ČŠI | Česká školní inspekce

Sada typizovaných analytických nástrojů v prostředí standardních statistických programů vycházejících z metodiky

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

2021/2022

ČŠI | Česká školní inspekce

Sada typizovaných analytických nástrojů v prostředí standardních statistických programů vycházejících z metodiky

---

doc. PhDr. Tomáš Lebeda, Ph.D.

Mgr. et Mgr. Jakub Lysek, PhD.

doc. Mgr. Daniel Marek, M.A., Ph.D.

Mgr. Monika Brusenbauch Meislová, Ph.D.

Mgr. Roman Folwarczny

Mgr. Michal Soukop

Mgr. Kateřina Zymová

Mgr. Markéta Zapletalová, PhD.

doc. RNDr. PhDr. Oldřich Hájek, Ph.D., MBA

Bc. Jakub Janega

Bc. Barbora Macková

Bc. Stanislav Daniel

ISBN 978-80-88087-72-4 (online ; pdf)

ISBN 978-80-88087-73-1 (online ; ePub)

OBSAH

1

Úvod

1 ÚVOD

Cílem tohoto dokumentu je zcela konkrétně představit jednotlivé techniky používané pro úpravu a analýzu dat a následně pro vizualizaci analytických výstupů.[1] Smyslem dokumentu je umožnit uživatelům replikaci těchto technik a tím snadno a rychle získat schopnost analyzovat data pomocí prověřených metod. Uživatelům tento výstup umožní přímo implementovat popsané techniky do vlastních analýz na vlastních datech. Tento výstup navazuje na dokument *Metodika sběru a analýzy dat z výsledků šetření ČŠI*, kde jsou obecně popsány jednotlivé analytické postupy. Oba dokumenty jsou tak vůči sobě komplementární a společně dávají uživatelům silnou podporu pro jejich další analytickou práci v oblasti dat z české vzdělávací soustavy. Cílem tohoto dokumentu je shrnout a detailně popsat jednotlivé výzkumné techniky a metody, které využíval výzkumný tým KA5 projektu KSH: *Komplexní systém hodnocení*.

Tento dokument slouží pro realizaci statistických analýz a datových analytických postupů ve statistickém softwaru IBM SPSS a R. IBM SPSS je nejběžnější používaný komerční software ve státní správě a v akademické sféře. Program R je freeware, ale získává si postupně čím dál větší oblibu nejen u odborné veřejnosti, ale i ve státní správě, v médiích a samozřejmě v soukromé sféře. Celou řadu statistických technik běžných pro edukační výzkum SPSS neimplementuje, nicméně v programu RStudio existují specializované balíčky pro analýzu dat. Například oficiální balíček OECD "instvy", který je možné využít při analýze mezinárodních šetření této organizace.[2] Velkou výhodou programu R je jeho síla v oblasti grafické úpravy a vizualizace dat. V dnešní době je především důležité komunikovat složité statistické modely veřejnosti tak, aby i uživatel, který nemá komplexní znalosti statistických metod, pochopil, k čemu analýza dospěla a jaké jsou její hlavní závěry. A to vše přívětivou a vhodnou formou.

Sada typizovaných analytických nástrojů vychází z již provedených pilotáží I až III, ve kterých se ověřovaly jednotlivé statistické postupy a techniky a kde byla ukázána vizualizace a analýza dat. Následující kapitoly tak představují jednotlivé skripty, návody a postupy, jak tyto analýzy provádět. Nejedná se ani tak o učebnici datových analýz, jako spíše o návod, jakým způsobem zpracovat, analyzovat a vizualizovat data pro analýzu vzdělávací soustavy orgány státní správy.

Analytické nástroje jsou členěny do několika částí. V první části seznamujeme uživatele se softwarem, s jeho instalací a základním ovládáním. V druhé části představujeme úpravu datasetu. Před samotnou analýzou musíme data spojit, pročistit a často i rekódovat tak, aby jednotlivé proměnné mohly vůbec vstoupit do analýzy. Samotná úprava dat zabere mnohdy více času než samotná analýza. Po úpravě datasetu zpravidla začínáme deskriptivní jednorozměrnou analýzou, které se dokument věnuje ve třetí části. Čtvrtá část se zabývá mnohorozměrnými metodami analýzy dat, kdy výzkumník zpravidla analyzuje vztahy mezi proměnnými. Pátá část se zabývá technikami, které jsou na pomezí deskriptivní a mnohorozměrné analýzy dat. Cílem je redukce proměnných nebo shlukování případů dle nějakých vlastností. Poslední část se zabývá dílčími otázkami ve statistické analýze, jako jsou chybějící hodnoty a vážení.

Každá část začíná vždy popisem nástrojů v programu SPSS a následně v programu RStudio. Jednotlivé příklady fungují na datových souborech ČŠI, zejména mezinárodních šetřeních PISA 2015, TALIS 2018 a pak dílčích šetřeních provedených ČŠI či MŠMT. Datové soubory sloužily také pro analýzu dat v rámci výstupů Kvalita vzdělávání v jednotlivých krajích ČR[3] a Důležité faktory vzdělávací soustavy v kontextu prostorových dat českých okresů.

[1] Zdrojem vložených obrázků je ČŠI.

[2] Dostupné na: https://www.oecd.org/pisa/data/httpoecdorgpisadatabase-instructions.htm

[3] Dostupné na: https://www.csicr.cz/cz/Aktuality/Kvalita-vzdelavani-v-jednotlivych-krajich-CR

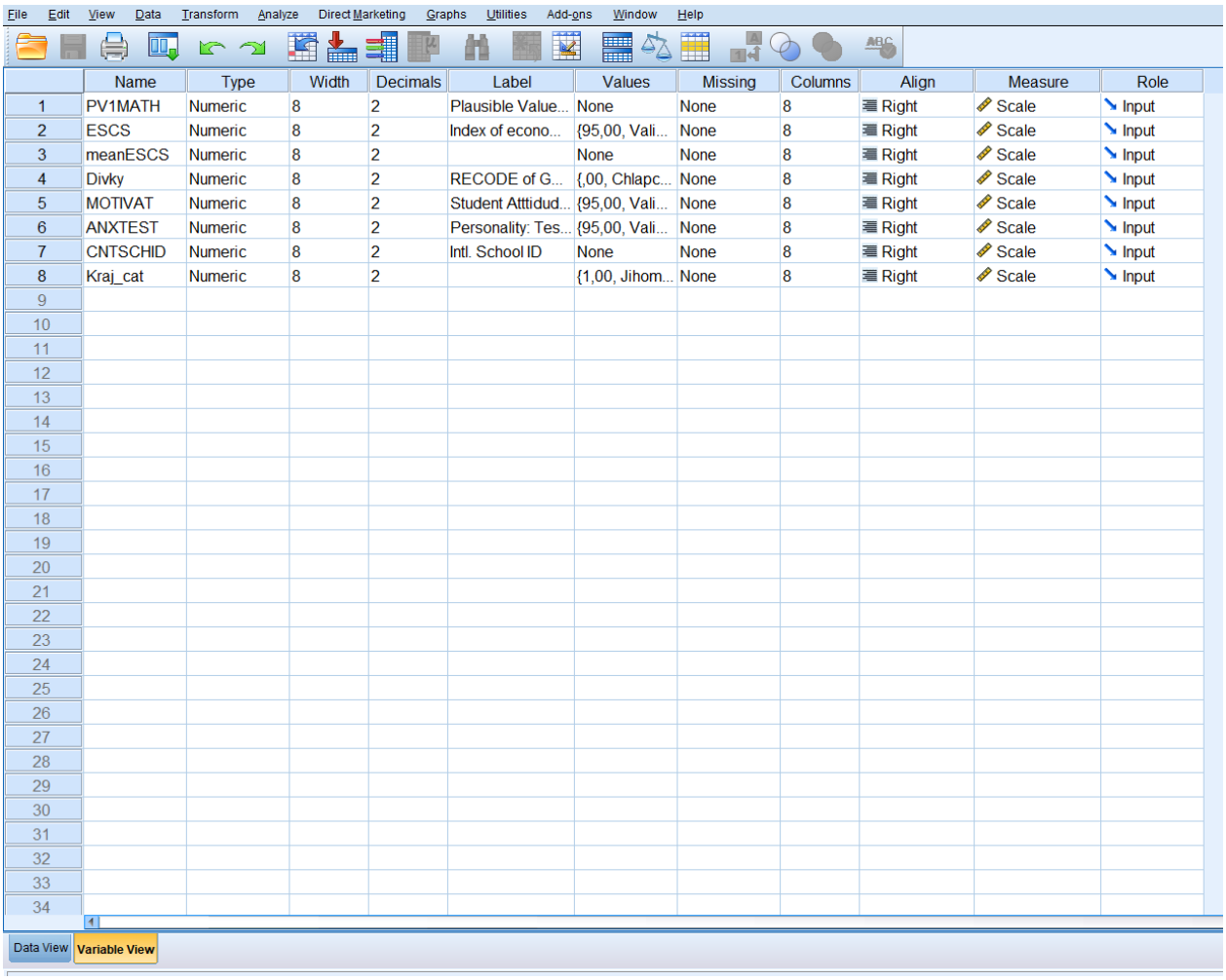
2

Seznámení s programem SPSS

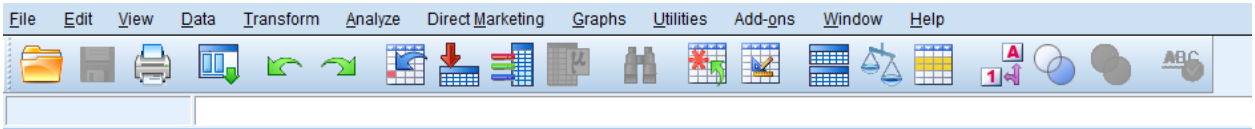
2 SEZNÁMENÍ S PROGRAMEM SPSS

IBM SPSS (dále jen "SPSS") je komerční statistický software pro základní i pokročilou statistickou analýzu dat, který poskytuje velice širokou paletu různých nástrojů. Významnou výhodou při použití programu SPSS je relativní uživatelská přívětivost a absence nutné znalosti skriptovacího jazyka programu. Všechny statistické operace je možné provádět pomocí propracovaného intuitivního interface a dialogových oken, včetně využití sofistikovanějších statistických metod.

Základními prostředími programu jsou tzv. *Variable View*, které umožňuje celkový pohled na importovaný datový soubor z hlediska proměnných, jejich popisu, kódování, údajů o chybějících hodnotách apod., a tzv. *Data View* zobrazující datový soubor pomocí proměnných a jednotlivých případů k nim náležejících.



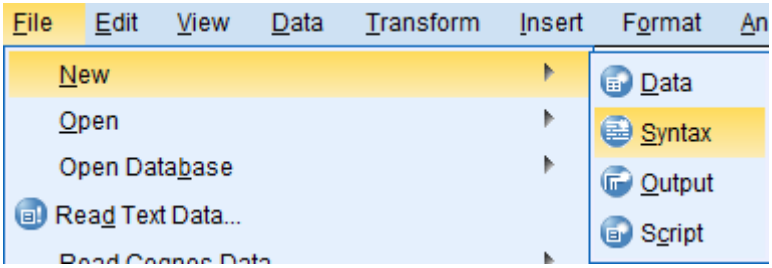
	Name	Type	Width	Decimals	Label	Values	Missing	Columns	Align	Measure	Role
1	PV1MATH	Numeric	8	2	Plausible Value...	None	None	8	Right	Scale	Input
2	ESCS	Numeric	8	2	Index of econo...	{95,00, Vali...	None	8	Right	Scale	Input
3	meanESCS	Numeric	8	2		None	None	8	Right	Scale	Input
4	Divky	Numeric	8	2	RECODE of G...	{,00, Chlapc...	None	8	Right	Scale	Input
5	MOTIVAT	Numeric	8	2	Student Atttidud...	{95,00, Vali...	None	8	Right	Scale	Input
6	ANXTEST	Numeric	8	2	Personality: Tes...	{95,00, Vali...	None	8	Right	Scale	Input
7	CNTSCHID	Numeric	8	2	Intl. School ID	None	None	8	Right	Scale	Input
8	Kraj_cat	Numeric	8	2		{1,00, Jihom...	None	8	Right	Scale	Input



	PV1MATH	ESCS	meanESCS	Divky	MOTIVAT	ANXTEST	CNTSCHID	Kraj_cat	var
1	401,72	-,27	-,01	1,00	-1,42	-,21	20300001,0	8,00	
2	577,35	,06	-,01	1,00	-,29	1,44	20300001,0	8,00	
3	566,81	-,31	-,01	,00	-,32	,37	20300001,0	8,00	
4	522,76	-,94	-,01	1,00	-,65	-2,51	20300001,0	8,00	
5	587,74	-,15	-,01	,00	-,17	-,31	20300001,0	8,00	
6	634,47	1,11	-,01	1,00	-,17	-,53	20300001,0	8,00	
7	508,43	-,92	-,01	1,00	-,48	1,18	20300001,0	8,00	
8	441,45	1,06	-,01	1,00	-,48	1,06	20300001,0	8,00	
9	581,47	,13	-,01	1,00	,54	-,95	20300001,0	8,00	
10	506,63	-,09	-,01	1,00	,55	-,05	20300001,0	8,00	
11	553,12	,98	-,01	,00	-,09	,57	20300001,0	8,00	
12	433,38	-,83	-,01	1,00	-,48	1,61	20300001,0	8,00	
13	540,91	-,42	-,01	,00	-1,45	,14	20300001,0	8,00	
14	622,02	-,51	-,01	,00	,66	,71	20300001,0	8,00	
15	511,59	-,38	-,01	,00	,23	-,20	20300001,0	8,00	
16	483,71	,87	-,01	,00	,96	1,80	20300001,0	8,00	
17	551,32	,63	-,01	1,00	1,85	-,03	20300001,0	8,00	
18	496,72	,47	-,01	1,00	-,88	,38	20300001,0	8,00	
19	537,85	,36	-,01	1,00	-1,27	-,68	20300001,0	8,00	
20	548,41	-,60	-,01	1,00	-,33	-,69	20300001,0	8,00	

Třetí využívané okno v prostředí SPSS je tzv. *Output* (okno se otevírá vždy se spuštěním programu), který zobrazuje jednak provedené syntaxe, navolené buď příkazem, nebo skrze dialogová okna, jednak uvádí výstupy jednotlivých statistických analýz.

Pro pokročilejší uživatele SPSS je k dispozici vstup pro zadávání syntaxe ve vlastním jazyce. Dle výrobce novější verze programu umí pracovat též s programovacím jazykem python. Zadávání, kontrola a spouštění syntaxe funguje na obdobném principu jako jiné analytické programy (STATA, R aj.). Využití zadávání syntaxe je vhodné zejména při opakujících se analýzách, které vyžadují minimální úpravy některé z proměnných. V takovém případě je zadávání pomocí dialogových oken zbytečně zdlouhavé. Zadání syntaxe umožňuje též spouštění analýz vždy se stejnými parametry, které jsou snadněji kontrolovány než pomocí neustále se resetujících dialogových oken. Nutno podotknout, že znalost syntaktických příkazů může práci s SPSS výrazně zefektivnit, nezbytná pro práci s programem ovšem není.



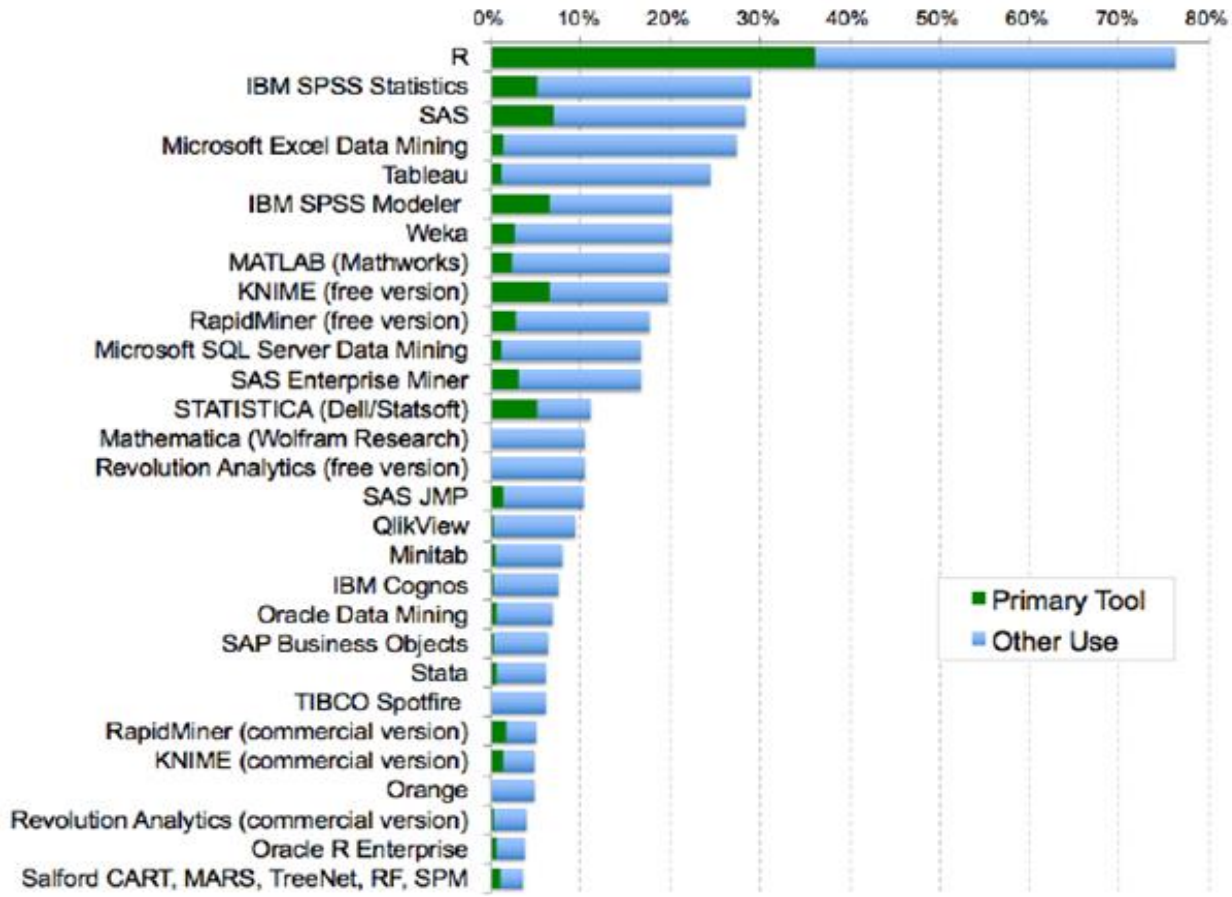
Využití SPSS pro účely ČŠI lze nalézt zejména v oblasti přípravy dat, jejich spojování, rekódování, tvorby nových proměnných či základní deskriptivní analýzy. Své uplatnění nalezne též jako nosná platforma pro využití IEA vydávaného *IDB analyzeru*. Provádění složitějších analýz je vhodné směřovat spíše do jiných programů.

3

Seznámení s programy R a RStudio

3 SEZNÁMENÍ S PROGRAMY R A RSTUDIO

R je programovací jazyk a prostředí pro výpočty zejména statistické povahy a zpracování dat různé povahy či rozsahu. Mimo samotných výpočtů poskytuje program také širokou paletu nástrojů určených k vizualizaci dat. Jedná se o volně šiřitelný software, který je k dispozici pro Linux, Windows, Mac, Android a další operační systémy. Právě jeho dostupnost je následně také jedním z důvodů jeho rozšíření a popularity mezi odbornou komunitou. Online výzkum mezi odborníky na statistiku z roku 2015 stavěl program R do pozice nejpoužívanějšího programu na zpracování dat (viz graf níže). Široká uživatelská komunita se následně projevuje také ve snadné dostupnosti kvalitních materiálů, jako jsou knihy, návody na webových stránkách (např. Stackoverflow.com) či video tutoriály, kde mohou uživatelé nalézt řadu inovativních postupů a rad.



Základní program R funguje na příkazovém řádku. V případě operačního systému Windows má k dispozici jednoduché grafické rozhraní. Pro uživatelsky přívětivější přístup je proto vhodné využít zvláštního vývojového prostředí. V současné době je pro program R nejčastěji využívaným prostředím RStudio. Tento software je obdobně jako základní výpočetní program R volně dostupný na internetu.

3.1 Instalace programů R a RStudio

Prvním krokem, který je před využíváním programu R nutné učinit, je jeho instalace. Jak bylo již jednou zmíněno výše, veškeré potřebné softwary jsou dostupné online. Nejprve je tedy nutné začít s instalací základního programu R. Ten je možné dohledat na těchto webových stránkách:

https://cran.r-project.org/

Webové stránky jsou pouze v anglickém jazyce, i přesto jsou však uživatelsky přehledné. Pro instalaci je nejprve třeba zvolit z nabídky správnou verzi programu, jež by měla být shodná s operačním programem, který má uživatel nainstalovaný na svém zařízení. V případě tohoto instruktážního dokumentu bude pracováno s operačním programem Windows. Proto v tomto případě na výše uvedených webových stránkách bude zvolen odkaz *Download R for Windows*.

Odkaz uživatele přesměruje na novou stránku. Na té by měl uživatel následně kliknout na možnost *Install R for the first time*. Tento odkaz následně přivede uživatele na již poslední stránku, kde nalezne stahovací odkaz na

nejaktuálnější verzi programu R. Stačí pouze kliknout na odkaz *Download R 4.1.1 for Windows* a mělo by dojít k automatickému stažení instalačního balíčku programu. Po úspěšném stažení je třeba spustit instalační balíček a provést instalaci programu R. Po úspěšné instalaci je možné začít využívat program R.

Pro přístupnější a pohodlnější práci s programem R je však nutné stáhnout také druhý program, kterým je RStudio. Ten uživateli nabízí přehledné pracovní prostředí, v jehož rámci může maximálně využít všech funkcí, které program R a jeho rozšíření v podobě programových balíčků nabízí. Instalační soubor s programem RStudio je možné nalézt na těchto stránkách:

https://www.rstudio.com/products/rstudio/download/

Zde je následně možné zvolit ke stažení bezplatnou verzi programu RStudio. Konkrétně stačí pod nabídkou *RStudio Desktop (Open Source License)* kliknout na tlačítko *Download*. Následně bude uživatel odkázán na stahovací odkaz níže na stránce. Zde již postačuje kliknout na stahovací odkaz *Download RStudio for Windows* a mělo by dojít k automatickému stažení instalačního balíčku. Po úspěšném stažení dokumentu je třeba následně program nainstalovat. Po instalaci obou výše zmíněných programů již postačuje vždy spustit pouze RStudio.

3.2 Představení prostředí programu RStudio

Zásadní rozdíl oproti jiným statistickým softwarům je ten, že v prostředí R se píší příkazy neboli skripty, pomocí kterých spouštíme jednotlivé analytické funkce a statistické analýzy. Na jednu stranu se může zdát psaní skriptů uživatelsky náročné, ale po pochopení základní logiky psaní skriptu datová analýza pomocí programu RStudio je velmi rychlá a šetří mnoho práce oproti jiným způsobům analýzy, kde bychom museli tzv. klikat. Počáteční investice do naučení se psaní skriptů se tak bohatě vyplatí. Není přitom nutné mít znalosti programování, přestože jazyk R umožňuje programování vlastních funkcí, které mohou usnadnit práci.

Před zahájením práce s programem je vhodné krátce představit nabídku horní lišty. Jedná se o klasickou rozbalovací nabídku jednotlivých funkcí. Mezi ty nejdůležitější patří nabídka *File*. V ní uživatel nalezne například možnost otevření nového skriptu či možnost uložení. Nabídka *Edit* nabízí klasické nástroje pro vrácení či obnovení, které je ovšem možné nahradit také klasickými klávesovými zkratkami Ctrl+Z v případě vrácení a Crtl+Shift+Z pro navrácení změny. Nabídka *View* poskytuje uživateli možnost zvětšit či zmenšit zobrazení obsahu programu či zobrazení vybraných nabídek. Poměrně podstatnou je také nabídka *Session*. Ta mimo jiné nabízí možnost nastavení pracovní složky (konkrétně funkce *Set Working Directory*), této problematice se bude dále věnovat sekce níže. Zejména v případě nastalých problémů je nakonec podstatnou také nabídka *Help*, s jejíž pomocí může uživatel dohledat řadu základních informací o fungování programu a jeho jednotlivých rozšířeních.

Po spuštění programu RStudio by se měla uživateli na obrazovce zobrazit tři okna. Ještě před zahájením práce je vhodné otevřít si v programu také zdrojové okno, do kterého bude následně vpisován kód s jednotlivými příkazy pro různé výpočty, úpravu dat atd. Pro otevření tohoto okna je třeba rozkliknout nabídku *File*, najet na možnost *New file* a z nabídky zvolit možnost *R Script*. Tento krok může být případně nahrazen klávesovou zkratkou Crtl+Shift+N. V tomto okamžiku by se měla v rámci programu zobrazit čtyři okna.

V levém horním rohu se zobrazuje tzv. zdrojové okno, též označované jako skript. Do tohoto okna by měl uživatel psát svůj kód, tedy jednotlivé výpočetní příkazy, které chce po programu vykonat. Na tomto místě je vhodné podotknout, že při práci s programem RStudio je při psaní kódu vždy třeba pamatovat na dodržení velkých a malých písmen. V případech některých funkcí totiž při nedodržení správné diakritiky nemusí program kód přijmout a nahlásí chybu.

V horní části tohoto okna jsou zobrazeny záložky všech skriptů, které jsou v rámci programu zrovna otevřeny. Uživatel mezi jednotlivými skripty může překlikávat či je pomocí křížku zavřít. Je vhodné poznamenat, že v případě, kdy nejsou změny v rámci skriptu uloženy, název skriptu zobrazený na dané záložce zčervená a na jeho konci se objeví hvězdička. Po uložení text na záložce zčerná. Nejpodstatnější funkci nalezneme v pravém horním rohu tohoto okna. Jedná se o příkaz *Run*, respektive bílý obdélník se zelenou šipkou a nápisem Run. Tato funkce slouží k provedení výpočtu příkazů, které uživatel vepsal do svého skriptu. Kliknutí na tlačítko *Run* je možné nahradit také klávesovou zkratkou Ctrl+Enter.

V pravém horním rohu se zobrazuje okno s pracovním prostředím a historií. Zde uživatel nalezne veškeré datové soubory, které do programu nahrál a aktuálně s nimi pracuje. Zároveň zde, v záložce *History*, nalezne výpis všech dosavadních příkazů, které zadal programu ke zpracování.

V levém dolním rohu nalezne uživatel další podstatné okno, též označované jako konzole. V této části dochází ke zpracování a vyhodnocení příkazů, které uživatel vepsal do svého skriptu. Jinými slovy je zde možné nalézt výsledky jednotlivých příkazů či chybová hlášení, které upozorňují uživatele na možnou chybu v kódu. Je vhodné zmínit, že při ukládání skriptu dojde vždy pouze k uložení informací, které jsou uvedeny ve zdrojovém okně. Údaje z konzole se neukládají.

Pokud by přece jen uživatel chtěl některé údaje, například výsledek matematické rovnice, zachovat společně s informacemi ve zdrojovém okně, je možné tyto údaje vykopírovat do zdrojového okna. V takovém případě je ovšem vhodné před tyto údaje vložit znak mřížky, též známý jako hashtag: #. Pokud tento znak ve zdrojovém okně vložíme před daný řádek textu, program RStudio tento řádek nezprocesuje a zbarví ho zeleně. Pomocí # je tak možné do zdrojového okna vkládat například komentáře, a to bez toho, aniž by byla narušena struktura kódu.

V pravém dolním rohu se nachází okno s několika záložkami a řadou funkcí. Toto okno nabízí možnost procházení adresáře počítače, jsou v něm zobrazovány výsledné grafy a další grafické výstupy. Dále okno slouží také ke správě doplňkových balíčků, které obsahují rozšiřující funkce.

3.3 Set working directory

Před zahájením práce s programem RStudio je vhodné vytvořit na některém z disků v počítači jednu základní složku. Do této složky by měly být nahrány a následně také ukládány veškeré dokumenty a datové soubory, které budou v rámci RStudia zpracovávány.

Na tuto složku či některou z jejich podsložek je následně nutné navázat pracovní prostředí RStudia. Navázání probíhá za pomoci funkce *Set working directory*. Tu nalezneme na horní liště v nabídce *Session*. Z nabídky *Set working directory* je následně nejvhodnější zvolit možnost *Choose directory…* (možné nahradit klávesovou zkratkou Ctrl+Shift+H). Po zobrazení prohlížeče je třeba s jeho pomocí v počítači nalézt požadovanou složku, ve které by měly být již uloženy datasety, jež budou v programu RStudio zpracovávány. Po nalezení složky je třeba její výběr potvrdit kliknutím na tlačítko *Open*.

Po úspěšném napojení by se v konzole mělo objevit umístění dané složky, a to v této podobě:

```
setwd("C:/Users/JanNovak/zakladnislozka/prvnilekce")
```

Je vhodné tento údaj následně zkopírovat do zdrojového okna, ideálně na začátek skriptu. Usnadní se tím následná práce, neboť nebude nutné opakovaně dohledávat umístění dat, která jsou v rámci daného skriptu zpracovávána.

3.4 Základní práce se skripty

Analýzy zahajujeme psaním skriptu v levém horním okně. Na začátek je dobré skript popsat, napsat název skriptu, který si zvolíme, vypsat datové soubory, které skript používá, autora skriptu a datum. Rovněž si nastavíme pracovní prostředí.

```
#####################################################
#                      PISA 2015                    #
#                   Sekundární analýza              #
#####################################################
# Datum: 23.1.2021
# Autor: E-mail autora
# Dataset: PISA 2015 OECD
setwd(C:/Users/JanNovak/PISA2015")
```

Při psaní skriptu platí, že pokud napíšeme hashtag před textem, RStudio bere text jako text a nepovažuje jej za příkaz. Takto si můžeme popsat skript užitečnými poznámkami a můžeme skript členit na několik částí, například na část s úpravou datasetu a rekódováním, po kterých následuje analýza. Text je označen zelenou barvou, příkazy černou a modrou. Pro účely tohoto dokumentu označujeme zeleně poznámky spojené se znakem #, modrou barvou pak zbylý text příkazu.

Ve skriptu se používají další znaky pro psaní příkazů:

- Dolary **$** (ctrl + alt + ů).
- Pipe operator **%>%** (ctrl + shift + m)
- < = (alt + 60) či (alt + ctrl + ,)
- & = ctrl+ alt + c

Většina příkazů je na jeden řádek, některé příkazy jsou ale delší. Například v případě rekódování proměnných či u regresního modelu nebo jiných analýz. Pokud chceme skript rozdělit na několik řádků, znaky oddělující skript jako "," a "+" musí končit řádek.

3.5 Ukládání a Reopen with encoding (UTF-8)

Při práci s programem RStudio je vhodné vždy pamatovat na to, že se jedná o anglický kódovací jazyk. Při pojmenovávání datasetů a práci v programu je proto vhodné spíše nevyužívat českou diakritiku. Pokud je však čeština nutná, například při pojmenovávání grafů nebo psaní komentářů, je vhodné při ukládání skriptu zvolit kódování znaků UTF-8. RStudio totiž automaticky ukládá s kódováním CP1250, které nerozpozná českou diakritiku. Pokud uživatel takovýto dokument otevře, namísto českých znaků se mu ve skriptu zobrazí červené tečky. V tomto případě je třeba využít funkci *Reopen with encoding*. Tu je možné nalézt v nabídce *File* na horní liště. Z nabídky znakových kódů je následně potřeba zvolit UTF-8. Po potvrzení by se měl skript automaticky znovu otevřít s již doplněnou českou diakritikou.

3.6 Programové balíčky

Program R nabízí v základu pouze omezené množství základních statistických funkcí. V rámci rozšíření a maximálního využití potenciálu, který program R nabízí, je proto nutné nainstalovat doplňkové programové balíčky. Každý z těchto balíčků obsahuje nástroje pro specifický typ analýz či úkolů, které uživatel potřebuje provést.

K instalaci nových balíčků slouží integrovaná funkce programu RStudio, není tedy nutné balíčky dohledávat na internetu a následně je ručně instalovat do počítače. Pro názornost bude v tomto případě představena instalace balíčku *ggplot2*, který je využíván k tvorbě grafů a vizualizaci dat.

K samotné instalaci balíčků následně slouží záložka *Packages* v pravém dolním okně. Po kliknutí na tuto záložku je třeba kliknout na možnost *Install* s vyobrazením malé modré krabice s šipkou. Následně by mělo dojít k otevření instalačního okna. Toto okno obsahuje celkem tři řádky. První z nich udává, odkud budou balíčky instalovány. V tomto případě je vhodné ponechat možnost *Repository (CRAN)*. Balíček se tak stáhne z centrálního sdíleného úložiště spravovaného samotnými vývojáři programu R, bude tedy zajištěna jeho aktuálnost a funkčnost. Druhý řádek slouží k vypsání požadovaných balíčků. Zde je tedy možné uvést již jednou výše zmíněný balíček *ggplot2*. V případě potřeby je možné do okna vepsat i více balíčků ke stažení. Jednotlivé názvy by měly však být odděleny mezerou nebo čárkou. Poslední řádek uvádí umístění, respektive složku, do které bude balíček stažen. Umístění je přednastaveno automaticky a není možné ho již změnit. Po vypsání požadovaného balíčku stačí pouze stisknout možnost *Install* v instalačním okně.

Po potvrzení instalace začne program R automaticky stahovat požadovaný balíček. Následuje instalace balíčku a jeho začlenění do nabídky programu. Celý proces je možné sledovat v konzole. Po dobu, kdy program balíček instaluje, se v pravém horním rohu konzole bude zobrazovat malé červené tlačítko *STOP*. To indikuje, že program zpracovává zadaný příkaz a zároveň je s jeho pomocí možné probíhající zpracovávání zastavit.

Po úspěšném nainstalování balíčku by se v konzole mělo zobrazit červené oznámení s umístěním staženého balíčku (*The downloaded source packages are in*). Nainstalování balíčku je možné zkontrolovat také v pravém dolním okně v záložce *Packages*. Ta obsahuje seznam všech stažených balíčků. Pokud je zde balíček dohledatelný, je stažen a připraven k aktivaci a následnému používání.

Po úspěšném stažení je nutné požadovaný balíček ještě aktivovat. Aktivaci lze provést zaškrtnutím políčka, které se nachází před názvem balíčku v již zmíněném seznamu balíčků v záložce *Packages*.

Celý výše popsaný proces instalace a aktivace balíčků je možné provést také s pomocí příkazů, které jsou vpisovány do zdrojového okna. Pro instalaci požadovaného balíčku je možné využít příkaz: install.packages(""). Mezi uvozovky v závorce je nutné vepsat požadovaný balíček. V případě výše zmíněného balíčku *ggplot2* by tedy výsledný kód vypadal takto: install.packages("ggplot2"). Příkaz je následně nutné spustit pomocí výše zmíněného tlačítka *Run*.

Pro následnou aktivaci balíčku je možné využít příkaz: library(). Také v tomto případě postačuje do závorek, nyní již bez potřeby uvozovek, vepsat název nainstalovaného balíčku a příkaz spustit. Úspěšné aktivování balíčku je možné opět zkontrolovat v seznamu záložky *Packages*, kde by mělo být zaškrtnuto políčko vedle názvu požadovaného balíčku.

Na závěr je vhodné také představit základní rozšiřující balíčky. Pro práci s regresními modely je možné doporučit balíček *car*, který vytvořil statistik John Fox. Pro významnější rozšíření funkcí programu R je vhodné nainstalovat balíček *tidyverse*. Ten obsahuje již jednou zmíněný balíček *ggplot2* na tvorbu grafů, dále balíček *dplyr* na úpravu dat, balíček *tidyr* na čištění dat a mnoho dalšího.

V případě, že chceme zkontrolovat aktuální verze balíčku pro daný skript, můžeme použít tuto naprogramovanou funkci. V ní stačí jen vypsat názvy potřebných balíčků, které budeme používat.

```
check.packages <- function(pkg){
        new.pkg <- pkg[!(pkg %in% installed.packages()[, "Package"])]
        if (length(new.pkg))
        install.packages(new.pkg, dependencies = TRUE,
        repos = "https://cran.rstudio.com")
        sapply(pkg, require, character.only = TRUE)
        }
        pkgList <- c("tidyverse","broom","ggeffects","rstudioapi",
        "texreg","arm","broom.mixed","ggthemes","stringr",
                "ICC", "lme4") #zde vypsat názvy balíčků
        check.packages(pkgList)
```

3.7 Příkaz Help

V případě jakýchkoliv problémů je možné využít integrované nápovědy. Tu je možné nalézt na horní liště v nabídce *Help* nebo v okně v pravém dolním rohu. V nápovědě je možné nalézt podrobný popis programových balíčků, funkcí, které zahrnují a také základy kódů, s jejichž pomocí je možné dané funkce provést. Mimo to RStudio nabízí také možnost nápovědy pomocí příkazu help(). Tímto způsobem se uživatel může dotázat na konkrétní funkci či balíček, jehož využitím nebo funkcemi si není jistý.

V případě potřeby dohledání informací o konkrétní funkci stačí do příkazu vepsat název požadované funkce. Například v případě hledání informací o funkci pro lineární regresi by kód vepsaný do zdrojového okna vypadal takto: help(lm).

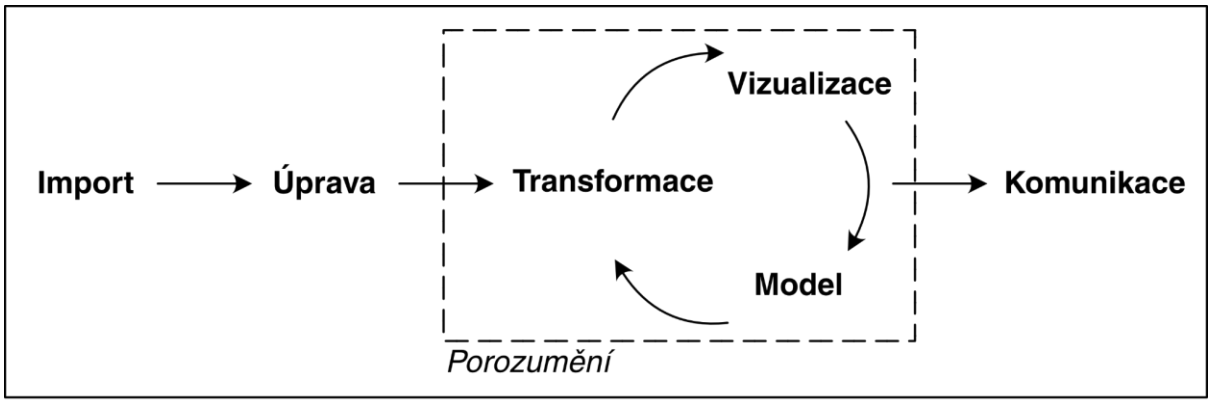
Po kliknutí na tlačítko *Run* by se v okně v pravém dolním rohu měla následně zobrazit dokumentace s požadovanými informacemi. Mimo to je možné dohledávat informace o celých programových balíčcích. V takovém případě by například při hledání dokumentace pro balíček *haven* měl příkaz tuto podobu: help(package="haven").

4

Úprava dat

4 ÚPRAVA DAT

Tato sekce se zabývá úpravou dat. Import, úprava a transformace dat je jedním z nejdůležitějších analytických kroků před samotnou analýzou. Nejedná se však o samostatnou část datové analýzy, úprava dat je nutně spojena se samotným modelováním a vizualizací dat, kdy zpětně zjistíme, že je nutné data dále upravit či transformovat. Tento proces ukazuje následující schéma:

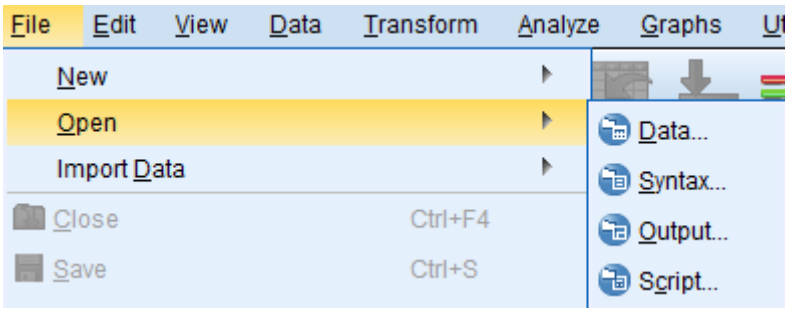


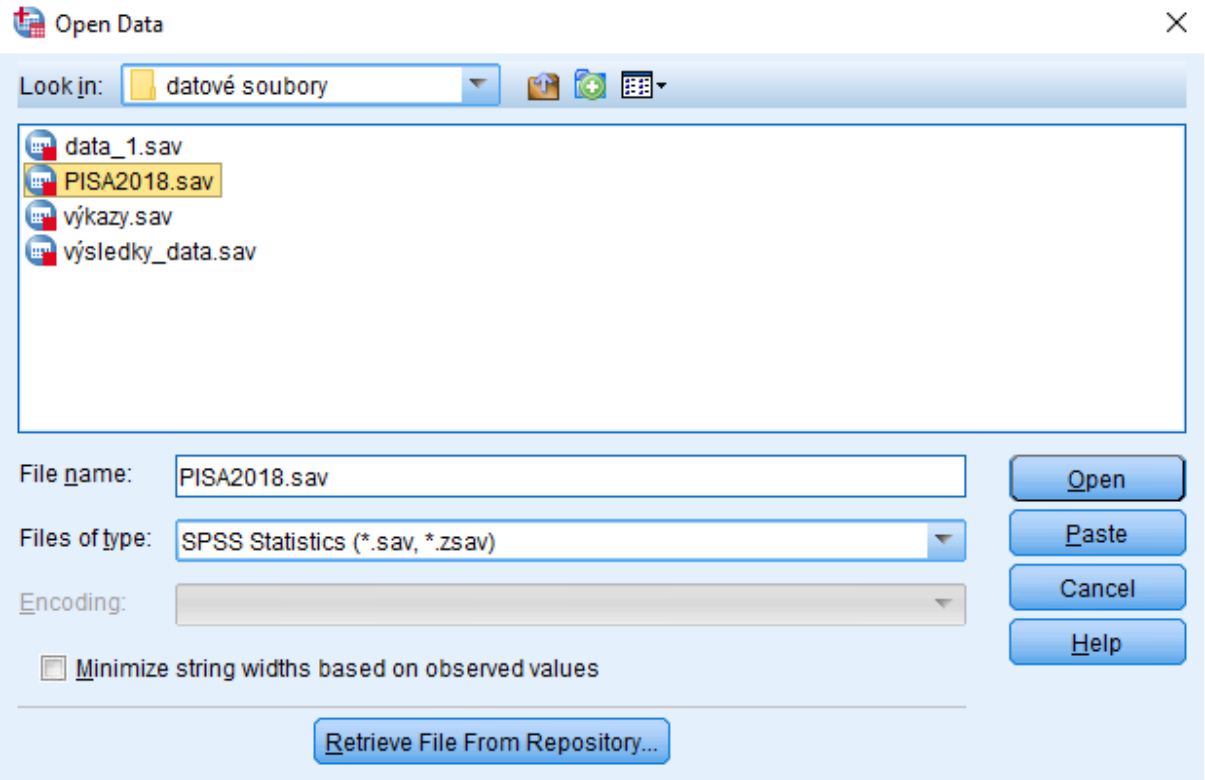
Nejdříve je představena úprava dat v programu SPSS, následně stejné či pokročilejší funkce úpravy dat jsou představeny v programu R.

4.1 Úprava dat v SPSS

4.1.1 Nahrávání datového souboru

Nahrát datový soubor do SPSS je možné po rozkliknutí *File* → *Open* → *Data*.

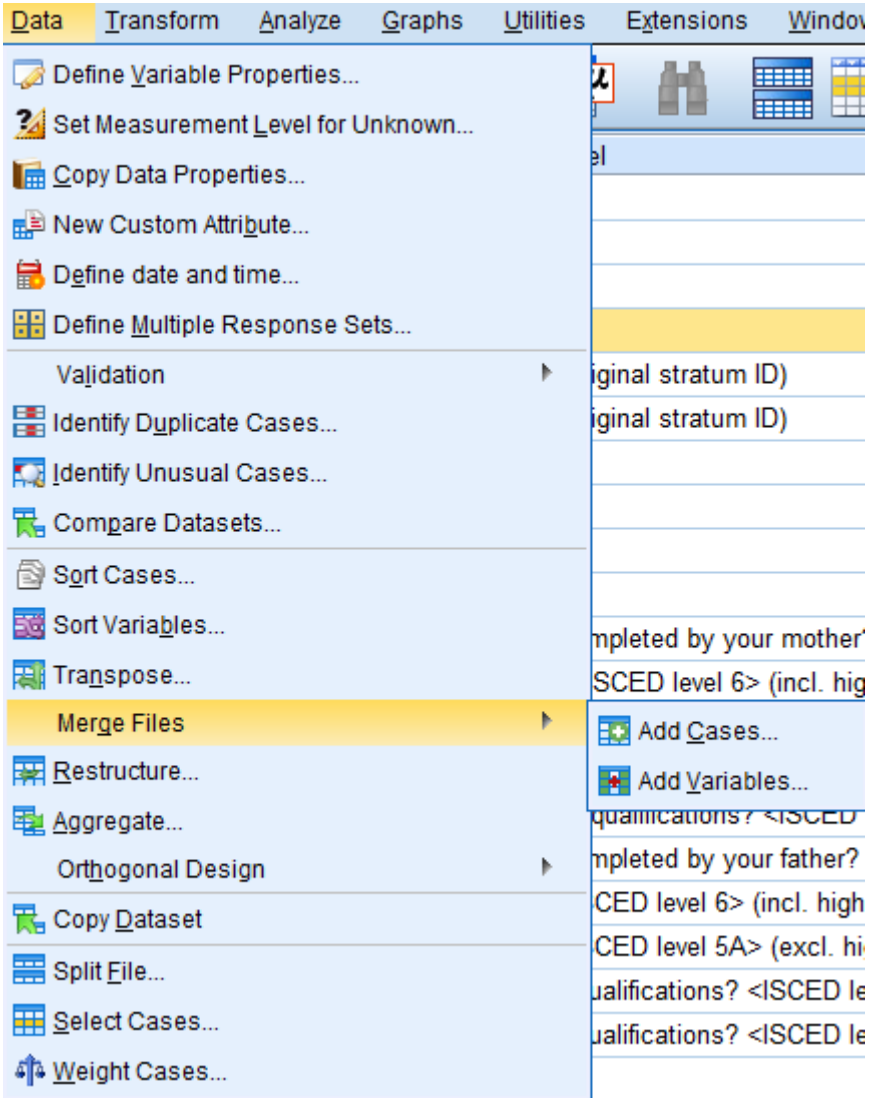




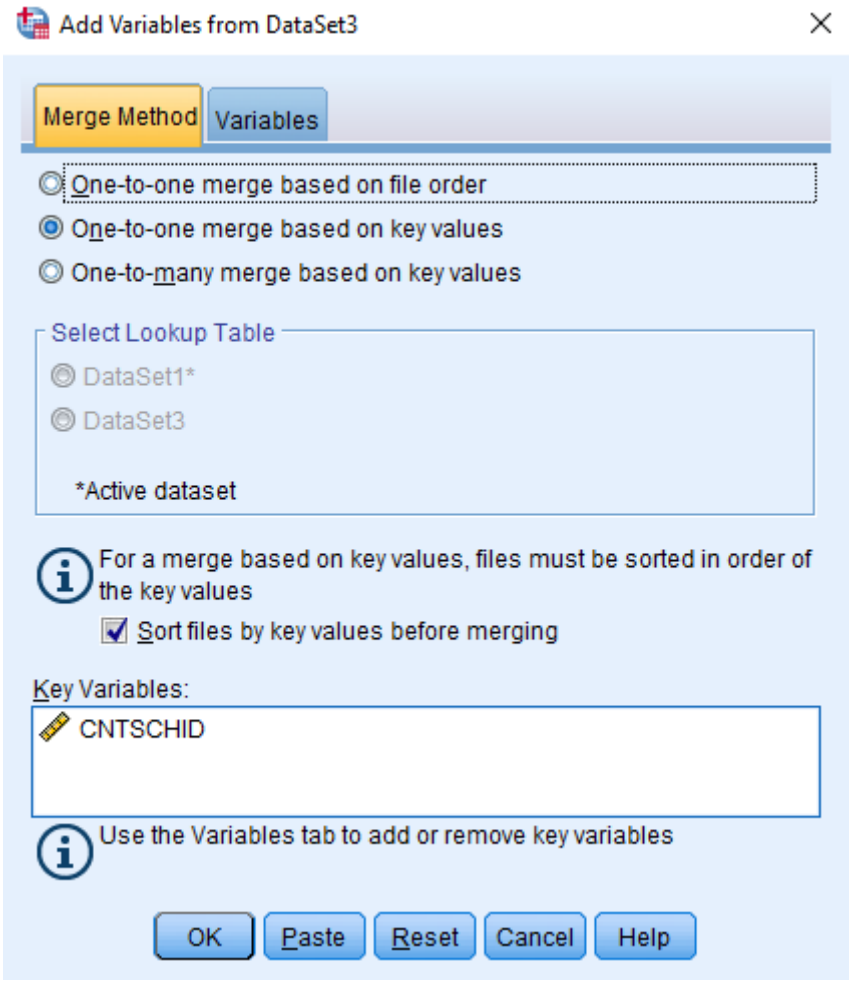
Datový soubor následně jednoduše dohledáme a otevřeme kliknutím na tlačítko *Open*. Po rozkliknutí šipky v okně typu souboru SPSS dále umožňuje otevřít soubory i v jiných formátech.

4.1.2 Spojování datových souborů

Nezřídka se stává, že potřebujeme spojit několik datových souborů dohromady. Možnost spojování datových souborů v SPSS nalezneme v záložce *Data → Merge Files*.



Při spojování datových souborů v SPSS máme dvě možnosti, jak je patrné z obrázku. Prvním způsobem je připojení případů (*Add Cases*), kdy k výchozímu datovému souboru přidáváme nové případy. Druhým způsobem je přidání proměnných (*Add Variables*), přičemž v tomto případě nepřipojujeme další případy, ale nové proměnné.



Spojovat soubory můžeme chtít například tehdy, kdy si přejeme k datovému souboru z testování žáků připojit soubor získaný z odpovědí učitelů v jejich škole. V takovém případě využijeme druhou možnost z nabídky, neboť budeme přidávat proměnné, nikoliv další žáky. Po zvolení možnosti *Add Variables* musíme nejdřív určit datový soubor, který chceme k výchozímu souboru připojovat. To je možné buď tak, že si tento soubor otevřeme (a rovnou ho v nabídce uvidíme), nebo soubor dohledáme v počítači. Jakmile máme soubor vybraný, otevře se nám další okno, v němž určujeme, jak se datové soubory spojí, na základě jakých klíčových proměnných a jaké proměnné případně chceme při spojování vyřadit. Z nabídnutých možností je zřejmé, že připojovat můžeme buď na základě řazení, nebo klíčových hodnot (*based on file order* x *based on key values*). Dál můžeme připojovat buď jednu hodnotu k jedné, nebo jednu hodnotu k několika (*one-to-one* x *one-to-many*). Pokud dojde k situaci, kdy máme datový soubor učitelů za určité školy, potom je možné připojovat tato data k jednotlivým žákům na základě identifikátoru školy. Identifikátor školy je tedy klíčová proměnná, je nezbytné ho mít v obou datových souborech a v obou souborech musí mít totožný název. Všem žákům z dané školy se tím pádem přiřadí jediná hodnota za učitele, což znamená, že možnost, kterou v našem případě zvolíme, je *One-to-many merge based on key values*. SPSS za klíčové pokládá totožné proměnné v datových souborech, což je v daném případě pouze identifikátor školy. Zde proto nic nastavit nemusíme. Pokud dál v záložce *Variables* nahoře žádné proměnné nevyřadíme, automaticky se připojí všechny. V tomto bodě tak můžeme zadání potvrdit kliknutím na tlačítko *OK*. Po spojování souborů je vždy vhodné datový soubor prohlédnout a zkontrolovat, jestli vše proběhlo v pořádku.

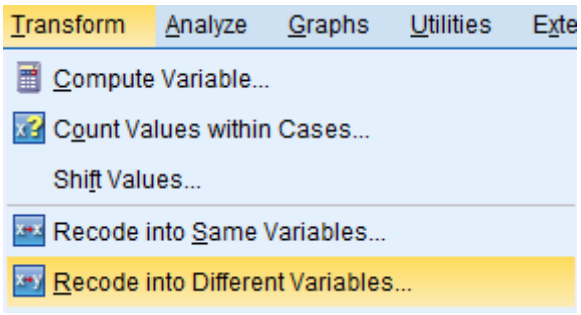
4.1.3 Další možnosti práce s daty

Kromě spojování datových souborů v záložce *Data* v SPSS nacházíme celou řadu dalších užitečných operací s daty. Například přes možnost *Aggregate* můžeme agregovat data za zvolenou jednotku. Pro situaci, kdy potřebujeme pracovat pouze s určitou částí datového souboru, SPSS nabízí možnost *Select Cases*. Výběr přitom může být učiněn různými způsoby (náhodný, na základě podmínky atd.). Přes nabídku *Weight Cases* zase nastavujeme vážení datového souboru.

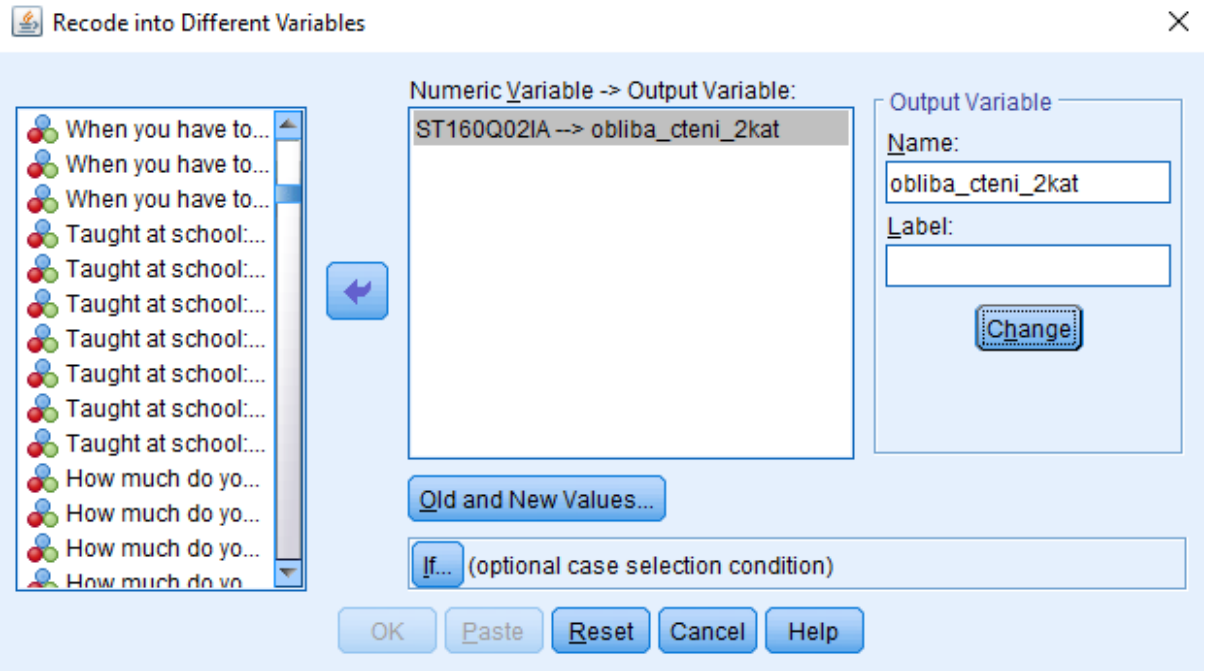
4.2 Transformace proměnných v SPSS

4.2.1 Rekódování proměnných

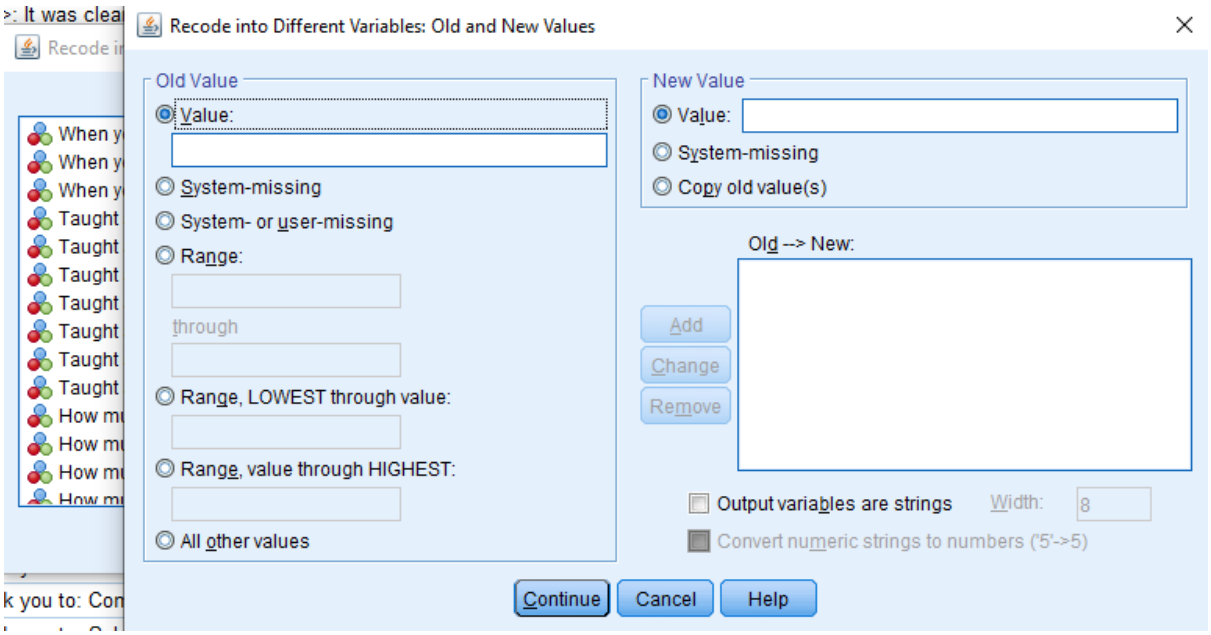
Rekódování proměnných v SPSS provádíme v záložce *Transform → Recode into Same Variables / Recode into Different Variables*.



Jak je z názvu patrné, v prvním případě rekódujeme proměnnou do stejné proměnné, což znamená, že si původní proměnnou přepíšeme. Obecně je tedy lepší postupovat druhou cestou, a sice rekódovat proměnné do nové proměnné. V tomto případě se nám v datovém souboru vytvoří nová proměnná a s původní rovněž nadále můžeme pracovat.



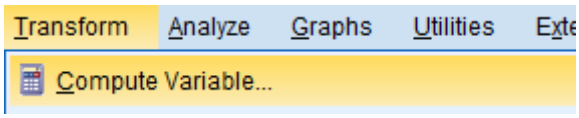
Demonstrovat postup rekódování proměnných můžeme například za využití dat z dotazování v rámci šetření PISA, kdy měli žáci reagovat na tvrzení, že je čtení jejich oblíbeným koníčkem. Vybírat přitom mohli ze čtyř kategorií odpovědí: rozhodně nesouhlasím, nesouhlasím, souhlasím a rozhodně souhlasím. Teoreticky můžeme chtít tuto proměnnou rekódovat na dvě kategorie souhlasu a nesouhlasu s uvedeným tvrzením. Využijeme k tomu právě proceduru *Recode into Different Variable*. Prvním krokem rekódování proměnných je nalezení proměnné, kterou chceme rekódovat, v seznamu proměnných a její následné přetažení do okna ve střední části obrazovky. Jelikož provádíme rekódování do nové proměnné, musíme novou proměnnou pojmenovat, k čemuž slouží pravá část okna. Proměnné rovnou můžeme přiřadit i český label. Jakmile máme nový název proměnné, klikneme na *Change*, čímž se přepíše výraz v okně uprostřed. Následně rozklikáváme *Old and New Values*, abychom mohli určit, jakým způsobem chceme proměnnou překódovat.



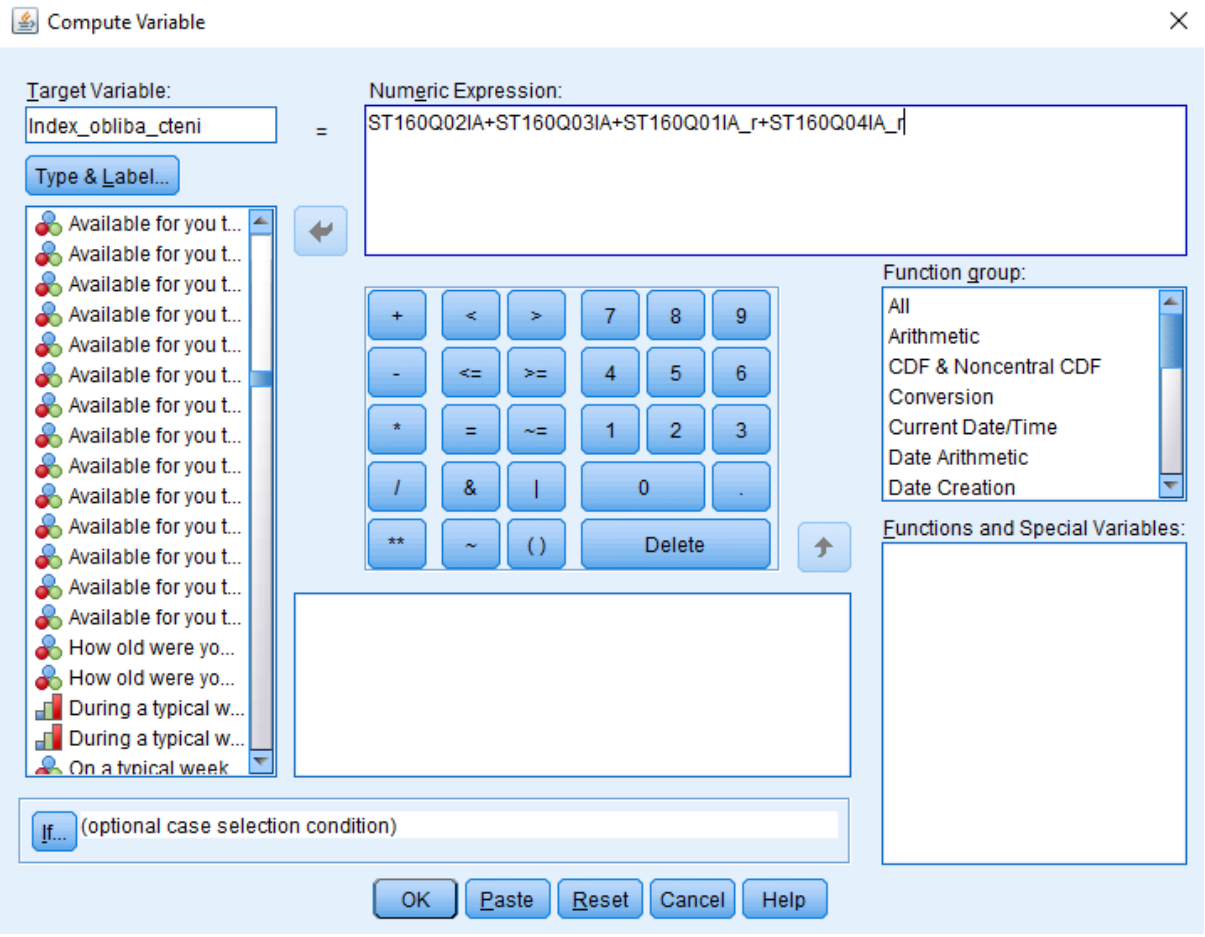
Zadávací okno je rozděleno na dvě poloviny. V levé části vyplňujeme původní hodnoty, v pravé potom dosazujeme hodnoty nové. Máme přitom několik možností, jak postupovat, přičemž rozhodující je samozřejmě konkrétní situace, kterou řešíme. První možností je specifikace konkrétních hodnot včetně práce s chybějícími hodnotami, dál můžeme určit rozsah hodnot, případně můžeme rekódovat všechny zbylé hodnoty najednou. V našem případě, kdy má stupnice pouze pár kategorií, můžeme postupovat cestou zadávání konkrétních hodnot. SPSS tedy zadáme, že chceme původní kategorie 1 a 2 nově překódovat do kategorie 1, která bude značit nesouhlas, a původní kategorie 3 a 4 do kategorie 2, která bude značit souhlas. Vždy když konkrétní hodnoty do SPSS zadáme, musíme zadání potvrdit kliknutím na tlačítko *Add*, přičemž nám daná informace přibude v nyní prázdném okně napravo. Jelikož jsme v datech původně rovněž měli kategorie tzv. chybějících odpovědí, můžeme využít možnosti *All other values* a v pravé části následně zaškrtnout *System-missing*, čímž je SPSS rovnou označí za chybějící hodnoty. Rovněž však lze využít možnosti *Copy old value(s)*, která tyto hodnoty zkopíruje do nové proměnné a my je potom jako chybějící označíme v přehledu proměnných. Jakmile máme zadáno, jakým způsobem chceme provést rekódování, potvrdíme zadání tlačítkem *Continue*. Následně je ještě vhodné v datovém souboru zkontrolovat, že se proměnná rekódovala správně.

4.2.2 Vytvoření nové proměnné

Jedním ze způsobů vytvoření nové proměnné je její výpočet. Vytvoření proměnné počítáním nalezneme v záložce *Transform → Compute Variable*.



Cestou výpočtu nové proměnné postupujeme například tehdy, když chceme ze sady proměnných vytvořit index. Pro ilustraci k tomu můžeme využít baterii otázek týkající se obliby čtení u žáků, do níž se řadí i dříve zmíněné tvrzení ohledně čtení jako oblíbeného koníčku. Kromě tohoto tvrzení žáci v rámci dotazování určovali mimo jiné míru svého souhlasu s tvrzeními, že čtou, jen když musí, rádi si povídají o knihách s jinými lidmi a že je pro ně čtení ztráta času. Vzhledem k tomu, že byla předkládaná stupnice odpovědí pro všechna tvrzení totožná, je na první pohled zřejmé, že je před tvorbou indexu potřeba některé stupnice rekódovat (otočit), k čemuž můžeme využít výše zmíněnou proceduru *Recode into Different Variables*. Před samotným vytvořením nové proměnné je na místě dodat, že je v rámci šetření PISA celá řada indexů vytvářena již mezinárodním týmem, tedy příklad není návodem, jak s těmito konkrétními daty pracovat, avšak slouží pouze pro demonstraci zmiňovaných postupů v SPSS.



Výpočet proměnné následně zadáváme po rozkliknutí *Compute Variable*. I v tomto případě je nejdříve potřeba novou proměnnou pojmenovat, k čemuž slouží pole v levé části nahoře nadepsané jako *Target Variable*. Do políčka *Numeric Expression* následně zadáváme způsob výpočtu nové proměnné. V našem případě můžeme hodnoty proměnných sečíst či vypočítat průměr. Zadání nakonec potvrdíme tlačítkem *OK* a správnost provedení ideálně rovnou v datech zkontrolujeme.

4.3 Úprava dat v programu RStudio

4.3.1 Nahrávání datového souboru

Do programu RStudio je možné nahrát datové soubory z různých zdrojů. Ať už se jedná o data vyhotovená v Microsoft Excel, SPSS, poznámkovém bloku, programu STATA, či v jiných programech. Je však zapotřebí využití správných příkazových řádků a instalace správného balíčku, který se následně aktivuje příkazem library. Aktivace příkazu je vždy nutná pomocí kláves Ctrl+Enter, popřípadě stisknutím tlačítka *Run* v horní liště. Program RStudio také potřebuje vědět, kde přesně se datový soubor ve vašem počítači nachází. Následující úkony tak lze jednoduše vykonat následujícími příkazy:

Příkaz library(haven) řekne programu RStudio, že budeme pracovat s datasetem[4], který nepochází přímo z RStudia. Pracovní prostředí si nastavíme pomocí setwd[5] (zkratka pro set working directory). Následně si zvolíme sami jméno,

```
library(haven)
setwd("přesné umístění souboru ve vašem počítači")
názevdatasetu <- read.table("přesný název datasetu ve složce")
```

které chceme pro práci s datovým souborem v RStudiu využít (názevdatasetu), řekneme programu, aby načetl tato data

[4] Při hledání informací se na stránkách R používá oficiálně název pro datový soubor data frame, zde ale bude užíván také název dataset.

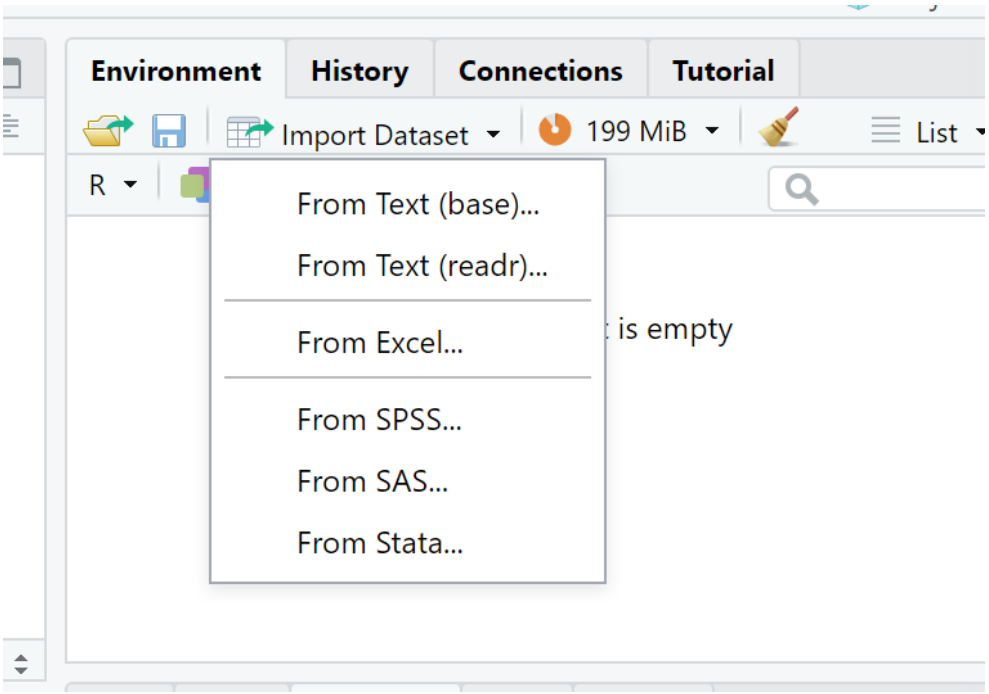
[5] Jestli se chceme pouze podívat, v jaké složce v počítači se váš skript nachází, můžeme využít příkaz getwd().

(read.table) a sdělíme mu název datasetu, který má ve složce hledat, a to i se správnou koncovkou (“přesný název datasetu ve složce”). Je důležité dodržovat správný postup i s uvozovkami.

Při nahrávání datového souboru se nám běžně může stát, že je dataset nahrán v souboru jiném než RStudio. Poté je důležité nahrát vždy správnou library, která soubor umožní otevřít. Pro Excel se například používá příkaz library(readxl) a namísto read.table využijeme read_excel, pro SPSS library(haven) a následně místo read.table využijeme read_sav. Pro data z poznámkového bloku, která nejsou oddělena mezerami, ale čárkami, využijeme namísto read.table příkaz read.csv a data oddělená tabem načteme pomocí read.delim.

Pro zjednodušení postupu nahrávání dat však existuje v RStudiu také možnost tzv. klikací. Jestliže máme nastaveno prostředí (rozumějme, program RStudio ví, s jakou složkou v počítači budeme pracovat pomocí příkazu setwd), můžeme nahrát datový soubor následujícím způsobem:

Z možnosti vybereme program, ve kterém jsou data uložena, a jednoduše otevřeme soubor.



Při nahrávání dat sepsaných v bloku se také vyžaduje, abychom programu RStudio sdělili, zda jsou v prvním řádku názvy proměnných. Příkaz pro tento úkon je header=TRUE. Jestliže máme v datovém souboru prázdná pole a chceme, aby je RStudio určitým způsobem pojmenovalo, což se může hodit v práci později, nastavíme chybějící hodnoty jako na=”NA”. Celý příkaz pak může vypadat takto:

```
názevdatasetu <- read.table("přesný název datasetu ve složce", header=TRUE, na="NA")
```

V případě, že jsou data umístěna v jiné složce, než ve které jsme si náš projekt v RStudiu nastavili, lze využít příkaz:

```
názevdatasetu <- read.table("přesné umístění datasetu / název datasetu", header=TRUE,
na="NA")
```

4.3.2 Náhled na atributy dat

Pro kontrolu dat, zdali se správně načetla a mají správný formát, můžeme využít následující možnosti příkazů. Malé datové soubory lze zobrazit jednoduše, a to napsáním názvu datového souboru do příkazového řádku a následně jeho spuštěním. Tento postup však není doporučován u dat většího rozsahu. Chceme-li se podívat v rychlosti na názvy proměnných v našem datasetu, můžeme využít příkaz:

```
names(názevdatasetu)
```

Další možností náhledu na data je snadné kliknutí na náš dataset v horním rámečku programu. Popřípadě využitím příkazu view(názevdatasetu).

Jakmile máme data nahraná v prostředí RStudia, uloží se do virtuální mezipaměti. V RStudiu v okně vlevo nahoře. Oproti SPSS je rozdíl ten, že RStudio dokáže pracovat s více datovými soubory najednou. Jedná se o velkou výhodu, kdy můžeme provést několik stejných analýz na jiných datasetech a výsledky porovnat v jednom analytickém výstupu.

4.3.3 Subsetování datasetu a spojování datasetů

První funkcí pro úpravu datasetu (v prostředí R se hovoří o tzv. dataframu) je subsetování datasetu. Příkaz níže ukazuje, že ve fiktivním datasetu máme proměnnou region, nicméně chceme provést analýzu bez případů za Ústecký kraj. Důvodem může být třeba to, že Středočeský kraj se svými charakteristikami odlišuje od zbylých krajů a mohl by zkreslit analýzu vzdělávací soustavy. Příkaz tedy vybere všechny ostatní regiony. Pokud zvolíme stejný název dataframu, data se nám přeuloží. Pokud zvolíme nový název dataframu, vytvoří se zcela nový dataframe.

Pro subsetování datasetů slouží logické operátory, dle nich můžeme zadat jakýkoliv logický příkaz pro výběr případů do nového dataframu.

```
názevdatasetu <- názevdatasetu[ which(názevdatasetu$REGION != 'Středočeský kraj'), ]
```

Logický operátor v R	Značení
Menší než	<
Menší než nebo rovno	<=
Větší než	>
Větší než nebo rovno	>=
Rovná se	==
Nerovná se	!=
A	&
Nebo	|
Pipe operátor	%>%

Nový dataframe lze opět uložit fyzicky na disk v libovolném datovém formátu. Například excel pomocí balíčku “xlsx”:

```
library(xlsx)
write.xlsx(názevdatasetu, file = "názevdatasetu.xlsx")
```

nebo SPSS:

```
write_sav(názevdatasetu, "./názevdatasetu.sav")
```

Dalším krokem může být úprava názvů proměnných, kdy v dataframu chceme z nějakého důvodu název proměnné změnit. Například chceme změnit obec_kod na OBEC_KOD, protože v jiných datasetech, například od ČSÚ, je tato proměnná zapsána kapitálkami a pomocí této proměnné chceme napárovat data. Identifikátor musí mít tak stejný název, aby se data propojila.

```
names(názevdatasetu)[names(názevdatasetu) == 'obec_kod'] <- 'OBEC_KOD'
```

Párování dat pak lze provést jednoduchým příkazem left_join. Předpokládáme spojení nového datasetu, který jsme si například stáhli z webu Českého statistického úřadu a chceme jej napárovat s naším cílovým datasetem. Identifikátor je v tomto případě kód obce.

4.3.4 Výpočet proměnných a základní rekódování

Pokračovat můžeme rekódováním proměnných. Zde můžeme rekódování rozdělit do dvou druhů. První je výpočet nové proměnné na základě klasických matematických vzorců, druhé se týká sociologických dat, kdy například chceme otočit ordinální škálu tak, aby odpovídala sémantickému významu.

V případě jednoduchých výpočtů pouze vypíšeme matematickou funkci, například chceme sečíst dvě proměnné.

```
názevdatasetu <- left_join(názevdatasetu, dataCSU, by = c("OBEC_KOD"))
```

Je nutné jen pamatovat na základní matematická pravidla a na správné používání závorek apod.

```
dataframe$proměnná_součet <- (dataframe$proměnná1 + dataframe$proměnná2)
```

Pokud chceme u dotazníkových dat otočit škálu, použijeme funkci recode z balíčku "car". Například spokojenost s učitelskou profesí s názvem TEACH_SAT, která v původním datovém zdroji byla kódována tak, že 5 znamenala vysokou nespokojenost a hodnota 1 vysokou spokojenost. Využijeme balíček "car" od statistika Johna Foxe.

```
library(car)
názevdatasetu$TEACH_SAT <- recode(názevdatasetu$TEACH_SAT, "'1"=5; "2"=4; "3"=3;
"4"=2; "5"=1)
```

Nyní máme proměnnou rekódovanou tak, že hodnota 5 znamená vysokou spokojenost.

Někdy je cílem vytvořit dichotomickou proměnnou. Například se rozhodneme, že spokojenost s profesí chceme analyzovat na této dichotomní škále. To můžeme provést tak, že hodnoty menší než 2 budou znamenat nespokojenost 1 a zbylé hodnoty budou znamenat spokojenost 0.

```
názevdatasetu$NESPOKOJEN_IND<-as.numeric(názevdatasetu$TEACH_SAT<="2")
názevdatasetu$NESPOKOJEN_IND<-as.numeric(názevdatasetu$TEACH_SAT>"2")
```

Někdy potřebujeme převést škálovou proměnnou na kategorickou proměnnou. Například můžeme uvažovat, že příjem respondenta rekódujeme do několika kategorií. Nejdříve zjistíme minimum a maximum (viz dále v sekci deskriptivní statistika), rozložení dat a poté se rozhodneme, jaké intervaly zvolíme pro dané kategorie.

Dalším častým úkonem při rekódování je standardizace proměnných na z-skóre, respektive do jednotek směrodatných odchylek.

```
názevdatasetu$PRIJEM_KAT <- recode(názevdatasetu$prijem, "10000:15000=1;
15001:30000=2; 30000:50000=3; 500001:300000=4")
```

```
dataframe$proměnná <- scale(dataframe$proměnná)
```

V případě, že chceme použít Gelmanovu metodu[6] dvou směrodatných odchylek, která se používá v případě vizualizace a prezentace regresních koeficientů, pustíme nejdříve tuhle funkci, kterou uložíme pod názvem twoSD. V příkazu na rekódování místo scale dáme twoSD. Proměnná bude standardizována do dvou směrodatných odchylek. Výhoda je pak porovnatelnost efektu, tedy regresních koeficientů, na rozdílných škálách. Interpretace je podobná jako v případě indikátorových proměnných, které de facto měří změnu z minima na maximum.

[6] Odborný článek Andrew Gelmana dostupný zde: http://www.stat.columbia.edu/~gelman/research/published/standardizing7.pdf

4.3.5 Práce s datasetem a úprava proměnných dle balíčku tidyverse

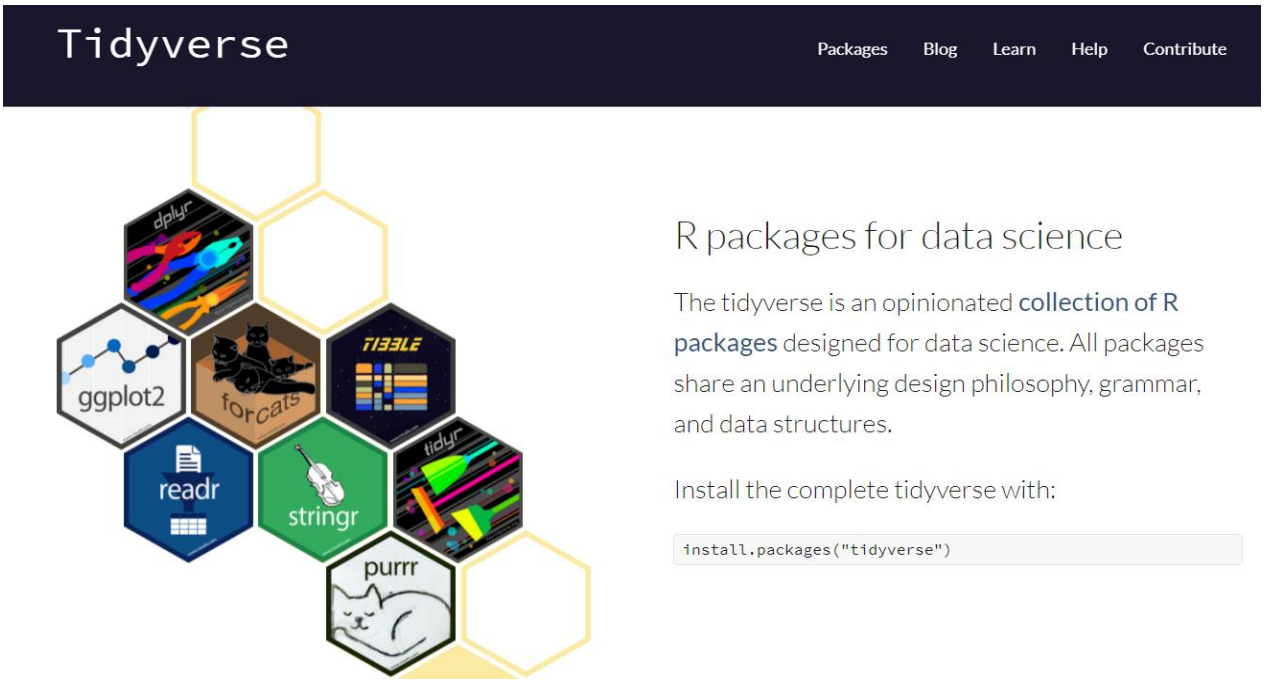
Kromě balíčku car, jehož cílem je spíše regresní diagnostika, existuje balíček dplyr ze sady tidyverse

```
# Gelmanova metoda - 2SD - nutné spustit tuto funkci s názvem "twoSD"

twoSD <- function (x) {
  (x - mean(x, na.rm = TRUE))/(2 * sd(x, na.rm = TRUE))
}

dataframe$proměnná <- twoSD(dataframe$proměnná)
```

(https://www.tidyverse.org/). Pokročilejší rekódování dat je tak vhodné v prostředí tohoto balíčku. Jedná se o jeden z nejpoužívanějších balíčků vytvořený týmem okolo slavného datového analytika Hadleyho Wickhama. Specifikum používání balíčku v rámci týmu okolo Hadlehy Wickhama z RStudia je tzv. pipe operator: "%>%". Pomocí něj lze skládat příkazy za sebou a kombinovat několik funkcí z celé rodiny balíčků tidyverse.



Subsetování datasetu (výběr podmnožiny případů):

```
library(dplyr)
dataframe <- dataframe %>%
  select(c(proměnná, proměnná, proměnná, proměnná, proměnná,
           proměnná, proměnná, proměnná))
```

Selekce případů, kdy chceme analyzovat všechny proměnné, ale jen některé případy dle selekce na základě identifikátoru. Například budeme chtít místo všech zemí v rámci šetření PISA analyzovat jen vybrané země.

Agregace proměnné na vyšší úroveň dle skupinového identifikátoru (např. úroveň školy).

```
novýdataframe <- dataframe %>%
```

```
library(dplyr)
novýdataframe <- dataframe %>%
  dplyr::group_by(identifikátor) %>% # split data pomocí identifikátoru
  dplyr::summarise(proměnná = mean(proměnná, na.rm = TRUE),
                   proměnná = mean(proměnná, na.rm = TRUE))
```

Balíček *dplyr* je následně možné využít také pro snadné rekódování proměnných. Konkrétně se v tomto případě využívá funkce *mutate*. Ve všech níže uvedených případech lze rekódovat z textových na číselné hodnoty a naopak. Rekódovat je možné i více hodnot najednou, v takovém případě by jednotlivé hodnoty měly být odděleny čárkou. Základní příkaz má následně tuto podobu:

```
dataframe %>% mutate(názevproměnné=recode(názevproměnné,
'původníhodnota1'='nováhodnota1', 'původníhodnota2'='nováhodnota2'))
```

Rekódování lze provádět i u více proměnných najednou, a to za pomoci jednoho příkazu.

```
Dataframe %>% mutate(názevproměnné1=recode(názevproměnné1,
'původníhodnota1'='nováhodnota1'),názevproměnné2=recode(názevproměnné2,
'původníhodnota1'='nováhodnota1'))
```

Pokud je potřeba některou z hodnot označit jako chybějící hodnotu, postačuje do příkazu doplnit *.default=NA_character_*. Po vložení tohoto příkazu program převede všechny hodnoty proměnné, u kterých není specifikována jiná nová hodnota, na chybějící hodnoty (NA).

```
dataframe %>% mutate(názevproměnné=recode(názevproměnné,
'původníhodnota1'='nováhodnota1', .default=NA_character_))
```

Pro rekódování proměnné do tzv. *dummy proměnných* je následně nutné funkci *mutate* rozšířit ještě o funkci *spread*.

```
dataframe %>% mutate(dummy=1) %>%
spread(key=názevrekódovanéproměnné,value=dummy)
```

Takto nastavený příkaz by původní proměnnou rozdělil na nové proměnné, a to na základě hodnot původní rekódované proměnné. Výskyt hodnoty původní proměnné by byl v nových proměnných indikován hodnotou 1. Tuto hodnotu je možné změnit úpravou části příkazu *dummy=1*. Zároveň by tento příkaz zbylé případy v nových proměnných označil hodnotou *NA*, tedy jako chybějící hodnoty. Pokud by však bylo třeba z nových proměnných udělat dichotomické proměnné, tedy namísto *NA* doplnit hodnotu 0, musí být příkaz rozšířen následujícím způsobem.

V případech, kdy je potřeba rozdělit intervalovou proměnnou na několik kategorií, je možné využít tento příkaz:

```
dataframe %>% mutate(dummy=1) %>%
spread(key=názevrekódovanéproměnné,value=dummy, fill=0)
```

```
dataframe %>% mutate(názevpůvodníproměnné=cut(názevnovéproměnné, breaks=c(-Inf, 0.5, 1,
Inf), labels=c("nízký","střední","vysoký")))
```

V rámci části příkazu *breaks=c()* je možné nastavit dělící hodnoty, s jejichž pomocí budou rozřazeny případy původní proměnné. Počet hodnot, a tedy i kategorií není nijak omezen. V tomto případě došlo k rozdělení do tří kategorií – od minus nekonečna do 0,5; od 0,5 do 1; od 1 do nekonečna. Následně v části příkazu *labels=c()* je možné jednotlivé kategorie, a tedy i hodnoty nové kategorické proměnné pojmenovat.

Datasety obsahují chybějící hodnoty. Pokud je datový soubor nahrán z SPSS pomocí balíčku haven, R již s chybějícími hodnotami pracuje tak, jak byly ve variable view nastaveny v programu SPSS. V případě dalšího softwaru (STATA, SAS) již R s chybějícími hodnotami nepracuje, respektive pracuje nepřesně. Z tohoto důvodu je třeba některé chybějící hodnoty nastavit. Obecně v R jsou chybějící hodnoty v dataframu označeny jako “NA”. Pokud má dataset kódy pro chybějící hodnoty například standardně 99 nebo -9999, můžeme je převést na chybějící hodnoty pomocí příkazu:

```
dataframe $názevproměnné[dataframe $názevproměnné==99] <- NA
```

Zde je nutné mít na paměti, že chybějící hodnoty musí být vždy správně nastaveny, jinak hrozí, že analýzy budou zkresleny. Při analýze krátké ordinální stupnice 1 až 4 bez nastavení chybějících hodnot bude software počítat s hodnotami 99 a celkově zkreslí analýzu.

Některé analýzy nelze provést na dataframu, který obsahuje chybějící hodnoty. Z tohoto důvodu musíme chybějící hodnoty odstranit. Zde však pozor, příkazem odstraníme celé řádky. Příkaz pro odstranění chybějících hodnot z datasetu je na.omit. Zde doporučujeme nejdříve dataset subsetovat a vybrat jen a pouze ty proměnné, které chceme dále analyzovat. Důvodem je to, že pokud nám zůstane proměnná, kterou sice v analýze nevyužijeme, ale ta obsahuje chybějící hodnoty, přijdeme i o ty případy, které by byly kompletní bez započtení této proměnné.

```
dataframe<- na.omit(dataframe)
```

Práce s labely je v RStudiu poněkud komplikovanější. Je nutné odlišit název proměnné, label proměnné a labely hodnot proměnné. V případě změny labelu názvu proměnné používáme balíček Hmisc. Skript níže nám přidá label, který se pak bude zobrazovat například na osách grafů a dalších grafických výstupů v RStudiu. V případě, že je soubor z SPSS (sav), dochází k převodu labelů i do prostředí RStudia. Pokud ale importujeme excelový soubor, nebo obecně datový formát oddělený oddělovačem (csv), musíme labely doplnit manuálně dle daného codebooku či dotazníku.

```
library(Hmisc)
describe(dataframe)
Hmisc::label(dataframe$t_VZD) <- "Vzdělání"
```

Jestliže chceme upravit labely hodnot proměnné, použijeme následující skript, který je RStudiu defaultní a není třeba balíčku. Liší se však úroveň měření, měli bychom použít příkaz factor pro kategorické proměnné a příkaz ordered pro ordinální. Například u proměnné s kódem IDE_8, což je pohlaví respondenta, doplníme label muž a žena následovně:

```
dataframe$IDE_8 <- factor(dataframe$IDE_8,
                          levels = c(0, 1, 2),
                          labels = c("BEZ ODPOVĚDI", "muž", "žena"))
```

V případě ordinální proměnné věk 5 kategorií.

```
dataframe$t_VEK_5  <- ordered(dataframe$t_VEK_5,
                          levels = c(0, 1, 2, 3, 4, 5),
                          labels = c("BEZ ODPOVĚDI", "15-19", "20-29", "30-44", "45-59", "60+"))
```

V obou případech je pak vhodné kategorii 0 rekódovat na chybějící hodnotu.

```
dataframe $IDE_8[dataframe $IDE_8==0] <- NA
dataframe $t_VEK_5[dataframe $t_VEK_5==0] <- NA
```

5

Deskriptivní analýza

5 DESKRIPTIVNÍ ANALÝZA

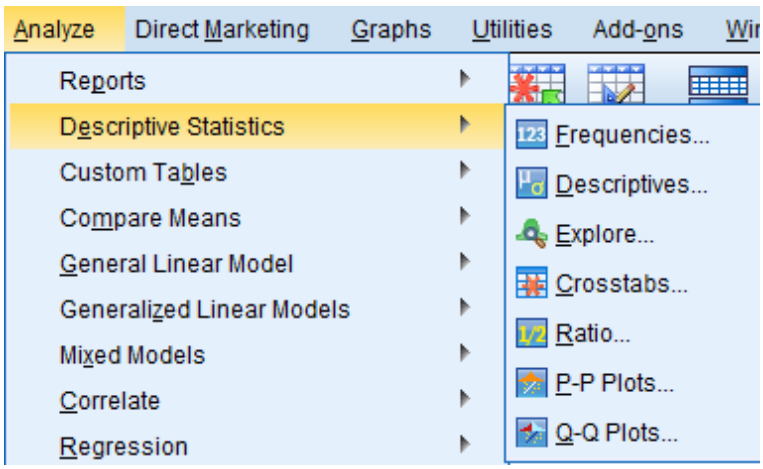
Základním typem analýz prováděným při de facto jakékoli analytické činnosti (nejen) v souvislosti s daty ze vzdělávacího systému je analýza deskriptivní (popisná). Ta zahrnuje různé základní tabulkové a grafické analýzy podílů či nominálních hodnot jednotlivých proměnných, jejichž primárním cílem není ani odhalování příčinných souvislostí, ani odhalování korelací mezi proměnnými, ale pouhý základní zjednodušený pohled na strukturu dat. Takový pohled je ovšem zcela zásadní, protože může dopomoci nalézt potenciálně významné vztahy mezi proměnnými, odchylné případy a jiné užitečné údaje nutné pro složitější statistické analýzy. Velice často (avšak ne výhradně) taktéž platí, že co nenaznačí již jednoduchá deskriptivní analýza, neodhalí ani složitější postupy.

Zejména v kontextu analýzy dat z mezinárodních šetření gramotností není užití SPSS pro deskriptivní statistiku vhodné, protože program nedokáže pracovat s komplexními váhami jinak než užitím doplňkové syntaxe. Tento způsob využívá např. IEA vydávaný IDB analyzer, určený přímo k základní analýze těchto dat. Pracovat s daty se správným vážením dokážou též další statistické programy v čele s nástroji založenými na jazyku R. Práci se systémem SPSS zde dáváme především pro úplnost a jako možnost analýzy národních dat např. ze systému InspIS, kde není nutné pracovat s komplexními váhami jako na úrovni dat z mezinárodních šetření.

5.1 Deskriptivní analýza v SPSS

Pozn. Přestože SPSS nabízí kromě práce s dialogovými okny také práci se skriptovacím jazykem, pro účely tohoto dokumentu uvádíme pouze verzi práce s dialogovými okny. Čtenáře, který zvládá i práci se systémem příkazového řádku, odkazujeme mj. na jazyk R.

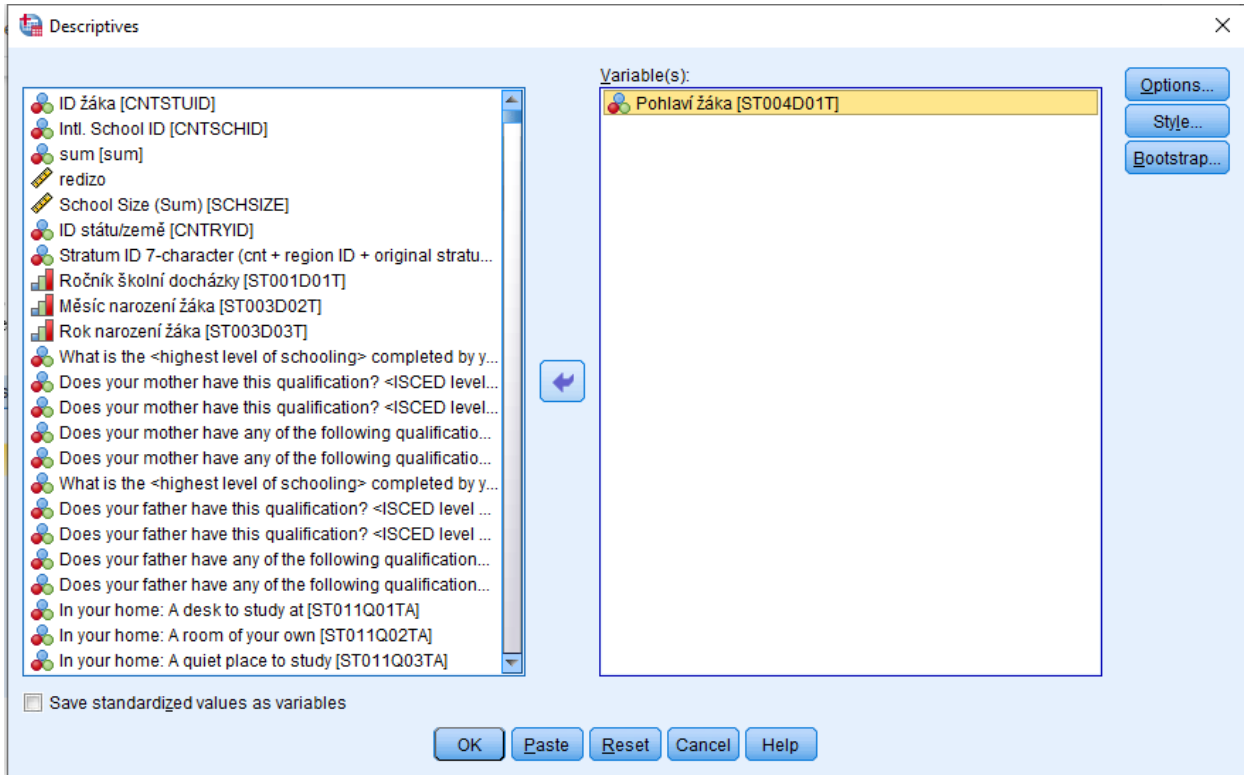
Základní deskriptivní techniky se v programu SPSS skrývají v záložce *Analyze → Descriptive Statistics*. Zcela základními nástroji, které budou popsány i zde, jsou funkce *Frequencies*, *Descriptives* a *Crosstabs*, v další záložce *Analyze → Reports* pak nalezneme ještě často využívanou funkci *OLAP Cubes*.

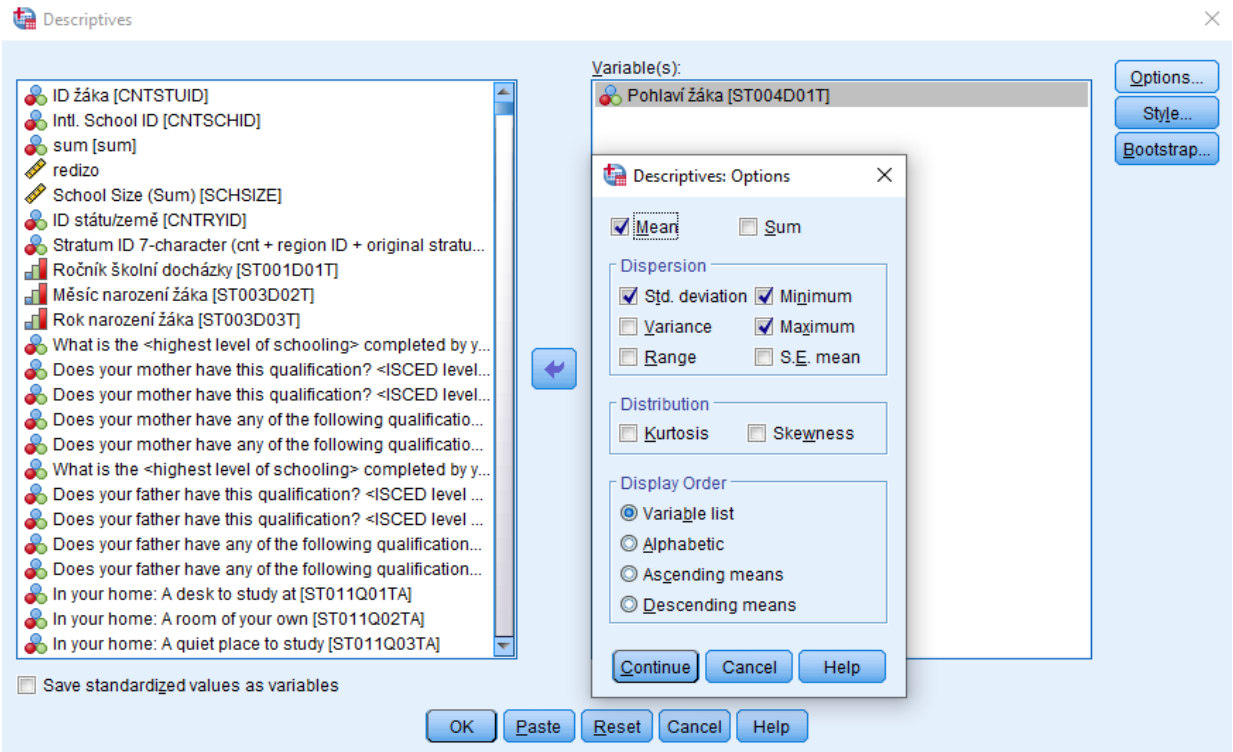


5.1.1 Funkce Descriptives

Funkce *Descriptives*, jak již název napovídá, zobrazuje zcela základní popisné charakteristiky vybraných proměnných, jako je průměr, minimální či maximální hodnota, rozptyl apod. To umožňuje udělat si základní představu o charakteru proměnné a odhadovat její chování v dalších analýzách. V okně *Descriptives* lze buď dvojitým poklepáním na proměnnou, nebo jejím označením a kliknutím na symbol šipky mezi dvěma sloupečky přesunout proměnnou do okna *Variable(s)*, se kterými bude dále pracováno. Vhodné je zde otevření záložky *Options*, kde je možné nastavit další popisné statistiky, které budou pro danou proměnnou zobrazeny. Kromě běžných charakteristik jako průměr (Mean), minimální hodnota (Minimum) či maximální hodnota (Maximum) a směrodatná odchylka (Std. deviation) lze zobrazit např. též údaje o rozptylu či směrodatné chybě (S.E.Mean). Známe-li distribuční charakteristiky dat a případný typický sklon dat, lze v sekci *Distribution* pracovat i s tímto.

(Nejen) funkce *Descriptives* nabízí i záložky *Style*, která umožňuje např. podmínečné formátování jednotlivých výstupních buněk, a *Bootstrap*, která pracuje s variantou tzv. metody *Monte Carlo*, která dokáže náhodně vracet hodnoty dle zadaných parametrů a modelovat tak významně větší vzorky dat, než které jsou analytikovi k dispozici. Používá se pro odhad přesnosti výběrového vzorku nebo pro situace, kdy je např. nutné ověřovat nějaký předpoklad na situacích se známými parametry, pro které ovšem nemáme reálná data. Pro běžnou analýzu jsou tyto funkce nepotřebné, a proto je dále nebudeme popisovat.





Důležitou funkcí v rámci funkce *Descriptives* je možnost uložení standardizovaných hodnot jako nové proměnné (zaškrtávací políčko vlevo dole), čímž lze jakoukoli proměnnou jednoduše převést na proměnnou novou ve formátu standardizovaného Z-skóre (převedení hodnot proměnných z různých škál na standardizovanou společnou škálu, což umožňuje porovnávání hodnot proměnných, tvorbu standardizovaných indexů atd.).

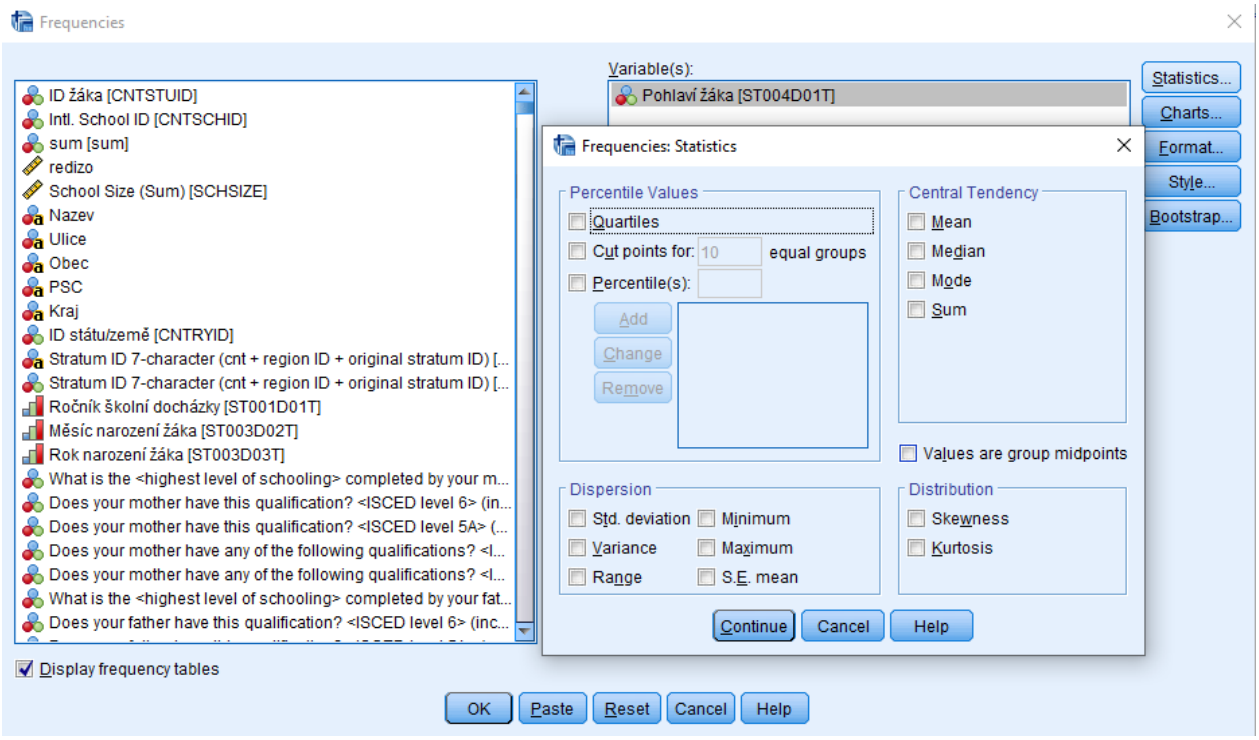
Výstupem analýzy je jednoduchá přehledná tabulka pro neomezený počet proměnných (lze provádět hromadně pro více proměnných).

Descriptive Statistics

	N	Range	Minimum	Maximum	Mean		Std. Deviation	Variance
	Statistic	Statistic	Statistic	Statistic	Statistic	Std. Error	Statistic	Statistic
Pohlaví žáka	7019	1	1	2	1,50	,006	,500	,250
Valid N (listwise)	7019							

5.1.2 Funkce Frequencies

Funkce *Frequencies* využívá obdobný interface jako funkce *Descriptives*. Cílem není poskytnutí obecných popisných charakteristik, ale údajů o rozložení dat a hodnotách jednotlivých kategorií proměnné. Důležitá je zde především záložka *Statistics*, kde lze nastavit jednotlivé porovnávané skupiny. V případě nominálních a ordinálních proměnných lze ponechat skupiny definované samotným kódováním proměnné (v případě pohlaví typicky chlapci vs. dívky) a nastavit popisné charakteristiky pro každou jednotlivou skupinu – průměr (Mean), medián (Median) či jinou míru centrální tendence a údaje známé již z funkce *Descriptives*. V případě kardinálních proměnných je možné nastavit buď quartily, počet vygenerovaných stejných skupin, nebo lze ručně nastavit percentily jako hraniční body oddělující jednotlivé skupiny.



V záložce *Charts* lze nastavit, zda se data za jednotlivé skupiny mají zobrazit např. ve sloupcovém či koláčovém grafu, v případě kardinálních proměnných i ve formě histogramu s křivkou normálního rozdělení, která pomáhá odhalit případný sklon dat od normálního rozdělení. V záložce *Format* pak sestupnost či vzestupnost dat ve výstupní tabulce dle velikosti procentuálního podílu.

Výstupem funkce jsou opět jednoduché tabulky zahrnující celkový počet případů a chybějících hodnot, v samostatné tabulce pak navolené údaje pro jednotlivé definované skupiny. V základním zobrazení funkce vrací počet případů jednotlivých skupin (*Frequency*) a jejich procentuální podíl, spolu s hodnotou chybějících hodnot lze dopočítat validní a kumulativní procenta.

Statistics

Pohlaví žáka

N	Valid	7019
	Missing	0

Pohlaví žáka

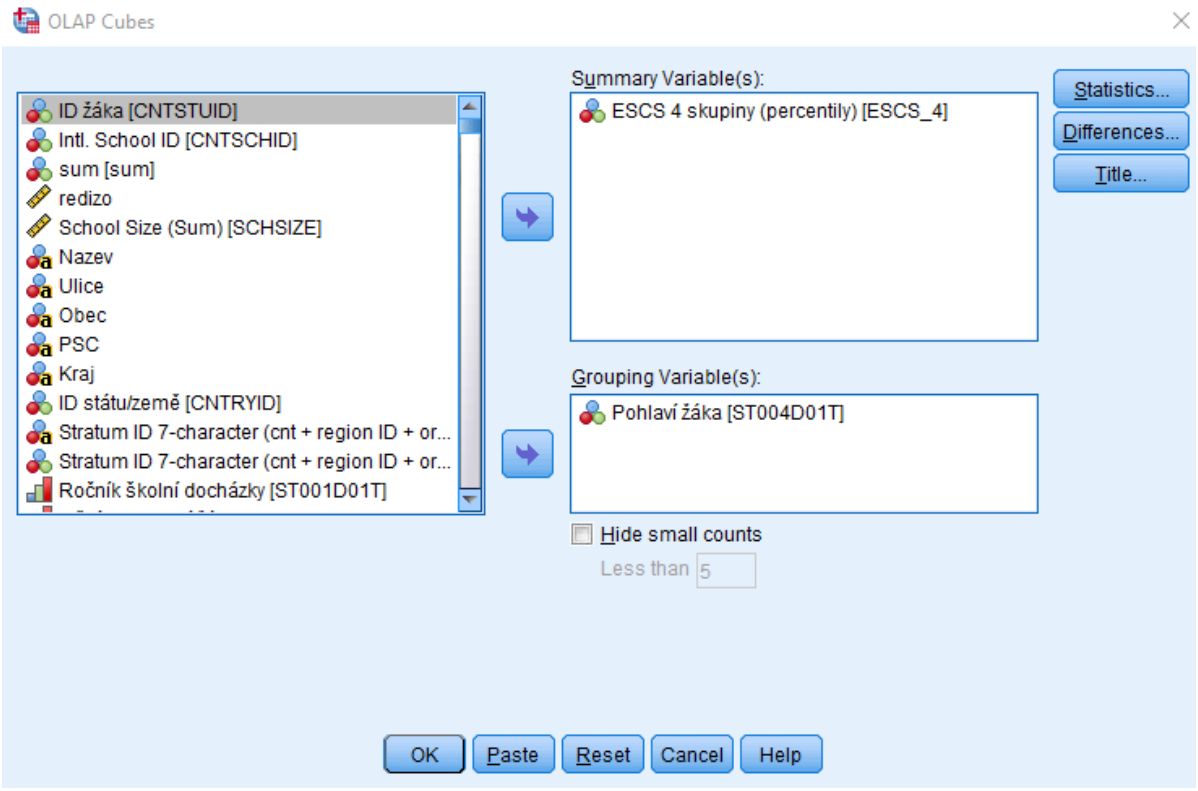
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	dívka	3518	50,1	50,1	50,1
	chlapec	3501	49,9	49,9	100,0
	Total	7019	100,0	100,0	

5.1.3 Funkce Crosstabs (kontingenční tabulky)

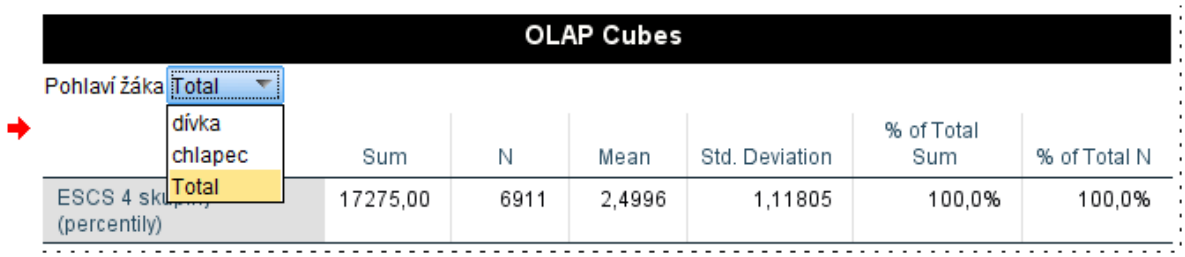
Funkce *Crosstabs* neboli funkce *kontingenčních tabulek* funguje v programu SPSS na obdobném principu jako v kterémkoli jiném statistickém programu nebo tabulkovém editoru. Nutné je použití nominálních či ordinálních proměnných. Definuje se proměnná, která se má zobrazit v řádcích a proměnná, která se má zobrazit ve sloupcích. Třídění druhého a vyššího stupně lze docílit přidáním další vrstvy dat (*Layer*). Takto definovaná tabulka posléze pracuje s hodnotami v křížení proměnných v řádcích a sloupcích (odtud název *Crosstabs*). K tomuto křížení se pak váže nastavení výstupních parametrů od typu zobrazených hodnot po různé dodatečné statistické hodnoty, čemuž je věnována důkladnější pozornost v následující části textu.

5.1.4 Funkce OLAP Cubes

Funkce OLAP Cubes v SPSS je velice zjednodušeně nástroj kombinující funkce *Crosstabs* a *Descriptives*. Po zvolení této funkce máme možnost definovat tzv. souhrnné proměnné (*Summary variables*) a seskupovací proměnné (*Grouping variables*). Výstupem analýzy bude souhrnná deskriptivní statistika souhrnných proměnných, a to pro jednotlivé skupiny dle proměnných seskupovacích. Pro ilustraci jsme jako souhrnnou proměnnou zadali ESCS žáků a jako seskupovací proměnnou jejich pohlaví.

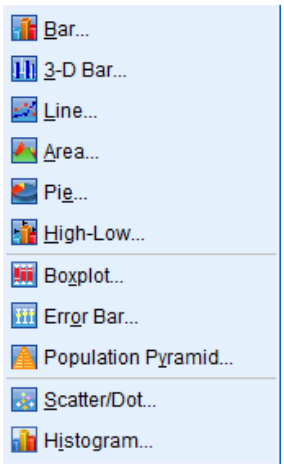


Výstupem takové analýzy je tabulka, kde však žádné skupiny nevidíme. Je potřeba na tabulku dvojitě poklepat a v záložce seskupovací proměnné vybrat příslušnou skupinu (nebo nechat bez rozlišení). Takto jsme schopni postupně procházet deskriptivní statistiky různých skupin dat dle zadaných proměnných. Zobrazené statistiky si přitom navolíme v záložce *Statistics*, v záložce *Differences* jsme schopni definovat, mezi jakými skupinami chceme pozorovat rozdíly. Takové páry (definované kódováním seskupovací proměnné) se pak zobrazí jako jedna ze skupin v záložce seskupovací proměnné v tabulce.

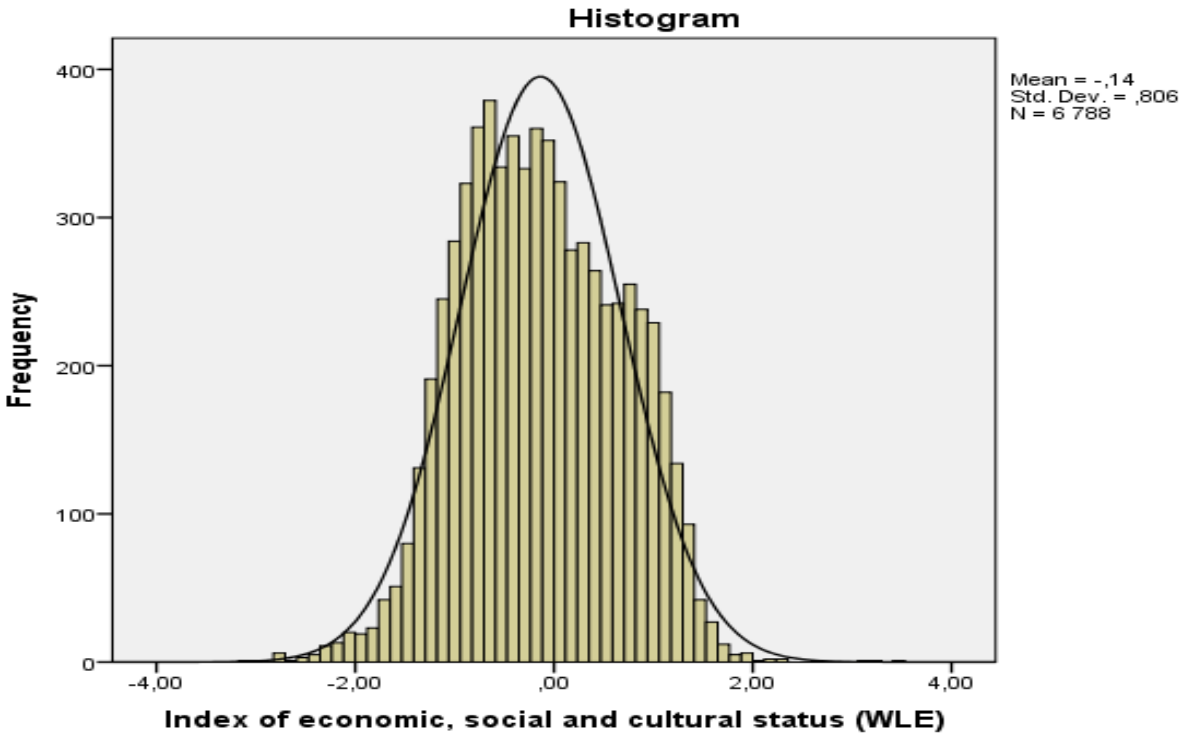


5.1.5 Konstrukce grafů – graf typu boxplot a graf typu histogram

Program SPSS umožňuje zobrazit data v různých typech grafů. V samostatné záložce *Graphs* → *Legacy dialogs* lze vybrat typ grafu a pomocí dialogového okna navolit proměnné další parametry, které se zobrazí v daném typu grafu.

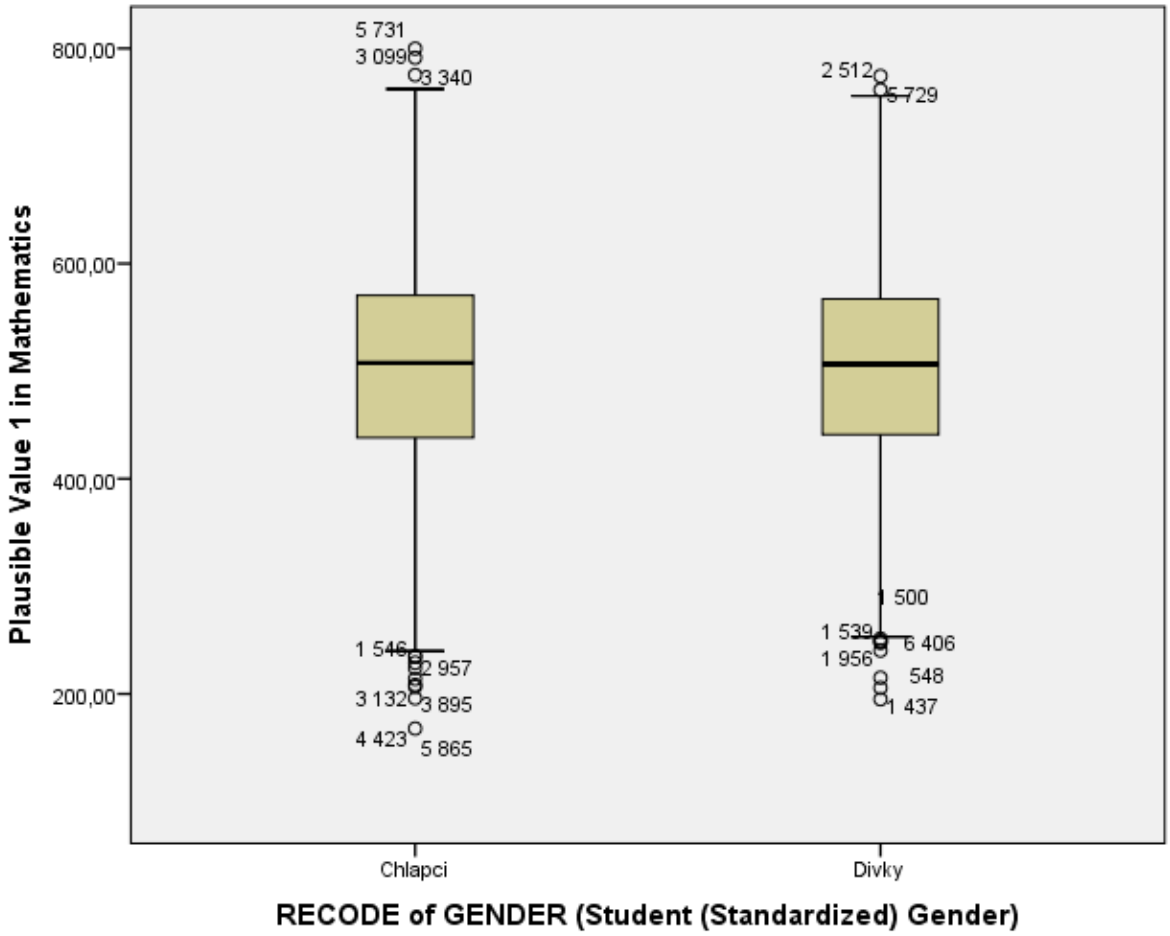


Vynecháme-li běžné sloupcové, bodové, koláčové aj. grafy, zejména funkce grafu typu *Histogram* patří mezi nejčastěji využívané základní popisné statistiky. Lze ji vyvolat buď v nabídce grafů, nebo jako doplňující graf při spuštění funkce *Frequencies*. S pomocí prolnutí křivky tzv. normálního rozdělení (které odkazuje na ideální rozložení reprezentativních dat při přísně náhodném výběru), je možné pomocí tohoto grafu odlišit např. sklon dat, identifikovat odchylné případy či poukázat na nutnost transformace proměnných před jejich použitím např. v regresním modelování.



Druhým často využívaným typem grafu je tzv. *Boxplot* (známý jako tzv. krabicový graf), který umožňuje pomocí pěti údajů zobrazit rozložení dat dané proměnné, zejména nalezne využití při porovnávání skupin případů. Graf lze vyvolat pomocí dialogového okna z nabídky grafů.

Graf typu boxplot v SPSS zobrazuje pět viditelných horizontálních linií, které vzestupně označují minimum, první kvartil, medián, třetí kvartil a maximum. S pomocí boxplotu lze odhalit rozptyl v datech dané proměnné, zejména při porovnání s druhou skupinou, v modelovém případě chlapci vs. dívky, pak lze předvídat chování např. v testech statistické významnosti rozdílů v průměrech (t-test aj.). Užitečnou funkcí je prolínání grafu boxplotu s jednotlivými případy okolo minima a maxima, které poukazují na odchylné případy (outliers) daných skupin, což umožňuje cílení analýz i na tyto přesně identifikované krajní případy.



5.2 Deskriptivní analýza v R

Výše jsme se seznámili se základními funkcemi pro tvorbu deskriptivní statistiky v programu SPSS. Následující část přiblíží, jak tyto funkce spustit v programu RStudio. Po nastavení pracovního prostředí a správném nahrání dat do softwaru spustíme základní deskriptivní statistiku pomocí příkazu summary. V praxi pak vypadá příkaz následovně:

První příkaz ve spodní konzole zobrazí základní statistické atributy všech proměnných v datasetu. Jmenovitě se jedná o nejnižší hodnotu proměnné (min.), hodnotu proměnné v prvním kvartilu (1st Qu.), medián, aritmetický průměr (mean), hodnotu proměnné ve třetím kvartilu (3rd Qu.) a nakonec nejvyšší hodnotu proměnné (max.). Číselné hodnoty pochopitelně nevyjedou u proměnných, které místo čísel obsahují text, například u názvů škol.

```
summary(názevdatasetu)
        summary(názevdatasetu$názevproměnné)
```

Druhý příkaz ukáže tyto hodnoty pouze u jedné zvolené proměnné z příslušného datasetu. Aby RStudio vědělo, s jakou proměnnou, kterého datasetu má pracovat, musíme nejprve definovat dataset a za symbol $ napsat proměnnou, se kterou chceme pracovat.

Zajímá-li nás v datasetu pouze jedna proměnná a její charakteristiky, můžeme využít následující příkazy jednotlivě:

```
mean(názevdatasetu$názevproměnné)
median(názevdatasetu$názevproměnné)
quantile(názevdatastu$názevproměnné)
IQR(názevdatasestu$názevproměnné)
```

Uvedené příkazy nám postupně vypočítají aritmetický průměr (mean), medián (median), hodnoty jednotlivých kvantilů a nakonec mezikvartilové rozpětí dané proměnné (IQR).

Pro další práci s daty se silně doporučuje využití balíčku vytvořeného statistikem Hadleyem Wickamem. Jedná se o balíček tidyverse, který nejenže obsahuje další balíčky a řadu nástrojů, ale využívá se pro zkracování příkazů pomocí tzv. pipe operator %>% (Ctrl+Shift+M). Po instalaci a následné aktivaci balíčku pomocí příkazu library popsaného výše můžeme základní inspekci dat provést pomocí příkazu:

```
názevdatasetu %>% glimpse()
```

Příkaz ve spodní konzole zobrazí počet řádků a sloupců v datasetu a část hodnot jednotlivých proměnných. Proměnné jsou vždy uvedeny pod zpravidla zkratkovitým názvem, který usnadňuje psaní příkazů, proto je nutné mít po ruce codebook, který vysvětluje, co daná zkratka znamená.

5.2.1 Standardizace proměnných

Pro porovnatelnost proměnných mezi sebou je zapotřebí proměnné tzv. standardizovat. Nejsnadnější cesta pro standardizaci v programu RStudio je využití příkazu scale. Úkon pak může vypadat následovně:

První část příkazu říká, že chceme vytvořit proměnnou novou (*STD před názvem proměnné je zvoleno pro následnou*

```
názevdatasetu$STD_názevproměnné <- scale(názevdatasetu$názevproměnné*(-1))
```

snadnější orientaci v datech. Název si ale můžeme zvolit zcela libovolný). Za příkazem scale pak následují závorky, v nichž definujeme jasně dataset a proměnnou, kterou chceme standardizovat. Za názvem dále pokračuje *(-1), což není vždy potřeba, využívá se pouze v případě, kdy chceme obrátit škálu dané proměnné. Jestliže chceme zkontrolovat, zda se nám nové proměnné skutečně vytvořily, využijeme nám již známý příkaz view(názevdatasetu). Na konci seznamu proměnných by se již měly standardizované proměnné zobrazovat.

5.2.2 Inspekce disperze dat

Deskriptivní statistika je dále charakterizována analýzou disperze dat. Konkrétně se jedná o rozptyl, variační rozpětí od nejnižší po nejvyšší hodnotu dané proměnné, směrodatnou odchylku, standardní chybu a interkvartilové rozpětí (které je uvedeno již v části textu výše). Příkazy pro tyto hodnoty jsou následující:

```
var(názevdatasetu$názevproměnné)
sd(názevdatasetu$názevproměnné)
library(plotrix)
std.error(názevdatasetu$názevproměnné)
```

Hodnotu rozptylu vypočítá software při spuštění příkazu var (*zkratka pro variation*). Směrodatná odchylka je spočítána příkazem sd (*zkratka pro standard deviation*), standardní chyba, pomocí které pak můžeme dále vypočítat interval spolehlivosti, vypočítá RStudio pomocí příkazu std.error, avšak pozor, zde jde nutno nainstalovat a aktivovat balíček s názvem plotrix. Variační rozpětí můžeme snadno vypočítat pomocí již zmíněného příkazu summary, interkvartilové rozpětí je výše zmíněno pod příkazem IQR, popřípadě lze vypočítat ručně z příkazu summary jako hodnota 3. kvartilu minus hodnota 1. kvartilu.

Chceme-li získat hodnoty jiných kvantilů, využijeme následující možnost:

```
quantile(názevdatasestu$názevproměnné, c(0, 0.1, 0.5))
```

Ve spodní konzole programu se nám následně ukážou hodnoty, pod kterými leží 10 % a 50 % případů.

5.2.3 Frequencies

Již jsme si ukázali funkci frequencies v programu SPSS, kde ji lze vytvořit pouze snadným klikáním. Program RStudio v případě tvorby takové tabulky není tak uživatelsky přívětivý. Hlavním problémem, který musíme řešit, je otázka labelů neboli názvů proměnných a názvů jednotlivých hodnot. Chceme-li se v rychlosti podívat, jaké proměnné náš datový soubor obsahuje a jaké labely mají jednotlivé hodnoty, musíme nejprve nainstalovat **balíčet sjPlot** a pomocí příkazu library jej následně spustit. Využijeme pak následující příkaz:

```
library(sjPlot)
view_df(názevdatasetu, enconding="1250")
```

Tento příkaz zobrazí v pravém dolním okně seznam proměnných, který můžeme zvětšit pomocí ikonky tabulky s šipkou v liště nad tímto oknem (*Show in new window*). Problematický je při tomto zobrazení český jazyk. Příkaz neumí pracovat s diakritikou.

Ekvivalentem pro možnost frequencies v SPSS je v RStudiu jednoduchý příkaz:

```
table(názevdatasetu$názevproměnné)
```

Problematické je opět nezobrazení labelů daných hodnot, což způsobuje náročnější orientaci zejména při větších datasetech. Chceme-li labely zobrazit, můžeme využít **balíček sjlabelled**. Následně zapíšeme:

```
library(sjlabelled)
table(as_label(názevdatasetu$názevproměnné)
```

Tento příkaz je nejsnadnější způsob, jak zobrazit v tabulce frekvencí labely. Ty ale musí být v datasetu již vytvořeny. Nejsou-li labely v datasetu vytvořeny již dříve, můžeme je také vytvořit manuálně. Chceme-li přidat label proměnné, jelikož ji nechceme zobrazovat pouze pod daným kódem, využijeme po nainstalování **balíček Hmisc**:

```
library(Hmisc)
label(názevdatasetu$názevproměnné) <- "Název proměnné, který si zvolíme"
```

Tento příkaz je poměrně snadný. O něco složitější je vytvořit labely hodnot proměnné. Příkaz se liší v případech nominálních a ordinálních hodnot. Pro nominální hodnoty je níže uveden příklad s proměnnou pohlaví:

```
názevdatasetu$pohlaví <- factor(názevdatasetu$pohlaví,
levels = c(0, 1, 2),
labels = c("BEZ ODPOVĚDI", "muž", "žena"))
```

Tímto příkazem přiřadíme k hodnotám 0, 1 a 2 labely BEZ ODPOVĚDI, muž a žena. Musíme dodržovat v závorkách po "c" stejná pořadí. V případě ordinálních proměnných na příkladu proměnné věku:

```
názevdatasetu$věk <- ordered(názevdatasetu$věk,
levels = c(0, 1, 2, 3, 4, 5),
labels = c("BEZ ODPOVĚDI", "15-19", "20-29", "30-44", "45-59", "60+"))
```

Pomocí tohoto příkazu vytvoříme labely pro věkové kategorie, které se nacházejí v datasetu. Při využití základního příkazu table(názevdatasetu$názevproměnné) se nám již vytvoří tabulka frequencies s labely jednotlivých hodnot proměnné.

Těmito příkazy převedeme hodnoty proměnných na tzv. faktor. Program RStudio má poté problém v dalších analýzách s faktory počítat. Vidí je jako text, ne jako čísla, proto se doporučuje pro další postup v analýze převést hodnoty proměnných opět na numerickou škálu. Jestliže chceme provést tento úkon, ale zároveň zachovat naše labely, využijeme příkaz (příklad věku):

```
názevdatasetu$věk <- as_labelled(názevdatasetu$věk, add.labels = TRUE)
```

Při následném využití příkazu table(názevdatasetu$názevproměnné) se již labely nezobrazí. Labely jsme ale vytvořili, takže můžeme využít příkaz s pomocí dříve zmiňovaného balíčku *sjlabelled*:

```
table(as_label(názevdatasetu$názevproměnné).
```

Jestliže chceme hodnoty frekvence převést na procenta, upravíme příkaz následujícím způsobem:

```
table(názevdatasetu$názevproměnné) /počet případů*100
```

Jestliže neznáme přesný počet případů, využijeme příkaz:

```
100*table(názevdatasetu$názevproměnné)/sum(table(názevdatasetu$názevproměnné))
```

Pokud chceme některé proměnné z tabulky vyřadit, protože jsou pro nás nepodstatné, například hodnotu bez odpovědi či odpověď nevím, musíme tyto hodnoty nastavit jako chybějící hodnoty. Jestliže jsme tyto hodnoty pomocí předešlých kroků převedli do textové podoby čili jsme jim dali label a nepřevedli je posléze zpět do numerické podoby, RStudio je stále vidí jako text. Příkaz pro nastavení kterékoliv takové hodnoty na hodnotu chybějící zní:

```
názevdatasetu$názevproměnné <- dplyr::na_if(názevdatasetu$názevproměnné, 'nevím')
```

Jestliže máme místo textu numerickou hodnotu, jednoduše nahradíme tento text číselnou hodnotou:

```
názevdatasetu$názevproměnné <- dplyr::na_if(názevdatasetu$názevproměnné, 0)
```

Pro tento úkon je nutné mít zapnuty balíčky *tidyverse* a *dplyr*. Hodnotu textovou musíme napsat do jednoduchých uvozovek. Chceme-li převést na chybějící hodnotu numerickou hodnotu, uvozovky již třeba nejsou.

5.2.4 Základní vizualizace dat pro deskriptivní analýzu

Pro důkladnou inspekci vlastností proměnných je vhodným nástrojem také vizualizace pomocí různých typů grafů. V rámci výše uvedeného balíčku *sjPlot* je například k dispozici frekvenční graf, který lze vytvořit jednoduchým příkazem:

```
plot_frq(názevdatasetu$názevproměnné) +
xlab("Vzdělání") + ylab("Počet")
```

Za příkaz plot_frq jednoduše napíšeme proměnnou, kterou chceme v grafu zobrazit. Xlab a ylab (lab = label) jsou zkratkami, které RStudiu říkají, jak pojmenovat osy x a y v grafu. Tato funkce zobrazí v grafu nejen počty případů, ale také procentuální podíl dané hodnoty proměnné. Labely zde fungují i v českém jazyce bez problémů.

Zcela jednoduchý černobílý histogram (graf četnosti hodnot) lze vytvořit příkazem:

```
hist(názevdatasetu$názevproměnné)
```

Dále lze využít tzv. density plot – graf hustoty hodnot sloužící pro inspekci četnosti případů pro každou hodnotu proměnné:

```
dens <-density(PISA2015$PV1MATH)
plot(dens, main="Matematická gramotnost")
abline(v=mean(PISA2015$PV1MATH),col="red")
```

Pro snadnou tvorbu různých druhů grafů lze využít oblíbený **balíček ggplot2**. Základní příkaz zní následovně:

```
ggplot(data = názevdatasetu, mapping = aes(x, y))
ggplot(názevdatasetu, aes(x, y))
```

První i druhá verze příkazu vytváří podklad pro další tvorbu grafu (druhá verze je pouze zkrácený první příkaz).

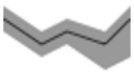
Ggplot se tvoří vrstvením. Další vrstvy přidáváme pomocí symbolu +. Do druhé vrstvy je vhodné definovat, jaký druh grafu chceme zobrazit (kuličkový, histogram, spojnicový atd.):

```
ggplot(názevdatasetu) +
geom_point(aes (x, y))
```

```
ggplot(názevdatasetu) +
geom_point(aes (x, y)) +
geom_smooth(aes(x y))
```

Tento příkaz vytvoří kuličkový graf. Za x a y píšeme proměnné, které budou na osách grafu. Chceme-li do grafu přidat například spojnici, přidáme do příkazu další vrstvu:

Základní typy grafů

geom_point	geom_abline	geom_bar	geom_smooth	geom_density	geom_boxplot
					

Další typy grafů lze nalézt na webové stránce https://ggplot2.tidyverse.org/reference/.

Ggplot také umožňuje odlišení hodnot proměnné pomocí rozdílné velikosti bodů, barvy či symbolů.

V příkazu jednoduše pokračujeme. Definujeme, zda si přejeme odlišit body pomocí velikosti (size), barvy (color),

```
ggplot(názevdatasetu) +
geom_point(aes (x, y), size/color/alpha/shape = kategorizujícíproměnná)
```

postupným blednutím (alpha), nebo tvarem (shape). Kategorizující proměnnou je pak myšlena proměnná, s jejíž pomocí chceme vytvořit rozdílné kategorie, může se jednat například o odlišné ročníky v dané škole. Jestliže si pouze přejeme, aby byl graf jako celek v jiném barevném provedení, musíme závorky v příkazu upravit a jméno barvy vepsat anglicky do uvozovek:

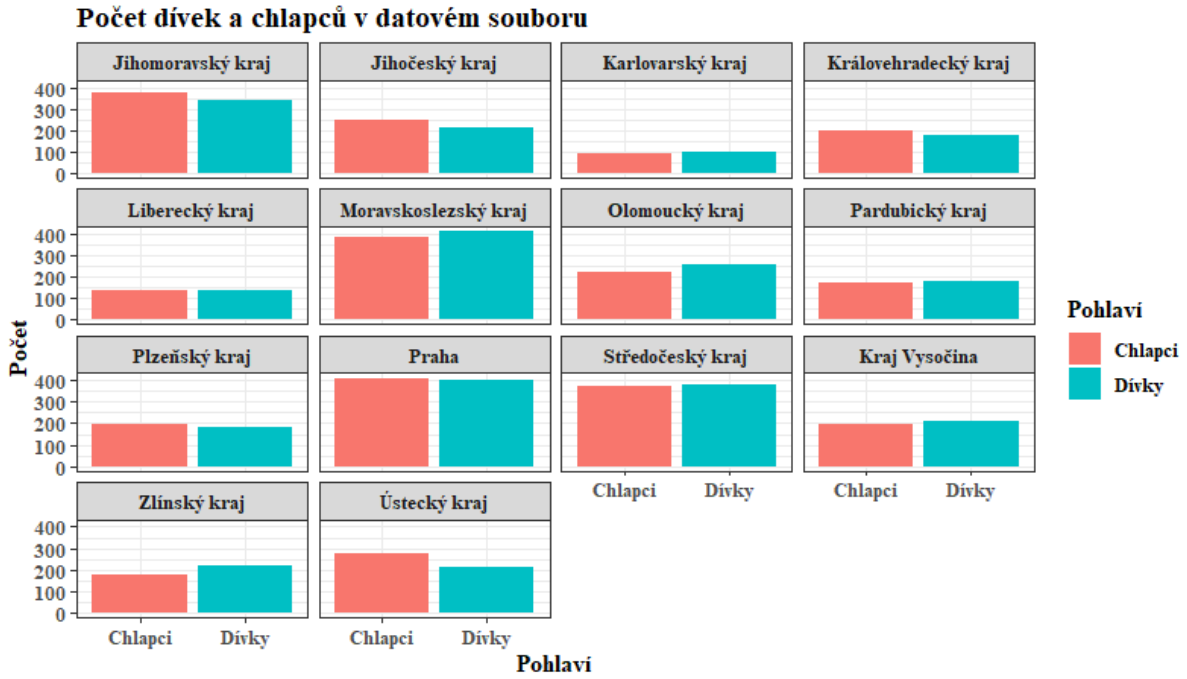
```
ggplot(názevdatasetu) +
geom_bar(aes (x, y)), color=”blue”)
```

Další užitečnou součástí ggplot2 je možnost rozdělení grafu do menších grafů dle určené kategorie. Jedná se o příkazy facet_wrap a facet_grid, které lze využít následovně:

```
ggplot(PISA2015, aes(Pohlaví, fill=Pohlaví)) + geom_bar() + facet_wrap(~Region)+
labs(x="Pohlaví", y="Počet") +
ggtitle("Počet dívek a chlapců v datovém souboru") +
theme_bw() +
theme(text=element_text(family="A", face="bold", size=12))
```

Tento příkaz vytvoří následující tabulku grafů. Jedná se o tabulku, která nám zobrazí, kolik respondentů v daném kraji a jakého pohlaví v datech máme. Jedná se pouze o příklad pro účely demonstrace zmíněných částí příkazu.

V prvním řádku řekneme softwaru RStudio, že chceme vytvořit bar graf (geom_bar), kde bude na ose x pohlaví, které bude rozděleno dle hodnot dané proměnné, zde se jedná o dvě hodnoty chlapce a dívky (fill=Pohlaví), a také chceme grafy rozdělit podle krajů (~Region[7]). S pomocí labs a ggtitle přidáme potřebné labely. Theme_bw vytvoří kolem grafů ohraničení a theme(text=element_text(family="A", face="bold", size12)) definuje styl písma v grafu. Family značí font, kdy "A" je označení pro Times New Roman, face="bolt" napíše text tučně a s pomocí size vybereme velikost písma. Jestliže bude se změnou fontu problém, napíšeme zvlášť příkaz windowsFonts(A=windowsFont("Times New Roman")) a poté příkaz ggplot spustíme znovu.



Využitím příkazu facet_grid se vytvoří podobná tabulka grafů, ty by ale nebyly uspořádány "dlaždicovitě", nýbrž vedle sebe, kde by tvořily sloupce. Pokud bychom chtěli jen dvě řady grafů, můžeme příkaz napsat takto: facet_wrap(~Region, nrow=2).

Balíček ggplot2 nabízí dále množství úprav grafů. Můžeme například přejmenovat legendu proměnné, upravit barevné provedení, odstupňovat graf barevně atd. Proto se doporučuje hledat další možnosti tvorby a úpravy grafů v tomto balíčku na různých webových stránkách, kde ostatní analytičtí pracovníci sdílejí své postřehy a rady.

[7] Vlnovka před Region je kvůli úspoře psaní, lze ji využít při aktivaci balíčku ggplot, funkční je i klasický zápis, zde tedy PISA2015$Region.

6

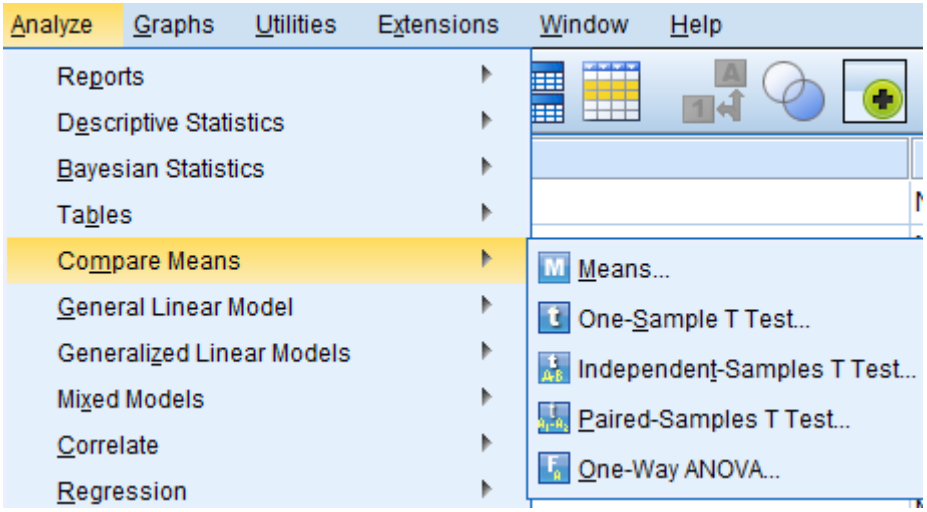
Základní multivariační analýza

6 ZÁKLADNÍ MULTIVARIAČNÍ ANALÝZA

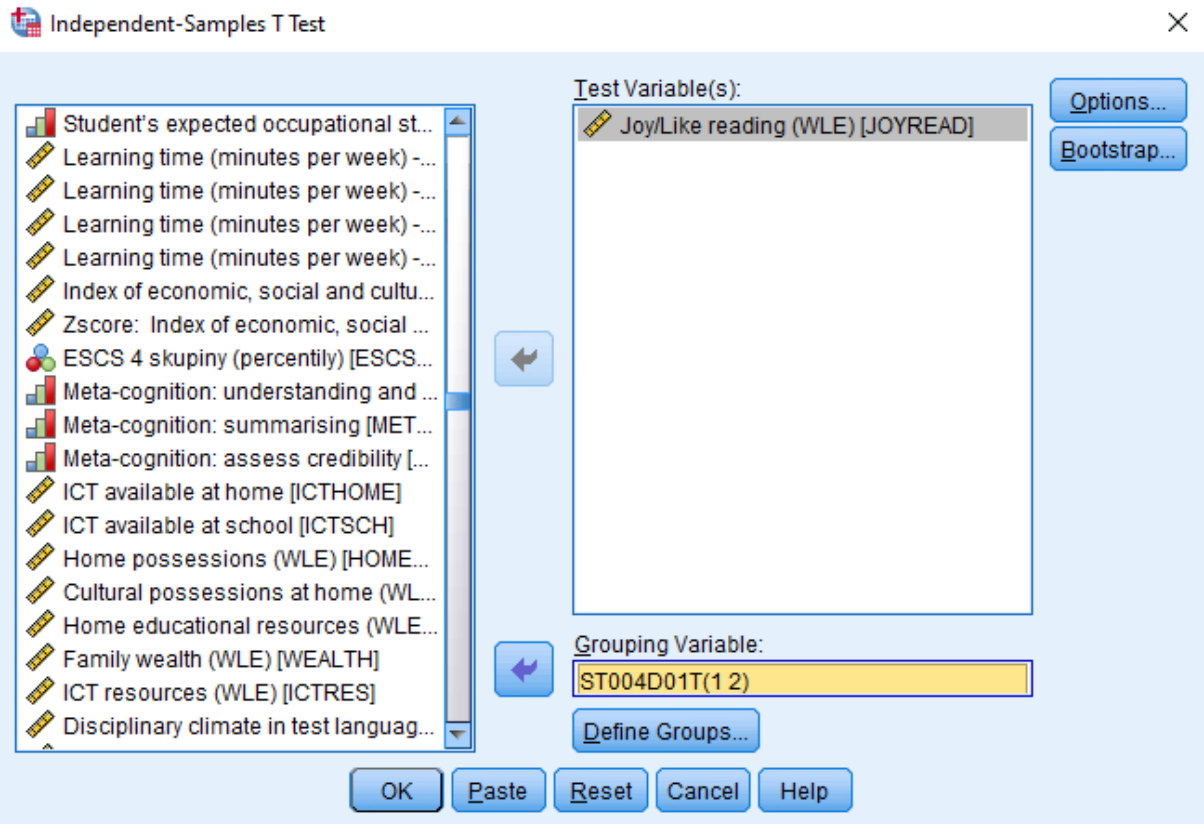
6.1 Multivariační analýza v SPSS

6.1.1 T-test

T-test slouží k testování statistické významnosti rozdílu dvou středních hodnot (dvou aritmetických průměrů). V SPSS jej můžeme najít v záložce *Analyze → Compare Means*.



Při pohledu do nabídky záložky Compare means se nám nabízí tři formy t-testu: t-test pro jediný výběr (*One-Sample T Test*), t-test pro dva nezávislé výběry (*Independent-Samples T Test*) a t-test pro párová data (*Paired-Samples T Test*). T-test pro jediný výběr využíváme v situaci, kdy chceme srovnat průměr nějaké proměnné z výběrového souboru se známou hodnotou (tedy když například známe průměr v populaci), t-test pro dva nezávislé výběry srovnává průměry dvou skupin případů a t-test pro párová data používáme tehdy, kdy pracujeme s párovými daty.



T-test můžeme využít například tehdy, kdy chceme porovnat průměrné hodnoty indexu obliby čtení podle pohlaví, tedy jinak řečeno chceme zjistit, zda se dívky a chlapci liší v tom, jak rádi čtou. V daném případě použijeme t-test pro dva nezávislé výběry. V zadání stačí do okna *Test Variable(s)* vložit proměnnou, jejíž průměrné hodnoty budeme srovnávat (index obliby čtení), do okna *Grouping Variable* následně vkládáme pohlaví. Po vložení této proměnné je ještě nezbytné určit, které skupiny chceme srovnávat, vyplněním příslušných hodnot v rámci *Define Groups*. V nabídce *Options* dále můžeme upravit požadovanou významnost nebo můžeme provést bootstrapping pomocí tlačítka *Bootstrap*.

Group Statistics

	Pohlaví žáka	N	Mean	Std. Deviation	Std. Error Mean
Joy/Like reading (WLE)	dívka	3434	,429513	1,1237343	,0191762
	chlapec	3403	-,379558	1,0078402	,0172767

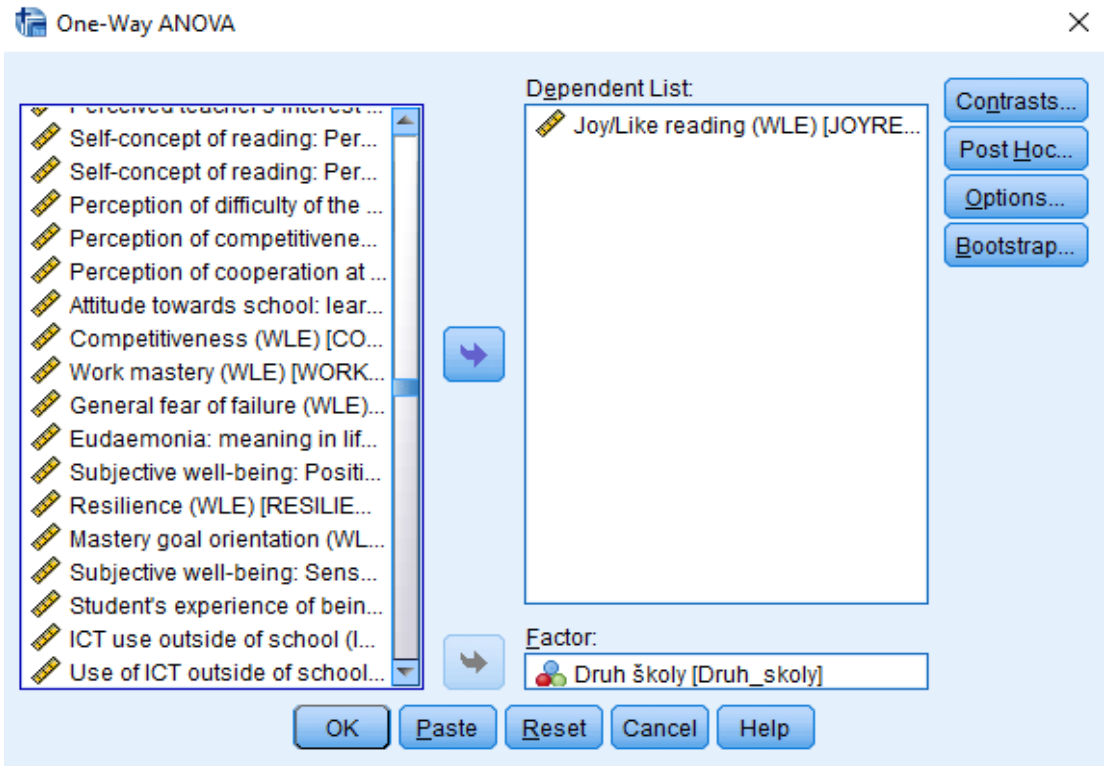
Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Joy/Like reading (WLE)	Equal variances assumed	54,713	,000	31,330	6835	,000	,8090708	,0258238	,7584481	,8596934
	Equal variances not assumed			31,346	6768,125	,000	,8090708	,0258111	,7584729	,8596687

Základním výstupem jsou dvě tabulky. V první vidíme popisnou statistiku pro dvě skupiny, v našem případě pro dívky a chlapce, druhá zobrazuje samotný výsledek t-testu. O tom, do jakého řádku se díváme, rozhodujeme na základě výsledku Leveneho testu, který je v tabulce zobrazen v prvních dvou sloupcích.

6.1.2 ANOVA

T-test slouží ke srovnání průměrů pouze dvou skupin. Pro porovnání průměrů více než dvou skupin se využívá ANOVA. Podobně jako t-test ji lze najít v záložce *Analyze → Compare Means*.



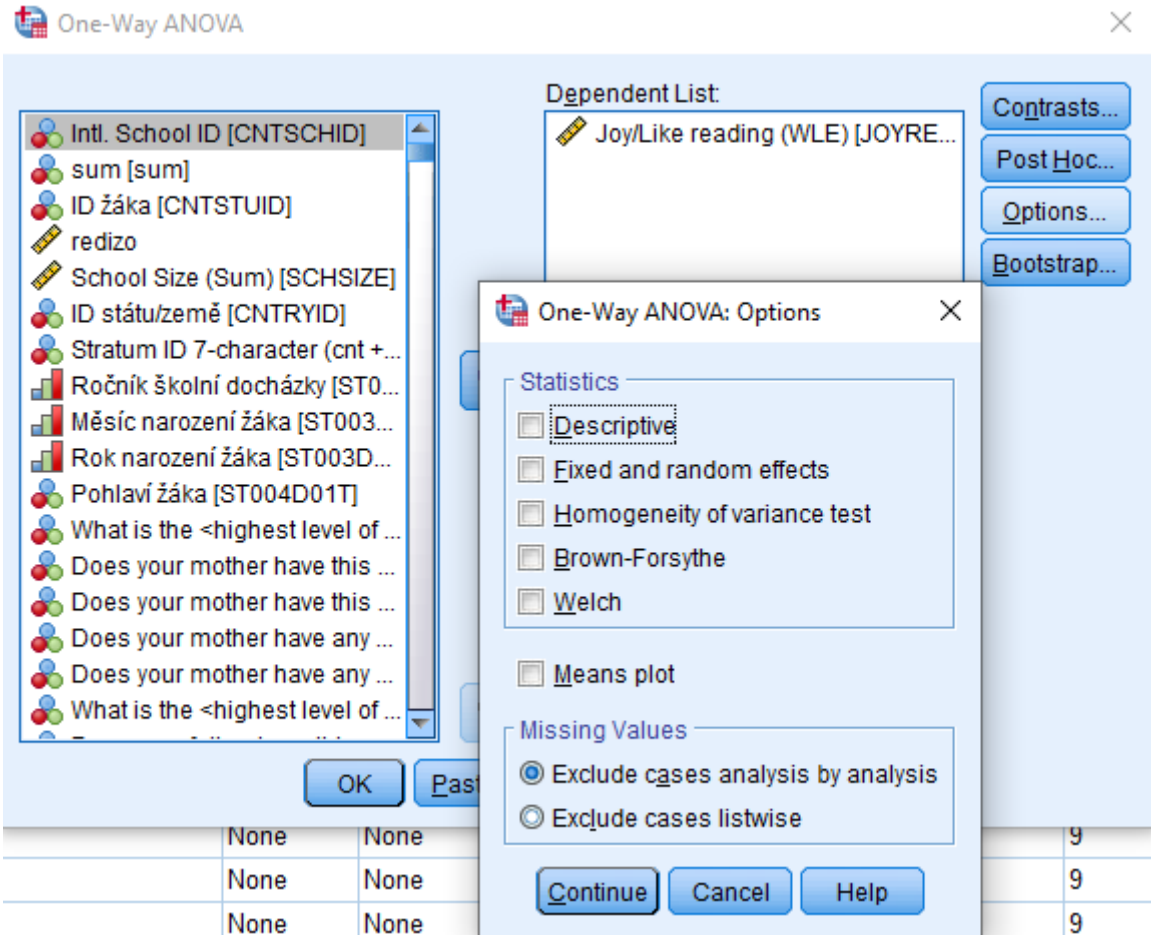
Pro ilustraci nás kromě toho, zda se obliba čtení liší u děvčat a chlapců, může zajímat například to, zda se v tom, nakolik mají čtení rádi, liší žáci odlišných druhů škol. Zadání podobného výpočtu v SPSS na první pohled vypadá obdobně jako v předchozím případě. Do horního políčka označeného jako *Dependent List* znovu vkládáme index obliby čtení, do políčka *Factor* tentokrát vkládáme druh školy.

ANOVA

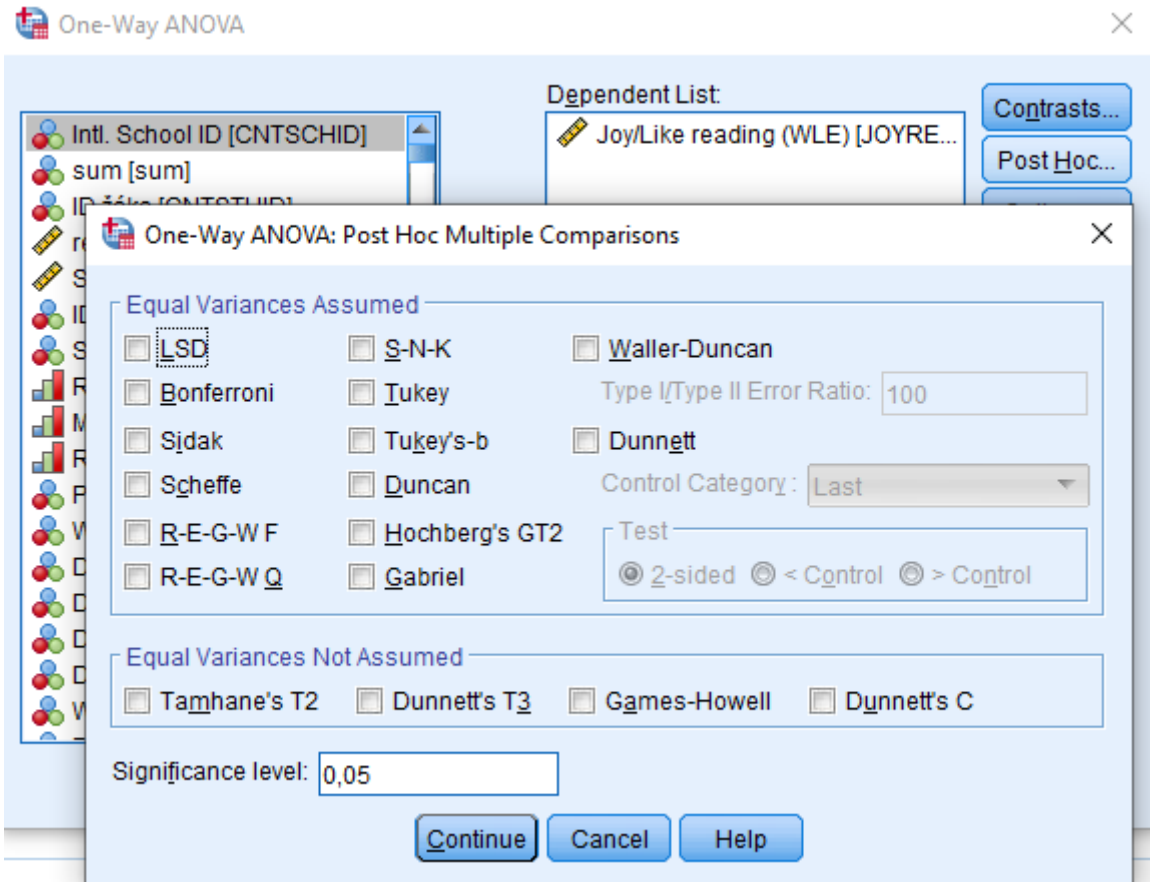
Joy/Like reading (WLE)

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	649,748	8	81,219	67,167	,000
Within Groups	8246,748	6820	1,209		
Total	8896,497	6828			

Bez další specifikace je výstup poněkud strohý, jedná se pouze o jednu tabulku. Ta nám sice umožňuje říct, zda se průměry mezi skupinami statisticky významně liší, avšak nedokážeme na základě ní určit, mezi kterými skupinami rozdíly existují, ani jak jsou velké. Za tímto účelem je potřeba rozšířit požadované výstupy v zadávacím okně.



Pomocí tlačítka *Options* můžeme hned k prvnímu výstupu přidat některé základní statistiky. Zaškrtnutí políčka *Descriptive* například přidá tabulku popisné statistiky, v níž můžeme na první pohled zhodnotit rozdíly v průměrech sledovaných skupin, políčko *Homogeneity of variance test* zase přidá Leveneho test, pomocí jehož výsledků dále rozhodujeme o tom, jakou proceduru mnohonásobného porovnání zvolíme.



Pokud předchozí výstupy nasvědčují tomu, že se průměry skupin skutečně nějak liší, je vhodné vrátit se do zadávacího okna a rozkliknout možnost *Post Hoc*, která nabízí procedury mnohonásobného porovnání. Okno je členěno do dvou základních částí, a sice pro situaci, kdy jsou rozptyly skupin shodné a pro situaci, kdy shodné nejsou. Při volbě tak postupujeme podle výsledku výše zmíněného Leveneho testu. Pokud zaškrtneme procedur více, bude tabulka s výsledky obsahovat porovnání za využití všech takto zvolených procedur.

Multiple Comparisons

Dependent Variable: Joy/Like reading (WLE)

Games-Howell

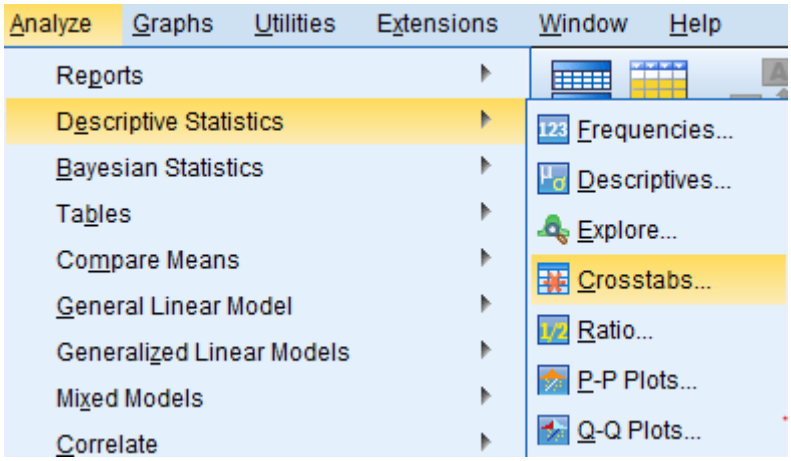
(I) Druh školy	(J) Druh školy	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
ZŠ	VG	-,6016152*	,0372553	,000	-,703299	-,499932
	ČG	-,5356993*	,0456205	,000	-,660298	-,411100
	SŠ s mat.	-,0195096	,0381723	,986	-,123706	,084687
	SŠ bez mat.	,2421232*	,0466945	,000	,114504	,369743
VG	ZŠ	,6016152*	,0372553	,000	,499932	,703299
	ČG	,0659159	,0492067	,666	-,068449	,200281
	SŠ s mat.	,5821056*	,0423934	,000	,466389	,697822
	SŠ bez mat.	,8437384*	,0502040	,000	,706579	,980898
ČG	ZŠ	,5356993*	,0456205	,000	,411100	,660298
	VG	-,0659159	,0492067	,666	-,200281	,068449
	SŠ s mat.	,5161897*	,0499046	,000	,379919	,652460
	SŠ bez mat.	,7778224*	,0566897	,000	,622973	,932672
SŠ s mat.	ZŠ	,0195096	,0381723	,986	-,084687	,123706
	VG	-,5821056*	,0423934	,000	-,697822	-,466389
	ČG	-,5161897*	,0499046	,000	-,652460	-,379919
	SŠ bez mat.	,2616327*	,0508882	,000	,122609	,400657
SŠ bez mat.	ZŠ	-,2421232*	,0466945	,000	-,369743	-,114504
	VG	-,8437384*	,0502040	,000	-,980898	-,706579
	ČG	-,7778224*	,0566897	,000	-,932672	-,622973
	SŠ s mat.	-,2616327*	,0508882	,000	-,400657	-,122609

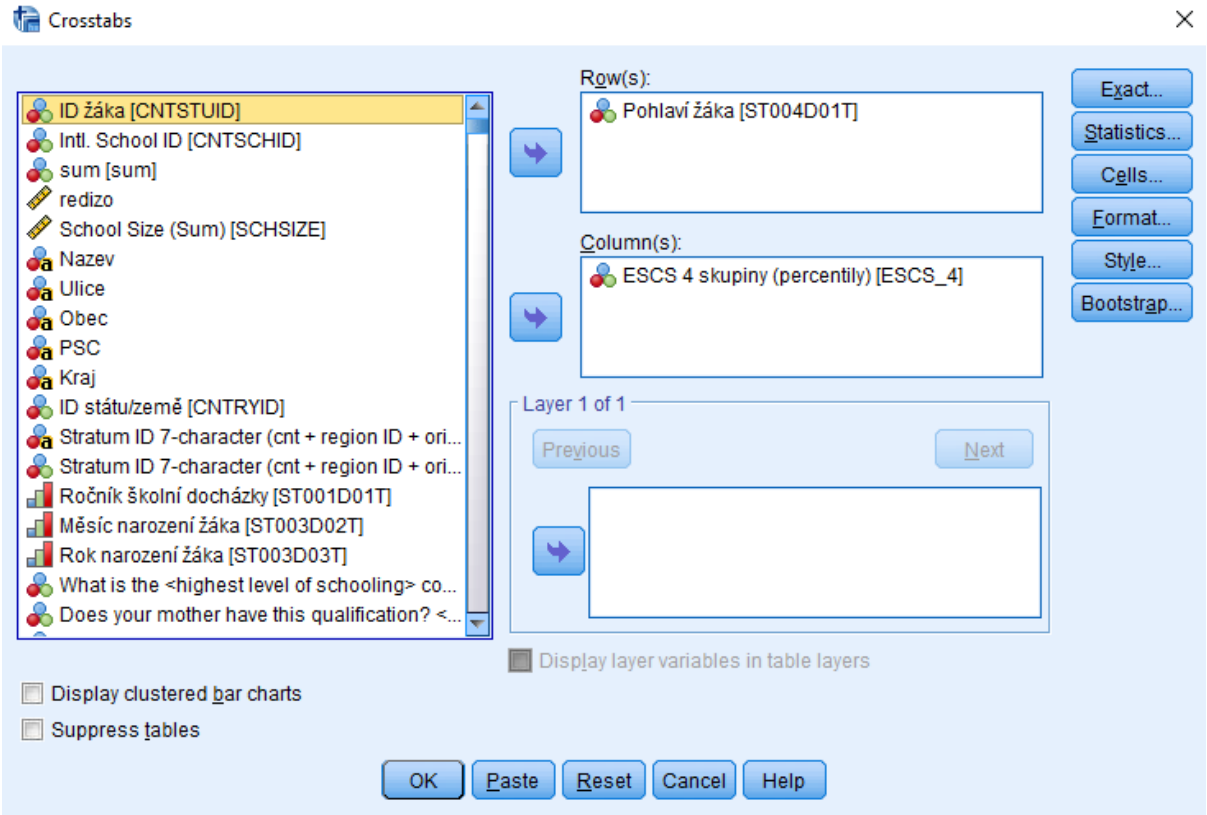
*. The mean difference is significant at the 0.05 level.

Kromě zmíněných tabulek vyžádaných skrze nabídku *Options* výstup nově obsahuje rovněž zde vloženou tabulku sloužící k porovnání jednotlivých skupin mezi sebou. Orientace v této tabulce je poměrně snadná, neboť jsou v ní statisticky významné rozdíly označeny hvězdičkou. Na první pohled je tak jasné, které skupiny se mezi sebou liší a o kolik.

6.1.3 Kontingenční tabulka

Kontingenční tabulku v SPSS můžeme vytvořit po rozkliknutí záložky *Analyze* → *Descriptive Statistics* → *Crosstabs*.





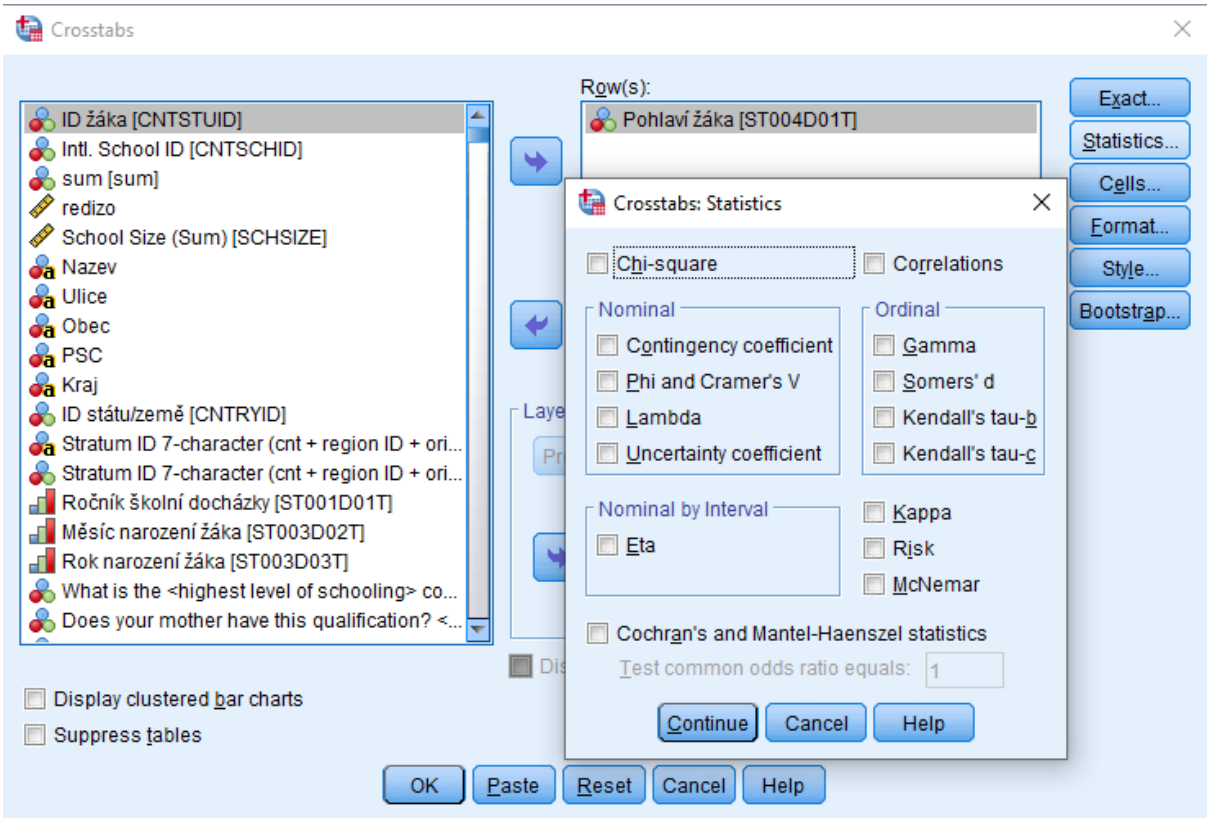
V základním nastavení funkce *Crosstabs* vrací počet případů splňujících zadané parametry, tedy příslušnost do skupiny danou řádkem a sloupcem a celkový počet případů (TOTAL).

Pohlaví žáka * ESCS 4 skupiny (percentily) Crosstabulation

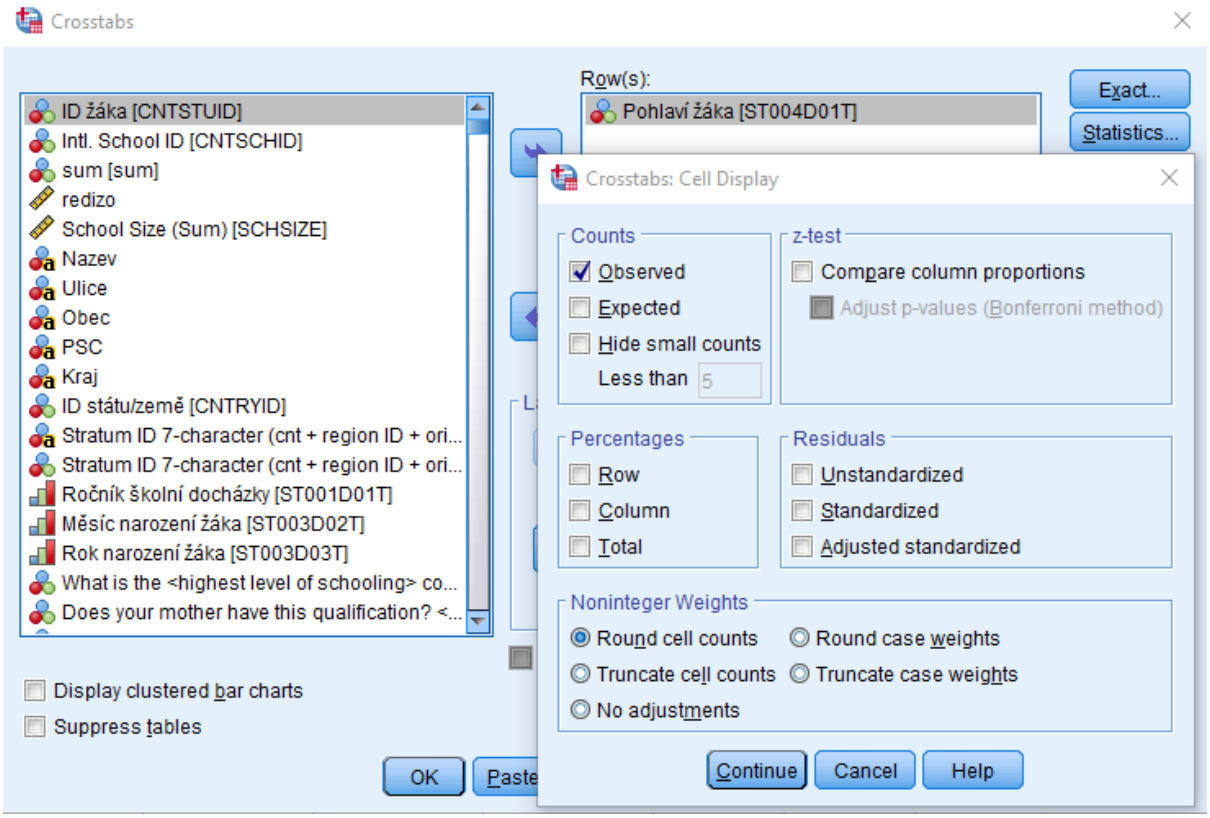
Count

		ESCS 4 skupiny (percentily)				
		Nejnižší SES	2	3	Nejvyšší SES	Total
Pohlaví žáka	dívka	808	900	859	905	3472
	chlapec	920	829	868	822	3439
Total		1728	1729	1727	1727	6911

Pro práci s kontingenčními tabulkami v SPSS jsou stěžejní záložky *Statistics* a *Cells*. V záložce Statistics nalezneme další nástroje, které mohou vypsat informace ve výstupní tabulce, zejména bohatou škálu korelačních koeficientů k základnímu testování předpokládaných vztahů mezi jednotlivými kategoriemi proměnných. Při zaškrtnutí *Correlations* bude program pracovat s běžným *Pearsonovým korelačním koeficientem* a *Spermannovým koeficientem*, při jeho použití získáme de facto korelační matici. Využít lze též koeficient *Eta* k testování síly vztahu a běžný *chí-kvadrátový test*.



Druhá důležitá záložka – Cells – nám pomáhá definovat, v jakém formátu se mají zobrazovat hodnoty v jednotlivých buňkách. Pro většinu dat využíváme zobrazení pozorovaných hodnot (Observed) spolu s nastavením zobrazení řádkových či sloupcových procent (Percentages) a celkových součtů dle toho, jakou informaci chceme získat a jaké proměnné jsme definovali pro řádky a sloupce výstupní tabulky. Nabídka Cells rovněž obsahuje možnost zobrazení reziduí včetně adjustovaných reziduí, která lze testovat z hlediska statistické významnosti. Zaškrtnutím některého z polí v této části se tak v tabulce navíc zobrazí i tyto hodnoty.



Pro ilustraci jsme v následující analýze zadali zobrazení řádkových procent pro jednotlivé skupiny dívek a chlapců dle hodnoty jejich ESCS (SES), zároveň jsme zrušili zobrazení pozorovaných hodnot.

Pohlaví žáka * ESCS 4 skupiny (percentily) Crosstabulation

% within Pohlaví žáka

		ESCS 4 skupiny (percentily)				
		Nejnižší SES	2	3	Nejvyšší SES	Total
Pohlaví žáka	dívka	23,3%	25,9%	24,7%	26,1%	100,0%
	chlapec	26,8%	24,1%	25,2%	23,9%	100,0%
Total		25,0%	25,0%	25,0%	25,0%	100,0%

6.2 Multivariační analýza v RStudiu

6.2.1 T-test

T-test v programu RStudio lze spustit následujícím jednoduchým příkazem. Název proměnné 1 je závislá proměnná ideálně na kontinuální škále, název proměnné 2 je pak proměnná identifikující dvě skupiny, která bude ideálně rekódována do dvou kategorií s tím, že jsou odstraněny chybějící hodnoty.

Levenův test homogenity rozptylu pak provedeme následovně:

```
t.test(jménodatasetu $názevproměnné1~jménodatasetu $názevproměnné2)
```

```
leveneTest(názevproměnné1 ~ as.factor(názevproměnné2), data=jménodatasetu , center=mean)
```

V příkazu je doplněno, že se jedná o kategorickou proměnnou pomocí příkazu as.factor.

Pokud není splněno, zadáme t-test za předpokladu porušení homogenity rozptylu doplněním příkazu var.equal=F.

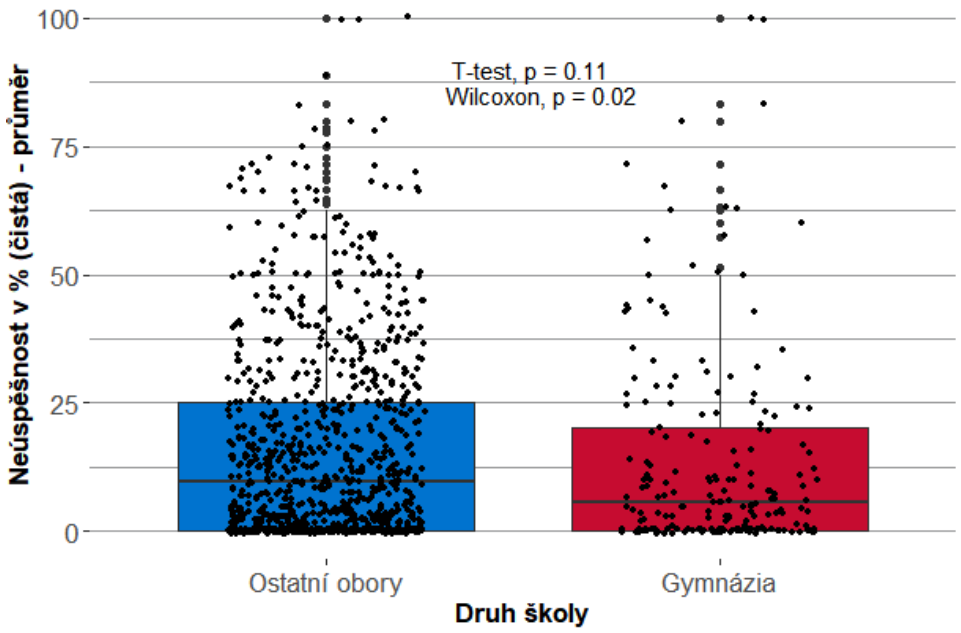
Analýzu porovnávající rozdíly v průměru je vhodné vizualizovat pomocí boxplot grafu se zobrazením případů

```
t.test(jménodatasetu $názevproměnné1~jménodatasetu $názevproměnné2, var.equal=F)
```

a doplněním p hodnot pro určení statistické významnosti na zvolené hladině alfa. Je třeba použít balíček ggpubr a mít spuštěný balíček tidyverse, který obsahuje i knihovnu ggplot2 pro vizualizaci dat.[8]

```
library(ggpubr)
ggboxplot(jménodatasetu, x = "názevproměnné2", y = "názevproměnné1",
          color = "názevproměnné2",
          add = "jitter") +
stat_compare_means(label.x = 1.3, label.y = 5) +   # přidání  p-value, umístění a metoda
labs(x = "Label proměnné X",
     y = "Label  proměnné Y")
```

Příklad Boxplot grafu, zobrazení případů a statistické významnosti dle t-testu a neparametrického Wilcoxonova testu (blíže Metodika sběru a analýzy dat):



6.2.2 ANOVA

Analýzu rozptylu (ANOVA) lze rovněž provést v programu RStudio. Základní příkaz pouze určí, zdali jsou rozdíly mezi skupinami statisticky významné, nicméně nám nepoví, mezi kterými páry skupin. Příkaz začneme názvem

```
ANOVA_TEST <- aov(proměnná1 ~ proměnná2, data = jménodatasetu)
summary(ANOVA_TEST)
```

modelu, který si můžeme libovolně zvolit. V příkladu je název "ANOVA_TEST". Do příkazového řádku pak vložíme název závislé proměnné, která je ideálně na kontinuální škále, a název nezávislé proměnné, která v případě této analýzy je kategorickou proměnnou označující dané skupiny pro porovnání.

Výsledky modelu vyvoláme příkazem summary.

[8] Více informací zde: http://www.sthda.com/english/articles/24-ggpubr-publication-ready-plots/76-add-p-values-and-significance-levels-to-ggplots/

V případě, že naše data jsou neparametrická, tedy máme malý počet případů a proměnné nejsou normálně rozděleny, je vhodné použít alternativní metodu, a to Kruskalův–Wallisův test.

```
kruskal.test(proměnná1  ~ proměnná2, data = jménodatasetu)
```

Pro vizualizaci rozdílů mezi skupinami se jako vhodná forma jeví graf chybových úseček s průměry. V prvním kroku musíme spustit vlastní naprogramovanou funkci pro výpočet standardní chyby, která není integrální součástí základního jazyka R.

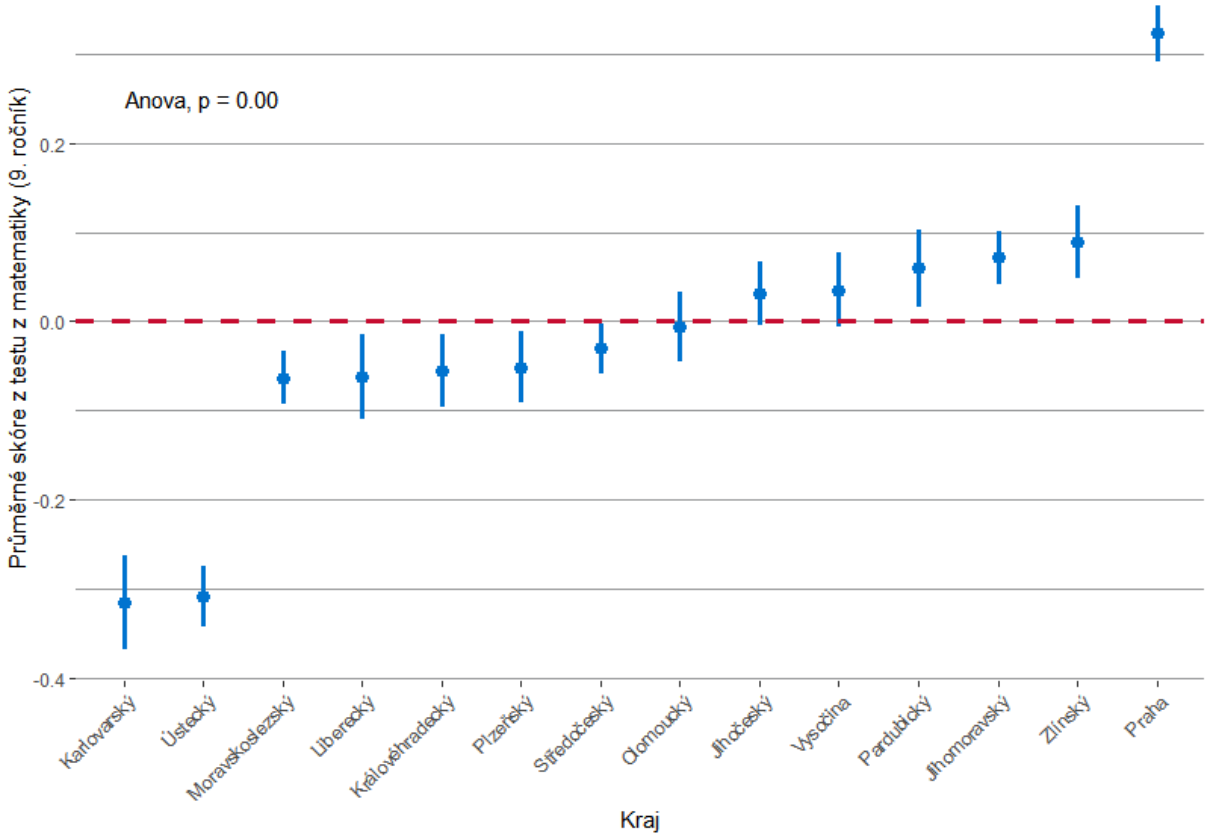
```
stderr <- function(x, na.rm=TRUE) {
if (na.rm) x <- na.omit(x)
sqrt(var(x)/length(x))
}
```

Poté následuje samotné vytvoření grafu.

```
library(tidyverse)
jménodatasetu %>%
  filter(proměnná2!="NA") %>%
  group_by(proměnná2) %>%
  summarise(ERR = mean(proměnná1, na.rm = TRUE),
            SE = stderr(proměnná1)) %>%
  ggplot(aes(x = proměnná2,
             y = ERR)) +
  geom_point(size = 3, color ="#0073CF") +
  geom_errorbar(aes(ymin = ERR- 1.96*SE,
                    ymax = ERR+ 1.96*SE),
                    width = 0, size = 1.25, color ="#0073CF") +
  labs(x = "Label kategorické proměnné X",
       y = "Label závislé proměnné Y") +
  theme(panel.background = element_rect(fill = "white"),
        panel.grid = element_line(color = "#9A9B9C"),
        panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank()) +
  annotate("text", x=1.5, y=2.85, label = "F-test, p = 0.0003") +
  annotate("text", x=3, y=2.85, label = "Kruskal-Wallis, p = 0.00024") +
  theme(plot.title = element_text(size=14,face="bold")) +
  theme(axis.text=element_text(size=13), axis.title=element_text(size=13,face="bold"),
        axis.text.x = element_text(angle = 45, hjust = 1, size=10))
```

Skript je komplexní úpravou grafu, kdy je možné doplnit manuálně hodnoty z výsledků ANOVA či Kruskal-Wallisova testu do grafu. Přepsáním kódu barev ve skriptu lze dále graf graficky upravit, změnit velikost a styl písma popisků grafu do stylu ČSI.

Příklad vizualizace ANOVA pomocí grafu chybových úseček.



6.2.3 Kontingenční tabulka

Stejně jako u funkce frequencies ani tvorba kontingenční tabulky v programu R není zdaleka tak uživatelsky přívětivá jako v programu SPSS. V první řadě si musíme nainstalovat a aktivovat **balíček gmodels**. Příkaz pro tvorbu kontingenční tabulky pak vypadá následovně (*platí nepsané pravidlo, že závislá proměnná patří do sloupců a nezávislá proměnná do řádků*):

```
library(gmodels)
CrossTable(názevdatasetu$názevproměnné1, názevdatasetu$názevproměnné2)
```

V konzole se objeví tabulka plná hodnot, která je velice špatně čitelná. Proto je lepší příkaz dále upravit. Hodnoty, které do tabulky můžeme zahrnout, nalezneme pod příkazem help(CrossTable). Příkaz pak může vypadat například takto:

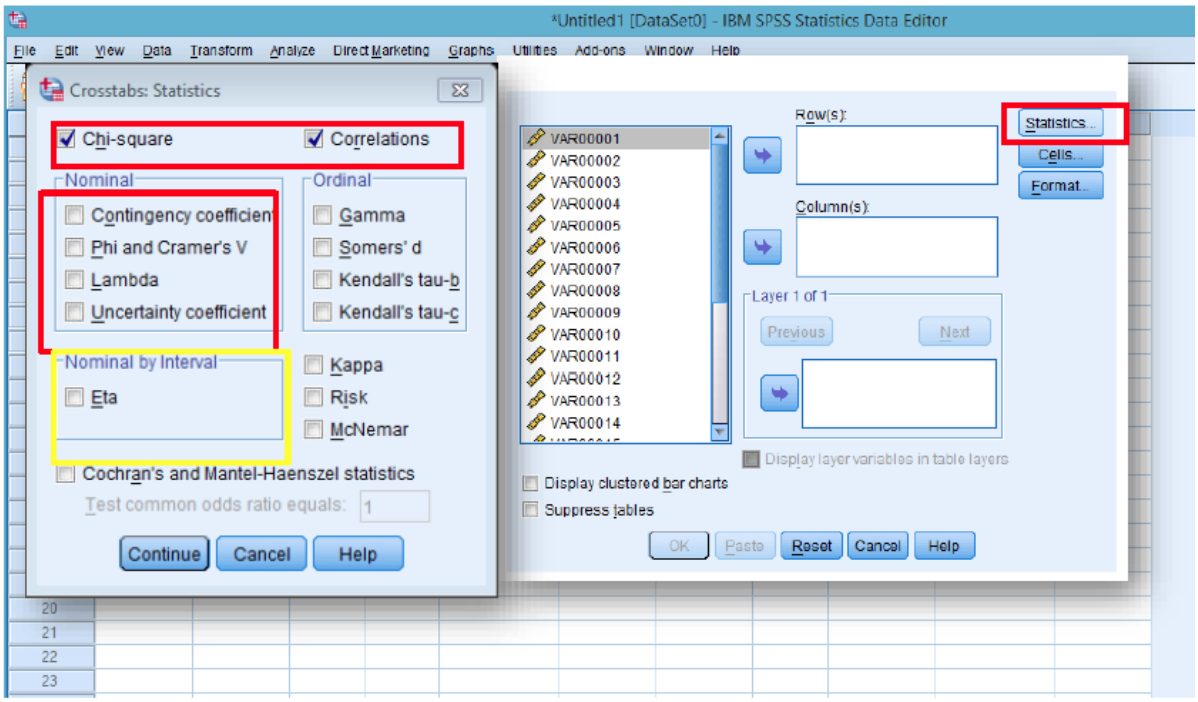
```
CrossTable(názevdatasetu$názevproměnné1,
           názevdatasetu$názevproměnné2,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t=FALSE,
           prop.chisq=FALSE,
           asresid=TRUE,
           format = "SPSS",
           digits=2,
           chisq = TRUE,
           dnn = c("Vzdělání", "Pohlaví"))
```

Příkaz je rozepsán do řádků pro přehlednost, ale jednotlivé atributy tohoto příkazu mohou být i vedle sebe. Prop.r.=T zobrazí řádkové proporce, které dále příkazem format="SPSS" nastavíme jako procentuální (defaultně je v CrossTable formát SAS). Prop.c=F vypneme sloupcové proporce. Prop.chisq = F vypneme zobrazování chí-kvadrátového testu v každé buňce. Asresid=T zapneme adjustovaná standardizovaná rezidua. Digit=2 nastavíme, že hodnoty budou zaokrouhleny na dvě desetinná místa. Chisq=T zapneme zobrazení chí-kvadrátového testu. Dnn=c("Vzdělání","Pohlaví") vytvoříme label pro závislou a nezávislou proměnnou. Funkcí je v příkazu CrossTable více, například resid=TRUE/FALSE nebo sresid=TRUE/FALSE, které přidávají do tabulky rezidua a standardizovaná rezidua, záleží na nás, jaké hodnoty chceme v tabulce zobrazit.

6.3 Korelační analýza

Korelační koeficienty ukazují standardizovanou kovarianci mezi dvěma proměnnými. Obvykle nabývají hodnot od -1 do 1. Příslušné korelační koeficienty bychom měli používat dle charakteru dat a předpokládaného vztahu. Důležitý je počet případů, rozložení případů a předpoklad lineárního nebo nelineárního vztahu. V SPSS najdeme korelační koeficienty v nabídce *Analyze → Correlate → Bivariate correlation*. Zde najdeme Pearsonův korelační koeficient, Spearmanův a Kendallův korelační koeficient.

Existují i další korelační koeficienty pro nominální proměnné a kombinace s jinými úrovněmi měření. Ty pak nalezneme v nabídce kontingenční tabulky *Analyze → Descriptive Statistics → Crosstabs* – následně pak nabídka statistics, kde ale doporučujeme odznačit vytvoření tabulky. V outputu tak nevyjede celá tabulka, která v případě kardinálních proměnných nemá smysl, ale pouze výsledek asociačního vztahu a příslušného korelačního koeficientu.



Korelace je vhodné vizualizovat v rámci korelační matice. Zde je nutné výstup vykopírovat do excelu a dále upravit.

Program RStudio pak nabízí větší míru využití korelačních koeficientů díky balíčkům, které vizualizují korelační matice. Na výběr je mnoho způsobů vizualizace korelací publiku. Základní balíček pro vizualizaci je "corrplot". Následující skript ukazuje vytvoření korelační matice pro tři proměnné. Nejdříve vyselektujeme proměnné do nového dataframu. Poté spočítáme korelace funkcí cor a uložíme do nového objektu. Přejmenujeme naše proměnné dle požadovaného názvu. A pomocí funkce corrplot vytvoříme korelační matici.

Příkaz můžeme libovolně upravovat do graficky požadované podoby. Existuje velké množství grafických designů,

```
dataframe_COR <- dataframe[, c("proměnná1", "proměnná12", "proměnná3")]
Corelace_objekt <-cor(dataframe_COR, use="complete.obs", method="pearson")
        colnames(Corelace_objekt ) <- c("Název proměnné 1", "Název proměnné 2", "Název
        proměnné 3")
        rownames(Corelace_objekt ) <- c("Název proměnné 1", "Název proměnné 2", "Název
        proměnné 3")
library(corrplot)
corrplot(Corelace_objekt, method="number", type="upper", tl.col = "black",  number.cex = 0.7,
cl.cex = 0.8, tl.cex = 0.7)
```

které můžeme použít v galerii korelačních matic, viz:

(https://cran.r-project.org/web/packages/corrplot/vignettes/corrplot-intro.html).

V základním příkazu můžeme upravit typ korelačního koeficientu, například určit method="spearman".

Program RStudio umí i složitější funkce v rámci korelačních koeficientů.[9] Například pomocí balíčků see a ggraph, které jsou potřeba pro Gaussův grafický model. Jedná se o populární metodu vizualizace parciálních korelací. Tyto parciální korelace je vhodné analyzovat před tvorbou regresních modelů. Ukážou nám, jak jednotlivé proměnné navzájem spolu korelují po očištění všech vzájemných parciálních korelací.

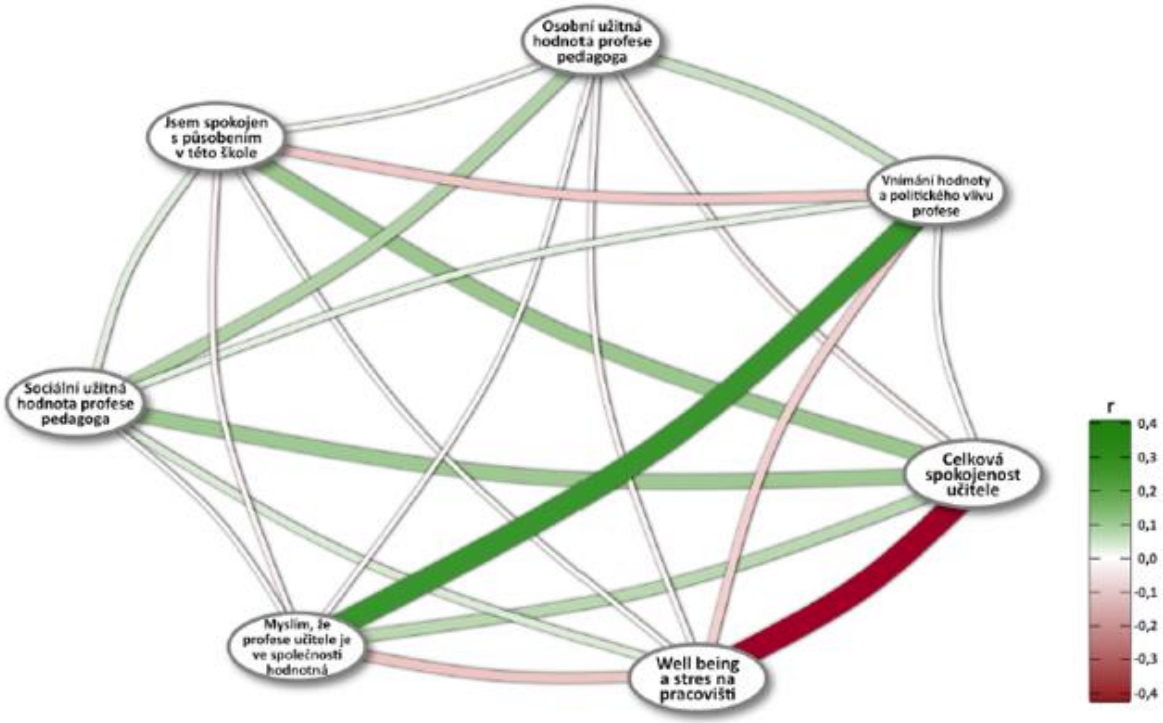
Tento typ analýzy byl nově využit v případě Sekundární analýzy TALIS 2018, ale i v rámci analýzy okresů v prostoru (dosud nevydáno).

```
library(see)
library(ggraph)

KOR_MATICE <- TALIS2018[, c( "T3JOBSA",
                    "T3WELS",
                    "T3PERUT", "T3SOCUT","T3VALP",
                    "TT3G53H", "TT3G53I")]
KOR_MATICE <- na.omit(KOR_MATICE)
write_xlsx(KOR_MATICE, "KOR_MATICE.xlsx")
KOR_MATICE <- read_excel("KOR_MATICE.xlsx")
KOR_MATICE %>%
  correlation::correlation(partial = TRUE) %>%
  plot()
```

[9] Balíček *easystats* a článek o korelačních koeficientech v R viz zde: https://www.r-bloggers.com/2020/03/the-ulimate-package-for-correlations-by-easystats/

Příklad Gaussova modelu – Hodnotový rámec učitelů a celková spokojenost s prací učitele (TALIS 2018)



Pozn.: Individuální úroveň žáka. Naprogramováno v softwaru R pomocí balíčků „correlation" a „ggraph".

7

Regresní modely

7 REGRESNÍ MODELY

Regresní modely jsou základními statistickými technikami analýzy dat. Jejich výhoda spočívá v tom, že na rozdíl od prostého porovnání průměru u skupin, popřípadě analýzy závislostí četností v rámci kontingenčních tabulek, regresní modely umožňují modelovat efekty proměnných na všech úrovních měření a dokážou analyzovat více proměnných současně. Hlavním cílem modelů je pak statisticky kontrolovat efekt dalších proměnných, které mohou být zkorelované současně s dalším prediktorem i predikovanou (závislou) proměnnou. Zatímco software SPSS obsahuje příkazy pro základní regresní modely, složitější specifikace zvládá program RStudio. Ten je v případě regresních modelů velmi variabilní, protože pomocí uživatelských balíčků můžeme využít typy modelů, které do komerčního softwaru nebyly ještě implementovány.

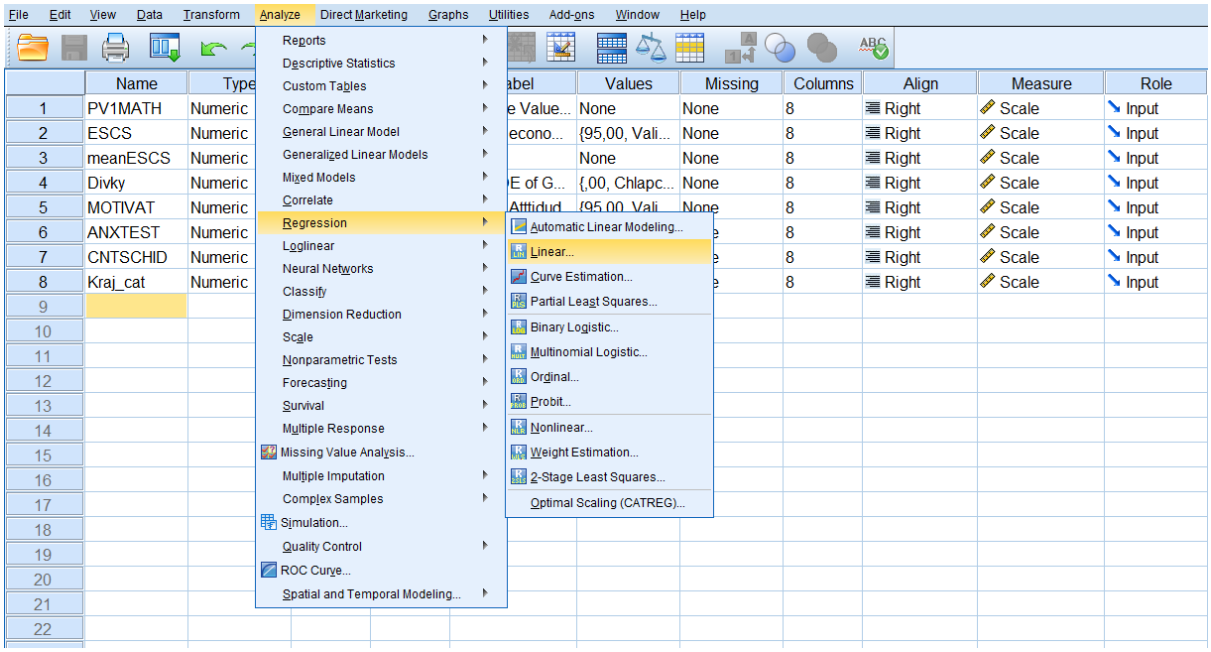
Datový soubor použitý v této kapitole na regresní modely má název PISA2015. Je možné jej stáhnout v příslušné sekci na webu České školní inspekce ve formátu SPSS. Pro jednoduchost pracujeme s vybranými proměnnými, jako je socioekonomický status (ESCS), pohlaví dívka (Gender), motivovanost a úzkost žáka.

7.1 Regresní modely v programu SPSS

Program SPSS nabízí využití celé škály různých základních regresních modelů. Zde je ovšem nutné podotknout, že se příliš nehodí pro analýzu víceúrovňových dat, jakými jsou i mezinárodní šetření gramotností, a to jednak pro absenci příslušných funkcí a jednak pro nemožnost pracovat s vážením na obou úrovních – např. úrovních žák a škola. Jistou možností je využití IDB analyzeru[10], který připraví skript pro SPSS se správným nastavením vah a využitím plausibilních hodnot, nebo jiných statistických programů (STATA, R, Mplus aj.). IDB analyzer v nové rozšířené verzi nabízí i spolupráci s programovacím jazykem R. Modelové příklady v tomto dokumentu jsou uvedeny s daty PISA jen pro ilustraci a jen na jedné úrovni. Používána je rovněž pouze jedna plausibilní hodnota vyjadřující výsledek žáka v testu. V rámci analýzy výsledků vzdělávání žáků je správným postupem práce se všemi plausibilními hodnotami v dané testované oblasti, kterou ovšem SPSS samostatně neumožňuje. V případě použití jednoúrovňových dat (např. data ČŠI sbíraná pouze na úrovni škol) lze regresní modely konstruovat i pomocí SPSS.

Typ využitého regresního modelování se určuje mj. podle typu závisle proměnné – v případě kardinální závisle proměnné (a předpokladu lineárního průběhu dat) lze využít jednoduché lineární regresní modelování, v případě nominálních proměnných se dvěma kategoriemi (typicky pohlaví) pak regresi logistickou.

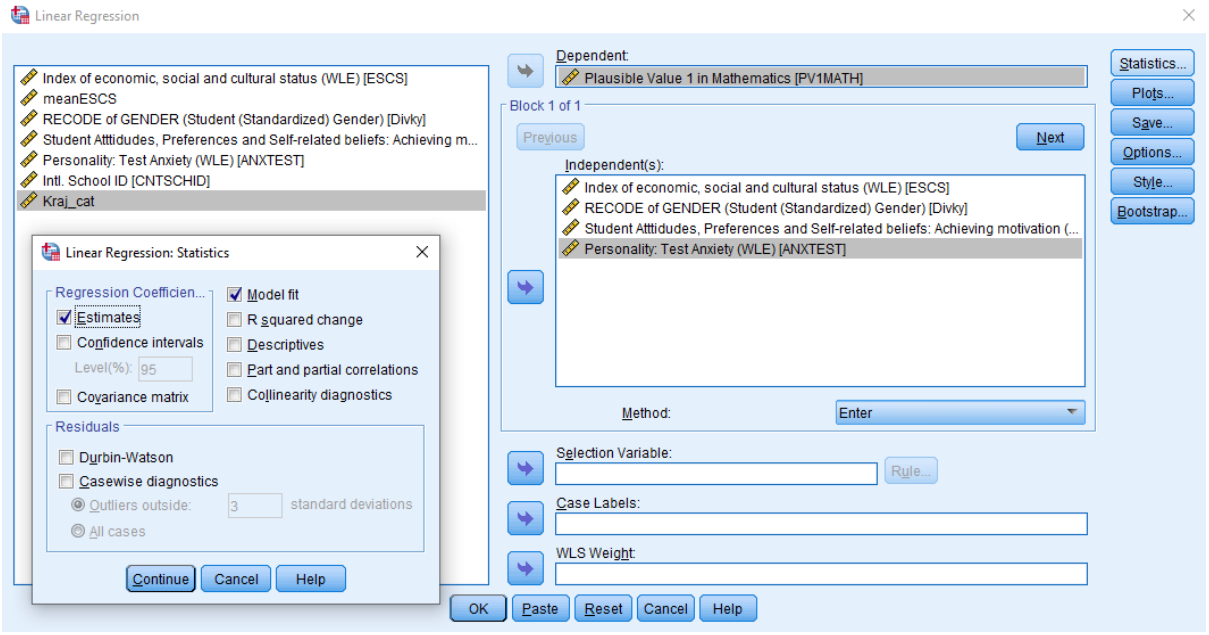
Všechny typy regresních modelů nalezneme v záložce *Analyze → Regression*.



[10] IDB Analyzer je volně dostupný software určený pro usnadnění práce s daty z mezinárodních šetření. Odkaz ke stažení je k dispozici zde: https://www.iea.nl/data-tools/tools.

7.1.1 Lineární regrese

Nejběžnější základní typ regresního modelu je tzv. lineární regrese. Dialogové okno lineární regrese vyžaduje zadání závisle (vysvětlované) proměnné a souboru jedné či více nezávisle proměnných (vysvětlujících). Nastavení SPSS umožňuje použití více metod vstupu, kde je dominantně nejužívanější metoda Enter, při které do výpočtu vstupují veškeré nezávisle proměnné naráz. Další metody pracují s různými algoritmy přidávání nezávisle proměnných. Druhou často používanou metodou je tzv. Stepwise, kdy jsou nezávisle proměnné do výpočtu přidávány postupně jedna po druhé a pro každý krok jsou vypočítávány samostatné výstupní hodnoty např. podílu vysvětlené variace. Pro většinu aplikací postačí využití metody Enter.

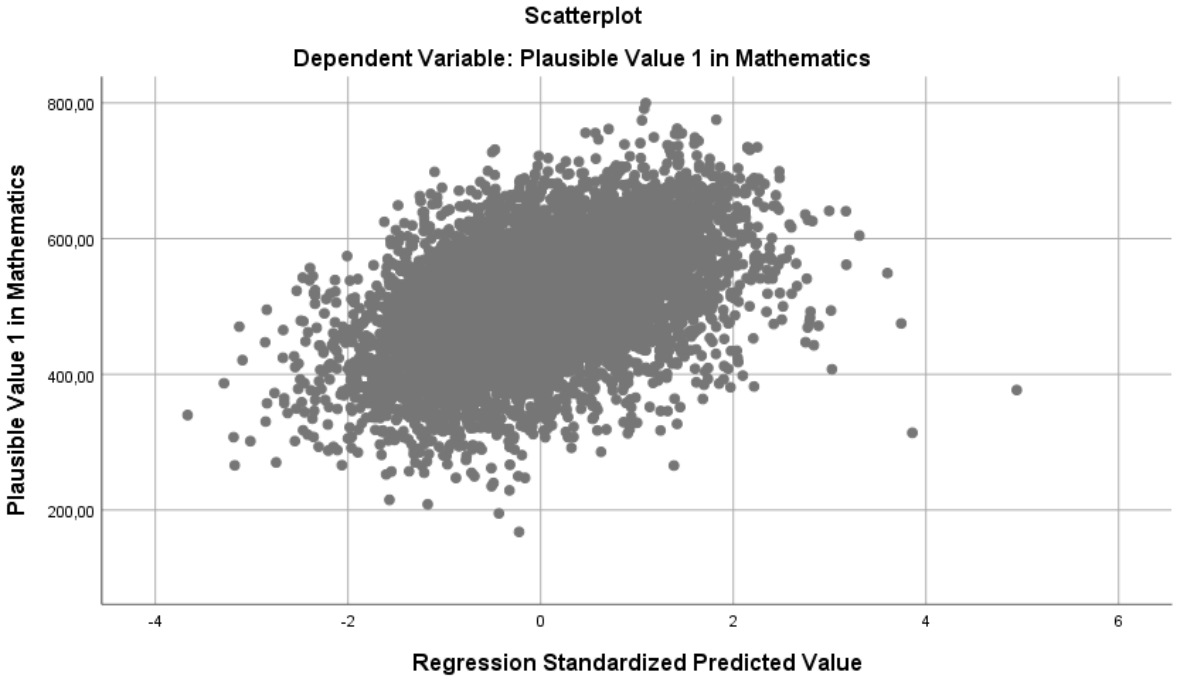


Záložky Statistics, Plots a Save v dialogovém okně umožňují další nastavení a doplňující výstupy. V záložce Statistics lze zvolit např. zobrazení konfidenčních intervalů nebo provedení dodatečných testů jako test Durbin-Watson, který pomáhá odhalovat autokorelace pomocí residuí.

Záložka Plots umožňuje vynesení různých výstupů regresního modelování, od hodnot nezávisle proměnné přes hodnoty predikované modelováním až po varianty residuí, do grafu. Často využívané je vynesení hodnot nezávisle proměnné na osu Y a standardizovaných predikovaných hodnot na osu X, které pomáhá odhalit odchylné případy obtížně vysvětlitelné sestaveným modelem. V modelovém příkladu jsme schopni identifikovat např. odchylné případy (umístěné ve větší vzdálenosti od regresní přímky), jež představují žáky, kteří dosáhli nižšího skóre z matematiky v rámci šetření PISA, než kolik by měli získat dle prediktivního modelu, tedy předpokládaného efektu použitého souboru nezávisle proměnných. Na tyto případy lze cílit mj. další analýzy.

Poslední důležitou záložkou je záložka Save, ve které lze navolit uložení různých výstupů regresního modelu, např. (ne)standardizovaná adjustovaná rezidua nebo predikované hodnoty, ale také různé specializované výstupy, které v rámci tohoto dokumentu dále nerozebíráme.

Výstupem lineární regrese je různý počet tabulek (dle nastavených parametrů), přičemž nás zajímá zejména tabulka s názvem Coefficients, která sleduje vypočítané efekty jednotlivých nezávisle proměnných na hodnotu závisle proměnné, a to vždy za situace, kdy je hodnota všech ostatních nezávisle proměnných konstantní (proto říkáme, že tzv. "kontrolujeme" vliv dalších proměnných). Z výstupní tabulky koeficientů je patrné ve sloupci Sig., že všechny proměnné vyjma pohlaví žáka jsou statisticky významné ($p<0,05$). Ve sloupci nestandardizovaných Beta koeficientů vidíme, že čím vyšší je index ESCS, tím vyšší je skóre žáka z matematiky (dle odhadu modelu v průměru 49,5 bodu). Také motivace zvyšuje skóre žáka o téměř 14 bodů. Naopak pocit úzkosti vede k poklesu skóre žáka o téměř 11 bodů.



Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients		
		B	Std. Error	Beta	t	Sig.
1	(Constant)	514,148	1,462		351,724	,000
	Index of economic, social and cultural status (WLE)	49,547	1,218	,440	40,664	,000
	RECODE of GENDER (Student (Standardized) Gender)	1,455	1,972	,008	,738	,461
	Student Atttidudes, Preferences and Self-related beliefs: Achieving motivation (	13,972	1,209	,126	11,556	,000
	Personality: Test Anxiety (WLE)	-10,976	1,108	-,109	-9,906	,000

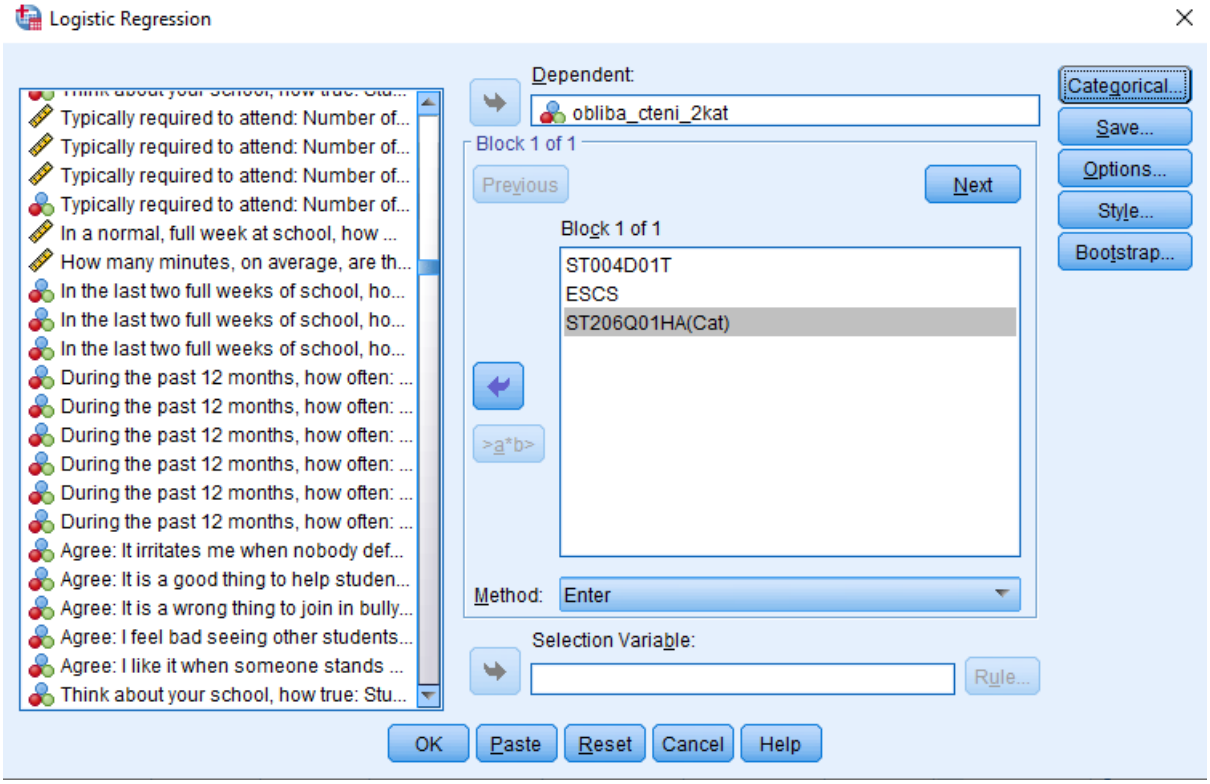
a. Dependent Variable: Plausible Value 1 in Mathematics

Regresní modelování je především náročné na správnou specifikaci. To znamená na rozhodnutí, jaké proměnné vložit do modelu, jakým způsobem kontrolovat neměřitelné faktory (náhodný vs. fixní efekt?), jakým způsobem zohlednit případnou heteroskedasticitu. SPSS má pouze omezené možnosti, protože neumožňuje výpočet celé řady složitějších regresních modelů. S ohledem na tvorbu modelu je důležité zvážit problém multikolinearity. To znamená situaci, kdy jednotlivé proměnné jsou spolu silně zkorelované. Multikolinearita může zkreslit hodnoty koeficientů a může ovlivnit i hodnotu směrodatných chyb. Pro zjištění přítomnosti multikolinearity se používá Variační inflační faktor (VIF), který lze zobrazit po zaškrtnutí v nabídce *Statistics -> Multicollinearity diagnostics*. Pokud je VIF vyšší než hodnota 5, je indikována multikolinearita. Bohužel neexistuje jedno řešení a výzkumník musí zvažovat mnoho faktorů. Za prvé, neměří proměnné, které spolu silně korelují, jeden koncept či faktor? Pokud ano, který indikátor je více validní a který ponecháme v modelu? Za druhé, není vhodnějším postupem vytvoření nové proměnné pomocí faktorové analýzy (viz dále)?

7.1.2 Logistická regrese

Logistické regrese obecně využíváme v situacích, kdy jsou závisle proměnné kategorické (nominální nebo ordinální). V případě, že je závisle proměnná dichotomická, používáme binární logistickou regresi, pokud má více než dvě kategorie, využíváme regresi multinomiální, a pakliže je závisle proměnná ordinální, použijeme regresi ordinální.

Pro ilustraci můžeme využít proměnnou vytvořenou v rámci části rekódování, a sice oblibu čtení, kterou jsme rekódovali do dvou kategorií – nepovažuje/považuje čtení za svůj koníček. Jelikož se jedná o proměnnou dichotomickou, rozklikneme v SPSS záložku *Analyze → Regression → Binary Logistic* (půjde tedy o binární logistickou regresi).



Do prvního okna přetáhneme závisle proměnnou (v našem případě oblibu čtení), do následujícího potom nezávisle proměnné (pohlaví, index socioekonomického statusu a pocit spolupráce žáků ve škole měřený na čtyřbodové ordinální stupnici). Rozkliknutím políčka *Categorical* vpravo lze určit, které z těchto proměnných jsou kategorické. V tomto případě zde jako kategorickou označíme proměnnou pocitu spolupráce žáků ve škole. Nastavení v tomto okně nám dále umožňuje určit, která z kategorií má být kategorií referenční. Jelikož v tomto případě vyšší hodnota značí silnější pocit spolupráce ve škole, změníme nastavení referenční kategorie z poslední (*Last*) na první (*First*) pro usnadnění interpretace. Dichotomické proměnné (jako v našem případě pohlaví) zde specifikovány být nemusí. Další nastavení můžeme provést rozkliknutím tlačítka *Options*. Zde nalezneme například možnost přidání konfidenčních intervalů pro poměry šancí či Hosmer-Lemeshowův test. V rámci možnosti *Save* můžeme nastavit uložení predikovaných hodnot z modelu.

Model Summary

Step	-2 Log likelihood	Cox & Snell R Square	Nagelkerke R Square
1	7272,721[a]	,125	,169

a. Estimation terminated at iteration number 4 because parameter estimates changed by less than ,001.

Classification Table[a,b]

			Predicted		
			obliba_cteni_2kat		Percentage Correct
	Observed		nesouhlas	souhlas	
Step 0	obliba_cteni_2kat	nesouhlas	3640	0	100,0
		souhlas	2377	0	,0
	Overall Percentage				60,5

a. Constant is included in the model.

b. The cut value is ,500

Classification Table[a]

			Predicted		
			obliba_cteni_2kat		Percentage Correct
	Observed		nesouhlas	souhlas	
Step 1	obliba_cteni_2kat	nesouhlas	2874	766	79,0
		souhlas	1202	1175	49,4
	Overall Percentage				67,3

a. The cut value is ,500

Variables in the Equation

		B	S.E.	Wald	df	Sig.	Exp(B)
Step 1[a]	Pohlaví žáka	-1,337	,057	541,879	1	,000	,263
	Index of economic, social and cultural status	,464	,034	185,894	1	,000	1,591
	Think about your school, how true: Students seem to value cooperation.			11,277	3	,010	
	Think about your school, how true: Students seem to value cooperation.(1)	,164	,105	2,432	1	,119	1,178
	Think about your school, how true: Students seem to value cooperation.(2)	,284	,107	6,963	1	,008	1,328
	Think about your school, how true: Students seem to value cooperation.(3)	,371	,135	7,599	1	,006	1,449
	Constant	1,333	,128	107,970	1	,000	3,791

a. Variable(s) entered on step 1: Pohlaví žáka, Index of economic, social and cultural status, Think about your school, how true: Students seem to value cooperation..

I bez dalšího nastavení a přidání požadovaných výstupů je výsledkem provedení logistického modelu celá řada tabulek. První sada tabulek (které zde nejsou prezentovány) je přehledová a zahrnuje počet případů obsažených v analýze a kódování závisle proměnné a kategorických proměnných. Další výstupy jsou tříděny do dvou bloků, počátečního bloku označeného jako *Block 0* a bloku s výsledky provedeného modelu označeného jako *Block 1*. Z prvního bloku je zde vložena pouze klasifikační tabulka úspěšnosti klasifikace bez nezávisle proměnných pro porovnání s klasifikační tabulkou zobrazující úspěšnost klasifikace na základě provedeného modelu, která je součástí druhého bloku. Obecně platí, že čím vyšší podíl případů dokáže model zařadit správně, tím lépe. Z dalších ukazatelů kvality modelu je vložena tabulka s hodnotami ukazatelů pseudo R^2. Nakonec poslední zde uvedená tabulka již zobrazuje výsledek logistického

modelu. V prvním sloupci jsou zobrazeny koeficienty, které jsou v logitech. Poslední sloupec (označený jako *Exp(B)*) prezentuje lépe interpretovatelné (a tak mnohdy upřednostňované) poměry šancí.

7.2 Regresní modely v RStudiu

Základní příkaz pro regresní model v programu RStudio je lm. Závisle proměnná je žákovo skóre z testu matematické gramotnosti. Závisle proměnná od nezávisle proměnné je oddělena vlnovkou.

Pokud chceme do modelu přidat další prediktor, napíšeme "+" a přidáme název proměnné. V našem případě index

```
Model_1 <- lm(PV1MATH ~  Gender, data=PISA2015)
summary(Model_1)
```

socioekonomického statusu žáka "ESCS".

```
Model_2 <- lm(PV1MATH ~  Gender + ESCS, data=PISA2015)
summary(Model_2)
```

Výsledné modely je ale vhodné prezentovat buď v klasické výsledné tabulce, nebo pomocí grafu regresních koeficientů. Pro rychlé zobrazení více modelů v jedné přehledné tabulce můžeme využít funkci screenreg z balíčku "texreg". Stačí vypsat názvy modelů, které jsme si uložili.

V případě, že chceme výsledky regresního modelu uložit do klasické tabulky, můžeme opět použít balíček texreg. Níže

```
Model_1 <- lm(PV1MATH ~  Gender, data=PISA2015)
summary(Model_1)

Model_2 <- lm(PV1MATH ~  ESCS + Gender, data=PISA2015)
summary(Model_2)

Model_3 <- lm(PV1MATH ~ ESCS + Gender + MOTIVAT + ANXTEST, data=PISA2015)
summary(Model_3)

library(texreg)
screenreg(list(Model_1,Model_2, Model_3),
        digits = 3,
        single.row = FALSE) #
```

je defaultní skript pro zobrazení předchozích tří modelů. Skript lze variabilně upravit pro jakýkoliv model, kdy stačí upravit názvy modelů, názvy proměnných a změnit počet koeficientů a jejich pořadí. Zde balíček automaticky dává konstantu na první místo, proto je nutné pořadí změnit tak, že konstanta označená číslem 1 bude na konci. Skript uloží výsledky regresních modelů do tabulky ve formátu html do složky, která byla nastavena jako pracovní prostředí (set working directory). Soubor v html lze otevřít v prohlížeči, vybrat tabulku myší a vykopírovat ji například do MS Word.

```
library(texreg)
htmlreg(list(Model_1,Model_2, Model_3), # názvy modelů
      file = "./PISA2015.html", # output file
      custom.model.names = c("Model_1","Model_2", "Model_3"), # název modelů
      custom.coef.names = c("Konstanta","SES žáka",
                  "Dívky","SES školy","Motivace", "Úzkost"), # názvy proměnných
      reorder.coef = c( 2,4, 3, 5, 6, 1),
      custom.gof.names = c("R2","Adj.R2","Počet případů", "RMSE"), # názvy statistik
      digits = 3, # desetinná místa pro koeficienty a směrodatné chyby
      single.row = FALSE) # zobrazení směrodatných chyb pod koeficientem
```

Někdy je vhodné vizualizovat regresní model pomocí grafu regresních koeficientů pomocí balíčků "coefplot". V grafu jsou vyneseny hodnoty koeficientů a spolu s nimi intervaly spolehlivosti. Pokud protnou nulovou osu, koeficient není statisticky významný.

```
library(coefplot)
coefplot(Model_3, title = "ZP: Matematická gramotnost",
       legend.reverse = TRUE, intercept=FALSE,
       xlab = "Standardizované koeficienty (2SD Gelman 2008) + 90% k.i.", ylab="Koeficienty",
          pointSize = 3, color = "black", zeroColor = "black",  innerCI = 1.645,   outerCI = 1.645,
          sort = c("magnitude"),
       coefficients=c("ESCS ",
                  "Gender ",
                  "MOTIVAT ",
                  "ANXTEST"),
       newNames=c(ESCS="SES žáka",
                Gender="Dívka",
                MOTIVAT="Motivace",
      ANXTEST="Nervozita úzkost")) +
 theme(legend.position="bottom") +
 theme(plot.title = element_text(size=10,face="bold")) +
 theme(axis.text=element_text(size=10, face="bold"),
axis.title=element_text(size=8,face="bold")) +
 scale_color_manual(values=c("red","blue"))
```

V případě, že jsou proměnné na rozdílné škále, může být graf méně přehledný, protože koeficienty nebudou ve srovnatelném měřítku. Z tohoto důvodu je vhodné při použití grafu regresních koeficientů proměnné standardizovat a graf vytvořit na modelu ze standardizovaných hodnot.

```
PISA2015$ESCS <- scale(PISA2015$ESCS)

# Možné je použít i Gelmanovu metodu dvou směrodatných odchylek. Pro výpočet je nutné
vytvořit funkci.

twoSD <- function (x) {
 (x - mean(x, na.rm = TRUE))/(2 * sd(x, na.rm = TRUE))
}
```

Standardizace do dvou směrodatných odchylek:

```
PISA2015$ESCS <- twoSD(PISA2015$ESCS)
```

Regresní modely obecně mohou sloužit k predikci hodnot na základě nezávislé proměnné, při kontrole všech ostatních třetích proměnných v modelu. Pro zobrazení grafu predikovaných hodnot Y v závislosti na hodnotách X slouží balíček "ggeffect". Na příkladu níže zobrazíme graf predikovaných hodnot z plného (třetího) modelu, který kontroluje i efekt motivovanosti a úzkosti žáka.

```
library(ggeffects)
dat <- ggpredict(model = FULL,
                 terms = "ESCS",
                 ci.lvl = 0.95)
plot(dat)
```

7.2.1 Interakční efekty v regresních modelech

Program RStudio umožňuje velmi dobrou analýzu tzv. interakčních efektů, které předpokládají, že efekt nezávislé proměnné na závislou proměnnou bude podmíněn hodnotou další třetí proměnné Z. Na modelovém příkladě si ukážeme interakci mezi individuálním SES žáka a průměrným SES školy. Nejdříve pomocí úpravy datasetu vytvoříme novou proměnnou, kterou agregujeme na úroveň školy jako průměr individuálních hodnot indexu ESCS žáků. Použijeme balíček pro úpravu proměnných "dplyr" ze sady tidyverse.

Samotnou interakci provedeme jednoduše tak, že v rovnici regresní přímky pronásobíme dvě proměnné, které by interakci měly tvořit. Oproti jinému softwaru R již automaticky ví, že se jedná o interakci a model spočítá se všemi

```
library(dplyr)
PISA2015<- PISA2015%>%
  dplyr::group_by(CNTSCHID) %>% # rozdělení dat dle ID školy
  dplyr::summarise(meanESCS = mean(ESCS, na.rm = TRUE))
```

konstitutivními prvky interakce. Tedy "X", "Z" a "XZ".

Ze samotné tabulky regresních koeficientů je obtížné určit výsledky interakce, protože efekty koeficientů nelze

```
INTERAKCE_MODEL <- lm(PV1MATH ~ ESCS*meanESCS + Divky + MOTIVAT +
ANXTEST, data=PISA2015)
summary(INTERAKCE_MODEL)
```

interpretovat samostatně bez interakce, tedy hodnot Z. Máme několik možností, buď si vytvoříme tabulku při určitých hodnotách proměnných a efekt spočítáme dle rovnice regresní přímky, nebo necháme software R spočítat interakci pomocí grafu predikovaných hodnot či pomocí grafu marginálního efektu.

V případě grafu predikovaných hodnot použijeme balíček "interactions". Balíček používá grafiku ggplot, takže lze graf graficky upravit. Skript níže ukazuje grafickou úpravu pro využití grafu v Sekundárních analýzách PISA. Jako první doplníme název uloženého modelu, doplníme dvě proměnné tvořící interakci, kdy na ose X je meanESCS a dvě přímky predikovaných hodnot pro žáky s minimální hodnotu ESCS a maximální hodnotou ESCS. Index nabývá hodnot od -4 po 4, proto doplníme tyto hodnoty. Hodnoty je nutné vždy doplnit dle deskriptivní statistiky proměnných tvořících

interakci. Výsledný graf pak ukazuje to, jak se pro dvě skupiny žáků mění predikované hodnoty výsledků testů matematické gramotnosti v závislosti na průměrném ESCS školy.

Alternativním způsobem zobrazení interakce je graf marginálního efektu. Ten je však komplikovanější pro představení

```
#1) graf predikovaných hodnot
library(interactions)
interactions::interact_plot(INTERAKCE_MODEL, pred = "meanESCS", modx = "ESCS",
modx.values = c(-4, 4), modx.labels = c("min", "max"),
                    interval= TRUE, legend.main = "SES žáka", line.thickness = 0.8) +
theme_gray() + xlab("Průměrné SES školy") + ylab("Predikované hodnoty bodů z testu z
matematiky") + theme(plot.title = element_text(size=16,face="bold")) +
theme(axis.text=element_text(size=13), axis.title=element_text(size=13,face="bold")) +
theme(legend.position="top")
```

vztahu. Graf ukazuje, jak se mění hodnota koeficientu ESCS žáka v závislosti na hodnotě průměrného ESCS školy. Obecně pozorujeme několik možných situací, buď se hodnota koeficientu snižuje, nebo zvyšuje, nebo zde není interakce a přímka je vodorovná. Důležitá je také interpretace konfidenčních intervalů, které nám ukazují, při jakých hodnotách proměnné Z (v našem případě průměrné ESCS školy) je efekt X (individuální SES žáka) statisticky významný.

```
# 2) graf marginálního efektu
# pozor balíček může kolidovat s jinými balíčky (řešení jméno balíčku před příkazem "interplot::")
library(interplot)
interplot::interplot(m = INTERAKCE_MODEL, var1 = "ESCS", var2 = "meanESCS", hist =
TRUE) +
xlab("Průměrné SES školy") + ylab("Marginální efekt individuálního SES žáka na počet bodů z
matematiky") + theme(plot.title = element_text(size=14,face="bold")) +
theme(axis.text=element_text(size=13), axis.title=element_text(size=13,face="bold"))
```

```
# Multikolinearita
car::vif(název_modelu)
# Diagnóza vlivných bodů
influencePlot(název_modelu)
```

Regresní analýzu je nutné doplnit o tzv. regresní diagnostiku. V prvé řadě o test multikolinearity, který zjišťuje, jestli jsou spolu nezávislé proměnné výrazně zkorelovány. Pro test multikolinearity je možné použít balíček car. Z téhož balíčku můžeme použít příkaz influencePlot, který nám identifikuje případy, jež výrazně ovlivňují výsledné koeficienty modelů.

Častým problémem regresních modelů je problém heteroskedasticity. Jedná se o jev, kdy se zvyšující se hodnotou nezávislé proměnné rostou hodnoty chyb regresních modelů. To může mít za následek to, že koeficient dané proměnné nebude statisticky významný, protože jeho směrodatné chyby budou příliš velké právě z důvodu heteroskedasticity. Hrozí nám tak chyba typu II, kdy přijímáme nulovou hypotézu, ačkoliv ve skutečnosti neplatí. Heteroskedasticitu lze identifikovat vizuálně vynesením reziduálních hodnot a hodnot proměnné X do bodového grafu. Pro zjištění heteroskedasticity existují i testy. Nejpoužívanější je Breusch-Paganův test.

Nulová hypotéza testu říká, že je zde homoskedasticita. Pokud vyjde test statisticky významný (p<0,05), zamítáme nulovou hypotézu a zjišťujeme, že reziduální hodnoty odpovídají heteroskedasticitě.

```
# Heteroskedasticita
library(lmtest)
bptest(název_modelu)

# Regresní model s robustními směrodatnými chybami
library(sandwich)
coeftest(food.ols, vcov = vcovHC(název_modelu, "HC1"))
```

Jaké máme možnosti v případě heteroskedasticity? V případě hierarchických modelů jsou automaticky počítány směrodatné chyby dle klastru vyšší úrovně. Pokud ale nemáme hierarchická data, existují možnosti, jakým způsobem spočítat směrodatné chyby tak, aby byly rezistentní vůči heteroskedasticitě, a snížili tak riziko chyby II. typu. V RStudiu máme možnost dopočítat tzv. robustní směrodatné chyby, respektive regresní model s robustními směrodatnými chybami. Tento typ modelu je nejčastěji používán v případě přítomnosti heteroskedasticity.

Balíčky použité v této kapitole

coefplot, https://cran.r-project.org/web/packages/coefplot/coefplot.pdf

interactions, https://cran.r-project.org/web/packages/interactions/interactions.pdf

texreg, https://cran.r-project.org/web/packages/texreg/texreg.pdf

interplot, https://cran.r-project.org/web/packages/interplot/vignettes/interplot-vignette.html

car, https://cran.r-project.org/web/packages/car/index.html

lmtest, https://cran.r-project.org/web/packages/lmtest/index.html

sandwich, https://cran.r-project.org/web/packages/sandwich/sandwich.pdf

7.2.2 Hierarchické regresní modely v RStudiu

Program RStudio je velmi vhodným nástrojem pro hierarchické modelování a obsahuje mnoho funkcí, které běžný software nezvládá. Jedná se například o vynesení náhodných koeficientů do grafu nebo výpočet různých testovacích statistik, které hodnotí hierarchické regresní modely.

Skript pro hierarchický model se liší oproti příkazu pro lineární regresi v tom, že specifikujeme funkci lmer a přidáme proměnnou, která identifikuje skupinu, klastr či hierarchii. Technicky se jedná o druhou či další úroveň, kterou modelujeme jako náhodnou. Můžeme modelovat náhodnou konstantu, tedy rozdíly mezi skupinami v průměru (konstantě), nebo i náhodnou směrnici. V druhém případě se jedná o situaci, kdy předpokládáme, že efekt proměnné se bude lišit napříč skupinami. Například můžeme modelovat jako náhodný individuální SES žáka. Příklady skriptů jsou opět vytvořeny pro dataset PISA 2015.

Při použití hierarchických modelů nejdříve začínáme tzv. nulovým modelem, který je technicky podobný analýze rozptylu. Nulový model je bez prediktorů (nutné přidat "1" pro označení konstanty) a pouze nám ukáže varianci na

```
# Porovnání modelů
library(texreg)
screenreg(list(NULL_MODEL,SES, FULL, RANDOM_ESCS),
        digits = 3,
        custom.model.names = c("Null","SES", "FULL"),
        single.row = TRUE) #
```

```
library(lme4)
NULL_MODEL <- lmer(PV1MATH ~ 1 + (1| CNTSCHID), data=PISA2015, REML = TRUE)
summary(NULL_MODEL)

library(ICC)
ICCest(CNTSCHID, PV1MATH, data = PISA2015, alpha = 0.05, CI.type = c("THD", "Smith"))
SES <- lmer(PV1MATH ~  ESCS + Divky + (1|CNTSCHID), data=PISA2015, REML = TRUE)
summary(SES)
FULL <- lmer(PV1MATH ~ ESCS + meanESCS + Divky + MOTIVAT + ANXTEST +
(1|CNTSCHID), data=PISA2015, REML = TRUE)
summary(FULL)
```

první a druhé úrovni. Snahou modelování je pak vysvětlit varianci v závislé proměnné na rozdílné úrovni. Kolik procent variance je dáno rozdíly mezi školami (druhá úroveň), nám ukazuje tzv. vnitrotřídní korelační koeficient. Spočítat jej můžeme z nulového modelu ručně, kdy porovnáme varianci na první a druhé úrovni. Nebo pomocí balíčku ICC. Celkem vytvoříme cvičně tři modely: první je tedy nulový model, druhý modeluje efekt ESCS a pohlaví, třetí model přidává průměrné SES na úrovni školy a další individuální charakteristiky žáků.

Porovnat modely můžeme pomocí funkce screenreg balíčku texreg. Funkce automaticky zobrazí všechny potřebné statistiky pro popis hierarchického modelu.

Výstup příkazu

```
=============================================================================================
                          Null                    SES                    FULL
---------------------------------------------------------------------------------------------
(Intercept)                489.463 (3.644) ***     500.565 (3.361) ***    523.852 (2.260) ***
ESCS                                                21.813 (1.180) ***     16.092 (1.183) ***
Divky                                              -12.107 (1.726) ***     -7.767 (1.702) ***
meanESCS                                                                   96.026 (4.335) ***
MOTIVAT                                                                    14.199 (0.988) ***
ANXTEST                                                                   -11.847 (0.911) ***
---------------------------------------------------------------------------------------------
AIC                      78323.802               76751.802              74298.974
BIC                      78344.317               76785.917              74353.382
Log Likelihood          -39158.901              -38370.901             -37141.487
Num. obs.                 6894                    6788                   6641
Num. groups: CNTSCHID      344                     344                    333
Var: CNTSCHID (Intercept) 4239.884                3312.351               935.857
Var: Residual             4363.336                4168.711               3900.375
=============================================================================================
*** p < 0.001, ** p < 0.01, * p < 0.05
```

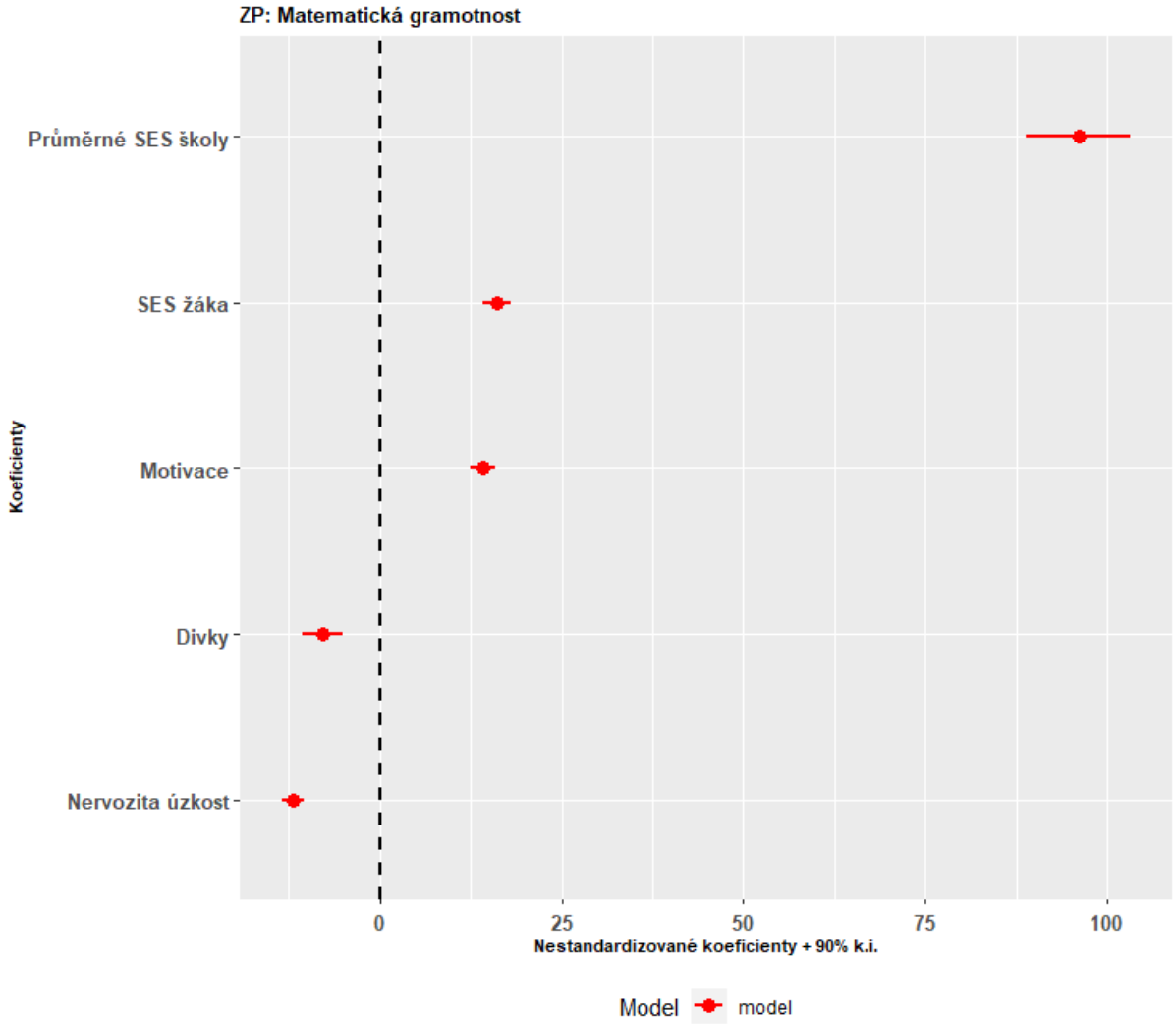
Modely můžeme rovněž exportovat do přehledné tabulky. Zde je nutné upravit názvy statistik, které hodnotí model. Jedná se o AIC, BIC a variance na dané úrovni. Jinak je skript stejný jako v případě klasické lineární regrese.

```
# Export modelů
library(texreg)
htmlreg(list(NULL_MODEL,SES, FULL), # názvy modelů
file = "./PISA2015.html", # output file
custom.model.names = c("Null","SES", "FULL"), # název modelů
custom.coef.names = c("Konstanta","SES žáka",
                      "Dívky","SES školy","Motivace", "Úzkost"), # názvy proměnných
reorder.coef = c( 2,4, 3, 5, 6, 1),
custom.gof.names = c("AIC","BIC","ll", "Počet pozorování", "Počet skupin druhé
úrovně","Intercept variance","Residual variance"), # názvy proměnných
      digits = 3, # desetinná místa
      single.row = FALSE) # zobrazení koeficientů pod sebe
```

Popřípadě vizualizovat pomocí balíčku coefplot. Zde je možné zvážit standardizaci proměnných před samotnou vizualizací.

```
#Pomocí balíčku coefplot
library(coefplot)
coefplot(FULL, title = "ZP: Matematická gramotnost",
      legend.reverse = TRUE, intercept=FALSE,
      xlab = "Nestandardizované koeficienty + 90% k.i.", ylab="Koeficienty",
      pointSize = 3, color = "black", zeroColor = "black",  innerCI = 1.645,   outerCI = 1.645,
      sort = c("magnitude"),
      coefficients=c("ESCS",
                  "Gender",
                  "MOTIVAT",
                  "ANXTEST"),
      newNames=c(ESCS="SES žáka",
                 Gender="Dívka",
                 MOTIVAT="Motivace",
                 ANXTEST="Nervozita úzkost")) +
  theme(legend.position="bottom") +
  theme(plot.title = element_text(size=10,face="bold")) +
  theme(axis.text=element_text(size=10, face="bold"),
axis.title=element_text(size=8,face="bold")) +
  scale_color_manual(values=c("red","blue"))
```

Výstup z programu R



Pokud chceme modelovat náhodnou směrnici, například to, že efekt ESCS se bude lišit napříč školami, musíme dát proměnnou ESCS do závorky, která specifikuje druhou úroveň.

```
MODEL <- lmer(PV1MATH ~ ESCS + meanESCS + Divky + MOTIVAT + ANXTEST + (1 + ESCS | CNTSCHID), data=PISA2015, REML = TRUE)
summary(MODEL)
```

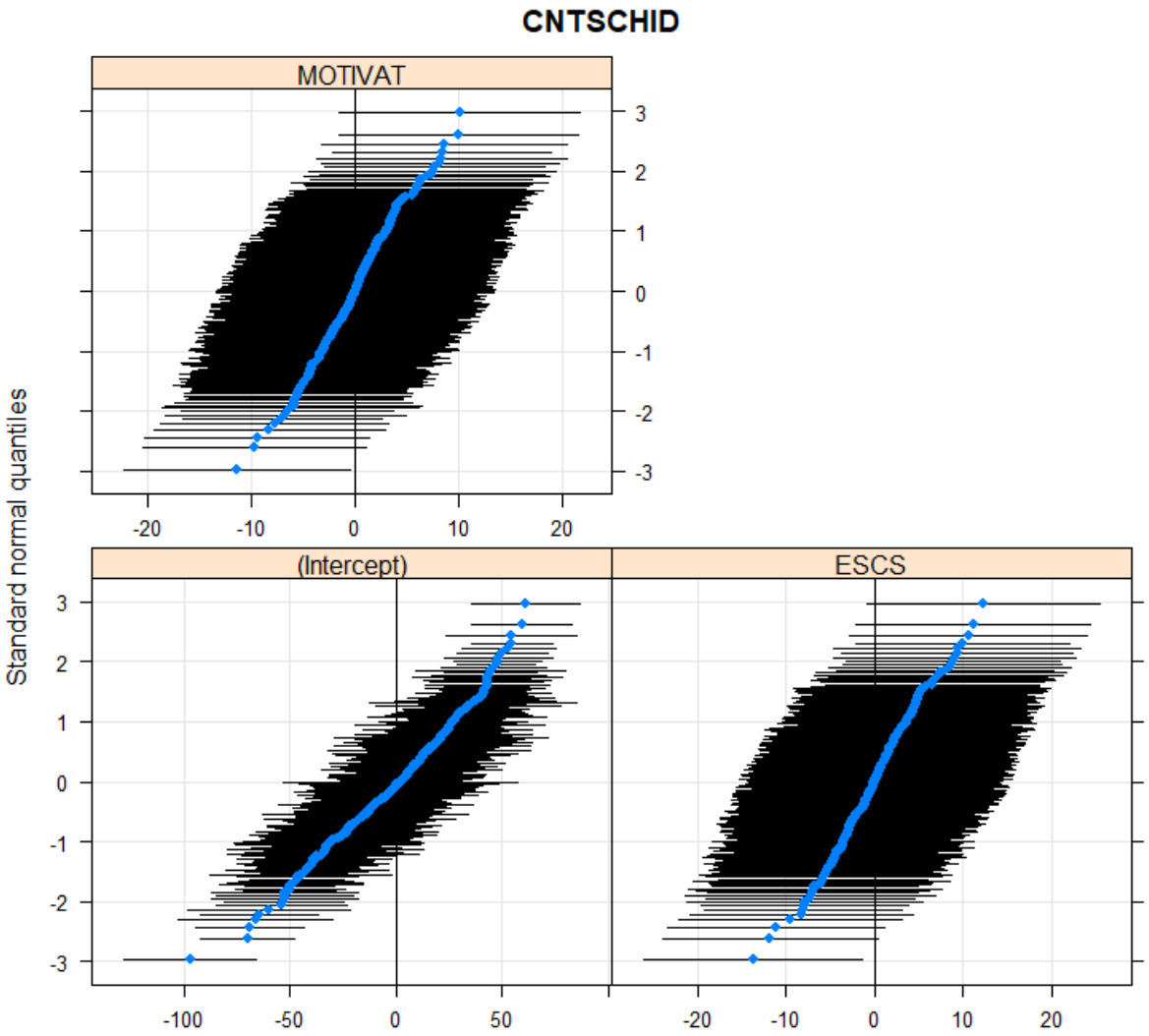
```
ranef(MODEL)
```

Jestli se efekt liší napříč školami, můžeme zjistit pomocí funkce ranef z balíčku lme4.

Vizualizovat hodnoty koeficientů napříč školami můžeme pomocí balíčku lattice a tohoto příkazu.

Výstup z programu R

```
library(lattice)
ranef(MODEL)
str(rr1 <- ranef(MODEL))
dotplot(rr1)  ## default
qqmath(rr1)
```



Popřípadě pomocí složitějšího skriptu, který vytvoří graf pomocí grafiky ggplot2. K fungování skriptu je třeba mít puštěný balíček tidyverse a broom.mixed. Broom.mixed balíček slouží pro ukládání různých výstupů z hierarchických regresních modelů do mezipaměti pro další použití u jiných funkcí.

Smyslem hierarchických modelů je testovat, zdali se efekty proměnných liší napříč druhou skupinou. Pokud ano, vyjde

```
library(tidyverse)
library(broom.mixed)

ranef(MODEL) %>%
  augment(ci.level = 0.95) %>%
  filter(variable=="ESCS") %>%
  dplyr::select(c(level, estimate, std.error)) %>%
  rename(stateString_1 = level) %>%
  mutate(estimate = estimate + fixef(MODEL)["ESCS"]) %>%
    ggplot(aes(x = reorder(stateString_1, -estimate),
           y = estimate)) +
  geom_point(size = 3, color = "blue" ) +
  labs(x = "Škola",
       y = "Efekt ESCS žáka") +
  geom_errorbar(aes(ymin = estimate - 1.96*std.error,
                    ymax = estimate + 1.96*std.error),
                size = 1.25,
                width = 0, color = "blue") +
  geom_hline(yintercept = fixef(MODEL)["ESCS"],
             size = 1.25,
             linetype = "dashed",
             color = "red") +
  geom_hline(yintercept = fixef(MODEL)["ESCS"] +
               1.96*se.fixef(MODEL)["ESCS"],
             size = 1.25,
             linetype = "dotted") +
  geom_hline(yintercept = fixef(MODEL)["ESCS"] -
               1.96*se.fixef(MODEL)["ESCS"],
             size = 1.25,
             linetype = "dotted") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.95, hjust = 1)) +
  theme(axis.text.x = element_text(size = 10.5))
```

nám variance koeficientu statisticky významná. V tomto případě bychom měli vizualizovat koeficienty pomocí funkce ranef a přidružených vizualizačních funkcí. Hierarchické modely můžeme použít nejen pro data z velkých mezinárodních šetření, ale pro libovolná data, kde je nějaká hierarchie a kde se nám případy klastrují dle dalších úrovní. Lze například modelovat vyšší územní samosprávné celky, jako jsou okresy a kraje. Můžeme předpokládat, že efekt proměnné se bude lišit napříč okresy.

V ideálním případě pak můžeme provést interakci mezi proměnnou, kterou modelujeme náhodně, a proměnnou na druhé úrovni, která nám rozdílné koeficienty dokáže vysvětlit. To je ale v praxi velmi obtížné.

V neposlední řadě je vhodné říci, že lze všechny další typy regresních modelů, jako je logistická regrese, multinomická regrese, negativní binomická regrese a Poissonova regrese, modelovat jako hierarchické modely.

7.3 Další typy regresních modelů

V této sekci ukážeme skripty pro další typy regresních modelů. Druh regresního modelu zvažujeme vždy na základě typu dat a zejména na úrovni měření závislé proměnné. Pro binární závisle proměnnou používáme logistickou regresi.

Logistická regrese

```
model <- glm(dichotomická závislá proměnná ~  ESCS + meanESCS, data =PISA2015,
family = binomial)
summary(model)
```

V případě, že naše závislá proměnná je na ordinální škále, můžeme použít ordinální regresi. Například spokojenost s profesí učitele na škále 1 až 4, kdy 1 znamená nejmenší spokojenost a 4 znamená největší spokojenost s profesí učitele (souhlasí s výrokem, že je v práci spokojený, proměnná pod kódem TT3G53). V R ordinální regrese nemá defaultní příkaz, musíme proto použít balíček MASS. Skript je vytvořen na základě datasetu TALIS 2018 (TT3G02="Věk učitele"), pohlaví žena byla rekódována z původní proměnné TT3G01 TALIS 2018.

Ordinální regrese

```
library(MASS)
model <- polr(TT3G53 ~ pohlaví+ TT3G02, data = TALIS2018, Hess=TRUE)

summary(model )
```

Interpretace koeficientů je složitější, protože logaritmus šancí interpretuje vždy k páru kategorií závislé proměnné.[11]

V případě, že naše závislá proměnná má několik nezávislých kategorií, použijeme multinomickou regresi. Někdy můžeme zvažovat i to, že ordinální škálu budeme modelovat jako nominální kategorie, pokud k tomu máme oporu v teorii. Pro zjednodušení použijeme stejné proměnné z šetření TALIS 2018. Pro multinomickou regresi musíme použít balíček nnet, protože základní jazyk R nemá v sobě naprogramovanou danou funkci.

Multinomická regrese

```
library(nnet)
model <- multinom(TT3G53 ~ pohlaví+ TT3G02, data = TALIS2018)
summary(model )
```

Multinomická regrese přináší komplexnější výstup než obyčejná logistická regrese. Musí brát vždy v potaz referenční kategorii a vůči ní vždy porovnávat koeficienty (buď logaritmus šancí, nebo po převedení na poměr šancí). Rovněž můžeme dále specifikovat, která kategorie kategorické nezávislé proměnné má být referenční. Například místo muže je možné nastavit referenční kategorii ženu atd.

Poté opět spustíme skript na výpočet multinomické regrese.

```
TALIS2018$pohlaví <- relevel(TALIS2018$pohlaví, ref = "Žena")
```

Poissonova regrese

Tuto regresi použijeme v případě, že data nejsou normálně rozdělena, ale vykazují velkou míru disperze. Nemůžeme použít obyčejnou lineární regresi, ale příslušný typ regresního modelu. V případě disperzních dat, která nejsou normálně rozdělena, se používá Poissonova regrese. Využít ji lze v případě, kdy třeba závislá proměnná ukazuje, kolik procent maturantů neuspělo u maturitní zkoušky v dané škole. Tato proměnná je rozložena tak, že pozorujeme velký počet škol s nízkou neúspěšností. Ale máme i nezanedbatelný počet škol, kde je neúspěšnost vysoká. Data tak nejsou normálně rozdělena a je nutné je modelovat pomocí Poissonovy regrese. Pro tento typ regrese není třeba žádný balíček, stačí příkaz glm v RStudiu a specifikace rodiny modelu.

Koeficienty jsou pak logaritmované očekávané četnosti (expected log count)[12]. Přestože je neúspěšnost u maturitní zkoušky jako proporce, lze tuto proměnnou vzhledem k poissonovu rozdělení modelovat tímto typem modelu. Nevýhodou je obtížná interpretace koeficientů.

```
model <-  glm(NEUS_CIST_CELK ~ GYMPL + lnPOC_OBYV + Vzd_vysok_ + Eko_nezam_ +
EXEKUCE_PODIL, family="poisson", data=MATURITY)

summary(model)
```

[11] Viz blíže stránky Kalifornské univerzity v Los Angeles, https://stats.idre.ucla.edu/r/faq/ologit-coefficients/

[12] Viz blíže stránky Kalifornské univerzity v Los Angeles, https://stats.idre.ucla.edu/stata/dae/poisson-regression/

7.4 Negativní binomická regrese

V případě, že závislá proměnná je početní (count variable), například počet incidentů v souvislosti s šikanou ve školách, je možné tuto závisle proměnnou modelovat pomocí negativní binomické regrese. Principiálně se neliší od Poissonovy regrese, rozdíl je v tom, že se používá pro velmi disperzní data. V tomto případě je ale nutné použít balíček MASS. Protože pro tento typ modelu nejsou k dispozici vhodná data, jedná se o obecný příkaz.

```
library(MASS)
model <- glm-nb(incidence ~ meanESCS + neúspěšnostžáků, data = vlastnídata)
summary(model)
```

8

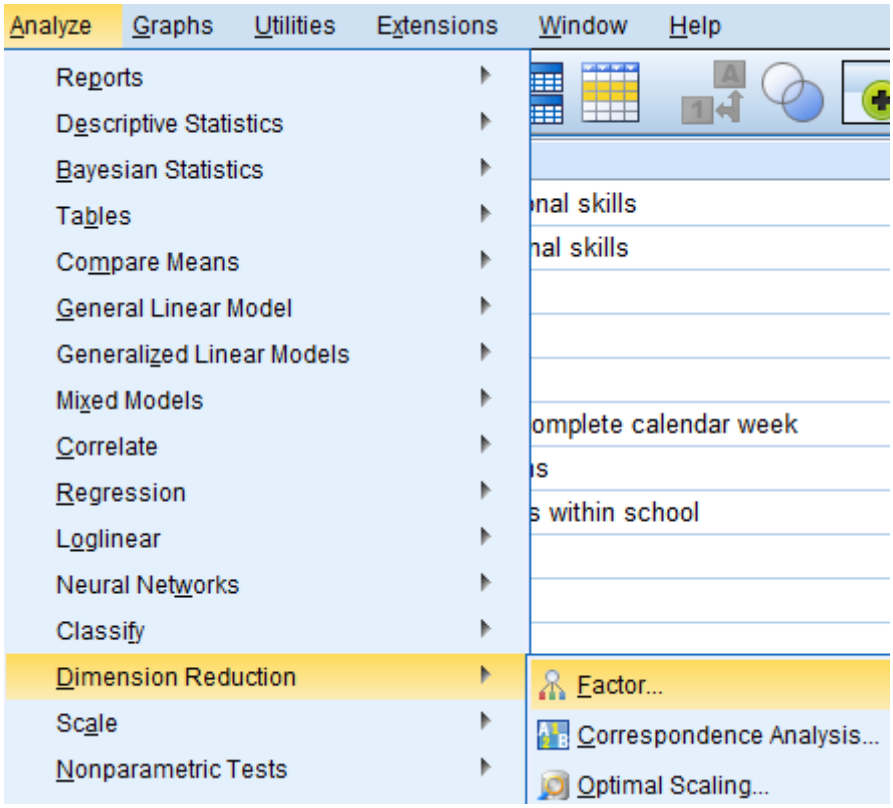
Analýza dimenzí a seskupovací techniky

8 ANALÝZA DIMENZÍ A SESKUPOVACÍ TECHNIKY

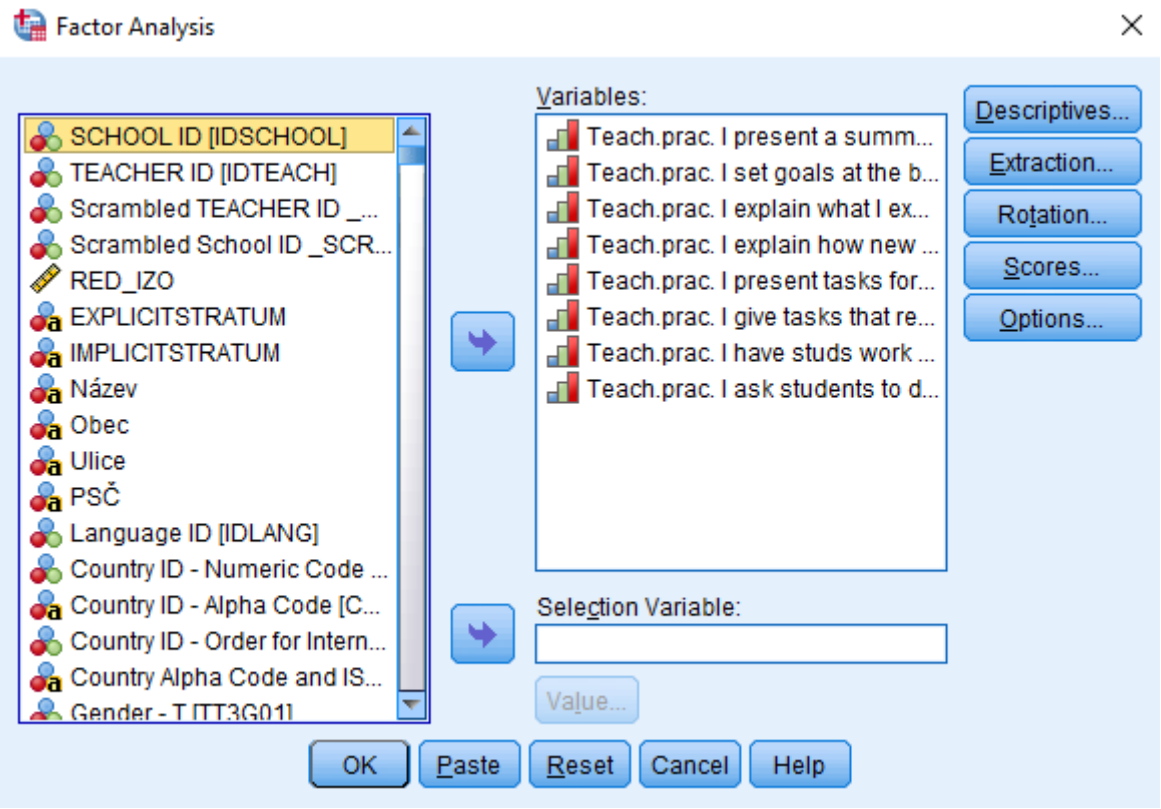
Tato sekce se zabývá technikami analýzy dat, které mají za úkol explorační analýzu a konkrétně to, zdali lze některé proměnné použít pro klasifikaci dimenzí (a tím pádem zredukovat počet proměnných do jednoho zastřešujícího konceptu) nebo pro klasifikaci případů na základě jejich znaků.

8.1 Explorační faktorová analýza

Explorační faktorová analýza je technikou, která slouží k redukci dat. Jinak řečeno je jejím smyslem nahradit sadu vzájemně spjatých proměnných malým počtem ne přímo pozorovaných znaků – latentních proměnných či jiných faktorů. V SPSS tak explorační faktorovou analýzu nalézáme v záložce *Analyze → Dimension Reduction → Factor*.



Explorační faktorová analýza je využitelná tehdy, kdy máme řadu proměnných, u nichž se domníváme, že mohou skrývat latentní (ne přímo pozorované) proměnné. Příkladem mohou být různé výukové praktiky, které učitelé využívají ve svých hodinách.



Prvním krokem zadání faktorové analýzy v SPSS je přesunutí všech proměnných, s nimiž chceme pracovat, do okna *Variables*. Množství výstupů následně můžeme rozšířit za využití polí vpravo. První nabídka *Descriptives* umožňuje zobrazit celou řadu užitečných údajů včetně popisné statistiky proměnných a korelací mezi nimi. Vzhledem k tomu, že faktorová analýza staví na korelacích mezi proměnnými, může se jejich prozkoumání jevit jako důležitý první krok k pochopení struktur v datech. Po rozkliknutí políčka *Extraction* dále můžeme upravovat možnosti nalezení faktorů, konkrétně zde můžeme změnit metodu či způsob stanovení počtu extrahovaných faktorů – v tomto případě necháváme základní nastavení, tedy používáme metodu hlavních komponent (*Principal components analysis*, PCA), vycházíme z matice korelací a počet faktorů extrahujeme podle Kaiserova pravidla (*eigenvalue* vyšší než 1)[13]. *Options* nabízí volbu způsobu práce s chybějícími hodnotami a různé způsoby zobrazení koeficientů (vhodné například může být potlačení zobrazení koeficientů s nízkou hodnotou za účelem zpřehlednění výsledné tabulky). *Rotation* umožňuje provést rotaci faktorů a pod možností *Scores* se skrývá možnost uložení nových proměnných ve formě faktorových skórů, které zachycují faktory (latentní koncepty) nalezené v datech a které jsou využitelné v dalších analýzách.

[13] Co se týče využití Kaiserova pravidla, v literatuře se objevují zmínky, že se nejedná o postup ideální (Soukup 2021). Alternativou může být například využití paralelní analýzy, avšak tu lze v SPSS provést pouze za využití skriptu.

Communalities

	Initial	Extraction
Teach.prac. I present a summary of recently learned content	1,000	,431
Teach.prac. I set goals at the beginning of instruction	1,000	,601
Teach.prac. I explain what I expect the students to learn	1,000	,627
Teach.prac. I explain how new and old topics are related	1,000	,502
Teach.prac. I present tasks for which there is no obvious solution	1,000	,473
Teach.prac. I give tasks that require students to think critically	1,000	,480
Teach.prac. I have studs work in small groups to come up with a joint solution	1,000	,349
Teach.prac. I ask students to decide on own procedures for solving complex tasks	1,000	,488

Extraction Method: Principal Component Analysis.

Total Variance Explained

	Initial Eigenvalues			Extraction Sums of Squared Loadings		
Component	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	2,461	30,768	30,768	2,461	30,768	30,768
2	1,490	18,626	49,394	1,490	18,626	49,394
3	,880	10,996	60,390			
4	,763	9,541	69,930			
5	,680	8,496	78,426			
6	,671	8,382	86,808			
7	,636	7,944	94,753			
8	,420	5,247	100,000			

Extraction Method: Principal Component Analysis.

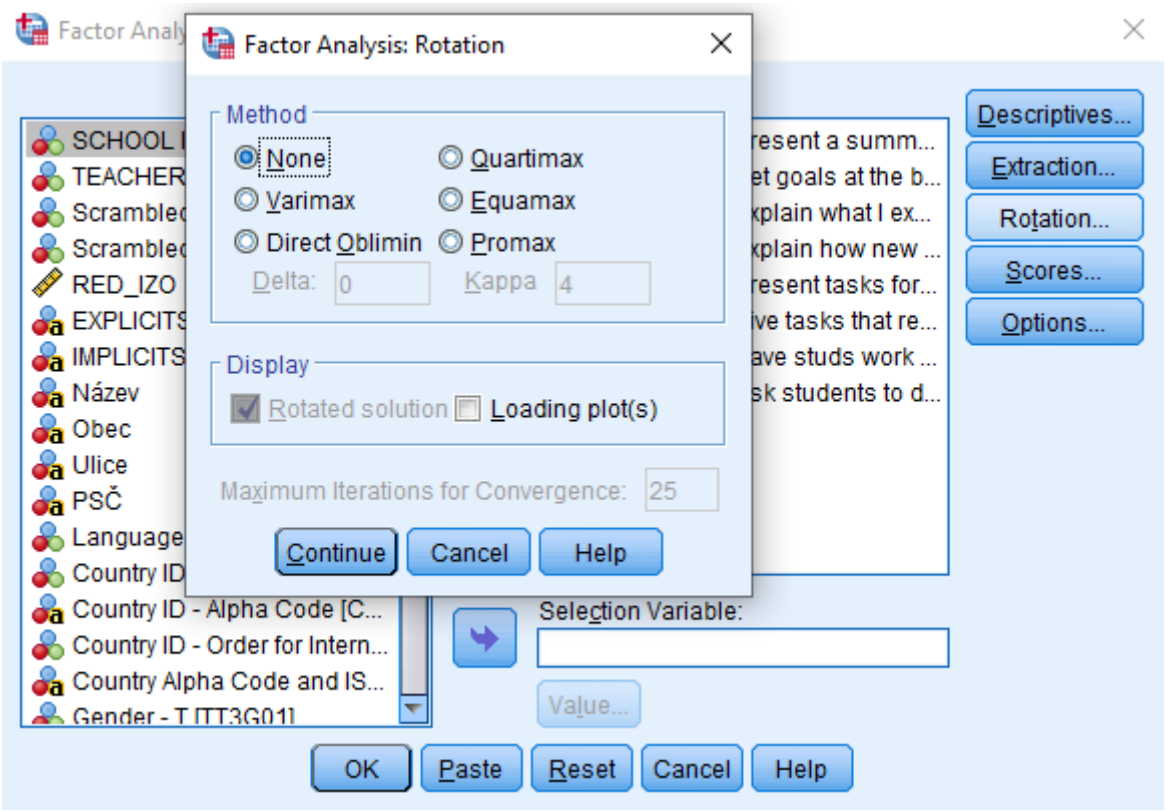
Component Matrix[a]

	Component	
	1	2
Teach.prac. I present a summary of recently learned content	,553	-,354
Teach.prac. I set goals at the beginning of instruction	,661	-,406
Teach.prac. I explain what I expect the students to learn	,708	-,354
Teach.prac. I explain how new and old topics are related	,676	-,212
Teach.prac. I present tasks for which there is no obvious solution	,357	,587
Teach.prac. I give tasks that require students to think critically	,512	,466
Teach.prac. I have studs work in small groups to come up with a joint solution	,342	,482
Teach.prac. I ask students to decide on own procedures for solving complex tasks	,504	,485

Extraction Method: Principal Component Analysis.

a. 2 components extracted.

Pokud však v prvním kroku rozšiřující nabídku nevyužijeme, získáme tři tabulky. První tabulka komunalit v podstatě ukazuje, nakolik daná proměnná koreluje se všemi extrahovanými faktory. Nízké hodnoty svědčí o tom, že daná proměnná není pro faktorové řešení příliš vhodná a lze zvážit její vyřazení. Druhá tabulka zobrazuje podíl vyčerpaného rozptylu jednotlivými faktory. Platí, že čím více rozptylu faktory vyčerpávají, tím lépe, neboť neztrácíme příliš z původní informace. Dohromady by se přitom mělo jednat alespoň o 50 %. Třetí tabulka následně uvádí samotné faktorové zátěže, tedy korelace mezi danou proměnnou a příslušným faktorem. Čím vyšší je zde hodnota, tím více je faktor danou proměnnou sycen. Minimální hodnota faktorové zátěže potom závisí i na velikosti datového souboru. V případě větších datových souborů se hovoří o hodnotě 0,3, u menších datových souborů by ideálně měla být vyšší. V některých případech (včetně zde uvedeného) se stává, že je faktorová zátěž u některých proměnných silná ve více faktorech. Řešením takové situace je provedení rotace faktorů pomocí již zmíněné možnosti *Rotation*.



Cílem rotace konkrétně je přimknout proměnné k jednomu z faktorů. Ve vloženém okně nabídky *Rotation* vidíme hned několik metod rotací. Výběr přitom závisí i na tom, zda předpokládáme, že spolu faktory souvisejí, či nikoliv. Jelikož v tomto případě souvislost předpokládáme, volíme metodu oblimin.

Pattern Matrix[a]

	Component 1	Component 2
Teach.prac. I present a summary of recently learned content	,666	-,056
Teach.prac. I set goals at the beginning of instruction	,784	-,052
Teach.prac. I explain what I expect the students to learn	,788	,017
Teach.prac. I explain how new and old topics are related	,671	,131
Teach.prac. I present tasks for which there is no obvious solution	-,098	,700
Teach.prac. I give tasks that require students to think critically	,103	,665
Teach.prac. I have studs work in small groups to come up with a joint solution	-,042	,598
Teach.prac. I ask students to decide on own procedures for solving complex tasks	,084	,677

Extraction Method: Principal Component Analysis.
Rotation Method: Oblimin with Kaiser Normalization.

a. Rotation converged in 4 iterations.

Výstupem je tabulka zobrazující faktorové zátěže po provedené rotaci. Z pohledu na tabulku je přitom zřejmé, že se rotací skutečně podařilo výsledek vyjasnit. Na rozdíl od původního nerotovaného řešení v tomto případě již žádná z proměnných není silná v obou faktorech.

V případě, že chceme provést analýzu v programu RStudio, použijeme příkaz factanal, kde specifikujeme dataset a počet faktorů, které se mají extrahovat.

```
FA_VYSLEDEK  <- factanal(dataset, factors = 2)
FA_VYSLEDEK
```

```
FA_VYSLEDEK_VARIMAX  <- factanal(dataset, factors = 2, rotation = "varimax")
FA_VYSLEDEK_PROMAX  <- factanal(dataset, factors = 2, rotation = "promax")
```

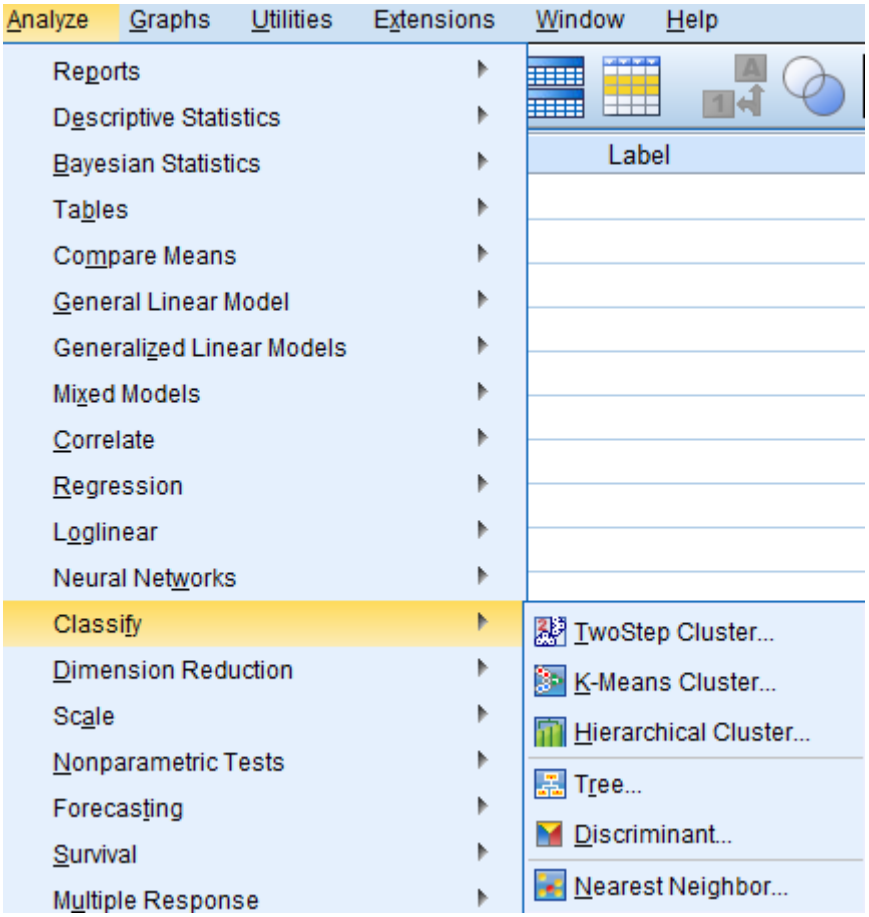
Rovněž můžeme nastavit rotaci faktorů.

V případě RStudia je možné dále jednotlivé faktory tvořené danými proměnnými vizualizovat. Vizualizovat lze vždy dva faktory (přestože třeba analýza identifikovala více faktorů). Jaké dva faktory vizualizujeme, pak záleží především na teoretických kritériích. Výsledný skript nám vytvoří graf, kde na ose X bude faktor 1, na ose Y faktor 2. Jednotlivé proměnné budou umístěny dle jejich zátěží.

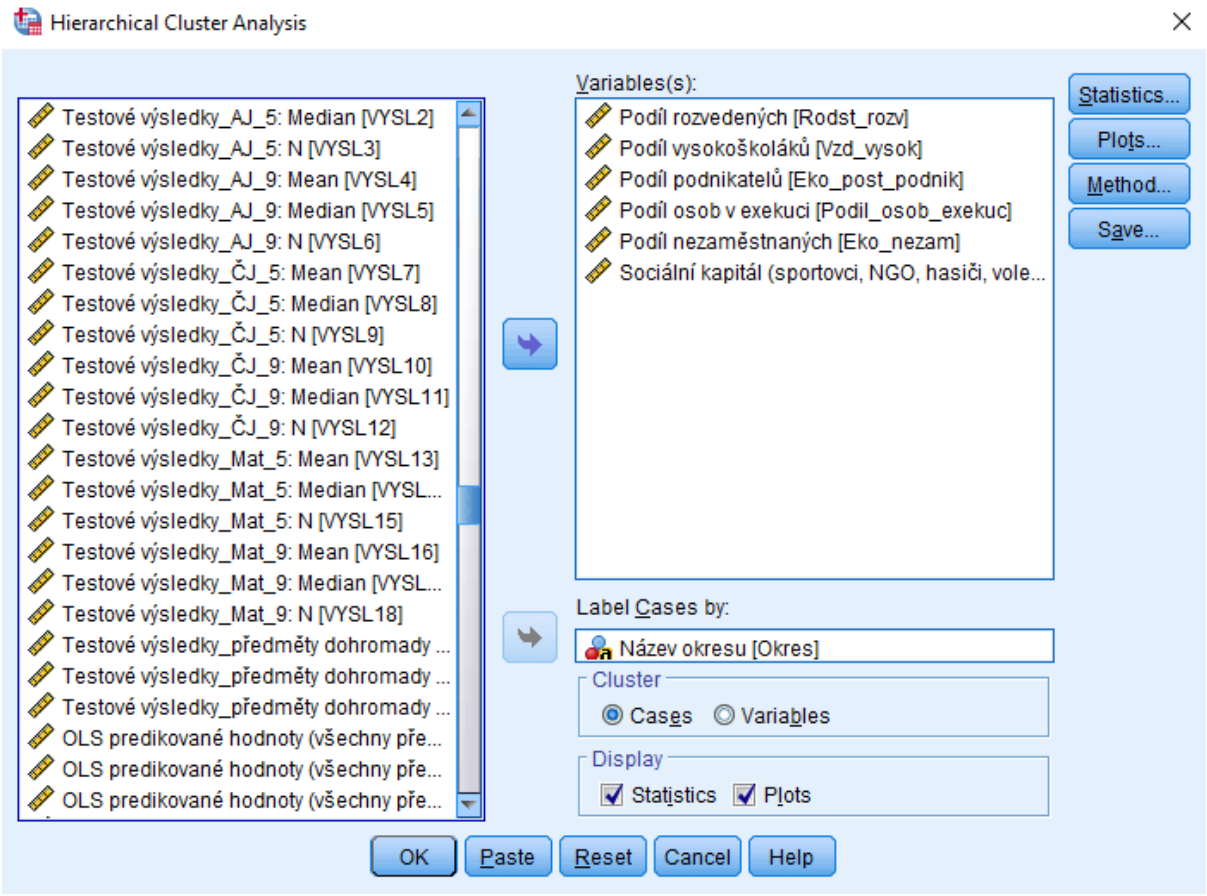
```
par(mfrow = c(1,3))
plot(FA_VYSLEDEK$loadings[,1],
    FA_VYSLEDEK$loadings[,2],
    xlab = "Factor 1",
    ylab = "Factor 2",
    ylim = c(-1,1),
    xlim = c(-1,1),
    main = "No rotation")
text(FA_VYSLEDEK$loadings[,1]-0.08,
    FA_VYSLEDEK$loadings[,2]+0.08,
    colnames(dataset)
abline(h = 0, v = 0)
```

8.2 Shluková analýza

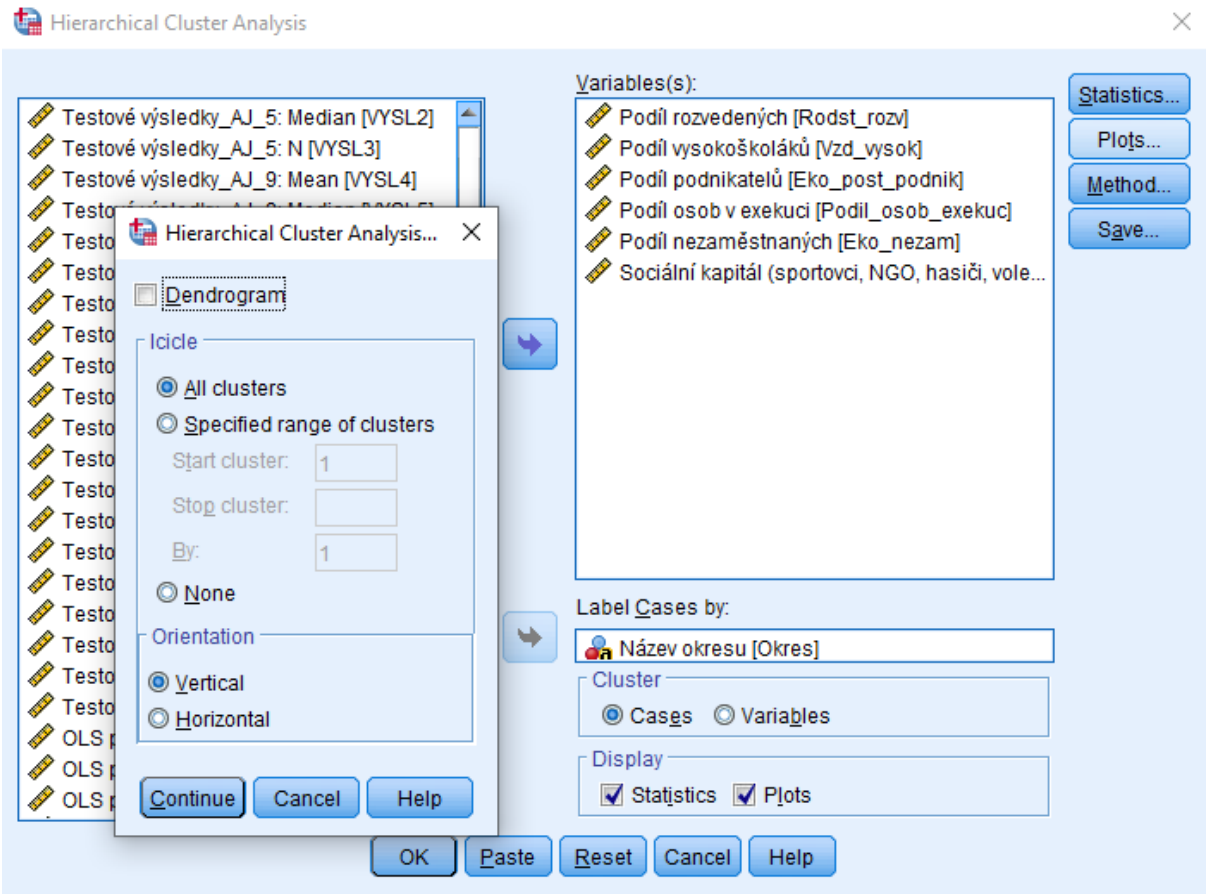
Shluková analýza se podobně jako explorační faktorová analýza řadí mezi vícerozměrné explorační techniky. Oproti faktorové analýze se však častěji používá k hledání podobných případů, a ne proměnných (ačkoliv ji lze použít oběma způsoby). V SPSS můžeme shlukovou analýzu provést po rozbalení záložky *Analyze → Classify*.



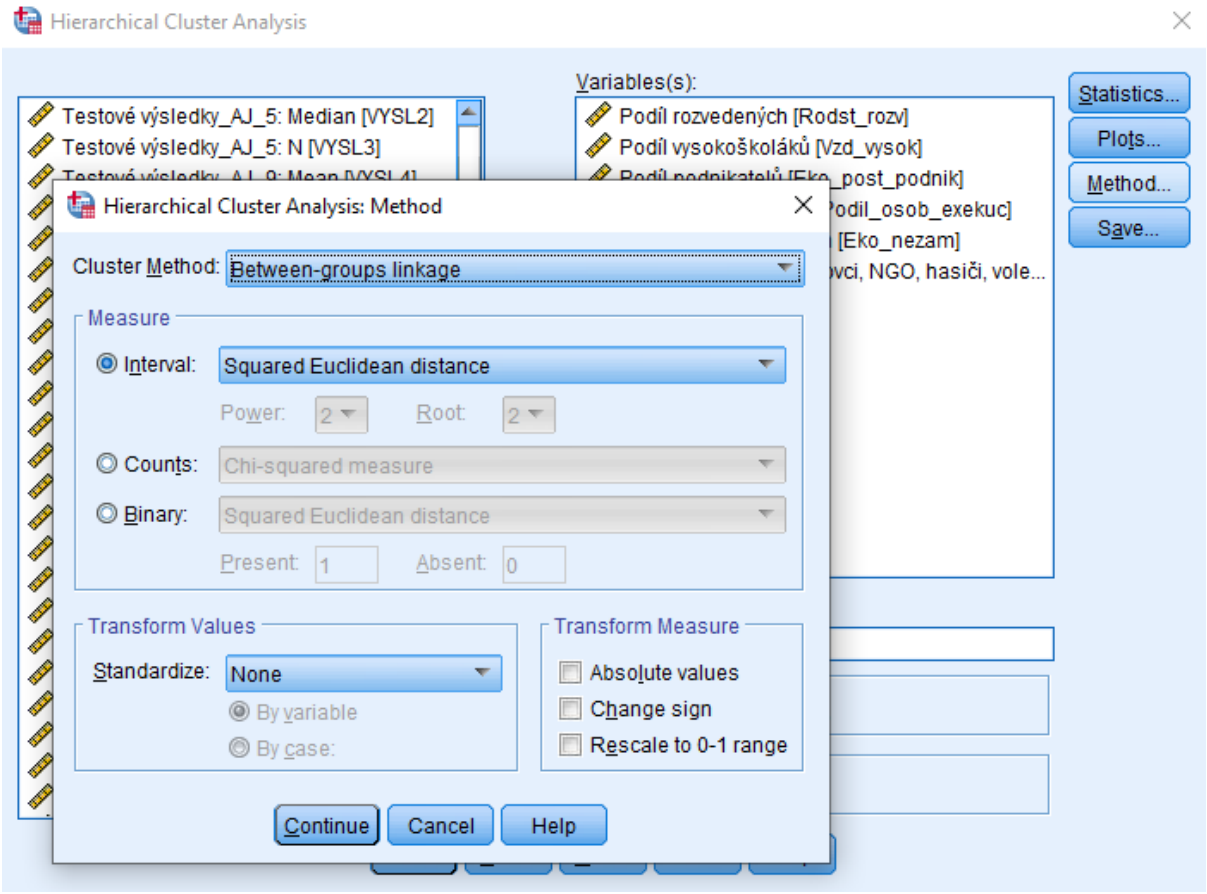
Existuje více přístupů ke shlukování. Nabídka SPSS obsahuje hierarchickou shlukovací analýzu (*Hierarchical Cluster*), K-means shlukovací analýzu (*K-Means Cluster*) a dvoustupňovou shlukovací analýzu (*TwoStep Cluster*). Zjednodušeně řečeno hierarchické shlukování provádíme zejména tehdy, kdy máme nižší počet případů, K-means shlukování vyžaduje určení počtu shluků předem, kdežto dvoustupňové shlukování je poměrně flexibilní a umožňuje práci s větším počtem případů a kombinaci různých typů proměnných.



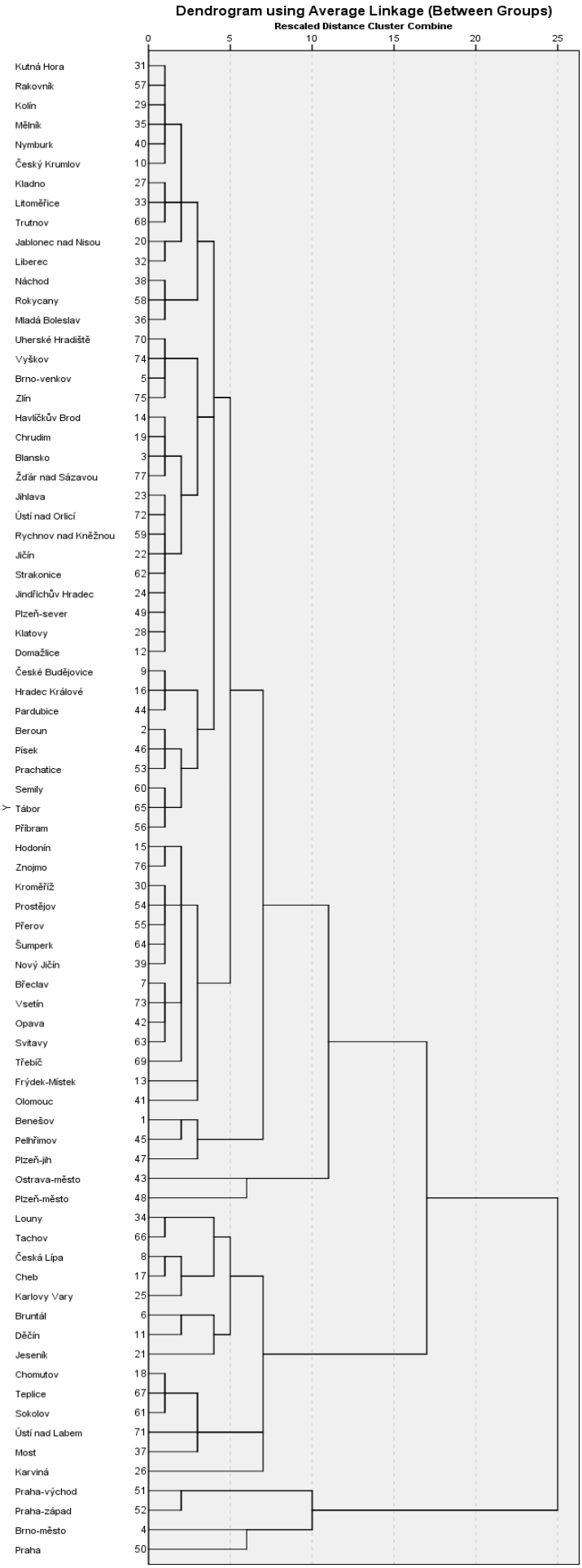
Shlukovací analýzu můžeme provést například tehdy, kdy chceme hledat podobné okresy z hlediska socioekonomických charakteristik. K tomuto úkolu konkrétně využijeme hierarchické shlukování. Po otevření zadání hierarchické shlukovací analýzy v SPSS představuje další krok přetažení všech proměnných, na jejichž základě chceme okresy sdružovat, do okna *Variables*. Pokud ve výstupu chceme mít případy označeny, vyplníme rovněž okno *Label Cases by* – v našem případě do něj přetáhneme proměnnou okres. Jak bylo zmíněno výše, ačkoliv se shlukování využívá především k hledání podobných případů, zvolením možnosti *Variables* místo *Cases* v políčku *Cluster* pod seznamem proměnných můžeme místo podobných případů hledat podobné proměnné (v tomto příkladu s okresy nicméně zůstáváme u defaultní možnosti *Cases*).



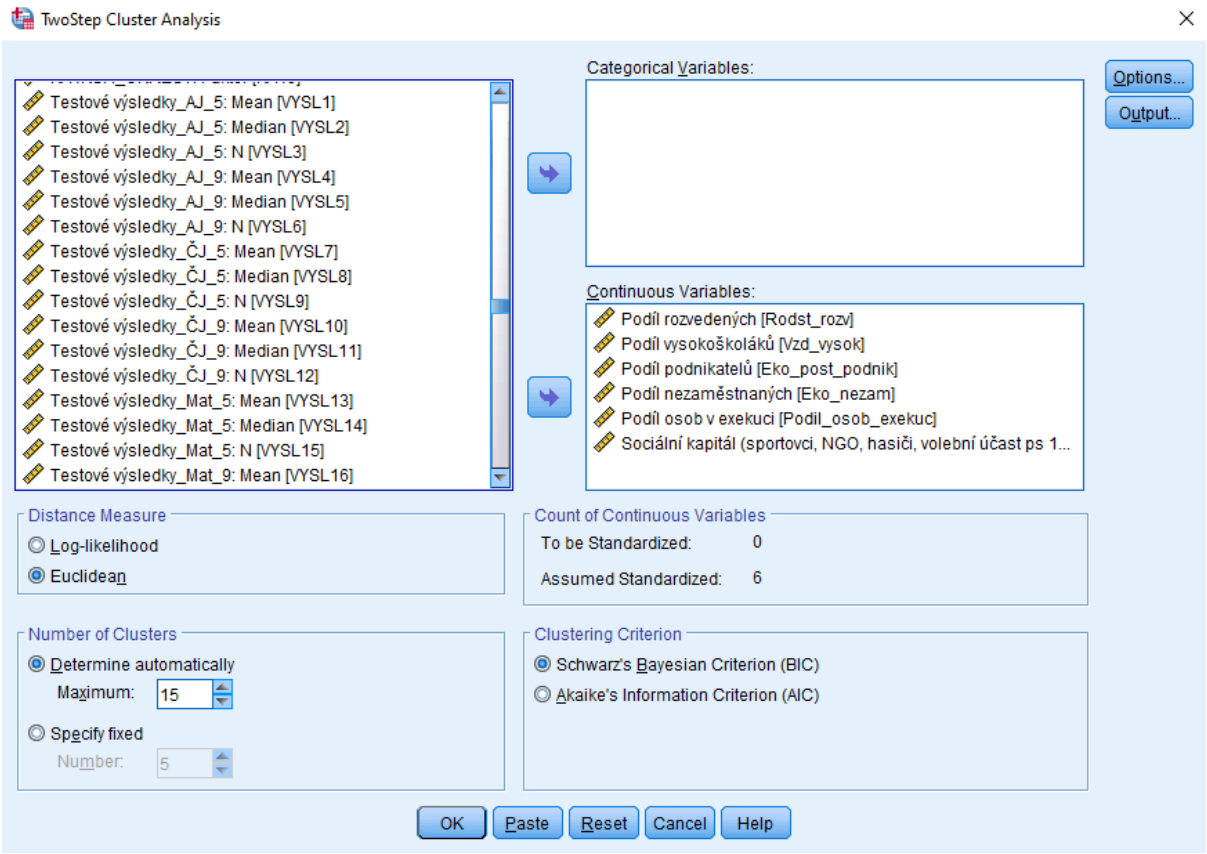
Jakmile jsou proměnné přetaženy, lze výpočet dále upravovat skrze kolonky vpravo. Nejdříve je vhodné zaškrtnout možnost dendrogramu v okně *Plots*.



V okně *Method* se dále nacházejí důležitá nastavení provedení shlukové analýzy, jako metoda shlukování a míra vzdálenosti. Je zde rovněž možnost proměnné standardizovat, což je před provedením shlukové analýzy většinou žádoucí, avšak v našem případě proměnné již standardizovány jsou. Nakonec skrze zde nezobrazenou nabídku *Statistics* určujeme, kolik shluků chceme získat (pakliže to víme), a přes další nabídku *Save* můžeme výsledek shlukové analýzy ve formě příslušnosti případů do skupin uložit jako novou proměnnou.


Dendrogram using Average Linkage (Between Groups)
Rescaled Distance Cluster Combine
0
5
10
15
20
25
Kutná Hora 31
Rakovník 57
Kolín 29
Mělník 35
Nymburk 40
Český Krumlov 10
Kladno 27
Litoměřice 33
Trutnov 68
Jablonec nad Nisou 20
Liberec 32
Náchod 38
Rokycany 58
Mladá Boleslav 36
Uherské Hradiště 70
Vyškov 74
Brno-venkov 5
Zlín 75
Havlíčkův Brod 14
Chrudim 19
Blansko 3
Žďár nad Sázavou 77
Jihlava 23
Ústí nad Orlicí 72
Rychnov nad Kněžnou 59
Jičín 22
Strakonice 62
Jindřichův Hradec 24
Plzeň-sever 49
Klatovy 28
Domažlice 12
České Budějovice 9
Hradec Králové 16
Pardubice 44
Beroun 2
Písek 46
Prachatice 53
Semily 60
Tábor 65
Příbram 56
Hodonín 15
Znojmo 76
Kroměříž 30
Prostějov 54
Přerov 55
Šumperk 64
Nový Jičín 39
Břeclav 7
Vsetín 73
Opava 42
Svitavy 63
Třebíč 69
Frýdek-Místek 13
Olomouc 41
Benešov 1
Pelhřimov 45
Plzeň-jih 47
Ostrava-město 43
Plzeň-město 48
Louny 34
Tachov 66
Česká Lípa 8
Cheb 17
Karlovy Vary 25
Bruntál 6
Děčín 11
Jeseník 21
Chomutov 18
Teplice 67
Sokolov 61
Ústí nad Labem 71
Most 37
Karviná 26
Praha-východ 51
Praha-západ 52
Brno-město 4
Praha 50

Výstupem zadaného provedení shlukové analýzy je několik tabulek a dva grafy, rampouchový a vyžádaný dendrogram, který je zde rovněž prezentován. Dendrogram zobrazuje proces shlukování, přičemž nám prostý pohled na něj může naznačit, kolik shluků může být ideálních. Zařadit případy do skupin je dále možné skrze nabídku *Statistics*, jak je zmíněno výše. Proměnnou příslušnosti do skupin následně ukládáme přes nabídku *Save*.



Alternativou by bylo využití dvoustupňového shlukování. Dvoustupňové shlukování umožňuje pracovat s různými typy proměnných, jak je patrné i z pohledu na jeho zadání. SPSS automaticky proměnné standardizuje, což můžeme změnit v nabídce *Options*. Rozkliknutím tlačítka *Output* lze upravovat způsob zobrazení výsledku (v tabulce či v interaktivním přehledu, který aktivujeme dvojím poklepáním na základní výstup graficky zobrazující celkovou vhodnost provedení). Kromě toho můžeme v základním okně provést pár dalších úprav včetně určení počtu požadovaných shluků.

9

Structural Equation Modelling (SEM)

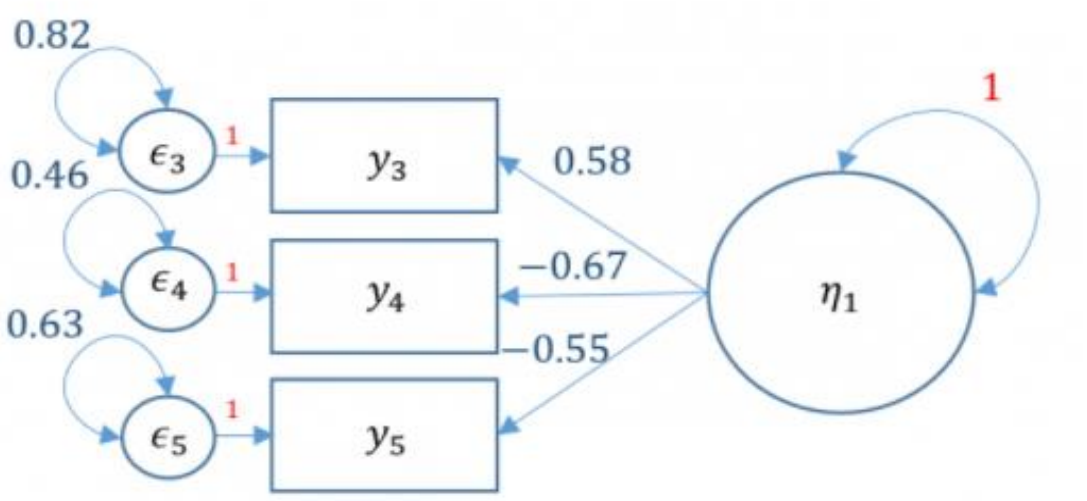
9 STRUCTURAL EQUATION MODELLING (SEM)

V případě strukturního modelování (SEM) ukážeme jen jednoduché příkazy v programu RStudio. SPSS sice SEM zvládá, ale jen v případě instalování propojení s programem R, popřípadě instalace softwaru AMOS, které běží na enginu SPSS, kdy v AMOSu analyzujeme datové soubory typu .sav. Alternativou může být freeware Jasp. Strukturní modelování je nejčastěji prováděno v programu Mplus, v prostředí R je pak pro SEM balíček lavaan.

V rámci datové základny ČŠI lze nejčastěji využít takzvaný measurement model. Tedy analýzu toho, jestli položky měřící nějaký koncept spolu skutečně souvisí. Model ohodnotí, jestli všechny položky například z baterie otázek v rámci dotazníku skutečně měří jeden či více latentních konceptů. Příkaz v balíčku lavaan pro testování jednoho latentního konceptu, který se skládá ze tří položek:

```
m1a  <- ' f  =~ proměnná_1 + proměnná_2 + proměnná_3
f ~~ 1*f '
jedenfaktor_tři_proměnné<- cfa(m1a, data=dat)
summary(jedenfaktor_tři_proměnné)
# Pro stnadardizované hodnoty
jedenfaktor_tři_proměnné<- cfa(m1a, data=dat,std.lv=TRUE)
summary(jedenfaktor_tři_proměnné)
```

Pro názornost je lepší proměnné daného konceptu nakreslit a vizualizovat. Schéma níže ukazuje "measurement model" neboli konfirmační faktorovou analýzu, kdy testujeme, že daný koncept je měřen námi zvolenými proměnnými (y1 až y3). Ideálně by standardizované korelace s latentním faktorem (kruh) u daných proměnných (obdélník) měly být nad hodnotou ± 0,3.



Obecně pak nám output dá modelovou statistiku, zde platí, že ideální model má hodnoty CFI nad 0,95 a hodnoty RMSEA pod 0,050. Statistiku vyvoláme tímto příkazem:

```
summary(jedenfaktor_tři_proměnné, fit.measures=TRUE, standardized=TRUE)
```

Strukturní modelování je složitou záležitostí, která by zabrala mnoho stran skriptů. Základem je ale tzv. "measurement model". Následovat může analýza více faktorů a jejich vzájemných kovariancí, pokud přidáme regresní přímku, budeme mít plnohodnotný SEM model, který bude předpokládat, že mezi latentními faktory existuje vztah, kdy jeden vysvětluje varianci druhého. Jedná se v podstatě o kombinaci konfirmační faktorové analýzy, regrese a "path analysis" (úsekové analýzy). Blíže k balíčku lavaan jeho internetové stránky provozované Univerzitou v Gentu (https://lavaan.ugent.be/).

10

Matching Methods

10 MATCHING METHODS

Problémem observačních studií je to, že naše nezávislá proměnná není distribuována náhodně, jako je tomu v případě experimentu. Proto o nalezených vztazích nemůžeme říct, že jsou kauzální. Jedinou možností, jakým způsobem z observačních studií vyvozovat tentativní kauzální závěry, je buď snaha o kontrolu všech relevantních proměnných, které by efekt nezávislé proměnné na závislou proměnnou mohl zkreslit, nebo vytvoření datasetu, který bude tvořen podobnými případy, jež se budou lišit pouze v nezávislé proměnné X. Algoritmus pro vytvoření tohoto datasetu se nazývá "matching algoritmus". Příkladem může být analýza používání alternativních učebnic matematiky. Problémem observačních dat je to, že my nevíme, jestli tyto učebnice používají školy, které se nějakým systémovým způsobem odlišují od jiných škol, které alternativní učebnice matematiky nepoužívají. Matching metody v programu R implementoval tým okolo statistika Harvardovy univerzity Garyho Kinga. Jedná se o balíček "matchit".

V prvním kroku se vytvoří dataset na základě algoritmu, kdy se snažíme vytvořit dvě skupiny případů. První skupina, kde jsou školy využívající alternativní učebnice, a druhá skupina škol, které jsou co nejvíce podobné první skupině škol, ale alternativní učebnice nevyužívají. Ideální je do algoritmu vložit co nejvíce proměnných. V našem modelovém příkladě jen pro názornost přidáváme počet obyvatel obce, ve které škola leží, a průměrný SES školy. V reálné analýze je vhodné přidat i další relevantní proměnné na úrovni školy. Algoritmus v první řadě tedy vybere ke skupině škol používajících alternativní učebnice co nejpodobnější školy. Metoda tak simuluje experiment, nicméně se pořád jedná o observační data.

Ve skriptu výše je možné upravit metodu algoritmu, která vybírá kontrolní skupinu škol. Existuje několik typů, z nichž nejpoužívanější je "propensity score matching". Ten je založený na predikovaných hodnotách logistického regresního

```
library(MatchIt)
m.out_NEAREST <- matchit(ALTERNATIVNI_UCEBNICE ~ POCET_OBYVATEL_OBCE +
meanESCS, data = PISA2015, method = "nearest") #zatím dává asi nejlepší výsledky
summary(m.out_NEAREST) #sumarizační statistika porovnávající obě skupiny škol
plot(m.out_NEAREST)
plot(m.out_NEAREST, type = "jitter")
plot(m.out_NEAREST, type = "hist")
```

modelu. Nastavit můžeme i to, jestli se má přiřadit pouze jedna kontrolní škola k dané podobné škole první skupiny škol, nebo více kontrolních škol. Dále je možné povolit, že k rozdílným školám používajícím alternativní učebnice se přiřadí více kontrolních škol, nebo se jedna kontrolní škola přiřadí k více školám první "experimentální" skupiny.

Po provedení výpočtu pak použijeme nový dataframe a provedeme standardní statistické metody analýzy dat, jako je t-test nebo regrese.

```
MODEL <- lm(PV1MATH ~ ALTERNATIVNI_UCEBNICE, data = out_NEAREST)
summary(MODEL )
```

11

Kvalitativní komparativní analýza (QCA)

11 KVALITATIVNÍ KOMPARATIVNÍ ANALÝZA (QCA)

Kvalitativní komparativní analýza je technika, která je na pomezí kvantitativních a kvalitativních metod. Cílem metody je určit kombinaci dostatečných a nutných podmínek, které vedou ke sledovanému důsledku. V současnosti se metoda používá i na datech, která mají kvantitativní povahu. V případě, kdy máme zhruba 50 až 200 případů, je dle metodologů vhodné využít QCA jako doplněk pro regresní modelování. QCA analýza je možná buď ve specializovaném programu Charlese Ragina, viz:

(https://www.socsci.uci.edu/~cragin/fsQCA/software.shtml),

nebo v RStudiu. V případě RStudia je nutné nahrát tyto balíčky:

```
library(QCA)
library(foreign)
library(KRLS)
require(pROC)

x <- read.csv("Vlastnídataset.csv", header = TRUE, sep=";")
```

Pro účely představení metody ukážeme tzv. csQCA, která počítá s indikátorovými dummy proměnnými. Proměnné tak musí být všechny na binární úrovni (1 a 0).

QCA sleduje komplexní kauzální vztahy, které nejsou založeny na korelaci, ale na konfiguraci podmínek. Výsledkem jsou kombinace nutných a dostatečných podmínek/příčin.

Důležité je znát logické výroky:

- Disjunkce = OR, +, | , (v některých softwarech ^).
- Konjunkce = AND,* , &.

Proměnné se v QCA nazývají causal conditions (nezávisle proměnné) a outcome (závisle proměnná)

QCA předpokládá:

- Asymetrické vztahy: pro vysvětlení Y=1 máme jiné podmínky než pro vysvětlení Y=0.
- Conjunctural causation: efekt proměnné A závisí na kombinaci dalších podmínek.
- Ekvifinalita: odlišné příčinné kombinace vedou v jiných případech ke stejnému výsledku.
- Multifinalita: odlišné důsledky mají v jiných případech stejnou příčinu.

Všechny tyto kauzální vztahy nedokážou statistické metody vhodně postihnout (interakční efekt 3 či 4 proměnných).

```
help("superSubset")
```

```
# 1. Krok –  analýza nutných podmínek
#######################################
condsfull <-
c("proměnná_1","proměnná_2","proměnná_3","proměnná_4","proměnná_5","proměnná_6",
"proměnná_7", proměnná_8)  # výběr všech nezávisle proměnných
superSubset(x, outcome="závislá_proměnná", neg.out=FALSE, conditions=condsfull,
        relation="necessity", incl.cut=0.95, cov.cut=0.5,
        use.tilde=FALSE, use.letters=FALSE)
superSubset(x, outcome="onset", neg.out=FALSE, conditions=condsfull, relation="necessity",
incl.cut=0.95, cov.cut=0.2, use.tilde=TRUE, use.letters=FALSE)
```

Pokud neznáme příkaz a jeho nastavení, je vhodné se podívat do nápovědy *Help*.

V ní najdeme popis, jak si příkaz upravit pro libovolná data a jaké parametry můžeme nastavit:

- outcome = název proměnné, která značí důsledek (závisle proměnnou),

- incl.cut = nastavení konzistence
- cov.cut = nastavení pokrytí (coverage)

```
# 2. Krok – analýza dostatečných podmínek
########################################
# Replikace tabulky 2 – analýza dostatečných podmínek

# Nejdříve vytvoříme pravdivostní tabulku (tt = truth table), která obsahuje všechny odlišné empiricky existující konfigurace kauzálních podmínek.
# Maximální možná počet kombinací podmínek, 2^k. V našem případě 2 na osmou kombinací (2^8), 256 kombinací! (k představuje počet nezávisle proměnných).
# 1. vytvoření pravdivostní tabulky
conds <- c("polx","oust","ruler","petrol","precon","tiny","instab","tekcon")
tt <- truthTable(x, outcome = "onset", conditions = conds, incl.cut1 = 0.7, #nastavení hranice konzistence (0.7, doporučuje se 0.8)
sort.by = c("incl","n"), complete = TRUE)
tt
```

Logická minimalizace

Minimalizace eliminuje všechny logicky zbytečné faktory tím, že postupně porovnává všechny páry případů. Minimalizace generuje tzv. prvotní implikanty (prime implicants) – minimální konfigurace podmínek nutných pro generování daného důsledku. Použití Millových metod k redukci "zbytečných" proměnných. Většina softwaru produkuje tři řešení konfigurací: Complex/Conservative, Parsimonious a Intermediate. Liší se v závislosti na tom, jak definujeme empiricky neexistující kombinace, "remainders" – kontrafaktuály. Obecně se doporučuje použití "intermediate solution".

Komplexní řešení – Complex solution / Conservative

```
sol.com <- minimize(tt, details = TRUE, use.tilde = TRUE)
sol.com
```

Střední řešení – Intermediate solution

U středního řešení ještě příkazem určíme, v jakém hypotetickém vztahu mají proměnné být. Minimalizace pak vyloučí ty konfigurace, které teoreticky nedávají smysl. Funkce dir.exp:

- musíme si zapamatovat pořadí proměnných, 1 pozitivní vztah, 0 negativní vztah.

Preferovaný výstup – střední řešení (intermediate solution term).

```
sol.int <- minimize(tt, include = "?", dir.exp = c(1,1,0,1,1,0,1,1), details = TRUE,
use.tilde = TRUE)
sol.int
```

Hodnocení každé kombinace příčin lze pomocí:

- konzistence (sloupec incls) – kolik procent případů empiricky odpovídá dané teoretické konfiguraci,
- pokrytí – kolik procent daná kombinace pokrývá příkladů.

Blíže ke QCA: https://cran.r-project.org/web/packages/QCA/QCA.pdf

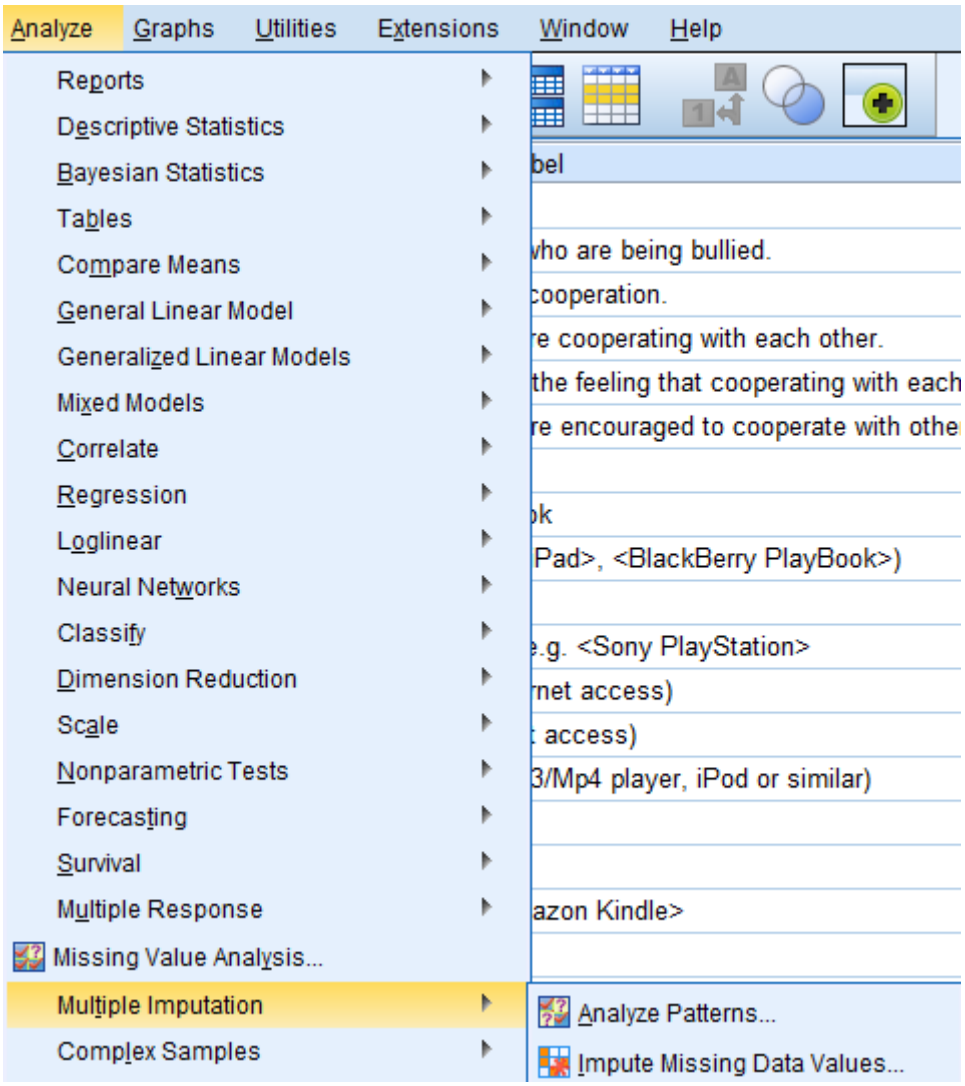
12

Chybějící hodnoty a jejich imputace

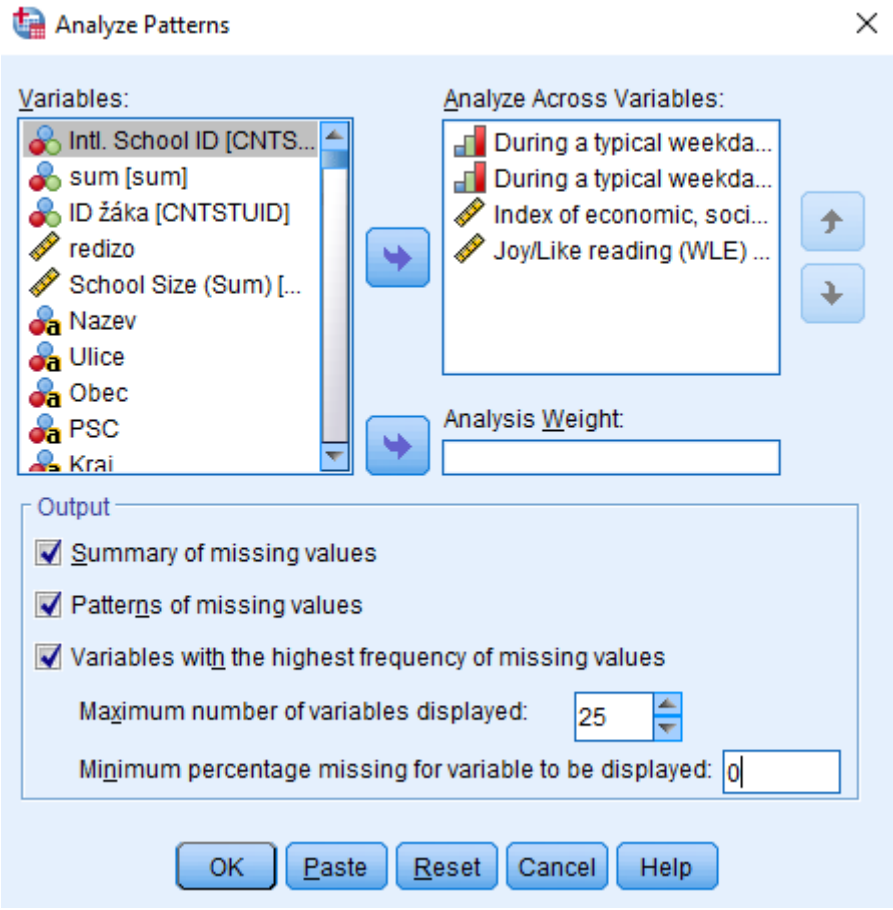
12 CHYBĚJÍCÍ HODNOTY A JEJICH IMPUTACE

12.1 Mnohonásobné imputace v SPSS

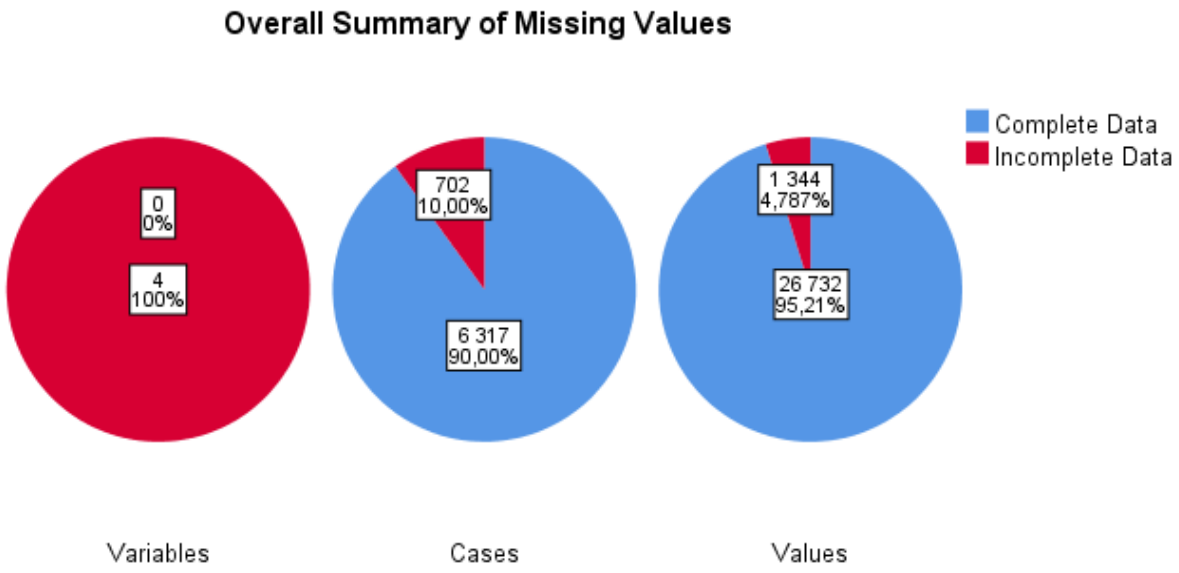
Mnohonásobné imputace v SPSS provádíme po rozkliknutí záložky *Analyze -> Multiple Imputation*.



Záložka *Multiple Imputation* umožňuje prozkoumat chybějící hodnoty (*Analyze Patterns*) stejně jako provést samotné imputace (*Impute Missing Data Values*). Provedení mnohonásobných imputací je v prostředí SPSS poměrně snadné. V podstatě totiž stačí vložit proměnné s chybějícími hodnotami včetně proměnných, na jejichž základě budou chybějící hodnoty imputovány, a SPSS proces dokončí automaticky i bez dalšího nastavení. Pro ilustraci lze využít datový soubor získaný z dotazování žáků v rámci mezinárodního šetření, z něhož je vybráno pár proměnných s chybějícími hodnotami.

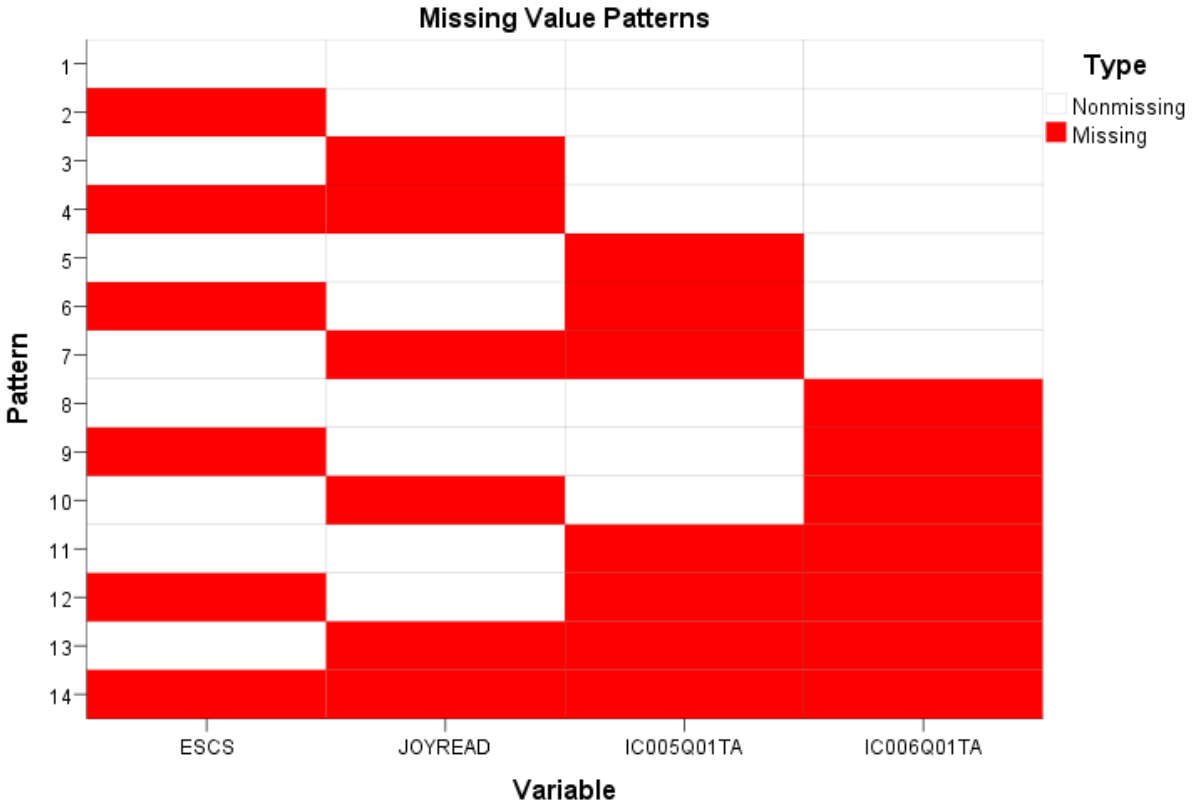


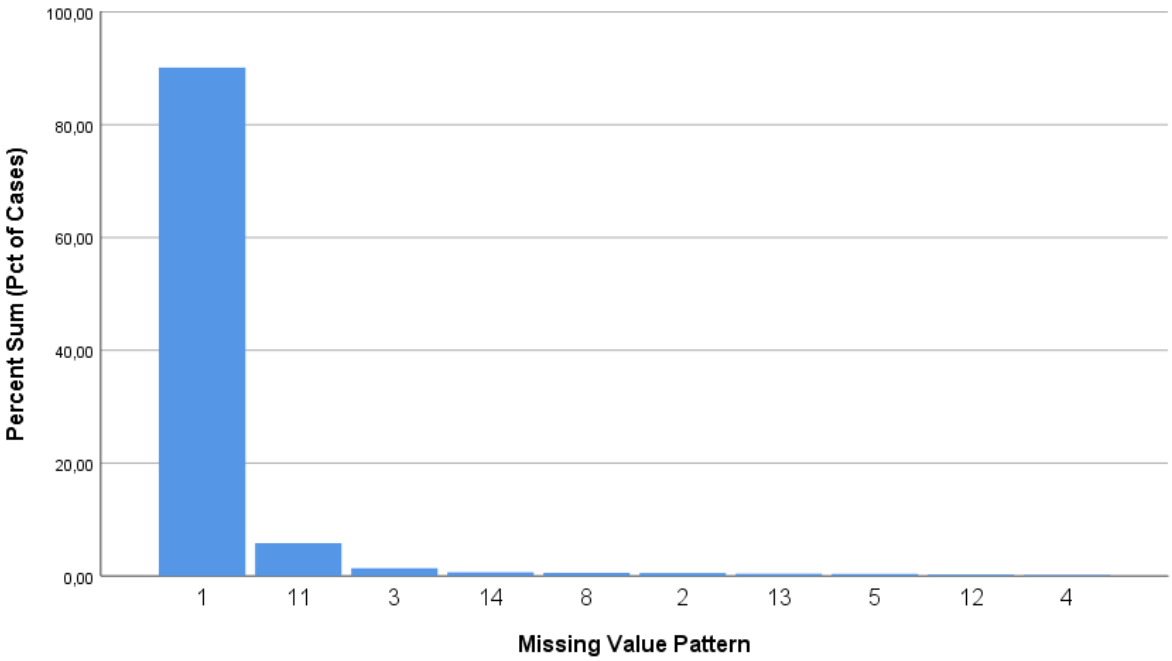
Před imputováním je vhodné chybějící hodnoty prozkoumat. K tomu slouží první možnost *Analyze Patterns* v záložce *Multiple Imputation*. Po vložení proměnných a případně váhy SPSS dále umožňuje upravit zobrazené výstupy v části *Output*.



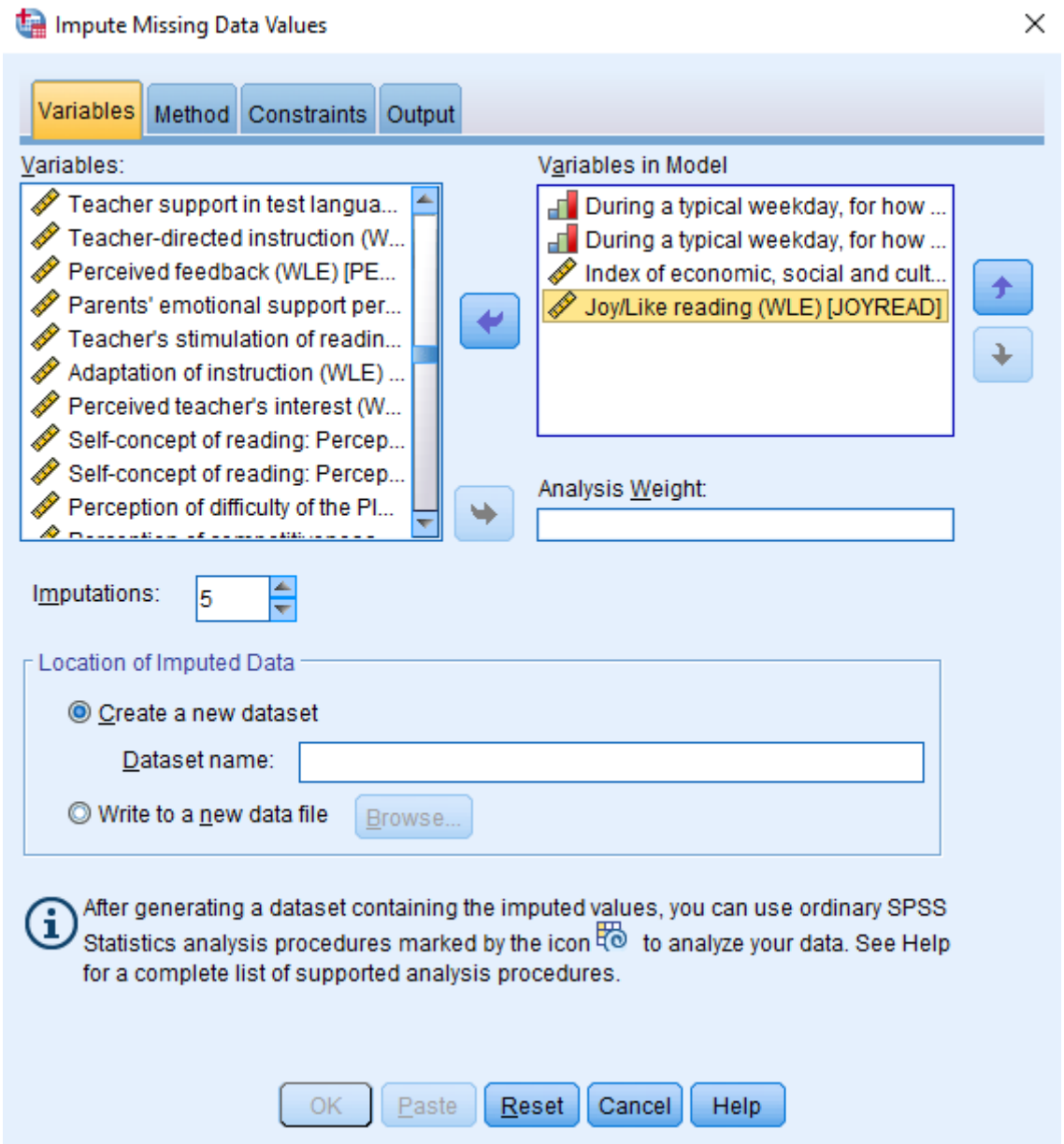
Variable Summary

	Missing				
	N	Percent	Valid N	Mean	Std. Deviation
During a typical weekday, for how long do you use the Internet outside of school?	536	7,6%	6483		
During a typical weekday, for how long do you use the Internet at school?	518	7,4%	6501		
Joy/Like reading (WLE)	182	2,6%	6837	,026812	1,1416319
Index of economic, social and cultural status	108	1,5%	6911	-,085855	,8811155





Pokud necháme všechny možnosti zaškrtnuté, získáme sadu zde vložených výstupů. První sada koláčových grafů zobrazuje celkový souhrn chybějících hodnot v rámci proměnných, případů a hodnot. Následující tabulka zobrazuje, kolik případů má u seznamu zvolených proměnných chybějící hodnotu a kolik tvoří procent. Dále jsou zobrazeny vzorce chybějících hodnot, tedy různé kombinace chybějících hodnot napříč sledovanými proměnnými, které se v datech vyskytují. Poslední graf poté demonstruje, jakého podílu případů se dané vzorce týkají.



V dalším kroku již následuje provedení imputací, které zadáváme přes *Impute Missing Data Values*. Hlavní okno slouží k zadání proměnných, váhy, počtu provedených imputací a instrukce, jak naložit s imputovanými daty. Jak je naznačeno dříve v textu, vyplnění tohoto okna k provedení imputací postačí. Pokud bychom nyní ještě pojmenovali nový datový soubor, vysvítila by se možnost OK, jejímž stisknutím bychom proces mohli dokončit. Přesto však může být užitečné se alespoň v krátkosti podívat na to, jaká nastavení lze činit skrze zbylé záložky v okně nahoře. Samostatné provedení mnohonásobných imputací je umožněno skrze automatickou volbu metody ve druhé záložce *Method*. Po rozkliknutí toto nastavení můžeme změnit. V další záložce *Constraints* můžeme upravovat roli proměnných v modelu či vyřazovat proměnné s velkým počtem chybějících hodnot. Nakonec v záložce *Output* upravujeme, jaké výstupy chceme zobrazit. Výsledný výstup tedy odpovídá tomu, co zde zvolíme. Souběžně s tím se vytvoří nový datový soubor s imputovanými hodnotami. Jak je uvedeno v zadávacím okně mnohonásobných imputací dole, různé techniky jsou v tomto souboru označeny ikonou, která poukazuje na to, že jsou pro imputovaný datový soubor vhodné. Výsledky jsou následně zobrazeny pro každý soubor zvlášť a nakonec rovněž dohromady.

12.2 Mnohonásobné imputace v R

Po nahrání datového souboru PISA2015 nejdříve zkontrolujeme, jestli R správně načetlo úrovně měření, protože algoritmus potřebuje vědět úroveň měření dané proměnné, rozdíl mezi ordinální a kategorickou škálou atd.

```
lapply(PISA2015, class)
```

Po spuštění funkce jsme zjistili, že tři proměnné nejsou správně zařazeny. U proměnné 1 chceme, aby R počítalo s tím, že se jedná o ordinální škálu, ne o číselnou (scale) proměnnou. V případě proměnné 2 chceme, aby R považovalo tuto proměnnou za číselnou. V případě proměnné 3 potřebujeme, aby R považovalo proměnnou za kategorickou.

Následuje příkaz z balíčku naniar, který ukáže vzorec chybějících hodnot pomocí funkce gg_miss_var.

Pomocí funkce vis_miss zjistíme, jak jsou chybějící hodnoty rozloženy.

Samotné mnohonásobné imputace provedeme pomocí balíčku mice.

```
PISA2015$Proměnná_1 <- as.ordered(PISA2015$Proměnná_1)
PISA2015$Proměnná_2 <- as.numeric(PISA2015$Proměnná_2)
```

```
library(mice)
matice_prediktoru <- quickpred(PISA2015, mincor = 0.1,
                        include = c("ESCS",  "MOTIVAT",),
                        exclude = c("ANXTEST",))
dim(matice_prediktoru) # Kontrola matice
imp_PISA <- mice(data = PISA2015, m = 10, maxit = 20, predictorMatrix = matice_prediktoru,
            diagnostics = TRUE, nnet.MaxNWts = 2500, seed = 1234)
```

M je počet imputací, kdy se doporučuje 10. Maxit je počet iterací, kdy optimální hodnota je zhruba 20. Seed je generátor náhodných čísel, zde stačí dát libovolné číslo. "nnet.MaxNWts = 2500" představuje nastavení parametrů při nahrazování nominálních kategorických proměnných s více než dvěma kategoriemi. Algoritmus defaultně počítá s tím, že pro kardinální proměnné se použije predictive mean matching, pro binární proměnné logistická regrese, pro kategorické proměnné multinomická regrese a pro ordinální proměnné ordinální logistická regrese. Následně po výpočtu musíme imputované hodnoty uložit do dataframu pomocí příkazu complete.

```
PISA2015_IMPUTED <- complete(data = imp_PISA , action = "long", include = TRUE)
```

Nyní spočítáme základní regresní model na imputovaných datech.

```
MODEL_IMP <- with(data = PISA2015_IMPUTED , expr = lm(PV1MATH ~ ESCS +
MOTIVAT))
print(MODEL_IMP )
```

Modely pak sloučíme příkazem pool.

```
pool(MODEL_IMP)
summary(pool(MODEL_IMP))
```

13

Závěry a doporučení

13 ZÁVĚRY A DOPORUČENÍ

Sada typizovaných analytických nástrojů v prostředí standardních statistických programů představila nejen běžně používané statistické techniky, ale i analyticky složitější postupy při analýze dat. Dokument kladl důraz i na možnosti vizualizace datových analýz, které pro běžného čtenáře a uživatele analytických výstupů ČŠI mohou být složité. V tomto ohledu je třeba co nejvíce datových analýz vizualizovat. Například regresní modely pomocí grafu regresních koeficientů, korelační vztahy pomocí korelačních matic či Gaussova grafického modelu. V tomto ohledu je velmi silným nástrojem program RStudio, který disponuje balíčky pro vizualizaci statistických technik. Je nutné podotknout, že vizualizace dat z programu R sleduje jednoduchost a "akademický" styl prezentace dat. Pro kvalitnější vizualizaci a doplnění o prvky zvyšující srozumitelnost je nutné grafy, tabulky a schémata z programu R exportovat ve formátu vektorové grafiky (např. .svg) a pomocí příslušných programů (Adobe Illustrator, Affinity Designer či jiný editor vektorové grafiky) dále doupravit dle účelu použití.

Datová analýza a statistická analýza dat se velmi dynamicky vyvíjí. To, co před pár lety software neumožňoval, nyní můžeme pomocí doplňku využít při vlastní analýze dat. I program SPSS přinesl některá vylepšení a to, co zatím neumožňuje, jde řešit spojením SPSS a programu R či pomocí dalších rozšíření SPSS, které jazyk programu R využívají.

Předkládaný dokument není vyčerpávajícím přehledem všech statistických postupů a analýz. Nicméně představuje nejběžnější statistické techniky analýzy dat, které se využívají nejen v mezinárodních šetřeních, jako je PISA, ale které se ukázaly být využitelné také v rámci dílčích analýz ČŠI a analýz na vlastních datech (Kvalita vzdělávání v jednotlivých krajích ČR či Důležité faktory vzdělávací soustavy v kontextu prostorových dat českých okresů).

Přestože většina postupů a skriptů vznikla na základě zkušenosti v práci s daty, kdy některé funkce v R byly naprogramovány přímo autorským týmem, je vhodné znalosti skriptů dále rozšířit. V tomto ohledu odkazujeme na důležitou literaturu v oblasti datových analýz. Zatímco základní učebnice statistických metod využívají softwaru SPSS, nejnovější publikace v oblasti datových analýz se již dominantně spoléhají na prostředí RStudia. Jmenovitě knihy statistiků Andrew Gelmana a Jennifer Hillové *Hierarchical regression model* či nejnovější kniha *Regression and other stories*. Patrně nejvýznamnější teoretik regresních modelů John Fox spolu s kolegou Sanfordem Weisbergerem publikovali nedávno knihu *An R companion to applied regression*. V oblasti vizualizace dat je stěžejní kniha statistika Hadleyho Wickhama, který nyní pracuje pro RStudio, s názvem *R for Data Science*. Hadley Wickham ve svém balíčku ggplot implementuje myšlenky vlivné knihy autora Lelanda Wilkinsona *The Grammar of Graphics*, která se zabývá způsobem vizualizace datových analýz a dat. Vydavatel akademické odborné literatury Springer nyní v řadě věnované R eviduje 79 knih o statistických metodách v různých vědních oblastech.[14]

[14] Dostupné zde: https://link.springer.com/search?facet-series=%226991%22&facet-content-type=%22Book%22

L

Vybraná literatura

VYBRANÁ LITERATURA

Gelman, A., Hill, J., & Vehtari, A. 2020. Regression and Other Stories. Cambridge University Press.

Dunning, T. 2008. „Improving Causal Inference: Strengths and Limitations of Natural Experiments.“ *Political Research Quarterly* 61 (2): 282–293.

Ragin, C. 1987. *The Comparative Method: Moving Beyond Qualitative and Quantitative Strategies.* Berkeley: University of California Press.

Gelman, A., & Hill, J. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

Wickham, H., & Grolemund, G. 2016. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly Media, Inc.

Fox, J., & Weisberg, S. 2018. *An R Companion to Applied Regression*. Sage publications.

Soukup, P. 2021. „Faktorová analýza jako známá neznámá (aneb metoda hlavních komponent a varimax není vždy ideální postup).“ *Sociologický časopis* 57 (4): 1–30.

ČŠI | Česká školní inspekce

Fráni Šrámka 37 | 150 21 Praha 5 | www.csicr.cz