DeWiss
Delphi-Verfahren in den Gesundheits- und Sozialwissenschaften

DEFLEX
Delphi-Verfahren zur Konsensfindung. Eine methodologische Reflexion.

Einführung in die open-source Statistiksoftware R

Datenanalyse einer Delphi-Studie

Julia Schifano, Jessica Dieudonné, Rebecca Abele und Marlen Niederberger

Forschungsmethoden in der Gesundheitsförderung und Prävention
Institut für Gesundheitswissenschaften
Pädagogische Hochschule Schwäbisch Gmünd

Schwäbisch Gmünd

Oktober 2024

CAMPUS OF METHODS
PH Schwäbisch Gmünd

Inhalt

Kontakt

Julia Schifano (M.Sc.)

Forschungsmethoden in der Gesundheitsförderung und Prävention

Institut für Gesundheitswissenschaften

Pädagogische Hochschule Schwäbisch Gmünd

E-Mail: julia.schifano@ph-gmuend.de

Ziel der Einführung

Die Einführung hat zum Ziel, den Einstieg in die Datenanalyse mit R zu erleichtern. In einer Übung beginnen wir mit den ersten Schritten der Auswertung des Datensatzes einer Online-Umfrage. Die Umfrage ist Teil einer Delphi-Studie. Der Datensatz, den wir in der Übung nutzen, ist öffentlich zugänglich. Was du für das Tutorial brauchst: Zugang zum Internet, einen Computer und etwa 60 Minuten Zeit.

Als Erstes erläutern wir dir aber, was R ist, was die Vorteile der Software sind und wo du Hilfe bekommen kannst.

Was ist R?

- R ist *die* statistische Programmiersprache im Themenfeld Datenanalyse
- Kein „Anklicken möglich“, sondern Eingabe der Befehle in eine Kommandozeile
- R wird typischerweise im Zusammenhang mit RStudio genutzt
- Beispiele für Anwendungsfelder: Biologie/Bioinformatik, Sozial- und Gesundheitswissenschaften, Ökonomie

Was ist RStudio?

- RStudio ist eine integrierte Entwicklungsumgebung und grafische Benutzeroberfläche für R. Der Aufbau gliedert sich in vier Fenster (Abbildung 1).

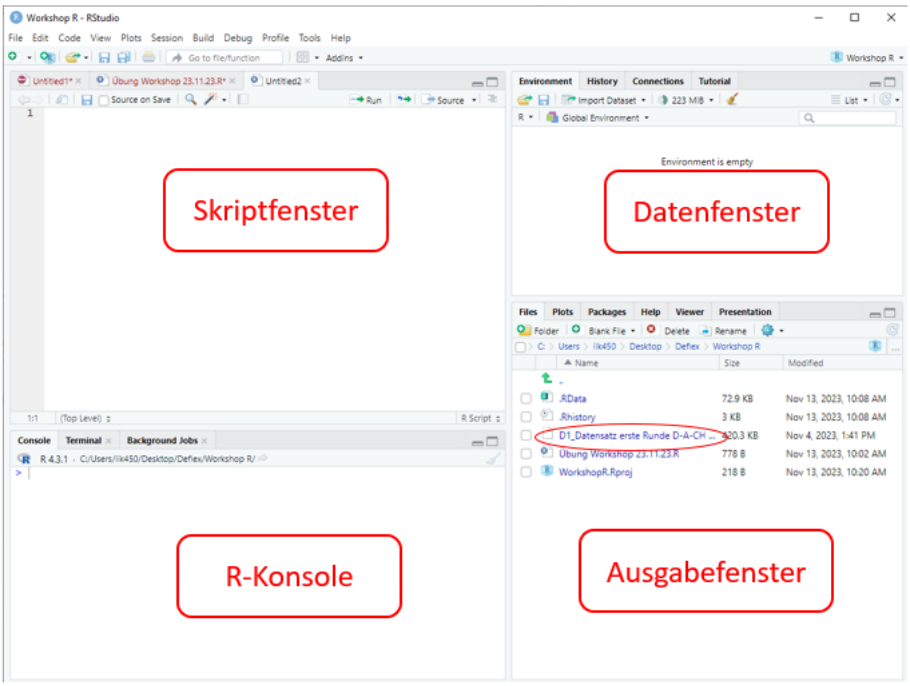

Abbildung 1: Aufbau von RStudio (Screenshot der Version 4.3.1).

Skriptfenster: Im Skriptfenster können wir ein Dokument erstellen, um unseren Code abzuspeichern. Hier können Codes, also Befehle, die der Computer ausführen soll, und Kommentare reingeschrieben werden. Codes führt man aus, indem man sie markiert und „strg“ + Enter drückt. Das Skript speichert man, indem man File > Save as… in der oberen Zeile drückt.
R-Konsole: In der R-Konsole wird der Code angezeigt, den wir gerade ausführen. Das heißt, wenn du den Code, den du im Skriptfenster eingetragen hast, ausführst, wird er dir in der Konsole angezeigt. In der Konsole erscheinen auch Fehlermeldungen, falls du bei einem deiner Codes einen Fehler eingebaut hast. Man kann in der Konsole auch direkt Code eingeben und enter klicken, um diesen auszuführen. Gibst du dort z.B. 1+1 und klickst *enter*, wird dir das Ergebnis angezeigt. Weil der Inhalt in der Konsole durch das Schließen von RStudio gelöscht wird, ist es wichtig, dass du dein Skript immer speicherst.
Datenfenster: Im Datenfenster werden uns Objekte angezeigt, die wir erstellt haben, bspw. einen Datensatz, den wir eingelesen haben. Dazu erfährst du mehr im Laufe des Tutorials.
Ausgabefenster: Das Ausgabefenster zeigt dir unter dem Reiter:

- *Files*, alle Dateien an, die in deinem R-Projekt sind.
- *Plots*, die Abbildungen, die du generiert hast
- *Packages*, alle Packages. Die, die mit einem Haken markiert sind, hast du installiert und geladen.
- *Help*, die Hilfeseiten zu Funktionen der Packages
- *Viewer*, alle anderen Ausgaben, die keine Plots sind, bspw. HTML-Dokumente, Tabellen

Falls du noch mehr über die RStudio Oberfläche lernen möchtest oder dir eine Veranschaulichung der Funktionen in den einzelnen Fenstern helfen würde, schaue dir gerne noch das folgende Video an: https://www.youtube.com/watch?v=IliRHh0VF_k

Wozu R?

