematický aparát aplikace

Kromě znalosti programování potřebujeme k vytvoření, popsání a uchopení správně fungující aplikace i vhodné náčiní ze sady matematické statistiky. Neobejdeme se tedy bez řádného vymezení pojmů, které už byly mnohokrát popsány a rozmyšleny v četných statistických knihách. Jako důležitý zdroj vezmeme již zmíněná skripta [2], protože se chceme vyhnout nechtěným zmatkům ve značení (například nebudeme značit obecnou statistiku písmenem S, jak je to v [18], ale T). Další důležitým literární dílem je obsáhlá kniha jak názvem tak obsahem *Probability & statistics for engineers & scientists* [17], která je přínosná tím, jak se na aspekty statistiky dívá několikrát z mnoha různých úhlů a vše se snaží důkladně přiblížit. Velký pedagogický potenciál dokazuje počet příkladů na procvičení správného myšlení i praktického počítání (je jich několik set). V neposlední řadě budeme čerpat z knihy [18], *Základy matematické statistiky*, která objemností a přesností matematických postupů, definic a vět otevírá cestu k hlubokému porozumění obecné teorie.

2.1 Motivace k statistické inferenci

Pustíme-li se do teoretického zkoumání, je užitečné se nejdříve podívat na to, za jakým účelem je tak třeba učinit (zvláště chceme-li namotivovat studenty). V praxi se setkáváme s mnoha různými daty, která můžeme popsat prostředky *popisné statistiky*, mezi které patří výpočet *momentových* charakteristik (např. aritmetický průměr, rozptyl) nebo *robustnějších kvantilových* charakteristik. Oba typy disponují různými výhodami, nevýhodami, vlastnostmi a interpretací. Závěry popisné statistiky činíme pouze pro konkrétní data. Naopak při řešení statistických problémů jako je např. otázka, zda-li je průměrná výška člověka 180 cm (hledáme neznámou hodnotu parametru rozdělení náhodné veličiny), se chystáme učinit *obecné* rozhodnutí (indukci) na základě dat. Ta ale reprezentují jen určitou část reality. Proto využíváme hypotetických rozdělení testové nebo odhadové statistiky, která určují jak by se data chovala, pokud by splňovala požadavky z hypotézy (např. kdyby průměrná výška člověka byla opravdu 180 cm). Z těchto hypotetických rozdělení vypočítáme pravděpodobnosti, na základě kterých vyslovíme závěr o datech. Učíníme to třeba tak, že porovnáme hodnotu speciální funkce dat (realizace testové statistiky) s kvantilem rozdělení pravděpodobností. Teorie pravděpodobnosti nám tedy slouží k přechodu mezi popisnou statistikou a metodami statistické inference. Tento právě pojednávaný princip je velmi vhodně vizualizován a popsán v [17] na str. 6 nejen obrázkem, jehož mírně upravenou formu zde přikládám jako obrázek číslo 3.



Obrázek 3: Vztah pravděpodobnosti a statistické inference

Koncept je také v jednoduchosti demonstrován v již zmíněné knize [17] v příkladu 1.1, str. 4. V něm pojednáváme o praktickém problému, kdy zkoumáme data o 100 výrobcích z výrobního procesu. Na základě dat zjistíme, že 10 z těchto výrobků je vadných, ale firma si může dovolit pouze 5% vadnost výrobků. Za pomocí teorie pravděpodobnosti určíme, jak závažný je fakt, že jsme místo 10 vadných výrobků našli 5. Pravděpodobnost objevení 10 vadných výrobků v náhodném výběru velikosti 100 za předpokladu 5% vadnosti je 0.0282, což je velmi malé číslo. Můžeme tedy říci, že nám data ukazují, že vadnost výrobků bude spíše vyšší než povolených 5 %.

Příklad slouží pouze k ilustrování použití pravděpodobnosti a není zde užito řádné matematické teorie *testování statistických hypotéz*. Na základě zmíněných úvah je zřejmé, že v zájmu naší práce bude třeba výletu do světa pravděpodobnosti.

2.2 Charakteristiky náhodné veličiny

Nejzákladnějšími pojmy teorie pravděpodobnosti se nebudeme zabývat a zaměříme se rovnou na charakteristiky náhodné veličiny. Možné realizace náhodné veličiny můžeme shrnout nejen vykreslováním tvaru rozdělení pomocí hustoty nebo distribuční funkce, ale i pouhým jediným reálným číslem. Nechť máme náhodnou veličinu X. Číslo $\mathrm{E}(X)$ definované vztahem:

$$\mathrm{E}(X) = \sum_{i=1}^{n} x_i p_i,$$

respektive

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x f(x) dx$$

nazveme *střední hodnotou* případně *očekávanou hodnotou* náhodné veličiny, pokud takové číslo existuje. Číslem vyjadříme hodnotu, v jejíž blízkosti se budou vyskytovat konkrétní realizace veličiny.

Distribuční funkci můžeme přiřadit i odpovídající *kvantilovou funkci*, vyjádřenou:

$$F^{-1}(\alpha) = inf\{x \in \mathbb{R} : F(x) \geq \alpha\},\ 0 < \alpha < 1,$$

jejíž funkční hodnoty x_α nazveme α - *kvantily* rozdělení. Jsou charakteristické tím, že náhodná veličina se realizuje hodnotou menší nebo rovnu x_α s pravděpodobností nejméně α a větší nebo rovnu x_α s pravděpodobností nejméně $1 - \alpha$.

Hojně používané jsou v různých pravděpodobnostních i statistických výpočtech tzv. *momenty* r-tého řádu. Pokud existují, jsou definovány střední hodnotou příslušné mocniny náhodné veličiny. $\mathrm{E}(X^r)$ nazýváme r-tý obecný moment; pro r = 1 dostáváme střední hodnotu. $\mathrm{E}(X - E(X))^r$ nazýváme r-tý centrální moment; pro r = 2 definujeme *rozptyl* náhodné veličiny a značíme:

$$\mathrm{var}(X) = \mathrm{E}(X - \mathrm{E}(X))^2.$$

Odmocninou z rozptylu rozumíme *směrodatnou odchylku*. V definici rozptylu je důležité, že je odchylka náhodné veličiny od své střední hodnoty $|X - \mathrm{E}(X)|$ umocněna na druhou. To v praxi znamená, že pokud se hodnoty náhodné veličiny realizují blízko střední hodnoty, bude hodnota rozptylu malá (malé číslo umocněné na druhou může být dokonce zmenšeno, pokud je mezi 0 a 1), a naopak pokud je odchylka velká, bude hodnota rozptylu vysoká.

2.3 Některá konkrétní rozdělení

V následující podkapitole si uvedeme některá rozdělení náhodných veličin, jejich charakteristiky, vlastnosti a zvláště *parametry*, které tvar rozdělení jednoznačně určují. Ve statistické inferenci jsou rozdělení ústředním tématem, protože právě jejich parametry nebo tvar chceme odhadnout a o nich formulujeme hypotézy. Uvedeny budou postupně tak, jak se s nimi budeme setkávat při pojednání o naší aplikaci. Neopomeneme také grafickou vizualizaci hustot, ze kterých si můžeme o rozděleních udělat decentní představu.

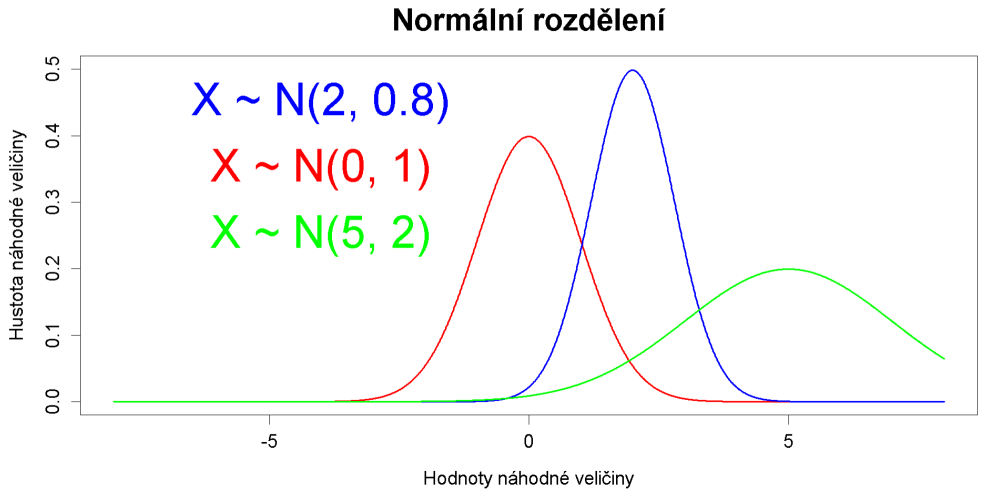
Normální rozdělení

Nejpoužívanějším rozdělením je *normální rozdělení*. Svým symetrickým, spojitým charakterem a tvarem hustoty nazývaným *Bellova křivka* přibližně

popisuje mnoho jevů, které se vyskytují v přírodě, v průmyslu a ve výzkumu. Je základem, na kterém stavíme teorii induktivní statistiky. Značíme $X \sim \mathrm{N}(\mu, \sigma^2)$, kde μ a σ^2 jsou parametry rozdělení, které mají interpretaci střední hodnoty a rozptylu. Hustota rozdělení má tvar:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Pro $\mu = 0$ a $\sigma^2 = 1$ dostáváme *normální normované rozdělení*, jehož distribuční funkci značíme Φ.

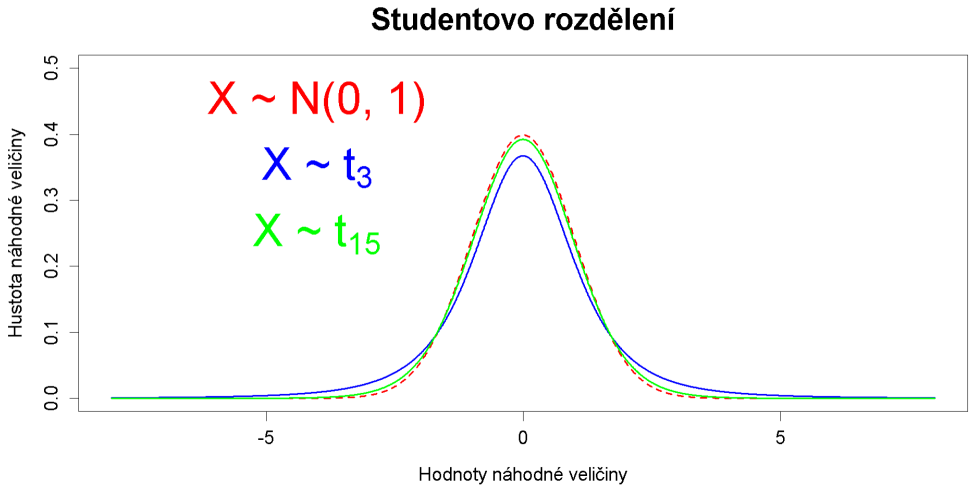


Obrázek 4: Hustota normálního rozdělení s různými parametry

Na obrázku 4 vidíme barevně rozlišené Gaussovy (bellovy) křivky pro různé parametry normálního rozdělení.

Studentovo t-rozdělení

Dalším spojitým, symetrickým rozdělením je *Studentovo t-rozdělení* s jediným parametrem n, který označujeme jako *stupeň volnosti*. Veličinu řídící se tímto rozdělením budeme označovat $X \sim t_n$. Můžeme se dočíst zde [19], že rozdělení vzniklo za účelem kontroly kvality na základě malého rozsahu dat.



