

Einführung in die quantitative Datenanalyse

Jan Vanhove

Universität Freiburg
Frühlingssemester 2024

Teil I

Grundlagen

Kapitel 1

Software

Unsere erste Priorität ist es, die Software, die wir in diesem Kurs verwenden werden, zu installieren und zu konfigurieren. Wir werden auch sehen, wie man diese Software als Rechenmaschine verwenden kann und wie man ihre Funktionalität um selbst geschriebene Funktionen erweitern kann.

1.1 R und RStudio installieren und konfigurieren

Um alle Schritte in diesem Skript mitverfolgen zu können, brauchen Sie die Gratis-Software R. Zwar gibt es noch andere Gratis-Software, mit der man gut Daten analysieren kann (z.B. Python, Julia, JASP), aber im Vergleich zu diesen Alternativen hat R die Vorteile, dass es in den Geistes- und Sozialwissenschaften stärker verbreitet ist und dass ich mich eben selber mit R am besten auskenne. Auf das weitere Preisen von R verzichte ich hier, siehe dazu <https://adv-r.hadley.nz/introduction.html#why-r>.

Aufgabe 1.1 (R installieren). Wenn Sie noch nicht über R verfügen (mindestens Version 4.2.0), so laden Sie es unter <https://www.r-project.org> herunter und installieren Sie es. ◇

Übrigens zeigt das Symbol ◇, wo eine Aufgabe oder Bemerkung aufhört.

Aufgabe 1.2 (RStudio installieren). Nachdem Sie R installiert haben, lohnt es sich, RStudio zu installieren. Dies ist ein benutzerfreundlicheres und kostenloses Interface, in dem Sie mit R arbeiten können. Laden Sie die Open Source Desktop-Version von RStudio unter <https://posit.co> herunter und installieren Sie diese. ◇

Öffnen Sie RStudio. Ihr Bildschirm soll jetzt so aussehen wie in Abbildung 1.1. Wenn Sie statt vier Quadranten nur drei sehen, klicken Sie auf **File**, **New File**, **R Script**.

- Links unten sehen Sie die R-Konsole. Befehle, die hier eingetragen werden, werden von R ausgeführt. Auch der Output dieser Befehle erscheint hier.

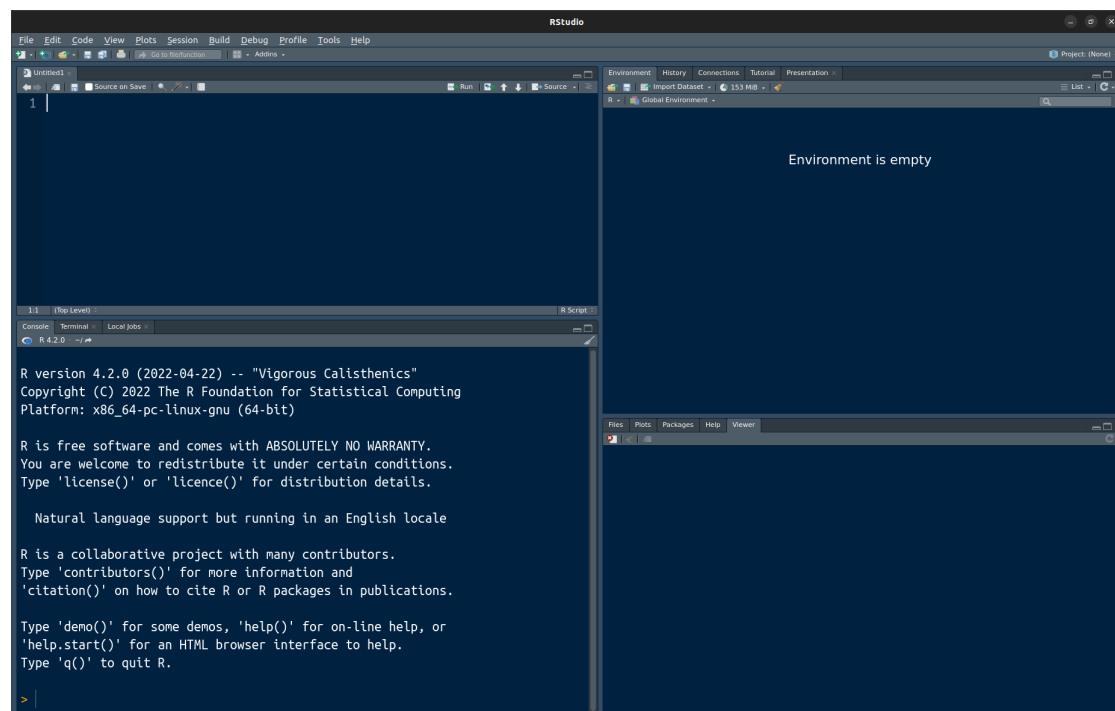


Abbildung 1.1: RStudio mit links unten der R-Konsole, links oben einem Texteditor, rechts oben dem Verzeichnis über die Arbeitsumgebung (jetzt noch leer) und rechts unten allfälligen Hilfeseiten und Grafiken.

- Links oben sehen Sie einen Texteditor. Anstatt Ihre Befehle direkt in die Konsole (links unten) einzutragen, sollten Sie diese zunächst hier eintragen. Dies macht es einfacher, die Befehle klar zu strukturieren und zu formatieren und Tippfehler aufzudecken. Ausserdem können Sie diese Skripts als .R-Datei speichern, sodass Sie Ihre Analyse nachher reproduzieren können.
- Rechts oben werden alle Objekte in der R-Arbeitsumgebung aufgelistet. Da Sie noch keine Daten eingelesen oder kreiert haben, ist diese Umgebung momentan leer.
- Wenn Sie eine Grafik zeichnen, sehen Sie diese rechts unten. Wenn Sie eine Hilfeseite abfragen, erscheint diese ebenfalls hier. Dieses Fenster kann auch als Dateimanager (wie Windows-Explorer) verwendet werden.

Aufgabe 1.3 (RStudio konfigurieren). Klicken Sie dazu in RStudio auf **Tools** und wählen Sie **Global options...** aus. Stellen Sie sicher, dass unter **General**, **Basic** die Option 'Restore .RData into workspace at startup' *nicht* angekreuzt ist und dass bei 'Save workspace to .RData on exit' die Option 'never' ausgewählt wurde. Mit diesen Einstellungen vermeiden Sie, dass bei einem Neustart noch alte Resultate und Objekte in der Arbeitsumgebung herumliegen, die Ihre neuen Berechnungen beeinflussen, ohne dass Sie sich dessen bewusst sind.

Weiter sollten Sie unter Code, Editing das Kästchen 'Use native pipe operator, |>' ankreuzen. Was dieser 'pipe operator' vermag, werden Sie schon bald merken. Zum Schluss empfehle ich Ihnen, dass Sie unter Code > Saving noch die 'Default text encoding' auf UTF-8 wechseln. Das sollte die Wahrscheinlichkeit erhöhen, dass Ihre Skripts auf anderen Computern richtig angezeigt werden, wenn diese Sonderzeichen (wie Umlaute) enthalten. ◇

1.2 R als Rechenmaschine

1.2.1 Rechenoperationen

Mit R verfügen Sie über eine Rechenmaschine. Summen, Produkte und Quotienten können Sie berechnen, indem Sie Befehle wie die unten stehenden auf die Konsole (unten links) eintragen. Die Ergebnisse gehören nicht zu den Befehlen; diese werden auf der Konsole angezeigt, sobald Sie den Befehl eingetragten haben und mit ENTER bestätigt haben. Alles, was rechts von einem Doppelkreuz (#) steht, wird von R ignoriert. So können wir Befehle mit Kommentaren versehen:

```
10 + 7      # Addition
[1] 17

12 - 28     # Subtraktion
[1] -16

7 * 3.5     # Multiplikation
[1] 24.5

11 / 3      # übliche Division
[1] 3.666667

6^3         # Potenz
[1] 216

10^(-2)     # = 1/10^2 = 1/100
[1] 0.01

4^0         # x^0 = 1; für R auch 0^0 = 1.
[1] 1

sqrt(16)    # Quadratwurzel
[1] 4

log(16, 2)  # 2er-Logarithmus von 16, d.h., 2^4 = 16
[1] 4
```

```
ceiling(2.3) # aufrunden
[1] 3

floor(11.7) # abrunden
[1] 11

round(2.3) # symmetrische Rundung
[1] 2

round(2.7)
[1] 3

round(2.5) # falls ~.5: auf nächstgelegene gerade Zahl runden
[1] 2

round(3.5)
[1] 4

abs(-3) # Absolutbetrag
[1] 3

sign(-3) # Vorzeichen
[1] -1

sign(3)
[1] 1

sign(0)
[1] 0
```

R respektiert die übliche Reihenfolge, in der Rechenoperationen durchgeführt werden. Um eine andere Reihenfolge festzulegen, werden runde oder geschweifte Klammern verwendet, wie auch in einigen der Beispiele oben:

```
4 + 8 * 2
[1] 20

(4 + 8) * 2
[1] 24

{4 + 8} * 2
[1] 24
```

Bemerkung 1.4 (Wurzeln). Wenn wir $\sqrt{-3}$ berechnen wollen, so erhalten wir als Resultat NaN (*not a number*), denn es gibt keine reelle Zahl x mit $x^2 = -3$.

```
sqrt(-3)
```

```
Warning in sqrt(-3): NaNs produced
```

```
[1] NaN
```

In den komplexen Zahlen gibt es schon eine Zahl x , sodass $x^2 = -3$. (Genau gesagt gibt es zwei solche Zahlen, nämlich $0 \pm \sqrt{3}i$.) Mit komplexen Zahlen wollen wir uns hier aber nicht befassen.

Fürs Ziehen von beliebigen Wurzeln von nicht-negativen Zahlen ist zu bemerken, dass $\sqrt[n]{x} = x^{1/n}$ gilt. Somit lässt sich $\sqrt[3]{216}$ als $216^{1/3}$ berechnen:

```
216 ^ (1/3)
```

```
[1] 6
```

Fürs Ziehen von beliebigen Wurzeln von negativen Zahlen (z.B. $\sqrt[3]{-8}$) müssten wir aber wieder einen Umweg über die komplexen Zahlen machen, worauf wir hier verzichten. ◇

Bemerkung 1.5 (Wissenschaftliche Notation). Manchmal wird das Ergebnis einer Berechnung in einem anderen Format dargestellt. Berechnen wir beispielsweise 3328^{27} , so erhalten wir als Ergebnis $1.26\text{e}+95$. (Die Zahlen im Fliesstext stimmen übrigens nicht immer mit den Zahlen im R-Output überein, da ich sie stärker runde.)

```
3328^27
```

```
[1] 1.255884e+95
```

Diese Schreibweise nennt man **wissenschaftliche Notation** und ist zu lesen als $1.26 \cdot 10^{95}$ – also als 1.26, aber mit dem Dezimaltrennzeichen 95 Stellen nach rechts verschoben. Das Ergebnis von 2^{-21} wird als $4.8\text{e}-07$ angezeigt, was $4.8 \cdot 10^{-7}$ heisst – also als 4.8, aber mit dem Dezimaltrennzeichen sieben Stellen nach links verschoben, d.h., 0.00000048.

```
2^(-21)
```

```
[1] 4.768372e-07
```

◇

Bemerkung 1.6 (Hilfeseiten). Im Prinzip gibt es für jede R-Funktion eine Hilfeseite. Diese können Sie anzeigen, indem Sie `?Funktionsname` in die Konsole eintippen. Das Lesen der Hilfeseiten will gelernt sein. Nehmen wir uns also doch einmal die Rundungsfunktion `round()` genauer unter die Lupe:

```
?round # Output nicht gezeigt im Skript
```

Diese Hilfeseite besteht aus neun Abschnitten; fünf von diesen findet man auf den meisten Hilfeseiten:

1. Description: Kurze Beschreibung der Funktion.
2. Usage: Welche Parameter hat die Funktion? Die Funktion `round()` hat zwei Parameter: `x` und `digits`. Wird kein Wert für `digits` angegeben, so wird die Defaulteinstellung von 0 verwendet.
3. Arguments: Kurze Angaben zu den erwarteten Parameterwerten.
4. Details: Eben detailliertere Angaben zur Funktion. Bei `round()` wird beispielsweise erklärt, wie und wieso 1.5 gerundet wird und wie und wieso der Befehl `round(0.15, 1)` zu unerwarteten Ergebnissen führen kann.
5. Examples: Beispiele zur Anwendung der Funktion. Sie können auf 'Run examples' klicken, um zu sehen, welchen Output die Beispiele generieren.