- R bietet die Möglichkeit, Daten (statistisch) auszuwerten
- Ständige Weiterentwicklung und Integration neuer Auswertungsverfahren (bspw. text mining)
- Automatisiertes und dokumentiertes Vorgehen möglich
- Große Datenmengen können relativ schnell verarbeitet werden
- Vielfältige Möglichkeiten der grafischen Darstellung
- Kostenlos im Vergleich zur kostenpflichtigen Alternative SPSS

Wo bekomme ich Hilfe?

1. ChatGPT (Kann dir einen ersten Entwurf für einen Code liefern, wenn du beschreibst, was du machen möchtest, dir diesen erklären und Tipps geben, um Fehler zu beheben)
2. Google (Suche ggf. auch auf Englisch, hier gibt es oft mehr Einträge)
3. Hilfe-Seiten und Foren (z.B. https://stackoverflow.com/, CRAN Project, Suche nach Funktion im jeweiligen package)
4. Cheat Sheets zu Packages (z.B. https://geomoer.github.io/moer-base-r/assets/data/cheatsheets/base-r.pdf)
5. Handbücher (z.B. Luhmann, M. (2020). R für Einsteiger. Einführung in die Statistiksoftware für die Sozialwissenschaften. Beltz. ISBN: 9783621287913.)

Wie R installieren?

Falls du Hilfe bei der Installation brauchst, findest du eine Anleitung bspw. hier: https://bjoernwalther.com/r-installieren-windows-und-rstudio-installieren/

Das Delphi-Verfahren

In einer Übung werten wir den Datensatz der ersten Befragungsrunde einer Delphi-Studie aus. Hier findest du eine kurze Erläuterung zum Delphi-Verfahren und Informationen zum Hintergrund der Delphi-Studie, die Teil der nachfolgenden Übung ist.

Definition

In einem Delphi-Verfahren werden Expert*innen zu einem Thema in mehreren Runden befragt (Häder, 2014). Die Runden werden in der Regel als Online-Umfragen durchgeführt.

Ab der zweiten Befragungsrunde erhalten die Expert*innen ein Feedback, welches eine Übersicht der Ergebnisse der vorherigen Befragungsrunde beinhaltet (Häder, 2014, Niederberger & Spranger, 2020).

Die Expert*innen haben die Möglichkeit, ihr Urteil zu ändern. Ziel ist es häufig, herauszufinden, bei welchen Aspekten eines Themas eine hohe Übereinstimmung (Konsens) besteht (Niederberger & Spranger, 2020).

In Delphi-Studien kommen meist deskriptive Auswertungsverfahren zum Einsatz. Die Datenauswertung stellt die Grundlage für die Weiterentwicklung des Fragebogens und für die Feedbackgestaltung dar. Das Feedback beinhaltet häufig Maße der zentralen Tendenz (z.B. Mittelwert), Maße der Streuung (z.B. Standardabweichung) und Häufigkeitsverteilungen der Antworten.

Den Datensatz einer Delphi-Studie auswerten

Im Folgenden erläutern wir zunächst kurz den Hintergrund und das methodische Vorgehen der Delphi-Studie aus der Übung. Weitere Informationen zur Studie findest du in der Publikation von Kauschke et al. (2023).

Hintergrund

- Im deutschsprachigen Raum werden unterschiedliche Definitionskriterien und Begriffe für Sprachentwicklungsstörungen verwendet
- Unterschiedliche Fachbereiche und Disziplinen arbeiten in diesem Themenfeld, z.B. Sprachtherapie/Logopädie, Linguistik, Medizin, (Sonder-) Pädagogik, Neurowissenschaften und Psychologie
- Ziel der Delphi-Studie war es, einen Konsens über Definitionen und Terminologien zu Sprachentwicklungsstörungen im Kindesalter zu erreichen

Methode

- Durchführung einer Delphi-Studie im deutschsprachigen Raum zur Definition und Terminologie von Störungen der Sprache im Kindesalter
- Die Delphi-Befragung fand 2021-2022 statt und umfasste drei Online-Befragungsrunden
- Vertreter*innen der Bereiche Praxis, Forschung und Lehre aller relevanten Disziplinen bewerteten mittels eines standardisierten online Fragebogens Aussagen zur Terminologie und Definition von Sprachstörungen im Kindesalter
- 449 Personen haben an der ersten Delphi-Runde teilgenommen

Die folgenden vier Schritte zeigen dir, wie du bei der Auswertung vorgehen kannst. Voraussetzung für die Übung ist, dass R und RStudio installiert sind. Wir nutzen den Datensatz der ersten Befragungsrunde einer Delphi-Studie (Kauschke et al., 2023).

Vorbereitung

Lege zunächst ein neues Projekt an ( ). In diesem Ordner ist bislang dein R-Projekt: WorkshopR.Rproj. Wenn du die Daten woanders ablegst, musst du den Dateipfad in der Übung anpassen.

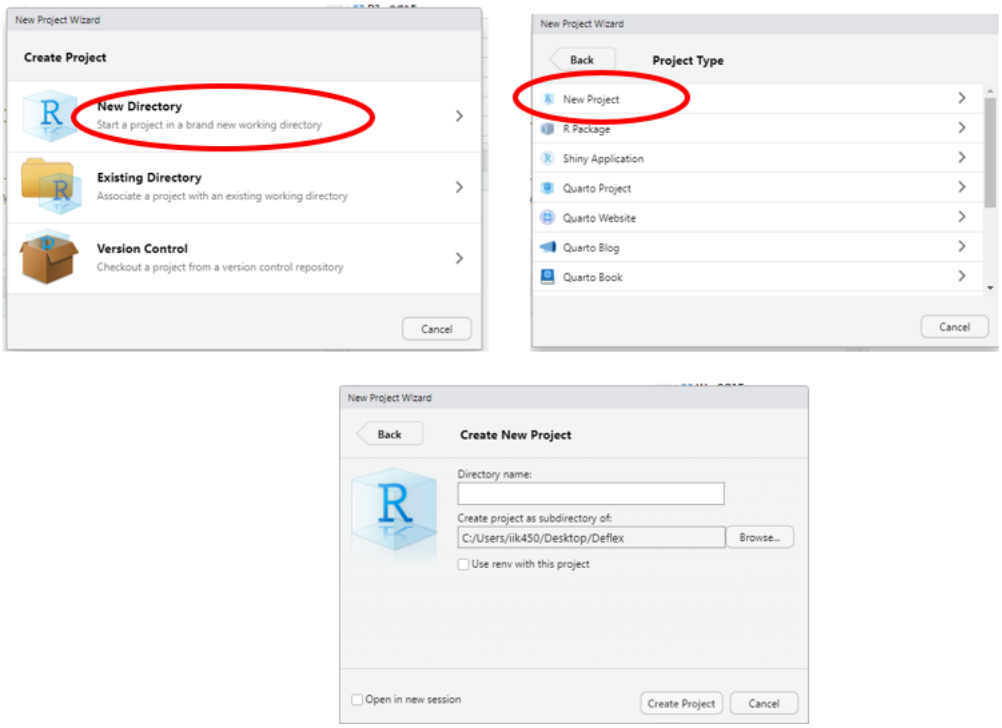

Abbildung 2: Ein Projekt in RStudio anlegen (Screenshots der Version 4.3.1).

Öffne nun ein R-Skript, um dort deinen Code abzuspeichern.

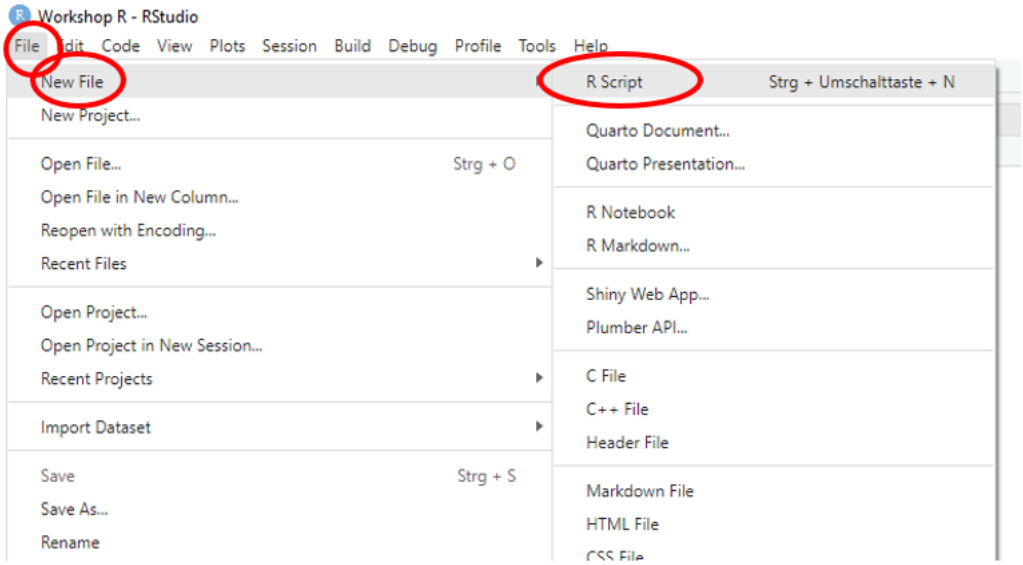

Abbildung 3: Ein R-Skript anlegen in RStudio (Screenshot der Version 4.3.1).

Schritt 1: Packages installieren und laden

Als Erstes musst du die benötigten Packages installieren und laden. R ist eine Community, in welcher jeder sein Wissen teilen und anderen Usern entwickelte „packages“ zur Verfügung stellen kann. Packages sind Zusatzpakete und ein Zusammenschluss von einer oder mehreren Funktionen, die du aufrufen kannst und nicht selbst programmieren musst.

Erläuterung der Packages Im Folgenden erläutern wir kurz die Packages aus diesem Tutorial.

- **dplyr**: enthält Funktionen, um Daten zu manipulieren, bspw. Variablen auswählen, umwandeln und ordnen (Wickham et al., 2023)
- **ggplot2**: enthält Funktionen, um Daten graphisch zu veranschaulichen (Wickham, 2016)
- **summarytools**: enthält Funktionen, um eine Übersicht deskriptiver statistischer Maße von einem Datensatz zu generieren (Comtois, 2022)
- **psych**: enthält Funktionen, um multivariate Analysen durchzuführen (Revelle, 2023)
- **haven**: enthält Funktionen, um SPSS-Datensätze in andere Dateiformate umzuformatieren (Wickham, Miller, & Smith, 2023)
- **rmarkdown**: enthält Funktionen, um dynamische Dokumente, wie bspw. dieses Tutorial, zu generieren (Allaire et al., 2024)

Kopiere hierzu den nachfolgenden Code in dein R Skript. Achte darauf, dass der Code von der Formatierung wie abgebildet eingefügt wurde. Anschließend markierst du den Code mit deiner Maus und drückst auf deiner Tastatur gleichzeitig “strg” und “enter”. Damit führst du den Code aus. Was der Computer macht, wird dir in der Konsole angezeigt.

```
#installieren (muss nur in der ersten R-Session erfolgen)

install.packages("dplyr")
install.packages("ggplot2")
install.packages("summarytools")
install.packages("psych")
install.packages("haven")
install.packages("rmarkdown")

#laden (muss jede R-Session durchgeführt werden)

library("ggplot2")
library("dplyr")
library("summarytools")
library("psych")
library("haven")
library("rmarkdown")
```

Erläuterung zum Code Das „#“ Zeichen kennzeichnet einen Kommentar. Kommentare sind Text im Code ohne Funktion. Sie strukturieren den Code und erläutern diesen. So weißt du auch in einigen Wochen noch, was genau du mit dem Code programmiert hast und auch andere können deine Schritte besser nachvollziehen. **Diesen Schritt, strg + enter klicken, musst du auch nachfolgend durchführen, um deinen Code auszuführen.**

Schritt 2: Daten einlesen und formatieren

Lade den Datensatz der ersten Delphi-Runde von Kauschke et al. (2023) runter (https://osf.io/mgpue). Lasse dir die Datei in deinen Downloads anzeigen und lege die Datei dann im Ordner deines R-Projekts ab (Abbildung 4).

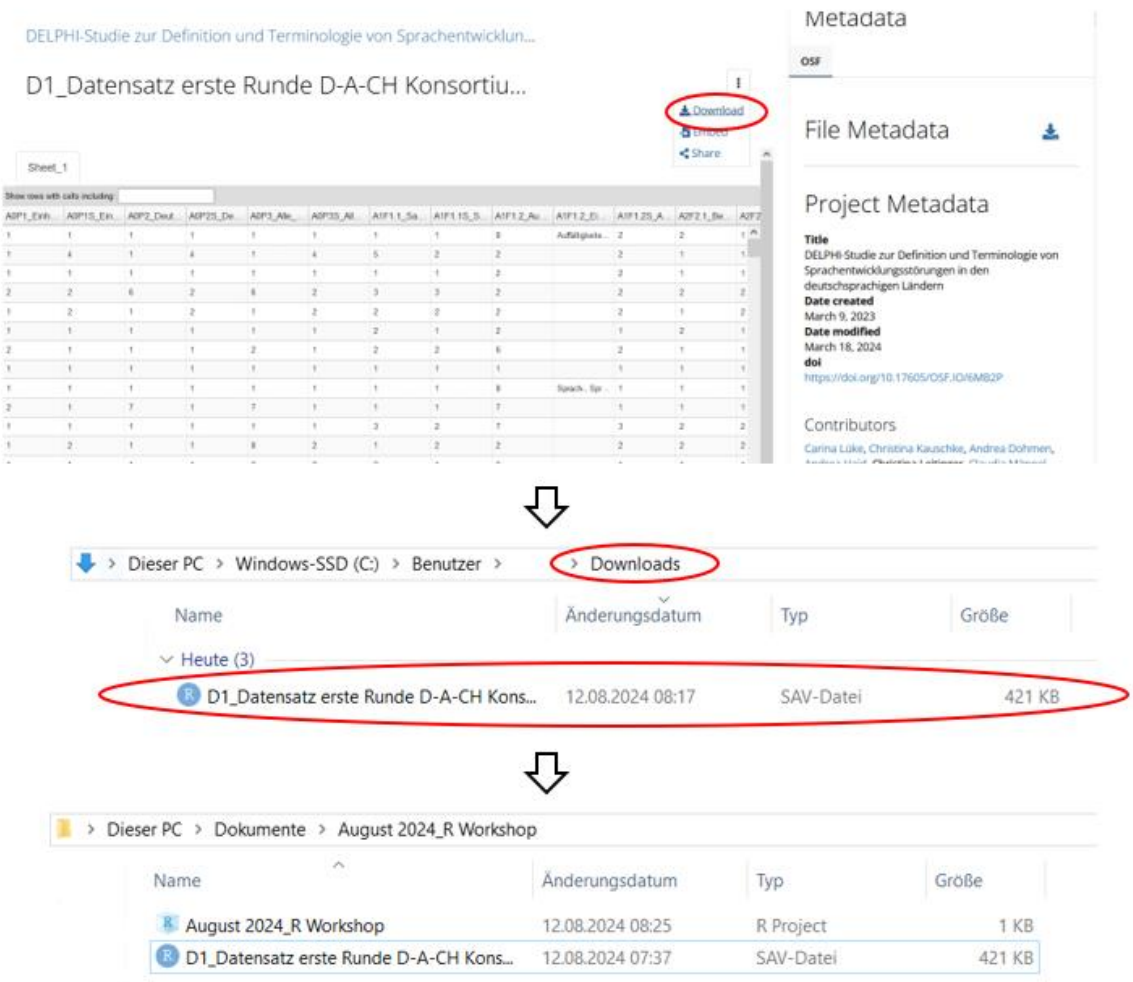

Abbildung 4: Download und Speichern des Datensatzes der ersten Delphi-Runde

Sobald du den Datensatz im Ordner deines R-Projekts abgelegt hast, wird er dir im Ausgabefenster in RStudio angezeigt (Abbildung 5).

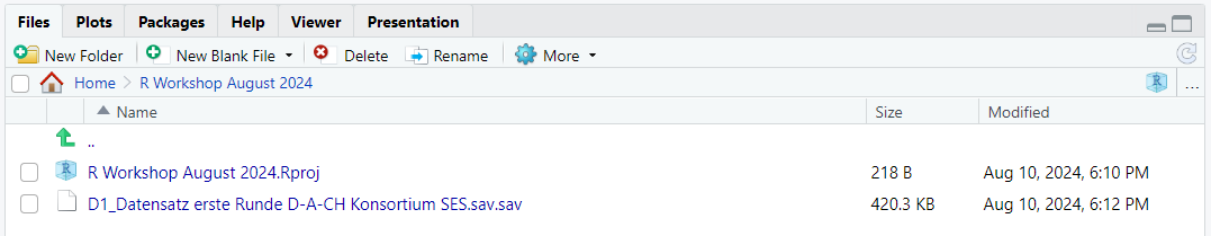

Abbildung 5: Dateien im Ausgabefenster (Screenshot der Version 4.3.2).

Du findest auf der Seite in OSF auch den Fragebogen der ersten Delphi-Runde (https://osf.io/q4en9), um dich über die Inhalte des Fragebogens zu informieren. Im Dokument ist auch eine Erläuterung der Variablen zu finden. Damit kannst du die Variablen aus dem Datensatz den Fragen im Fragebogen zuordnen.

Kopiere den folgenden Code in dein R-Skript und lass ihn laufen (strg + enter).

```
#SPSS-Datensatz einlesen

data <- read_sav("D1_Datensatz erste Runde D-A-CH Konsortium SES.sav", enco
ding="latin1")
```

Erläuterung zum Code: Du hast nun den Datensatz unter dem Objekt "data" abgespeichert (für „data" kannst du einen beliebigen Begriff einsetzen). Damit kann man sich die Arbeit erleichtern und einzelne Zwischenschritte machen. Wichtig ist außerdem, dass du immer alles exakt eintippst. Wenn du den Datensatz beim Download aus OSF bspw. umbenannt hast, musst du diesen auch in deinem Code umbenennen. Ansonsten kann die Datei nicht gefunden werden. Du gibst dem Computer mit dem Code eine Anweisung. Ist diese nicht korrekt, kann der Code nicht ausgeführt werden. Ähnlich ist es auch mit dem Ort an dem du die Datei abgespeichert hast. Ist sie im Ordner deines R-Projekts, kannst du die Datei im Ausgabefenster unter dem Reiter „Files" sehen. Wird sie hier nicht angezeigt, hast du sie woanders abgespeichert. Gib in diesem Fall nicht nur den Namen der Datei, sondern den ganzen Dateipfad an, z.B. „C:/Users/max.musternann/Desktop/Tutorial R".

Wenn du den Datensatz unter dem Objekt „data" abgespeichert hast, sieht dein Datenfenster wie folgt aus:

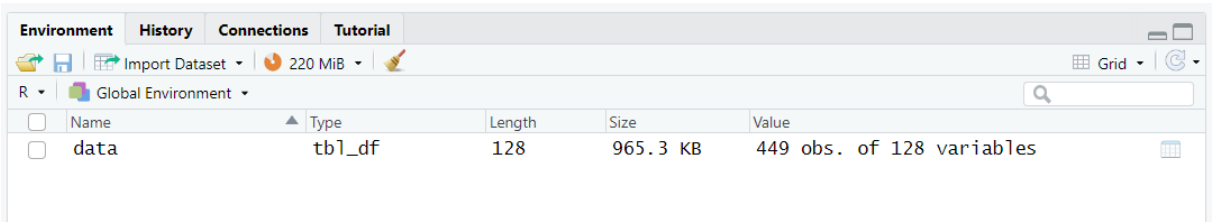

Abbildung 6: Objekt im Datenfenster (Screenshot der Version 4.3.2).

Nicht alle der Funktionen, die wir nutzen wollen, lassen sich im sav-Format anwenden. Wir wandeln deshalb das Dateiformat mit einer Funktion des *haven*-packages um und speichern den Datensatz als csv-Datei ab. Anschließend liest du den Datensatz nochmal ein. Wir speichern ihn dieses Mal unter einem anderen Objektnamen (data_edit) ab.

```
#leere Zellen werden mit -99 kodiert

data[is.na(data)]=-99

#Daten als .csv-Datei abspeichern
```

```
write.csv(data, "D1_Datensatz erste Runde D-A-CH Konsortium SES.csv")

data_edit <- read.table("D1_Datensatz erste Runde D-A-CH Konsortium SES.csv
", sep = ",", header = TRUE)
```

Achte beim Kopieren des Codes wieder darauf, dass die Zeilenumbrüche in deinem R-Projekt gleich sind und füge sie ggf. selbst ein.

> **Erläuterung zum Code:** Im Datensatz gibt es leere Zellen. Damit diese erkannt werden und der Datensatz richtig als Tabelle formatiert wird, setzen wir als Fehlwert (missing value) „-99“ ein.

Schau dir den Datensatz an. Er wird dir unter dem Objektnamen im Datenfenster angezeigt. Auf der rechten Seite findest du ein kleines Tabellen-Symbol. Klicke darauf und dir wird der Datensatz im Skriptfenster angezeigt.

Schritt 3: Deskriptive Auswertung

Zur deskriptiven Analyse der Daten nutzen wir das *SummaryTools* Package (Comtois, 2022). Mit dem Code wird eine Tabelle generiert, die zentrale Maße der zentralen Tendenz und Streuung enthält. Die Tabelle wird im Ausgabefenster angezeigt. Über den Reiter rechts oberhalb der Tabelle, kannst du diese vergrößern und in einem neuen Fenster anschauen.

```
#Summary Table aus dem SummaryTools-Package

#Erstellt eine Übersicht der deskriptiven Statistik pro Frage

view(dfSummary(data_edit, style = "grid", graph.magnif = 0.759,  plain.ascii = FALSE))
```

Als nächstes wollen wir uns nicht alle Personen im Datensatz anschauen, sondern nur einen Teilbereich und zwar die Gruppe der Mediziner*innen. Dazu nutzen wir die subset-Funktion und geben den Wert „3“ als Bedingung an, die zur Filterung der Daten genutzt wird.

```
#Nun wollen wir uns eine bestimmte Disziplin genauer anschauen. Die Info, dass Medizin mit der Zahl 3 kodiert ist, finden wir im Fragebogen.

subset_Medizin <- subset(data_edit, Disziplin==3)
```

> **Erläuterung zum Code:** Bei der subset-Funktion geben wir in der Klammer als erstes den Datensatz an, auf den wir uns beziehen. Anschließend wird die Variable, nach der wir filtern wollen, genannt und welche Ausprägung diese Variable annehmen soll.

Durch die Ausführung des Codes hast du für die Gruppe der Mediziner*innen im Datenfenster einen eigenen Datensatz „erstellt“, mit dem du jetzt weiterarbeiten kannst.

Aufgabe

1) Erstelle eine Tabelle zur deskriptiven Auswertung für die Teilgruppe der Mediziner*innen.

Schritt 4: Häufigkeitsdiagramme erstellen

Im Folgenden ist ein Beispiel gezeigt, wie man ein Säulendiagramm in R erstellen kann. Dazu nutzen wir vor allem Funktionen des ggplot2-packages. Mit dem Code erstellt man ein Säulendiagramm zur ersten Frage. Die Frage findest du auch im Fragebogen in der ersten Delphi-Runde (https://osf.io/q4en9).

P.1: Es ist sinnvoll, dass alle Personen, die professionell mit Kindern und Jugendlichen mit sprachlichen Schwierigkeiten arbeiten, eine einheitliche Terminologie zur Bezeichnung sprachlicher Beeinträchtigungen verwenden.

Inwieweit stimmen Sie dieser Aussage zu? Bitte geben Sie eine 1 an, wenn Sie voll und ganz zustimmen und eine 8, wenn Sie überhaupt nicht zustimmen. Mit den Werten dazwischen können Sie abstufen.

1 Stimme voll und ganz zu	2	3	4	5	6	7	8 Stimme über-haupt nicht zu	Urteilssicherheit (sehr sicher - eher sicher - eher unsi-cher - sehr unsi-cher)
☐	☐	☐	☐	☐	☐	☐	☐	

Abbildung 7: Auszug aus dem Fragebogen der ersten Delphi-Runde der Studie von Kauschke et. al. (2023).

```
#Erstellen eines Säulendiagramms, bspw. für das Feedback an die Befragten, zur ersten Frage

#gesamt
ggplot(data_edit, aes(x= A0P1_Einheitl._Terminologie)) +
  geom_bar(aes(y = after_stat(prop), fill = factor(after_stat(x))), stat="count") +
  geom_text(aes(label = scales::percent(after_stat(prop),1),
                y= after_stat(prop) ),
            stat= "count", vjust = -.25, size=4) +
  scale_y_continuous(labels = scales::percent, limits = c(0,1))+
  scale_fill_manual(values = c("chartreuse4", "chartreuse3", "springgreen2", "darkseagreen2", "lightsalmon", "tomato2", "firebrick1", "firebrick"), name="1=Stimme voll und ganz zu\n8=Stimme überhaupt nicht zu")+
  scale_x_continuous(breaks = seq(1, 8, 1))+
  labs(y = "Abstimmung [%]", title = "n=449") +
  theme_bw(base_size = 14)+
  theme(legend.direction = "horizontal",
        legend.position = "bottom",
        axis.title.x=element_blank(),
        legend.title=element_text(size=14),
        plot.title=element_text(size=14),
        axis.ticks.x=element_blank())
```

Erläuterung zum Code: In der ersten Zeile (in Bezug auf die Darstellung in diesem Tutorial, s. die Nummerierung links) von deinem Code kannst du angeben, aus welchem Objekt die Daten zur Erstellung des Säulendiagramms entnommen werden sollen. In diesem Beispiel wurden die Daten aus dem Objekt „data_edit“ benutzt. Du hättest aber beispielsweise auch ein Diagramm für die Gruppe der Mediziner*innen erstellen können und dazu als Objekt „subset_Medizin“ nutzen. Außerdem gibst du an, für welche Frage das Diagramm erstellt werden soll (hier: „x= A0P1_Einheitl._Terminologie“). In Zeile 6 kannst du die Schriftgröße der Prozentzahlen auf den Balken ändern (hier size=4). In den Zeilen 8-9 kannst du angeben, welche Farben die Balken haben sollen. In Zeile 10 gibst du die Bildunterschrift bzw. Legende des Diagramms ein. „\” codiert dabei den Absatz. In Zeile 12 kannst du die Beschriftung der y-Achse, z.B. „Abstimmung [%]“ und einen Titel für das Diagramm festlegen, z.B. wie hier „n=449“. Außerdem kannst du die Größe der Beschriftung von y- und x-Achse in der darauffolgenden Zeile anpassen. In Zeile 14 kannst du festlegen, in welche Richtung die Legende verläuft, also z.B. könnte die Legende auch vertikal angegeben werden und in der darauffolgenden Zeile kannst du angeben, ob die Legende über oder unter dem Diagramm oder auch rechts oder links vom Diagramm erscheinen soll. Außerdem kannst du auch noch die Schriftgröße des Titels und der Legende festlegen (hier: „size=14“).

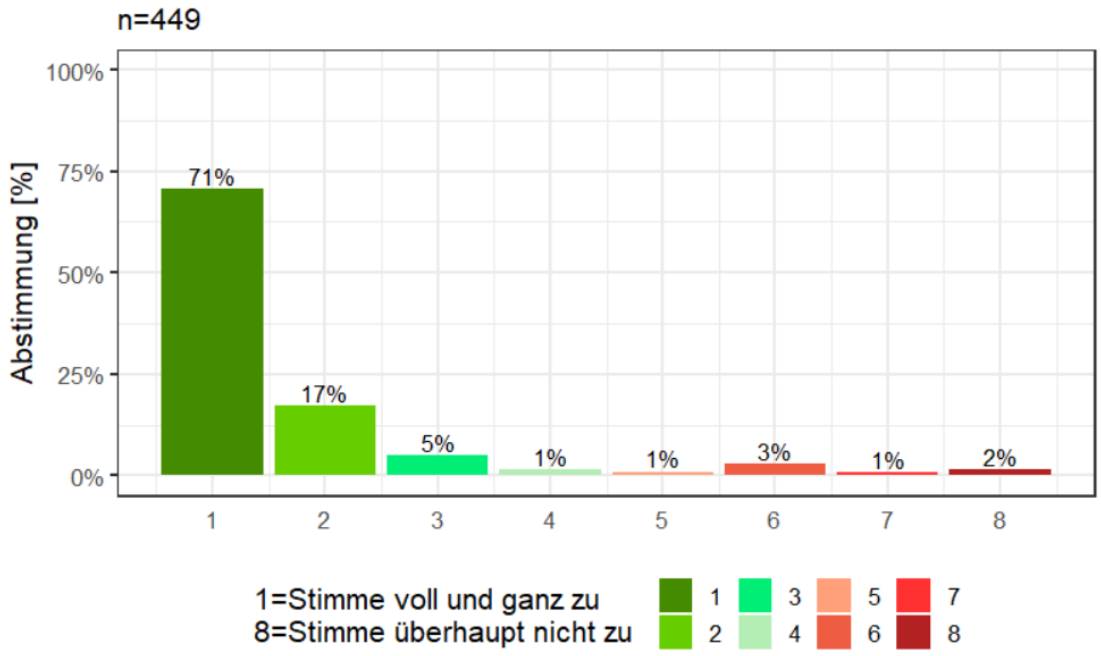

Abbildung 8: Relative Häufigkeiten der Zustimmung zur Frage nach einer einheitlichen Terminologie zur Bezeichnung sprachlicher Beeinträchtigungen. Eigene Darstellung.

Dieses Diagramm wurde den befragten Expert*innen der Delphi-Studie als Feedback rückgemeldet.

Aufgaben

1) Erstelle ein Säulendiagramm für eine andere Variable des Datensatzes, bspw. A0P2_Deutschsp_Raum.

2) Ändere die Farbe einer Säule

Du kannst auch mehrere Säulendiagramme in einer Abbildung darstellen, bspw. wenn du ein Diagramm pro Ausprägung einer Variable zeigen möchtest. Im Folgenden wird ein Diagramm für jede der fünf Disziplinen erstellt. Hierfür benötigen wir zwei weitere Packages.

```
#packages laden
library(stringr)
library(scales)

#Berechnung außerhalb der Plot-Funktion
plot_data <- data_edit %>%
  group_by(Disziplin, A0P1_Einheitl._Terminologie) %>%
  tally %>%
  mutate(percent = (n/sum(n)))

ggplot(plot_data, aes(x = A0P1_Einheitl._Terminologie, y = percent, fill=as
.factor(A0P1_Einheitl._Terminologie))) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = percent(percent,1)), vjust = -0.5, size=3.5) +
  scale_fill_manual(values = c("chartreuse4", "chartreuse3", "springgreen2"
, "darkseagreen2", "lightsalmon", "tomato2", "firebrick1", "firebrick"), na
me="1=Stimme voll und ganz zu\n8=Stimme überhaupt nicht zu")+
  labs(y = "Abstimmung [%]") +
  scale_y_continuous(labels = percent, limits = c(0,1)) +
  facet_wrap(~Disziplin, ncol = 2, labeller = labeller(Disziplin = c("1" = "
Sprachtherapie/Logopädie (n=212)","2" = "Linguistik und Sprechwissenschafte
n (n=31)","3" = "Medizin (n=142)","4" = "(Sonder-)Pädagogik (n=40)","6"="Ps
ychologie/Neurowissenschaften (n=24)")))+
  theme_bw(base_size = 12)+
  theme(legend.direction = "horizontal", legend.position = "bottom", axis.t
itle.x=element_blank(),
        legend.title=element_text(size=9),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())
```

Erläuterung zum Code: In den Zeilen 2 bis 3 lädst du erstmal die Packages, die du benötigst, um mehrere Säulendiagramme in einer Abbildung zu erstellen. In der fünften Zeile kannst du angeben, aus welchem Objekt die Daten verwendet werden sollen, also hier aus dem Objekt „data_edit“. Das Objekt „subset_Medizin“ könntest du diesmal nicht nutzen, da du ja die verschiedenen Disziplinen in einer Abbildung darstellen willst und nicht nur die Gruppe der Mediziner*innen. In der folgenden Zeile gibst du an, nach welchem Kriterium du die Diagramme gruppieren möchtest. Hier wurde nach „Disziplin“ gruppiert, das heißt pro Disziplin gibt es ein Säulendiagramm. Außerdem gibst du hier an, für welche Frage die Diagramme erstellt werden sollen, hier war es die Frage „A0P1_Einheitl._Terminologie“. Wenn du hier z.B. „A0P2_Deutschsp_Raum“ wählst, musst du innerhalb des gesamten Codes „A0P2_Deutschsp_Raum“ statt „A0P1_Einheitl._Terminologie“ angeben. In den Zeilen 13 und 14 kannst du wieder die Farben der Säulen verändern. In Zeile 15 kannst du außerdem auch wieder ein Name für deine Abbildung festlegen sowie in der darauffolgenden Zeile eine Beschriftung der „y-Achse“. In den Zeilen 18 bis 21 legst du fest, wie du die einzelnen Diagramme innerhalb deiner Abbildung beschriften möchtest. Hier wurden als Überschriften jeweils die Disziplin sowie n (Anzahl der Befragten pro Disziplin) genannt. In Zeile 23 kannst du wieder festlegen, wo die Legende sein soll und ob sie horizontal oder vertikal verlaufen soll. Probiere am besten selbst einmal aus, wie sich die Abbildung verändert, wenn du verschiedene Anpassungen innerhalb des Codes vornimmst!

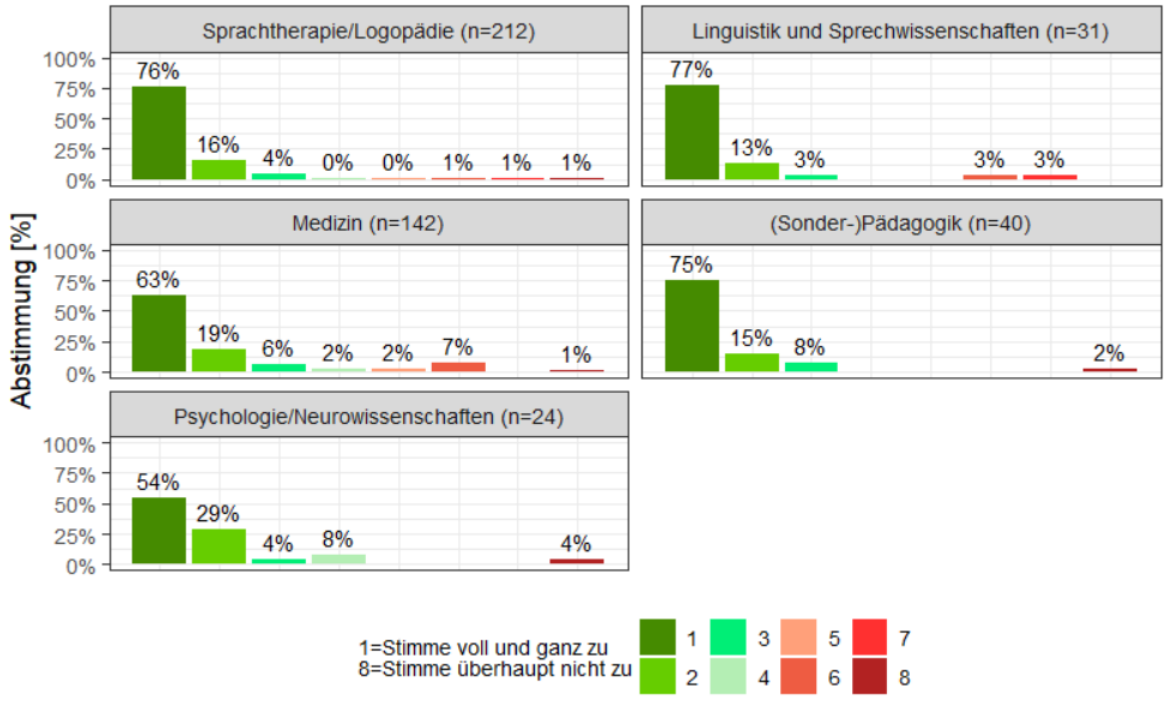

Abbildung 9: Relative Häufigkeiten der Zustimmung zur Frage nach einer einheitlichen Terminologie zur Bezeichnung sprachlicher Beeinträchtigungen (Verteilung nach Disziplin). Eigene Darstellung.

Ende des Tutorials

Vergiss nicht, dein Skript zu speichern. Wir hoffen, dass dir das Tutorial geholfen hat. Wenn du nun auf eigene Faust weiter programmierst, denk daran: Es ist ganz normal, dass nicht alles immer gleich funktioniert. Man lernt mit der Zeit, Fehler zu entdecken und diese zu korrigieren.

Falls du doch mal nicht weiterkommst, melde dich gerne bei uns.

Referenzen

Allaire, J., Xie, Y., Dervieux, C., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W. & Iannone R (2024). _rmarkdown: Dynamic Documents for R_. R package version 2.26, https://github.com/rstudio/rmarkdown

Comtois, D. (2022). *summarytools: Tools to Quickly and Neatly Summarize Data*. R package version 1.0.1, https://CRAN.R-project.org/package=summarytools

Häder, M. (2014). Delphi-Befragungen: Ein Arbeitsbuch (3. Aufl.). Springer VS. https://doi.org/10.1007/978-3-658-01928-0

Kauschke, C., Lüke, C., Dohmen, A., Haid, A., Leitinger, C., Männel, C., Penz, T., Sachse, S., Scharff Rethfeldt, W., Spranger, J., Vogt, S., Neumann, K. & Niederberger, M. (2023). Delphi-Studie zur Definition und Terminologie von Sprachentwicklungsstörungen. Eine interdisziplinäre Neubestimmung für den deutschsprachigen Raum. Logos, 31, 84-102.

Luhmann, M. (2020). R für Einsteiger. Einführung in die Statistiksoftware für die Sozialwissenschaften. Beltz. ISBN: 9783621287913.

Niederberger, M. & Spranger, J. (2020). Delphi technique in health sciences: A Map. Frontiers in public health, 8 (457), 1–10. https://doi.org/10.3389/fpubh.2020.00457

Revelle, W. (2023). psych: Procedures for Psychological, Psychometric, and Personality Research. Northwestern University, Evanston, Illinois. R package version 2.3.9, https://CRAN.R-project.org/package=psych

Wickham, H. (2022). stringr: Simple, Consistent Wrappers for Common String Operations. R package version 1.5.0, https://CRAN.R-project.org/package=stringr

Wickham, H. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Wickham, H., François, R., Henry, L., Müller, K. & Vaughan, D. (2023). _dplyr: A Grammar of Data Manipulation_. R package version 1.1.3, https://CRAN.R-project.org/package=dplyr

Wickham, H., Miller, E. & Smith, D. (2023). haven: Import and Export 'SPSS', 'Stata' and 'SAS' Files. R package version 2.5.4, https://CRAN.R-project.org/package=haven

Wickham, H., Seidel, D. (2022). scales: Scale Functions for Visualization. R package version 1.2.1, https://CRAN.R-project.org/package=scales