Obrázek 5: Studentovo rozdělení s 3 a 15 stupni volnosti

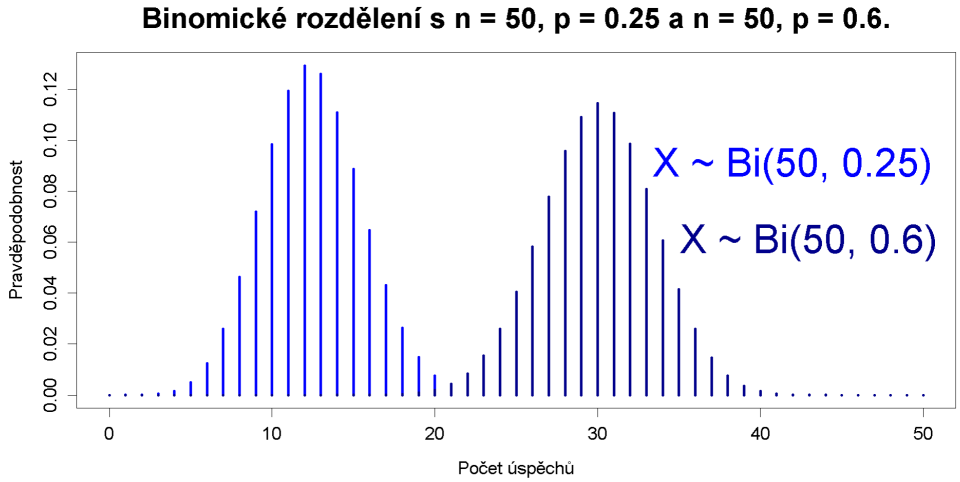
Obrázek 5 ilustruje konvergenci studentova rozdělení. S rostoucími stupni volnosti se totiž více a více přibližujeme normálnímu normovanému rozdělení. Při malém n má hustota menší výšku (maximum) a těžší chvosty (klesá pomaleji). Slouží k popsání standardizovaných vzdáleností střední hodnoty od výběrového průměru za předpokladu, že neznáme rozptyl u dat z normálního rozdělení. Střední hodnota je definována jen pro $n > 1$ a je rovna $\mathrm{E}(X) = 0$. Matematické vyjádření hustoty nalezneme např. v [18] na str. 28, kde je rozdělení označováno jako *rozdělení t*.

Binomické rozdělení

Binomickým rozdělením s parametry n a p se řídí diskrétní náhodná veličina, která popisuje výsledek n náhodných Bernoulliho pokusů, nabývajících hodnot 0 a 1, které reprezentují neúspěch a úspěch. Úspěch nastane s pravděpodobností p. Vyjadřujeme ho pravděpodobnostní funkcí:

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k},\ k = 0, 1, ..., n$$

a značíme $X \sim \mathrm{Bi}(n, p)$. Prakticky často zmiňovaným příkladem tohoto rozdělení je výsledek n hodů mincí. Použití nalezneme ale i ve zdravotnictví a v armádě, kde můžeme počítat např. počet uzdravených, respektive počet zásahů cíle. Na obrázku 6 vidíme pravděpodobnostní funkci dvou veličin s binomickým rozdělením vyjádřenou pomocí histogramu.



Obrázek 6: Binomické rozdělení s parametry $n = 50$ a $p = 0.25$, $n = 50$ a $p = 0.6$

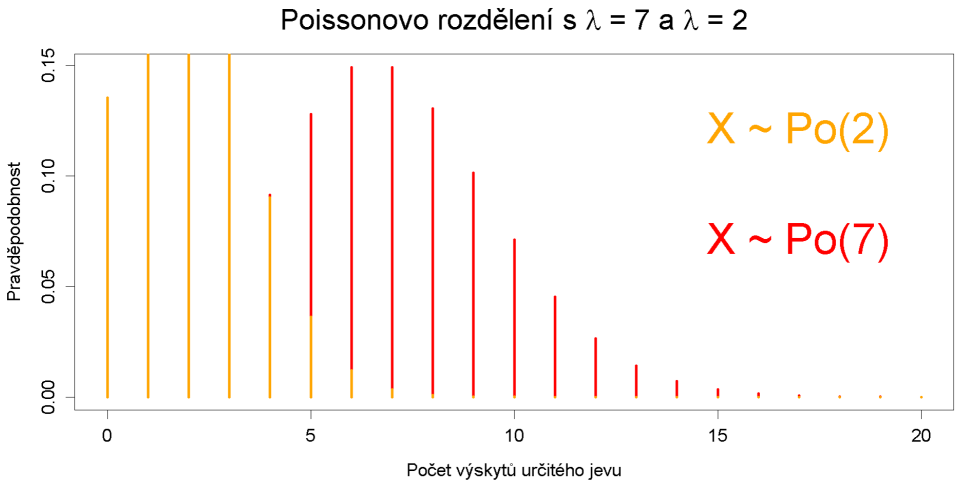
Střední hodnota a rozptyl jsou určeny vztahy $\mathrm{E}(X) = np$ a $\mathrm{var}(X) = np(1-p)$.

Poissonovo rozdělení

Poissonovo rozdělení má diskrétní náhodná veličina s parametrem λ, která se realizuje libovolným nezáporným reálným číslem k s pravděpodobností:

$$\mathrm{P}(X = k) = \frac{\lambda^k}{k!} e^{-\lambda},\ \lambda > 0$$

Rozdělení je charakterizováno parametrem λ, který vyjadřuje počet výskytů určitého jevu v nějakém časovém úseku (na nějaké ploše, v nějaké vzdálenosti...) a reprezentuje střední hodnotu a dokonce i rozptyl. Veličina se označuje $X \sim \mathrm{Po}(\lambda)$. Na obrázku 7 je vyobrazena hustota rozdělení $X_1 \sim \mathrm{Po}(7)$ a $X_2 \sim \mathrm{Po}(2)$.



Obrázek 7: Poissonovo rozdělení s parametrem $\lambda = 7$ a $\lambda = 2$

2.4 Náhodný výběr a statistika

Věřím, že jsme probrali všechny potřebné aspekty teorie pravděpodobnosti; nyní už je snad konečně čas postarat se o důvod pravděpodobnostního snažení. V následujících podkapitolách se budeme zaobírat statistikou a jejími metodami. Úvodem je důležité se zmínit o formalitě, že praktickou interpretaci pravděpodobnosti budeme chápat *frekventisticky* v souladu s literaturou [2], str. 155. To znamená, že pokud je pravděpodobnost zpoždění vlaku o 5 minut a více rovná 10 % ($\mathrm{P}(X \geq 5) = 0.1$, kde X je zpoždění vlaku), tak 10 ze 100 vlaků bude mít zpoždění.

Abychom mohli zkoumat počet vlaků se zpožděním nebo testovat, zda-li je mince falešná či pravá - čili souhrně testovat hodnotu vektorového parametru $\boldsymbol{\theta}$ z *parametrického prostoru* $\boldsymbol{\Theta}$ příslušný *statistickému znaku* X mající *kvantitativní* nebo *kvalitativní charakter* - potřebujeme definovat data sadou náhodných veličin, které nám na základě náhodného pokusu (např. přijíždění vlaku či hod mince) poskytnou konkrétní hodnoty. Přesně toto nám umožňuje *náhodný výběr* vyslovený následující definicí z [2], str. 160:

Definice 1. n-tice nezávislých náhodných veličin $X_1, \ldots, X_n$, které mají stejné rozdělení jako zkoumaná náhodná veličina X, se nazývá náhodný výběr rozsahu n z rozdělení náhodné veličiny X.

Abychom zamezeli nedorozumění podotkněme, že náhodné veličiny $X_1, \ldots, X_n$ nejsou jen z rozdělení stejného typu (např. binomického), ale že dané rozdělení má také stejné parametry a tím i celý tvar. Matematický ob-

jekt, který označuje více veličin s obecně jakýmkoli rozdělením, nazýváme *náhodný vektor* a značíme $\mathbf{X} = (X_1, \dots, X_n)$.

Konkrétní data (*statistický soubor*) získáme realizací náhodného výběru (např. opakovaným zapisováním délky zpoždění, případně zapisováním, zdali měl či neměl vlak zpoždění, zaobíráme-li se diskrétní veličinou), kterou stejně jako v případě náhodné veličiny nazýváme *realizací* resp. hodnotou náhodného výběru a značíme $\mathbf{X}(\omega) = \mathbf{x} = (x_1, \dots, x_n)$. Množina možných hodnot náhodného výběru se nazývá *výběrový prostor*.

Výběrovou funkcí nebo-li *statistikou* rozumíme každou *borelovsky měřitelnou* funkci náhodného výběru a značíme $T(X_1, \dots, X_n) = T(\mathbf{X})$ s realizacemi, které označujeme $T(x_1, \dots, x_n)$. Je třeba zdůraznit, že hodnoty výběrové funkce jsou nepřímo závislé na hodnotě parametru zkoumaného rozdělení a to prostřednictvím náhodného výběru. Důležité je si také uvědomit, že statistika je stále náhodná veličina, protože borelovsky měřitelná funkce náhodné veličiny je znovu náhodná veličina. Podle účelu statistiky rozlišujeme dva základní typy:

Bodovým odhadem parametru θ rozumíme výběrovou funkci $T(X_1, \dots, X_n)$, jejíž rozdělení nezávisí na θ, avšak její funkční předpis ano. Její reálné hodnoty musí dobře aproximovat hodnotu parametru θ. V naší aplikaci odhadujeme pomocí bodového odhadu vždy pouze jednodimenzionální parametr.

Jako testovou statistikou chápeme tu, která přináší informaci o platnosti nulové hypotézy. Stejně jako v případě bodového odhadu se zaměříme pouze na 1-dimenzionální parametr $\theta \subset \Theta$.

2.5 Vlastnosti odhadů a některé důležité bodové odhady

Odhady charakterizujeme pomocí jejich vlastností. Ty jsou úzce spjaty s charakteristikami, které značíme $\mathrm{E}_\theta[T(\mathbf{X})]$, resp. $\mathrm{var}_\theta[T(\mathbf{X})]$ (v praxi např. $\mathrm{E}_\mu[T(\mathbf{X})]$), protože jsou jejich hodnoty různé při změně neznámého vektorového parametru θ.

Častým požadavkem na vlastnost odhadu je jeho *nestrannost*.

Definice 2. Výběrová funkce $T(\mathbf{X})$ je nestranným odhadem hodnoty parametru θ, jestliže platí:

$$\mathrm{E}_\theta[T(\mathbf{X})] = \theta, \forall\theta \in \Theta.$$

Rozdíl $b(\theta) = |\mathrm{E}_\theta[T(\mathbf{X})] - \theta|$ nazveme *vychýlením odhadu*.

Pokud platí $\lim_{n\to\infty} \mathrm{E}_\theta[T(\mathbf{X})] = \theta, \forall\theta \in \Theta$ je odhad *asymptoticky nestranný*. To můžeme chápat tak, že se blíží nestrannosti s rostoucím n. Nestrannost nemusí být sama o sobě garantem kvality odhadu, pouze vyjadřuje, že se hodnota bodového odhadu bude okolo reálné hodnoty parametru. Konkrétní realizace však může být vzdálená, pokud má statistika velký rozptyl.

Další důležitou vlastností je *konzistence odhadu*.

Definice 3. Výběrová funkce $T(X_1, \ldots, X_n)$ je konzistentním odhadem parametru θ, platí-li:

$$\lim_{n\to\infty} \mathrm{P}_\theta(|T(\mathbf{X}) - \theta| < \varepsilon) = 1, \forall\varepsilon > 0, \forall\theta \in \Theta$$

, tj. posloupnost s rostoucím počtem pozorování statistického znaku X konverguje podle pravděpodobnosti ke skutečné hodnotě parametru θ.

Pomocí konvergence podle pravděpodobnosti vyjadřujeme přiblížování testové statistiky k reálné hodnotě parametru. V praxi to znamená, že se hodnoty bodových odhadů pro nízká n pohybují dále od hodnoty parametru a pro vysoká n velmi blízko. Hodnota n se pak v praxi vyčísluje, pro požadovaný rozptyl, který chceme co nejmenší.

Kvalitu odhadu (estimátoru) můžeme posuzovat také pomocí jeho rozptylu. Čím menší rozptyl, tím je odhad lepší - má vyšší *eficienci*. Dále můžeme posuzovat kvalitu odhadu pomocí *intervalů spolehlivosti*, které sdělují, do jakého intervalu bude testová statistika spadat s $1 - \alpha\%$ pravděpodobností. V naší aplikaci používáme k posouzení kvality odhadu dvou čísel, které nejsou závislé na nestrannosti. Prvním je *střední absolutní chybu* $E_\theta|\hat{\theta} - \theta|$, která udává hodnotu, kolem které se budou soustřeďovat absolutní hodnoty chyb. Využijeme také *střední čtvercovou chybu* neboli MSE (mean squared error) vyjádřenou vztahem $E_\theta(\hat{\theta} - \theta)^2$, která udává rozptyl chyb. $\hat{\theta}$ chápeme jako odhadovou statistiku parametru θ.