Die Hilfeseite für etwa den Additionsoperator `+` können Sie übrigens mit `?"+` abrufen. ◇

1.2.2 Funktionsgraphen

Mit dem unten stehenden Befehl kann man den Graphen einer mathematischen Funktion zeichnen; siehe Abbildung 1.2. Tippen Sie diesen Befehl in RStudio ins Fenster links oben ein. Und ich meine tatsächlich 'tippen', nicht 'selektieren, kopieren und einkleben': Sie werden viel mehr lernen, wenn Sie die Befehle in diesem Skript abtippen als wenn Sie diese einfach aus dem PDF kopieren und einkleben! Selektieren Sie dann eine Zeile dieses Befehls und drücken Sie `CTRL + ENTER` (Windows und Linux) oder `CMD + ENTER` (macOS). Der Befehl wird nun an die Konsole (links unten) weitergeleitet. Im Prinzip können Sie diesen kurzen Befehl auch sofort in die Konsole eintragen, aber Sie sollten es sich angewöhnen, im Texteditor zu arbeiten. Fehler können so wesentlich einfacher aufgedeckt und behoben werden. Ausserdem erleichtert das Arbeiten im Texteditor das Dokumentieren Ihrer Analyse, da Sie Ihre Scripts abspeichern können.

```
curve(2^(-x^2/2) * x/(1 + x^2),  
      from = -5, to = 5,  
      xlab = "x", ylab = "f(x)")
```

Jeder mit `curve()` gezeichnete Funktionsgraph sieht stetig, also lückenlos, aus. Aber beispielsweise die Vorzeichen- (oder Signum-)Funktion,

$$\text{sgn}(x) := \begin{cases} -1, & \text{falls } x < 0, \\ 0, & \text{falls } x = 0, \\ 1, & \text{falls } x > 0, \end{cases}$$

hat eine Unstetigkeitsstelle bei $x = 0$. Wenn Sie den Graphen dieser Funktion zeichnen würden, so würde es aussehen, als wäre dieser zwar stark wachsend um den Nullpunkt herum, aber

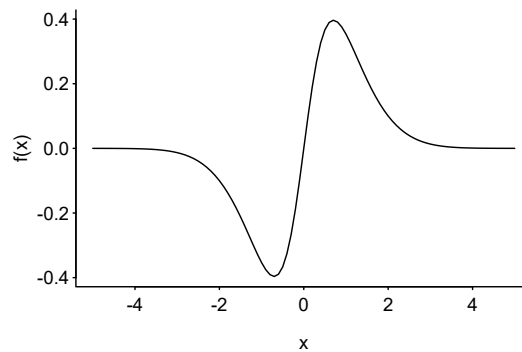


Abbildung 1.2: Der Graph der Funktion $x \mapsto 2^{-\frac{x^2}{2}} \frac{x}{1+x^2}$.

schon ununterbrochen. Um den richtigen Graphen dieser Funktion zu berechnen, müsste man sich etwas mehr Mühe machen. Darauf verzichten wir hier aber erst mal.

Auch kann ein mit `curve()` gezeichneter Graph es aussehen lassen, als ob die gezeichnete Funktion überall definiert ist, während dies nicht so ist. So ist die Funktion $x \mapsto \frac{\sin(x)}{x}$ bei $x = 0$ nicht definiert, denn durch 0 kann man nicht teilen. Zeichnet man jedoch den Graphen dieser Funktion, so könnte man denken, dass $0 \mapsto 1$. Man könnte daher im Graphen ein Kreischen bei den Koordinaten $(0,1)$ zeichnen, um deutlich zu machen, dass die Funktion bei $x = 0$ nicht definiert ist:

```
curve(sin(x)/x,
      from = -8, to = 8,
      xlab = "x", ylab = "f(x)")
# Mit 'cex' (character expansion) kann die Grösse geregelt werden.
points(0, 1, cex = 2)
```

1.2.3 Vektorisierte Rechenoperationen

In R ist ein **Vektor** eine bestimmte Anzahl von Zahlen (oder anderen Objekten), die zusammen gruppiert werden. Mit der Funktion `c()` (für *concatenate*) kann man (unter anderem) Zahlen zu einem Vektor zusammenfügen. Mit dem *assignment operator* `<-` kann man diesem Vektor einen Namen geben (hier `a`):

```
a <- c(-4, 2, 0.3, 2014)
```

Der Vektor `a` enthält nun die vier aufgeführten Zahlen. Wir können den Vektor ganz anzeigen, indem wir seinen Namen eintippen:

```
a
[1] -4.0 2.0 0.3 2014.0
```

Wir können bestimmte Komponenten nach ihrer Position aus dem Vektor herauslesen. Mit `a[i]` lesen wir den i -ten Eintrag von a heraus, also a_i , $1 \leq i \leq n$. Die folgenden Beispiele zeigen weitere Möglichkeiten, um Einträge abzufragen.

```
a[1]           # 1. Komponente
[1] -4
a[-3]          # Alle ausser der 3. Komponente
[1] -4      2 2014
a[c(2, 4)]     # Verwendung von c() um mehrere Elemente herauszulesen
[1]      2 2014
a[-c(2, 4)]    # Alle ausser der 2. und der 4. Komponente
[1] -4.0  0.3
```

Falls Sie bereits Erfahrung mit anderen Computersprachen haben, so sollten Sie aus diesem Beispiel mitnehmen, dass R kein *zero-based indexing* verwendet: Die erste Komponente von a wird anders als vielen anderen Programmiersprachen mit `a[1]`, nicht mit `a[0]`, abgefragt.

Die Summe aller Einträge eines Vektors a – in sog. **Gross-Sigma-Notation**

$$\sum_{i=1}^n a_i := a_1 + a_2 + \cdots + a_n,$$

mit n der Anzahl Einträge im Vektor, können wir einfach mit `sum()` berechnen:

```
sum(a)
[1] 2012.3
```

Das Produkt aller Einträge eines Vektors, also

$$\prod_{i=1}^n a_i := a_1 \cdot a_2 \cdot \cdots \cdot a_n,$$

können wir mit `prod()` berechnen:

```
prod(a)
[1] -4833.6
```

Die Länge eines Vektors kann mit `length()` abgefragt werden; das Maximum und Minimum mit `max()` bzw. `min()`:

```
length(a)
[1] 4
```

```
max(a)
[1] 2014
min(a)
[1] -4
```

Die Operationen aus Abschnitt 1.2.1 können auch auf Vektoren angewandt werden. In diesem Fall werden sie komponentenweise ausgeführt:

```
2 * a
[1] -8.0 4.0 0.6 4028.0
a - 7
[1] -11.0 -5.0 -6.7 2007.0
sqrt(a) # Warnung wg. Wurzel von negativer Zahl
Warning in sqrt(a): NaNs produced
[1] NaN 1.4142136 0.5477226 44.8776113
log(a, 2)
Warning: NaNs produced
[1] NaN 1.000000 -1.736966 10.975848
```

Wenn wir noch einen zweiten Vektor der gleichen Länge definieren, so können wir gewisse Operationen komponentenweise mit den übereinstimmenden Elementen durchführen:

```
b <- c(3, 8.2, -1.2, sqrt(20))
a - b
[1] -7.000 -6.200 1.500 2009.528
a + b
[1] -1.000 10.200 -0.900 2018.472
a / b
[1] -1.3333333 0.2439024 -0.2500000 450.3440907
a * b
[1] -12.000 16.400 -0.360 9006.882
a ^ b
[1] -6.400000e+01 2.940668e+02 4.240865e+00 5.973146e+14
```

Um das komponentenweise Maximum bzw. Minimum von zwei oder mehr Vektoren zu berechnen, brauchen wir `pmax()` bzw. `pmin()` (*p* für *parallel*):

```
pmax(a, b)
[1] 3.0 8.2 0.3 2014.0

pmin(a, b)
[1] -4.000000 2.000000 -1.200000 4.472136
```

Wollen wir den maximalen bzw. minimalen Eintrag mehrerer Vektoren berechnen, so bieten sich wieder `max()` bzw. `min()` an:

```
max(a, b)
[1] 2014
```

Bemerkung 1.7 (Produkte von Vektoren). Es gibt mehrere Arten und Weisen, wie man ein Produkt zweier Vektoren definieren kann. Das komponentenweise Produkt (oder Hadamard-Produkt), das wir hier berechnen, ist nicht gleich dem sog. Vektorprodukt oder dem sog. Skalarprodukt. Da gerade das Skalarprodukt noch oft in Texten zu Statistik auftaucht, musste dies hier kurz erwähnt werden. ◇

Bemerkung 1.8 (Vektoren ungleicher Länge). Streng genommen muss der zweite Vektor nicht gleich lang sein wie der erste. Wenn die Vektoren aber nicht gleich lang sind, kann es schwierig sein, die Ergebnisse der Berechnungen zu antizipieren. Als Beispiel definieren wir einen dreistelligen Vektor *d* und addieren wir diesen zum vierstelligen Vektor *a*. (Es gibt eine R-Funktion `c()`, weshalb wir diesen dritten Vektor nicht *c*, sondern *d* nennen.)

```
d <- c(3, 8.2, -1.2)
a + d
Warning in a + d: longer object length is not a multiple of shorter object length
[1] -1.0 10.2 -0.9 2017.0
```

Da die Länge des zweiten Vektors kein Vielfaches der Länge des ersten Vektors ist, wird die vierte Addition mit dem vierten Element des ersten Vektors und dem ersten Element des zweiten Vektors durchgeführt. Dies sorgt garantiert für Verwirrung. Vermeiden Sie diese Situation.

Ist die Länge des zweiten Vektors schon ein Vielfaches der Länge des ersten Vektors, so wird der zweite Vektor rezykliert, ohne dass man eine Warnung erhält. Im folgenden Beispiel ist der dritte Eintrag von *a + d* daher gleich dem dritten Eintrag von *a* plus den ersten Eintrag von *d*.

```
d <- c(3, 8.2)
a + d
[1] -1.0 10.2 3.3 2022.2
```

Auch dies dürfte für Verwirrung sorgen. ◇

Bemerkung 1.9 (Zahlenfolgen). Wir haben bereits gesehen, dass man Vektoren mit der `c()`-Funktion definieren kann. Will man eine Zahlenfolge als Vektor definieren, so gibt es aber einfachere Varianten:

```
z <- 3:12
z
[1] 3 4 5 6 7 8 9 10 11 12

z <- 7:-1
z
[1] 7 6 5 4 3 2 1 0 -1

z <- seq(from = 1, to = 3.5, by = 0.25)
z
[1] 1.00 1.25 1.50 1.75 2.00 2.25 2.50 2.75 3.00 3.25 3.50

z <- seq(from = 0, to = 2, length.out = 7)
z
[1] 0.0000000 0.3333333 0.6666667 1.0000000 1.3333333
[6] 1.6666667 2.0000000

z <- seq(to = 2, by = 12, length.out = 5)
z
[1] -46 -34 -22 -10 2

z <- seq_len(5)
z
[1] 1 2 3 4 5
```

Erklären Sie sich, was jeder dieser Befehle bewirkt. ◇

1.3 R als Programmiersprache

Die Funktionalität von R kann um eigens geschriebene Funktionen ergänzt werden. Wir wollen anhand von zwei Beispielen zeigen, wie dies funktioniert. Weitere Beispiele folgen in späteren Kapiteln.

Beispiel 1.10 (Kumulative Summe). Wir erhalten einen Vektor mit numerischen Werten und wollen einen neuen Vektor kreieren, der die kumulativen Summen der Werte dieses Vektors enthält. Wenn wir zum Beispiel die kumulativen Summen für den Vektor $(1, 4, 9)$ berechnen, so erhalten wir $(1, 1 + 4, 1 + 4 + 9) = (1, 5, 15)$.

Mit dem unten stehenden Code erzeugen wir eine neue Funktion namens `kum_summe()`, der einen einzigen Parameter `x` hat. Dieses `x` spielt die Rolle des Eingabevektors. Die Funktion kreiert zunächst einen neuen Vektor mit der gleichen Länge des Eingabevektors. Den ersten Wert dieses neuen Vektors wird auf den ersten Wert des Eingabevektors gestellt. Danach wird eine **for-Schleife** verwendet, um die weiteren kumulativen Summen zu bestimmen. Am Ende verlassen wir die Funktion und geben den neuen Vektor aus:

```
kum_summe <- function(x) {
  resultat <- vector(length = length(x))
  resultat[1] <- x[1]
  for (i in 2:length(x)) {
    resultat[i] <- resultat[i - 1] + x[i]
  }
  return(resultat)
}
```

Verwenden können wir die neue Funktion wie jede andere R-Funktion:

```
a <- c(-8, 12, pi, 203, 2)
b <- kum_summe(a)
b

[1] -8.000000  4.000000  7.141593 210.141593 212.141593

kum_summe(b)

[1] -8.000000 -4.000000  3.141593 213.283185 425.424778
```

Das Objekt `resultat` existiert übrigens nur innerhalb der Funktion und ist also nicht in der Arbeitsumgebung vorhanden:

```
resultat

Error in eval(expr, envir, enclos): object 'resultat' not found
```

Die neue Funktion `kum_summe()` dient nur didaktischen Zwecken, da es bereits die R-Funktion `cumsum()` gibt, die die gleiche Berechnung ausführt. ◇

Beispiel 1.11 (Fakultät). Seit 2023 spielen zwölf Teams in der Schweizer Super League. Wie viele möglichen Tabellenstände gibt es, wenn wir die Möglichkeit gleich platzierter Teams ausser Acht lassen?

Es gibt 12 Möglichkeiten, den ersten Rang zu belegen. Sobald wir ein Team für den ersten Rang ausgewählt haben, bleiben nur noch 11 Möglichkeiten für den zweiten Rang übrig. Haben wir die ersten zwei Ränge festgelegt, so bleiben noch 10 Möglichkeiten für den dritten Rang übrig. So geht es weiter, bis wir am Schluss nur noch eine Möglichkeit für das letzte Team haben. Insgesamt gibt es also

$$12 \cdot 11 \cdot 10 \cdot \dots \cdot 2 \cdot 1 = 479'001'600$$

möglichen Tabellenstände in der Super League.

Diese Erkenntnis kann verallgemeinert werden: Bei n unterschiedlichen Objekten gibt es $n \cdot (n-1) \cdot \dots \cdot 1$ unterschiedliche Möglichkeiten, diese anzuordnen. Entsprechend bezeichnen wir mit $n!$ die **Fakultät** (*factorial*) einer natürlichen Zahl n als

$$n! := \begin{cases} 1, & \text{falls } n = 0, \\ n \cdot (n-1) \cdot \dots \cdot 1, & \text{sonst.} \end{cases}$$

Mit der Definition von $0! := 1$ könnte man sich anfreunden, indem man sich überlegt, dass da man nur eine Möglichkeit hat, keine Objekte anzuordnen (nämlich: nichts tun). Aber vor allem erleichtert diese Definition viele Berechnungen und Beweise.

Wir können diese Definition nun als Grundlage einer eigens geschriebenen R-Funktion zur Berechnung der Fakultät verwenden.

```
fakultaet <- function(n) {
  if (n == 0) {
    return(1)
  }
  resultat <- 1
  for (i in 1:n) {
    resultat <- i * resultat
  }
  return(resultat)
}
```

Die neu definierte Funktion `fakultaet()` akzeptiert einen Parameter, n (Zeile 1). Wenn dieses n gleich 0 ist, so gibt die Funktion 1 als Resultat zurück, und wir verlassen die Funktion (Zeilen 2–4). Zu bemerken sind hier folgende Sachen:

- Wir verwenden `'=='` (nicht `'='`), um zwei Ganzzahlen auf Gleichheit zu testen. (Um reelle Zahlen auf Gleichheit zu testen, siehe `all.equal()`!)
- Wir verwenden geschweifte Klammern, um zu definieren, was alles gemacht werden muss, wenn die Bedingung in der *if*-Klausel erfüllt ist.
- Wir können `return()` mehrmals verwenden. Die Funktion wird beim zuerst angetroffenen `return()` verlassen und nicht weiter ausgeführt.

Ist die Bedingung in der *if*-Klausel nicht erfüllt, so verlassen wir die Funktion vorübergehend nicht. Zunächst initialisieren wird eine Variable namens `resultat` mit dem Anfangswert von 1 (Zeile 5). In einer *for*-Schleife iterieren wir nun durch alle Ganzzahlen zwischen 1 und n und multiplizieren jeweils diese Zahl mit dem jetzigen Wert von `resultat`, um so den nächsten Wert von `resultat` zu erhalten. Am Ende verlassen wir die Funktion und geben dieses `resultat` aus.

Die neu geschriebene Funktion kann man nun wie jede andere R-Funktion verwenden:

```
fakultaet(12)
[1] 479001600
```

Wenn Sie Ihre R-Session neu starten, so werden Sie jedoch die neue Funktion nochmals definieren müssen, bevor Sie diese verwenden können.

Übrigens können *if*-Klauseln noch um *else if*- und *else*-Klauseln ergänzt werden, wie in der Code-Schablone unten. In der Funktion `fakultaet()` brauchen wir solche *else (if)*-Klauseln jedoch nicht, da wir die Funktion bereits in der *if*-Klausel verlassen.

```
# Beispiel mit else (if):
if (Bedingung1) {
  # Dieser Code wird ausgeführt, wenn Bedingung1 erfüllt ist
} else if (Bedingung2) {
  # Dieser Code wird ausgeführt, wenn Bedingung2 erfüllt ist,
  # aber Bedingung1 nicht.
} else {
  # Dieser Code wird ausgeführt, wenn weder Bedingung 1
  # noch Bedingung2 erfüllt ist.
}
```

Statt eine *for*-Schleife zu verwenden, können wir auch einen Vektor mit allen Ganzzahlen von 1 bis n kreieren und diese miteinander multiplizieren:

```
fakultaet2 <- function(n) {
  if (n == 0) {
    return(1)
  }
  return(prod(1:n))
}
fakultaet2(12)
[1] 479001600
```

Eine weitere Variante bietet sich an, wenn wir bemerken, dass wir $n!$ rekursiv umschreiben können, denn für $n \geq 1$ gilt

$$n! = n \cdot \underbrace{(n-1) \cdot (n-2) \cdot \dots \cdot 1}_{=(n-1)!}.$$

Die Funktion `fakultaet3()` ruft entsprechend sich selbst auf, um $n!$ zu berechnen:

```
fakultaet3 <- function(n) {
  if (n == 0) {
    return(1)
  }
  return(n * fakultaet3(n-1))
}
```



```

    }
    return(n * fakultaet3(n - 1))
}

fakultaet3(12)

[1] 479001600

```

Bei dieser Lösung ist jedoch Vorsicht geboten: Wenn wir zum Beispiel den Befehl `fakultaet3(-1)` oder `fakultaet3(2.5)` ausführen, so geraten wir in eine endlose Rekursion, da der Basisfall ($n = 0$) nie erreicht wird.

Natürlich hat R eine eingebaute und optimierte Funktion, um Fakultäten zu berechnen:

```

factorial(12)

[1] 479001600

```

◇

Aufgabe 1.12 (Skalarprodukt). Seien $x = (x_1, \dots, x_n)$ und $y = (y_1, \dots, y_n)$ zwei Vektoren, beide der Länge $n \geq 1$ und mit reellen Zahlen als Komponenten. Das sogenannte (Standard-)Skalarprodukt weist diesem Paar von Vektoren eine reelle Zahl zu und zwar

$$\langle x, y \rangle := \sum_{i=1}^n x_i y_i = x_1 y_1 + x_2 y_2 + \dots + x_n y_n.$$

Schreiben Sie eine R-Funktion `skp()`, welche zwei Vektoren als Parameter akzeptiert und ihr Skalarprodukt berechnet. Wenn die zwei Eingabevektoren nicht gleich lang sind, so soll Ihre Funktion eine Fehlermeldung geben. Diese können Sie mit dem Befehl `stop()` erzeugen:

```
stop("Vectors not of same length.")
```

Hinweis zur Selbstkontrolle: Der Output des folgenden Befehls müsste -6 sein.

```
skp(c(1, -2, 3), c(-1, -2, -3))
```

◇

Aufgabe 1.13 (Fibonacci-Zahlen). Die Fibonacci-Folge $1, 1, 2, 3, 5, \dots$ ist wie folgt definiert:

$$\text{Fib}(n) := \begin{cases} 1, & \text{falls } n \leq 2, \\ \text{Fib}(n-1) + \text{Fib}(n-2), & \text{sonst,} \end{cases}$$

wobei n eine natürliche Zahl ungleich 0 ist. Schreiben Sie eine Funktion `Fib()`, mit der Sie die n -te Fibonacci-Zahl berechnen können. Verwenden Sie diese Funktion, um die 25. Fibonacci-Zahl zu berechnen. Was stellen Sie fest, wenn Sie die 100. Fibonacci-Zahl berechnen möchten?

Hinweis: Sie können eine Berechnung meistens abbrechen, indem Sie bei der Konsole auf STOP klicken. ◇

1.4 Erweiterungspakete

Es gibt für R einen Haufen Erweiterungspakete, mit denen man z.B. informative Grafiken gestalten kann oder spezialisierte statistische Modelle rechnen kann.

Aufgabe 1.14 (tidyverse installieren). Mit dem unten stehenden Befehl installieren Sie das tidyverse-Bündel: eine Sammlung unterschiedlicher Pakete, die alle auf der gleichen Philosophie basieren und die das Arbeiten mit Datensätzen wesentlich erleichtern.

```
install.packages("tidyverse")
```

Für die Erweiterungspakete, die im tidyverse-Bündel zusammengepackt wurden, finden Sie unter <https://www.tidyverse.org/> Anleitungen und weitere Informationen. ◇

Aufgabe 1.15 (here installieren). Ein weiteres nützliches Erweiterungspaket, das wir bald verwenden werden, ist das here-Package. Installieren Sie es. ◇

1.5 R-Projekte

Es ist sinnvoll, wenn sich die Scripts, Datensätze, Grafiken, usw., die Sie für ein Forschungsprojekt brauchen oder kreiert haben, alle im gleichen Ordner ('Arbeitsordner') befinden. Am besten richten Sie dazu für jedes Forschungsprojekt, an dem Sie beteiligt sind (inklusive Seminar- und Masterarbeiten), ein R-Projekt ein. Auch für diesen Kurs sollten Sie ein solches Projekt einrichten.

Aufgabe 1.16 (R-Projekt einrichten). Klicken Sie in RStudio auf File, New Project..., New directory, New Project und geben Sie dem Projekt einen sinnvollen Namen (z.B. Statistikkurs). Für diesen Kurs brauchen Sie die Option Use renv with this project nicht einzuschalten. Es wird jetzt ein Ordner kreiert, der eine Datei mit der Endung .Rproj enthält. Um das R-Projekt zu öffnen, können Sie diese Datei öffnen oder das Projekt in RStudio unter File, Open Project... auswählen.

Wenn Sie das Projekt geöffnet haben, sollten Sie im Fenster rechts unten noch die Registerkarte Files öffnen und mit New Folder die Unterordner data, figs, scripts und assignments kreieren. Die Datensätze, mit denen wir arbeiten werden, sollten Sie in data ablegen; in figs werden wir Abbildungen (Grafiken) speichern; in scripts ein paar R-Scripts; und in assignments können Sie Ihre Hausaufgaben speichern. ◇

1.6 Softwareversionen und Updates

R und seine Erweiterungspakete sind in ständiger Entwicklung. Um ein Update der installierten R-Packages durchzuführen, können Sie den Befehl update.packages() verwenden. Um R

selber auf den neusten Stand zu bringen, finde ich es eigentlich am einfachsten, die alte Version komplett zu löschen und die neue zu installieren. Danach muss ich dann aber die Packages, die ich brauche, erneut installieren. Dies kann recht mühsam sein, weshalb ich Ihnen empfehle, solche Upgrades in der vorlesungsfreien Zeit durchzuführen statt mitten im Semesterstress.

Aber Achtung: Es kommt nicht selten vor, dass alter R-Code nach einem Update nicht mehr oder etwas anders funktioniert. Für grössere Forschungsprojekte empfiehlt es sich daher, die Option `Use renv with this project` anzukreuzen. Diese sorgt dafür, dass die Versionen der Packages, die Sie im Projekt verwenden, als Teil des Projekts gespeichert werden. Die verwendeten Packages müssen jedoch im Projekt neu installiert werden. Auch wenn Sie nachher für ein anderes Projekt eine neuere Version des Packages verwenden, werden die Befehle im ersten Projekt noch mit der alten Version ausgeführt. Dies verringert die Gefahr, dass Ihr alter Code ein paar Jahre später nicht mehr funktioniert. Für mehr Informationen, siehe <https://rstudio.github.io/renv/articles/renv.html>.

Für dieses Skript wurde R-Version 4.3.1 verwendet. Eine Übersicht über die Packageversionen finden Sie im Anhang.

1.7 Software zitieren

R und R-Pakete sind gratis und werden mehrheitlich von anderen Forschenden entwickelt. Wenn Sie bei Ihrer Arbeit sehr von R oder einem Erweiterungspaket profitiert haben, ziehen Sie es dann bitte in Erwägung, diesen Freiwilligen mit einer Referenz zu danken.

Eine Referenz für R erhalten Sie, wenn Sie den folgenden Befehl in die Konsole eintippen. Den Output dieses Befehls wird hier im Skript nicht gezeigt.

```
citation()
```

Wenn Sie ein bestimmtes Erweiterungspaket zitieren möchten, stellen Sie den Namen des Pakets zwischen Klammern und Anführungszeichen, etwa so:

```
# Output nicht im Skript  
citation("tidyverse")
```

Es ist eine gute Idee, der Referenz an ein Erweiterungspaket auch noch die Softwareversion hinzuzufügen. Es kann nämlich vorkommen, dass gewisse Berechnungen je nach der Softwareversion ein anderes—eventuell falsches—Ergebnis liefern. Um die Softwareversion von R und eventuell geladenen Erweiterungspaketen abzurufen, können Sie den Befehl `sessionInfo()` verwenden.

```
# Output nicht im Skript  
sessionInfo()
```

1.8 R Markdown-Berichte

R-Skripts können zu Berichten kompiliert werden. Dies ist nützlich, um sicherzustellen, dass Ihre Analysen reproduzierbar sind, denn kompiliert werden nur jene Befehle, die auch im Script

vorhanden sind – und nicht irgendwelche Befehle, die man zwar ausgeführt, aber nicht ins Script aufgenommen hat. Ausserdem kann ein Script nur kompiliert werden, wenn alle Befehle geparkt werden können. Ihre Hausaufgaben sollten Sie daher sowohl als Script als auch in Form eines Berichts einreichen.

Aufgabe 1.17 (Berichtschablone). Speichern Sie die Datei `schablone.R` in den Unterordner `aufgaben` in Ihrem R-Projekt. Kompilieren Sie diesen Bericht: `File, Compile Report....` Vergleichen Sie die `.R`-Datei mit dem kompilierten HTML-Bericht. Für die weiteren Hausaufgaben können Sie dann jeweils diese Schablone als Grundlage verwenden. ◇

Kapitel 2

Arbeiten mit Datensätzen

Die erste Hürde, die es bei einer quantitativen Analyse zu überwinden gilt, ist, die Daten so zu organisieren, dass diese überhaupt analysierbar sind. Hat man dies geschafft, muss man die Daten noch in das Computerprogramm, mit dem man sie analysieren wird (hier: R), einlesen. Öfters muss man die eingelesenen Datensätze anschliessend mit anderen Datensätzen kombinieren und umgestalten und in der Regel will man auch noch gewisse Informationen aus den Datensätzen herauslesen. Dieses Kapitel widmet sich diesen Schritten.

2.1 Daten organisieren

Stellen Sie sich die folgende Datenerhebung vor. Sie möchten untersuchen, wie sich die Fähigkeit, die Bedeutung von Wörtern in einer nicht beherrschten Sprache auf der Basis ihrer Ähnlichkeit zu Wörtern in beherrschten Sprachen zu erschliessen, im Laufe des Lebens verändert. Dazu legen Sie einer Reihe von deutschsprachigen Versuchspersonen unterschiedlichen Alters eine Anzahl geschriebener schwedischer Wörter vor und bitten Sie sie, diese ins Deutsche zu übersetzen. Der Übersichtlichkeit halber werden hier die Übersetzungen von vier Versuchspersonen für fünf Wörter gezeigt, und zwar für jede Versuchsperson in der Reihenfolge, in der die Wörter übersetzt wurden.

- Versuchsperson 1034. Frau, 51 Jahre.
 - Wort: *söka*. Übersetzung: *Socken* (falsch).
 - Wort: *försiktig*. Übersetzung: *vorsichtig* (richtig).
 - Wort: *mjölk*. Übersetzung: *Milch* (richtig).
 - Wort: *behärska*. Keine Übersetzung gegeben.
 - Wort: *fiende*. Übersetzung: *finden* (falsch).
- Versuchsperson 2384. Frau, 27 Jahre.
 - Wort: *fiende*. Keine Übersetzung gegeben.

- Wort: *behärska*. Keine Übersetzung gegeben.
- Wort: *försiktig*. Übersetzung: *vorsichtig* (richtig).
- Wort: *mjölk*. Übersetzung: *Milch* (richtig).
- Wort: *söka*. Übersetzung: *Socke* (falsch).
- Versuchsperson 8667. Frau, 27 Jahre.
 - Wort: *mjölk*. Übersetzung: *Milch* (richtig).
 - Wort: *behärska*. Keine Übersetzung gegeben.
 - Wort: *fiende*. Übersetzung: *finden* (falsch).
 - Wort: *söka*. Übersetzung: *suchen* (richtig).
 - Wort: *försiktig*. Übersetzung: *vorsichtig* (richtig).
- Versuchsperson 5901. Mann, 15 Jahre.
 - Wort: *behärska*. Übersetzung: *beherrschen* (richtig).
 - Wort: *mjölk*. Übersetzung: *milch* (sic.) (richtig).
 - Wort: *försiktig*. Übersetzung: *vorsichtig* (richtig).
 - Wort: *fiende*. Übersetzung: *feinde* (sic.) (richtig; eigentlich *Feind*).
 - Wort: *söka*. Übersetzung: *socken* (sic.) (falsch).

Wie trägt man solche Angaben am besten in ein Spreadsheet ein? Bevor wir uns ein paar Faustregeln anschauen, möchte ich ein bisschen Werbung für ein Spreadsheetprogramm machen, dass Sie vielleicht noch nicht kennen.

Bemerkung 2.1 (Kostenloses Spreadsheetprogramm). LibreOffice.org ist eine kostenlose Applikationssuite, die—wie Microsoft Office—aus einem Textbearbeitungsprogramm (Write), einem Spreadsheetprogramm (Calc), einem Präsentationsprogramm (Impress) usw., besteht. Selber finde ich LibreOffice Calc nützlicher als MS Excel, weil man beim Speichern von Spreadsheets gewisse Einstellungen viel einfacher ändern kann. Darauf werden wir später zurückkommen. ◇

2.1.1 Lange Datensätze sind praktischer als breite

Wir befassen uns in diesem Kurs mit sog. rechteckigen Datensätzen, d.h., Datensätze, in denen die Informationen in Zeilen und Spalten organisiert werden und in denen alle Zeilen und Spalten gleich lang sind. (Beispiele von anderen Datensatzformaten sind XML und JSON.) Die zwei üblichsten Formate, in denen Datensätze organisiert werden, sind das breite und das lange Format. Im breiten Format werden alle Angaben zu einer bestimmten **Erhebungseinheit** (z.B., zu einer Versuchsperson) in die gleiche Zeile eingetragen. In unserem Beispiel könnte ein

	A	B	C	D	E	F	G	H	I	J	K
1	Versuchsperson	Geschlecht	Alter	söka_Position	söka_Übersetzung	söka_richtig	försiktig_Position	försiktig_Übersetzung	försiktig_richtig	mjölk_Position	mjölk_Übersetzung
2	1034	Frau	51	1	Socken	0	2	vorsichtig	1	3	Milch
3	2384	Frau	27	5	Socke	0	3	vorsichtig	1	4	Milch
4	8667	Frau	27	4	suchen	1	5	vorsichtig	1	1	Milch
5	5901	Mann	15	5	socken	1	3	vorsichtig	1	2	milch

Abbildung 2.1: Ein breiter Datensatz mit einer Zeile pro Versuchsperson.

	A	B	C	D	E	F	G	H	I	J
1	Wort	1034	Geschlecht	1034	Alter	1034	Positio	1034	Überse	1034
2	söka	Frau	51	1	Socken	0	Frau	27	5	Socke
3	försiktig	Frau	51	2	vorsichtig	1	Frau	27	3	vorsichtig
4	mjölk	Frau	51	3	Milch	1	Frau	27	4	Milch
5	behärska	Frau	51	4		0	Frau	27	2	
6	fiende	Frau	51	5	finden	0	Frau	27	1	

Abbildung 2.2: Ein breiter Datensatz mit einer Zeile pro Stimulus.

breiter Datensatz wie in Abbildung 2.1 aussehen (7 Spalten werden nicht gezeigt). Bemerken Sie, dass es für jedes Wort drei Spalten gibt: eine, in der steht, an welcher Stelle das Wort übersetzt wurde; eine, in der die Übersetzung steht; und eine, in der vermerkt wird, ob die Übersetzung richtig war. Auch die Anordnung in Abbildung 2.2, in der es eine Zeile pro Wort gibt und alle Übersetzungen für dieses Wort auf der gleichen Zeile stehen, ist ein Beispiel eines breiten Datensatzes (10 Spalten werden nicht gezeigt).

Im langen Format werden die Angaben zu einer **Beobachtungseinheit** in der gleichen Zeile arrangiert. Eine Definition von 'Erhebungseinheit' und 'Beobachtungseinheit' ist schwierig zu geben (siehe Wickham, 2014) und würde ausserdem wenig bringen. In diesem Beispiel wären die Beobachtungseinheiten aber die einzelnen Übersetzungen. Die gleichen Daten im langen Format könnten aussehen wie in Abbildung 2.3. Es ist in der Regel wesentlich einfacher mit langen Datensätzen als mit breiten zu arbeiten. Und falls es trotzdem einmal nötig sein sollte,

	A	B	C	D	E	F	G	H	I	J
	Versuchsperson	Geschlecht	Alter	Position	Wort	Übersetzung	Richtig			
1										
2	1034	Frau	51		1 söka	Socken	0			
3	1034	Frau	51		2 försiktig	vorsichtig	1			
4	1034	Frau	51		3 mjölk	Milch	1			
5	1034	Frau	51		4 behärska		0			
6	1034	Frau	51		5 fiende	finden	0			
7	2384	Frau	27		1 fiende		0			
8	2384	Frau	27		2 behärska		0			
9	2384	Frau	27		3 försiktig	vorsichtig	1			
10	2384	Frau	27		4 mjölk	Milch	1			
11	2384	Frau	27		5 söka	Socke	0			
12	8667	Frau	27		1 mjölk	Milch	1			
13	8667	Frau	27		2 behärska		0			
14	8667	Frau	27		3 fiende	finden	0			
15	8667	Frau	27		4 söka	suchen	1			
16	8667	Frau	27		5 försiktig	vorsichtig	1			
17	5901	Mann	15		1 behärska	beherrschen	1			
18	5901	Mann	15		2 mjölk	milch	1			
19	5901	Mann	15		3 försiktig	vorsichtig	1			
20	5901	Mann	15		4 fiende	feinde	1			
21	5901	Mann	15		5 söka	socken	0			
22										
23										
24										

Abbildung 2.3: Ein langer Datensatz mit einer Zeile pro Stimulus pro Versuchsperson. Lange Datensätze sind in der Regel einfacher zu verwalten und zu analysieren als breite.

mit einem breiten Datensatz zu arbeiten: Lange Datensätze zu breiten zu konvertieren, ist einfacher als umgekehrt; siehe hierzu Abschnitt 2.5.

Um zu vermeiden, dass das absichtliche oder versehentliche Löschen einer Zeile dazu führt, dass die anderen Zeilen nicht mehr interpretiert werden können, werden die Angaben zu den Versuchspersonen in jeder Zeile wiederholt. Lassen Sie weder im langen noch im breiten Format Informationen weg, die man einer anderen Zeile entnehmen kann. Also *nicht* so wie in Abbildung 2.4!

Bemerken Sie auch, dass es eine Spalte gibt, welche die Reihenfolge, in der die Wörter übersetzt wurden, explizit macht. Im Prinzip könnte man diese Information aus der Struktur des Datensatzes ableiten. Indem man diese Information jedoch explizit hinzufügt, vermeidet man, dass sie verloren geht, wenn der Datensatz anders sortiert wird.

Sowohl breite als auch lange Datensätze sind **rechteckig**:

- Sie haben eine Anzahl Zeilen und Spalten. Alle Zeilen sind gleich lang. Dies gilt auch für die Spalten.

	A	B	C	D	E	F	G	H	I	J
	Versuchsperson	Geschlecht	Alter	Position	Wort	Übersetzung	Richtig			
2	1034	Frau	51	1	söka	Socken	0			
3				2	försiktig	vorsichtig	1			
4				3	mjölk	Milch	1			
5				4	behärska		0			
6				5	fiende	finden	0			
7	2384	Frau	27	1	fiende		0			
8				2	behärska		0			
9				3	försiktig	vorsichtig	1			
10				4	mjölk	Milch	1			
11				5	söka	Socke	0			
12	8667	Frau	27	2	mjölk	Milch	1			
13				2	behärska		0			
14				3	fiende	finden	0			
15				4	söka	suchen	1			
16				5	försiktig	vorsichtig	1			
17	5901	Mann	15	1	behärska	beherrschen	1			
18				2	mjölk	milch	1			
19				3	försiktig	vorsichtig	1			
20				4	fiende	feinde	1			
21				5	söka	socken	0			

Abbildung 2.4: Nicht so! In diesem Datensatz wurden mehrere Zellen leer gelassen, da man deren Inhalt anderen Zellen entnehmen kann. Dies wird bei einer Analyse aber zu Schwierigkeiten führen. Ausserdem kann das Löschen einer Zeile dafür sorgen, dass die Infos auf anderen Zeilen nicht mehr rekonstruiert werden können.

- Es gibt keine komplett leeren Zeilen und Spalten. Einzelne leere Zellen gibt es öfters schon, aber es ist nicht so, dass es Angaben in Spalten A–D gibt, überhaupt keine in Spalten E–F und dann wieder welche in Spalte G.
- In der Regel haben alle Spalten einen Namen. Manchmal kommt es zwar vor, dass keine einzige Spalte einen Namen hat, aber geben Sie nicht ein paar Spalten einen Namen und anderen nicht.
- Alle Spaltennamen stehen in *einer* Zeile. Die Spaltenbezeichnungen stellen sich also nicht aus mehreren Zellen zusammen.

Zum Vergleich: Das Spreadsheet in Abbildung 2.5 zeigt einen nicht-rechteckigen Datensatz. Die Zusammenfassungen unten haben in diesem Datensatz nichts zu suchen und würden beim Einlesen zu Problemen führen.

2.1.2 Kurze, aber selbsterklärende Bezeichnungen verwenden

Machen Sie sich die spätere Analyse einfacher, indem Sie den Spalten und anderen Angaben in Ihren Datensätzen deutliche Namen geben. Dadurch vermeiden Sie, dass Sie während der Analyse ständig wieder nachschlagen müssen, was die Angaben überhaupt heissen. Dies verringert wiederum die Wahrscheinlichkeit, dass Sie Fehler machen.

Ein paar Beispiele:

- Sie werten einen Fragebogen aus. Machen Sie in jeder Spalte deutlich, worum es in den Fragen ging. Vermeiden Sie also Spaltennamen wie 'Q3' oder 'Frage8'. Verwenden Sie stattdessen Spaltennamen wie 'DiplomVater' (wenn die Frage war, welchen Schulabschluss der Vater der Gewährsperson hat) oder 'DialectUse' (wenn die Frage war, wie oft die Gewährsperson Dialekt redet).
- Wenn Sie eine Spalte namens 'Geschlecht' haben, die mit Nullen und Einsen gefüllt ist, müssten Sie ständig nachschlagen, ob 0 jetzt für 'Frau' oder 'Mann' steht. Verwenden Sie stattdessen lieber direkt 'Frau' und 'Mann', oder sogar 'f' und 'm'. Eine andere Möglichkeit ist, dass Sie die Spalte zu 'Frau' umbenennen, sodass es deutlich ist, dass eine 1 heisst, dass die Gewährsperson eine Frau war, und eine 0, dass es sich um einen Mann handelte.
- Verwenden Sie eher kurze Bezeichnungen. In der späteren Analyse werden Sie nämlich insbesondere die Spaltennamen mehrmals wieder eintippen müssen. Vermeiden Sie daher Spaltennamen wie 'wie_of_t_sprechen_Sie_Hochdeutsch' und verwenden Sie stattdessen 'use_hochdeutsch' oder Ähnliches.

Am besten verwenden Sie übrigens keine Leertasten und Lesezeichen in den Spaltennamen.

2.1.3 Fehlende Angaben unzweideutig vermerken

Im Beispiel oben habe ich fehlende Übersetzungen einfach leer gelassen. Daraus kann ich ableiten, dass der Versuchsperson das Wort zwar vorgelegt wurde, aber sie dieses nicht übersetzt

	A	B	C	D	E	F	G
1	Versuchsperson	Geschlecht	Alter	Position	Wort	Übersetzung	Richtig
2	1034	Frau	51	1	söka	Socken	0
3	1034	Frau	51	2	försiktig	vorsichtig	1
4	1034	Frau	51	3	mjök	Milch	1
5	1034	Frau	51	4	behärska		0
6	1034	Frau	51	5	fiende	finden	0
7	2384	Frau	27	1	fiende		0
8	2384	Frau	27	2	behärska		0
9	2384	Frau	27	3	försiktig	vorsichtig	1
10	2384	Frau	27	4	mjök	Milch	1
11	2384	Frau	27	5	söka	Socke	0
12	8667	Frau	27	1	mjök	Milch	1
13	8667	Frau	27	2	behärska		0
14	8667	Frau	27	3	fiende	finden	0
15	8667	Frau	27	4	söka	suchen	1
16	8667	Frau	27	5	försiktig	vorsichtig	1
17	5901	Mann	15	1	behärska	beherrschen	1
18	5901	Mann	15	2	mjök	milch	1
19	5901	Mann	15	3	försiktig	vorsichtig	1
20	5901	Mann	15	4	fiende	feinde	1
21	5901	Mann	15	5	söka	socken	0
22							
23	Prozent Männer:	0.25					
24	Durchschnittsalter:	30					
25	Prozent richtig:	0.55					
26							
27							
28							

Abbildung 2.5: Nicht so! Dieser Datensatz ist nicht rechteckig: “Prozent Männer:” ist keine Versuchspersonnummer, und “0.25” ist kein Geschlecht. Ausserdem ist Zeile 22 komplett leer.

hat. Es wäre jedoch auch möglich gewesen, dass einigen Versuchspersonen bestimmte Wörter gar nie vorgelegt wurden, z.B. aufgrund eines Softwarefehlers. Es wäre wichtig, solche Fälle von den ersten zu unterscheiden, indem man diese Fälle mit einer Kürzel (z.B. 'NA' für 'not available' oder 'not applicable') vermerkt.

Gegebenenfalls kann man auch mehrere Kürzel verwenden, um unterschiedliche Gründe für das Nicht-Vorhanden-Seins voneinander zu unterscheiden. In der Regel ist es aber am einfachsten, sämtliche fehlende Daten mit 'NA' zu vermerken und den Grund hierfür in eine Kommentarspalte einzutragen.

Verwenden Sie aber keine Zahlen (wie -99 oder -9999), um fehlende Angaben zu vermerken. Der Grund ist, dass solche Zahlen manchmal zulässige Werte sind. Ausserdem können solche Angaben schwieriger auf den ersten Blick erkannt werden, wenn man eine Zusammenfassung des Datensatzes generiert.

2.1.4 Mehrere kleinere Datensätze sind handlicher als ein riesiger

In den Spreadsheets oben wurden bestimmte Informationen mehrfach wiederholt. Zum Beispiel musste man bei den Übersetzungen von Versuchsperson 1034 fünf Mal eintragen, dass sie eine Frau im Alter von 51 Jahren war. In diesem Fall ist der Datensatz trotz wiederholten Informationen übersichtlich. Wenn man sich aber überlegt, dass in der Regel für jede Versuchsperson viel mehr Informationen vorliegen (z.B., Daten aus einem Hintergrundsfragebogen) und dass man öfters auch Informationen zu den verwendeten Stimuli (hier: den Wörtern) mit einbeziehen will, wird klar, dass man es schnell mit grossen Datensätzen zu tun hat, in denen viele Informationen mehrfach wiederholt werden.

Um die Übersicht zu bewahren und um sich eine Menge Tipp- oder Kopierarbeit zu sparen, lohnt es sich, statt eines grossen Datensatzes mehrere kleinere zu verwalten. In unserem Beispiel würde man dann ein Spreadsheet mit Informationen zu den Versuchspersonen gestalten (Abbildung 2.6). Daneben kann man noch ein Spreadsheet mit Informationen zu den Wörtern anfertigen (Abbildung 2.7). In einem dritten Spreadsheet kann man dann die Antworten bei der Übersetzungsaufgabe aufführen (Abbildung 2.8).

Da im letzten Spreadsheet sowohl eine Spalte mit den Identifikationen der Versuchspersonen als auch mit den Bezeichnungen der Stimuli vorhanden ist, können ihm die Informationen aus den ersten zwei kleineren Datensätzen nachher problemlos hinzugefügt werden. Wie man dies in R machen kann, erfahren Sie in Abschnitt 2.3.

2.1.5 Weitere Bemerkungen

- Beachten Sie Gross- und Kleinschreibung. Für manche Statistikprogramme ist 'Frau' gleich 'frau', für andere (darunter R) nicht.
- Sonderzeichen, wie Umlaute, führen manchmal zu Problemen.
- Beachten Sie Leerzeichen. Für ein Computer ist 'Mann' nicht gleich 'Mann ' (mit Leerzeichen).

	A	B	C	D	E	F	G
1	Versuchsperson	Geschlecht	Alter	Englisch			
2	1034 Frau		51 B2				
3	2384 Frau		27 C1				
4	8667 Frau		27 B2				
5	5901 Mann		15 B1				
6							
7							
8							
9							

Abbildung 2.6: Ein erster Datensatz mit Informationen, die nur die Versuchspersonen betreffen.

	A	B	C	D	E	F
1	Wort	RichtigeÜbersetzung				
2	behärska	beherrschen				
3	fiende	Feind				
4	försiktig	vorsichtig				
5	mjök	milch				
6	söka	suchen				
7						
8						
9						
10						

Abbildung 2.7: Ein zweiter Datensatz mit Informationen, die nur die Stimuli betreffen.

	A	B	C	D	E	F
1	Versuchsperson	Position	Wort	Übersetzung	Richtig	
2	1034	1 söka	1 söka	Socken	0	
3	1034	2 försiktig	2 försiktig	vorsichtig	1	
4	1034	3 mjölk	3 mjölk	Milch	1	
5	1034	4 behärska	4 behärska		0	
6	1034	5 fiende	5 fiende	finden	0	
7	2384	1 fiende	1 fiende		0	
8	2384	2 behärska	2 behärska		0	
9	2384	3 försiktig	3 försiktig	vorsichtig	1	
10	2384	4 mjölk	4 mjölk	Milch	1	
11	2384	5 söka	5 söka	Socke	0	
12	8667	1 mjölk	1 mjölk	Milch	1	
13	8667	2 behärska	2 behärska		0	
14	8667	3 fiende	3 fiende	finden	0	
15	8667	4 söka	4 söka	suchen	1	
16	8667	5 försiktig	5 försiktig	vorsichtig	1	
17	5901	1 behärska	1 behärska	beherrschen	1	
18	5901	2 mjölk	2 mjölk	milch	1	
19	5901	3 försiktig	3 försiktig	vorsichtig	1	
20	5901	4 fiende	4 fiende	feinde	1	
21	5901	5 söka	5 söka	socken	0	
22						

Abbildung 2.8: Ein dritter Datensatz, in dem die Übersetzungen aufgeführt werden.

- Wenn Sie in Ihren Spreadsheets gerne mit Farben arbeiten: Diese gehen verloren, wenn Sie das Spreadsheet in R einlesen. Wenn die Farben Informationen kodieren, die nicht den Angaben im Spreadsheet entnommen werden können, fügen Sie diese Informationen also besser noch selbst hinzu.
- Arbeiten Sie möglichst wenig im Spreadsheet! Nachdem Sie die Daten eingetragen haben, sollten Sie grundsätzlich nicht mehr im Spreadsheet, sondern in R selber arbeiten. Also nicht in Excel herumrechnen, sortieren, kopieren, kleben, neu formatieren usw. Wenn Sie diese Schritte in R ausführen und Ihren Code speichern, ist eindeutig festgelegt, wie Sie den Datensatz umgestaltet haben, um Grafiken zu zeichnen und Modelle zu rechnen. Der ursprüngliche Datensatz bleibt dabei aber unverändert, sodass Sie immer wieder aufs Original zurückgreifen können.

2.2 Datensätze einlesen

Wenn die Daten in einem analysierbaren Format vorliegen, besteht die nächste Herausforderung darin, diese in R einzulesen. Wir behandeln hier nur zwei Fälle: das Einlesen von Excel-Spreadsheets im XLS(X)-Format und das Einlesen von Spreadsheets im CSV-Format.

2.2.1 Excel-Spreadsheets (XLS, XLSX)

Speichern Sie den Datensatz `uebersetzungen.xlsx` im Ordner `data` in Ihrem Arbeitsordner. Dieser Datensatz ist eine Exceldatei, die aus einem einzigen Spreadsheet besteht. Um ihn in R einzulesen, verwenden wir die Funktion `read_excel()` aus dem `readxl`-Package. Dieses Package ist Teil des `tidyverse`-Bündels, das wir bereits installiert haben. Wenn wir seine Funktionen aber verwenden möchten, müssen wir das Package noch laden. Das machen wir mit der Funktion `library()`:

```
library(readxl)
```

Wenn keine Fehlermeldung kommt, ist alles gut.

Installieren müssen Sie Packages übrigens nicht immer wieder, aber Sie müssen die Packages, deren Funktionen Sie verwenden, schon bei jeder neuen Session wieder mit `library()` laden.

Um den Datensatz einzulesen, verwenden wir nun die Funktion `read_excel()`, der wir den Pfad zur Exceldatei übergeben. R akzeptiert sowohl absolute als auch relative Pfade, aber damit Sie bereits jetzt gute Gewohnheiten entwickeln, verweisen wir auf Dateien ab dem Ordner, in der sich die `.Rproj`-Datei des aktuellen Projekts befindet. Das geht am einfachsten mit der `here()`-Funktion aus dem `here`-Package, das wir auch noch laden müssen. Für unsere jetzigen Zwecke ist die Verwendung von `here()` eigentlich übertrieben, aber für grössere Projekte ist es eine sehr nützliche Funktion, sodass wir sie hier sofort verwenden.

```
library(here)
```

```
here() starts at C:/Users/VanhoveJ/switchdrive/statintro2024
```

```
translations <- read_excel(here("data", "uebersetzungen.xlsx"))
```

Wie Sie sehen, erkennt die `here()`-Funktion, in welchem Ordner sich die `.Rproj`-Datei befindet. Bei Ihnen wird dieser Pfad natürlich anders aussehen. Die Alternative ohne `here()` und mit einem relativen Pfad sähe übrigens so aus:

```
translations <- read_excel("data/uebersetzungen.xlsx")
```

Der Vorteil von der `here()`-Funktion ist, dass der Befehl auch genau so funktioniert, wenn das Skript, in dem es vorkommt, in einem Unterordner gespeichert ist: Die einzulesende Datei wird ab dem *project root* gesucht, nicht ab dem Pfad des Skripts.

Angezeigt wird der Datensatz übrigens noch nicht, aber im Fenster rechts oben sollten Sie jetzt ein Objekt namens `translations` sehen, zusammen mit der Angabe '20 obs. of 5 variables'.

Der Datensatz ist nun zugänglich in einem sog. *tibble* namens `translations`.¹

Bemerkung 2.2 (`<-`, `=` und `==`). Die Symbolenfolge `<-` ist der *assignment operator*. Sie kreiert ein neues Objekt im Arbeitsgedächtnis bzw. überschreibt ein bereits vorhandenes Objekt. Dieses Objekt trägt den Namen links von `<-`. Kürzel in RStudio: ALT + -.

Oft wird auch das Ist-Gleich-Zeichen (`=`) als *assignment operator* verwendet. Es wird aber auch verwendet, um Parameter in Funktionen festzulegen (wie Sie bald sehen werden). Zwecks *one form, one function* werde ich daher ausschliesslich `<-` als *assignment operator* verwenden.

Dann gibt es noch die Symbolenfolge `==`. Diese überprüft, ob zwei Werte identisch sind. Beispiel:

```
4 == 2 * 2
[1] TRUE

8 == 2 * 3
[1] FALSE
```

Oft ist es jedoch besser, die Funktion `all.equal()` zu verwenden, um auf Gleichheit zu testen. ◇

Bemerkung 2.3 (Fehlermeldungen). Fehlermeldungen in R sind notorisch unverständlich. Im Anhang finden Sie eine Liste mit den häufigsten Fehlermeldungen sowie möglichen Auslösern und Lösungen. Wenn der Anhang Ihnen nicht weiterhilft, kleben Sie die Fehlermeldung am besten in Google ein.

Wenn Sie die folgende Fehlermeldung erhalten, heisst das, dass R die Datei am falschen Ort gesucht hat.

¹Rechteckige Datensätze heissen in R eigentlich *data frames*. *Tibbles* sind die Entsprechung von *data frames* in den *tidyverse*-Paketen, darunter auch das Paket `readxl`. Wer schon viel Erfahrung mit R hat, wird im Laufe des Skripts vielleicht ein paar subtile Unterschiede zwischen *data frames* und *tibbles* feststellen, aber im Grossen und Ganzen sind sie klein.


```
> translations <- read_excel(here("Data", "uebersetzungen.xlsx"))
Error: 'path' does not exist:
  '/home/jan/ownCloud/StatIntro2022/Data/uebersetzungen.xlsx'
```

Haben Sie die Befehle richtig eingetippt? Haben Sie den Arbeitsordner richtig eingestellt? Haben Sie die Datei am richtigen Ort gespeichert? Hier ist das Problem, dass der Ordner data und nicht Data heisst. ◇

Auch wenn Sie keine Fehlermeldung erhalten, sollten Sie kontrollieren, ob der Datensatz richtig eingelesen wurde. Dazu können Sie beispielsweise die ersten paar Zeilen des Datensatzes anzeigen lassen. Mit dem folgenden Befehl sollten die ersten vier Zeilen gezeigt werden.

```
slice_head(translations, n = 4)
Error in slice_head(translations, n = 4): could not find function "slice_head"
```

Die Fehlermeldung sollte uns nicht beunruhigen. Die Funktion `slice_head()` ist Teil des `dplyr`-Pakets, welches wiederum zum `tidyverse`-Bündel gehört. Dieses Bündel haben wir zwar installiert, aber noch nicht geladen. Laden wir das `tidyverse`-Bündel, so funktioniert die Funktion schon; die Mitteilungen über Attaching packages und Conflicts sind eben nur das: Mitteilungen, keine Fehlermeldungen.

```
library(tidyverse)

- Attaching core tidyverse packages -- tidyverse 2.0.0 -
v dplyr      1.1.2      v readr      2.1.4
v forcats    1.0.0      v stringr    1.5.0
v ggplot2    3.4.2      v tibble     3.2.1
v lubridate  1.9.2      v tidyr      1.3.0
v purrr      1.0.1

- Conflicts ----- tidyverse_conflicts() -
x dplyr::filter() masks stats::filter()
x dplyr::lag()     masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
slice_head(translations, n = 4)

# A tibble: 4 x 5
  Versuchsperson Position Wort      Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>          <dbl>
1      1034         1 söka      Socken          0
2      1034         2 försiktig vorsichtig      1
3      1034         3 mjölk     Milch           1
4      1034         4 behärska <NA>            0
```

Alles scheint in Ordnung zu sein. Bemerken Sie aber, dass leere Zellen als NA vermerkt wurden.

Auch können Sie der Sicherheit halber kontrollieren, ob der Datensatz die richtige Anzahl Zeilen und Spalten zählt:

```
# Anzahl Zeilen
nrow(translations)

[1] 20

# Anzahl Spalten
ncol(translations)

[1] 5
```

Um den ganzen Datensatz in RStudio anzuschauen, können Sie die `View()`-Funktion verwenden:

```
View(translations)
```

Wenn die Daten nicht richtig eingelesen wurden, kontrollieren Sie am besten nochmals, ob das Spreadsheet nach den Regeln der Kunst formatiert wurde.

Für mehr Details zur `read_excel()`-Funktion, siehe <https://readxl.tidyverse.org/>

2.2.2 CSV-Dateien

Ein beliebtes Format, um Datensätze zu speichern und mit anderen zu teilen, ist das CSV-Format. CSV steht für *comma-separated values*: Die Zellen auf der gleichen Zeile werden durch Kommas voneinander getrennt; siehe Abbildung 2.9.

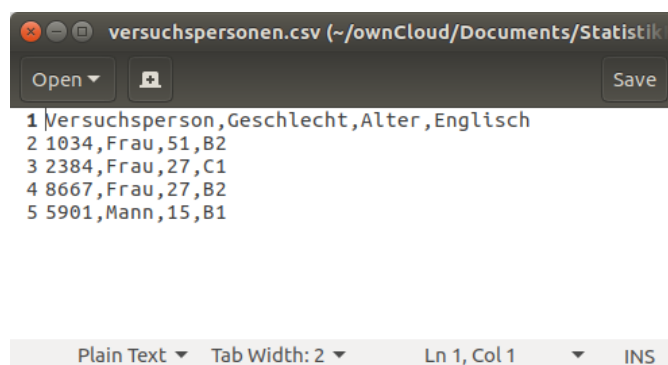


Abbildung 2.9: Ein Datensatz, der als *comma-separated values* gespeichert ist.

In Excel ist es blöderweise eher schwierig, Datensätze im CSV-Format zu speichern: Zwar gibt es diese Option, aber auf deutsch- oder französischsprachigen Systemen werden statt Kommas Semikolonen als Trennzeichen verwendet. In LibreOffice.org hingegen wird man jedes Mal gefragt, ob man Kommas oder Semikolonen verwenden will.

Manchmal werden Texteinträge auch noch zwischen Anführungszeichen gestellt, sodass Kommas auch in einem Textfeld vorkommen können. Die folgenden Funktionen erkennen dies in

der Regel automatisch.² Die `read_csv()`-Funktion gehört zum `readr`-Package, das automatisch geladen wird, wenn das `tidyverse`-Bündel geladen wird.

```
participants <- read_csv(here("data", "versuchspersonen.csv"))

Rows: 4 Columns: 4
- Column specification -----
Delimiter: ","
chr (2): Geschlecht, Englisch
dbl (2): Versuchsperson, Alter

i Use 'spec()' to retrieve the full column specification for this data.
i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

Beim Ausführen dieser Befehle werden ein paar Mitteilungen (keine Fehlermeldungen!) angezeigt, die unter anderem zeigen, dass die `read_csv()`-Funktion erkannt hat, dass in den Spalten Versuchsperson und Alter nur Zahlen stehen ('dbl' für 'double', ein Zahlenformat) und in den Spalten Geschlecht und Englisch auch Buchstaben ('chr' für 'character'). Lesen Sie nun noch den Datensatz mit den zu übersetzenden Wörtern ein; die Mitteilungen, die Sie in der R-Konsole beim Ausführen solcher Befehle sehen werden, werden in diesem Skript nicht mehr angezeigt.

```
items <- read_csv(here("data", "woerter.csv"))
```

Inspizieren Sie die beiden Datensätze `participants` und `items`.

Bemerkung 2.4 (unterschiedliche CSV-Formate). Wenn Sie auf einem französisch- oder deutschsprachigen Computersystem in Excel ein Spreadsheet im 'CSV-Format' speichern, werden die unterschiedlichen Zellen nicht mit Kommas sondern mit Semikolonen voneinander getrennt. Der Grund ist, dass das Komma in diesen Sprachen als Dezimaltrennzeichen dient und daher nicht mehr zur Trennung von Zellen verwendet werden kann. 'CSV'-Dateien, in denen Zellen durch Semikolonen getrennt werden, können Sie in R einlesen, indem Sie statt der Funktion `read_csv()` die Funktion `read_csv2()` verwenden.

In LibreOffice.org kann man für jede Datei selber einstellen, ob Kommas oder Semikolonen zur Trennung von Zellen verwendet werden sollten, und welches Symbol als Dezimaltrennzeichen dienen soll. ◇

2.3 Datensätze zusammenfügen

Wir haben nun drei Datensätze eingelesen: einen mit den Antworten in der Übersetzungsaufgabe (translations), einen mit Informationen zu den zu übersetzenden Wörtern (items), und einen mit Informationen zu den Teilnehmenden (participants). Um die Daten auszuwerten, müssten

²Wenn Sie vorher schon mit R gearbeitet haben, ist die Wahrscheinlichkeit gross, dass Sie statt der `read_csv()`-Funktion (mit `_`) die `read.csv()`-Funktion (mit `.`) verwendet haben. `read.csv()` ist die Einlesefunktion von *base R*; `read_csv()` ist ihre Entsprechung aus dem *tidyverse*.

diese Datensätze miteinander verknüpft werden. Zum Beispiel müssten wir dem Datensatz `translations` drei Spalten mit Informationen zu den jeweiligen Versuchspersonen hinzufügen: Geschlecht, Alter, Englisch. Für Zeilen in `translations`, für die Versuchsperson 1034 ist, sind die Einträge also Frau, 51 respektive B2; ist Versuchsperson = 5901, sind die Einträge Mann, 15 respektive B1. Wenn die gemeinsame Spalte in den beiden Datensätzen identisch heisst, ist dies ein Kinderspiel:

```
all_data <- left_join(x = translations, y = participants)

Joining with 'by = join_by(Versuchsperson)'
```

Verwenden Sie die `View()`-Funktion, um `all_data` zu inspizieren.

Die `left_join()`-Funktion erkennt, dass es in beiden Datensätzen eine Spalte `Versuchsperson` gibt und verwendet diese als 'Reissverschluss'. Wenn die Funktion Schwierigkeiten hat, zu erkennen, welche Variable oder welche Variablen sie als Reissverschluss nehmen soll, kann man diese auch explizit einstellen:

```
all_data <- left_join(x = translations, y = participants,
                     by = "Versuchsperson")
```

Um auch noch Informationen zu den zu übersetzenden Wörtern hinzuzufügen, wiederholen wir den Befehl mit `y = items`. Das Ergebnis dieser Aktion sollten Sie wiederum selber inspizieren.

```
all_data <- left_join(x = all_data, y = items, by = "Wort")
```

Die `left_join()`-Funktion bewirkt, dass alle Einträge in Datensatz `x` bewahrt bleiben und diesem Datensatz die entsprechenden Informationen aus Datensatz `y` hinzugefügt werden, insofern welche vorhanden sind. Wenn es keine Entsprechung in `y` gibt, erscheint in den hinzugefügten Spalten `NA`. Weitere 'join'-Funktionen sind die folgenden; siehe <https://dplyr.tidyverse.org/reference/join.html> für Details:

- `right_join()`: Alle Einträge aus Datensatz `y` bleiben bewahrt; Entsprechungen aus `x` werden hinzugefügt, falls vorhanden.
- `full_join()`: Alle Einträge aus beiden Datensätzen bleiben bewahrt. `NA`, falls es im jeweils anderen Datensatz keine Entsprechung gibt.
- `inner_join()`: Nur Einträge aus Datensatz `x`, für die es eine Entsprechung in `y` gibt, bleiben bewahrt. Diese Entsprechungen werden hinzugefügt.
- `semi_join()`: Nur Einträge aus Datensatz `x`, für die es eine Entsprechung in `y` gibt, bleiben bewahrt. Diese Entsprechungen werden nicht hinzugefügt.
- `anti_join()`: Nur Einträge aus Datensatz `x`, für die es keine Entsprechung in `y` gibt, bleiben bewahrt.

In diesem Beispiel würden `left_join()`, `right_join()`, `full_join()` und `inner_join()` zum

gleichen Resultat führen, aber dies ist nicht immer der Fall.

Aufgabe 2.5 (join-Funktionen). Das Ziel dieser Übung ist es, die Unterschiede zwischen den sechs *join*-Funktionen klarer zu machen.

1. Verwenden Sie den unten stehenden Code, um zwei Objekte (*links* und *rechts*) zu kreieren:

```
links <- tibble(A = c("a", "b", "c", NA),
               B = c(1, 2, NA, 4))

rechts <- tibble(B = c(1, 3, 4, 4),
                C = c(10, NA, 12, 7))
```

2. Inspizieren Sie die beiden neu kreierten Objekte, z.B. mit `View()` oder indem Sie die Objektnamen auf die Konsole eintragen.
3. Sagen Sie vorher, wie das Ergebnis der folgenden Befehle aussehen wird. Kontrollieren Sie erst *danach* Ihre Antwort, indem Sie die Befehle ausführen.

```
left_join(x = links, y = rechts)
right_join(x = links, y = rechts)
full_join(x = links, y = rechts)
inner_join(x = links, y = rechts)
semi_join(x = links, y = rechts)
semi_join(x = rechts, y = links) # Achtung!
anti_join(x = links, y = rechts)
anti_join(x = rechts, y = links) # Achtung!
```

4. Kreieren Sie mit dem folgenden Codeabschnitt wieder zwei Objekte:

```
links <- tibble(A = c("a", "b"),
               B = c(1, NA))
rechts <- tibble(B = c(1, NA, NA),
                C = c(0, 1, 2))
```

Suchen Sie auf der Hilfeseite von `left_join` unter *Arguments* nach der Erläuterung zum Parameter `na_matches`. Sagen Sie vorher, wie der Output der unten stehenden Codeblöcke aussehen wird, und kontrollieren Sie Ihre Antwort.

```
left_join(links, rechts)

left_join(links, rechts, na_matches = "never")
```

Wenn Sie zwei Datensätze zusammenfügen möchten, aber die Variable, die als 'Reissverschluss' dienen soll, in den Datensätzen unterschiedlich heisst, stösst man schnell auf ein Problem:

```
links <- tibble(A = c("a", "a", "b"),
               b = c(1, 2, 3))
rechts <- tibble(B = c(1, 2),
                C = c(10, 38))
left_join(links, rechts)

Error in 'left_join()':
! 'by' must be supplied when 'x' and 'y' have no
common variables.
i Use 'cross_join()' to perform a cross-join.
```

Das Problem ist, dass die Variable, die als Reissverschluss dienen soll, im einen tibble `b` heisst und im anderen `B`. Konsultieren Sie die Hilfeseite von `left_join()` und lösen Sie das Problem.

Hinweis: Schauen Sie auf der Hilfeseite unter `Arguments > by` oder auch unter `Examples`. ◇

2.4 Informationen abfragen

Wir wissen bereits, dass wir mit `View()` einen ganzen *tibble* oder *data frame* (siehe Fussnote 1) inspizieren kann. Um diese auf der Konsole zu zeigen, kann man stattdessen auch einfach den Namen des Objekts eintippen. Wenn der Datensatz zu gross ist, werden dann aber nur einige Zeilen und Spalten gezeigt:

```
all_data

# A tibble: 20 x 9
  Versuchsperson Position Wort Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>      <dbl>
1         1034         1 söka      Socken         0
2         1034         2 försiktig vorsichtig         1
3         1034         3 mjölk      Milch         1
4         1034         4 behärska <NA>         0
5         1034         5 fiende     finden         0
6         2384         1 fiende     <NA>         0
7         2384         2 behärska <NA>         0
# i 13 more rows
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Bei grösseren Datensätzen wird es natürlich auch schwieriger, spezifische Informationen im Datensatz selber nachzuschlagen. Im Folgenden werden daher einige Techniken vorgestellt, um die Suche zu erleichtern.

2.4.1 Zeilen nach Zeilennummer auswählen

Mit diesem Befehl zeigen wir die dritte Zeile des Datensatzes `all_data` an. Am Datensatz ändert sich hierdurch nichts. Wir verlieren die anderen 19 Zeilen also nicht.

```
slice(all_data, 3)

# A tibble: 1 x 9
  Versuchsperson Position Wort Übersetzung Richtig
      <dbl>      <dbl> <chr> <chr>          <dbl>
1         1034         3 mjölk Milch            1
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Eine alternative Schreibweise ist die folgende. Mit der Symbolenfolge `|>` wird das Objekt vor ihr (hier: `all_data`) der Funktion nach ihr als erster Funktionsparameter übergeben:

```
all_data |>
  slice(3)

# A tibble: 1 x 9
  Versuchsperson Position Wort Übersetzung Richtig
      <dbl>      <dbl> <chr> <chr>          <dbl>
1         1034         3 mjölk Milch            1
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Bemerkung 2.6 (`|>`). Die Symbolenfolge `|>` wird *pipe* genannt und wird verwendet, um Befehle übersichtlicher zu organisieren. Sie wird als *dann (then)* ausgesprochen. Für einfache Befehle wie diesen gibt es eigentlich keinen Mehrwert. Aber sobald wir mehrere Befehle kombinieren, ist die Notation mit *pipes* wesentlich einfacher zu lesen und zu verstehen.

Kürzel in RStudio: CTRL + SHIFT + M.



Im Folgenden werden die Ergebnisse der Befehle nicht mehr angezeigt. Probieren Sie die Befehle aber dennoch aus. Bemerken Sie die Verwendung von `c()` (für 'combine') sowie von `:`. Auch können mehrere Befehle verkettet werden.

```
# Zeilen 5 und 7 auswählen.
all_data |>
  slice(c(5, 7))

# Zeilen 5 bis 7 einschliesslich auswählen
all_data |>
  slice(5:7)

# Zeilen 5 bis 7 auswählen, dann vollständig zeigen
```

```
all_data |>
  slice(5:7) |>
  View()
```

Mit den obigen Befehlen werden nur gewisse Zeilen in der Konsole angezeigt. Man kann den Output stattdessen auch als neues Objekt speichern:

```
zeilen7_12 <- all_data |>
  slice(7:12)
```

Jetzt werden die Zeilen nicht angezeigt, aber sie sind fortan verfügbar als neues Objekt im Arbeitsspeicher. Um dieses Objekt zu inspizieren, können Sie seinen Namen eintippen oder View verwenden:

```
zeilen7_12

# A tibble: 6 x 9
  Versuchsperson Position Wort      Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>      <dbl>
1         2384         2 behärska <NA>          0
2         2384         3 försiktig vorsichtig    1
3         2384         4 mjölk    Milch         1
4         2384         5 söka     Socke          0
5         8667         1 mjölk    Milch         1
6         8667         2 behärska <NA>          0
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

2.4.2 Zeilen nach bestimmten Werten auswählen

Mit `slice()` können wir Zeilen nach ihrer Position im Datensatz auswählen. In der Regel wählen wir jedoch die Zeilen nach gewissen Eigenschaften dieser Zeilen aus. Dazu verwenden wir die `filter()`-Funktion. Beispielsweise können wir nur jene Zeilen, die das Wort `fiende` betreffen auswählen. Die Zeichenkombination `==` wird verwendet, um auf Gleichheit zu testen:

```
all_data |>
  filter(Wort == "fiende")

# A tibble: 4 x 9
  Versuchsperson Position Wort      Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>      <dbl>
1         1034         5 fiende finden        0
2         2384         1 fiende <NA>          0
3         8667         3 fiende finden        0
4         5901         4 fiende feinde        1
```



```
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,  
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Um nur die Zeilen auszuwählen, die nicht das Wort `fiende` betreffen, kann man `==` durch `!=` ersetzen.

Wir können auch jene Zeilen auswählen, die Versuchspersonen mit einem Alter über 30 betreffen:

```
all_data |>  
  filter(Alter > 30)  
  
# A tibble: 5 x 9  
  Versuchsperson Position Wort      Übersetzung Richtig  
      <dbl>      <dbl> <chr>      <chr>      <dbl>  
1         1034         1 söka      Socken         0  
2         1034         2 försiktig vorsichtig       1  
3         1034         3 mjölk      Milch         1  
4         1034         4 behärska <NA>          0  
5         1034         5 fiende      finden         0  
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,  
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Für Versuchspersonen unter 30 würde man `<` verwenden; für Versuchspersonen unter 30 einschliesslich `<=`.

Wir können auch nur jene Zeilen beibehalten, für die keine Übersetzung (also mit `NA` als Übersetzung) gegeben wurde. Dann verwendet man aber am besten die Hilfsfunktion `is.na()`:

```
all_data |>  
  filter(is.na(Übersetzung))  
  
# A tibble: 4 x 9  
  Versuchsperson Position Wort      Übersetzung Richtig  
      <dbl>      <dbl> <chr>      <chr>      <dbl>  
1         1034         4 behärska <NA>          0  
2         2384         1 fiende   <NA>          0  
3         2384         2 behärska <NA>          0  
4         8667         2 behärska <NA>          0  
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,  
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Mit `!is.na()` selektieren wir dann wieder nur jene Zeilen, wo die Übersetzung nicht fehlt:

```
all_data |>  
  filter(!is.na(Übersetzung))
```

```
# A tibble: 16 x 9
  Versuchsperson Position Wort      Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>      <dbl>
1         1034         1 söka      Socken         0
2         1034         2 försiktig  vorsichtig      1
3         1034         3 mjölk      Milch         1
4         1034         5 fiende      finden         0
5         2384         3 försiktig  vorsichtig      1
6         2384         4 mjölk      Milch         1
7         2384         5 söka      Socke         0
# i 9 more rows
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Wir können auch mehrere `filter()`-Befehle verketteten. Beispielsweise können wir aus dem Datensatz jene Zeilen auslesen, für die Position gleich 1 ist und für die eine falsche Antwort gegeben wurde:

```
all_data |>
  filter(Position == 1) |>
  filter(Richtig == 0)

# A tibble: 2 x 9
  Versuchsperson Position Wort      Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>      <dbl>
1         1034         1 söka      Socken         0
2         2384         1 fiende      <NA>         0
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Eine Alternative ist diese:

```
# Output nicht im Skript
all_data |>
  filter(Position == 1 & Richtig == 0)
```

Wollen wir die Zeilen auslesen, für die Position gleich 1 ist oder für die eine falsche Antwort gegeben wurde, so verwenden wir diesen Befehl:

```
# Output nicht im Skript
all_data |>
  filter(Position == 1 | Richtig == 0)
```

Die Ergebnisse all dieser Auswahlaktionen können auch als separate Objekte gespeichert und angezeigt werden.

```
nur_fiende <- all_data |>
  filter(Wort == "fiende")

nur_fiende

# A tibble: 4 x 9
  Versuchsperson Position Wort Übersetzung Richtig
      <dbl>      <dbl> <chr>   <chr>      <dbl>
1         1034         5 fiende finden         0
2         2384         1 fiende <NA>         0
3         8667         3 fiende finden         0
4         5901         4 fiende feinde         1
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

2.4.3 Spalten auswählen

Manchmal enthält ein Datensatz schlicht zu viele Spalten, die für die aktuelle Analyse nicht relevant sind. Mit `select()` können wir die Spalten auswählen, die wir gerade brauchen:³

```
all_data |>
  select(Wort, RichtigeÜbersetzung, Übersetzung) |>
  slice_head(n = 5)

# A tibble: 5 x 3
  Wort      RichtigeÜbersetzung Übersetzung
  <chr>      <chr>                <chr>
1 söka      suchen              Socken
2 försiktig vorsichtig        vorsichtig
3 mjölk     milch              Milch
4 behärska  beherrschen           <NA>
5 fiende    Feind              finden
```

Es gibt auch ein paar Hilfsfunktionen, mit denen man effizienter Spalten auswählen kann. Diese sind insbesondere bei grossen Datensätzen nützlich. Beispiele sind `contains()` und `starts_with()`.

```
all_data |>
  select(contains("Übersetzung")) |>
  slice(5:7)

# A tibble: 3 x 2
  Übersetzung RichtigeÜbersetzung
  <chr>        <chr>
```

³Für diejenigen unter Ihnen mit SQL-Erfahrung: Beachten Sie, dass der SQL-Befehl `SELECT` nicht dem R-Befehl `select()`, sondern `filter()` entspricht.

```

1 finden      Feind
2 <NA>        Feind
3 <NA>        beherrschen

all_data |>
  select(starts_with("Richt")) |>
  slice_tail(n = 4)

# A tibble: 4 x 2
  Richtig RichtigeÜbersetzung
  <dbl> <chr>
1      1 milch
2      1 vorsichtig
3      1 Feind
4      0 suchen

```

Für weitere Hilfsfunktionen, siehe <https://tidyselect.r-lib.org>.

2.4.4 Weitere Beispiele

Die unterschiedlichen Befehle können verkettet werden. So können wir nur die Übersetzungen fürs Wort *fiende* abrufen:

```

all_data |>
  filter(Wort == "fiende") |>
  select(Übersetzung)

# A tibble: 4 x 1
  Übersetzung
  <chr>
1 finden
2 <NA>
3 finden
4 fiende

```

Oder wir können mit `distinct()` auch nur die unterschiedlichen Übersetzungen fürs Wort *behärska* abrufen:

```

all_data |>
  filter(Wort == "behärska") |>
  select(Übersetzung) |>
  distinct()

# A tibble: 2 x 1
  Übersetzung
  <chr>
1 <NA>

```

2 beherrschen

Führen Sie auch einmal diesen Befehl ohne `distinct()` aus. Im letzten Beispiel wird auch langsam klar, wieso es sich lohnt, das *pipe* (`|>`) zu verwenden. Ohne sähe diese Befehlskombination nämlich so aus:

```
distinct(select(filter(all_data, Wort == "behärska"), Übersetzung))
```

`filter()` ist der Befehl, der zuerst ausgeführt werden muss, aber in dieser Notation wird er als letzter geschrieben. Mit der *pipe*-Notation schreibt man die Befehle in der Reihenfolge, in der sie ausgeführt werden müssen.

2.5 Datensätze umgestalten

Die meiste Zeit, die man sich für die Analyse eines Datensatzes reserviert, verbringt man oft nicht mit Berechnungen und mit dem Modellieren, sondern mit sog. *data wrangling*: Man muss zunächst einmal dafür sorgen, dass der Datensatz in einem Format vorliegt, in dem er analysiert werden kann. Data wrangling eignet sich wohl am besten für *learning by doing*. Eine Technik, die man aber oft braucht, ist das Konvertieren zwischen langen bzw. längeren und breiten bzw. breiteren Formaten. Diese Technik soll hier illustriert werden anhand eines Datensatzes zu einer Längsschnittstudie zur Entwicklung von Lese- und Schreibfähigkeiten bei Portugiesisch–Französisch- und Portugiesisch–Deutsch-Zweisprachigen (Desgrippes et al., 2017; Pestana et al., 2017).

Aufgabe 2.7 (Daten einlesen). Lesen Sie den Datensatz `helascot_skills.csv` als `skills` ein und inspizieren Sie seine Struktur. ◇

Sie werden bemerken, dass pro Versuchsperson (Subject) pro Zeitpunkt und pro getestete Sprache drei Messungen vorliegen: Reading, Argumentation und Narration. Wir können diesen Datensatz länger machen, indem wir diese Messungen unter- statt nebeneinander stellen. Hierzu verwenden wir die Funktion `pivot_longer()`. