Ivo Züchner

Statistische Datenanalyse mit BlueSky Statistics – Einführung in die Anwendung der graphischen Benutzeroberfläche zu R (Version 10)

Stand April 2022

Vorwort

Mit diesem Text wird die Statistik-Software *BlueSky Statistics* vorgestellt und eine Einführung zu deren Anwendung für interessierte Nutzer*innen gegeben. Entstanden ist dies aus Arbeitszusammenhängen an der Philipps-Universität Marburg, über die Anforderungen von Statistikseminaren und in der Zusammenarbeit mit studentischen Hilfskräften.

Entsprechend ist diese Einführung für interessierte Statistikanwender*innen geschrieben, die mit oder ohne Erfahrung in proprietärer Statistik-Software (wie bspw. SPSS©) einfache oder komplexere statistische Auswertungen mit quantitativen Datensätzen oder auch Ergebnisgrafiken erstellen wollen. BlueSky Statistics ist eine Application zur Nutzung der Softwareumgebung R, die dynamisch weiterentwickelt wird, für die aber bislang nur vereinzelte Ressourcen zur Einführung in die Anwendung vorhanden sind.

Die meisten Hilfen finden sich als Präsentationen und Trainingsvideos über die Projekt-Website (vgl. https://www.blueskystatistics.com/Articles.asp?ID=301), für deutschsprachige Nutzer:innen sind hier ein paar Grundlagen zusammengestellt, um Interessierten den Einstieg zu erleichtern. Die nachfolgende inhaltliche Beschreibung lehnt sich an ein Online-Review von Robert A. Muenchen von der University of Tennessee zu Vorgängerversion („A Comparative Review of the BlueSky Statistics GUI for R“ (Muenchen 2020a)). Muenchen macht schon seit längerer Zeit die Mühe macht, grafische Nutzeroberflächen (GUIs) für R zu beschreiben und zu vergleichen. Er hat auch umfangreiche „User Guide“ für BlueSky Statistics 7.1 und auch 10.0 online gestellt (vgl. Muenchen 2021). Ein Buch über den Einsatz von Raschmodellen mit R und BlueSky Statistics in den Sozialwissenschaften hat Lamprianou (2019) publiziert.

Selbstverständlich ersetzt dieser Text weder eine systematische Einführung in Statistik bzw. in die für die nachfolgende erwähnten statistischen Verfahren, hier wird nur die Anwendung in BlueSky Statistics beschrieben. Auch kann hiermit keine Einführung in R gegeben werden, für einen praktischen Einstieg inklusive Hintergründe und Erklärung der Logik der Programmiersprache vgl. bspw. Luhmann (2015).

Das vorliegende Dokument bezieht sich auf die Version 10 von BlueSky Statistics, die sich gegenüber den Vorgängerversionen 7.X doch deutlich unterscheidet. Insgesamt lehnt sie sich BSS nun stärker an GUIs wie RStudio an und vereint die verschiedene Arbeitsfenster in einer Gesamtansicht.

Inhalt

Einleitung

BlueSky Statistics (im Folgenden auch BSS abgekürzt) ist eine Benutzeroberfläche für die Statistik-Software-Umgebung R. Sie ist sowohl in einer Open-Source Variante als auch in einer kommerziellen Variante (die technischen Support und eine Version für Windows-Terminalserver wie Remote Desktop oder Citrix enthält) erhältlich (Download unter https://www.BlueSkyStatistics.com/Articles.asp?ID=301), derzeit wohl nur für Microsoft Windows. Hinter BSS steht die Firma BlueSky Statistics mit Sitz in Chicago – über die allerdings nicht viel im Internet herauszubekommen ist.

BSS bietet sich für Benutzer*innen an, die mit R arbeiten möchten und bislang vorwiegend Erfahrungen mit SPSS, Stata oder SAS gesammelt haben und/oder vor der R-Syntax zunächst zurückschrecken. Mit BSS kann eine große Bandbreite an Datenmanagement und Auswertungsverfahren als auch grafischen Darstellungen über das Menü aufgerufen werden, die Logik des Programms ist dabei sehr an SPSS orientiert.[1] München (2020) schreibt, dass die Entwickler*innen zum Teil früher an SPSS kommen gearbeitet sind und das sieht man dem Programm schnell an, viele Funktionalitäten sind ähnlich oder gleich angelegt, auch wenn die Befehlsstruktur auf R basiert, so dass gerade SPSS-erfahrende Anwender*innen wenig Einstiegsschwierigkeiten haben sollten.

BlueSky Statistics bietet sowohl die Möglichkeit, per „point & klick“ statistische Auswertungen Datenmanagement und Grafiken vorzunehmen oder diese über R-Syntax im R Editor anzufordern.

Die Nutzung von BlueSky Statistics als GUI bietet sich aus Sicht des Autors an,

- wenn Kosten von proprietärer Software gespart werden soll
- wenn sowohl eine Statistische Analysen mit einer GUI angestrebt werden und gleichzeitig die vielen Möglichkeiten und Pakete von R genutzt werden sollen
- wenn ein Übergang von SPSS zu R angestrebt wird
- wenn schon mit R gearbeitet wird, aber gerade der Transfer und Weiterbearbeitung von Ausgaben in Office-Programme Probleme bereitet
- wenn mit R gearbeitet werden soll, aber noch viel Datenaufbereitung notwendig ist und R noch nicht gut beherrscht wird
- wenn es in kooperativen Arbeitsformen wichtig ist, die jeweiligen Arbeitsschritte zu dokumentieren und wiederholen zu können.

Unbestritten ist, dass es auch andere und immer mehr GUI für R gibt – wie bspw. R Commander oder jamovi -, das hier vorgestellte BlueSky Statistics ist eine sehr funktionale Option mit einem großen Umfang klickbarer Routinen für Datenmanagement und Analysen. Gerade das einfache Datenmanagement macht es aus Sicht des Autoren sehr attraktiv. Mit BlueSky Statistics kann über das Syntaxfenster und mit dem R-Code auch die zugrundeliegende R-Version direkt verwendet werden – dann erscheint im Ausgabefenster aber auch „nur“ die klassische R-Ausgabe. Für elegantere und vor allem leichter exportierbare Ausgaben verwendet BlueSky Statistics zum Teil modernere und teilweise auch spezifisch auf BSS angepassten R-Code (s.u.).

[1] An dieser Stelle soll nicht die Diskussion geführt werden, ob es nicht sinnvoller ist, R ohne ein Klick- & Point-Menü zu erlernen, weil dieses ein tieferes Verständnis erzeuge und langfristig für die Ausschöpfung der Möglichkeiten von R notwendig ist. Von BSS ausgehend ist ebenfalls ein weiterer Einstieg in R möglich, und gerade für die Arbeit im Team bzw. für Personen, die nicht längerfristig mit statistischen Auswertungen arbeiten, sprechen nach der Meinung des Verfassers die leichte Zugänglichkeit und Vergleichbarkeit mit SPSS, die gute Reproduzierbarkeit der Operationen, ein recht einfaches Datenmanagement als auch der einfache Export von Ergebnissen für die Nutzung von BSS.

Teil 1: Übersicht über Funktionen und „Handling“ des Programms

0. Installation und Aufbau des Programms/R-Pakete

Die *Hauptinstallation* von BlueSky kann nach dem Download (von https://www.BlueSky Statistics.com/) in einem einzigen Schritt durchgeführt werden. Das Installationsprogramm beinhaltet auch eine Version von R, die die Kompatibilität zwischen BlueSky und dem R Grundprogramm gewährleistet. Diese wird auch dann installiert, wenn sich auf dem Rechner schon eine R Installation befindet.

Zentrale operative Basis von BlueSky Statistics ist R bzw. die Paketstruktur von R. BlueSky Statistics zielt darauf, R „anwendungsfreundlich“ über die Menüführung zugänglich zu machen, kann aber auch über Syntax/R-Code gesteuert werden. Mit dem Syntaxfenster (s.u.) von BlueSky Statistics kann über die in der Menüführung angebotenen Optionen der volle Funktionsumfang von R genutzt werden, wenn entsprechende weitere R-Pakete (über die Menüleiste oder die Syntax) installiert werden. Für Pakete, die noch keinen Eingang in die Menüführung gefunden haben, muss allerdings mit der Syntax wie in R oder RStudio gearbeitet werden. Die Befehle werden hierfür im Syntaxfenster eingegeben, der Output findet sich im Outputfenster (s.u.), dieser ist dann allerdings nicht extra formatiert und entspricht dann den Ausgaben, wie sie auch in R oder RStudio ausgegeben werden.

I. Die grafische Benutzeroberfläche (Graphical User interface – GUI)

Beim Start von BlueSky Statistics erscheint die Arbeitsoberfläche, die in einer Gesamtansicht einen Dateneditor (der wie SPSS mit Datenansicht und Variablenansicht zwei Ebenen hat), ein Outputfenster sowie unter dem Dateneditor eine Zeile zur Eingabe von Syntaxbefehlen („R Editor“) enthält.

I.1 Startansicht und Fensterebenen

BlueSky Statistics beginnt, indem auf dem Hauptbildschirm links ein Feld mit Datenansicht und rechts das Ausgabefenster (Abbildung 1) angezeigt wird. Unter der Datenansicht findet sich als drittes, zunächst unscheinbares Element eine Bedienzeile R Editor (s.u.). Über dem Datenfenster findet sich eine typische Reiterstruktur, in unterschiedliche Ebenen aufgerufen werden können. Der erste Reiter mit den drei Querbalken führt zu Schaltflächen, die ganz grundsätzlich die Verwendung von Datensätzen betreffen.

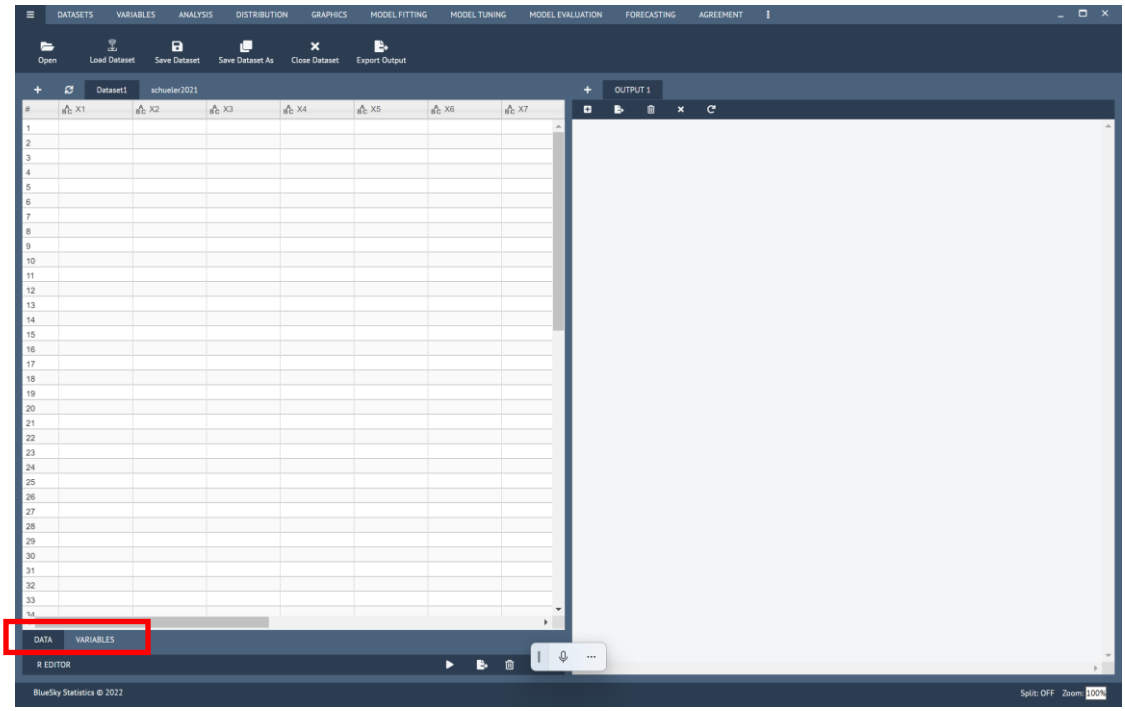


Abb. 1 Hauptansicht mit der Option Datenansicht

Unter dem Dateneditor-Fester und über der Zeile R Editor finden sich zwei Reiter mit den Bezeichnungen "Data" und "Variables". Die Registerkarte "Data" wird bei Erstaufruf standardmäßig angezeigt, mit einem Klick auf die Registerkarte "Variables" wechselt man zur Variablenansicht, die die Metadaten anzeigt. Es können gleichzeitig mehrere Datensätze geöffnet werden, diese werden dann über dem Datenfenster als Reiter Dataset 1, Dataset2 bzw. mit dem jeweiligen Namen angezeigt. Der jeweils helle Reiter zeigt den für die Menüführung aktuellen Datensatz an. Ein Wechsel zwischen den Datensätzen ist jederzeit möglich, eingegebene Befehle oder geklickte Optionen beziehen sich auf den jeweils sichtbaren.

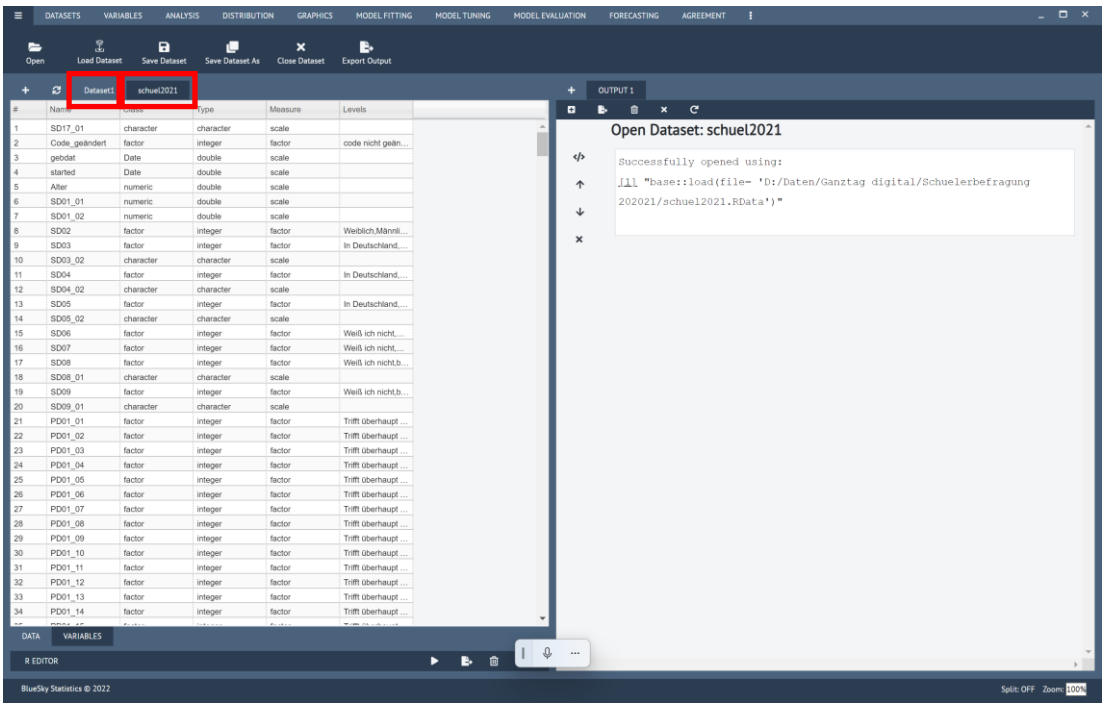


Abb. 2 Hauptansicht mit der Option Variablenansicht

I.1.1 Dateneingabe

BlueSky Statiscs wird beim Start mit einem leeren Datenfenster geöffnet, ein neues leeres kann mit dem „+“ Zeichen über dem Datenfenster aufgerufen werden. Alle Variablen tragen zunächst den Namen x1 x2 usw. Eingegeben werden können zahlen oder Buchstaben, können auch Tabellen aus anderen Programmen in das Datenfenster eingefügt werden. Auch können im Datenfenster neue Zeilen eingefügt werden. Bei Faktorvariablen macht es Sinn, diese zunächst als Werte einzugeben. In der Variablenansicht kann dann daraus ein Faktorgemacht und die Ausprägungen als values definiert werden.

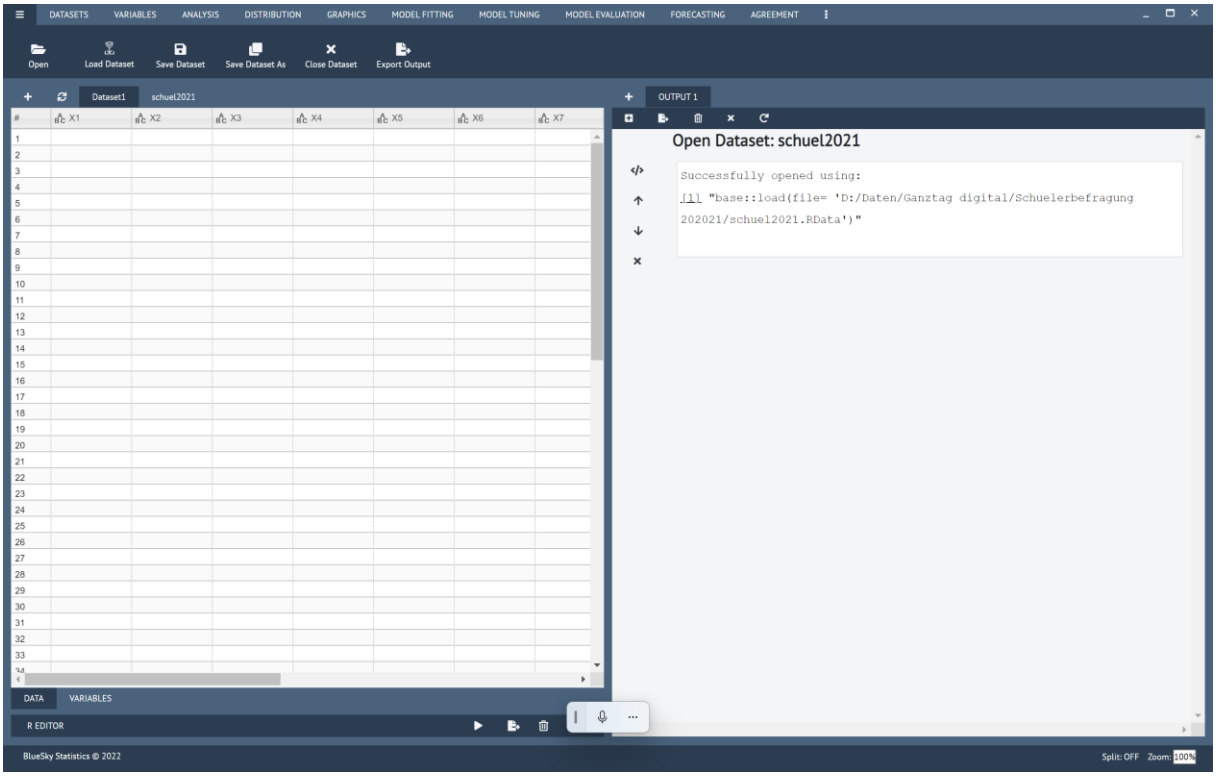


Abb. 3 Hauptansicht mit leerer Variablenansicht

Die *Datenspeicherung* erfolgt im Menü mit “Save Dataset”. Standardmäßig speichert BSS neue Datensätze als R-Data Object, unter “Save Dataset as” wird aber auch eine Speicherung unter andere Datenformaten (SAS, SPSS, Stata, Excel, CSV) angeboten (vgl. Punkt I.1.3).

Die Bearbeitung der Variablen erfolgt – wiederum anlog zu SPSS – in der Variablenansicht (vgl. Abb. 2): Hier können neue Variablen angelegt werden und Datenformate und Anzeigen sowie die Wertelabels festgelegt werden.

Unter der **Variablenansicht** kann – wie bspw. in SPSS – jede einzelne Variable näher definiert werden (mit rechter Maustaste):

- Name = Kurzname (bspw. Fragebogennummer, sinnvoll sind kurze Namen, gerade wenn auch mit der Syntax gearbeitet wird)
  Class = Hier wird zwischen faktor (gestufter, kategorialer Variable), numeric (metrischer Variable) oder character (Zeichenvariable) unterschieden.[2]
- Type = Feindifferenzierung der Datenart, es gibt bei R hier eine Differenzierung numerisch in „integers“ (ganze Zahlen) und double („Gleitkommazahlen“) gemacht
- Measure = Beschreibt das Skalenniveau, mit den drei Standardausprägungen „numeric“, „scale“ und „ordinal“.
- Levels = Namen der Ausprägungen bei ordinalen und kategorialen Daten, z.B. 1= Mann, 2

[2] R unterscheidet zwischen Faktorvariablen (abgestufte Variablen) und Vektorvariablen (metrische Variablen). Faktorvariablen können über die Syntax auch als ordinal definiert werden (s.u.).

= Frau, 3= Divers) (Wertelabels bei SPSS), diese können im Fenster selbst geändert werden, allerdings nur bei Faktoren.

Die Variablenart kann in ihrem Typ über die rechte Maustaste mit „make numeric oder „make factor“ oder mit „make string“ geändert werden. Falls für ein Faktorvariable ein ordinales Skalenniveau festgelegt werden soll (bspw. für eine ordinale Regression), muss dies mit einem R-Syntax Befehl erfolgen (ein Syntaxbeispiel hierfür findet sich unter V.3 Ordinale Regression).

I.1.2 Datenimport

Bestehende Daten können leicht in BSS importiert werden, dies geschieht dateiformatübergreifend einfach über das Menü mit dem Button „Open”.

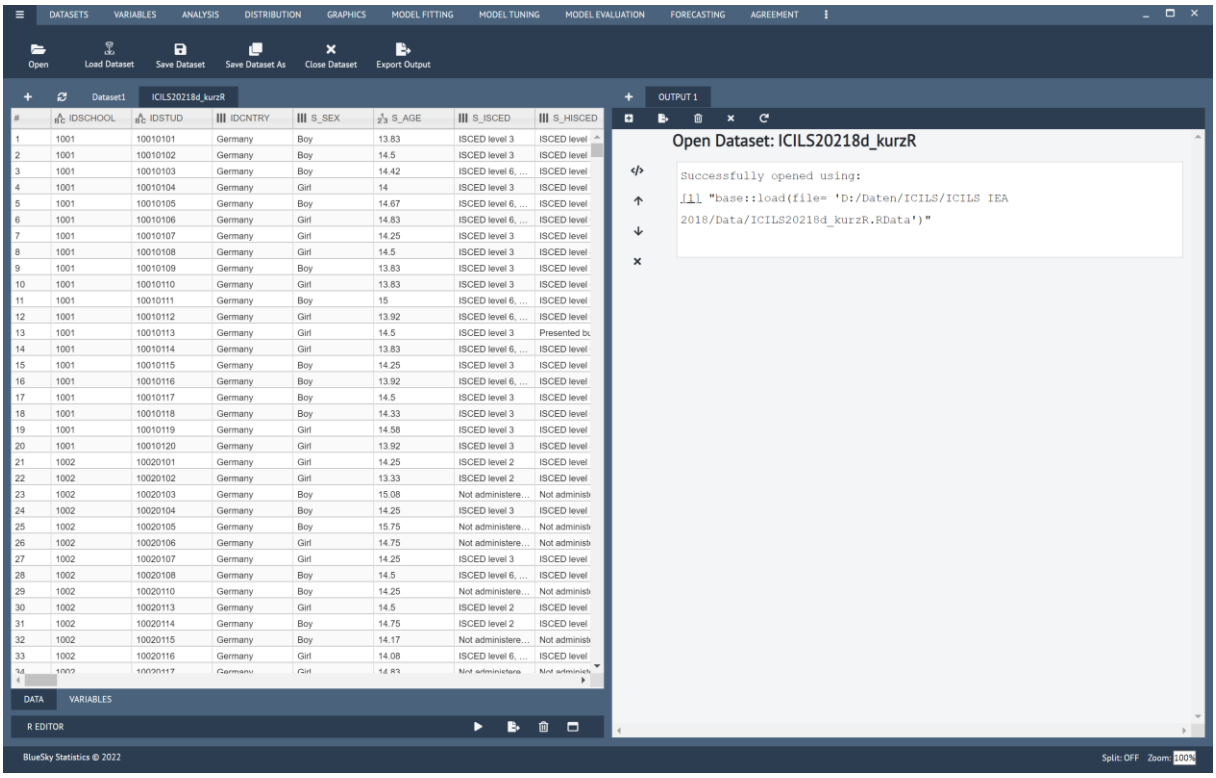


Abb. 4 Öffnen von bestehenden Daten (verschiedener Formate)

Die Open-Source Version von BlueSky Statistics unterstützt dabei laut München (2022) die folgenden Formate

- Comma Separated Values (.csv)
- einfache Textdateien (.txt)
- Excel (alte xls und neue xlsx Dateien)
- dBase’s (dbf)
- SPSS (sav)
- SAS binary files (sas7bdat)
- Stata (dta)
- Standard R workspace Dateien (RData)

Nach dem Import kann die Datendatei dann als R-Datei (RData) abgespeichert werden, eine Speicherung im Original ist den folgend angegebenen Formaten (und für die verbreitetsten Statistikprogramme) aber ebenfalls möglich.

I.1.3 Speicherung/Datenexport

BlueSky Statistics bietet die Speicherung/den Export der Arbeitsdaten in andere Formate/für andere Programme an. Abgespeichert wird die jeweilige Datensatz im Menü und dem ersten Reiter mit den drei Balken dann über „Save dataset as“. Die verbreitetsten Statistikprogramme SPSS, Stata und SAS sind hier berücksichtigt Die angebotenen Formate sind:

- Comma Separated Values – *.csv
- Excel – *.xlsx
- R Objects – *.RData
- SPSS – *.sav
- Stata – *.dta
- SAS – *. sas

I.2 Outputfenster

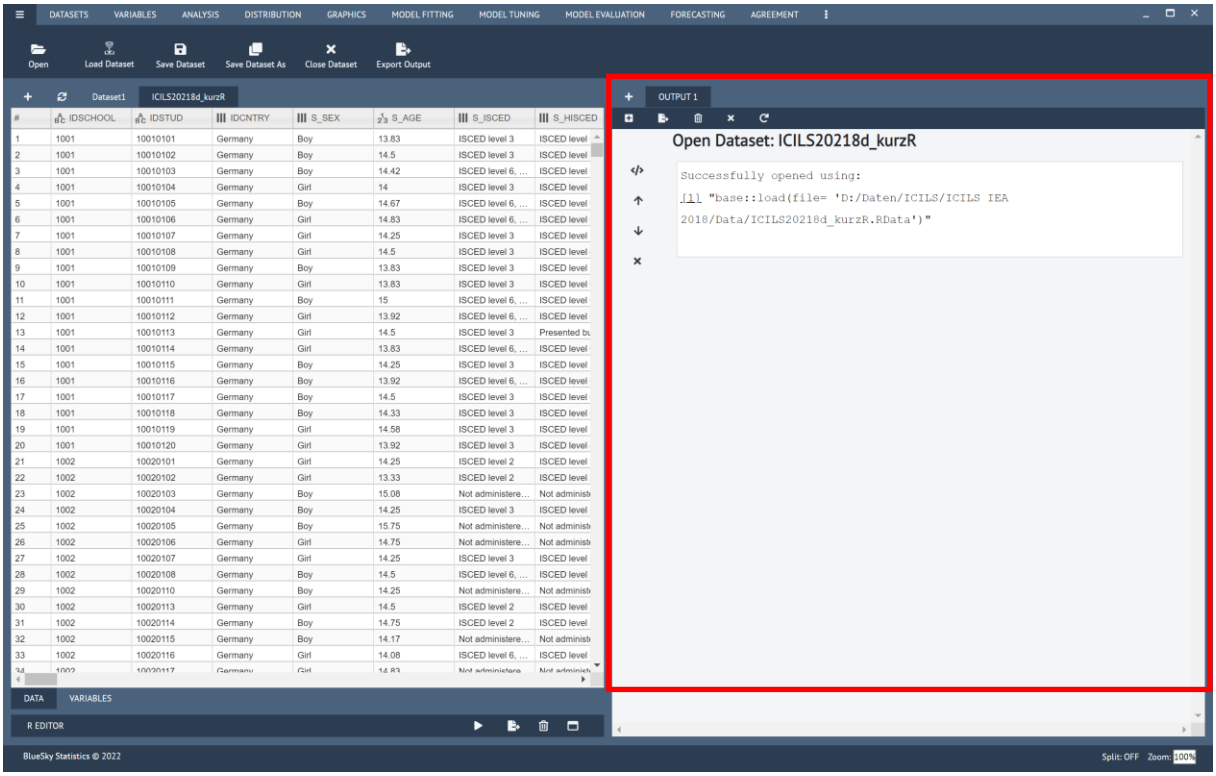


Abb. 5 Outputfenster

In dem Fenster auf der rechten Seite werden die (möglichen) Inhalte der Ausgabe angezeigt, über entsprechendes Anklicken der Elemente kann der Output erweitert werden. So kann der Output verändert werden in dem bspw. die verwendete Syntax zusätzlich angezeigt (und kopierbar) wird (durch Klicken auf </>), mit Klicken auf das x wird der Output gelöscht.

Ein Vorteil von BSS ist, dass die über die BBS-Befehle (Menü oder BSS-Syntax) erzeugten Tabellen neben der Exportfunktion des Outputs auch einzeln als Tabellen exportierbar sind – wenn der output mit der Maus unterlegt und kopiert wird, kann eine Tabelle bspw. in Excel oder word eingefügt werden.

I.2 Arbeiten mit der Syntax

Nach dem Start wird unter dem Datenfester die schmale Zeile R Editor angezeigt. Am Ende dieser Zeile finden sich 4 Klickoptionen, die Vierte („Toggle R Editor“) vergrößert die Zeile zu einem Fenster und bietet damit ein Syntaxfenster, vom dem aus alle Befehle mit der R Syntax gestartet werden können.

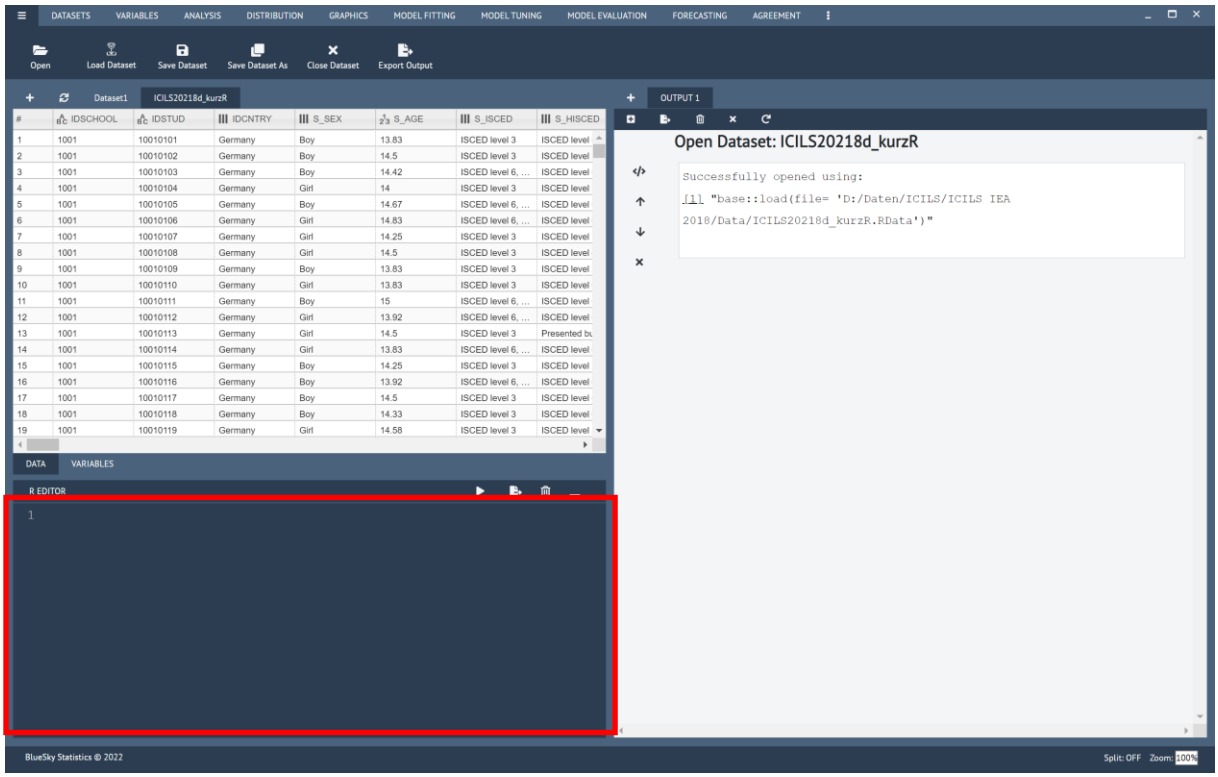


Abb. 6 Editorfenster

Dabei kann direkt R Code hineingeschrieben werden, R Code als Syntaxdatei importiert (über den Reiter Open oben links im Gesamtfenster= oder als Syntax exportiert werden. Über diesen R Editor (=Syntax-Fenster) ist BlueSky Statistics „ganz normal“ mit Syntaxeingabe wie in R zu bedienen (wobei bei Anwendung klassischer Syntax dann allerdings die komplexeren Formatierungen der Tabellen im Outputfenster entfallen, hier lohnt es sich – wenn möglich – die BSS Syntax zu verwenden). Der R Editor unterstützt das Erstellen der Syntax durch farbliche Hervorhebungen von Befehlen, Kommentaren und Variablen. Die Syntax kann als R Skript exportiert und gespeichert werden und dann in jeder R-Umgebung ausgeführt werden.

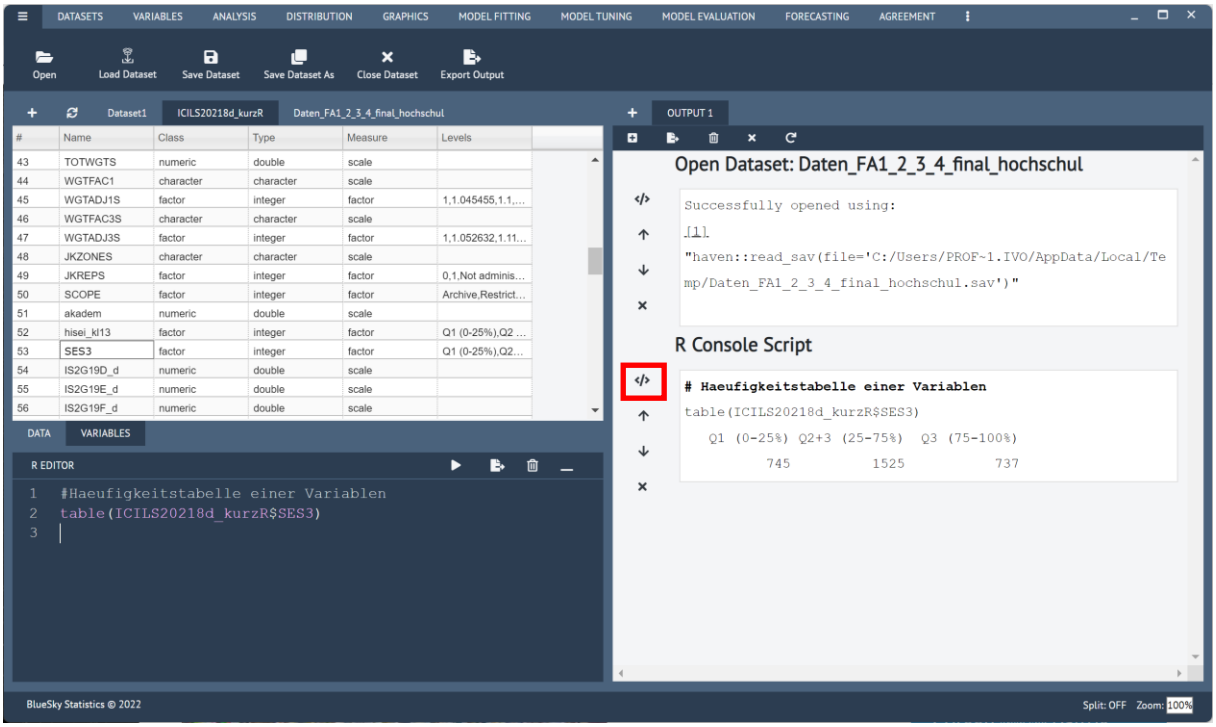


Abb. 7 Ausgabefenster mit Output

Syntax von geklickten bzw. ausgeführten Befehlen wird immer mit in das Outputfenster geschrieben und kann wie schon erwähnt durch das Anklicken des </> im Outputfenster ober8

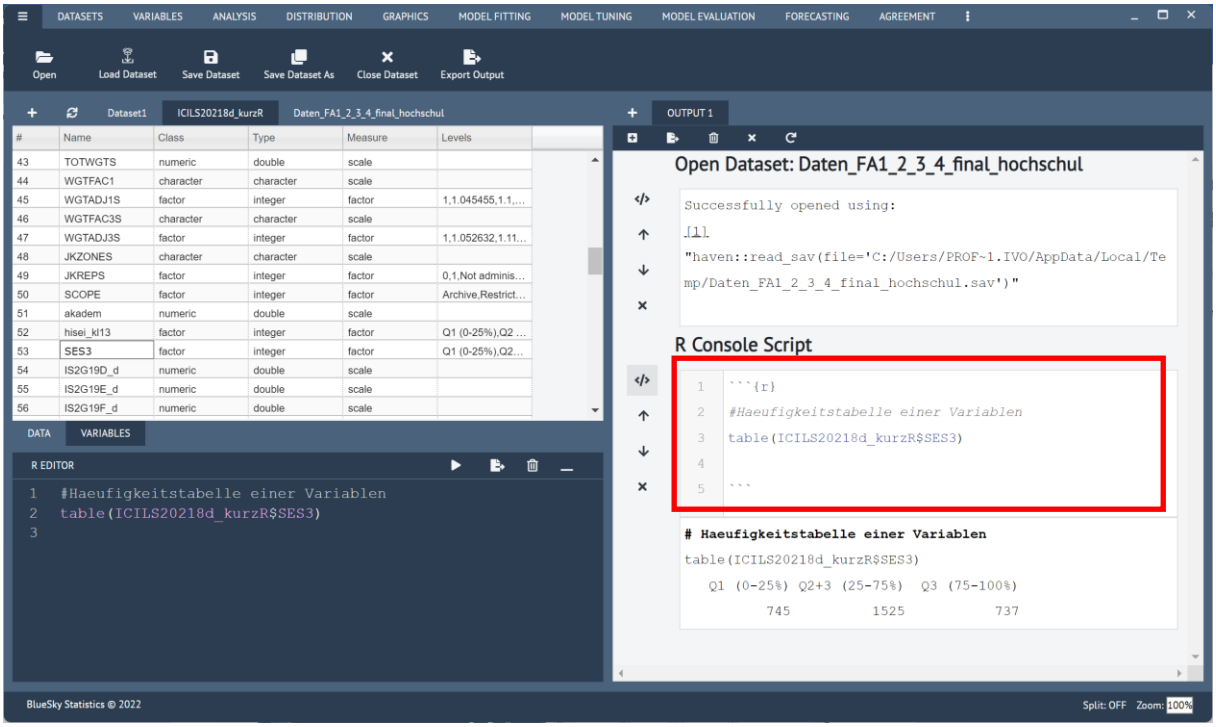


Abb. 8 Ausgabefenster mit Syntaxeditor

I.4 Menüs & Dialogfenster

Die Menüführung von BlueSky Statistics verwendet in der Kopfliste des Programms Reiter. Diese beginnen mit den drei Balken, dann folgen die Reiter mit „Datasets“ „Variables“ Analysis“ „Distribution“ etv. (s. Abb.)

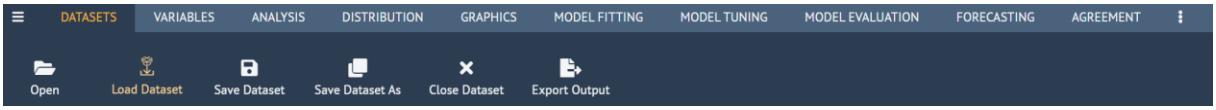


Abb. 9 Menüleiste

Für jeden dieser Reiter werden dann entsprechende größere Symbolfelder angezeigt, mit denen bestimmte Prozeduren aufgerufen werden können, hier bspw. „Open“ „Load Dataset“ „Save Dataset“ et.

Innerhalb der Prozeduren werden indem entweder der Variablenname mit der Maus in das Funktionsfeld gezogen oder mit der Maus die Variable ausgewählt und mit dem Pfeil auf das Funktionsfeld geklickt. Dann wird mit dem Klick auf den Pfeil die Prozedur zum Laufen gebracht. Die Folgenden Abbildung zeigt die Ausgangsposition für den Reiter „Analysis“> „Crosstab“.

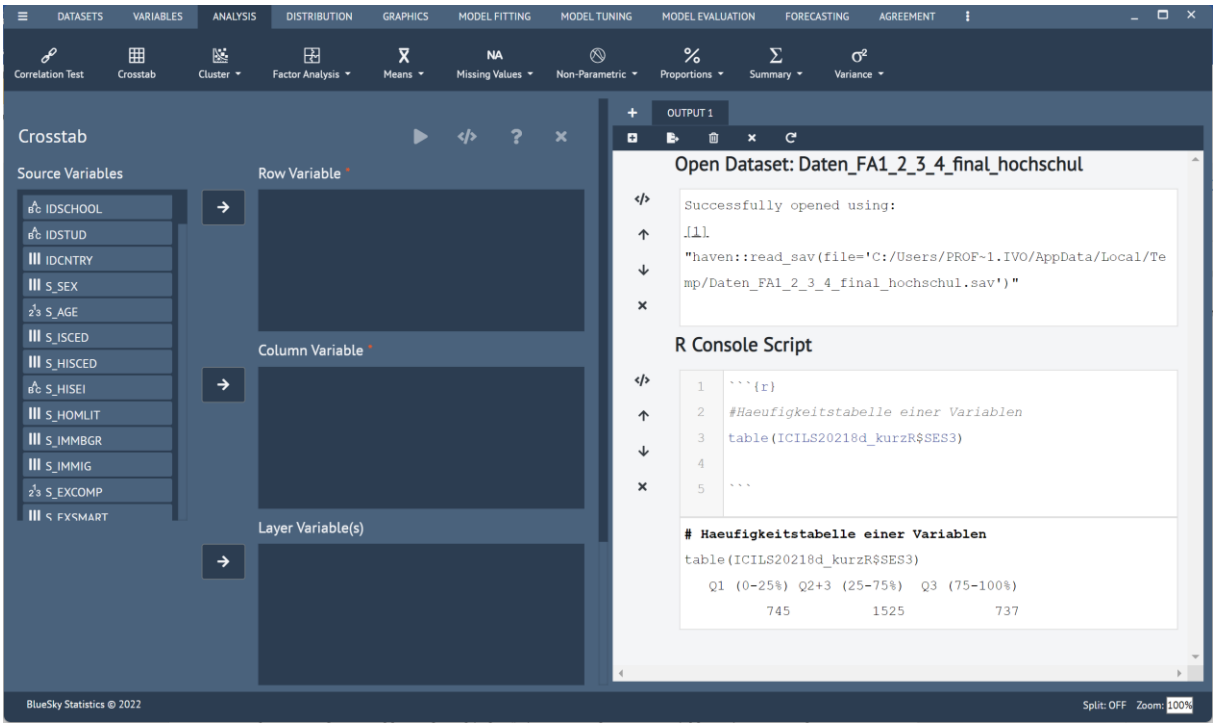


Abb. 10 Menüführung für die Erstellung von Kreuztabellen

Wenn mit dem Menüfenster gearbeitet wird, bleibt beim Wiederaufruf desselben Menüpunkts die vorherige Zusammenstellung im Dialogfenster (innerhalb einer Sitzung) **nicht** – wie bspw. in SPSS – erhalten, zur schnellen Reproduktion kann dann die Syntax aus dem Outputfenster genutzt werden. BSS berücksichtigt bei den Analysen das *Skalenniveau der Variablen* und unterbindet eine Verwendung von Variablen mit unpassendem Skalenniveau.

I.4 Syntax/Code – Steuerung von BSS über Syntax/Code

BlueSky Statistics selbst schreibt die Funktionen in verschiedenen „modernen“ R-Codes. Für die Datenverwaltung werden tidyverse-Pakete (vgl. hierzu https://riptutorial.com/de/r/example/25722/tidyverse) verwendet. Für Grafiken verwendet es ggplot2 und für die Modellabstimmung das Caret-Paket.

BSS arbeitet dabei sowohl mit dem Code der verwendeten Pakete, mit modernem R-Code sowie mit eigenen BSS Code. So gibt es Auswertungsroutinen im klassischem Code sowie in spezifischen BSS-Code. Die Aufbereitung und Darstellung erfolgt dann in speziellem BSS Coder und gerade über den modernen R Code. Dies verhilft zu den eleganten und funktionalen Tabellen und dem kopierbaren Output.

Entsprechend sind mit R oder RStudio (ohne die Installation von BSS) die reinen Analysen aus BSS dann anhand der Syntax ohne Voraussetzungen reprozierbar, wenn sie auf R-Cod oder modernem R-Code beruhen, für die spezifischen BSS-Auswertungsroutinen oder gerade die Formatierung des Outputs ist dann die Installation eines BSS Paket in R nötig (Das BlueSky Statistics R Package kann aus der Bibliothek der installierten Version von R in BSS kopiert und in die entsprechende „Iibrary“ des verwendeten R-Programms eingefügt werden).

Um dies zu veranschaulichen folgen einige Beispiele für den BlueSky R-Code: Um Mittelwerte zwischen Gruppen zu vergleichen nutzt BSS folgenden Code:

```
BSky_Dataset_Overview = data.frame(Dataset = c("uefagesamt"), Variables = length(names(uefagesamt)), Observations = nrow(uefagesamt), stringsAsFactors = TRUE)

BSky_Numerical_Statistics_Analysis = BSkySummaryStats(datasetColumnObjects = list(e01zielk = uefagesamt$e01zielk), groupByColumnObjects = list(subgr = uefagesamt$subgr), stats = c(min = FALSE, max = FALSE, mean = TRUE, median = FALSE, sum = FALSE, sd = TRUE, stderror = FALSE, iqr = FALSE, quantiles = FALSE), quantilesProbs = c(0, 0.25, 0.5, 0.75, 1), additionalStats = c(), datasetName = "uefagesamt")

BSkyFormat(BSky_Dataset_Overview, singleTableOutputHeader = c("Dataset Overview"))
```

(...)

Ein anderes Beispiel ist der BlueSky-Code für eine einfache lineare Regression. BlueSky liefert in der Anzeige der zugehörigen Syntax im Outputfenster sogar Kommentare, die jeden Schritt erklären und auch die Regressionsgleichung.

```
# Creating the model

LinearRegModel1 = lm(e01zielm ~ alter, na.action = na.exclude, data = uefagesamt)

# Display coefficients

reg_equation = equatiomatic::extract_eq(LinearRegModel1, use_coefs = TRUE, wrap = TRUE,
  ital_vars = FALSE, coef_digits = BSkyGetDecimalDigitSetting())

BSkyFormat(reg_equation)

# Summarizing the model

BSky_LM_Summary_LinearRegModel1 = summary(LinearRegModel1)

BSkyFormat(BSky_LM_Summary_LinearRegModel1, singleTableOutputHeader = "Model Summary")
```

An der Syntax wird deutlich, dass für die Schätzung der Regression die klassische Syntax verwendet wird, während der Befehl `BSky_LM_Summary_LinearRegModel1` für die Zusammenfassung der Ergebnisse oder `BSkyFormat()` für die Tabellenformatierung spezfische BSS Syntax darstellen (siehe ausführlich dazu Anhang 1).

I.5 Paketverwaltung

Ein Thema im Zusammenhang mit der Reproduzierbarkeit ist die *Paketverwaltung*. Einer der Hauptvorteile von R (und damit auch BSS) ist, dass die Funktionalität durch Add-On-Pakete immer mehr erweitert werden kann. In der Version 10 existiert jetzt auch die Möglichkeit, Pakete mit dem BSS-Menü zu installieren.

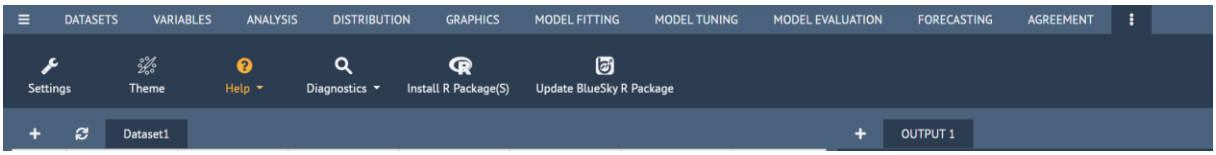


Abb. 11 Menüleiste – Installation von R Packages

Oben in der Leiste lassen sich ganz rechts bei den drei senkrecht untereinander stehenden Punkten die angezeigten Reiter aufrufen … Unter „Install R Packages“ können dann R Pakete zur Installation mit der Hand eingegeben werden.

Hilfe-Funktion: Jedes Dialogfeld verfügt über in der oberen rechten Ecke ein ?, mit der sich Hintergrundinfos aus den Paketen öffnen, die vor allem die Informationen der zugrundeliegenden Paketen bündeln (Kurzbeschreibungen, Syntaxbeispiele, Informationen über zugrundeliegende Pakete).

I.6 Berichte schreiben und Ausgabe exportieren

Die Analysen von BlueSky Statistics werden im Outputfenster immer mit Menütitel angezeigt, wie z.B. „Linear Regression“. Anmerkungen kann man im Output nicht machen, dies muss zuvor bei der Programmierung im R-Code im R-Syntax Editor geschehen.

Die Ausgabequalität von BlueSky Statistics ist höher als die von R (siehe Tab. 1).

Coefficients

	Estimate	Std. Error	t value	Pr(>|t|)	2.5 %	97.5 %
(Intercept)	3.902	0.469	8.327	2.406e-13 ***	2.973	4.830
KJH	0.439	0.159	2.768	0.007 **	0.125	0.753
Alter	0.001	0.016	0.079	0.937	-0.030	0.032

Note:
Signif. codes: 0 ' *** ' 0.001 ' ** ' 0.01 ' * ' 0.05 '.' 0.1 ' ' 1

Tab. 1 Tabelle in der Ausgabe in BlueSky Statistics

Die Formatierung der Tabellen im Outputfenster kann eingestellt werden (unter dem Reiter mit den drei Punkten, „Settings“ „Output Tabels“).

Der Output kann nun als Ganzes exportiert werden oder mit der Maus durch händisches Markieren, kopieren und einfügen in Software wie bspw. MS Word und Excel, was eine grundlegende Tabellenformatierung erhält und Nachformatierung möglich macht.

Bei der Exportfunktion wird die gesamte Ausgabe in einer einzigen Datei gespeichert, die dann im verknüpften Programm geöffnet werden kann (HTML).

II. Variablen- und Datenmanagement

BlueSky Statistics bietet eine Reihe von Optionen zum Datenmanagement im Menü an, unter den Reitern „Datasets“ und „Variables“ in der obersten Ebene. Sinnvoll ist es, die verschiedenen Optionen auszuprobieren und individuell den favorisierten Zugang zu finden. Über das Aufrufen der Befehle in das Syntaxfenster können diese gespeichert und immer wieder reproduziert und angepasst werden. (Anmerkung: Dem Autor erscheint die Arbeit mit entsprechenden Syntaxen oft einfacher, daher sind im Folgenden unter den Fenstern auch die entsprechenden BBS-Syntaxbefehle dargestellt.)

II.1 Bearbeiten des Datensatzes

Wichtige Befehle für das Datenmanagement bei Auswertungen sind:

- Aufteilen des Datensatzes

Über „Datasets“ > „Group by“ > „Split“ wird die Datei *in zwei oder mehr Gruppen aufgeteilt* (also beispielsweise nach Geschlecht), wenn verschiedene Gruppen (Stufen einer Variablen) getrennt analysieren werden sollen. Alle folgenden Analysen werden für jede Ebene der gewählten Faktoren durchgeführt.

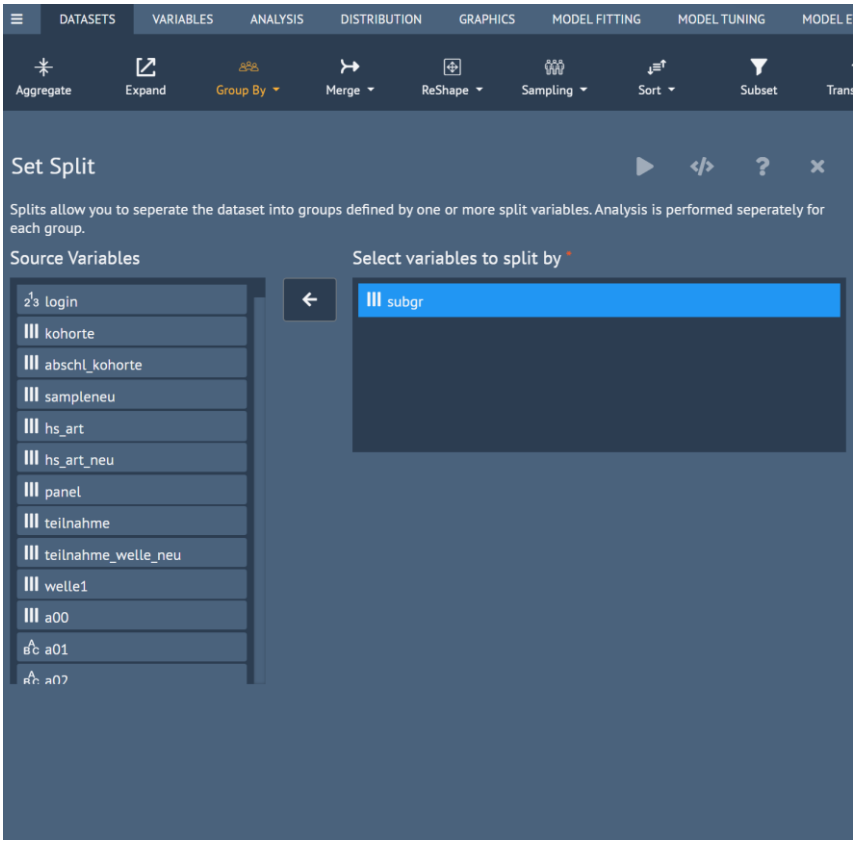


Abb. 12 Menüführung für das Aufteilen des Datensatzes

Die Syntax dafür ist:

```
BSkySetDataFrameSplit(c("subgr"), "dataset2")
```

Diese Funktion wird über "Datasets > Group by > Remove Split" deaktiviert.

- Auswahl einer Teilgruppe aus dem Datensatz

Wenn nur ein Teil des Datensatzes in Analysen verwendet werden soll, können über das Menü

mit „Data > Subset Dataset“ bestimmte Fälle ausgewählt werden (bspw. nur Personen einer bestimmten Altersgruppe, Personen aus ausgewählten Regionen oder Personen mit einem bestimmten Geschlecht. Es öffnet sich dann folgendes Fenster:

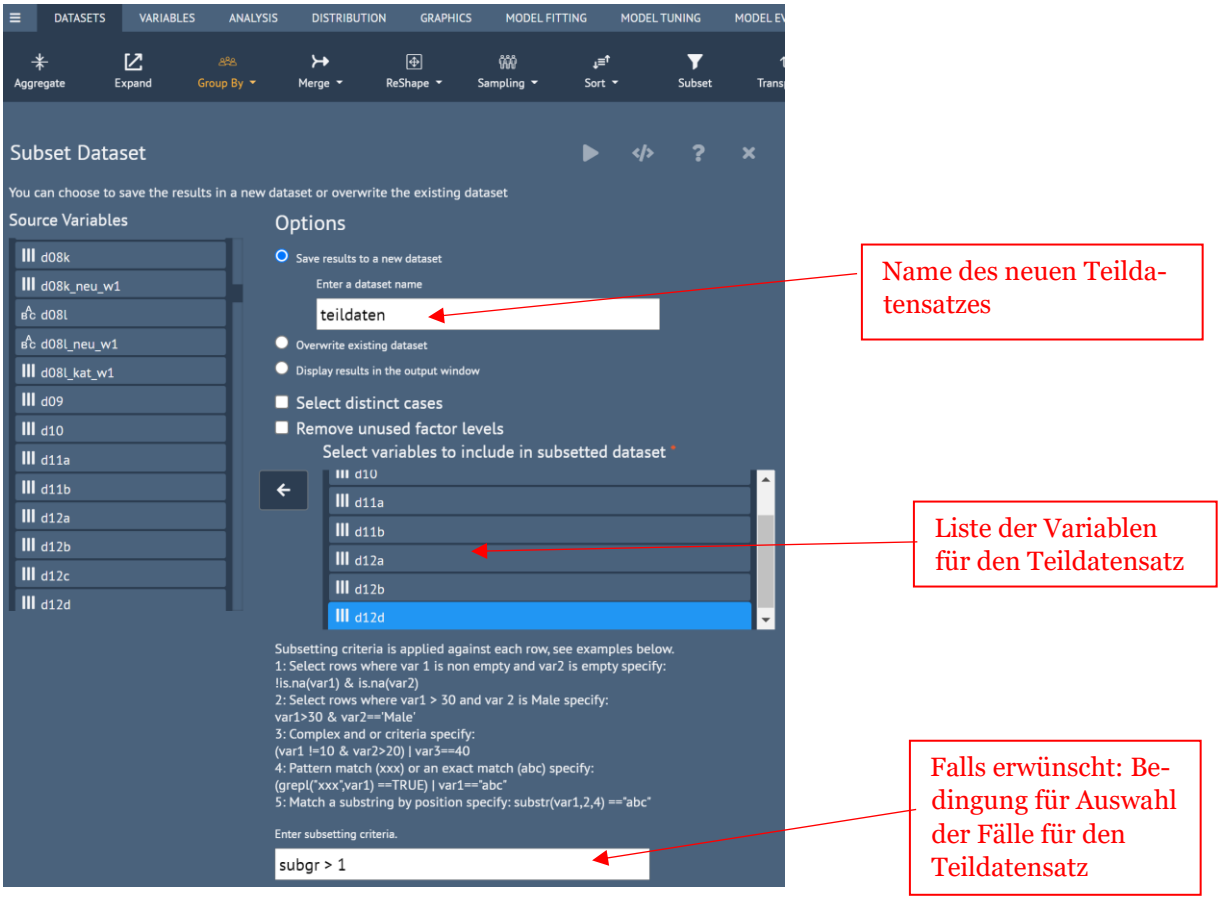


Abb. 13 Menüführung für die Bildung eines Teildatensatzes

Dabei können sowohl die Variablen, die im neuen Datensatz verwendet werden sollen (2), als auch die Bedingungen, die die Fälle erfüllen sollen (3), angegeben werden. Auch muss entschieden werden, ob der bisherige Datensatz überschrieben oder ein neuer Datensatz angelegt werden soll (im letzteren Fall Dabei technisch in BSS ein neuer Teildatensatz erstellt, der auch im Datenfenster als neuer Reiter angezeigt wird).

Die Syntax dafür:

```
#Creates the subsetted dataset
teildaten <- uefagesamt %>%
   dplyr::filter(alter > 18) %>%
    dplyr::select(d09, d10, d11a, d11b, d12a, d12b, d12d)
#Refreshes the subsetted dataset in the data grid
BSkyLoadRefresh('teildaten')
```

II. 2 Variablen bearbeiten

- Neue Variablen berechnen

Über „Variables“ > „Compute“ kommt man zu einem Menü, dass es erlaubt, auf unterschiedliche Weise neue variablen zu erstellen oder bestehende zu verändern. So kann über „Variables > Compute > Apply a function across all rows” auch eine Variable durch Berechnung mit vorgegebener Formel durchgeführt werden. Das folgende Beispiel erstellt eine Summenindex aus drei Variablen:

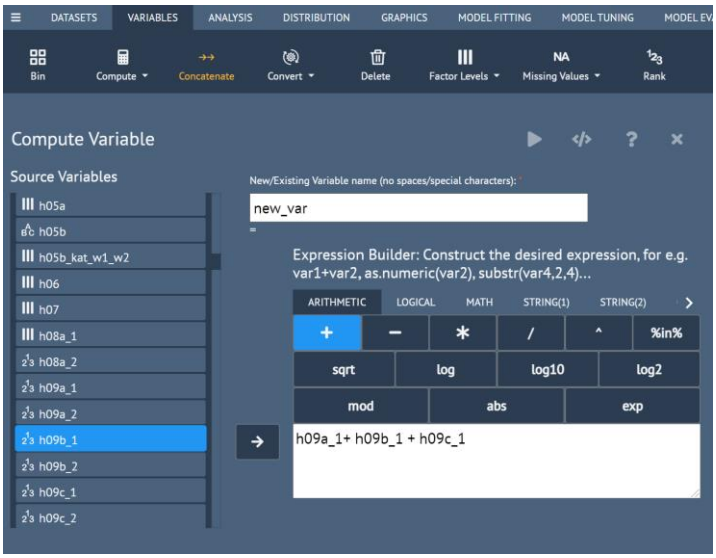


Abb. 14 Menüführung zur Berechnung neuer Variablen

```
require(dplyr)
# Computes the new/existing column
local({
    tryCatch({
        .GlobalEnv$uefagesamt <- uefagesamt %>%
            mutate(new_var = h09a_1 + h09b_1 + h09c_1)
    }, error = function(e) {
        cat(conditionMessage(e))
    }) })
BSkyLoadRefresh("uefagesamt")
```

In einem anderen Beispiel wird eine Skala aus dem Mittelwert dreier Variablen per Funktion erstellt:

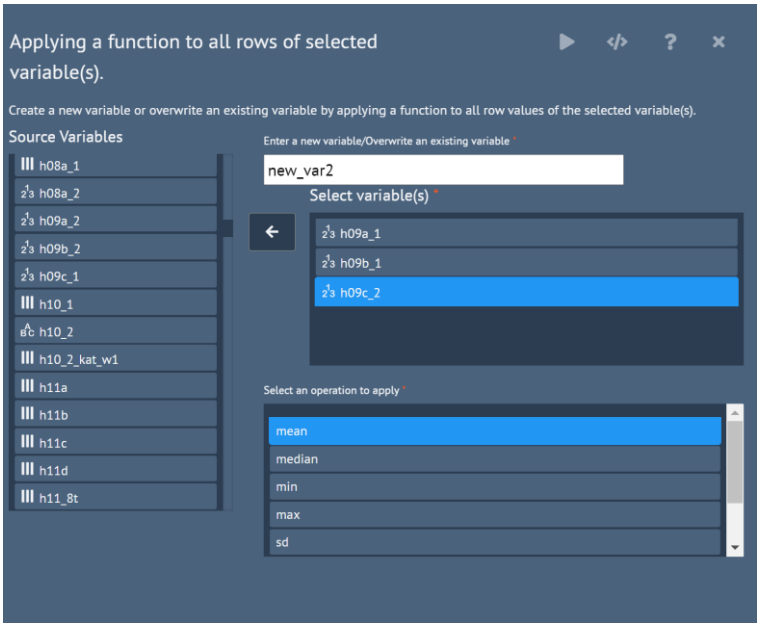


Abb. 15 Unterfenster zur Berechnung neuer Variablen über Funktionen

Die Syntax ist:

```
# Apply function to all rows
uefagesamt$new_var2 <- uefagesamt %>%
    select(h09a_1, h09b_1, h09c_1) %>%
    apply(1, mean, na.rm = TRUE)
# Refresh the dataset in the grid
BSkyLoadRefresh("uefagesamt")
```

- *Erstellung Dummy Variablen*

Für Erstellung von Dummyvariablen(0/1-Variablen) aus gestuften Variablen gibt es eine bei BSS eine eigene Routine „Data > Compute Dummy Variables“.

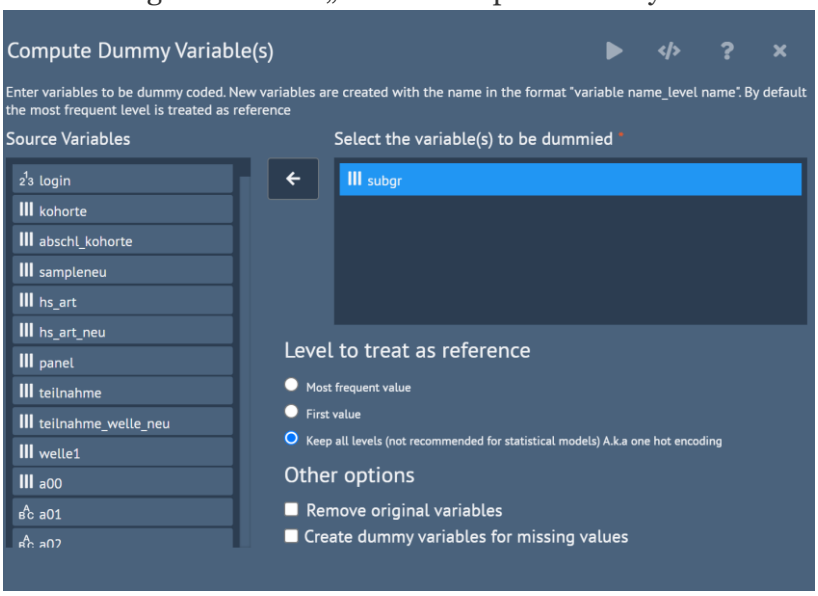


Abb. 16 Menüleiste – Installation von R Packages

```
# Dummy coding variables
uefagesamt <- fastDummies::dummy_cols(.data = uefagesamt, select_columns = c("subgr"),
    remove_selected_columns = FALSE, ignore_na = TRUE)
names(uefagesamt) <- stringr::str_replace_all(names(uefagesamt), pattern = " ", replacement = "_")
BSkyLoadRefresh(bskyDatasetName = "uefagesamt", load.dataframe = TRUE)
```

- Rekodieren einer Variablen

Dies wird im Menüfenster aufgerufen über „Variables > Recode“. Häufig ist es sinnvoll/notwendig, Variablen noch einmal umzucodieren und zum Beispiel Stufen zusammenzufassen oder bei Items für Skalen die Ausprägungen von bspw. 1-5 umzudrehen/zu „invertieren“.

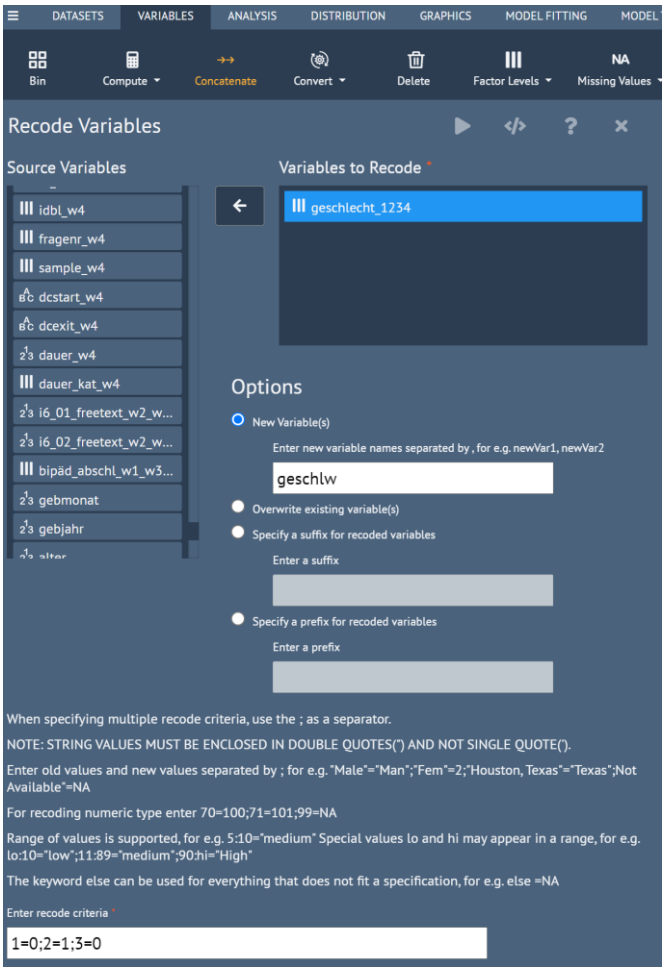


Abb. 17 Menüführung für die Rekodierung von Variablen

Die Syntax dazu ist:

```
# Perform the recode
BSkyRecode(colNames = c("geschlecht_1234"), newColNames = c("geschlw"), Old-
NewVals = "1=0;2=1;3=0",
    NewCol = TRUE, dontMakeFactor = FALSE, dataSetNameOrIndex = "uefagesamt")
# Refresh the dataset in the data grid
BSkyLoadRefresh("uefagesamt")
```

- *Nummerische Variable in Stufen aufteilen*

Hierfür hat BSS die sehr nützliche Funktion „Variables > Bin“, mit der über die Menüführung eine metrische Variable in eine gestufte rekodiert werden können, im entsprechenden Fenster müssen nur die „Grenzwerte“ für die Stufen eingegeben werden.

- *Entfernen fehlender Werte*

Fehlenden Werte sind bei einigen Analysen in R ein Problem. Diese können ausgeschlossen werden, wenn mit der Menüführung unter „Variables > Missing Values > Remove NA“ eine neuer Teildatensatz angelegt wird, der für die für die aktuellen Analyse relevanten Variablen nur gültige Fälle hat und dieser dann zur Grafikerstellung genutzt wird.

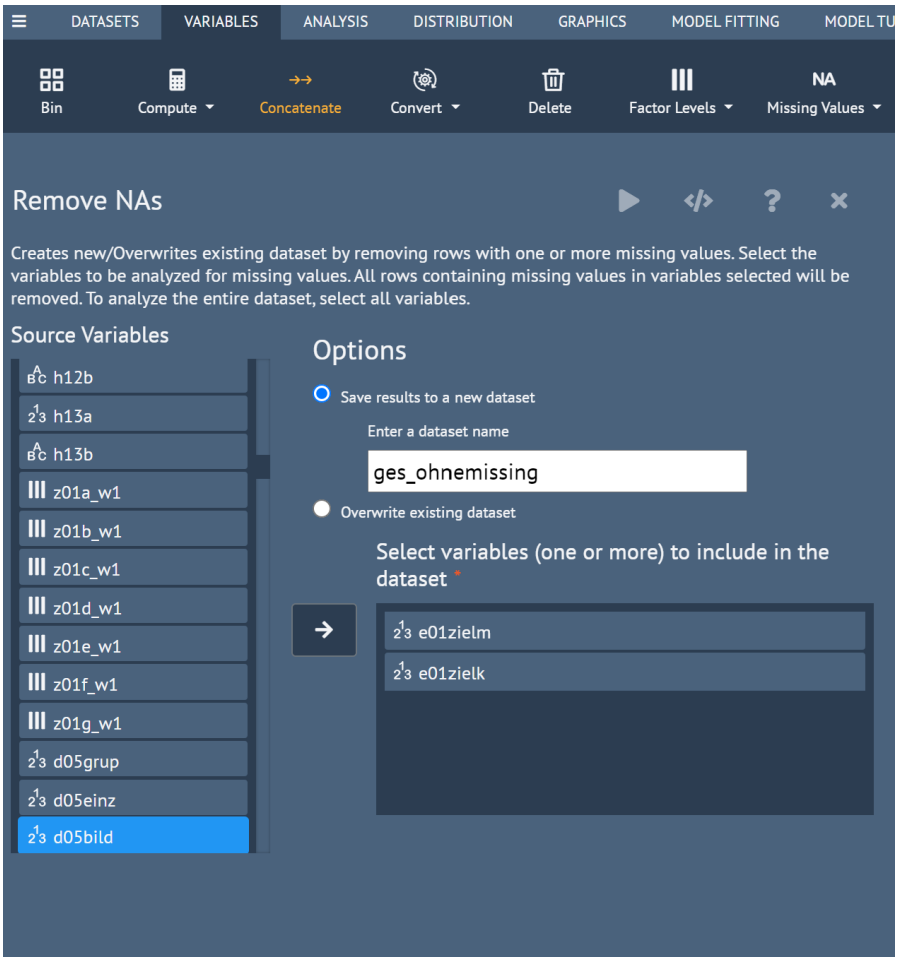


Abb. 18 Menüführung zur Entfernung von „missings“ in einem neuen (Teil)Datensatz

Als Syntax:

```
ges_ohnemissing <- na.omit(uefagesamt[c("e01zielm", "e01zielk")])

# Refreshes the dataset in the data grid
BSkyLoadRefresh("ges_ohnemissing")
```

III. Aufrufen von Datenauswertungen und -analysen

Viele Datendarstellungs- und Auswertungsroutinen sind – analog zu SPSS mit der Maus „klickbar“. Allerdings ist es wie generell in R bedeutsam, den Variablen vorab das richtige Skalenniveau zuzuweisen – da BSS *einerseits* viele beschreibende Auswertungsroutinen automatisch adäquat wählt und *andererseits* verhindert, dass Auswertungsroutinen mit Variablen mit nicht passendem Skalenniveau durchgeführt werden.

Für viele Auswertungsroutinen gibt es über das Menü mehrere Wege, zu einer entsprechender Auswertungstabelle zu kommen (s.u.), die Wahl hängt eher von den gewünschten Informationen und der Darstellungsart ab (und ist auch Geschmackssache). Viele der Tabellen lassen sich bei der Erstellung auch modifizieren (zusätzliche Angaben, Anzeige oder Weglassen fehlender Werte).

III.1 Einfache Häufigkeitstabellen (für ordinale oder kategoriale Variablen)

Über „Analysis > Summary > Frequencies“ und dann das Klicken der gewünschten Variablen in das entsprechende Fenster erzeugt eine zu SPSS vergleichbare einfache univariate Häufigkeitstabelle (ohne Anpassungsmöglichkeiten) (vgl, Abb. 19).

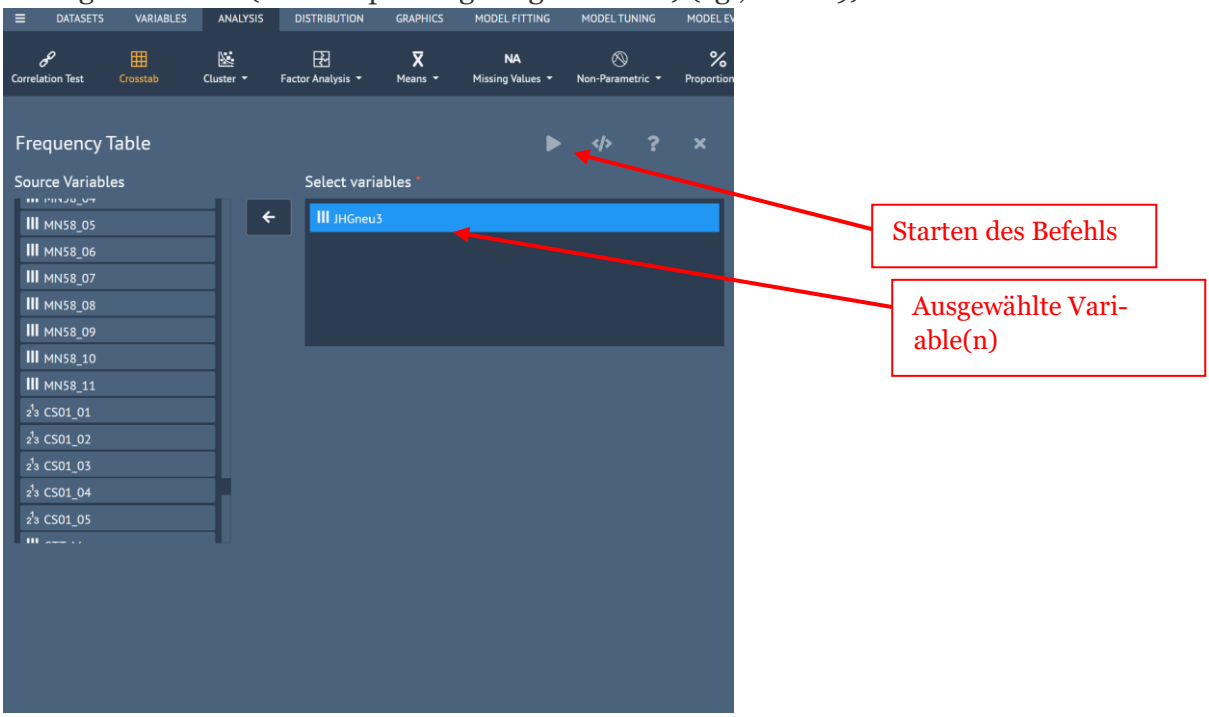
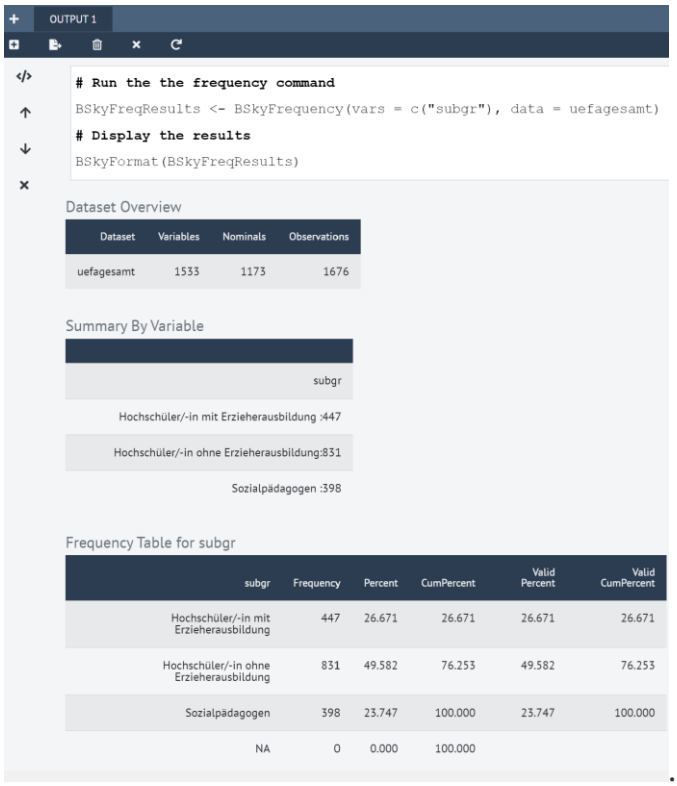


Abb. 19 Menüführung für Häufigkeitstabellen

Mit dem Klick auf den Pfeil wird die folgende Ausgabe im Outputfenster angezeigt



Tab. 2 Ausgabe Häufigkeitsauszählung

Die zugehörige Syntax wird im Outputfenster mit angezeigt.

```
# Run the frequency command
BSkyFreqResults <- BSkyFrequency(vars = c("subgr"), data = uefagesamt)
# Display the results
BSkyFormat(BSkyFreqResults)
```

III. 2 Kreuztabelle mit zwei Variablen

Über „Analysis > Crosstabs" kommt man zum Menü für Kreuztabellen, in dem dann eine Auswahl von zwei oder mehr Variablen Kreuztabellen in Analogie zu bspw. SPSS-Ausgaben erstellt werden (vgl. Abb. 12).

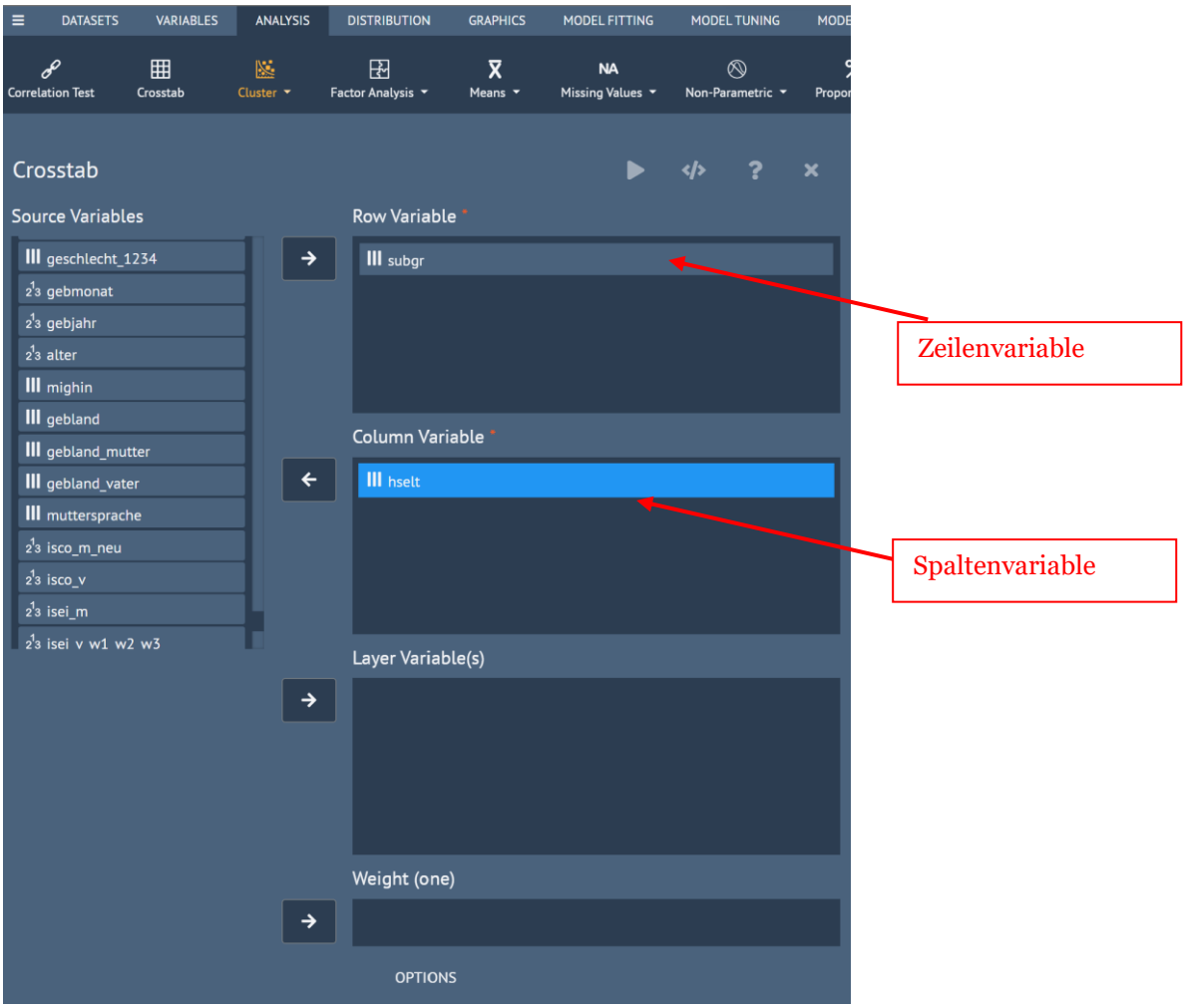


Abb. 20 Menüführung für die Erstellung von Kreuztabellen

Über das Untermenü „Options“ können Prozentwerte und auch der Chi-Quadrat Test angefordert werden.

Abb. 21 Unterfenster Optionen zur Anforderung von Auswertungsstatistiken bei Kreuztabellen

Nach Befehlseingabe erscheint im Outputfenster folgende Tabelle:

Multiway Cross Table: ~ subgr + hselt

subgr	count	hselt		
		Akademisch gebildetes Elternteil	Kein akdamisch gebildetes Elternteil	Total
Hochschüler/-in mit Erzieherausbildung	Count	95	331	426
	% within hselt	19.231	32.261	28.026
Hochschüler/-in ohne Erzieherausbildung	Count	284	486	770
	% within hselt	57.490	47.368	50.658
Sozialpädagogen	Count	115	209	324
	% within hselt	23.279	20.370	21.316
Total	Count	494	1026	1520
	% within hselt	100	100	100

Statistical Tests

Tests	Value	df	Asyp. Sig	Odds ratio	95% Confidence interval	
Pearson Chi Square	28.268	2	0.000	X	X	X

Note:
X indicates incomplete result due to the data not meeting the requirements for the requested test, warnings, or errors

Tab. 3 Ausgabe Kreuztabelle

Die Signifikanzangabe des Chi²-tests findet sich unter „Asyp. Sig".

Die zugehörige Syntax ist:

```
#Create the crosstab
BSky_Multiway_Cross_Tab = BSkyCrossTable(
    x=c('subgr'), y=c('hselt'),
    datasetname='uefagesamt',
    chisq=TRUE, mcnemar=FALSE, fisher=FALSE,
    prop.r=FALSE, prop.c=TRUE,
    resid=FALSE, sresid=FALSE, asresid=FALSE,
    expected=FALSE)
#Display the crosstab in the output grid
BSkyFormat(BSky_Multiway_Cross_Tab)
```

III.3 Mittelwert und Standardabweichung einer metrisch skalierten Variablen

Übersichtliche tabellarische Darstellungen zu Mittelwerten und Standardabweichungen sind über „Analysis > Summary Analysis > Numerical Statistical Analysis" bzw. „Numerical Statistical Analysis, using describe" oder „Numerical Statistical Analysis using summarize" anwählbar. Nachfolgend ein Beispiel anhand der Variante "describe" (vgl. Abb. 22).

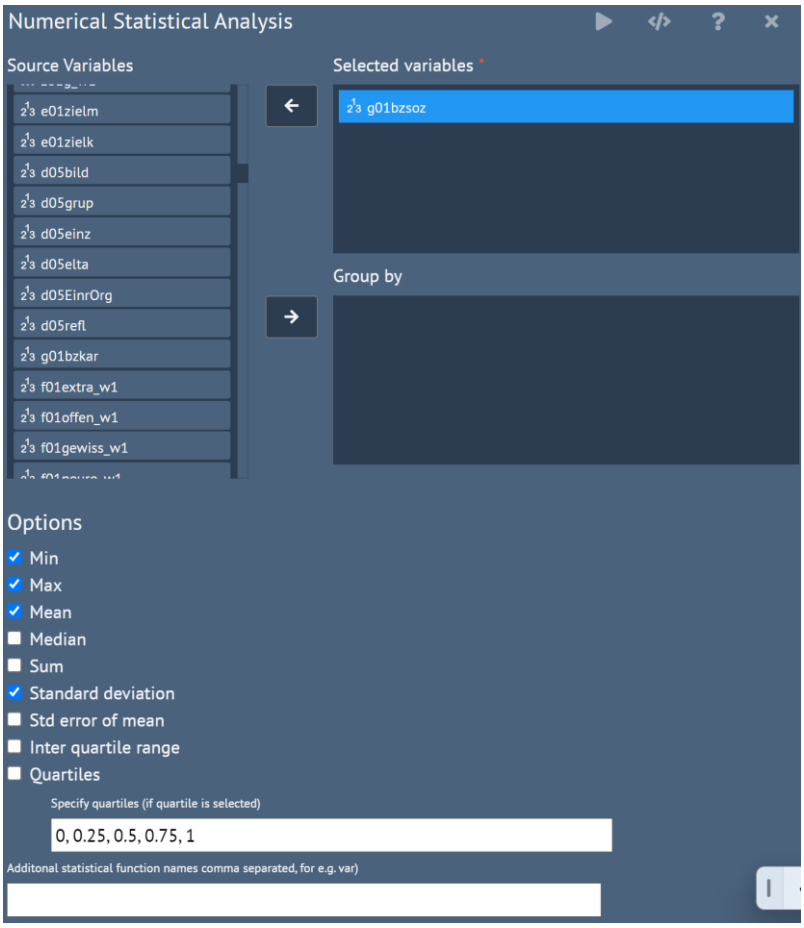


Abb. 22 Erstellung Mittelwertsausgabe

Im Fenster „group by“ kann für einen Mittelwertsvergleich auch eine Gruppierungsvariable angegeben werden (vgl. III.4). Nach Klick auf den Pfeil findet sich die Ausgabe im Outputfenster (vgl. den Auszug in Tab. 4).

Dataset Overview

Dataset	Variables	Observations
uefagesamt	1	1676

Numerical Statistical Analysis by Variable

stats	g01bzsoz
min	1.667
mean	4.056
max	5.000
sd	0.691
n	1345.000
NAs	331.000

Tab. 4 Ausgabe Mittelwerte

```
BSky_Dataset_Overview = data.frame(Dataset = c("uefagesamt"), Variables =
length(names(uefagesamt)),
    Observations = nrow(uefagesamt), stringsAsFactors = TRUE)
BSky_Numerical_Statistics_Analysis = BSkySummaryStats(datasetColumnObjects
= list(g01bzsoz = uefagesamt$g01bzsoz),
    groupByColumnObjects = list(), stats = c(min = TRUE, max = TRUE, mean =
TRUE,
        median = FALSE, sum = FALSE, sd = TRUE, stderror = FALSE, iqr =
FALSE, quantiles = FALSE),
    quantilesProbs = c(0, 0.25, 0.5, 0.75, 1), additionalStats = c(),
datasetName = "uefagesamt")
BSkyFormat(BSky_Dataset_Overview, singleTableOutputHeader = c("Dataset
Overview"))
```

III.4 Mittelwerte verschiedener Ausprägungen einer Variablen

Hier kann die eben angewendete Routine oder eine ähnliche unter *„Analysis* > Summary > Nummerical Summeries, Using Summarizes“, oder „Nummerical Summeries, Using Describe“ angefordert werden. Es muss dabei für den Gruppenvergleich eine Group Variable hinzugefügt werden (vgl. Abb. 23).

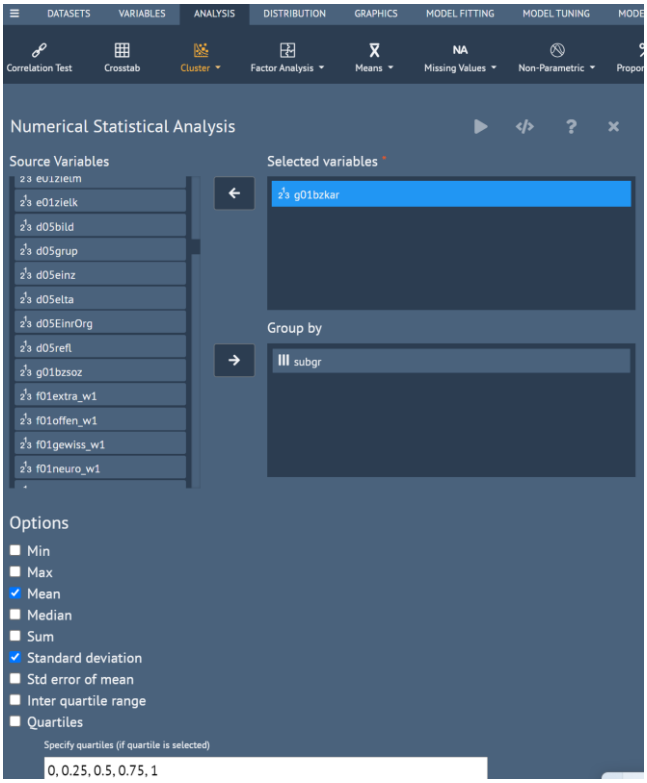


Abb. 23 Erstellung Mittelwerte nach Gruppen

Die Ausgabe gibt pro Ausprägung die angeforderten Mittelwerte etc. aus.

Numerical Statistical Analysis by Variable

subgr	stats	g01bzkar
Hochschüler/-in mit Erzieherausbildung	mean	4.131
	sd	0.597
	n	340.000
	NAs	107.000
Hochschüler/-in ohne Erzieherausbildung	mean	3.924
	sd	0.642
	n	681.000
	NAs	150.000
Sozialpädagogen	mean	3.686
	sd	0.672
	n	325.000
	NAs	73.000

Tab. 5 Ausgabe Mittelwerte nach Gruppen

Die zugehörige Syntax:

```
BSky_Dataset_Overview = data.frame(Dataset = c("uefagesamt"), Variables = length(names(uefagesamt)), Observations = nrow(uefagesamt), stringsAsFactors = TRUE)

BSky_Numerical_Statistics_Analysis = BSkySummaryStats(datasetColumnObjects = list(g01bzkar = uefagesamt$g01bzkar),
    groupByColumnObjects = list(subgr = uefagesamt$subgr), stats = c(min = FALSE, max = FALSE, mean = TRUE, median = FALSE, sum = FALSE, sd = TRUE, stderror = FALSE, iqr = FALSE, quantiles = FALSE), quantilesProbs = c(0, 0.25, 0.5, 0.75, 1),    additionalStats = c(), datasetName = "uefagesamt")
```

Unter „Analysis > Means“ stehen für *testende Mittelwertsvergleiche* eine Reihe von Varianzanalysen und T-Tests zur Verfügung, z.B. auch die SPSS-analoge Option „Analysis > Means > ANOVA, one Way and two Way“ (siehe Abschnitt V).

Generell sind in den Auswertungsroutinen von BlueSky Statistics keine Optionen für die Berechnung von Effektstärkemaßen für Kontingenztabellen oder Mittelwertsvergleiche implementiert, die aber, wenn gewünscht, mit zusätzlichen R Paketen über die R Syntax angefordert werden können. Anhang 2 zeigt hierfür exemplarische Möglichkeiten.

III.5 Korrelationen

Über „Analysis > Correlation test“ kommt man zu einem Menü für Korrelation. Hier können zwei oder mehr numerische Variablen in das Zielfenster geklickt werden.

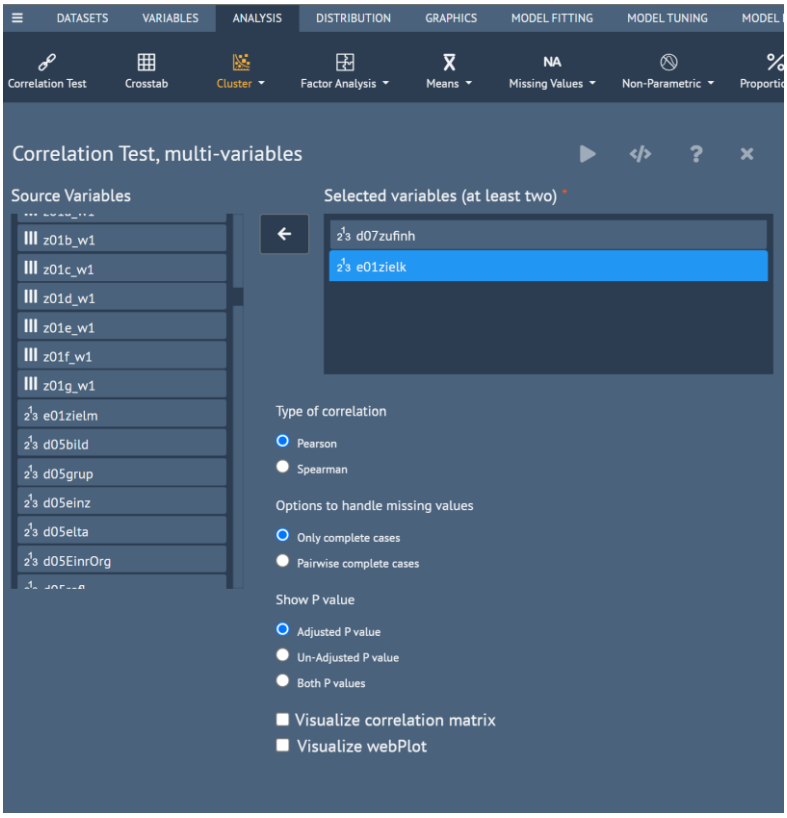


Abb. 24 Menüführung Erstellung Korrelationstabelle

Der Output kann nach Wahl leicht angepasst werden, in folgender Tabelle finden sich die wichtigsten Angaben.

Pearson correlation

		d07zufinh	e01zielk
d07zufinh	Correlation	1.000	0.028
	Adj-P		0.304
	n	1348.000	1348.000
e01zielk	Correlation	0.028	1.000
	Adj-P	0.304	
	n	1348.000	1348.000

Tab. 6 Ausgabe Korrelationstabelle

Die Syntax hierzu ist

```
BSkyResults <- BSkyCorrelationMatrix(data = uefagesamt, vars = c("d07zu-
finh", "e01zielk"),
    correlationType = "Pearson", missingValues = "complete.obs", pValue =
"adjP")
BSkyFormat(BSkyResults)
```

III.6 Aufruf weiterer statistischer Auswertungsverfahren

Über die Menüführung können eine Vielzahl statistischer Auswertungsverfahren aufgerufen werden – vor allem über die Hauptpunkte „Analysis“ und „Model Fitting“. Im Folgenden sollen nur de Zugangswege kurz veranschaulicht werden, die Durchführung ausgewählter Methoden wird ausführlich in Abschnitt V beschrieben.

a) Varianzanalysen/T-Test (siehe auch ausführlich Teil V)

Über „Analysis > Means“ gelangt man/frau zu einem ausführlichen Menüfenster für u.a. verschiedene T-Tests und Varianzanalysen. Abschnitt VI enthält hierfür Durchführungsbeispiele.

b) Regressionsmodelle (siehe ausführlich Teil V)

Unter „Modell Fitting“ steht ein Bereich mit vielen Optionen und unterschiedlichen Modellen zur Verfügung. Zum Beispiel ist der Weg für eine *Lineare Regression* (Einflussfaktoren auf eine metrisch skalierte Variable testen z.B. Geschlecht, Lesehäufigkeit und Elternbildung auf Lesekompetenz).

- „Model Fitting > Linear Regression“.

Dabei ist das Fenster zur Modellspezifikation bei allen Regressionen ähnlich (vgl. Abb. 10):

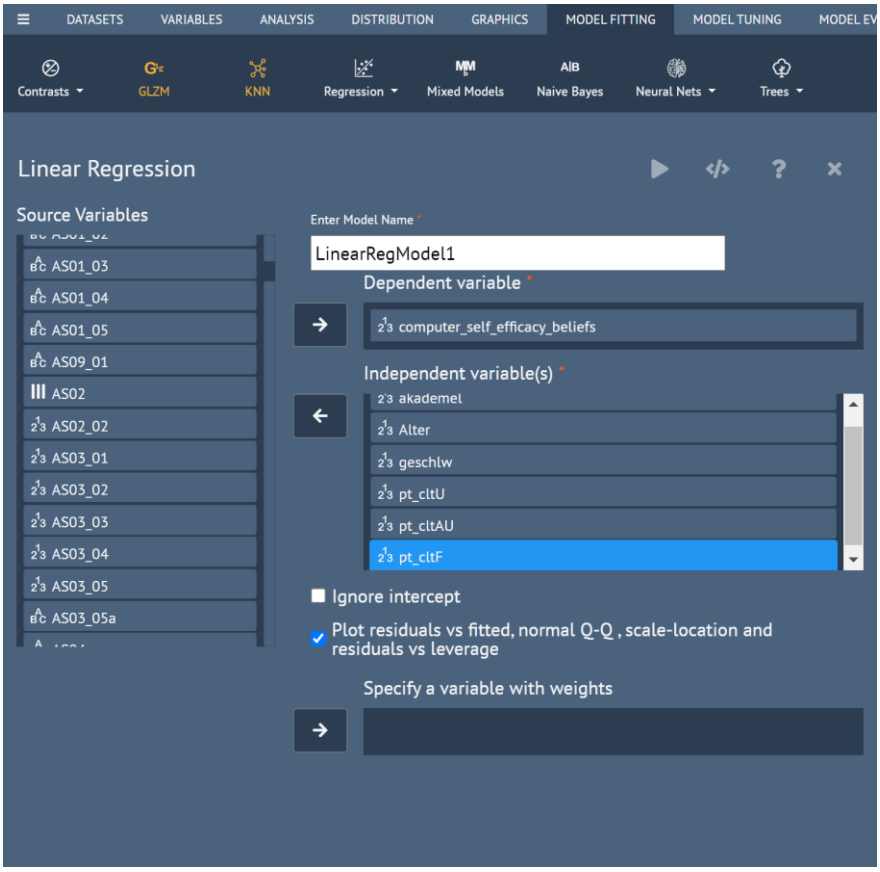


Abb. 25 Menüfenster zur Spezifizierung einer linearen Regression

Es lassen sich über das kleine Klickfenster auch Grafiken zu Inspektion der Voraussetzung der linearen Regression anfordern.

Analog werden weitere Regressionsmodelle mit

- „Model Fitting“ > „Logistic Regression“,
- „Model Fitting” > “Multinomial Logit“,
- „Model Fitting“ > Ordinal Regression“

Mehrebenenmodelle können über

- „Model Fitting > Mixed Models“ (hier können bislang nur lineare Modelle geschätzt werden)

angefordert (vgl. für die Umsetzung ausführlich Abschnitt V).

Der zugehörige R-Code zu allen Analysen kann wie immer angezeigt und so auch als eigene Syntax kopiert und gespeichert werden.

Bedeutsam ist, dass BSS die R-Logik der „Modellobjekte“ übernimmt – das heißt, jedes geschätzte Modell wird im Arbeitsspeicher als Objekt mit einem (veränderbaren) Namen wie „LinearRegModel1“ gespeichert, kann entsprechend wieder aufgerufen werden, und über die Funktionen „Model Tuning“ „Model Statistics“ können Modellanpassungen vorgenommen oder weitergehende Informationen über das jeweilige Modell (wie zum Beispiel Gütekriterien, Pseudo-R etc.) abgerufen werden. Oben rechts im Hauptmenüfenster können die in der Session erzeugten Modell aufgerufen werden (Drop-Down Fenster) und wieder analysiert/deren Informationen abgerufen werden. (Alle dann gewählten Auswahloptionen aus dem Menüfenster Model Statistics wie AIC (Modell Information) oder weitere Modellanalysen (wie „stepwise variable selection“) werden jeweils für das angezeigte Modell durchgeführt.

c) Faktoranalysen

- Unter „Analysis > Factor Analysis“ stehen explorative Faktoren- und Hauptkomponentenanalyse (siehe Abschnitt V.4).
- *Konfirmatorische* Faktorenanalysen sind bislang noch nicht in BSS nicht integriert. Hier kann bspw. über die Installation des R-Paketes lavaan und mit der entsprechende R Syntax die Funktionalität erweitert werden – dies setzt dann aber Syntaxarbeit voraus (vgl. Abschnitt VI.2).

d) Clusteranalysen/latente Klassenanalysen

- Clusteranalysen sind unter „Analysis > Cluster Analysis“ aufrufbar. Zur Verfügung stehen aus dem Basis-Paket die hierarchische Clusteranalyse oder die Clusteranalyse mit K-means-Verfahren. Die Funktionalität ist hier insgesamt beschränkt, sinnvoll ist die Heranziehung weiterer R-Pakete (vgl. Abschnitt V.5).
- Latente Klassenanalysen (LCA) oder auch latente Profilanalysen (LPA) lassen sich mit BSS nicht durchführen – hier kann bspw. auf das R Paket poLCA für die LCA oder das Paket *mclust* (vereinfacht noch in Verbindung mit dem Paket *tidyverse*) für die LPA verwiesen werden, die aber wiederum ausschließlich über die R-Syntax verwendet werden können (vgl. Abschnitt VI.3).

Teil 2: Erstellung von Grafiken und Durchführung von statistischen Analysen

IV. Grafiken

Für Grafiken greift BlueSky Statistics vor allem auf das ggplot2-Paket zurück, hier wird der Original Code verwendet (BlueSky bietet aber auch Grafikfunktionen aus dem R-Grundmodul bei dem Menüunterpunkt "Legacy“ an). Über das Menüfenster „Graphics“ öffnet sich eine Liste von möglichen Grafiken.

Hinweis: Fehlenden Werte („NA“) werden bei der Erstellung von Grafiken immer als eine eigene Kategorie ausgegeben. Diese kann verhindert werden, wenn zuvor mit der Menüführung unter „Data > Missing Values > Remove NA“ eine neuer Teildatensatz angelegt wurde, der für die betreffenden Variablen nur gültige Fälle hat und dieser dann zur Grafikerstellung genutzt wird (s. Abschnitt II).

IV.1 Erstellen von Balkendiagrammen

Über „Graphics > bar chart ” können u.a. Balkendiagramme von Mittelwerten erstellt werden.

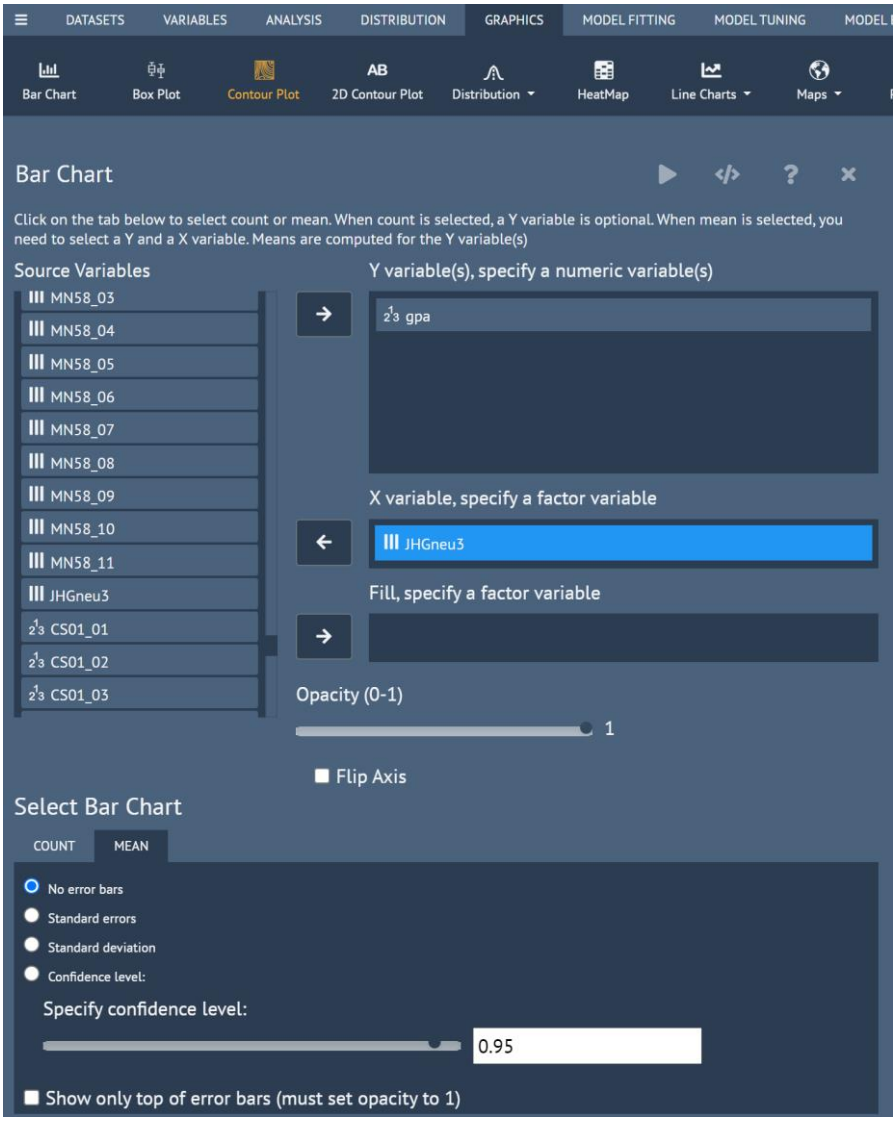


Abb. 26 Menüführung zur Erstellung von Balkendiagrammen

Mit der Option „Flip Axis“ kann das Säulen- zu einem Balkendiagramm gedreht werden.

Unter dem Untermenü „Options“ kann die Beschriftung angepasst werden

Im Ergebnis zeigt sich folgende Grafik:

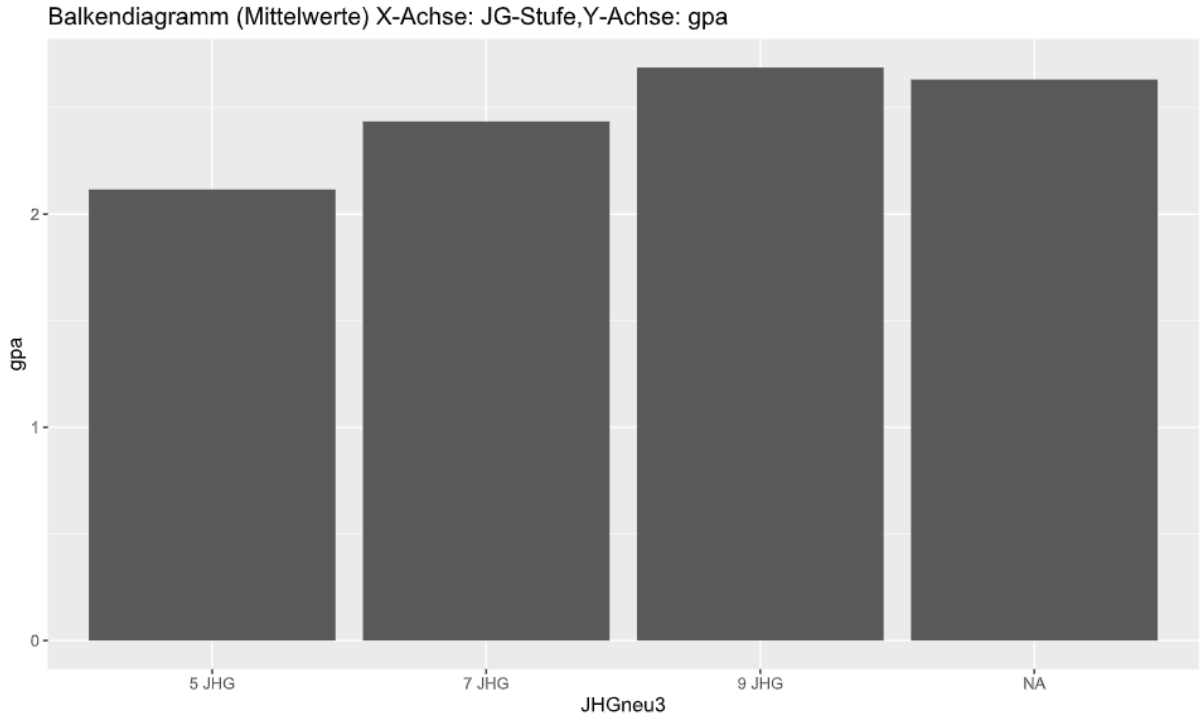


Abb. 21 Einfaches Balkendiagramm

Die Syntax hierfür lautet:

[Bar Chart with means)]

```
## [Bar Chart (with means)]
require(ggplot2);
require(Rmisc)
temp <- Rmisc::summarySE(schuel2021, measurevar = "gpa", groupvars = c("JHGneu3"),
    na.rm = TRUE, .drop = TRUE)
pd <- position_dodge(0.9)
ggplot(data = temp, aes(x = JHGneu3, y = gpa)) + geom_bar(position = "dodge", alpha = 1,
    stat = "identity") + labs(x = "JHGneu3", y = "gpa", title = "Balkendiagramm (Mittelwerte) X-Achse: JG-Stufe,Y-Achse: gpa") +
    xlab("JHGneu3") + ylab("gpa") + theme_grey() + theme(text = element_text(family = "sans",
    face = "plain", color = "#000000", size = 12, hjust = 0.5, vjust = 0.5))
```

Über die Menüführung gibt es ein paar weitere Veränderungsmöglichkeiten im ersten Unterfenster, für größere Veränderungen muss die Syntax im Outputfenster geöffnet werden – dann können bspw. noch die Schriftgröße, einzelne Farben etc. verändert werden.

Beispiel für Stapelbalken

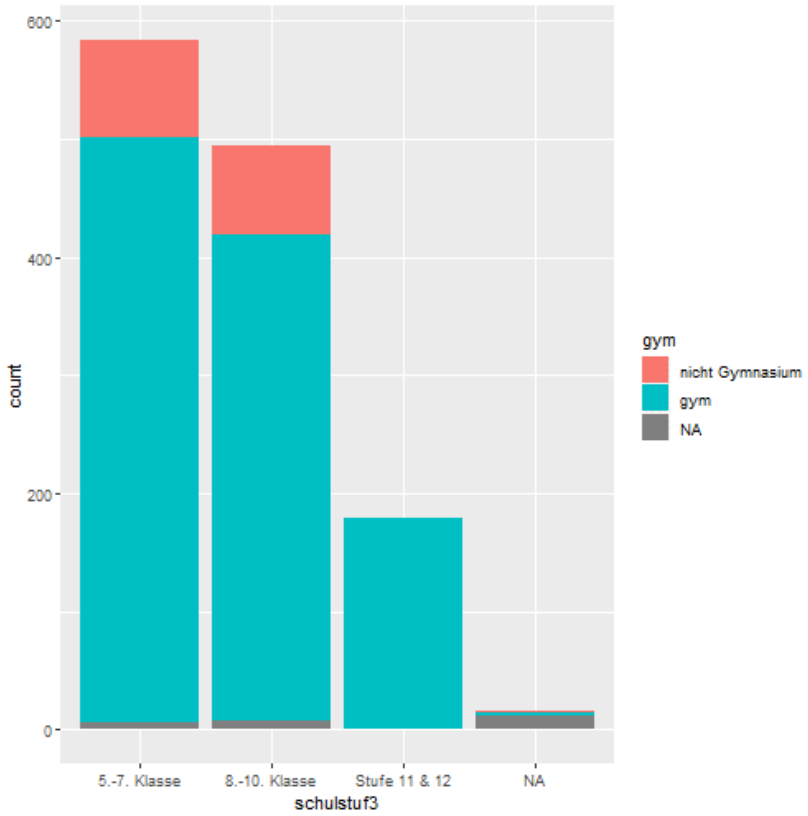


Abb. 22: Gestapeltes Balkendiagramm

```
require(ggplot2)
ggplot(data = homes3108, aes(x = schulstuf3, fill = gym)) + geom_bar ()
```

IV.2 Erstellen von Diagrammen mit Boxplots

Über die Menüführung „Graphics > boxplots" kommt man analog zur Erstellung von Boxplots, mit entsprechendem Unterfenster für die Beschriftung.

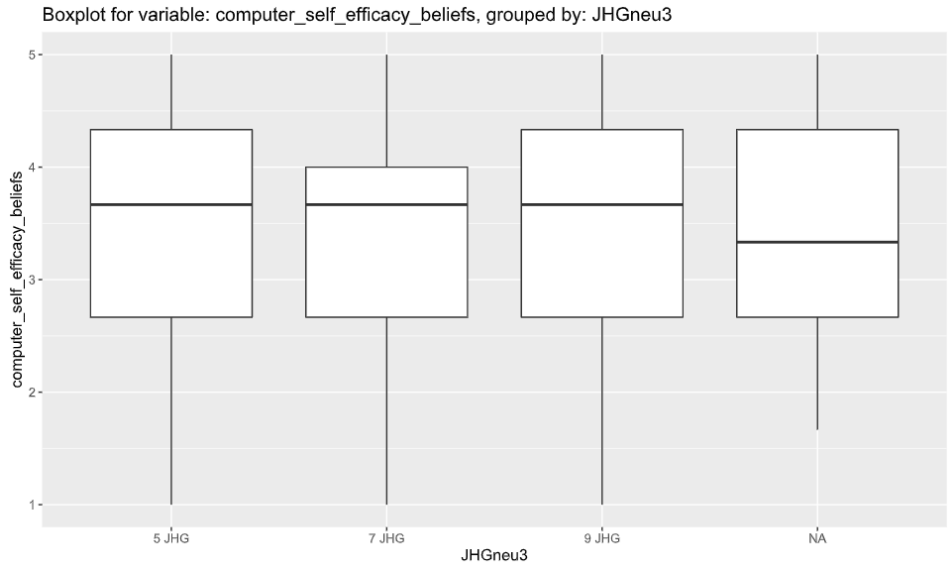


Abb. 23 Diagramm mit Boxplots

Die entsprechende Syntax lautet

```
## [BoxPlot]
require(ggplot2);
require(ggthemes);

ggplot(data = schuel2021, aes(x = JHGneu3, y = computer_self_efficacy_beliefs)) +
    geom_boxplot(alpha = 1, ) + labs(x = "JHGneu3", y = "computer_self_efficacy_beliefs",
    title = "Boxplot for variable: computer_self_efficacy_beliefs, grouped by: JHGneu3") +
    xlab("JHGneu3") + ylab("computer_self_efficacy_beliefs") + theme_grey() + theme(text = element_text(family = "sans",
    face = "plain", color = "#000000", size = 12, hjust = 0.5, vjust = 0.5))
```

IV.3 Erstellen von Histogrammen

Unter „Graphics“ > „Distribution “ > „Histogramm“ kann analog ein einfaches Histogramm erstellt werden.

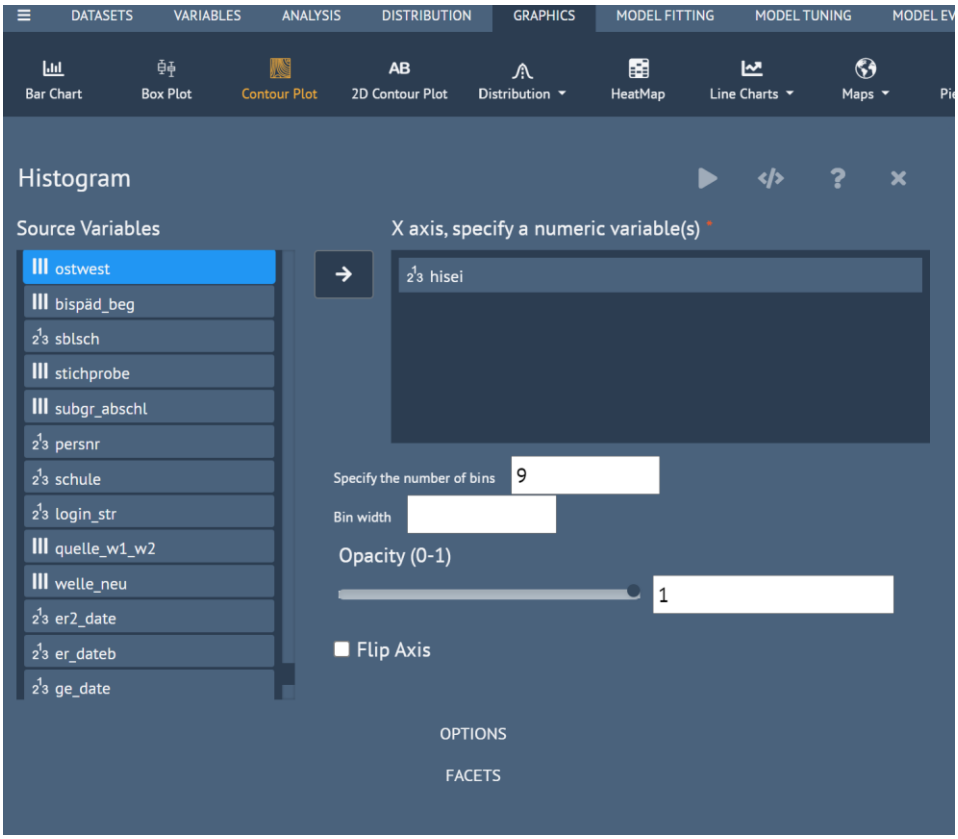


Abb. 27 Menüführung zur Erstellung eines Histogramms

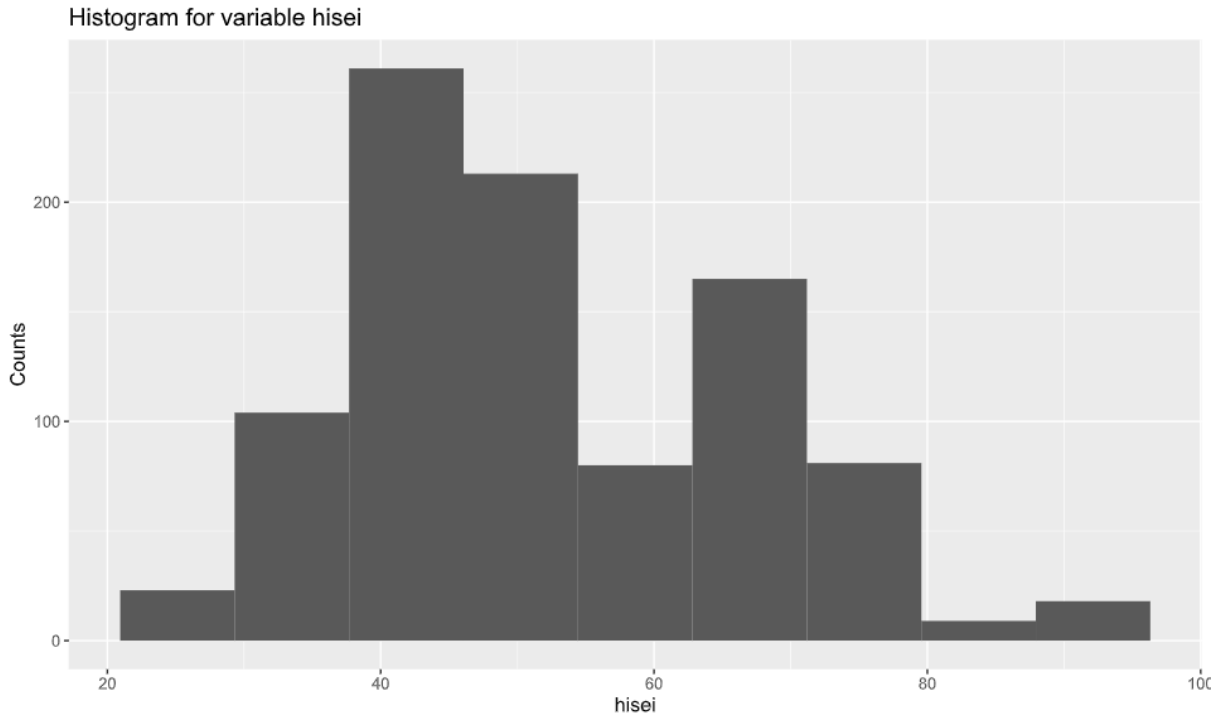


Abb. 28 Histogramm

Für das Histogramm kann die Zahl der Balken oder auch die Breite der Abstände frei gewählt werden

Über die Syntax wird dies über folgenden Weg erreicht:

```
require(ggplot2);
require(ggthemes);

ggplot(data=datensatzkurz,aes(x =gpa_noten)) +
       geom_histogram(alpha=0.5) +
       labs(x ="bewle3fach",y ="Counts",title= "Histogram for variable
bewle3fach") + theme_grey() + theme(text=element_text(fam-
ily="sans",face="plain",color="#000000",size=12,hjust=0.5,vjust=0.5))
```

Histogramme eignen sich auch zur Prüfung von Normalverteilungsannahmen, ergänzend können hierfür auch Q-Q-Plots erstellt werden (siehe Abschnitt V).

IV.4 Erstellen von Mehrfachdiagrammen

Bei BSS besteht die Möglichkeit, eine Abbildung mit mehreren Grafiken zu erstellen – mit der Option „facet“ (nebeneinander, untereinander oder auch als beidem („wrap“).

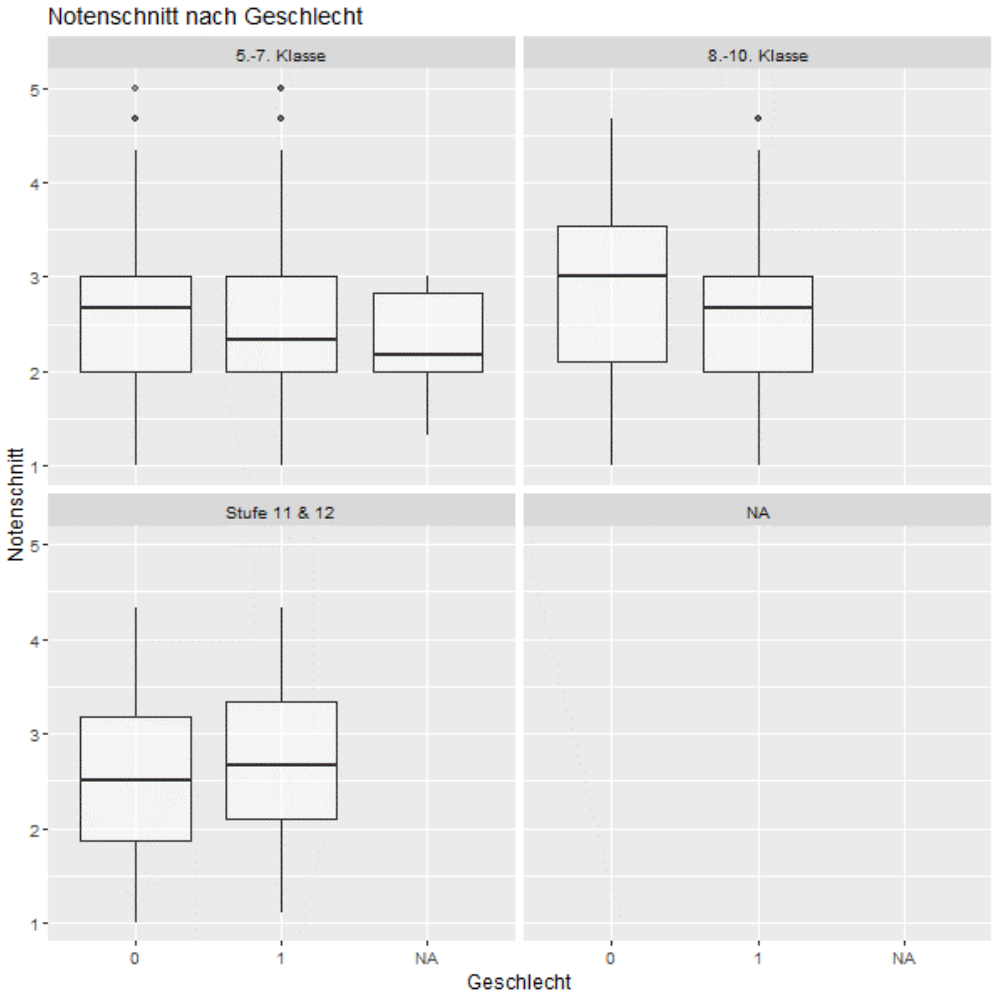


Abb. 29 Mehrfachgrafik mit Boxplots

Hier wurde die NA-Kategorie nicht zuvor entfernt. Die Syntax lautet:

```
## [BoxPlot]
require(ggplot2);
require(ggthemes);

ggplot(data=homes0208,aes(x =geschlw,y =
gpa_noten)) +
 geom_boxplot(alpha =0.5) +
 labs(x ="geschlw",y ="gpa_noten",title=
"Boxplot for variable gpa_noten,group by
geschlw") +
 xlab("Geschlecht") +
 ylab("Notenschnitt") +
 ggtitle("Notenschnitt nach Geschlecht")+
 facet_wrap(~schulstuf3) +
theme_grey() + theme(text=element_text(family="sans",face="plain",
color="#000000",size=12,hjust=0.5,vjust=0.5))}
} )
```

Die Grafiken können über die R Syntax nach Wunsch weiter angepasst werden (Farben, Schriftarten und -größen, Skalierungen). Weitere Hinweise dazu finden sich in der Hilfe zum Paket ggplots2 oder bspw. unter https://r-intro.tadaa-data.de/book/visualisierung.html.

V. Durchführung von statistischen Analysen mit BSS

Die Durchführung statistischer Analysen ist schon angeklungen, in diesem Abschnitt sollten exemplarisch statistische Analysen vorgestellt werden, die mit dem Menüführung aufgerufen werden können. Dabei steht die Anwendung des Programms im Mittelpunkt, für eine Erklärung der Verfahren sollte auf entsprechende Lehrbücher (wie bspw. Backhaus et al. 2013, Bortz Döring 200) zurückgegriffen werden. Ausführlich dargestellt werden im Folgenden die Durchführung von 1) T-tests und Varianzanalysen unabhängiger Stichproben, 2) einer linearen Regression, 3) einer binär logistischen Regression sowie 4) einer Faktorenanalyse – mit dem Plan, dies zukünftig weiter auszuweiten.

Die Attraktivität der Nutzung von BSS für entsprechende Analysen liegt für den Autor im Zusammenspiel von den in BlueSky Statistics implementierten Methoden und Routinen und den zugehörigen leicht exportierbaren und weiterverarbeitbaren Ausgaben sowie der Möglichkeit, die Analysen über weitere R Pakete und die R Syntax ergänzen zu können (wird im Folgenden auch dargestellt). Aber auch wenn die Analysen nur über den klassischen R-Code aufgerufen werden, sieht der Autor den erleichterten Datenimport, Datenzugang und das einfache Datenmanagement von BSS als eine Vereinfachung des Arbeitsprozesses an.

V.1 Durchführung T-test/Varianzanalyse unabhängiger Stichproben

V.1.1 Durchführung T-Test zweier unabhängiger Stichproben

Für den Vergleich von Mittelwerten zweier Gruppen kann bei Vorliegen entsprechender Voraussetzungen der T-Test eingesetzt werden. Zwei Voraussetzungen sollten dazu geprüft werden

a) Normalverteilung der abhängigen metrischen Variablen
b) Varianzhomogenität zwischen den beiden Gruppen

Bei der Verwendung des Menüs von BlueSky Statistics sollte die Normalverteilung vorab geprüft werden, die Prüfung der Varianzhomogenität zwischen den Gruppen wird in einem Prozedere mit dem T-test getestet.

Schritt 1: Prüfung der Normalverteilung
Dieses als Test kann über den Shapiro-Wilk-Test oder den Kolmogorov-Smirnov-Test erfolgen, diese Tests sind allerdings sehr „empfindlich“ und werden bei großen Stichproben durch die Stichprobengröße schnell signifikant. Empfehlenswert ist daher die grafische Inspektion der Residuen über Histogramm und Q-Q-Plot. Der Aufruf des Histogramms wurde eben beschrieben, ein Q-Q-Plot wird über „Graphics > Distribution > Q-Q-Plot“ aufgerufen (analog ist auch der Aufruf eines P-P-Plots möglich).

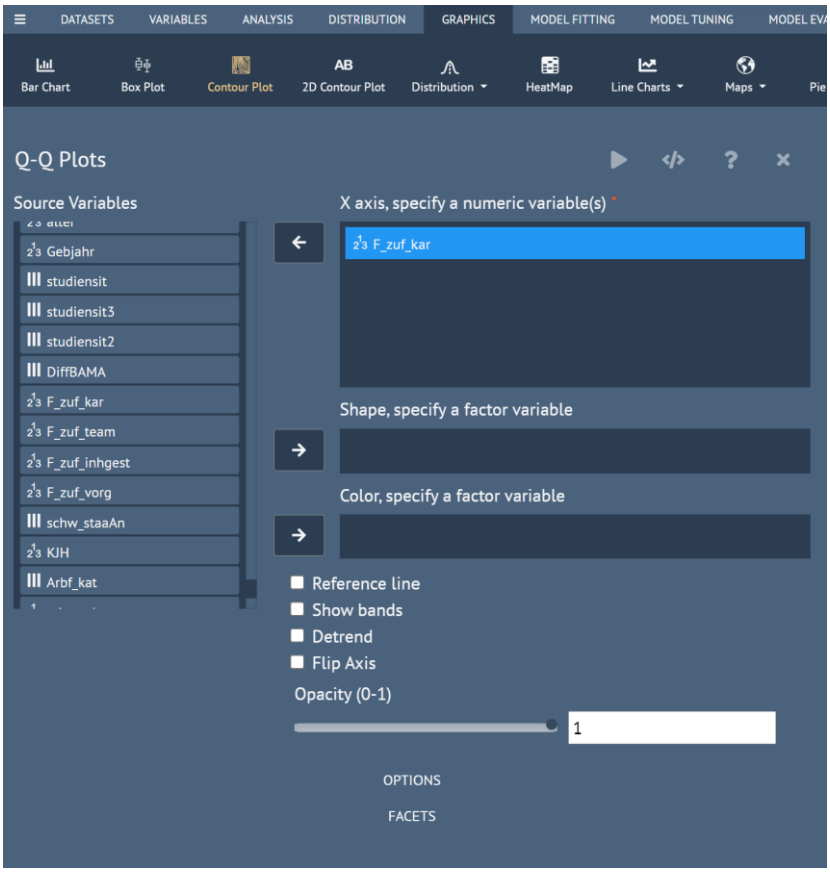


Abb. 30 Menüfenster zur Erstellung eines Q-Q-Plot zur Inspektion einer Normalverteilung

Im Ergebnis zeigt sich die Grafik– für eine Normalverteilung sollte sich die Verteilung möglichst gut an die gedachte Grade „anschmiegen“.

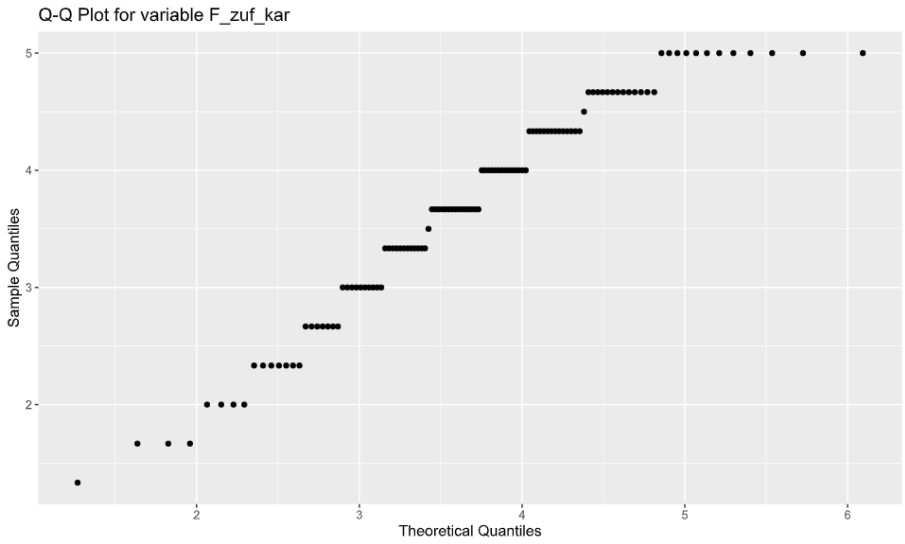


Abb. 31 Q-Q-Plot zur Inspektion einer Normalverteilung

Schritt 2: Voraussetzung aller Varianzanalysen ist die Homogenität der Varianzen in den Teilgruppen. Der Levene-test auf Varianzgleichheit in den zwei Gruppen und der eigentliche T-Test wird in einem Schritt aufgerufen über die Menüleiste mit „Analysis > Means > T-test, independent sample“).

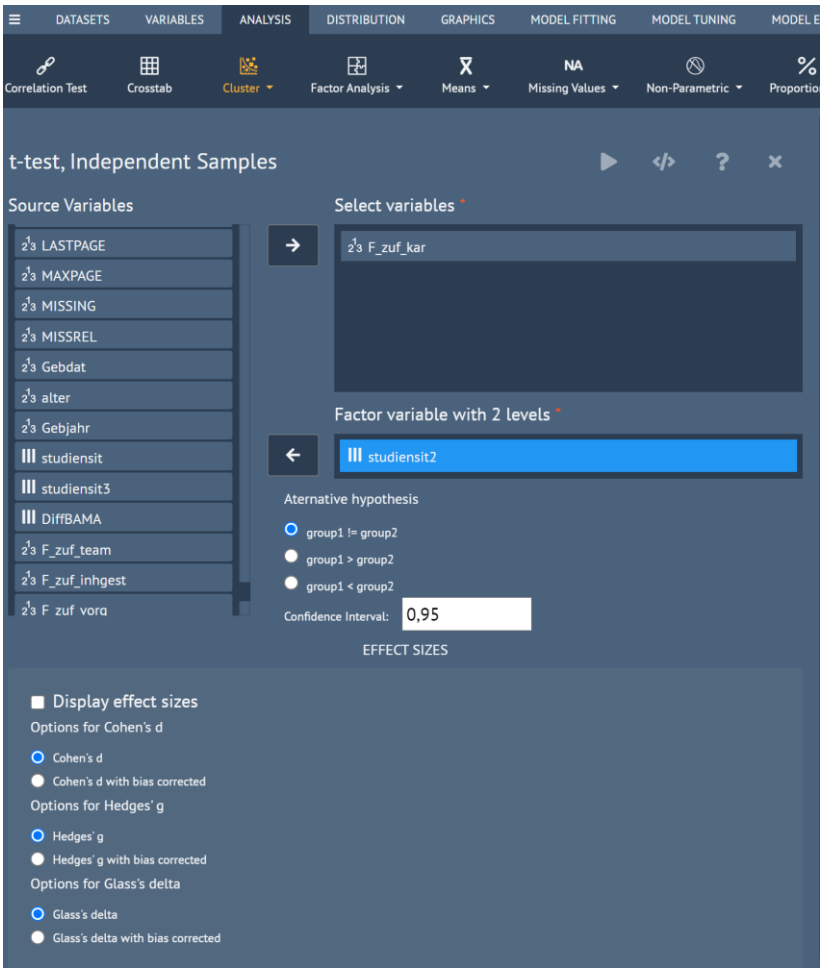


Abb. 32 Menüführung zur Durchführung eines T-Tests

Hier wird die abhängige Variablen und die Gruppierungsvariable angeben. Unter „Alternative hypotheses“ kann die Richtung des zu testenden Unterschieds vorgegeben werden. Unter Options ist es jetzt bspw. auch möglich, Cohens d anzufordern.

Als Ausgabe zeigt sich zunächst folgende deskriptive Vergleichsübersicht:

Group Statistics

	studiensit2	N	Mean	Std Deviation	Std Error Mean
F_zuf_kar	(bislang) nur BA abgeschlossen	64	3.622	0.937	0.117
	BA & MA abgeschlossen	50	3.757	0.912	0.129

Tab. 7 Deskriptive Tabelle als Ausgabe zum T-Test

Independent Samples t-test

		Levene's Test for Equality		t-test Equality of Means						
									confidence interval of the diffs: 0.95	confidence interval of the diffs: 0.95
		F	Sig.	t	df	Sig.(2-tail)	mean difference	std. error difference	lower	upper
F_zuf_kar	Equal variance assumed	0.625	0.431	-0.768	112.000	0.444	-0.134	-0.012	-0.481	0.212
	Equal variances not assumed	NA	NA	-0.771	106.728	0.443	-0.134	NA	-0.480	0.211

Tab. 8 Ausgaben zum T-Test

Die Tabelle 8 ist der SPSS-Darstellung von T- Test-Ergebnissen nachempfunden: Ist der Levene-Test nicht signifikant, kann der oberen Ergebniszeile gefolgt werden. Die untere Ergebniszeile gibt das Ergebnis des *Welch-Tests* wieder und wird bei signifikantem Unterschied der Varianzen im Levene-Test relevant.

```
# Run the T-test
BSky_Independent_Sample_T_Test    =    BSkyIndSmTTest(varNamesOrVarGlobalIn-
dices=c('F_zuf_kar'),
    group=c('studiensit2'),conf.level=0.95,
    alternative='two.sided', missing =0,
    datasetNameOrDatasetGlobalIndex='Absol21')
# Display the results
BSkyFormat(BSky_Independent_Sample_T_Test)
#remove(BSky_Independent_Sample_T_Test)
```

V.1.2 Durchführung Einfaktorielle Varianzanalyse/ANOVA

Die Durchführung einer einfaktoriellen oder zweifaktoriellen Varianzanalyse kann mit der Menüleiste über „Analysis > Means > ANOVA, 1 and 2 Way“ erfolgen (vgl. Abb. 32) (es sind aber auch mehrere weitere ANOVA-Optionen verfügbar).

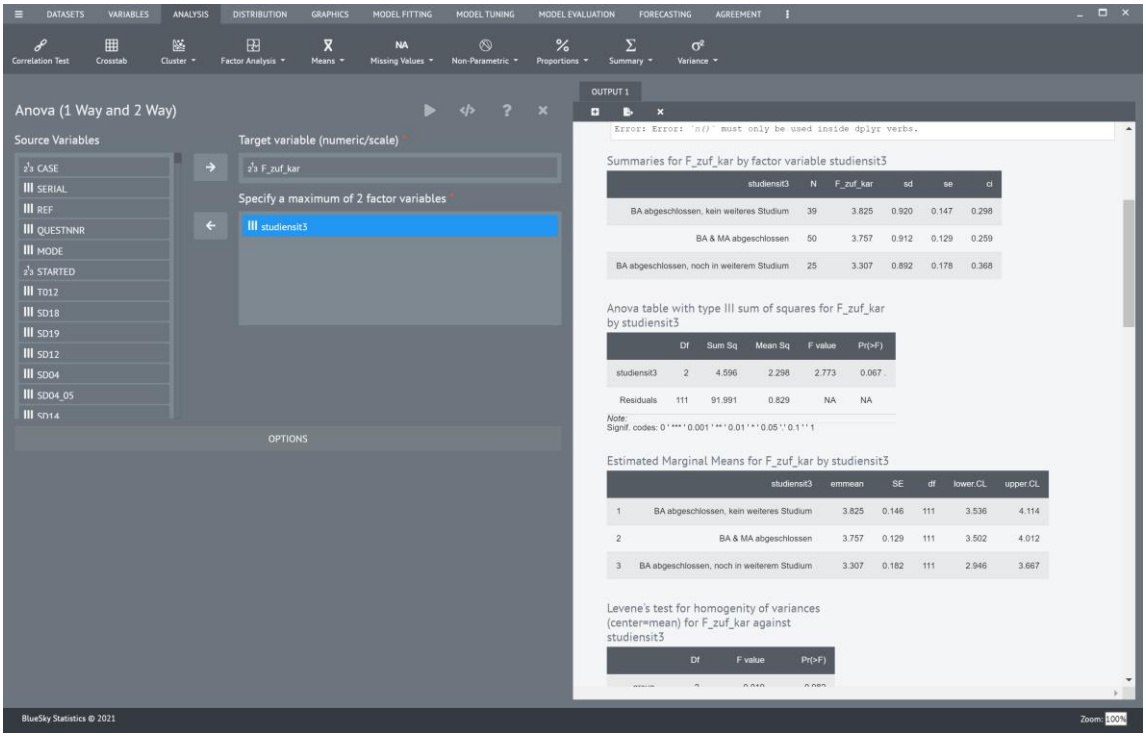


Abb. 33 Menüfenster für eine ein- oder zweifaktorielle Varianzanalyse

Unter „Options“ kann der notwendige Levene-Test für Varianzgleichheit angefordert werden sowie auch Post Hoc Tests für Vergleiche der Untergruppen sowie Diagnosegrafiken.

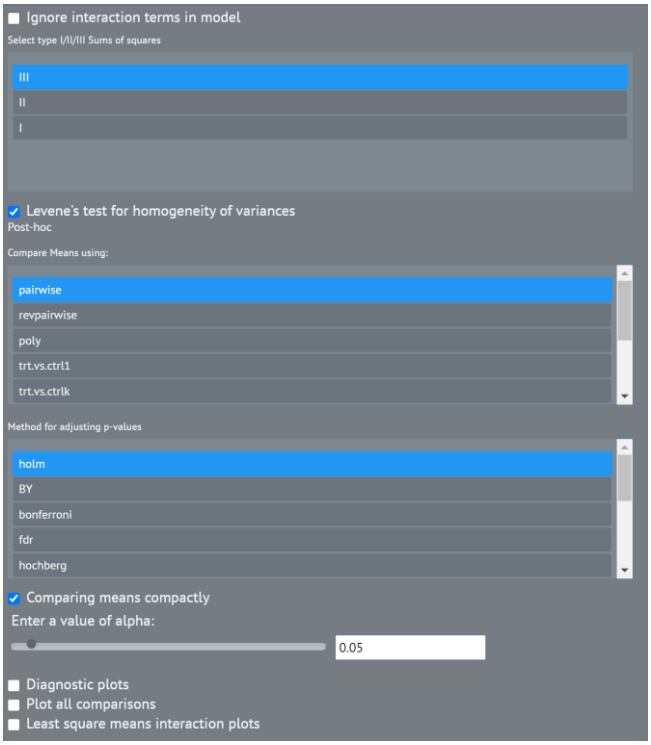


Abb. 34 Untermenü zur Anforderung u.a. des Levene-Tests auf Varianzgleichheit

Im Output wird zunächst eine Tabelle mit den deskriptiven Statistiken abgebildet, gefolgt von einer Tabelle mit erklärter und nichterklärter Varianz sowie dem Ergebnis des Signifikanztests.

Summaries for F_zuf_kar by factor variable studiensit3

studiensit3	N	F_zuf_kar	sd	se	ci
BA abgeschlossen, kein weiteres Studium	39	3.825	0.920	0.147	0.298
BA & MA abgeschlossen	50	3.757	0.912	0.129	0.259
BA abgeschlossen, noch in weiterem Studium	25	3.307	0.892	0.178	0.368

Anova table with type III sum of squares for F_zuf_kar by studiensit3

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
studiensit3	2	4.596	2.298	2.773	0.067 .
Residuals	111	91.991	0.829	NA	NA

Note:
Signif. codes: 0 ' *** ' 0.001 ' ** ' 0.01 ' * ' 0.05 '.' 0.1 ' ' 1

Tab. 9 Ausgaben zur einfaktoriellen Varianzanalyse

Dann folgt eine Tabelle mit dem Ergebnis des Levene-Tests auf Varianzgleichheit– hier für eine Variable mit 3 Stufen, das nichtsignifikante Ergebnis weist auf Varianzgleichheit hin. (Dieses Ergebnis sollte vor dem Ergebnis der ANOVA betrachtet werden, Varianzhomogenität ist Voraussetzung für die Gültigkeit des Tests).

Levene's test for homogenity of variances (center=mean) for F_zuf_kar against studiensit3

	Df	F value	Pr(>F)
group	2	0.019	0.982
	111	NA	NA

Tab. 10 Ausgaben zum Levene-Test für die einfaktorielle Varianzanalyse

Darüber hinaus gibt BSS auf die Anforderung sowohl die Post-hoc Testergebnisse (Vergleich der Mittelwerte einzelner Gruppenstufen) sowie die Estimated Marginal Means pro Faktorstufe/Gruppe aus (hier nicht dargestellt).

V.2 Durchführung einer linearen Regression

BSS bietet über die Menüführung zwei Wege zur linearen Regression, der erste (a) ist speziell für BSS entwickelt, während sich der zweite (b) stärker an der klassischen R-Syntax orientiert.

a) Der BSS-Ansatz der linearen Regression (Menüleiste: „Model Fitting > Regression > Linear Regression, basic“)

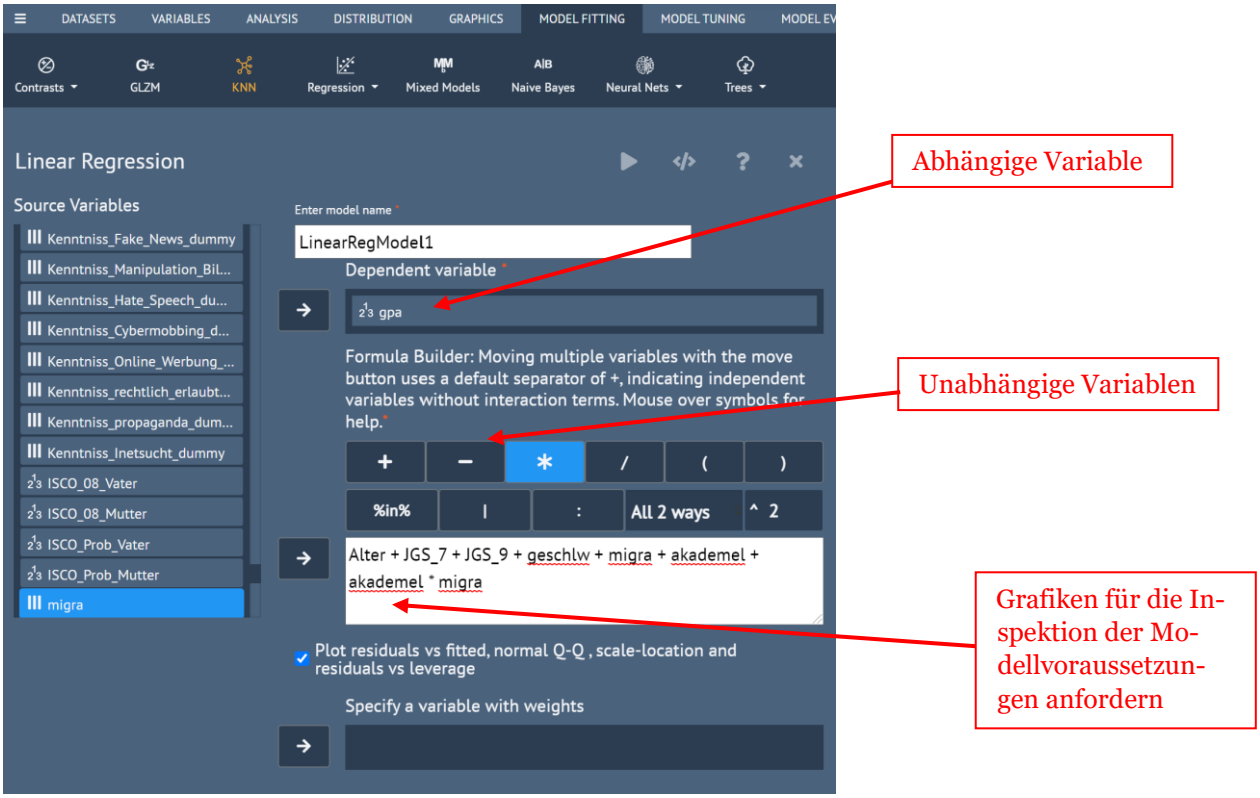


Abb. 35 Untermenü zur Modelformulierung der linearen Regression

Syntax

```
#Creating the model
LinearRegModel1 = lm(gpa ~ Alter + JGS_7 + JGS_9 + geschlw + migra + akademel
+ akademel *
    migra, weights = , na.action = na.exclude, data = schuel2021)
local({
    # Display theoretical model equation and coefficients Display theoretical
    # model
    reg_formula = equatiomatic::extract_eq(LinearRegModel1, raw_tex = FALSE,
wrap = TRUE,
        intercept = "alpha", ital_vars = FALSE)
    BSkyFormat(reg_formula)
  # Display coefficients
    reg_equation = equatiomatic::extract_eq(LinearRegModel1, use_coefs =
TRUE, wrap = TRUE,
        ital_vars = FALSE, coef_digits = BSkyGetDecimalDigitSetting())
    BSkyFormat(reg_equation)
  # Summarizing the model
    BSky_LM_Summary_LinearRegModel1 = summary(LinearRegModel1)
  # Computing 95% confidence interval of the coefficients
    BSky_LM_Summary_LinearRegModel1$coefficients <- cbind(BSky_LM_Sum-
mary_LinearRegModel1$coefficients,
```

```
        stats::confint(LinearRegModel1, level = 0.95, type = "LR")[rowSums(is.na(stats::confint(LinearRegModel1,
            level = 0.95, type = "LR"))) != ncol(stats::confint(LinearRegModel1,
            level = 0.95, type = "LR")), ])
    BSkyFormat(BSky_LM_Summary_LinearRegModel1, singleTableOutputHeader = "Model Summary")
  # Displaying the Anova table
    AnovaRes = anova(LinearRegModel1)
    BSkyFormat(as.data.frame(AnovaRes), singleTableOutputHeader = "Anova Table")
  # Displaying sum of squares table
    df = as.data.frame(AnovaRes)
    totalrows = nrow(df)
    regSumOfSquares = sum(df[1:totalrows - 1, 3])
    residualSumOfSquares = df[totalrows, 3]
    totalSumOfSquares = regSumOfSquares + residualSumOfSquares
    matSumOfSquares = matrix(c(regSumOfSquares, residualSumOfSquares, totalSumOfSquares),
        nrow = 3, ncol = 1, dimnames = list(c("Sum of squares of Regression", "Sum of squares of residuals",
            "Total sum of squares"), c("Values")))
    BSkyFormat(matSumOfSquares, singleTableOutputHeader = "Sum of squares Table")
    plot(LinearRegModel1)  #Adding attributes to support scoring
```

Output (u.a.)

Model: lm(formula = gpa ~ Alter + JGS_7 + JGS_9 + geschlw + migra + akademel + akademel * migra, data = schuel2021, na.action = na.exclude)

LM Summary

Residual Std. Error	df	R-squared	Adjusted R-squared	F-statistic	numdf	dendf	p-value
0.733	657	0.132	0.123	14.246	7	657	< .001***

Note:
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residuals

Min	1Q	Median	3Q	Max
-2.044	-0.538	0.005	0.524	2.104

Coefficients

	Estimate	Std. Error	t value	Pr(>|t|)	2.5 %	97.5 %	2.5 %	97.5 %
(Intercept)	1.361	0.470	2.896	0.004 **	0.438	2.284	0.438	2.284
Alter	0.088	0.044	1.979	0.048 *	0.001	0.175	0.001	0.175
JGS_7	0.124	0.112	1.107	0.269	-0.096	0.343	-0.096	0.343
JGS_9	0.232	0.198	1.171	0.242	-0.157	0.620	-0.157	0.620
geschlw	-0.079	0.058	-1.364	0.173	-0.193	0.035	-0.193	0.035
migra1	-0.066	0.037	-1.769	0.077 .	-0.139	0.007	-0.139	0.007
akademel	-0.200	0.067	-2.975	0.003 **	-0.333	-0.068	-0.333	-0.068
migra1:akademel	-0.085	0.067	-1.271	0.204	-0.217	0.046	-0.217	0.046

Note:
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tab. 11 Ausgabe zur Linearen Regression

Diese Tabellen können exportiert werden – copy paste in Textverarbetungsprogramme funktioniert, aber nur händisch, indem mit der Maus der Text unterlegt und kopiert wird.

Es wurden Grafiken zur Inspektion der Voraussetzungen der linearen Regression (Q-Qplots, Scale mit angefordert (siehe Punkt Prüfung der Voraussetzungen S. 48).

b) Alternative Menüführung zur linearen Regression (Menüleiste: „Model Fitting > Regression > Linear Regression, advanced“)

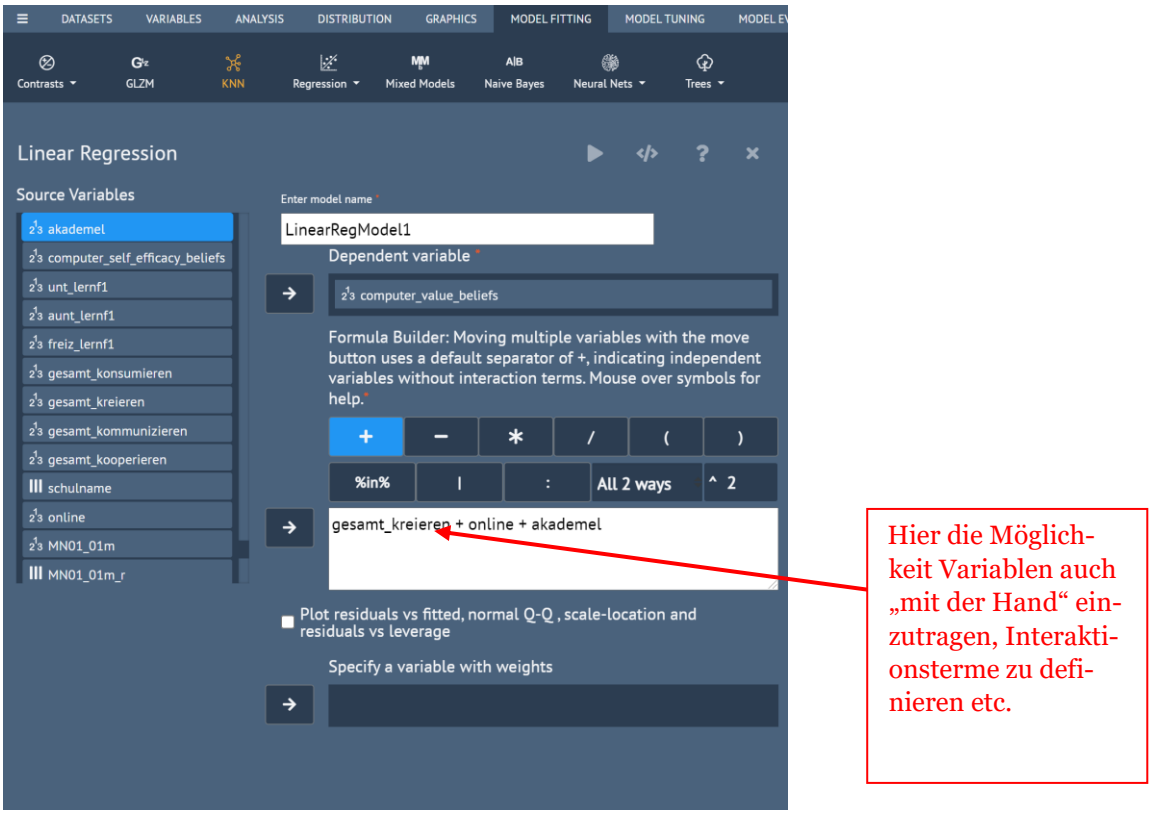


Abb. 36 Menüführung zur Spezifikation einer linearen Regression („advanced“)

Zum Vergleich die einfache/ „klassische“ R-Syntax für das zuletzt spezifizierte Fenster lautet:

```
LinearRegModel1 = lm(computer_value_beliefs ~ gesamt_kreieren + online + akademel, na.action = na.exclude, data = schuel2021)
summary(LinearRegModel1)
```

Der Output ist dann:

R Console Script

```
summary(LinearRegModel1)
Call:
lm(formula = computer_value_beliefs ~ gesamt_kreieren + online +
    akademel, data = schuel2021, na.action = na.exclude)
Residuals:
     Min       1Q   Median       3Q      Max
-2.31730 -0.61257  0.05345  0.65999  1.57023
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      3.22539    0.13026  24.761  < 2e-16 ***
gesamt_kreieren  0.20438    0.06197   3.298  0.00105 **
online           0.04670    0.07766   0.601  0.54787
akademel        -0.01185    0.07921  -0.150  0.88117
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.8401 on 476 degrees of freedom
  (301 Beobachtungen als fehlend gelöscht)
Multiple R-squared:  0.02458, Adjusted R-squared:  0.01843
F-statistic: 3.998 on 3 and 476 DF,  p-value: 0.00789
```

Tab. 11 Ausgabe einer linearen Regression mit der „klassischen“ R-Syntax

Prüfung der Voraussetzungen der Regression: Zur linearen Regression gehört stets die Prüfung der Voraussetzung der Regression (vgl. hierzu bspw. Backhaus et al. 2016, S. 97ff.). BSS bietet einige Optionen der Prüfung der Voraussetzungen über die Menüfenster, die durch weitere R-Syntaxbefehle ergänzt werden können – hier bietet sich das Paket *lmtest* (vgl. Zeileis & Hothorn 2002) an. Grafiken zur Inspektion des Modells können im Regressionsfenster schon mit angeklickt werden, andere Modelinformationen können über das Menüfenster „Modell Statistics“ abgerufen werden.

1. Prüfung Normalverteilung der abhängigen Variablen

(siehe Punkt T-Test/Varianzanalysen)

2. Prüfung linearer Zusammenhang: Möglich ist hier der Rainbow-test (Paket *lmtest*)

```
raintest(LRModel1)

Rainbow test
data: LRModel1

Rain = 1.0867, df1 = 549, df2 = 542, p-value = 0.1661
```

3. Prüfung Normalverteilung der Residuen: Hier hilft die grafische Inspektion mit einem Q-Q-Plot, die im Regressionsfenster als eine der Grafiken im Punkt „Plot Residuals …“ angefordert werden kann.

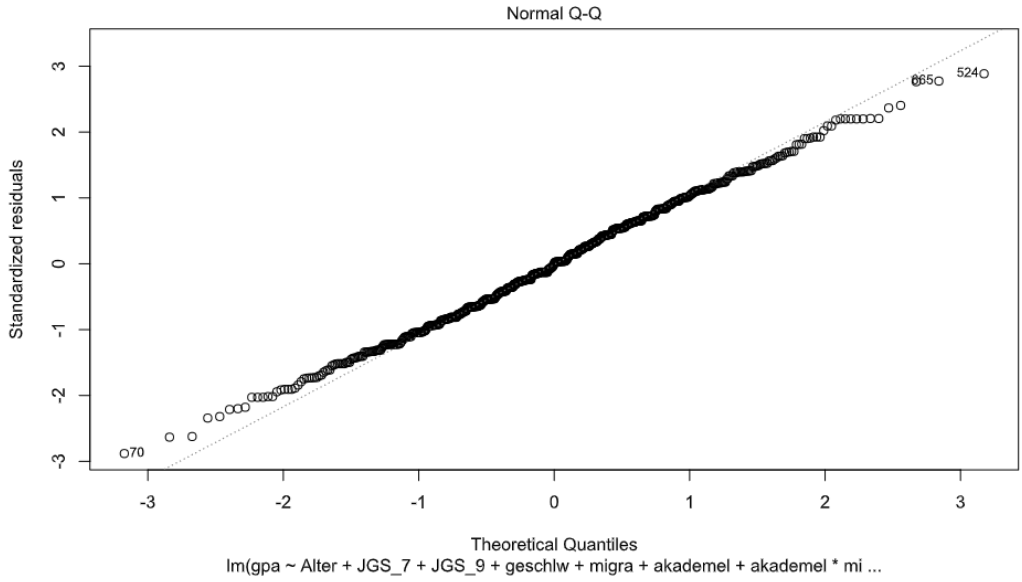


Abb. 36 Plot der studentisierten gegen die theoretischen Residuen zur Untersuchung der Normalverteilung der Residuen

Wenn die Residuen normalverteilt sind, sollten sie möglichst auf der gestrichelten Geraden liegen.

4. Prüfung auf Homoskedastizität: Auch dies kann grafisch mit Anklicken der Grafikoption im Menüfenster überprüft werden

Dafür werden die standardisierten Residuen gegen die durch das Modell vorhergesagten Werte geplottet.

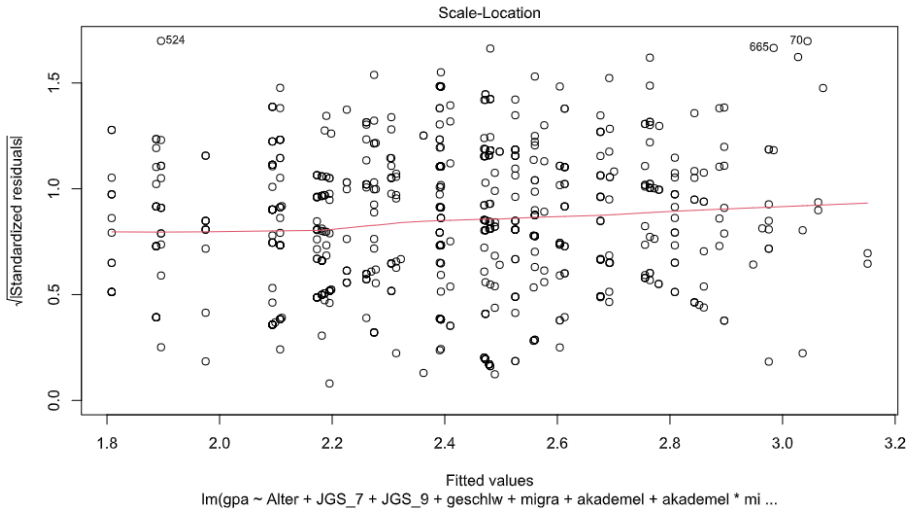


Abb. 38 Standardisierte, absolute Residuen gegen vorhergesagte Werte zur Untersuchung der Homoskedastizität des Fehlers

Bei Homoskedastizität streuen die Punkte eher unsystematisch um die Linie, eine Dreiecksform oder eine U-Form – also systematische Veränderungen – weisen auf Heteroskedastizität zeigt sich (vgl. hierzu bspw. Backhaus et al. 2016, S. 103). Anders formuliert: Wenn sich die Residualwerte mit wachsendem Wert von x systematisch verändern ist davon auszugehen,

dass Heteroskedastizität der Residualwerte vorliegt (Janssen & Laatz 2017, S. 444). Zudem lässt sich über den Breusch-Pagan-Test aus dem *lmtest*-Paket Homoskedastizität testen.

```
bptest(LRModel1)
```

```
studentized Breusch-Pagan test
data: LRModel1
BP = 10.064, df = 5, p-value = 0.07343
```

5. Prüfung Multikollinearität: Über das Menüfenster „Modell statistics“ kann mit der Option „Variance Inflation Factors“ das Modell auf mögliche Multikollinearität der Variablen geprüft werden (Empfehlung: VIF nicht > 5, vgl. Akinwande et al. 2015, S. 750).

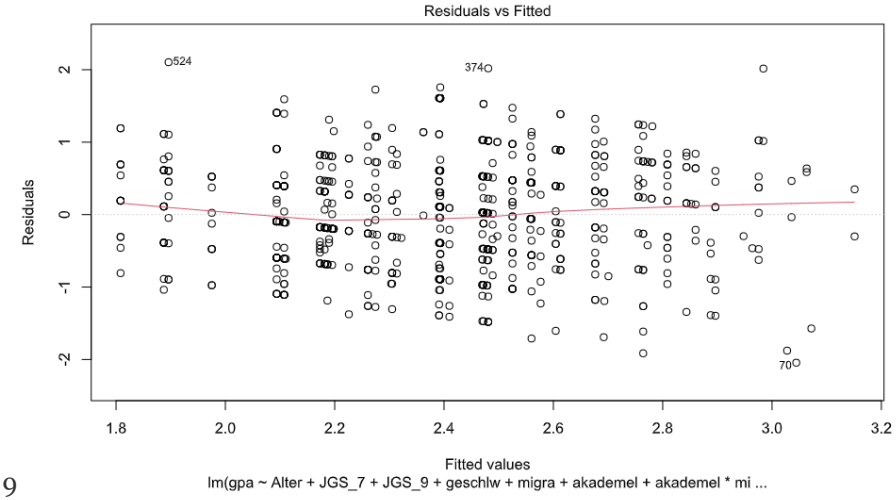
6. Prüfung Autokorrelation: Vor allem für Panel- und Längsschnittdaten kann sich das Problem der Autokorrelation stellen (Fehlerterme einer Person sind sich ähnlicher), hier wird in der Regel der Durbin-Watson-Test genutzt (auch möglich über das R- Paket lmtest).

```
durbinWatsonTest(LRModel1)
```

```
lag Autocorrelation D-W Statistic p-value
1   0.08728603   1.824524  0.008

 Alternative hypothesis: rho != 0
```

7. Prüfung wie gut das Modell an die Daten angepasst ist



9

Abb. 39 Plot der Residuen gegen vorhergesagte Werte zur Untersuchung der Anpassung des Modells an die Daten

Dabei sollte keine systematische Abweichung auftreten (U-Form oder Trend). Dazu kann auch ein Test auf korrekte Spezifizierung angefordert werden.

```
resettest(LRModel1)
```

```
      RESET test
data: LRModel1
RESET = 4.2873, df1 = 2, df2 = 1089, p-value = 0.01397
```

Ausgegeben wird in BSS auch eine Grafik zur Cook’s distance, mit der die Bedeutung von einzelnen Ausreisserwerten für die Regressionsschätzung beurteilt werden kann.

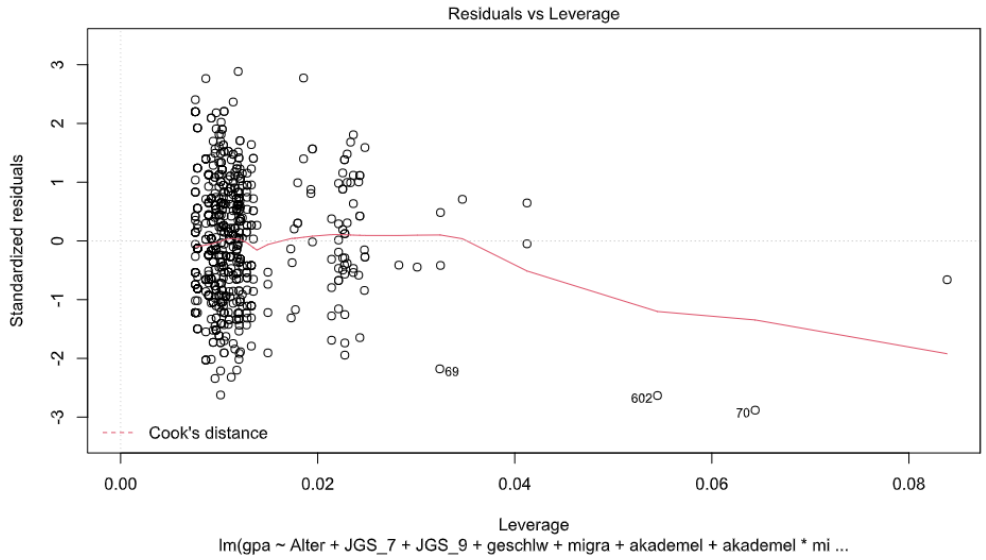


Abb. 40 Cooks Distance zur Analyse der Bedeutung von Ausreißerwerten

Die Grafik weist durchaus auf bedeutsame Ausreißer hin (Fälle 602, 70). Näheres hierzu findet sich online bspw. bei Thieme (2021) oder Walther (2021).

Eine facet-Zusammenstellung entsprechender Diagnosegrafiken können auch über das R Paket *performance* angefordert werden:

```
require(performance)
check_model(LRModel1)
```

V.3 Binäre logistische Regression mit BSS

Auch für die binäre logistische Regression bietet BSS über das Menü zwei Wege an, die derselben Logik wie bei der linearen Regression folgen. Hier wird die Option „Model Fitting > Regression“ > logistic regression, basic“ vorgestellt.

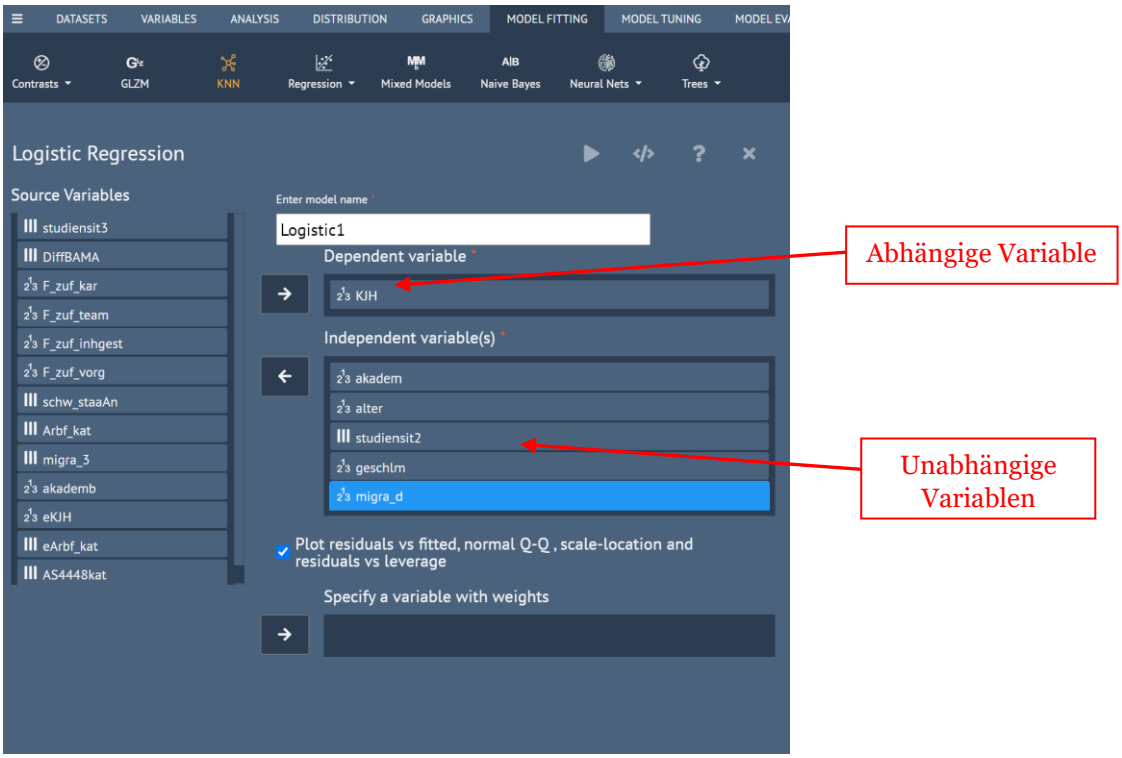


Abb. 41 Menüführung zur Durchführung einer logistischen Regression

Es öffnet sich das Regressionsfenster zur Modellspezifikation. Spezifiziert ist ein Modell mit vier unabhängigen Variablen.

Im Outputfenster zeigen sich anschließend eine Reihe von Angaben zum Modell (Modellformel, Modelstatistiken). Für die geschätzten Koeffizienten gibt BSS zwei Tabellen aus: Zunächst eine Tabelle mit den Koeffizienten, der Standardfehlern und den Werten für die Signifikanztest.

Model: glm(formula = KJH ~ akadem + alter + studiensit2 + geschlm + migra_d, family = binomial(link = "logit"), data = Absol21_spt, na.action = na.exclude)

GLM Summary

Null Deviance	df	Residual Deviance	df Residual	AIC	Dispersion	Iterations
157.897	113	151.575	108	163.575	1	4

Deviance Residual Summary

Min	1Q	Median	3Q	Max
-1.468	-1.082	-0.854	1.154	1.553

Coefficients

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.149	1.130	1.017	0.309
akadem	-0.420	0.389	-1.080	0.280
alter	-0.024	0.038	-0.636	0.525
studiensit2 BA & MA abgeschlossen	-0.803	0.394	-2.039	0.041 *
geschlm	-0.049	0.545	-0.091	0.928
migra_d	0.120	0.520	0.230	0.818

Note:
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tab. 12 Ausgabe zur logistischen Regression

Builds a logistic model

```
Logistic1 = glm(KJH ~ akadem + alter + studiensit2 + +geschlm + migra_d,
weights = , family = binomial(link = "logit"),
    na.action = na.exclude, data = Absol21_spt)
local({
    if (!is.null(Logistic1)) {
    # Display coefficients
        reg_equation = equatiomatic::extract_eq(Logistic1, use_coefs =
TRUE, wrap = TRUE,
            ital_vars = FALSE, coef_digits = BSkyGetDecimalDigitSetting())
        BSkyFormat(reg_equation)
    # Summarizing the model
        BSky_Logistic = summary(Logistic1)
        BSkyFormat(BSky_Logistic, singleTableOutputHeader = "Model Sum-
mary")
    # Analysis of variance
        BSky_anova = anova(Logistic1, test = "Chisq")
```

```
        BSkyFormat(as.data.frame(BSky_anova), singleTableOutputHeader =
"Analysis of Deviance Table")
        BSkyFormat(attr(BSky_anova, "heading"))
    # McFadden R2
        BSkyFormat(pR2(Logistic1), singleTableOutputHeader = "McFadden R2")
    # odds ratio and 95% confidence interval
        BSkyFormat(exp(cbind(OR = coef(Logistic1), confint.glm(Logistic1,
level = 0.95))),
            singleTableOutputHeader = "Odds ratio(OR) and 95% Confidence
interval ")
    # Displaying plots
        plot(Logistic1)
        # Adding attributes to support scoring We don't add dependent and
        # independent variables as this is handled by our functions
        attr(.GlobalEnv$Logistic1, "classDepVar") = class(Absol21_spt[,
c("KJH")])
        attr(.GlobalEnv$Logistic1, "depVarSample") = sample(Absol21_spt[,
c("KJH")],
            size = 2, replace = TRUE)
    } })
```

Dann folgt eine Tabelle die u.a. das Pseudo-R^2 Mc Faddden enthält:

McFadden R2

llh	llhNull	G2	McFadden	r2ML	r2CU
-75.791	-78.949	6.314	0.040	0.054	0.072

Tab. 13 Ausgabe zu Modellgütestatistiken

Weitere Pseudo-R^2 Maße können über die R-Syntax angefordert werden (bspw. mit dem Paket *DescTools*, das mit BSS installiert wird). Der R Syntax Befehl für das Cox & Snell R^2 sowie das Nagelkerke R^2 ist:

```
require(DescTools)
PseudoR2(Logistic1, c("CoxSnell", "Nagel"))
```

Im Outputfenster erscheint dann (unformatiert):

```
CoxSnell    Nagelkerke
0.2403605   0.3539258
```

Die Odds ratio und die Konfidenzintervalle der Schätzer werden in einer eigenen Tabelle ausgegeben:

Odds ratio(OR) and 95% Confidence interval

	OR	2.5 %	97.5 %
(Intercept)	3.154	0.346	31.456
akadem	0.657	0.304	1.405
alter	0.976	0.902	1.053
studiensit2 BA & MA abgeschlossen	0.448	0.204	0.962
geschlm	0.952	0.321	2.788
migra_d	1.127	0.404	3.176

Tab. 14 Ausgabe Odds Ratio der logistischen Regression

Die Güte der Klassifikation durch das statistische Modell kann durch den Hosmer-Lemeshow-Test geprüft werden, der beispielsweise mit R-Paket *performance* über die Syntax angefordert werden kann.

```
require(performance)
performance_hosmer(Logistic1)
```

```
# Hosmer-Lemeshow Goodness-of-Fit Test
 Chi-squared: 8.348
df: 8
p-value: 0.400
Summary: model seems to fit well.
```

Für die Interpretation der Koeffizienten/Schätzer kann über die Odds Ratio, die ein Relatives Maß darstellen, hinaus die average marginal effects (AME) verwendet werden (vgl. hierzu Leeper 2018). Dies ist mit der Installation und dem Aufrufen des Pakets *margins* über die R-Syntax möglich. Die Ausgabe erfolgt wie alle R-Code-gestützten Analysen, die nicht in BSS implementiert sind, unformatiert.

Befehle über die R-Syntax- (nach Paketinstallation):

```
require(margins)
ame_logit_ohilf <- margins(Logistic1)
summary(ame_logit_ohilf)
```

Im Ergebnis zeigt sich folgender Output:

```
summary(ame_logit_ohilf)
factor               AME     SE       z      p   lower     upper
akadem           -0.0991 0.0900 -1.1014 0.2707 -0.2755    0.0773
alter            -0.0057 0.0089 -0.6401 0.5221 -0.0233    0.0118
geschlm          -0.0116 0.1285 -0.0906 0.9278 -0.2636    0.2403
migra_d           0.0283 0.1227  0.2306 0.8176 -0.21210   0.2687
studiensit2 BA   -0.1944 0.0928 -2.0944 0.0362 -0.3763  -0.0125
```

Tab. 15 Ausgabe zur Average Marginal Effects einer logistischen Regression

Testung der Voraussetzungen der logistischen Regression:

Bei der logistischen Regression sollte mindestens getestet werden auf

a) das mögliche Vorhandensein bedeutsamer Ausreißer
b) mögliche Multikollinearität der unabhängigen Variablen sowie
c) Linearität des Logits

zu a) Mögliche bedeutsame Ausreißer können mit dem Paket *car* getestet werden. Einzelne bedeutsame Ausreißer können gerade bei kleineren Stichproben wie in der linearen Regression die Funktionsschätzung ungünstig beeinflussen.

#Laden des Pakets *car* und Durchführung Outliertest

```
require(car)
outlierTest(Logistic1)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:rstudent unadjusted p-value Bonferroni p
41          1.610745               0.10724           NA
```

#Zudem kann über die von BSS angebotenen Grafiken zu den Residuen und zu Ausreißern mit dem Paket *car* eine Grafik zur detailierten Inspektion von Ausreißern erzeugt werden:

```
influenceIndexPlot(Logistic1)
```

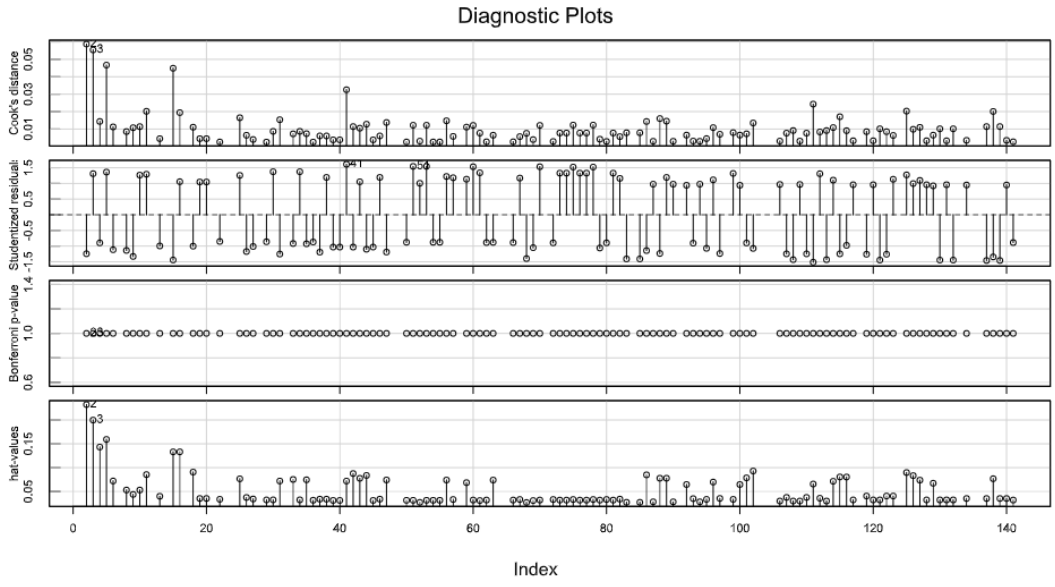


Abb. 42 Grafik zur Ausreißerdiagnostik im Paket *car*

Die Grafik zeigt sowohl the Cook's distance, die studentisierten Residuals, die Bonferroni p Werte sowie hh-Werte.

Zu b) Für die Testung auf Multikollinearität können nach Modellschätzung über die Menüoptition „Model statistics" können die „Variance Inflation Faktors" (VIF) aufgerufen werden (s. auch die Angaben zur linearen Regression, das Modell muss oben rechts im Hauptmenüfenster aufgerufen sein). Die Werte liegen hier alle sehr ideal um die 1.

Variance-inflation factors

akadem	alter	studiensit2	geschlm	migra_d
1.014	1.006	1.016	1.022	1.018

Tab. 16 Ausgabe zur Variance Inflation Factors (VIF)

Zu c) Linearität des Logits: Inspiziert werden sollte, ob es jeweils eine lineare Beziehung zwischen der/den metrischen unabhängigen Variable(n) und dem Logit (Log-Funktion) der abhängigen Variable gibt. Dies kann mit dem Paket *car* inspiziert werden

```
residualPlots(Logistic1)
```

Dabei werden sowohl Linearitätstests sowie Grafiken für die einzelnen unabhängigen Variablen erzeugt:

```
             Test stat        Pr(>|Test stat|)
akadem          0.432          0.4230
alter           0.226          0.6342
studiensit2    0.543          0.678
geschlm         0.245          0.567
migra_d         0.354          0.432
```

Tab 17 Ausgabe Linaritätstest aus dem Packet Performance

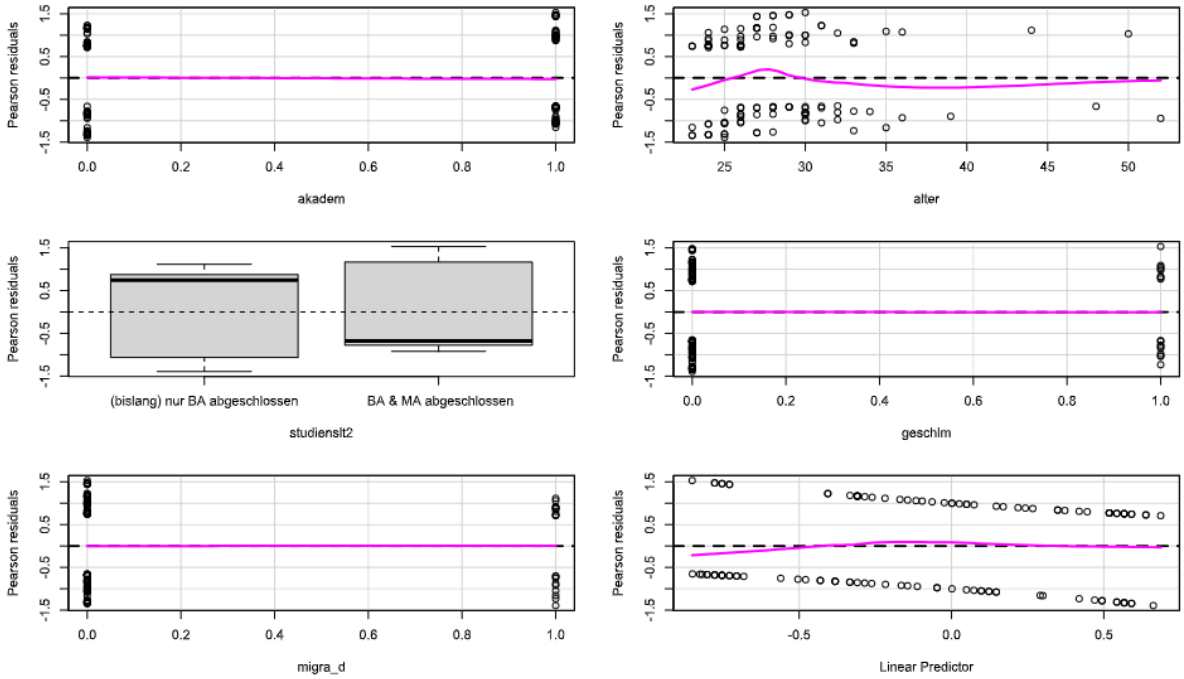


Abb. 43 Facet-Grafik zur Diagnostik der Linearität der Logits mit den Prädiktoren im Paket *car*

V.3 Ordinale Regression mit BSS

Für die Analyse ordinaler abhängiger Variablen lässt sich die Prozedur unter „Model Fitting > Ordinal Regression“ aufrufen (zur ordinalen Regression vgl. bspw. Janssen & Laatz 2017, S. 465 ff., Große Schlarmann & Galatsch 2014). Wichtige Voraussetzung für die Nutzung der Menüführung ist, dass die abhängige Variable vorher als „ordered factor“ definiert worden ist.[3]

Grundsätzlich gibt es hier zwei Modellarten: Proportional Odds (oder Kumulative Logit) Modelle und Sequenzielle Modelle/Continuation Ratio Modelle. Vorgestellt wird hier das kumulative Proportional Odds/Kumulative-Logit-Modell, dass hinter der ordinalen Skala eine dahinterliegende latente metrische Struktur annimmt (vgl. Große Schlarmann & Galatsch 2014).

Ohne hier weiter auf die mathematische Theorie einzugehen, wird hier nur die Operationalisierung in BSS gezeigt, die auf dem R-Paket *MASS* und der Funktion *polr* beruht (Venables & Ripley 2002). Über die Menüführung „Model Fitting > Regression > Ordinal“ öffnet sich folgendes Menüfenster, in dem dann abhängige und unabhängige Variablen festgelegt werden können:

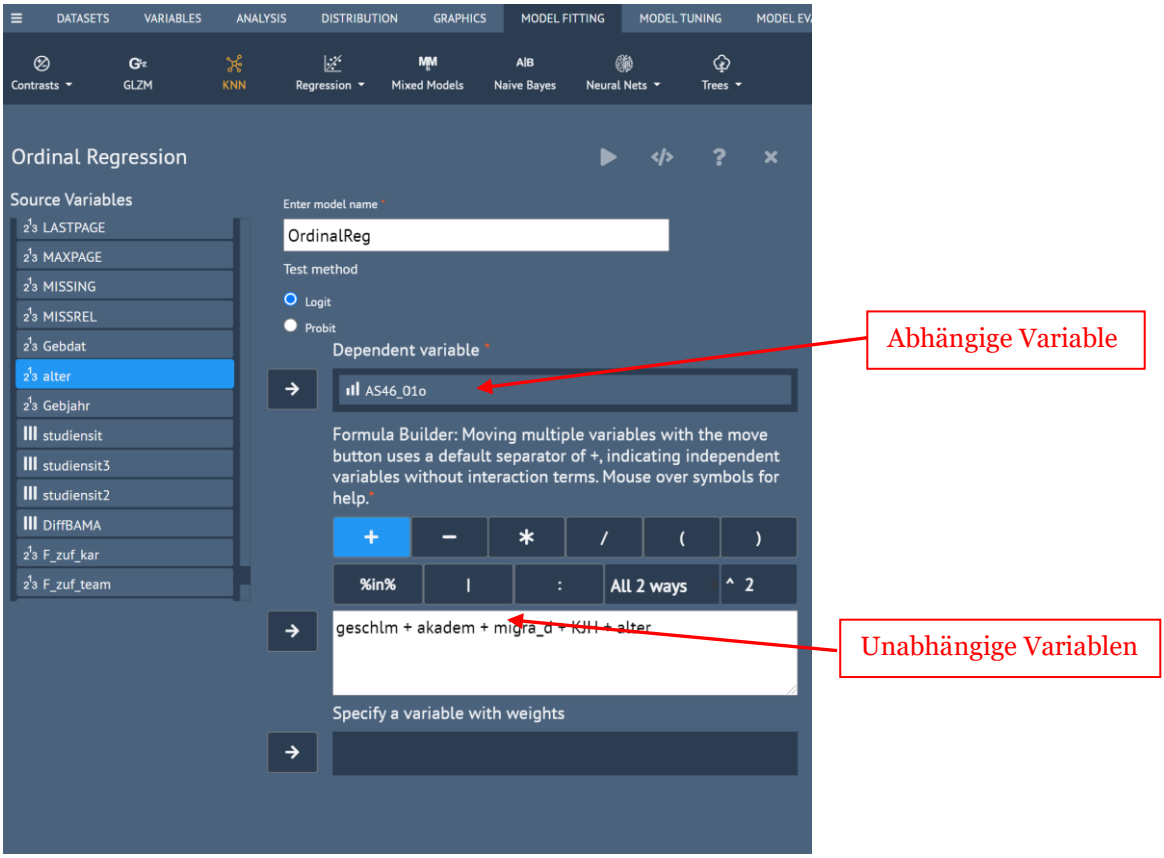


Abb. 44 Menüführung ordinale Regression

Die unabhängigen Variablen können aber auch direkt hineingeschrieben werden. Zudem ist es dann nötig, die unabhängigen Variablen mit einem + Zeichen zu Verbinden. Nach Spezifikation und „OK“ erscheint folgender output, der mit der Modellformel beginnt.

polr(formula=mons ~ sprache + geschlw + akadem + gym, data=homes0208, Hess=TRUE, method='logistic')

[3] Das gelingt beispielsweise mit folgender R-Syntax-Logik: `daten02$var24ord <- factor(daten02$var24,ordered = TRUE)`. Und um die neue Variable auch im Datenfenster angezeigt zu bekommen: `BSkyLoadRefresh("Absol21_spt")`

Es zeigen sich zunächst Angaben zur Modellgüte, die vor allem im Modellvergleich interessant werden.

Model Statistics

edf	logLik	AIC	BIC	deviance	df.residual	nobs
9	-122.398	262.796	287.422	244.796	105	114

Tab. 18 Ausgabe Modelgüteindikatoren der logistischen Regression

Die Tabelle mit der Koeffizientenschätzern zeigt die Odds ratios für die „Schwellenschätzer" (der Koeffizienten) und deren Konfidenzintervalle. Die Terminologie unterscheidet hier Schwellen- und Lageschätzer – die Lageschätzer sind die Koeffizienten, die für unabhängigen Variablen geschätzt werden, die Schwellenschätzer Werte für den Übergang von einer Stufe der abhängigen Variablen zur nächsten.

Exponentiated Coefficients and Confidence Intervals

	term	estimate	std.error	t.value	conf.low	conf.high	coef.type	p.value(z)	p.value(t)
geschlm	geschlm	0.512	0.497	-1.349	0.192	1.364	coefficient	0.177	0.180
akadem	akadem	1.369	0.366	0.858	0.670	2.821	coefficient	0.391	0.393
migra_d	migra_d	0.662	0.481	-0.857	0.258	1.722	coefficient	0.391	0.393
KJH	KJH	1.473	0.368	1.052	0.718	3.055	coefficient	0.293	0.295
alter	alter	1.077	0.041	1.780	0.998	1.178	coefficient	0.075 .	0.078 .
gar nicht zufrieden|2	gar nicht zufrieden|2	0.077	1.548	-1.652	NA	NA	scale	0.098 .	0.101
2|3	2|3	0.405	1.265	-0.715	NA	NA	scale	0.475	0.476
3|4	3|4	1.490	1.222	0.326	NA	NA	scale	0.744	0.745
4|sehr zufrieden	4|sehr zufrieden	10.609	1.233	1.916	NA	NA	scale	0.055 .	0.058 .

Note:
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tab. 19 Ausgabe Odds Ratio der ordinalen Regression

Weitere Angaben zur Modellgüte können über das Menüfenster „Model Evaluation" „Summarize Model statistics" aufgerufen werden, hier lassen sich auch die nicht exponenzierten Schätzer anzeigen.

Testung der Voraussetzungen der ordinalen Regression

Zwei Voraussetzungen der ordinalen Regression sollten auf jeden Fall getestet werden.

- Zum Testen der zentralen Annahme *der Proportional odds/"equal slope assumption"* (die Abstände zwischen den Gruppen haben keinen Einfluss auf die Koeffizienten = die logistischen Funktionen für die Wahrscheinlichkeit, mindestens Stufe x zu erreichen, sind für jede Stufe parallel verschoben) bietet sich bspw. das Paket *brant* (https://benjaminschlegel.ch/r/brant/) an, das aus diesem Test „besteht".

```
require(brant)
brant(oModel1)
```

- Auch sollte auf mögliche Multikollinearität der Prädiktoren geprüft werden – dadurch, dass das ordinale Modell keine Konstante enthält, ist der Aufruf der VIF über die Model Statistics wenig sinnvoll (und erzeugt auch eine Warnmeldung). Sinnvoll ist es, das ordinale Modell als lineares Modell zu schätzen und sich dann für dieses über das Menü und „Model Statistics“ die VIF ausgeben zu lassen.

Für weitere Erläuterungen der Anwendung der ordinalen Regression mit R und der Funktion polr kann sowohl die Vorstellung auf der Homepage der Statistikberatung der UCLA (https://stats.idre.ucla.edu/r/dae/ordinal-logistic-regression/) als auch die Seite https://www.r-bloggers.com/2019/06/how-to-perform-ordinal-logistic-regression-in-r/ empfohlen werden.

V.4 Lineare Mehrebenenanalyse

Für die Analyse genesteter Daten z.B. „Schüler*innen in Schulen“, „Teilnehmer*innen in Kursen“ bietet sich bei entsprechend hohem Varianzanteil auf Kontextebene Mehrebenenanalysen an (vgl. als Übersicht bspw. Langer 2010). In BSS implementiert sind bislang (einfache) lineare Mehrebenenregressionen, die über „Model Fitting > Mixed Models, basics“ aufgerufen werden können (BSS greift auf das Paket *lme4* zurück, vgl. Bates et al. 2015).

Für die Durchführung empfiehlt sich anfangs immer ein „Leermodell“ (für das nur die abhängige Variable und die Kontextvariable („Nesting unit“) spezifiziert werden muss).

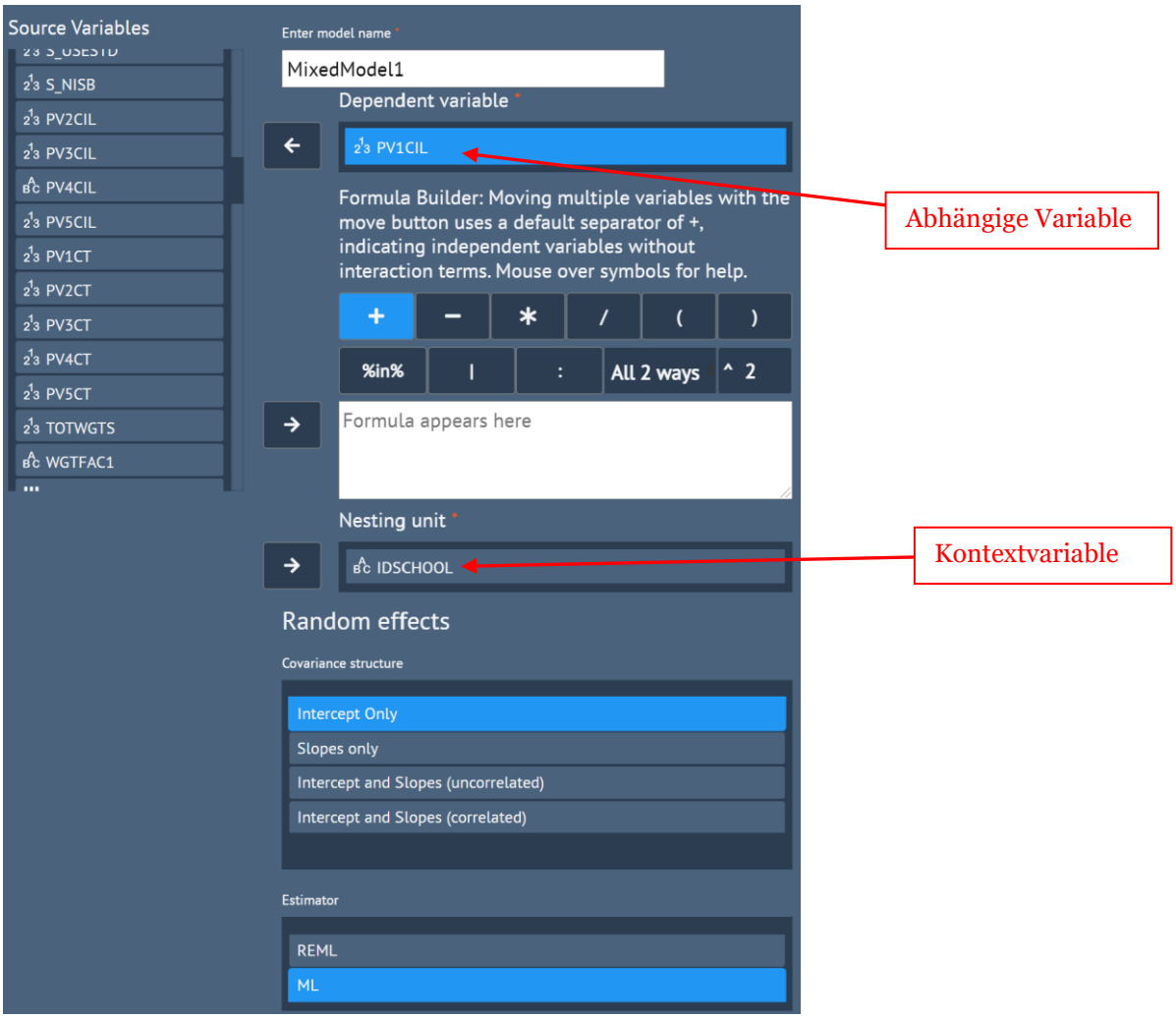


Abb. 45 Menüfenster zur Spezifizierung eines linearen Mehrebenenmodells

Über die „Nesting unit“ wird die Gruppierungs-/Kontextvariable angeben, die mehrere Fälle umfasst (hier: „IDSCHOOL“ als Identifier für die jeweilige Schule).

Mit diesem Modell ohne unabhängige Variable und nur mit Konstante kann u.a. die Varianzzerlegung geschätzt werden. Als Information muss in dem Unterfenster *Options* der ICC angefordert werden, der in einem Zwei-Ebenen-Modell (Schüler*innen in Schulen, Patient*innen in Krankenhäusern, Jugendliche in Jugendzentren) ohne unabhängige Variable den Varianzanteil auf Kontextebene abbildet. Unter Options können weitere Statistiken der Mehrebenenanalyse angefordert werden.

In dem umfangreichen Output findet sich dann auch eine entsprechende Angabe zum ICC:

Intraclass Correlation Coefficient

	Values
adjusted	0.400
conditional	0.400

Tab. 20 Varianzzerlegung im Nullmodell

In diesem Beispiel entfallen 40% der Varianz der abhängigen Variablen auf den Kontext, sprich die einzelnen Schulen.

Ein dann im zweiten Schritt spezifiziertes Modell (mit variierendem Intercept aber fixem Slope, ohne Interaktionen der unabhängigen Variablen) wird wie folgt aufgerufen:

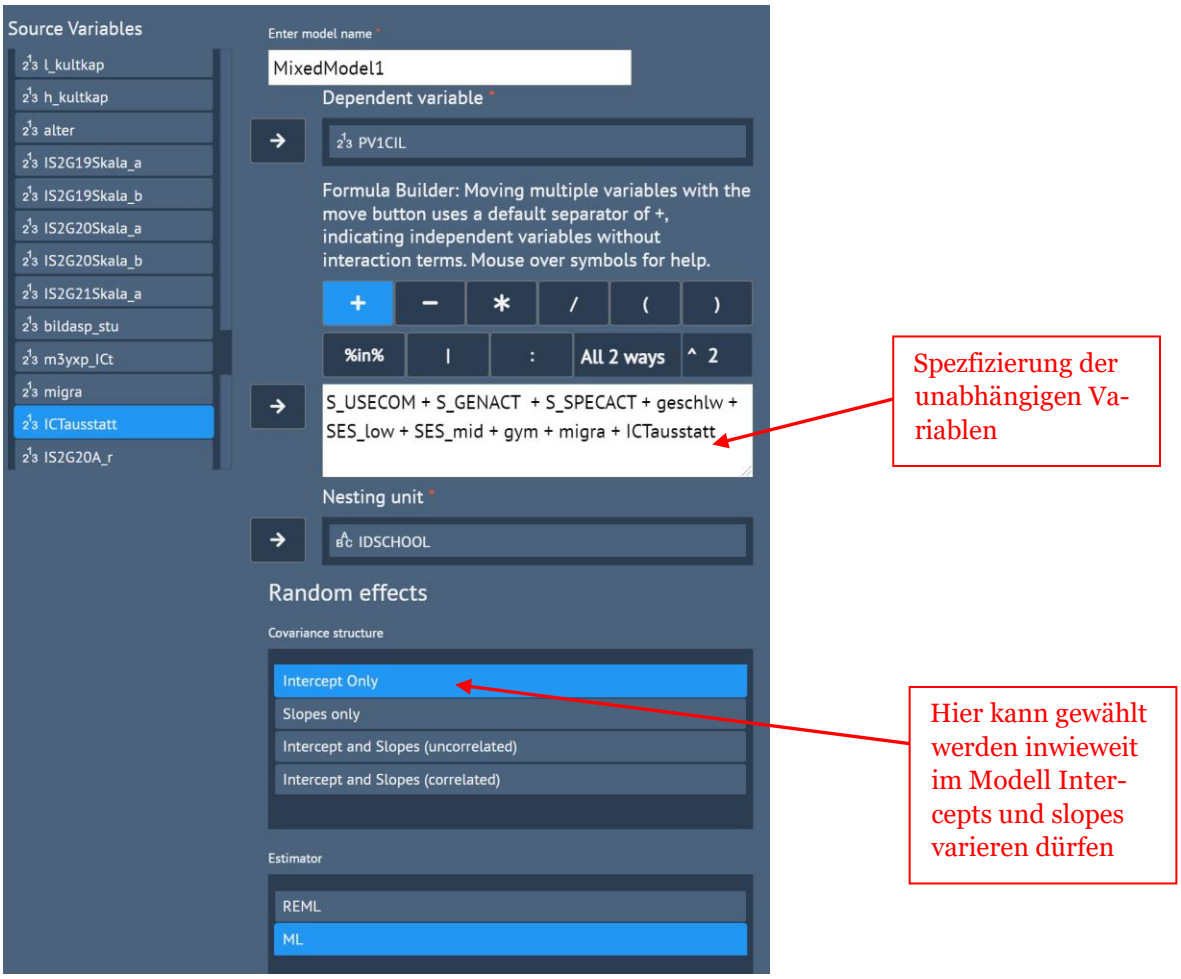


Abb. 46 Aufruf eines linearen Mehrebenenmodells mit random Intercept

Über Options können weiter Ausgaben angefordert werden, wie bspw. die Varianzzerlegung (ICC) oder auch Grafiken.

Die Syntax für die oben angeklickten Befehle ist

```
#Creating and summarizing the mixed model
MixedModel1 <- lmer(PV1CIL ~ S_USECOM + S_GENACT  + S_SPECACT + geschlw + SES_low +
SES_mid + gym + migra + ICTausstatt + (1| IDSCHOOL), REML=FALSE, data =
ICILS20218d_kurzR)
#Display theoretical model equation and coefficients
#Display theoretical model
reg_formula = equatiomatic::extract_eq(MixedModel1, raw_tex = FALSE, wrap = TRUE,
 intercept = "alpha", ital_vars = FALSE)
BSkyFormat(reg_formula)
#Display coefficients
reg_equation = equatiomatic::extract_eq(MixedModel1, wrap = TRUE,
ital_vars = FALSE, use_coefs = TRUE, coef_digits = BSkyGetDecimalDigitSetting())
BSkyFormat(reg_equation)
#Anova and Effect sizes
BSky.Anova.Table <-anova(MixedModel1)
BSkyFormat(as.data.frame(BSky.Anova.Table), singleTableOutputHeader = "ANOVA Ta-
ble")
```

```
#Calculating Effect sizes
BSky.effect.sizes <- with(BSky.Anova.Table, NumDF / DenDF * BSky.Anova.Table$"F value" / (1 + NumDF / DenDF * BSky.Anova.Table$"F value"))
names(BSky.effect.sizes) <- row.names(BSky.Anova.Table)
BSkyFormat(BSky.effect.sizes, singleTableOutputHeader = "Effect Sizes: Semi-partial R-squared")
BSkySummaryRes <- summary(MixedModel1)
#We store the results of the print into an object to suppress the plain text output from R
BSkySummaryRes <- BSkyprint.summary.merMod (BSkySummaryRes, correlation =TRUE )

#Intraclass Correlation Coefficient
BSkyICC <-performance::icc(MixedModel1)
BSkyICCAsMatrix <-matrix(c(BSkyICC$ICC_adjusted, BSkyICC$ICC_conditional), nrow = 2, ncol = 1, dimnames = list(c("adjusted", "conditional"), c("Values")))
BSkyFormat(BSkyICCAsMatrix, singleTableOutputHeader = "Intraclass Correlation Coefficient")

#Creating and summarizing the mixed model
MixedModel1=lmer(gpa_notena~hisklag1+as_mig2d+gym+geschlw+as_age+ (1|idsch),REML=TRUE,data =daten06)
#Anova and Effect sizes
BSky.Anova.Table <-anova(MixedModel1)
BSkyFormat(as.data.frame(BSky.Anova.Table),singleTableOutputHeader = "ANOVA Table")
#Calculating Effect sizes
BSky.effect.sizes <- with(BSky.Anova.Table,NumDF / DenDF * BSky.Anova.Table$"F value" / (1 + NumDF / DenDF * BSky.Anova.Table$"F value"))
names(BSky.effect.sizes) <-row.names(BSky.Anova.Table)
BSkyFormat(BSky.effect.sizes,singleTableOutputHeader = "Effect Sizes: Semi-partial R-squared")

BSkySummaryRes <- summary(MixedModel1)
#We store the results of the print into an object to suppress the plain text output from R
BSkySummaryRes <- BSkyprint.summary.merMod(BSkySummaryRes,correlation =TRUE)
```

Im Folgenden werden nur Teile des Outputs beschrieben. Zunächst zeigt BSS die geschätzte Gleichung und informiert über Fallzahlen und Zahl der Kontexteinheiten

```
Number of obs: 2582, groups:  IDSCHOOL, 180
```

,

sowie über die Varianzkomponenten (Random Effcets) zwischen den Ebenen (IDSCHOOL als Kontextvariable).

Random Effects

Groups	Name	Variance	Std.Dev.
IDSCHOOL	(Intercept)	953.600	30.880
Residual		3285.000	57.310

Tab. 21 Varianzkomponente des Mehrebenenmodells

Auf Anforderung wird auch erneut der ICC ausgegeben, der sich gegenüber dem Leermodell verändert hat (hier nicht dargestellt).

Die Tabelle mit den festen Effekten („Fixed Effects“) zeigen die geschätzten Koeffizienten für die unabhängigen Variablen.

Fixed Effects

	Estimate	Std. Error	df	t value	Pr(>|t|)
(Intercept)	478.655	10.311	2510.551	46.421	< .001***
S_USECOM	-0.372	0.136	2461.090	-2.732	0.006 **
S_GENACT	1.131	0.154	2527.416	7.328	< .001***
S_SPECACT	-0.283	0.146	2476.587	-1.939	0.053 .
geschlw	6.536	2.365	2454.760	2.763	0.006 **
SES_low	-3.777	3.735	2532.960	-1.011	0.312
SES_mid	-1.560	3.003	2496.403	-0.519	0.604
gym	58.894	5.661	166.613	10.404	< .001***
migra	-8.327	2.612	2547.991	-3.187	0.001 **
ICTausstatt	11.632	3.581	2459.103	3.248	0.001 **

Note:
Signif. codes: 0 ' *** ' 0.001 ' ** ' 0.01 ' * ' 0.05 '.' 0.1 ' ' 1

Tab. 22 Ausgabe fester Effekte und Effektgrößen im Mehrebenenmodell

Als Effektstärken für die unabhängigen Variablen gibt BBS Semiparitale R^2 aus.

Effect Sizes: Semi-partial R-squared

S_USECOM	S_GENACT	S_SPECACT	geschlw	SES_low	SES_mid	gym	migra	ICTausstatt
0.003	0.021	0.002	0.003	0.000	0.000	0.394	0.004	0.004

Tab. 23 Ausgabe Effektstärken im Mehrebenenmodell

Angefordert werden können über das BSS-Menü unter Options weitere Informationen und Grafiken.

Dies ist nur ein erster Einstieg in einfachere Mehrebenenmodelle, eine sehr hilfreiche Einführung in die lineare Mehrebenenanalyse mit R und dem (auch von BSS verwendeten) Paket *lme4* gibt F. Votta unter https://www.rpubs.com/favstats/multilevel. Mit dem Paket *lme4* sind auch andere Mehrebenenanalysen (u.a. binär logistische, ordinale) möglich (hier ist für die Schätzung eines binär logistische Multilevelmodell mit R https://stats.idre.ucla.edu/r/dae/mixed-effects-logistic-regression/ eine hilfreiche Seite).

V.5 Explorative Faktorenanalysen

Über das Menü von BSS können auch explorative Faktorenanalysen und Hauptkomponentenanalysen aufgerufen werden. Ohne hier ausführlich auf den Unterschied einzugehen (vgl. hierzu Bühner 2004, S. 154ff.), kann vereinfacht formuliert werden, dass eine Hauptkomponentenanalyse versucht, möglichst alle Varianz über Hauptkomponenten zu erklären, während die Faktorenanalyse darauf zielt, die Korrelation bzw. die Kovarianz der Variablen zu erklären (Noak 2007, S. 25).

Für die Durchführung bietet sich die Nutzung von BlueSky Statistics mit weiteren R-Pakete an. Exemplarisch soll dieses an einer Faktorenanalyse vorgestellt werden, für die BlueSky Statistics auf das R-Basis-Paket *stats* und das Paket *GPArotation* zurückgreift.

Es gibt es in R vielfältige Möglichkeit, Daten aus einem Datensatz für die Analysen auszuwählen, aber als eine pragmatischer Zugang kann für Faktorenanalysen folgende Vorgehensweise vorgeschlagen werden: Falls die Faktorscores nicht direkt im Arbeitsdatensatz benötigt werden, bietet es sich in R an, mit der Subsetfunktion die Items für die Faktorenanalyse in einen eigenen Teildatensatz auszuwählen und dann – bei Bedarf – auch mit diesen eine Z-Standardisierung vorzunehmen. Beides kann mit BSS mit dem Syntaxfenster erfolgen:

Mit „Analysis > Factor Analysis > FA Factor“ öffnet sich das Menu, für Faktoranalysen, alternativ kann auch über „Analysis > Factor Analysis > PCA Principal Components“ eine Hauptkomponentenanalyse aufgerufen werden.

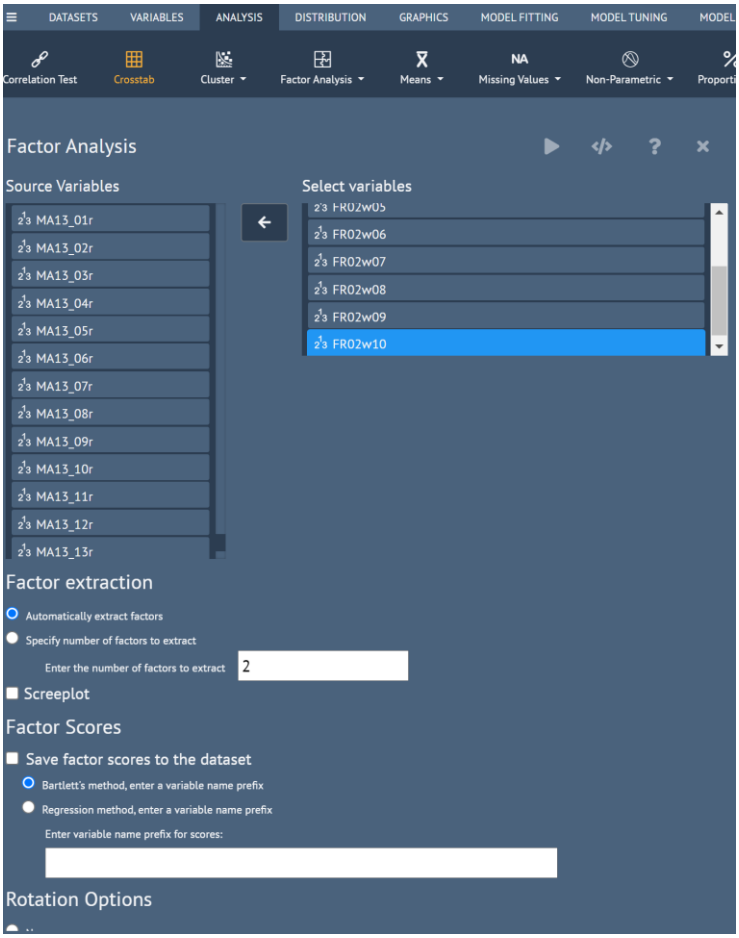


Abb. 47 Menüfenster zur Durchführung eine Faktoranalyse

Hier können dann die entsprechenden Variablen für die Faktoranalyse ausgewählt werden, zudem können dann Extraktionsmethode und Rotationsmethode ausgewählt werden – die Optionen sind hier ähnlich zu denen bei bspw. IBM SPSS ©. Erwogen werden sollte ob eine Z Standardisierung sinnvoll ist.

Natürlich kann die Faktoranalyse so durchgeführt werden, allerdings macht es inhaltlich viel Sinn, vor der eigentlichen Faktorenanalysen die Datenqualität für die Faktoranalyse zu prüfen als auch zu versuchen, die Zahl der zu bestimmenden Faktoren vorher einzugrenzen.

Wenn nun die Daten entsprechend vorliegen, sind die folgenden Schritte der Faktorenanalyse zu gehen (vgl. bspw. Böhner 2004; Backhaus et al. 2016):

Schritt 1: Prüfung Datenqualität für die Faktorenanalyse

Zur Prüfung der Datenqualität für die Faktorenanalyse bietet BSS keine Möglichkeit, die an, daher ist es sinnvoll, entsprechende Kennwerte wie das Kaiser-Maier-Olkin-Kriterium (KMO) mit andere R-Paketen anzufordern. Dies geht u.a. über das R-Paket *psych*, das hier auch im Weiteren für die Faktorenanalyse genutzt wird (und auch eigene Optionen für Faktoranalysen bereit hält). Hier muss der Befehl *KMO()* auf den Teildatensatz mit den standardisierten oder den unstandardisierten Variablen angewendet werden – daher sollten nicht beide in dem Datensatz vorhanden sein

```
#Laden des Pakets und Durchführung
require(psych)
kmo_data <- KMO(teildatensatz_fak)
```

Schritt 2: Auswahl der Zahl der Faktoren

Bei der eigentlichen Faktorenanalyse besteht in BSS besteht wie in SPSS die Möglichkeit die Zahl der Faktoren vorzugeben oder den Computer die Auswahl anhand des Eigenwertekriteriums (>=1) zu überlassen (und sich mit einem Screeplot auch das Knee-Kriterium anzuschauen). Da das Eigenwertekriterium auch eher als Anhaltspunkt denn als Determinate verstanden werden sollte und eigentlich auch nur für Hauptkomponentenanalysen sinnvoll ist, ist es ratsam, verschiedene Kriterien für die Auswahl der Zahl der Faktoren heranzuziehen (vgl. Bühner 2004; auch Backhaus et al. 2016). So bietet das eben genannte Paket *psych* weitere Möglichkeiten der die Zahl der auswählenden Faktoren einzugrenzen, was hier vorab genutzt wird. Konkret wird zur Entscheidungsfindung sowohl die Parallelanalyse sowie das VSS- als auch MAP-Informationskriterien (vgl. Luhmann 2015; Bühner 2004).

- Durchführung Parallelanalyse

Sinnvoll ist es, für die Parallelanalyse dieselbe *Art* der Faktorenanalyse bzw. die Hauptkomponentenanalyse zu verwenden, die auch für die eigentliche Faktorenanalyse geplant ist (Hautkomponentenanalyse, Hauptfaktoranalyse (siehe Punkt C).

```
#Laden des Pakets und Durchführung Parallelanalyse
require(psych)


fa.parallel(teildaten_efa, fa = "fa")
Parallel analysis suggests that the number of factors =  5  and the number
of components =  NA
```

Die Parallelanalyse wird auch grafisch dargestellt:

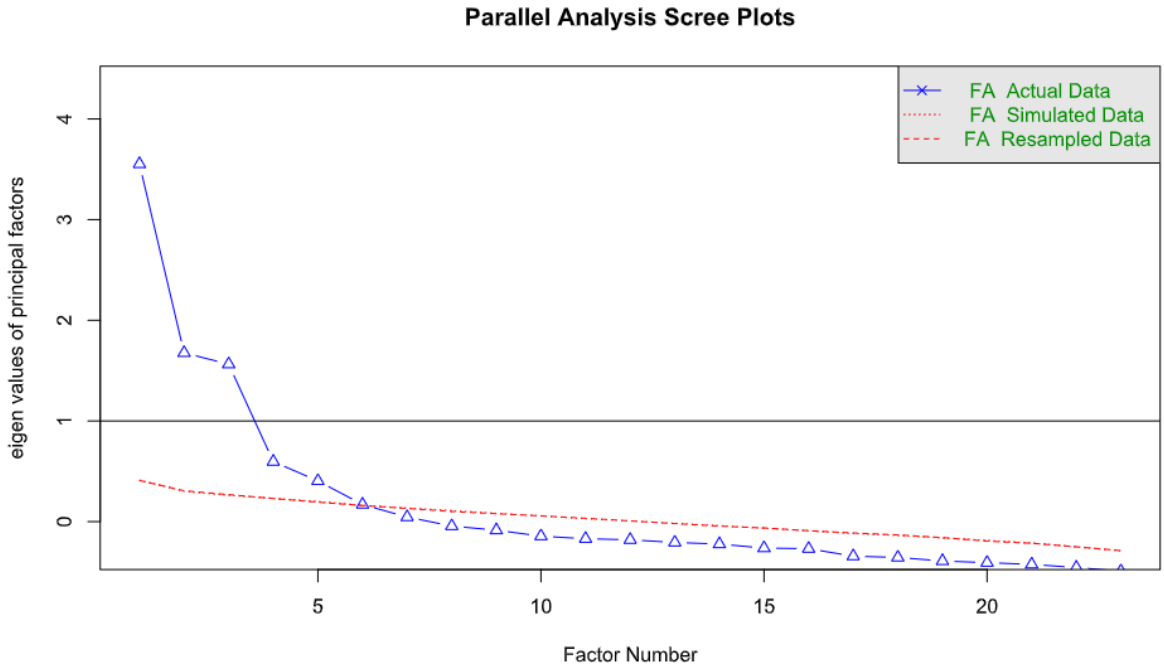


Abb. 47 Screeplot zur Parallelanalyse

In der Grafik zeigt sich der „Knick“ und der Verlauf unter der roten Linie nach 5 Faktoren, entsprechend würde auch nach dieser optischen Kontrolle eine 5-Faktor-Lösung bevorzugt.

Die Durchführung der Parallelanalyse ist in R alternativ auch mit dem Paket *paran* möglich (https://www.r-bloggers.com/2016/04/determining-the-number-of-factors-with-parallel-analysis-in-r/).

c) VSS und MAP

Die VSS-Funktion des Pakets *psych* vergleicht den Fit einer Reihe von Faktoranalysen mit dem der Ladungsmatrix, die vereinfacht wurde indem alle außer de c höchsten Ladungen per item entfernt wurden, wobei c ein Maß der Faktorkomplexität ist (Revelle & Rocklin (1979). Mitausgeben wird das von Bühner (2004) sehr empfohlene MAP-Kriterium (Minimum Absolute Partial correlation) von Velicer.

#Aufruf

```
require(psych) #wenn nicht schon aufgerufen
VSS(teildatensatz)
```

```
Very Simple Structure
Call: vss(x = x, n = n, rotate = rotate, diagonal = diagonal, fm = fm,
    n.obs = n.obs, plot = plot, title = title, use = use, cor = cor)
Although the VSS complexity 1 shows  5  factors, it is probably more
reasonable to think about  1  factors
VSS complexity 2 achieves a maximimum of 0.71  with  5  factors
The Velicer MAP achieves a minimum of 0.02  with  4  factors
BIC achieves a minimum of  -578.81  with  5  factors
Sample Size adjusted BIC achieves a minimum of  -166.41  with  7  factors
```

Nicht untypisch werden hier je nach Verfahren verschiedene Anzahlen an Faktoren favorisiert. R gibt auch eine Grafik dazu aus (hier nicht dargestellt).

Schritt 3: Durchführung der Faktorenanalyse

Nach Abwägung der Kriterien und Festlegung der Zahl der Faktoren kann nun über BSS über die Schaltfläche Analysis > Factor Analysis eine Faktorenanalyse (oder alternativ eine Hauptkomponentenanalyse (PCA)) aufgerufen werden (kann sie natürlich auch schon vorher, in BSS ist ein einfacher Screeplot als auch das Eigenwertekriterium implementiert und kann über das Menü in BSS angefordert werden). Alternativ – was hier nicht demonstriert wird – können über das Syntaxfester auch eine Faktorenanalysen mit anderen Paketen wie dem oben schon verwendeten *psych* durchgeführt werden.

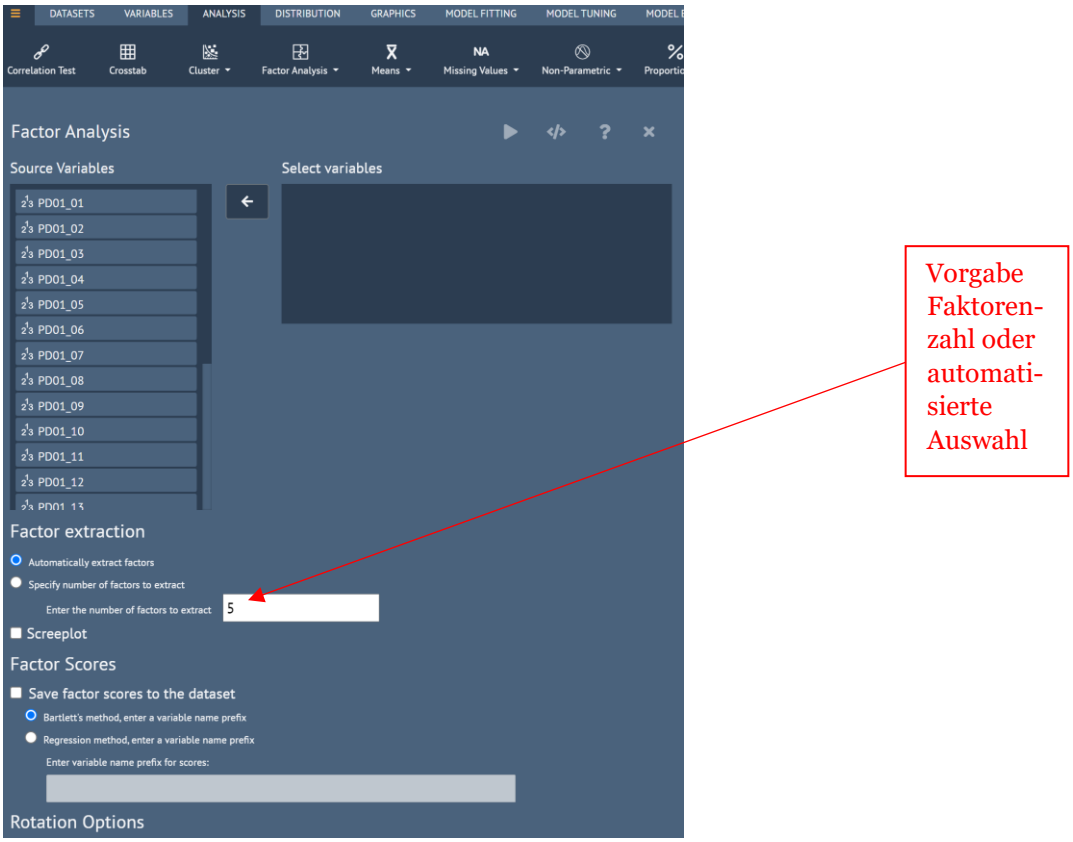


Abb. 48 Aufrufen der Faktorenanalyse in BSS

Spezifiziert werden kann die Rotationsmethode

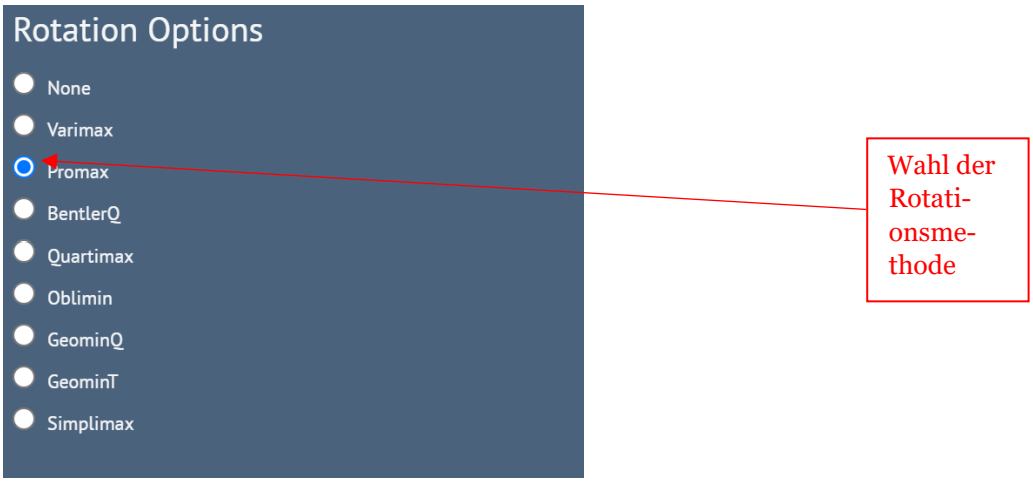


Abb. 49 Menüfenster zur Spezifizierung einer Faktorenanalyse

Ausgewählt werden kann eine Vielzahl von Rotationsmethoden, ein Klick auf die Option Screeplot fordert das entsprechende Diagramm (ohne Parallelanalyse) an.

Die von BSS erzeugte Befehlssyntax ist

```
## [Factor Analysis]
require(GPArotation)
BSkyFARes <- BSkyFactorAnalysis(vars = c("PD01_01", "PD01_02", "PD01_03",
"PD01_04", "PD01_05", "PD01_06", "PD01_07", "PD01_08", "PD01_09",
```

```
"PD01_10", "PD01_11", "PD01_12", "PD01_13", "PD01_14", "PD01_15"), autoex-
traction = FALSE, factors = 5,
    screeplot = FALSE, rotation = "promax", saveScores = FALSE, scores =
"Bartlett",
    prefixForScores = "", dataset = "schuel2021")
```

Display the results in the output grid

```
BSkyFormat(BSkyFARes)
```

In der Ergebnisdarstellung werden zunächst Kommunalitäten, Faktorladungen und Varianzaufklärungen der unrotierten Faktoren ausgegeben , dann die der rotierten. Dargestellt sind hier nur die rotierten

Uniqueness (un-rotated)

PD01_01	PD01_02	PD01_03	PD01_04	PD01_05	PD01_06	PD01_07	PD01_08	PD01_09	PD01_10
0.559	0.513	0.341	0.520	0.546	0.761	0.885	0.613	0.819	0.329

Tab. 24 rotierte Uniqueness/1-Kommunalitäten (Ausschnitt)

Zunächst wird die rotierte Faktormatrix angezeigt:

Rotated Loadings (promax rotation)

	Factor1	Factor2	Factor3	Factor4	Factor5
PD01_01	-0.695		0.147		
PD01_02			0.737		
PD01_03		0.794			
PD01_04			0.680		
PD01_05					0.666
PD01_06				0.134	0.370
PD01_07		0.141		0.256	0.178
PD01_08	0.583				
PD01_09			0.275		0.299
PD01_10		0.816			
PD01_11	-0.610				0.153
PD01_12	0.112				0.571
PD01_13				0.620	
PD01_14	-0.415			0.185	0.117
PD01_15		0.198	0.598		-0.110
PD01_16			0.106	-0.123	0.395

Tab. 25 Faktorladungen

Ausgeben werden zudem die Varianzzerlegung sowie die Korrelation der Faktoren.

Factors (promax rotation)

	Factor1	Factor2	Factor3	Factor4	Factor5
SS loadings	2.352	2.039	1.916	1.412	1.335
Proportion Var	0.102	0.089	0.083	0.061	0.058
Cumulative Var	0.102	0.191	0.274	0.336	0.394

Factor Correlations

	Factor1	Factor2	Factor3	Factor4	Factor5
Factor1	1.000	0.160	0.271	0.354	0.242
Factor2	0.160	1.000	0.340	0.307	0.266
Factor3	0.271	0.340	1.000	0.117	-0.142
Factor4	0.354	0.307	0.117	1.000	0.462
Factor5	0.242	0.266	-0.142	0.462	1.000

Tab. 26 Varianzen der Faktoren und Faktorenkorrelation

Eine ausführlichere Darstellung der Durchführung der Faktoranalyse mit dem R-Paket *psych* von Werner (2014) ist zu finden auf der Internetseite http://www.psychologie.uzh.ch/dam/jcr:ffffffff-852f-247f-ffff-ffff99c06567/explorative_faktorenanalyse_mit_r_cswerner.pdf.

Schritt 4: Testung der Reliabilität

In der Regel wird für eine Skalenbildung nach der erfolgten Faktorenanalyse die Reliabilität der gefundenen Lösung getestet, in der klassischen Testtheorie zumeist über die Ermittlung des Cronbachs alpha. Dies kann in BSS über den Reiter „Agreement > „Scale“ > „Cronbach’s Alpha“ aufgerufen werden:

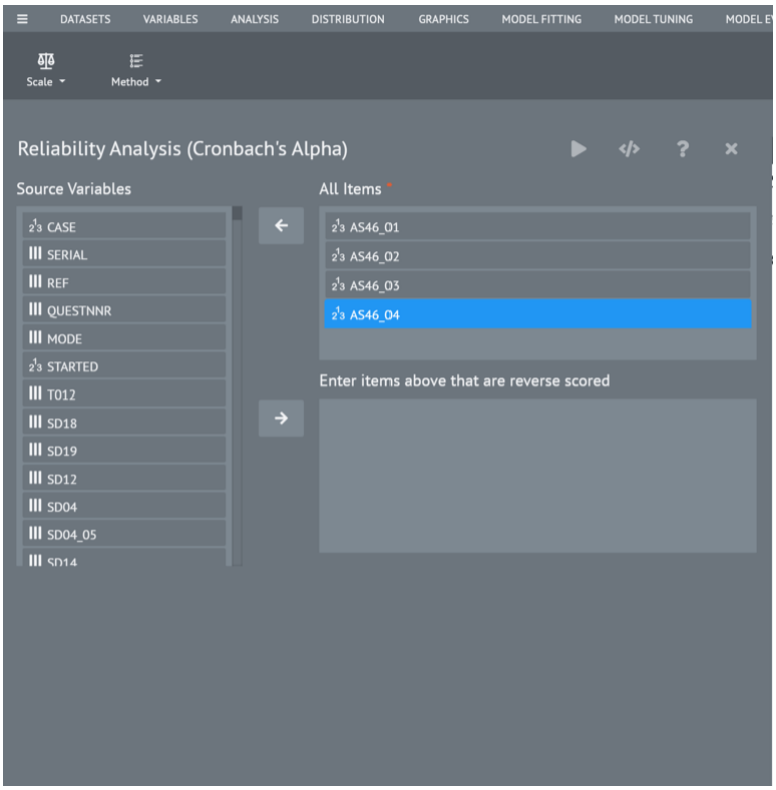


Abb. 50 Menüfenster Durchführung einer Reliabilitätsanalyse

Ein nettes Feature ist das untere Fenster in der Spezifikation (s. Abb. 50), in dem BSS mitgeteilt werden kann, wenn ein Item/mehrere Items „umgekehrt“ ausgerichtet ist/sind, was Rekodierungsarbeit erspart. Im Ausgabefenster zeigt sich eine Reihe von Tabellen, angegeben wird das einfache und das standardisierte Cronbachs alpha, und auch eine Tabelle, die veranschaulicht, wie sich das Cronbachs alpha verändert, wenn einzelne Items weggelassen werden.

Reliability Analysis (Cronbach's Alpha)

Reliability Statistics

Cronbach's Alpha	Cronbach's Standardized Alpha	Guttman's Lambda 6	average_r	S/N	ase	mean	sd	median_r
0.8553	0.8587	0.8304	0.6031	6.0791	0.0199	4.2826	0.7075	0.6074

Reliability if an item is dropped

	Cronbach's Alpha	Cronbach's Standardized Alpha	Guttman's Lambda 6	average_r	S/N	alpha se	var.r	med.r
AS46_01	0.7953	0.8026	0.7357	0.5755	4.0667	0.0299	0.0039	0.5563
AS46_02	0.8115	0.8132	0.7472	0.5920	4.3526	0.0271	0.0026	0.5695
AS46_03	0.8232	0.8279	0.7725	0.6159	4.8112	0.0257	0.0063	0.6501
AS46_04	0.8345	0.8358	0.7768	0.6292	5.0899	0.0243	0.0029	0.6452

Tab. 27 Output zur Reliabilitätsanalyse (Auswahl!)

Alternativ zum Cronbachs alpha kann auch McDonalds Omega über die Option „Agreement> Scale > McDonald’s Omega“ angefordert werden (vgl. zum McDonalds Omega Dunn et al. 2014).

V.6 Clusteranalysen

Auch für Clusteranalyen hält BlueSky Statistics Routinen vor, sowohl für hierarchische Clusteranalysen als auch für Clusteranalysen („Analysis > Cluster Analysis > Hierachical Cluster“) mit dem K-Means-Verfahren („Analysis > Cluster Analysis > K-Means Cluster“) (zur Unterscheidung vgl. bspw. Backhaus et al. 2016, S. 475). BSS nutzt dabei die Funktion hclust bzw. kmeans aus dem R-Basispaket.

V.6.1 Hierarchische Clusteranalysen

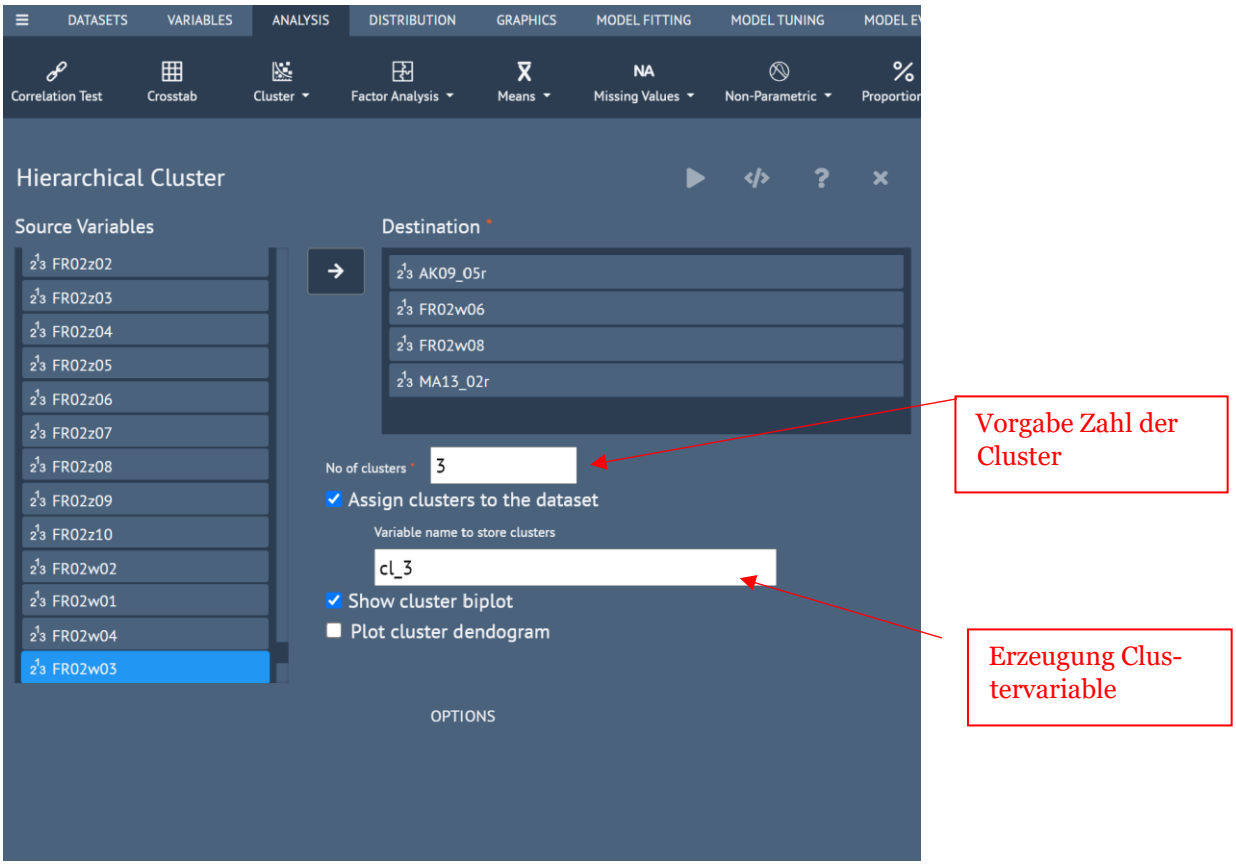


Abb. 51 Menüführung für eine hierarchische Clusteranalyse

In beiden implementierte Routinen muss die Zahl der Cluster vorgegeben werden. Wichtig auch: In diesen klickbaren Routinen ist keine Standardisierung der Variablen vorgesehen, dies sollte bei metrischen Variablen mit unterschiedlichem Skalenniveau über die scale-Funktion (s.o.) unbedingt erfolgen.

Bei der Voreinstellung von 3 Clustern erscheint im Output der hierarchischen Clusteranalyse zunächst die Verteilung der Fälle in die drei Cluster.

Cluster sizes

Cluster 1	Cluster 2	Cluster 3
215	99	90

Cluster centroids

INDICES	FUN			
	FR03z06	FR03z01	FR03z04	FR03z07
1	0.535	0.688	0.800	0.260
2	0.000	0.000	0.000	0.000
3	0.000	0.000	1.000	0.000

Tab. 28 Anzahl Fälle pro Cluster

Je nach Anforderung der Grafik wird das Dentrogramm und ein „Biplot“ ausgeben.

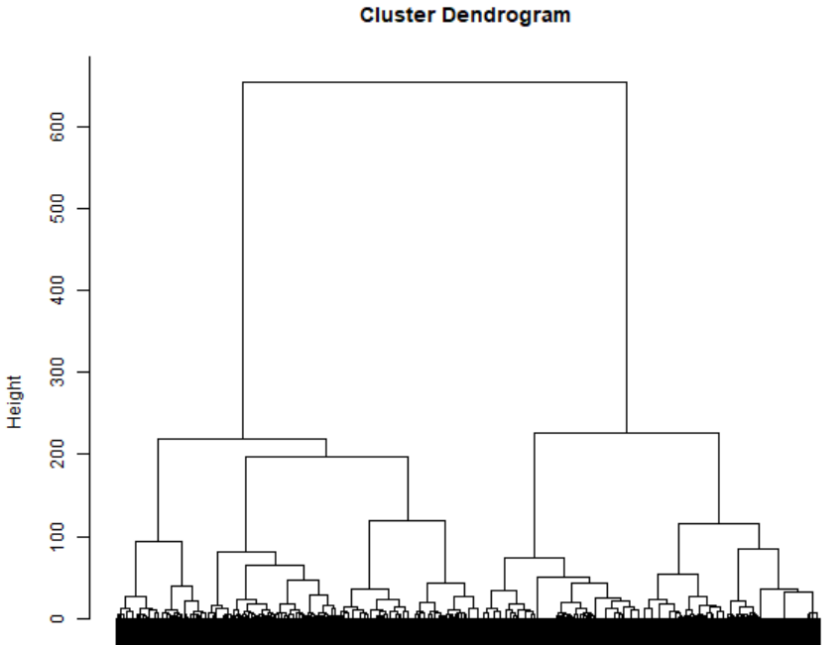


Abb. 52 Dendrogramm einer hierarchischen Clusteranalyse

V.6.2 Clusteranalysen mit K-Means Verfahren

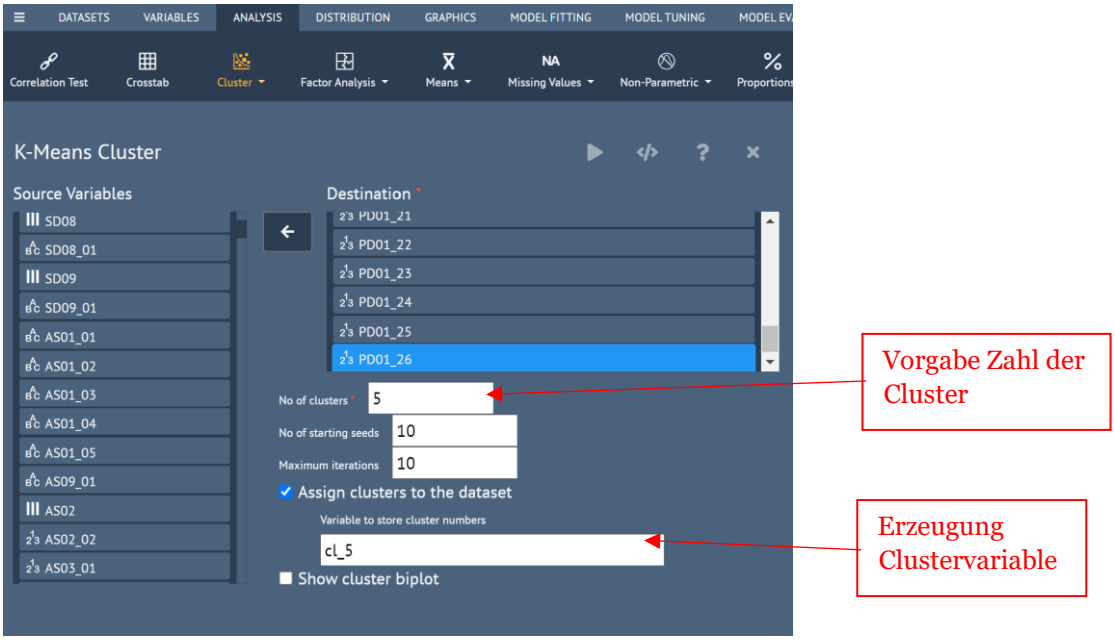


Abb. 53 Menüführung für eine K-Means Clusteranalyse

Gerade bei der Clusteranalyse scheint es sinnvoll, die Analysen mit weiteren Möglichkeiten aus R Paketen wie *cluster* oder *mclust* zu erweitern, die sowohl vielfältige grafische Darstellungsmöglichkeiten als auch statistische Tests (u.a. für eine gap statistics) mitbringen. Auch die Funktion eclust (aus dem Paket *factoextra)* bietet verschiedene Clusteroption inklusive einer gap statistics zur Bestimmung der optimalen Clusterzahl. Das R-Paket *prcr* bietet über eine Prozedur die Möglichkeit einer gestufte Clusteranalyse, die ähnlich wie die Option „Two Step Cluster" bei SPSS in einem Prozess das hierarchische als das k-means Verfahren anwendet und optimale Clusteranzahlen sucht (Näheres unter https://cran.r-project.org/web/packages/prcr/vignettes/introduction_to_prcr.html).

Zum Weiterlesen findet sich eine systematische *und* praktische Einführung mit R-Bezug bspw. unter https://www.inwt-statistics.de/blog-artikel-lesen/Clusteranalyse.html.

Alternative LCA/LPA: Als eine nicht deterministische, sondern probabilistische Herangehensweise sei an dieser Stellen noch die Latente Klassenanalyse (LCA) oder die Latente Profil Analyse (LPA), die nicht in BSS implementiert ist, aber über entsprechende Pakete (LCA: bspw. Paket *poLCA* (s.u.), für ordinale oder kategoriale Variablen; LPA: bspw. Pakete *mclust, tidyLPA*, für metrische Variablen) mit der Syntax durchgeführt werden kann (s. auch VI.2).

VI. Weitere Analysen auf der Basis der R-Syntax

Um exemplarisch die Potentiale anzudeuten, die in der Verwendung von R und seinen Paketen stecken, die weit über die Möglichkeiten des Menüs von BlueSky Statistics hinausgehen, werden im Folgenden noch exemplarische Analysen vorgestellt, die mit entsprechenden R-Paketen über die Syntax von BSS (oder R bzw. RStudio) durchgeführt werden können und die eine Nähe zu den beschriebenen Verfahren haben.

VI.1 Exploratorische Faktorenanalyse für kategoriale Daten

In Ergänzung zur oben dargestellten explorativen Faktorenanalyse soll im Folgenden kurz die Durchführung einer Faktoranalyse mit kategorialen Daten vorgestellt werden, die nicht mit den Rohdaten, sondern mit einer Korrelationsmatrix erfolgt. Die nachfolgenden Schritte sind orientiert an der Darstellung von Hosoya (2015).

```
require (psych)
require (polycor)  # für die polychorische Korrelationsmatrix (vgl. Fox 2010)
require (GPArotation)
```

Geschätzt werden muss zunächst eine polychorische Korrelationsmatrix der Items im Teildatensatz (hier: daten02_t).

```
poly_cor <- polychoric(daten02_t)
```

Mit dem Befehl `print(poly_cor)` werden bei die Korrelationsmatrix der Variablen und geschätzten Schwellenparameter angezeigt. Die exploratorische Faktorenanalyse wird dann auf Basis der erzeugten Korrelationsmatrix (`poly_cor$roh`) aufgerufen, wobei die Zahl der Fälle „mit der Hand“ angegeben werden muss. Vorgegeben wird in diesem Beispiel die Zahl von 3 Faktoren. Gewählt ist hier zudem die Rotationsmethode „oblimin“.

```
fa_b02 <-fa(r=poly_cor$rho, nfactors=3, n.obs=1273, rotate="oblimin")
```

Standardmäßig ist für die Schätzung die Option Methode der Residuenminimierung (minres) voreingestellt. Die Ausgabe erfolgt mit `print(fa_bo2)`. Im Outputfenster werden die standardisierten Ladungen der Items auf den Faktoren (MR1, MR2), die Kommunalitäten (h2) und Fehlervarianzen ausgegeben.

```
Factor Analysis using method =  minres
Call: fa(r = poly_cor$rho, nfactors = 3, n.obs = 1273, rotate = "oblimin")
Standardized loadings (pattern matrix) based upon correlation matrix
           MR2   MR1   MR3   h2    u2 com
B002_01r  0.54 -0.02 -0.35 0.51 0.486 1.7
B002_02r -0.05 -0.01  0.90 0.84 0.160 1.0
B002_03r -0.26  0.54 -0.01 0.29 0.709 1.4
B002_04r  0.11 -0.14  0.47 0.26 0.736 1.3
B002_05r  0.07  0.87  0.02 0.78 0.224 1.0
B002_06r  0.06  0.79 -0.01 0.65 0.346 1.0
B002_07r  0.01 -0.05  0.83 0.72 0.284 1.0
B002_08r -0.08  0.60 -0.21 0.47 0.527 1.3
B002_09r  0.84 -0.04 -0.17 0.79 0.210 1.1
B002_10r  0.98  0.04  0.09 0.94 0.059 1.0
B002_11r  0.59  0.14  0.12 0.37 0.633 1.2
```

Tab. 29 Ladungstabelle bei ordinaler Faktorenanalyse

Nach den Ladungen folgen Statistiken zur Varianzaufklärung durch die Faktoren. Die Zeile `SS loadings` gibt die Summe der quadrierten Faktorladungen an, die Zeile `Proportion Var` die Varianzaufklärung durch die Faktoren

```
                        MR2  MR1  MR3
SS loadings            2.43 2.15 2.04
Proportion Var         0.22 0.20 0.19
Cumulative Var         0.22 0.42 0.60
Proportion Explained   0.37 0.32 0.31
Cumulative Proportion  0.37 0.69 1.00
```

Tab. 30 Varianzaufklärung bei der ordinalen Faktorenanalyse

Zudem wird die Korrelationsmatrix der extrahierten Faktoren ausgegeben. Die geschätzten Korrelationen der Faktoren sind hier r=0.25 bzw. r=-0,29 bzw. r=0.39

```
 With factor correlations of
      MR2   MR1   MR3
MR2  1.00  0.25 -0.29
MR1  0.25  1.00 -0.39
MR3 -0.29 -0.39  1.00
```

Tab. 31 Korrelation der Faktoren

VI.2 Konfirmatorische Faktoranalyse

Konfirmatorische Faktorenanalysen dienen im Gegensatz zur exploratorischen Faktorenanalyse zur Prüfung einer angenommenen Faktorstruktur (vgl. Backhaus et al. 2016, S. 589ff.). Im eigenen Ausblick kündigen die Verantwortlichen von BSS an, dass in zukünftigen Versionen von BSS auch Strukturgleichungsmodelle über das Paket *lavaan* implementiert werden (Coming soon im Menüfenster, https://www.BlueSkyStatistics.com/Articles.asp?ID=298). Aktuell muss – wenn entsprechende Analyse mit dem Paket *lavaan* geplant sind – das Paket installiert und die entsprechende Syntax über das Syntaxfenster eingegeben werden. *Lavaan* eignet sich, um konfirmatorische Faktorenanalysen mit einem oder mehreren Faktoren durchzuführen oder aber auch Pfadanalysen, Strukturgleichungsmodelle oder auch latente Wachstumskurvenmodelle durchzuführen (vgl. Rosseel 2020).

Ein Modell mit zwei latenten Faktoren wird wie folgt spezifiziert (natürlich kann auch nur ein latenter Faktor spezifiziert werden):

```
#Laden des Pakets
require(lavaan)

# Das Model formulieren
model1 <- 'f1 =~ B002_09 + B002_10 + B002_11
'
```

(wichtig ist das einfache Anführungszeichen am Ende und am Anfang!)

#faktor1 ist keine erhobene Variable, sondern werden als latente Variablen durch die erhobenen Variablen B002_09, B002_10 und B002_11 „konstruiert“. Die Namen der latenten Variablen sind frei wählbar).

```
# Das Modell ausführen
fit <- cfa(model1, data=datensatz2)

# Anzeige des Outputs
summary(fit, fit.measures=TRUE)
```

Über das Paket *lavaan* sind auch Strukturgleichungsmodelle oder latente Wachstumskurvenmodelle spezifizierbar. Einführungen in das Paket, ausführliche Dokumentation sowie Anwendungsbeispiele finden sich auf der Homepage https://lavaan.ugent.be/. Alternativ sei auch die Einführung von Werner (2015, https://www.psychologie.uzh.ch/dam/jcr:ffffffff-b371-2797-ffff-ffffeb61aa16/einfuehrung_lavaan_cswerner.pdf) empfohlen.

Mit dem Paket lavaanPlot lässt sich dieses auch graphisch darstellen.

```
lavaanPlot(model = fit, edge_options = list(color = "grey"))
```

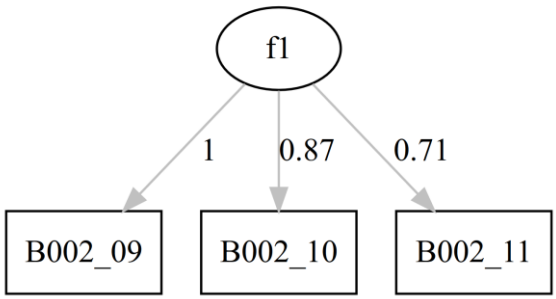


Abb. 54 Graphische Darstellung einer konfirmatorischen Faktorenanalyse mit dem lavaanPlot

VI.3 Latente Klassenanalyse (LCA) & latente Profilanalyse (LPA)

Inhaltlich verwandt zur Clusteranalyse können Cluster oder Klassen in einem Datensatz auch anhand eines statistischen Modells mithilfe der *latenten Klassenanalyse* oder der *latenten Profilanalyse* generiert werden (die beide das Ziel haben, aus einer Zahl an Fällen „Typen" oder Gruppen „ähnlicher" Fälle zu identifizieren) (vgl. als Einführung bspw. Bacher & Vermunt 2010; Rost 2004). Für nominale oder ordinale Variablen wird eine latente Klassenanalyse (LCA) durchgeführt – wie immer gibt es hierfür in R verschiedene Pakete. Bspw. kann das Paket *poLCA* verwendet werden (vgl. Linzer & Lewis 2011). Zunächst muss auch hier wieder das Paket installiert und dann aufgerufen werden. Im Folgenden findet sich ein Syntax-Beispiel, wie eine LCA mit 2, eine mit 3 und eine mit 4 latenten Klassen spezifiziert wird.

Ausgegangen wird von einem Datensatz mit Namen „datensatz03".

```
#Laden des Pakets
require(poLCA)

#Fehlende Werte löschen (durch Erstellung eines reduzierten Datensatzes)
daten03 <-na.omit(datensatz03)
#Variablen für die LCA zusammenstellen (ordinale und nominale Skalen möglich, Skalierung der Variablen muss in jedem Fall mit 1 beginnen)
f <- cbind(z01a,z01b,z01c,z01d, z01e,z01f,z01g)~1

# LCA durchführen (hier für 2, für 3 und für 4 Klassen, Modelle können verglichen werden)
z01.lca2 <- poLCA(f, daten03, graphs=TRUE, nclass=2,maxiter=5000,nrep=40)
z01.lca3 <- poLCA(f, daten03, graphs=TRUE, nclass=3,maxiter=5000,nrep=40)
z01.lca4 <- poLCA(f, daten03, graphs=TRUE, nclass=4,maxiter=5000,nrep=40)
```

Es folgt für jede Analyse – die bei mehreren Klassen auch länger dauern kann - jeweils ein ausführlicher statistischer Output, der zwischen den Modellen verglichen werden kann. Hier am Beispiel für 2 Klassen (nur das Ende des Outputs):

```
(…)

Estimated class population shares
 0.7641 0.2359

Predicted class memberships (by modal posterior prob.)
 0.8153 0.1847

=========================================================
Fit for 2 latent classes:
=========================================================
number of observations: 839
number of estimated parameters: 57
residual degrees of freedom: 782
maximum log-likelihood: -5684.162

AIC(2): 11482.32
BIC(2): 11752.06
G^2(2): 2151.928 (Likelihood ratio/deviance statistic)
X^2(2): 212948.4 (Chi-square goodness of fit)
```

Tab. 32 Modellfitkriterien für die LCA

Insbesondere die Informationskriterien AIC und BIC (niedrigster Wert) werden zur Auswahl der bestmöglichen Lösung herangezogen. Für jede Lösung wird auch eine Grafische Darstellung der Klassen über die Ausprägung über die konstituierenden Variablen ausgegeben, die die Klassen inhaltlich beschreibbar machen.

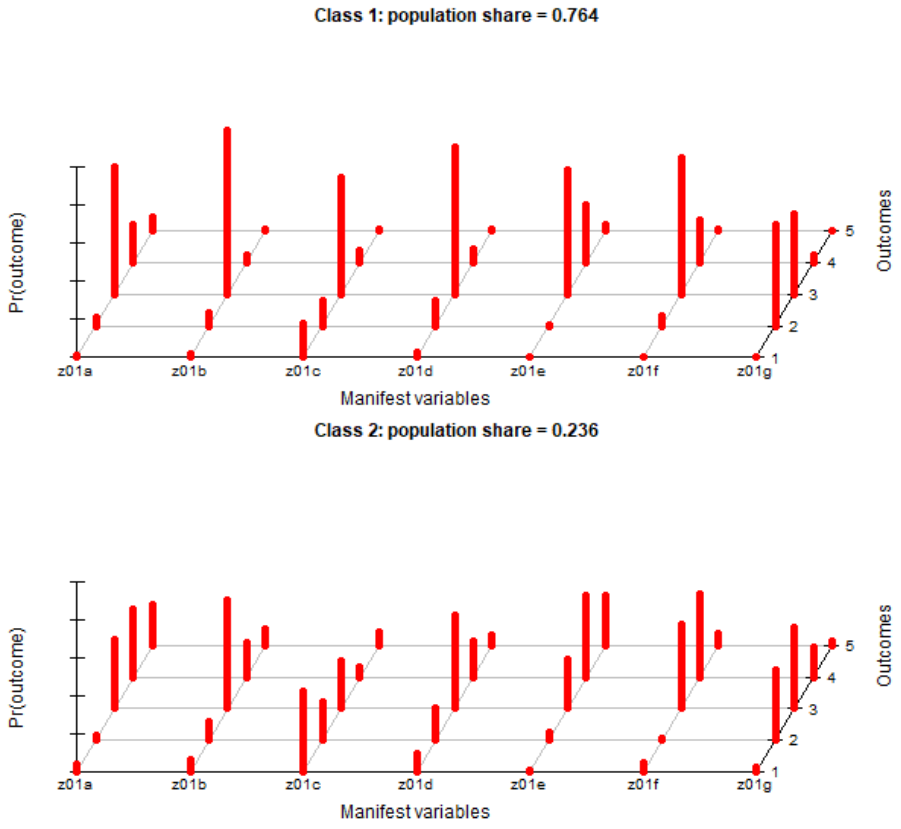


Abb. 55 Graphische Darstellung der Klassen über die Ausprägungen auf den für die LCA verwendeten Variablen

Eine ausführliche Beschreibung sowie Anwendungsbeispiele finden sich u.a. bei den Paketautoren Linzer & Lewis (2011).

Die *latente Profilanalyse* entspricht dem Vorgehen der LCA, nur dass die zugrundeliegenden Variablen metrisch skaliert sind. Ein möglicher Weg zur Durchführung ist das Paket *tidyLCA*, das in Verbindung mit dem Paket *mclust* die Analyse ermöglicht. Auch hierzu folgt ein kurzes Syntaxbeispiel mit dem fiktiven Datensatz datensatz02 und vier metrischen Variablen var1 bis var 4. Auch hier könnten die Variablen vorab skaliert werden, dies kann mit *tidyLPA* aber auch in der Syntax mit der „scale()"-Option erfolgen.

```
require(tidyLPA)
```

Für die Modellbildung werden hier beispielhaft 3 Klassen/Profile vorgegeben.

```
m3 <- datensatz02 %>%
  select(s_ID18_03, s_MA05_02, s_MA13_S3, s_sk_kommu) %>%
  single_imputation()  %>%
  scale() %>%
  estimate_profiles(3)
m3
```

Mit dem Befehl `plot_profiles(m3)` kann auch hier eine grafische Darstellung der Profile („Klassen") angefordert werden, die die Mittelwerte und Standardabweichungen der Klasen auf die die Klassen konstituierenden Variablen anzeigt.

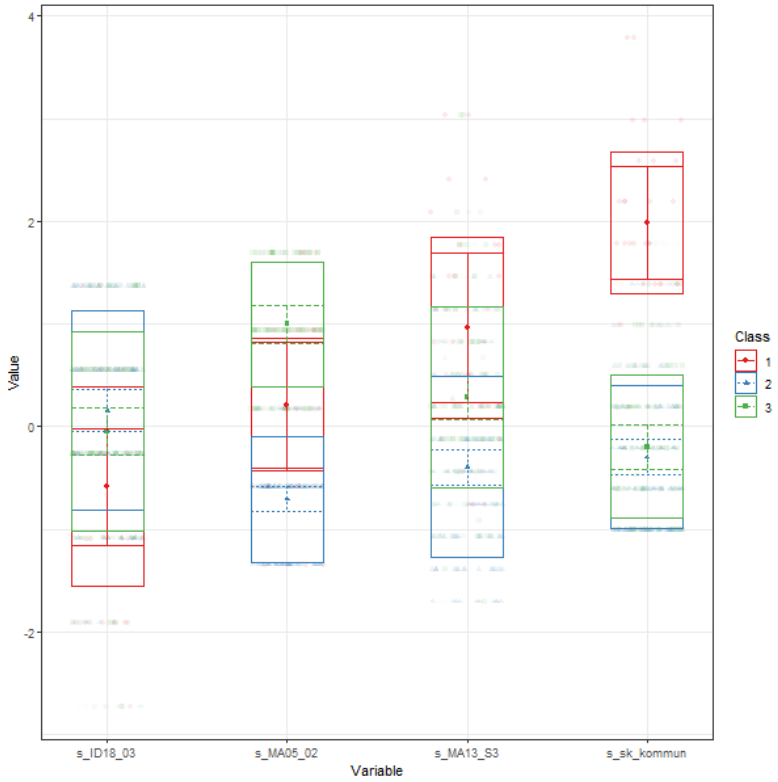


Abb. 56 Graphische Darstellung der Klassen über die Ausprägungen auf den für die LPA verwendeten Variablen

Dies kann wiederum mit der Vorgabe unterschiedlicher Zahlen an Profilen durchgeführt werden, über das niedrigste AIC und BIC lässt sich eine „beste" Lösung identifizieren.

Das Paket *tidyLPA* bietet auch die Möglichkeit, gleichzeitig mehrere LPA mit verschiedener Anzahl Klassen (`n_profiles=`) und unterschiedlicher Spezifizierung von Varianzen und Kovarianzen (`models=`) durchzuführen.

```
datensatz02 %>%
  subset(select = c("var1", " var2", " var3", " var4")) %>%
  single_imputation()  %>%
  scale() %>%
  estimate_profiles(n_profiles = 1:6,models = 1:3)
```

In der Ausgabe werden dann alle Modelle mit allen vorgegebenen Profilanzahlen dargestellt, auch hier wird entsprechend der niedrigsten AIC und BIC die „beste“ Lösung identifiziert.

```
tidyLPA analysis using mclust:
```

Model	Classes	AIC	BIC	Entropy	prob_min	prob_max	n_min	n_max	BLRT_p
1	1	2906.63	2934.96	1.00	1.00	1.00	1.00	1.00	
1	2	2836.63	2882.67	0.80	0.93	0.94	0.43	0.57	0.01
1	3	2788.16	2851.90	0.80	0.87	0.93	0.11	0.53	0.01
1	4	2763.65	2845.10	0.84	0.81	0.93	0.06	0.49	0.01
1	5	2764.83	2863.99	0.83	0.76	0.94	0.03	0.47	0.24
1	6	2749.48	2866.34	0.81	0.72	0.98	0.04	0.38	0.02
(...)									
3	1	2846.40	2895.97	1.00	1.00	1.00	1.00	1.00	
3	2	2807.06	2874.34	0.79	0.90	0.94	0.41	0.59	0.01
3	3	2781.14	2866.13	0.79	0.87	0.94	0.13	0.56	0.01
3	4	2778.90	2881.60	0.82	0.77	0.94	0.03	0.56	0.08
3	5	2755.25	2875.66	0.88	0.66	0.97	0.02	0.56	0.01
3	6	2758.05	2896.16	0.88	0.66	0.97	0.01	0.56	0.38

Tab. 33 Modellvergleich bei der LPA

Eine hilfreiche Einführung und Darstellung findet sich auch in diesem Fall über die Autor:innen des Pakets (Rosenberg et al. 2018), bspw. unter https://data-edu.github.io/tidyLPA/articles/Introduction_to_tidyLPA.html.

VI.4 IRT-Modelle

Mit der Item-Response-Theorie (IRT) werden modellhaft über „gemessene“ (dichotome oder ordinal skalierten) Variablen „zugrundeliegende“ latente Konstrukte identifiziert. Ausgehend von bestimmten Antwortmustern auf Einstellungsfragen oder auch der richtigen oder falschen Beantwortung von Testfragen wird auf Fähigkeiten/Kompetenzen bzw. Persönlichkeitsmerkmale der befragten Personen geschlossen (vgl. als Einführung in die IRT bspw. Geiser 2010).

BlueSky Statistics hatte in Version 7 hierfür Routinen implementiert, diese finden sich in der Version 10 (noch) nicht wieder.

Für eine syntaxbasierte Anforderung kann bspw. auf die Pakete *eRm* und *ltm* zurückgegriffen werden, womit ein Teil der etablierten IRT-Modelle durchführbar sind. Damit werden möglich a) das einfache Rasch-Modell für dichotome Variablen, b) das Partial-Credit-Modell, das im Prinzip ein Rasch-Modell für ordinale Daten darstellt, sowie c) das Rating-Scale-Modell, ebenfalls für ordinale Daten. Das Rating-Scale-Modell unterscheidet sich vom Partial Credit Modell vor allem dadurch, dass im Rating-Scale-Modell alle Items dieselbe Skalierung aufweisen,

während im Partial Credit Modell jedes Item seine eigene Rating-Skalen Struktur hat (wie bspw. bei multiple-choice tests, bei denen Antworten nicht nur falsch oder nur richtig sind, sondern auch in nicht ganz korrekten Antworten etwas Wissen/Fähigkeit ausgedrückt wird, also „partial credit“ für eine nicht ganz korrekte Antwort gegeben wird. Die Menge der “partial correctness“ variiert dabei zwischen den Items.

Zur Durchführung: Es wird ein Teildatensatz mit den heranzuziehenden Variablen benötigt, über Subset funktion (s.o.) bzw. die

```
# Creates the subsetted dataset
dat1 <- schuel2021 %>%
    dplyr::select(item01,  item02,  item03,  item04,  item05,  item06,
item07)
# Refreshes the subsetted dataset in the data grid
BSkyLoadRefresh("dat1")
```

Das einfachste Modell ist nun das eindimensionale Raschmodell. Das Rasch-Modell basiert auf der Annahme, dass alle Items eines Tests/einer Skala dieselbe latente Dimension (z. B. Fähigkeit, Einstellung, Persönlichkeitseigenschaft) mit unterschiedlichen „Itemschwierigkeiten“, aber jeweils gleicher Trennschärfe erfassen (vgl. Geiser 2010, S. 305).

Das klassische Raschmodell (auch 1-pl-Modell) genannt wird im Paket eRm aufgerufen über die Syntax

```
eRM_rm_1 <- RM(dat3)
eRM_rm_1                    # Short summary of item parameters
summary(eRM_rm_1)           # Longer summary of item parameters
betas <- -coef(eRM_rm_1)      # Item difficulty parameters
round(sort(betas), 2)
plotjointICC(eRM_rm_1, item.subset =  1:11, cex = .6)
```

Das Skalenniveau der Variablen muss hier dichotom (0/1) sein. Ausgegeben wird die Item Schwierigkeit in zwei Tabellen:

```
summary(eRM_rm_1)  # Longer summary of item parameters
Results of RM estimation:
Call:  RM(X = dat3)
Conditional log-likelihood: -1260.938
Number of iterations: 20
Number of parameters: 12
Item (Category) Difficulty Parameters (eta): with 0.95 CI:
                         Estimate Std. Error lower CI upper CI
Infos_Inet                  2.998      0.473    2.071    3.926
Infos_beurteilen           -0.767      0.134   -1.031   -0.504
Online_kommunizieren       -0.832      0.133   -1.093   -0.571
Com_nutzen                  1.016      0.209    0.606    1.426
Mail_schreiben              0.273      0.167   -0.054    0.600
Smartphone_nutzen           2.578      0.391    1.811    3.344
Datenschutzeinstellungen   -1.702      0.123   -1.943   -1.461
Autor_Inetinfos            -2.473      0.122   -2.713   -2.233
Nutzen_digiMedien          -0.009      0.155   -0.314    0.295
Chatequette                 0.389      0.172    0.052    0.726
Rechtliches                -1.677      0.123   -1.918   -1.436
Inet_verstehen              0.330      0.169   -0.002    0.662
```

```
                          Estimate Std. Error lower CI upper CI
beta Digi_Dokumente          0.124      0.151   -0.173    0.421
beta Infos_Inet             -2.998      0.473   -3.926   -2.071
beta Infos_beurteilen        0.767      0.134    0.504    1.031
beta Online_kommunizieren    0.832      0.133    0.571    1.093
beta Com_nutzen             -1.016      0.209   -1.426   -0.606
beta Mail_schreiben         -0.273      0.167   -0.600    0.054
beta Smartphone_nutzen      -2.578      0.391   -3.344   -1.811
beta Datenschutzeinstellungen  1.702    0.123    1.461    1.943
beta Autor_Inetinfos         2.473      0.122    2.233    2.713
beta Nutzen_digiMedien       0.009      0.155   -0.295    0.314
beta Chatequette            -0.389      0.172   -0.726   -0.052
beta Rechtliches             1.677      0.123    1.436    1.918
beta Inet_verstehen         -0.330      0.169   -0.662    0.002
```

Tab. 34 Item Difficulty Paramenter und Item Easiness Parameter

Über den Befehl

```
plotjointICC(eRM_rm_1, item.subset =  1:7, cex = .6)
```

lässt sich in einer Grafik die verschiedenen Kurven für die Itemschwierigkeiten ausgeben (vgl. Abb. 54).

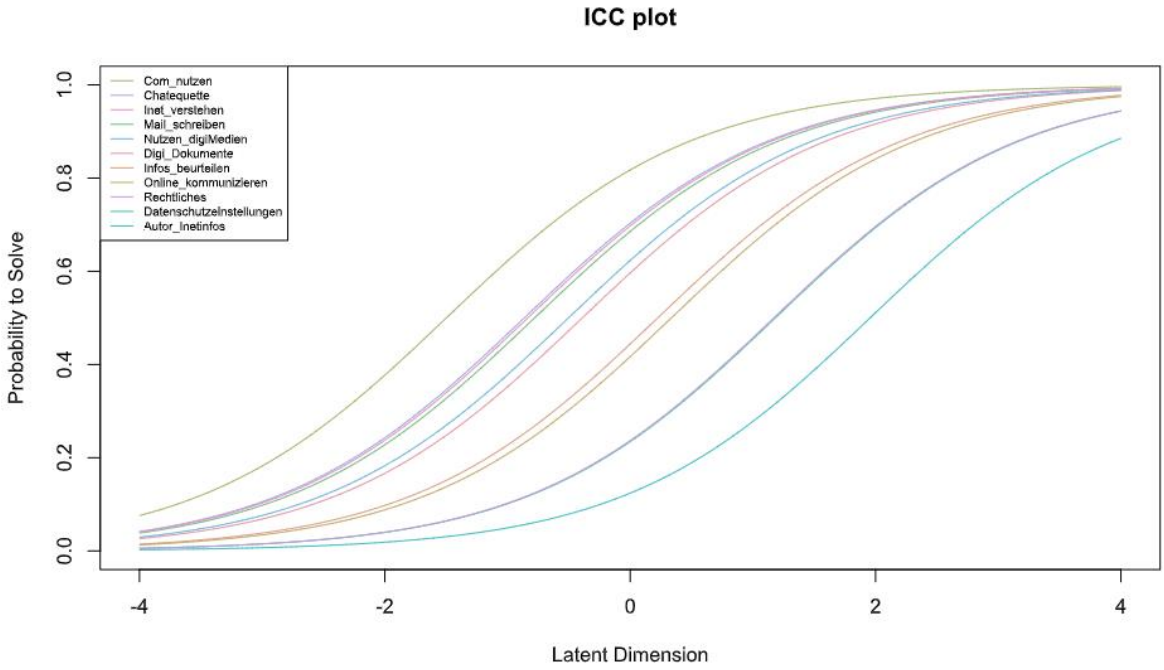


Abb. 57 Itemschwierigeiten der elf Items eines Raschmodell

Ein Test auf Eindimensionalität – die die Grundvoraussetzung für das Raschmodell darstellt – ist im Paket *eRm* implementiert.

```
# Andersen Test auf Eindimensionalität (sollte nicht signifikant werden)
# LR-test des dichotomen Rasch model mit Benutzerdefinierter Aufspittung (Hier Hälfte)
splitvec <- sample(1:2, 686, replace = TRUE) #aufteilen des Datens
lrt_1 <- LRtest(eRM_rm_1, splitcr = splitvec)
lrt_1
# LR-test des dichotomen Rasch model mit Benutzerdefinierter Aufspittung (Hier Median des scors)
LR_median <- LRtest(eRM_rm_1, splitcr = "median")
LR_median
```

Mit dem Waldtest können Items aufgespürt werden, die möglicherweise eine Eindimensionalität verhindern.

```
Waldtest(eRM_rm_1, splitcr = "median")
```

Im Paket ltm wird das Raschmodell geschätzt über

```
library(ltm)
ltm_rm1 <- rasch(dat3)
ltm_rm1
```

Ein 2-pl-Modell und ein 3-pl-Modell mit Rateanteil für dichotome Variablen kann über das Hinzuziehen eines weiteren R-Pakets (bspw. *ltm*) spezifiziert werden. Dabei wird im sogenannten „Birnbaum-Modell“ wird die Annahme homogener Trennschärfen aller Items aufgelöst und ein zweiter Itemparameter eingeführt (Diskriminations- oder Trennschärfeparameter) (vgl. Geiser 2010). Entsprechend wird dieses Modell auch als 2-parametriges logistisches Modell (2-parameter logistic model) bezeichnet.

```
library(ltm)
res_2pl_1 <- ltm(dat3 ~ z1)
res_2pl_1
plot
```

Ein Vergleich der Modelle kann angefordert werden über

```
anova(ltm_rm1, res_2pl_1)
```

Eine Erweiterung erfährt wiederum das Birnbaum-Modell durch das Hinzufügen eines Rate-Parameters (3-parametriges logistisches Modell).

Eine gut nachvollziehbare Einführung in die Anwendung von IRT-Modelle über die R-Syntax gibt Rieninger (https://hansjoerg.me/2018/04/23/rasch-in-r-tutorial/).

Quellen

Homepage von BlueSky Statistics: https://www.BlueSkyStatistics.com/Default.asp

Akinwande, O., Dikko, H.G. & Agboola, S. (2015). Variance Inflation Factor: As a Condition for the Inclusion of Suppressor Variable(s) in Regression Analysis. *Open Journal of Statistics*, 5(7), 754-767.

Bacher, J. & Vermunt, J.K. (2010). Analyse latenter Klassen. In C. Wolf & H. Best (Hrsg.), Handbuch der sozialwissenschaftlichen Datenanalyse (S. 553-574). Wiesbaden: VS.

Backhaus, K., Erichson, B., Plinke, W. & Weiber, R. (2016). Multivariate Analysemethoden. 14. Aufl., Berlin: Springer.

Bates, D., Mächler, M., Bolker, B. & Walker, S. (2015). Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software,* 67(1), 1–48.

Bühner, M. (2004). Einführung in die Test- und Fragebogenkonstruktion. München: Pearson Studium – Psychologie.

Dunn, T.J., Baguley, T. & Brunsden, V. (2014). From alpha to omega: A practical solution to the pervasive problem of internal consistency estimation. *British Journal of Psychology*, 105(3), 399-412.

Fox, J. (2022). Package 'polycor'. https://cran.r-project.org/web/packages/polycor/polycor.pdf [letzter Zugriff: 5.4.2022].

Große Schlarmann, J. & Galatsch, M. (2014). Regressionsmodelle für ordinale Zielvariablen. Medizinische Informatik, Biometrie und Epidemiologie, 10(1), 1-9.

Geiser, C. (2010). Item-Response-Theorie. In C. Wolf & H. Best (Hrsg.), Handbuch der sozialwissenschaftlichen Datenanalyse (S. 311-332). Wiesbaden: VS.

Hosoya, G. (2015). Einführung in die Analyse testtheoretischer Modelle mit R. Eine praxisorientierte Ergänzung zu dem Buch Testtheorie und Testkonstruktion von Michael Eid und Katharina Schmidt. http://www3.hogrefe.de/fileadmin/redakteure/hogrefe_de/Psychlehrbuchplus/Testtheorie_und_Testkonstruktion/Einfuehrung_Testtheoretische_Modelle_R_neu.pdf [letzter Zugriff: 5.4.2022].

Janssen, J. & Laatz, W. (2017). Statistische Datenanalyse mit SPSS. 9. Aufl., Berlin: Springer.

Lamprianou, I. (2019). Applying the Rasch Model in Social Sciences Using R and BlueSky Statistics. London: CRC Press.

Langer, W. (2010). Mehrebenenanalyse mit Querschnittsdaten. In C. Wolf & H. Best (Hrsg.), Handbuch der sozialwissenschaftlichen Datenanalyse (S. 741-774). Wiesbaden: VS.

Leeper, T.J. (2018). Interpreting Regression Results using Average Marginal Effects with R's margins. https://cran.r-project.org/web/packages/margins/vignettes/TechnicalDetails.pdf [letzter Zugriff: 5.4.2022].

Linzer, D.A. & Lewis, J.B. (2011). poLCA – An R Package for Polytomous Variable Latent Class Analysis. *Journal of Statistical Software* 42 (10). https://www.jstatsoft.org/article/download/v042i10/507 [letzter Zugriff: 5.4.2022].

Luhmann, M. (2015). R für Einsteiger. Einführung in die Statistiksoftware für die Sozialwissenschaften. 4., vollständig überarbeitete Auflage, Weinheim: Beltz.

Muenchen, R.A. (2020a). A Comparative Review of the BlueSky Statistics GUI for R, updated 8/3/2020 http://r4stats.com/articles/software-reviews/bluesky/ [letzter Zugriff: 16.10.2020].
Muenchen, R.A. (2022). BlueSky Statistics 10 User Guide. https://r4stats.com/books/bluesky-statistics-user-guide/ [letzter Zugriff: 15.3.2022],
Navarro, D.J. (2015). Learning statistics with R: A tutorial for psychology students and other beginners. University of Adelaide. Adelaide
Noak, M. (2007). Faktorenanalyse. https://www.uni-due.de/imperia/md/content/soziologie/stein/faktorenanalyse.pdf [letzter Zugriff: 5.4.2022].
Rosenberg J.M., Beymer, P.N., Anderson, D.J., van Lissa, C.J. & Schmidt, J.A. (2018). tidyLPA: An R Package to Easily Carry Out LatentProfile Analysis (LPA) Using Open-Source or Commercial Software. *The Journal of Open Source Software,* 3 (30), 978. https://doi.org/10.21105/joss.00978
Rosseel, Y. (2020). The lavaan tutorial. April 2022. https://lavaan.ugent.be/tutorial/tutorial.pdf [letzter Zugriff: 5.4.2022].
Rost, J. (2004). Lehrbuch der Testtheorie – Testkonstruktion. 2. Aufl., Bern: Huber.
Thieme (2021). Identifying Outliers in Linear Regression — Cook's Distance. https://towardsdatascience.com/identifying-outliers-in-linear-regression-cooks-distance-9e212e9136a[letzter Zugriff: 5.4.2020].
What Is Cook's Distance and How Can It Help You Identify and Remove Outliers From Your Dataset?
Venables, W.N. & Ripley, B.D. (2002). Modern Applied Statistics with S, Fourth edition. New York: Springer.
Walther, B. (2021). Cook Distanz in R ermitteln und interpretieren – Ausreißer erkennen. https://bjoernwalther.com/cook-distanz-in-r-ermitteln-und-interpretieren-ausreisser-erkennen/[letzter Zugriff: 5.4.2022].
Werner, C. (2014). Explorative Faktorenanalyse: Einführung und Analyse mit R. http://www.psychologie.uzh.ch/dam/jcr:ffffffff-852f-247f-ffff-ffff99c06567/explorative_faktorenanalyse_mit_r_cswerner.pdf.
Werner, C. (2015). Strukturgleichungsmodelle mit R und lavaan analysieren: Kurzeinführung. https://www.psychologie.uzh.ch/dam/jcr:ffffffff-b371-2797-ffff-ffffeb61aa16/einfuehrung_lavaan_cswerner.pdf [letzter Zugriff: 5.4.2020].
Zeileis A. & Hothorn, T. (2002). Diagnostic Checking in Regression Relationships. *R News*, 2(3), 7–10. http://ftp.uni-bayreuth.de/math/statlib/R/CRAN/doc/vignettes/lmtest/lmtest-intro.pdf [letzter Zugriff: 5.4.2022].

Schlusswort

Dieses Paper bleibt Work in Progress – für Hinweise, Kritik und Ergänzungswünsche ist der Autor dankbar.

Kontakt: Prof. Dr. Ivo Züchner, Philipps-Universität Marburg, Fachbereich 21, Institut für Erziehungswissenschaft, e-mail: zuechner@staff.uni-marburg.de

Anhang 1: Ansprechen/Aufrufen von Variablen über den R-Code

In R können Variablen auf verschiedene Weisen angesprochen werden. Der „klassische“ Zugang zu Variablen erfolgt mit der Verbindung von

Name des Datensatzes – Dollarzeichen – Kurzname der Variablen, wie z.B.

```
datensatz1$var1
```

So wird mit dem Befehl

```
table(daten03$geschlw)
```

eine einfache Häufigkeitsausgabe für die Variable „geschlw“ aus dem Datensatz „daten02“ erzeugt. Umfangreichere Befehle haben darüber hinaus oft die Option, erst das „Modell“ zu formulieren in diese Modellformulierung mit dem Zusatz `data=datensatz1` den Datensatz festzulegen wie z.B. `LinMod1 <- lm(skala1 ~ geschlw + alter, data = daten02)`

Anhang 2: Effektstärken über R-Syntax berechnen

In den Menüfenstern von BSS ist nicht immer das Aufrufen von Effektgrößen voreingestellt, dies wurde aber in der Version 10 erweitert. Hier exemplarische Hinweise, wie über die R Syntax eine Ausgabe von Effektstärken zu erreichen ist – wie immer gibt es verschiedene R Pakete, die dieses ermöglichen.

- **Kreuztabellen**: Phi (2*2 Tabelle) oder Cramers V (2*x Tabelle)

#zunächst Paket *DescTools* laden (ist schon mit BlueSky Statistics installiert)

```
require(DescTools)
```

Tabelle mit zwei Variablen speichern

```
table1 <- table(datensatz2$ZP02, datensatz2$ZP04)
```

#Cramers V berechnen

```
CramerV(table1, conf.level=0.95)
```

- **Mittelwertsvergleich von 2 Gruppen**: Cohens d

#zunächst Paket lsr (installieren und) laden

```
require(lsr)
cohensD(mathe ~ teacher, data = daten04)
```

(mathe ist die metrische abhängige Variable, teacher die zweistufige Gruppenvariable)

- **Mittelwertsvergleich von mehr als zwei Gruppen** (aus einer Varianzanalyse): eta²

#zunächst Paket *lsr* (installieren und) laden

```
require(lsr)
```

zunächst Varianzanalyse mit Befehl „aov“ durchführen

```
anova1 <- aov(mathe ~ gruppenvariable, data=daten04)
```

Darstellung der ANOVA-Tabelle

```
summary(anova1)
```

Ausgabe des eta²

```
etaSquared(anova1)
```

Dies sind nur Beispiele, in R gibt es über viele Pakete weitere Möglichkeiten, diese oder andere Effektstärken zu ermitteln.