Mezi všemi odhady se dále zaměříme na ty nejznámější. Ukážeme si, jak je matematicky vyjadřujeme, a řekneme si o některých jejich speciálních vlastnostech. Nechť máme náhodný výběr $X_1, \ldots, X_n$ příslušný libovolné náhodné veličině X.

V případě že existuje první moment, tak definujeme veličinu $\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$, kterou nazveme *výběrový průměr*. Je dokázáno, že je nejlepším nestranným lineárním odhadem střední hodnoty X; zároveň je odhadem konzistentním. Pokud máme výběr z normálního rozdělení, lze za pomocí vět o transformaci náhodných veličin dokázat, že platí

$$\overline{X}_n \sim \mathrm{N}(\mu, \frac{\sigma^2}{n}). \tag{1}$$

Dále uvažujeme existenci druhého centrálního momentu. *Výběrový rozptyl* budeme pro $n > 1$ značit $S_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$. Stejně jako výběrový průměr je tento odhad nestranným a konzistentním ale tentokrát aproximuje rozptyl veličiny X. Jeho realizaci značíme s_n^2, o číslu $n-1$ někdy hovoříme jako o stupních volnosti odhadu rozptylu protože při znalosti výběrového průměru, můžeme poslední pozorování dopočítat ze vztahu $\sum_{i=1}^{n}(x_i - \overline{x}) = 0$. Odmocnina z výběrového rozptylu se nazývá *výběrová směrodatná odchylka* a ztrácí nestrannost, protože pro náhodnou veličinu X obecně neplatí: $\sqrt{\mathrm{E}(X)} = \mathrm{E}(\sqrt{X})$.

Označení výběrový rozptyl se v některé statistické literatuře (např. [18]) přiřazuje *výběrovému momentu druhého řádu* určeného vztahem $M_2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$, protože je podobně účinný při aproximaci rozptylu (má přibližně stejný rozptyl). Není sice nestranný, ale je dokázáno v [18], str. 102-103, že má menší střední čtvercovou chybu. Nicméně dokonce ani M_2 není ve smyslu MSE optimální, jak se lze dočíst na stejném místě.

2.6 Empirické odhady hustoty

Kromě hodnot parametrů můžeme pro náhodný výběr odhadovat i hustotu zkoumané veličiny X pomocí tzv. *empirických odhadů hustoty*. Použijeme-li je, získáme slušnou představu o tvaru rozdělení pravděpodobnosti X. Řadí se mezi *neparametrické* metody, protože nemusíme mít předpoklady o tvaru rozdělení nebo o hodnotě parametru. V závislosti na správném zvolení určitých parametrů, které ovlivňují tvar výsledné funkce, dostáváme více či méně přesný odhad hustoty $f(x)$. V této kapitole bylo čerpáno ze zdrojů [20], [21] a [22].

Častým odhadem hustoty je *histogram*. Použijeme-li ho k vizualizaci dat, budeme pomocí sloupců aproximovat reálné hodnoty hustoty. Pro náhodný výběr rozsahu n můžeme odhad $\forall x \in [a_k, a_{k+1})$ vyjádřit (s úpravami převzato z [20], str. 11):

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} I_{[a_k, a_{k+1})}(X_i), \tag{2}$$

kde h rozumíme šířku jednoho sloupce vyjádřenou intervalem $[a_k, a_{k+1}) \in \mathbb{R}$. Každý sloupec reprezentuje počet pozorování, které do něj spadají, znormovaný velikostí výběru n a zmíněnou šířkou h, které zajišťují, že funkce bude mít vlastnosti hustoty. Funkci $I_{[a_k, a_{k+1})}(X_i)$ nazýváme *indikátorovou funkcí*, která nabývá hodnoty 1, v případě že realizace náhodné veličiny spádá do intervalu $[a_k, a_{k+1})$, a 0, pokud nespadá. Histogramem budeme v aplikaci aproximovat hustoty odhadových statistik.

Velmi používaný *jádrový odhad* je zpravidla hladší než histogram. Definujeme ho pomocí *jádra* $K(x)$, jehož tvar je určen splněním následujích podmínek zkráceně převzatých z [21], str. 2:

$$K(x) \text{ je symetrické} \tag{I}$$

$$\int_{\mathbb{R}} K(x)dx = 1 \tag{II}$$

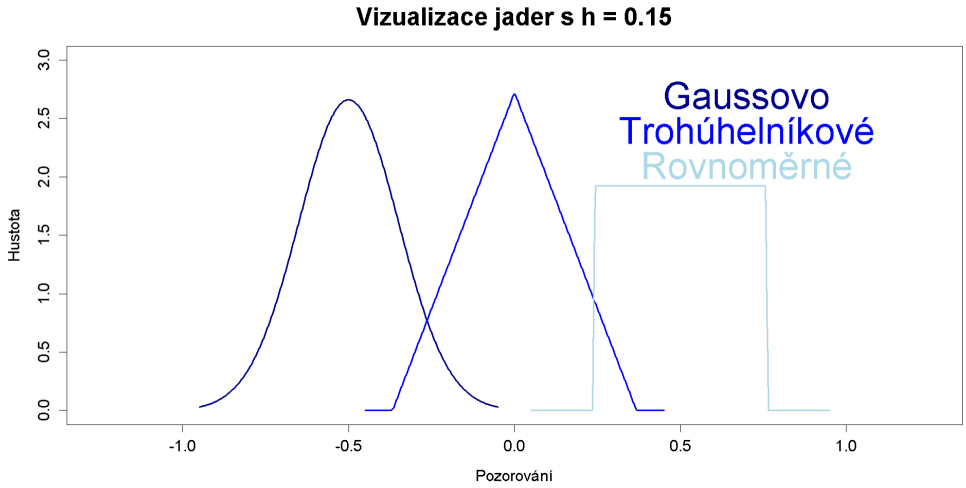
$$\int_{\mathbb{R}} x^j K(x)dx = 1, j = 1, \dots, k-1 \tag{III}$$

$$\int_{\mathbb{R}} x^k K(x)dx \neq 1 \tag{IV}$$

Podmínka I zajišťuje rozumný tvar aproximace. Podmínky III a IV pracují s řádem jádra k, které charakterizuje jeho první nenulový moment. Výsledná hustota se získá tak, že každému pozorování z naší datové sady přiřadíme jádro odpovídajícího typu se stejnou šířkou vyhlazovacího okna h. Tyto zpravidla miniaturní funkce nasčítáme a vydělíme velikostí výběru n a šířkou vyhlazovacího okna h. Výsledná funkce je také hustotou, protože platí podmínka II. Pro náhodný výběr $X_1, \dots, X_n$ vypočítáme hodnoty v bodě x ze vztahu:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{x - X_i}{h}), x \in \mathbb{R}. \tag{3}$$

V aplikaci se setkáme se třemi typy jader řádu $k = 2$: *Gaussovské* pracuje s hustotou normálního normovaného rozdělení. *Rovnoměrné* konstruuje kolem bodů hustotu tvaru obdélníku a *trojúhelníkové* se vyznačuje, jak už název napovídá, hustotou tvaru trojúhelníka. Jak jádra vypadají, přibližuje obrázek 8.



Obrázek 8: Vizualizace typů jádrových funkcí

Funkční předpisy těchto jader jsou dány tvary:

$$\text{Gaussovské}{-}K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

$$\text{Rovnoměrné}{-}K(x) = \frac{1}{2} I(|x| \leq 1)$$

$$\text{Trojúhelníkové}{-}K(x) = (1 - |x|) I(|x| \leq 1)$$

2.7 Testování hypotéz a příslušné pojmy

Ústředním tématem jak statistcké inference tak naší aplikace je *testování hypotéz*. Hypotézy obecně formulujeme, když chceme na základě dat rozhodnout o podpoře nějakého tvrzení nebo ho naopak vyvrátit (např. nevinnost obžalovaného u soudu). Rozhodnutí přiřazujeme každé datové sadě zvlášť. V aplikaci se zabýváme pouze jednorozměrnými *parametrickými testy*, kterými odhadujeme hodnotu parametru θ zkoumaného rozdělení. Také bych rád zmínil učební text Pardubické univerzity [23], ze kterého jsem čerpal při pojednání o statistických hypotézách, a [24] pojednávající o velikosti testu. Nechť máme náhodný výběr $\mathbf{X}$ ze zkoumaného rozdělení X. Při testování hypotézy o parametru X se řídíme následujícím postupem:

1. Pokud jsou známé, stanovíme předpoklady o rozdělení zkoumané veličiny a formulujeme *nulovou hypotézu* $H_0 : \theta = \theta_0$, jejíž přijetí podá důležitou informaci, a kterou považujeme apriori za platnou. *Alternativní hypotéza* H_A

je formulována tak, že se vylučuje s tvrzením nulové hypotézy. Podle umístění vzhledem k nulové hypotéze rozlišujeme *oboustrannou*, *levostrannou* resp. *pravostrannou* alternativu.

2. V dalším kroku volíme důležitou konstantu tzv. *hladinu testu* α, která omezuje pravděpodobnost zamítnutí nulové hypotézy, pokud platí. Rozhodnutí nemůžeme nikdy učinit se 100% jistotou, ledaže bychom znali přesný tvar rozdělení (pak bychom ale nemuseli nic testovat). Volíme zpravidla 5% nebo 1% chybovost, podle důsledku který by chyba mohla mít (např. vysoké riziko smrti pacienta nebo finanční ztráty).

3. Zvolíme vhodnou testovou statistiku, která bude na základě předpokladů dobře vypovídat o rozdělení dat, uvažujeme-li ho za platnosti nulové hypotézy, a vyčíslíme její hodnotu pro náš výběr.

4. V tomto kroku můžeme postupovat dvěma způsoby. Buď stanovíme podobu *kritického oboru* $\mathrm{W} \subset \mathbb{R}$ anebo vypočítáme *p-hodnotu*. V obou případech za platnosti H_0. Kritický obor obsahuje velmi nepravděpodobné hodnoty testové statistiky a p-hodnota udává pravděpodobnost, s jakou dostaneme naše konkrétní data anebo data ještě více odporující nulové hypotéze.

5. Ve finálním kroku vyslovujeme závěr testu, který stanovíme pomocí kroku 4. na zvolené hladině α. Hypotézu *zamítáme* na hladině testu α, pokud padne hodnota testového statistiky do kritického oboru anebo je p-hodnota velmi malá (menší než α). V opačnému případě hypotézu *nezamítáme ve prospěch alternativy*.

Pokud nulovou hypotézu zamítneme, neznamená to, že neplatí, nýbrž pouze to, že realizace náhodného výběru svědčí o její neplatnosti. Stejně tak při nezamítnutí nulové hypotézy nemůžeme se 100% jistotou tvrdit, že hypotéza je pravdivá; pravda je, že data svědčí v její prospěch. Množiny hodnot parametrů příslušných hypotéz označujeme Θ_0 a Θ_A. Tvrzení $H_0 : \theta = \theta_0$ tak lze matematicky zapsat jako $\theta \in \Theta_0$.

Číslo, které ohraničuje interval kritického oboru s oborem, ve kterém nulovou hypotézu nezamítáme, nazýváme *kritickou hodnotou*. Toto číslo je závislé na typu hypotézy, takže jich může být více. Jeho hodnotu získáme pomocí F^{-1} kvantilové funkce rozdělení testové statistiky a značíme např. u_α, respektive t_α. Množinový doplněk kritického oboru nazveme *obor přijetí nulové hypotézy*.

Při rozhodování se o platnosti hypotézy se můžeme dopustit 2 chyb. Zpravidla závažnější *chyby 1. druhu* se dopustíme, pokud zamítneme H_0, ačkoliv platí. Pravděpodobnost této chyby nemůže být pro konkrétní data vyšší než α. Nicméně je třeba vzít v úvahu také *chybu 2. druhu* β, která udává pravděpodobnost, že hypotézu nezamítneme, když neplatí. Ta s rostoucí chybou 1. druhu klesá. Zjišťujeme ji pro konkrétní hodnotu parametru θ z množiny alternativní hypotézy. Kromě toho se tato pravděpodobnost zmenšuje také s

rostoucí velikostí výběru. Správné rozhodnutí učiníme s pravděpodobnostmi, když uvedené chyby odečteme od 1.

V naší aplikaci se budeme kromě zmíněných pojmů souvisejících zabývat také ještě s jedním pojmem, který přímo nenajdeme ve skriptech základního kurzu [2], ale úzce souvisí s testováním hypotéz. Je jím pravděpodobnostní komplement chyby 2. druhu (tedy pravděpodobnost, že zamítneme H_0, když neplatí), tzv. *síla testu*, kterou lze vyjádřit následujícím vztahem $S = 1 - \beta$. Síla testu je rozhodující faktor, pomocí kterého si můžeme vybrat ze dvou a více možných testů, které splňují námi požadované předpoklady, a jsou vhodné pro testování hypotézy (test s větší silou je žádanější). Navíc se může stát, že je hypotéza formulována tak, že je chyba 2. druhu závažnější než chyba 1. druhu. V takovém případě stanovíme nejdříve požadovanou sílu testu a z ní následně dopočítáme hladinu testu. Nebo můžeme určit požadovaný poměr chyb a dopočítávat jak hladinu testu, tak sílu testu.

Nechť máme náhodný výběr o rozsahu n a testujeme hypotézu H_0. Sílu testu počítáme pro konkrétní hodnotu parametru hypotézy $H_1 : \theta = \theta_1$, která odporuje nulové hypotéze. Odchylka $|\theta_0 - \theta_1|$ vyjadřuje jakousi míru porušení nulové hypotézy a obecně se nazývá *velikost efektu* (anglicky magnitude of the effect). Síla testu pro příslušnou velikost efektu vyjadřuje schopnost testu rozlišit odchylky od nulové hypotézy (zvláště ty mírné). Čím menší je velikost efektu, tím menší hodnoty bude S dosahovat. S je také přímo úměrná hladině testu α; obě hodnoty však můžeme optimalizovat zvyšováním velikosti výběru.

Funkci, která vzhledem k formulaci H_0 přiřadí libovolné hodnotě parametru pravděpodobnost, že zamítneme H_0, nazveme *silofunkcí* a značíme $S(\theta)$. Její tvar (a tedy i předpis) je závislý na hodnotě parametru v nulové hypotéze. Minimum silofunkce v případě oboustranné alternativy je hodnota parametrického prostoru nulové hypotézy, tudíž v tomto případě $S(\theta_0)$ již není síla testu. Toto číslo nazýváme *velikost testu* V. Velmi často je rovno hladině α, ale v aplikaci si ukážeme příklad, kdy tomu tak být nemusí. Při dané hladině α chápeme sílu testu jako maximální pravděpodobnost zamítnutí nulové hypotézy, za předpokladu že platí, na základě výběru velikosti n. Vztah vyjádříme jako $V = \sup_{\theta \in \Theta_0} (T(\mathbf{X}) \in W)$.

2.8 Některé konkrétní testové statistiky

Při testování používáme vhodných testových statistik, které používáme podle formulace nulové hypotézy a známých předpokladů. V naší aplikaci použijeme pouze dvě testové statistiky.

V případě že máme výběr $X_1, \ldots, X_n$ z normálního rozdělení se známým

rozptylem, můžeme ze znalosti rozdělení výběrového průměru a následné standardizace odvodit rozdělení testové statistiky:

$$U = \frac{\overline{X}_n - \mu}{\sigma}\sqrt{n} \sim \mathrm{N}(0,1). \tag{4}$$

Nechť máme nyní výběr z normálního rozdělení, ale tentokrát s neznámým ropztylem. Lze dokázat (např [18], str. 74), že při nahrazení rozptylu výběrovou směrodatnou odchylkou bude mít za předpokladu $n \geq 2$ a $\sigma^2 > 0$ testová statistika

$$T = \frac{\overline{X}_n - \mu}{S_n}\sqrt{n} \sim \mathrm{t}_{n-1} \tag{5}$$

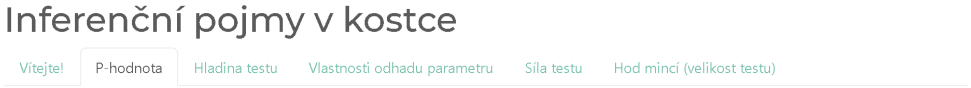
studentovo rozdělení o $n-1$ stupních volnosti. Nahrazení směrodatné odchylky výběrovým protějškem intuitivně vysvětluje konvergenci studentova rozdělení k normálnímu s rostoucím rozsahem výběru, protože výběrová směrodatná odchylka je nestranným a konzistentním odhadem σ. To znamená, že pro rostoucí n se budeme přibližovat vztahu 4. Řádný důkaz nalézáme v literatuře (např. v [2], str. 195). Pracujeme v něm s rovnicí 4, s vzájemnou nezávislostí S_n^2 a $\overline{X}_n$ a podílem $\frac{(n-1)S_n^2}{\sigma^2}$, který má rozdělení χ^2_{n-1}.

3 Rozbor aplikace

Finální 3. kapitola je stěžejní, neboť právě ona má ukázat praktický výsledek práce. Je koncipována tak, aby se skrze ní realizovaly předchozí dvě kapitoly, které slouží jako nástavby. Postupně si projdeme všechny části z matematického (a také trochu programátorského) hlediska a budeme demonstrovat, co může případného zvědavce obohatit. Je nutno podotknout, že ideálně by měl uživatel (obzvláště jedná-li se o nového studenta či statistického začátečníka) při použití aplikace náhlížet do této práce a projít si stěžejní stránky skript základního kurzu [2] (případně také pročíst další literaturu). V neposlední řadě budou detailně popsány 2 způsoby, kterými může uživatel aplikaci spustit.

Ústřední myšlenku aplikace můžeme vyjádřit jako interaktivní vizualizace pojmů na příkladech, za pomocí realizace náhodných výběrů z příslušných *neznámých* rozdělení. Simulujeme tak praktické úlohy, kdy dostaneme data, na jejichž základě chceme něco zjistit nebo ověřit.

Na obrázku 9 vidíme záhlaví aplikace, které je viditelné při jakékoli další akci. Nalezeneme na něm název a, jak už bylo zmíněno v úvodu práce, základní rozdělení do 6 částí; každá z nich je reprezentována 1 záložkou (`tab` z `tabsetPanel`). Většina záložek se drží jednoduchého schématu, ve kterém je na prvním místě název, následovaný vstupy, pomocnými texty a grafy vhodně umístěnými do rozložení `sideBarLayout`. V bočním panelu máme zpravidla umístěný text, který má za úkol uživateli vysvětlit úlohu demonstrující příslušný pojem. V hlavním panelu se setkáme s tlačítky, která aktivují vizuální znázornění prostřednictvím výstupů umístěných pod nimi. Ty mohou být také okomentovány pomocným textem.



Obrázek 9: Náhled záhlaví aplikace

3.1 Domovská záložka

Na obrázku číslo 10 je zobrazen náhled první záložky. Kromě názvu zde nalezneme text, který podává odpovědi na otázky v podnadpisech. Můžeme si všimnout, že se jedná pouze o `fluidPage`, v níž jsou uvnitř studnicových panelů vypsány důležité informace a některé bibliografické údaje. Pod číslem 1 se můžeme stručně seznámit s tím, co tato aplikace dokáže. U čísla 2 je v krátkosti naznačeno, co je obsahem aplikace. Číslo 3 označuje krátký od-

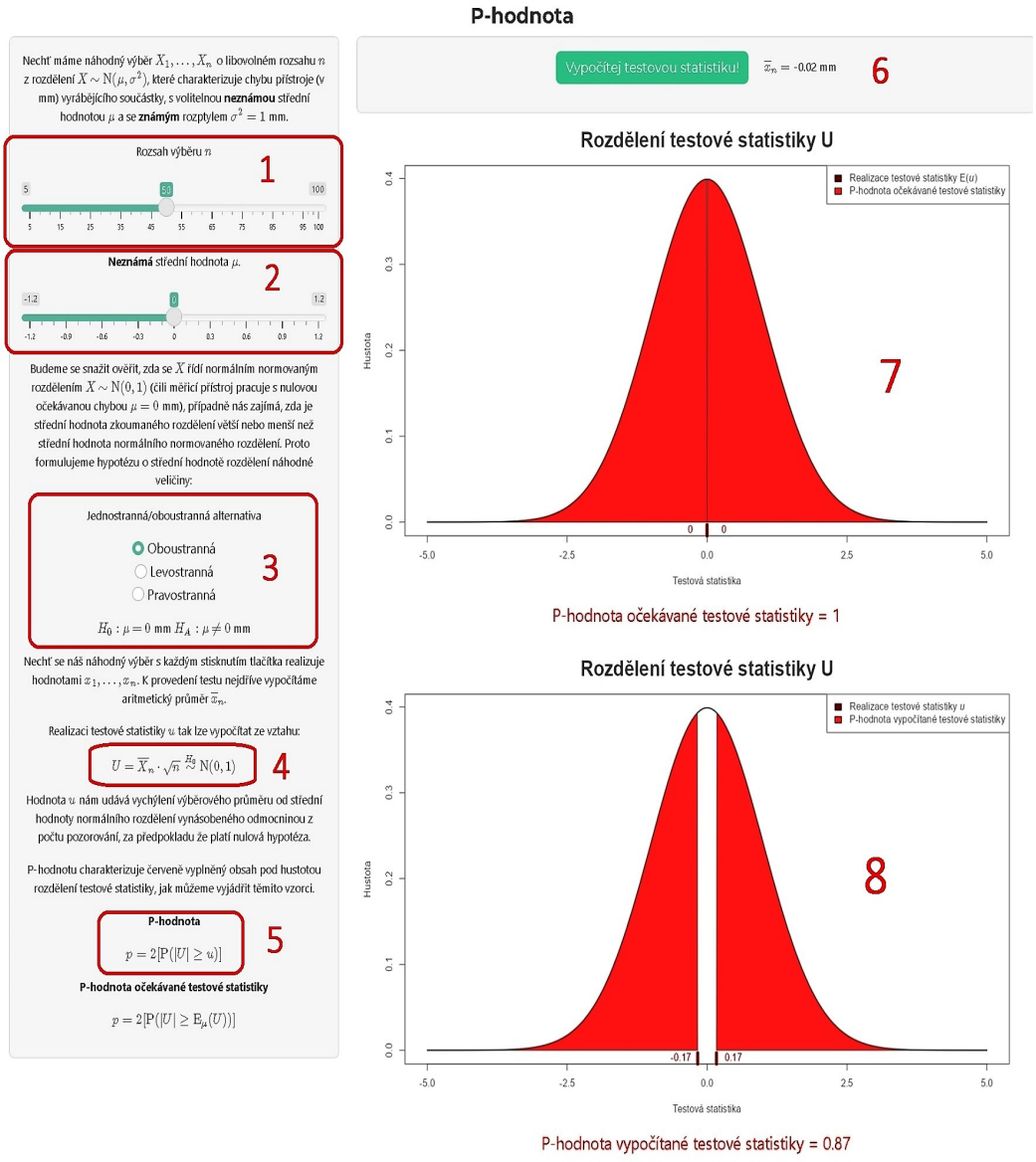
stavec s bibliografickými údaji. Nesmíme také opomenout poslední odstavec označený číslem 4, kde nalezneme email na autora, který může uživatel využít, chce-li podat dotaz či oznámit některé problémy. Na stejném místě si lze pomocí `downloadLink` stáhnout pdf této práce, která může pomoct k lepší orientaci a k objasnění matematiky.

Obrázek 10: Domovská stránka aplikace

3.2 P-hodnota

P-hodnota je při práci se statistickým softwarem velmi důležité číslo. Při testování hypotéz okamžitě dostáváme informaci o tom, jak si naše data vedou za platnosti nulové hypotézy H_0. Osvědčuje se rozhodování o platnosti H_0 na základě srovnání p-hodnoty s hladinou testu α, jak už bylo zmíněno

při popisu testování hypotéz. P-hodnotu totiž můžeme chápat jako nejmenší hladinu významnosti, při které bychom považovali výskyt našich dat za statisticky významný. V následujících řádcích si podrobněji rozebereme, jak máme p-hodnotu uchopenou v aplikaci.



Obrázek 11: Náhled záložky *p-hodnota*

Na obrázku číslo 11 můžeme vidět veškerý obsah záložky. Boční panel

umístěný nalevo v šedém pozadí studnicového panelu (`wellPanel`) nás seznamuje s úlohou testování pomocí p-hodnoty, zda-li máme být na základě dat znepokojeni výkonem přístroje vyrábějícího součástky (např. šrouby či matice). To znamená, zda-li je podezření, že přístroj pracuje s nenulovou chybou (v mm). P-hodnota je demonstrována pomocí dvou podobných grafů z nichž první znázorňuje p-hodnotu, kterou očekáváme, vyjádřenou pomocí střední hodnoty testové statistiky. Druhý graf ukazuje p-hodnotu pro náš konkrétní výběr. Jednotlivé části stránky jsou označeny červenými čísly, které slouží k lepší orientaci.

Červeným číslem 1 máme označený posuvník, kterým volíme rozsah výběru n, neboli počet pozorování normálně rozděleného statistického znaku X, který reprezentuje odchylky požadované délky od naměřené délky vyrobené součástky. Pro větší rozsah výběru budeme dostávat přesnější odhad parametru μ. V případě, že hypotéza neplatí, nám vynásobení odhadu $\sqrt{n}$, jak vidíme v oválu 4, brzy naznačí nepravost hypotézy. V oválu označeném 2 vybíráme střední hodnotu zkoumaného rozdělení, kterou v praxi *neznáme*. Můžeme demonstrovat, že se vzdalující se reálnou hodnotou parametru od hodnoty v nulové hypotéze (s narustající reálnou chybovostí přístroje) se bude p-hodnota očekávané testové statistiky i konkrétní p-hodnota našeho výběru zmenšovat (a tudíž data svědčí ve prospěch nenulové chyby přístroje). V sekci 3 si určujeme typ hypotézy podle toho, co chceme v praxi ověřit (např. můžeme chtít ověřit tvrzení, že se očekávaná chyba měření vychyluje pouze jedním směrem). Vztah 4 dostáváme užitím testové statistiky, kterou jsme si uvedli pod číslem 4, kde dosadíme hodnoty $\mu = 0$ a $\sigma^2 = 1$. Výpočet p-hodnoty, jak se provádí ve statistických softwarech, můžeme vidět v oválu s číslem 5. Výsledná p-hodnota bude kolísat kolem *p-hodnoty očekávané testové statistiky* (v našem případě $\mathrm{E}(U) = \mu\sqrt{n}$), kterou pro oboustrannou alternativu vyjádříme výpočtem:

$$\begin{aligned}\mathrm{P}(|U| \geq \mu\sqrt{n}) &= 1 - \mathrm{P}(-\mu\sqrt{n} \leq U \leq \mu\sqrt{n}) = 1 - \Phi(\mu\sqrt{n}) - \Phi(-\mu\sqrt{n}) = \\ &= 2\Phi(\mu\sqrt{n})\end{aligned}$$

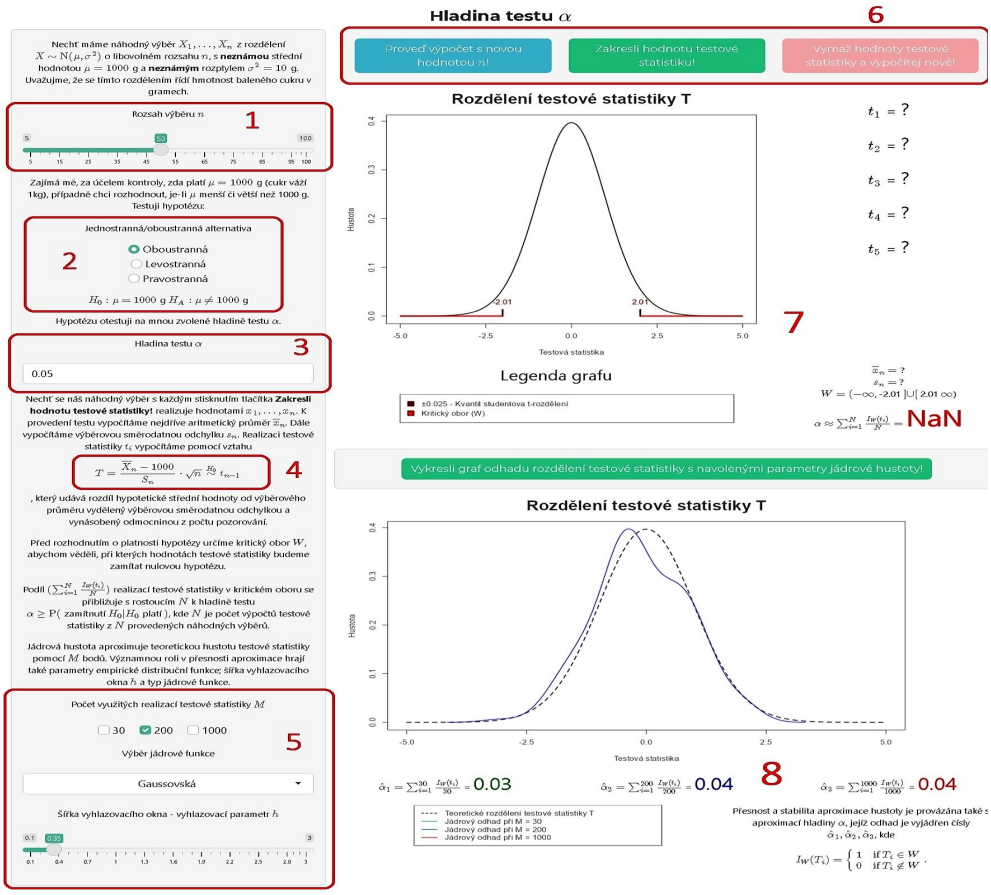
, kde jsme použili pravděpodobnosti komplementu náhodného jevu, vyjádření náhodného jevu pomocí distribuční funkce a symetrie normálního normovaného rozdělení kolem 0.

Studnicový panel označený číslem 6 obsahuje tlačítko, kterým se realizuje náhodný výběr z $\mathrm{N}(\mu, 1)$ a vykreslí do grafů příslušné p-hodnoty. Vedle tlačítka je uvedena realizace výběrového průměru (aritmetický průměr dat), kterou může zvědavec porovnávat s grafy, aby si udělal představu o vlivu výběrového průměru na p-hodnotu. Graf 7 zdůrazňuje, že testová statistika není číslo, nýbrž veličina, a tudíž je p-hodnota střední hodnoty testové statis-

tiky při neměnných parametrech úlohy stále stejná. Obsah plochy v grafu 8 by měl při opakování výběru kolísat kolem obsahu plochy v grafu 7.

3.3 Hladina testu α

Za pomoci hladiny α určujeme, jakou hodnotou je omezena chyba 1. druhu při testování hypotéz. Chápeme ji jako pravděpodobnost zamítnutí pravdivé hypotézy, protože předpokládáme, že H_0 platí. Je důležité se u konkrétních příkladů zamyslet, jak konstantu zvolíme. Důsledky volby α na testování hypotéz se nám pokusí přiblížit záložka znázorněná na obrázku číslo 12.



Obrázek 12: Náhled záložky *hladina testu* α

V záložce se zabýváme problémem, kdy chceme určit, zda-li je důvod k podezření, že normálně rozdělená hmotnost cukru v obchodě (X) nevyhovuje

předepsané hmotnosti 1000 g, vyznačené na obalu. Stejně jako v přechozím příkladu demonstrujeme pojem pomocí testování hypotézy o střední hodnotě normálně rozdělené X, ale tentokrát *neznáme* rozptyl, jehož hodnota reprezentuje určitou povolenou toleranci hmotnosti. Hypotézu testujeme pomocí kritického oboru. Způsob testování je podrobně popsán v bočním panelu, kde si také volíme různé hodnoty proměnných, které v testu figurují. Kromě toho si zde můžeme navolit také parametry jádrového odhadu hustoty pomocí odpovídajících vstupů, který slouží k aproximaci hypotetického rozdělení testové statistiky. V hlavním panelu budeme popisovat grafy a další výstupy, které interaktivně pobízejí k zamyšlení.

Červeným číslem 1 je označen posuvník s počtem výsledků vážení cukru n, který má podobné vlivy na testovou statistiku v oválu 4 (viz vzorec 5) jako tomu bylo v předchozí záložce. Z konzistence výběrového průměru a výběrové směrodatné odchylky plyne, že s rostoucím n dostáváme přesnější odhady parametrů μ a σ^2. Navíc nám rozsah výběru určuje i tvar t-rozdělení, protože $n-1$ označuje počet stupňů volnosti. Označení 2 nám umožňuje radiovými tlačítky volit typ hypotézy, která v tomto případě vždy platí (již nelze modifikovat reálnou hodnotu parametru μ) a také zde vidíme příslušnou formulaci hypotéz. V praxi se můžeme zaměřit, zda-li je cukru spíše více anebo méně. V sekci 3 volíme pomocí číselného vstupu hodnotu hladiny testu, kterou v praxi chceme co nejmenší. Nicméně uživatel si může navolit libovolnou hodnotu v intervalu (0, 0.3], abychom mohli i absurdnější volbou demonstrovat chování testu. Nižší α volíme, je-li rozhodování o cukru velmi důležité (např. v prestižním obchodě). Interval je zabezpečený uživatelskou zpětnou vazbou v případě, že uměle zadáme hodnotu příliš vysokou. Měnit α můžeme i tehdy, když jsme již provedli výpočet a otestovali hypotézu. Lze demonstrovat, jak by se zachoval náš konkrétní výsledek při různé hladině testu. Ovál s číslem 4 zobrazuje tvar testové statistiky, která má za platnosti nulové hypotézy (tedy v našem případě vždy) studentovo t-rozdělení.

Plocha označená číslem 5 obsahuje parametry jádrového odhadu. Zaškrtávacími políčky si navolíme jednu nebo více hodnot M - počtu použitých testových statistik při aproximaci. Defaultně je vybráno pouze $M = 200$. V následném vybíracím políčku si můžeme zvolit typ jádrové funkce; defaultně máme funkci Gaussovského jádra, které je nejhladší. Šířka vyhlazovacího okna h pro Gaussovské jádro je pomocí metody referenční hustoty pro $M = 200$ defaultně navolena na $h = 0.35$. Metoda je detailněji popsána v [20], str. 19, kde je také odvození použitého optimálního h Gaussovského jádra:

$$h = 1.06\sigma M^{-\frac{1}{5}}$$

, kde σ je směrodatná odchylka testových statistik. Pro $M = 1000$ hodnota

kolísá kolem $h = 0.25$ a pro $M = 30$ musíme pro co nejkvalitnější odhad navolit přibližně $h = 0.6$. Optimalita je měřena z hlediska asymptotické střední integrální kvadratické chyby, o které se lze dočíst v již zmíněné literatuře [20] na str. 16. Pro ostatní typy jader se defaultní hodnota h nemění.

U čísla 6 vidíme 3 tlačítka. První tlačítko slouží k aktualizaci hodnoty n a následnému výpočtu 1200 realizací testových statistik. Druhé tlačítko vykresluje testové statistiky po jedné do grafu a třetím tlačítkem se všechny zakreslené statistiky odstraní a následně se vypočítají nové pro stejnou hodnotu n. Číslem 7 je označeno prostředí, ve kterém vidíme graf zakreslených testových statistik spolu s teoretickou hustotou. Náležitosti grafu popisuje legenda umístěná pod grafem. Pro prvních 5 testových statistik se v pravé části zobrazí jejich hodnota spolu s rozhodnutím o nulové hypotéze. Podíváme-li se do pravého dolního rohu oblasti, uvidíme zapsané hodnoty výběrového průměru a výběrové směrodatné odchylky spolu s kritickým oborem ohraničeným kvantily studentova rozdělení (kritickými hodnotami). V pravém horním rohu sledujeme rozhodnutí o platnosti nulové hypotézy. Křížek značí statisticky signifikantní výsledek (nebo-li že data svědčí proti správné hmotnosti cukru), zatržítko ukazuje statistickou nevýznamnost (data svědčí ve prospěch H_0). Chápeme-li indikátorovu funkci $I_W(T_i)$ jako náhodnou veličinu s alternativním rozdělením s parametrem p, kde T_i je testová statistika pro $i-$tý výběr, můžeme pro N testových statistik psát:

$$\begin{aligned}\mathrm{E}(\hat{\alpha}) &= \mathrm{E}(\sum_{i=1}^{N} \frac{I_W(T_i)}{N}) = \sum_{i=1}^{N} \frac{\mathrm{E}(I_W(T_i))}{N} = \frac{N}{N}p = p = \\ &= \mathrm{P}(\text{zamínutí}H_0|H_0\text{platí}) = \alpha\end{aligned}$$

, kde p je pravděpodobnost, že testová statistika bude v kritickém oboru; či-li pravděpodobnost, že se dopustíme chyby 1. druhu, která je v tomto příkladu vždy rovna hladině α. Vidíme, že $\sum_{i=1}^{N} \frac{\mathrm{E}(I_W(T_i))}{N}$ je nestranným odhadem α. Celý člen $\sum_{i=1}^{N} \frac{I_W(T_i)}{N}$ vlastně můžeme chápat jako výběrový průměr náhodného výběru $I_W(T_1), ..., I_W(T_N)$ pro který víme, že je nestranným a konzistentním odhadem střední hodnoty; v tomto případě p. Indikátorová funkce kritického oboru je borelovsky měřitelná, tudíž jsou i členy transformovaného výběru nezávislé.

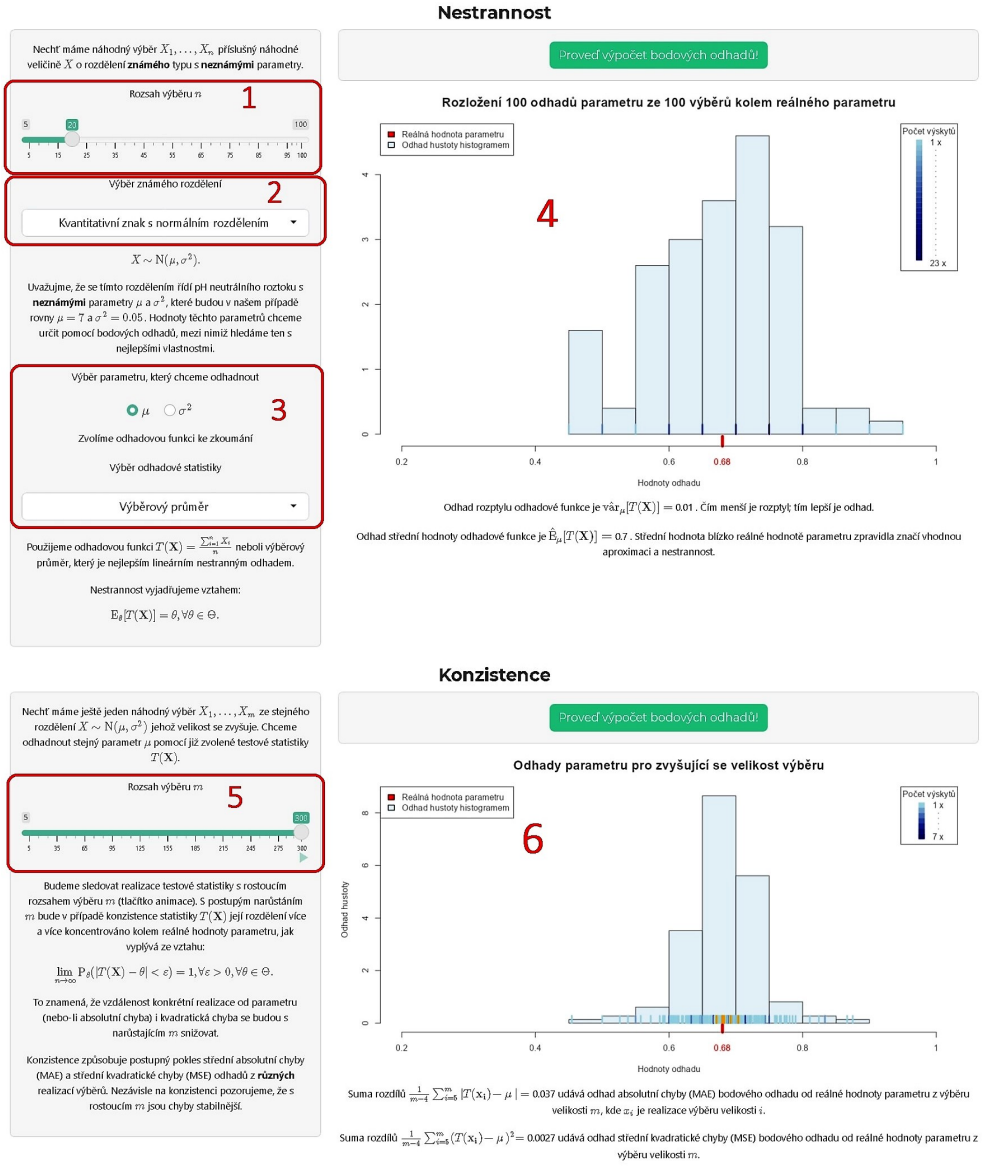
Pod číslem 8 máme vyobrazený graf s příslušnými jádrovými hustotami, které se zobrazí po stisknutí přilehlého tlačítka. Tím aplikujeme navolené parametry jádrové hustoty. Na grafu můžeme pozorovat, jak dobře odhady aproximují teoretické rozdělení. Pod grafem vidíme odhady $\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha}_3$ pro příslušný počet realizací testových statistik. Pro lepší aproximaci teoretické hustoty, bychom měli na příslušných hodnotách sledovat přesnější a stabil-

nější odhady. Nicméně nemusí to tak být pokaždé; velmi záleží také na navolení parametrů jádrové hustoty. Kromě legendy zde nalézáme i vyjádření indikátorové funkce.

3.4 Vlastnosti bodového odhadu parametru

Kromě testování hypotéz je součastí statistické inference i odhad parametrů. Využíváme jich, chceme-li zjistit přibližnou hodnotu parametru; nevyslovujeme žádné rozhodnutí. Od odhadů nemůžeme očekávat bezchybnost, ale velká důvěryhodnost v jejich přesnosti je žádoucí. V následující záložce se zaměříme zvláště na dvě důležité vlastnosti bodových odhadů: nestrannost a konzistenci. Příklady jsou inspirovány [17], str. 15, str. 297 a str. 260.

Při prvním pohledu na záložku, jejíž náhled vidíme na obrázku 13, si všimneme, že je rozdělena na 2 boční a 2 hlavní panely; první 2 slouží k vizualizaci nestrannosti odhadu, druhé dva k vizualizaci konzistence odhadu. Za pomocí vstupu možností (`selectInput`) si můžeme zvolit zadání úlohy, které transformuje vzhled bočního panelu za pomocí podmíněného panelu (`conditionPanel`). Jedná se o úlohy: zkoumání normálně rozděleného pH neutrálního roztoku, zjišťování proporce rodin s HBO kanálem v Hamiltonu, jejichž úspěšné nalezení se řídí alternativním rozdělením a sledování počtu hurikánu na východě USA v průběhu 1 roku, který se řídí poissonovým rozdělením. Veličinu reprezentující příslušná rozdělení označíme X. Ve druhé části přebíráme již navolené zadání z předchozí úlohy a pro konkrétní odhad pozorujeme tvar rozdělení a ukazatele kvality odhadu s rostoucím rozsahem výběru. Výsledky obou bočních panelů můžeme sledovat na hlavních panelech vykreslením 1-dimenzionální přímky s hodnotami realizací bodového odhadu a také histogramů, které aproximují hustotu rozdělení bodového odhadu. Pozorujeme také legendy popisující náležitosti grafu.



Obrázek 13: Náhled záložky o *vlastnostech bodového odhadu*

Číslem 1 znovu začínáme navolením rozsahu výběru n, který při vysokých hodnotách zvyšuje přesnost odhadu. Pro vyšší n bude x-ová osa histogramu více koncentrována kolem reálné hodnoty (odhad bude mít menší rozptyl). Ovál 2 nám nabízí možnost zvolit si příklady, kde odhadujeme konkrétní parametry rozdělení, a dynamicky vypíše příslušný matematický zápis. Bude-li nám to umožněno, vybereme si v sekci 3 parametr, který chceme odhadnout. Kromě toho si zde můžeme vybrat také vhodnou odhadovou funkci. Po

stisknutí tlačítka v prostředí 4 se zobrazí histogram zkonstruovaný pomocí 100 bodových odhadů. Na něm sledujeme v závislosti na zvolené statistice následující pozorování:

1. V případě, že odhadujeme parametr μ normálně rozděleného pH, dostáváme pro statistiku $T(\mathbf{X}) = X_1$ odhad hustoty s těžkými chvosty a s nižšími hodnotami na ose y. To značí vysoký rozptyl, který ukazuje nevhodnost této statistiky, a to i přestože je nestranná. Její konkrétní realizace může být s nezanedbatelnou pravděpodobností daleko od reálné hodnoty.

2. Distribuce odhadu σ^2 (rozptyl pH v neutrálním roztoku) je prakticky k nerozeznání, použijeme-li výběrový rozptyl S_n^2 nebo druhý výběrový centrální moment M_2. Jak už bylo zmíněno, M_2 má dokonce menší střední čtvercovou chybu a tudíž může být preferovaný před nestranným S_n^2.

3. Parametrický prostor parametru p (proporce rodin s HBO) binomického rozdělení obsahuje pouze hodnoty mezi 0 a 1, proto dostáváme (v porovnání s jinými aproximacemi výběrovým průměrem) velmi malé odhady rozptylu, i když jsou některé realizace vzhledem k velikosti intervalu velmi daleko od reálné hodnoty.

4. Aproximace parametru λ (počet hurikánů v USA) pomocí výběrového rozptylu v případě Poissonova rozdělení není vhodná, protože k výpočtu potřebujeme hodnotu výběrového průměru, který je navíc lepším odhadem parametru.

Všimněme si, že ačkoli je histogram často používán pro výčet absolutních četností, zde ho využíváme k aproximaci hustoty, jak jsme nastínili ve vzorci 2 při normování členem $\frac{1}{nh}$. Výpočet počtu sloupců k je přenechaný defaultnímu nastavení R *přibližně* se řídící *Sturgesovým pravidlem*, které vyjadřuje k vzorcem:

$$k = 1 + \lceil 2 \log_2 n \rceil .$$

Více viz [20], str. 12.

V oválu 5 si posuvníkem volíme velikost *jiného* výběru m, který má za úkol dopomoci k vizualizaci konzistence. Pro zvyšující se hodnotu m se po stisknutí tlačítka v sekci 6 zobrazí v grafu pod tlačítkem $m - 4$ bodových odhadů, které jsou realizacemi příslušného (velikostí) odpovídajícího výběru, a dále také odhad hustoty konstruovaný pomocí histogramu. Posuvník umožňuje sledovat animaci, která do grafu zakresluje po 5 nové realizace statistiky pro daná m, která v případě konzistence s rostoucím m konverguje podle pravděpodobnosti k reálné hodnotě parametru. Tomu odpovídá i histogram v pozadí, který se pro velká m přibližuje rozdělení, které se soustřeďuje s velmi vysokou pravděpodobností kolem reálné hodnoty parametru (je špičaté kolem reálné hodnoty parametru). V případě, že odhad není konzistentní (např. již zmíněný $T(\mathbf{X}) = X_1$), tento efekt nenastane. Ve spodní části sekce

můžeme pro různé hodnoty výběru vidět vyčíslení odhadů střední absolutní chyby (MAE) a střední kvadratické chyby (MSE). Znovu si uveďme zajímavá pozorování:

1. Při odhadu μ pomocí $T(\mathbf{X}) = X_1$ se pohybuje odhad MAD okolo stejné hodnoty (0.7-0.8). To je zapříčiněno tím, že odhad parametru není konzistentní. Odhad MSE je při použití této statistiky velmi nestabilní (kvůli povaze umocňování) a nevykazuje známky konvergence.

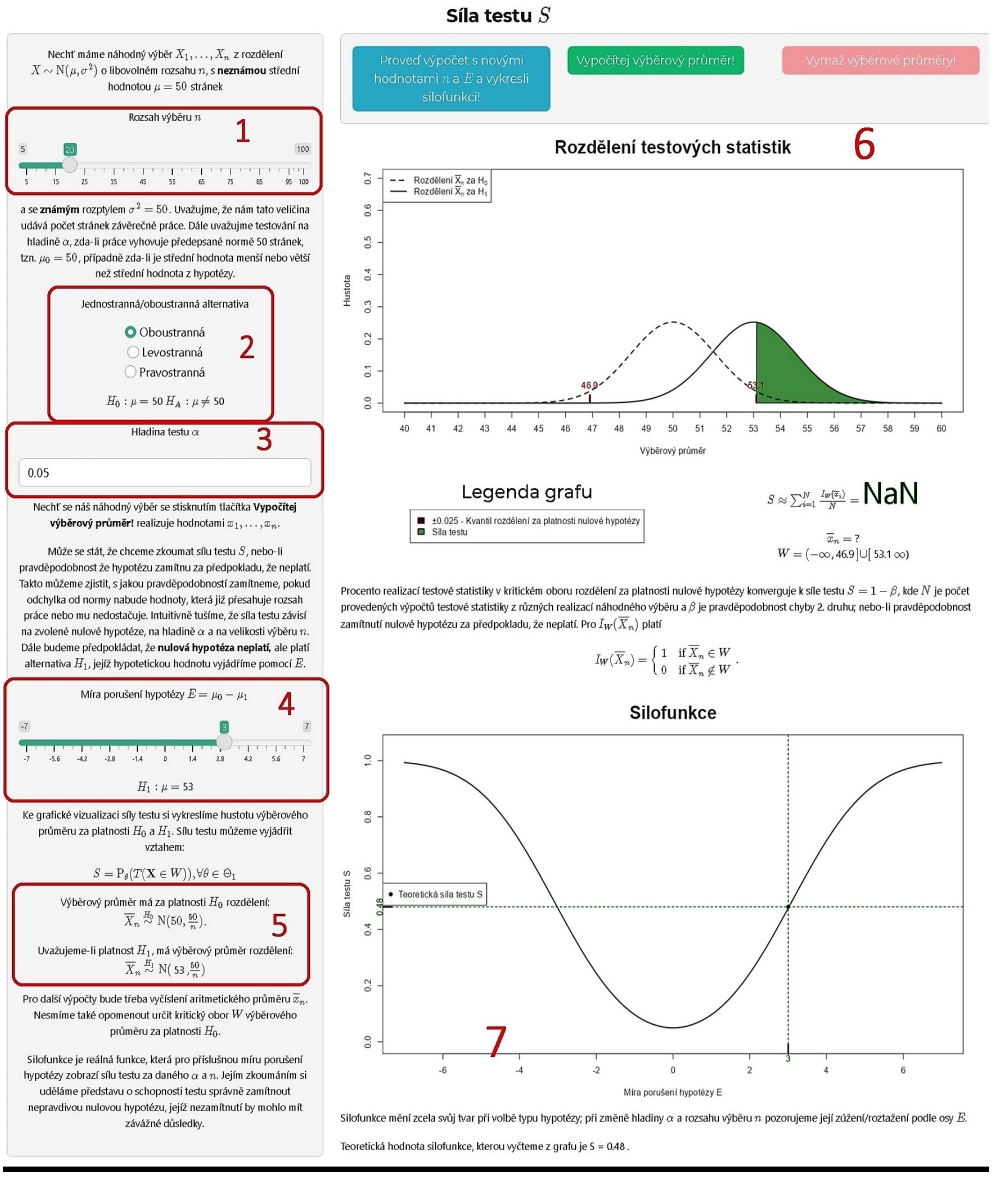
2. Odhadujeme-li pomocí $\overline{X}_m$, S_m^2 či M_2, pozorujeme při vysokém m pokles hodnot, jak odhadu MAE, tak i odhadu MSE, který klesá rychleji.

3. Při odhadování parametru p (proporce rodin s HBO kanálem) je pokles pomalejší než při odhadu jiných parametrů; hodnoty parametrického prostoru $\Theta = (0, 1)$ jsou totiž velmi malé.

3.5 Síla testu

O síle testu jsme se již zmiňovali v 2. kapitole. Víme, že jde o pravděpodobnost správného rozhodnutí, za předpokladu neplatnosti nulové hypotézy (tedy případ, kdy hypotézu zamítáme). Je to také jev opačný k chybě 2. druhu (nastane, pokud nenastane chyba 2. druhu za předpokladu, že neplatí nulová hypotéza). Můžeme ji tedy vypočítat jen pro konkrétní parametr z alternativní hypotézy. Pro jakou hodnotu je vhodné sílu testu počítat? Jedná se o hodnotu, jejíž nezamítnutí by mělo vážné důsledky (zpravidla to jsou kritické hodnoty testové statistiky uvažované za H_0). Vizualizace v následující záložce je postavena na literatuře [17], str. 325, figura 10.3.

Tato záložka má velmi podobnou strukturu, jako pojednání o hladině α. Opakuje se pro naši aplikaci typický vzhled ve formě hlavního a bočního panelu. V bočním panelu jsme seznámeni s úlohou, kde formulujeme hypotézu o střední hodnotě normálně rozdělené veličiny X, která vyjadřuje počet stránek závěrečné bakalářské práce. Máme totiž podezření, že odevzdávání prací neodpovídá příslušnému rozdělení $X \sim \mathrm{N}(50, 50)$. Dále si zde volíme vstupy, jež figurují při testování očekávaného počtu stránek μ za známého rozptylu σ^2, který můžeme chápat jako jakousi stránkovou toleranci. Pomocný text přibližuje hlavní panel s výstupy. V něm podobně jako v případě hladiny α sledujeme grafické zakreslení testových statistik, avšak nyní s pomocí dvou různých rozdělení. Spodní graf vykresluje silofunkci, která udává přehled síly testu, pro míru porušení hypotézy E při navolených testovacích parametrech.



Obrázek 14: Náhled záložky *síla testu*

Na obrázku 14 vidíme náhled celé záložky. Oválem číslo 1 je označena velikost výběru n, se kterou chceme pracovat. Tentokrát budeme uvažovat nejen vliv na odhad střední hodnoty, ale také vliv na rozptyl výběrového průměru, jehož rozdělení najdeme u čísla 5. V části označené číslem 2 pozorujeme formulaci nulové a alternativní hypotézy, která má podobu pouze jednoprvkové množiny. Alternativní hypotéza je v této záložce obzvlášť důležitá. V sekci 3 volíme hladinu testu α omezenou intevalem $(0, 0.3]$, který

je uživatelsky zabezpečený stejně jako v záložce *Hladina alfa*. V sekci 4 volíme míru porušení nulové hypotézy E. Výběr hodnot E je různý vzhledem k typu hypotézy. Pod tímto vstupem máme znázorněnou hypotézu, která je označena $H_1 : \mu = \mu_1$, vyjadřující jednoprvkovou podmnožinou hodnot parametrů z alternativní hypotézy; tedy až na výjimku $\mu_1 = \mu_0$, kdy dostáváme upozornění. Tuto hypotézu dále považujeme za platnou. V praxi volíme např. $E = 5$, když nás zajímá síla testu, pokud by se rozsah bakalářských prací řídil $\mathrm{N}(55, \frac{50}{n})$.

V oválu číslo 5 nalezneme statistiku $\overline{X}_n$ řídící se rozdělením určeným ze vztahu 1, do kterého dosadíme hodnotu $\sigma^2 = 1$ a hypotetické střední hodnoty μ_0, μ_1. Tím si ji připravíme, abychom ji mohli zkoumat za platnosti H_0 a H_1 zvlášť. Kritický obor, který určujeme níže, se používá zpravidla pro známé testové statistiky. Nicméně výběrový průměr můžeme v tomto konkrétním případě také uvažovat jako testovou statistiku, leč při rozhodování o platnosti H_0 je určitě preferována statistika ze vztahu 4, protože vede k normovanému normálnímu rozdělení, které má neměnné hodnoty parametrů, a jeho tvar rozdělení můžeme lehce popisovat.

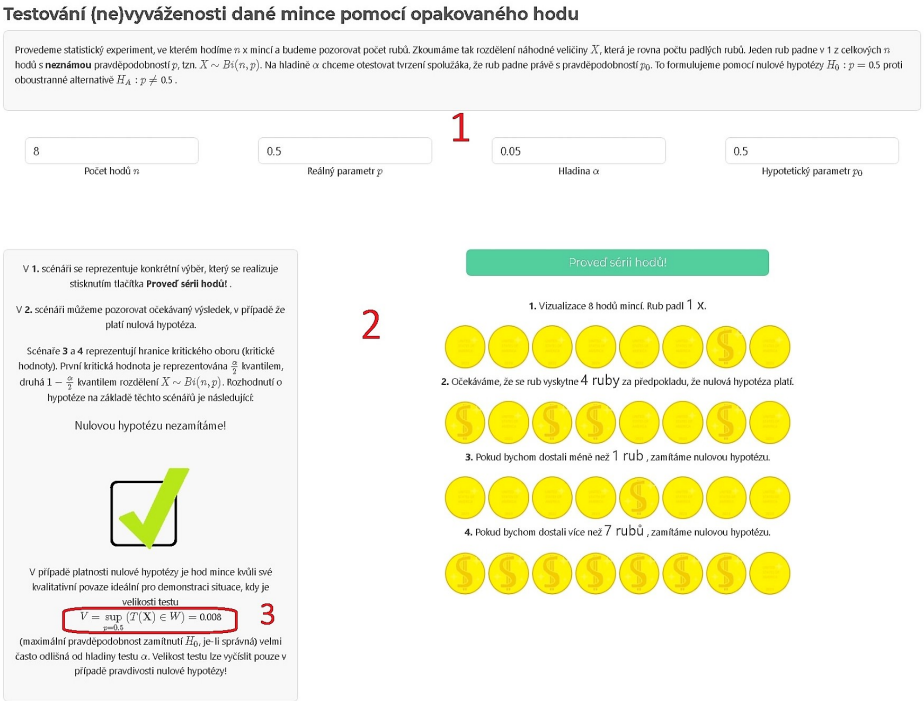
Výstupy v sekci označené číslem 6 nám demonstrují sílu testu. Nejdříve se zaměříme na funkčnost akčních tlačítek, které se nacházejí v horní části sekce. Prvním tlačítkem R zpozoruje a zapíše navolené vstupy (zejména E a n, které už nelze dále měnit) a vygeneruje 250 realizací. Zeleným tlačítkem zakreslujeme hodnoty výběrového průměru a aktualizujeme vstupy, které nemají nechtěné vlivy narušující logiku vykreslování grafu (zde pouze α). Posledním tlačítkem stejně jako v případě záložky o α vymažeme veškeré zakreslené body v grafu a provedeme nový výpočet s danými vstupy. V grafu pozorujeme rozdělení výběrového průměru za platnosti hypotéz. Zakreslujeme testové statistiky a sledujeme aproximaci síly testu pomocí podílu $\sum_{i=1}^{N} \frac{I_W(\overline{x}_n)}{N}$ umístěným pod grafem. Zde znovu vidíme indikátorovou funkci $I_W(\overline{X}_i)$, charakterizovanou alternativním rozdělením nabývající hodnoty 1, pokud výběrový průměr padne do kritického oboru a 0 nestane-li se tak. Vzhledem k tomu, že H_0 neplatí, změní se interpretace podílu, který nyní konverguje s rostoucím n právě k síle testu S. Kromě legendy nalezneme v sekci také vyčíslení kritického oboru a hodnotu aritmetického průměru.

Pod číslem 7 vidíme zobrazenou silofunkci přiřazující míře porušení hypotézy E sílu testu S. Při zvyšování velikosti výběru n se funkce zúžuje (rychleji stoupá) podle E. Minimum náleží střední hodnotě z nulové hypotézy μ_0, kdy se změní interpretace síly testu v případě oboustranné alternativy na velikost testu V. Rychlejší stoupání je také patrné, navolíme-li α příliš vysoko, neboť je známé, že chyba 1. druhu se zvyšuje při poklesu chyby 2. druhu, která je komplementem S. Při volbě jednostranné alternativní hypotézy dostáváme

silofunkci ve tvaru písmene S či jeho zrcadlového obrazu, protože v takovém případě je jedna strana reálné osy čistě *ve prospěch* nulové hypotézy (takže silofunkce už zde znovu nemá interpretaci síly testu). Pod grafem v dolní části sekce můžeme sledovat hodnotu teoretické síly testu pro námi zadané vstupy.

3.6 Hod mincí (velikost testu)

Hod mincí je jednoduchý statistický experiment, ve kterém můžeme dojít pouze ke 2 výsledkům: buď padne rub nebo líc. Jedná se o jednoduchý příklad, na kterém se dobře ilustrují zákonitosti, a to jak teorie pravděpodobnosti tak matematické statistiky (skripta kurzu [2] znázorňují tuto úlohu na straně 172, 173). V následující záložce popíšeme interaktivní úlohu testování (ne)vyváženosti mince. Nejčastěji nás zajímá, zda-li je mince vyvážená; nicméně tento příklad je pojmut demonstrativněji a můžeme se v něm zaobírat libovolně volenou hypotézou. V této úloze se setkáváme s mnoha již zmíněnými statistickými pojmy, jako je např. kritický obor či hladina α. Také zde bude kladen větší důraz na velikost testu V, kterou jsme již zmínili v minulé kapitole.



Obrázek 15: Náhled záložky *hod mincí (velikost testu)*

Na obrázku číslo 15 vidíme náhled záložky, která se skládá ze 3 sekcí označenými červenými čísly 1, 2 a 3. V první sekci vidíme zadání úlohy, spolu s prvními 2 kroky postupu testování statistických hypotéz (stanovení předpokladů, formulace hypotézy, volba hladiny α). Parametry zkoumaného rozdělení, hladinu α a hypotetickou hodnotu p_0 můžeme upravovat pomocí vstupů umístěných v řadě pod textem. Formulace nulové a alternativní hypotézy pak má tento tvar:

$$H_0 : p = p_0 \, H_A : p \neq p_0$$

Při testování hypotézy postupujeme od 3 kroku dále (podle postupu v 2. kapitole) pomocí vizualizací. Nebudeme již používat rozdělení testové statistiky, jako tomu bylo v předchozích příkladech, ale určíme kritický obor jednoduše vizuálním porovnáním našeho výběru. V části označené číslem 2 pozorujeme nejdříve vizualizaci konkrétního výběru, který se realizuje po stisknutí tlačítka *Proveď sérii hodů*. Rozložení rubů a líců je náhodné, abychom lépe vystihli reálnou situaci opakovaného hodu. Druhá skupina mincí ukazuje očekávanou situaci v n hodech s hypotetickou pravděpodobností p_0, kterou nám sdělí spolužák; tedy že padne $\lfloor np_0 \rfloor$ rubů (počet rubů zaokrouhlujeme dolů). Kolem této hodnoty kolísá za platnosti H_0 počet rubů první vizualizace. Rozhodnutí znázorněné výrazným křížkem či zatržítkem v bočním panelu je provedeno na základě 3. a 4. skupiny mincí, jejichž účelem je ohraničit kritický obor za pomocí kritických hodnot. Výběr bude svědčit ve prospěch nulové hypotézy v případě, že jeho hodnota padne do intervalu mezi kritickými hodnotami. V opačném případě nulovou hypotézu zamítáme ve prospěch alternativy. U čísla 3 máme v případě platnosti H_0 dánu velikost testu. Neplatí-li H_0, je hodnota nahrazena křížkem.

Příklad napomáhá porozumět principu testování hypotéz a ukazuje, že i v případě pravdivé hypotézy (které docílíme, navolíme-li $p = p_0$), nemusíme mít 100% jistotu nezamítnutí H_0. Tento typ příkladu také ukazuje, kdy je velikost testu odlišná od α. Je to díky kvalitativní povaze statistického znaku X.

Závěr

V této práci byla za pomocí programovacího jazyka *R* vyvinuta webová aplikace pro podporu výuky statistiky. Za tímto účelem jsme se nejprve zaobírali programovacím jazykem *R* a použitými knihovnami. Dále jsme podnikli zkoumání teorie pravděpodobnosti a matematické statistiky, abychom mohli ve finále rozebrat obsah aplikace (se kterou jsme se mohli setkat prostřednictvím obrázků).

Práci doprovázelo zevrubné studium statistické literatury. Při vývoji se objevovaly problémy s programováním aplikace, jak je to při prvotním seznamování s programovacími jazyky časté. Všechny byly naštěstí dříve či později vyřešeny. Dokonce i výkonnostní problémy, které z počátku často přicházely, byly všechny odstraněny po nahrátí aplikace prostřednictvím *shinyapps.io*, kde se již prakticky nevyskytují.

Literatura [2] základního statistického kurzu pojednává o velkém množství učiva. Jako autor jsem se zaměřil zejména na hlubší porozumění 5 kapitoly, protože jsem ji vnímal jako stěžejní. V literatuře [18] se objevilo mnoho dalších podnětů statistické inference, které by mohly být demonstrovány - např. *suficientní* či *ancilární* statistiky - nicméně zkoumání těchto vlastností už příliš přesahuje jak rozsah práce, tak učivo podporovaného kurzu.

Při teoretickém pojednání byly demonstrovány některé matematické pojmy na jednoduchých příkladech. Aplikace pak má formu spíše teoretického pohledu na příklady z praxe, jejichž zdrojem byla zejména literatura [17]. Při jejich řešení jsme se mohli zamyslet nad skrytými zákonitostmi, které řídí prováděné výpočty. To jsme udělali prostřednictvím dynamického uživatelského prostředí, grafů a obrázků. Podpora kurzu pak je zamýšlena tak, že si studenti sami zkusí tuto aplikaci; případně může být vhodné příklady demonstrovat vyučujícími při probírání dané látky nebo na konci semestru.

Literatura

[1] *Share your apps.* [online], [cit. 2023-04-11], https://shiny.rstudio.com/tutorial/written-tutorial/lesson7/?fbclid=IwAR0CdypnK40_qB1aVmQ6Ue32a61ehkM3eP0_HfIKz2MQmsGiCs2gEk1QjxY

[2] Hron, K., Kunderová, P., Vencálek, O.: *Základy počtu pravděpodobnosti a metod matematické statistiky (4. doplněné vydání).* Univerzita Palackého v Olomouci, Olomouc 2018. Skripta. ISBN: 9788024459905.

[3] R Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Austria, Vienna, 2022. [online], [cit. 2023-04-12] Dostupné z: https://cran.r-project.org/.

[4] *Data Visualization with ggplot2, verze 3.1.0.* RStudio, Inc, 2018. [online], [cit. 2023-02-23], Dostupné z: https://www.maths.usyd.edu.au/u/UG/SM/STAT3022/r/current/Misc/data-visualization-2.1.pdf

[5] Beelen, Ch.: *Web Application Development with R Using Shiny (2. edition).* Packt Publishing. ISBN: 9781785289682.

[6] *Interactive Web Apps with shiny Cheat Sheet 0.12.0.* RStudio, Inc, 2015. [online], [cit. 2023-02-24] Dostupné z: https://shiny.rstudio.com/images/shiny-cheatsheet.pdf

[7] Wickham, H.: *Mastering Shiny: build interactive apps, reports, and dashboards powered by R.* O'Reilly Media, Inc., 2021. [online], [cit. 2023-02-28] Dostupné z: https://mastering-shiny.org/. License: Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. ISBN: 9781492047339.

[8] Grolemund, G., Cheng, J., Cetinkaya-Rundel, M.: *Customize your UI with HTML.* 2017. [online], [cit. 2023-03-03] Dostupné z: https://shiny.rstudio.com/articles/html-tags.html

[9] *Interactive plots - advanced.* 2017. [online], [cit. 2023-03-06] Dostupné z: https://shiny.rstudio.com/articles/plot-interaction-advanced.html

[10] Sievert, C.: *Themes.* 2021. [online], [cit. 2023-03-02] Dostupné z: https://shiny.rstudio.com/articles/themes.html

[11] Coene, J., Kim, J., Granda V.: *Package 'waiter'*. CRAN, 2022. [online] Dostupné z: https://cran.r-project.org/web/packages/waiter/waiter.pdf

[12] *MathJax TeX and LaTeX Support*. The MathJax Consortium Revision, 2018. [online], [cit. 2023-03-09] Dostupné z: https://docs.mathjax.org/en/v2.7-latest/tex.html

[13] Grolemund, G.: *Shiny HTML Tags Glossary*. 2017. [online], [cit. 2023-03-14] Dostupné z: https://shiny.rstudio.com/articles/tag-glossary.html

[14] *Reactivity - An overview*. 2017. [online], [cit. 2023-03-15] Dostupné z: https://shiny.rstudio.com/articles/reactivity-overview.html

[15] Bořil, T.: *Reaktivní funkce a reaktivní proměnné v Shiny*. 2017. [online], [cit. 2023-03-15] Dostupné z: https://fu.ff.cuni.cz/PROG/prog10reaktivni.html

[16] Bates, C.: *23 RStudio Tips, Tricks, and Shortcuts*. Dataquest Labs, Inc., 2020. [online], [cit. 2023-03-01] Dostupné z: https://www.dataquest.io/blog/rstudio-tips-tricks-shortcuts/

[17] Walpole, R., Myers, R., H., Myers, S., L., Ye, K.: *Probability & statistics for engineers & scientists (9. edition)*. Pearson Education, Inc., Boston 2012. ISBN: 9780321629111.

[18] Anděl, J.: *Základy matematické statistiky (3. vydání)*. MATFYZPRESS, Praha 2011. ISBN: 9788073781620.

[19] Hartmann, K., Krois, J., Waske, B.: *E-Learning Project SOGA: Statistics and Geospatial Data Analysis*. Department of Earth Sciences, Freie Universitaet Berlin, 2018. [online], [cit. 2023-03-22] Dostupné z: https://www.geo.fu-berlin.de/en/v/soga-r/Basics-of-statistics/Continous-Random-Variables/Students-t-Distribution/index.html

[20] Dvořáková, A.: *Pokročilé grafické metody v R*. Univerzita Palackého v Olomouci, Olomouc 2022.

[21] Turlach, B., A.: *Bandwidth Selection in Kernel Density Estimation: A Review*. Research Gate, 1999. [online] Dostupné: https://www.researchgate.net/publication/2316108_Bandwidth_Selection_in_Kernel_Density_Estimation_A_Review

[22] Hengartner, N., W., Matzner-Løber, E.: *Asymptotic unbiased density estimators*. EDP Sciences, 2009. [online], [cit. 2023-03-24] Dostupné z: https://www.esaim-ps.org/articles/ps/pdf/2009/01/ps0714.pdf

[23] Meloun, M.: *Testování statistických hypotéz*. Univerzita Pardubice, Pardubice. Slidy. [online], [cit. 2023-03-28] Dostupné z: https://meloun.upce.cz/docs/lecture/chemometrics/slidy/36testy.pdf

[24] Taboga, Marco: *Size of a test, Lectures on probability theory and mathematical statistics*. Kindle Direct Publishing, 2021. Online appendix. [online], [cit. 2023-04-07] Dostupné z: https://www.statlect.com/glossary/size-of-a-test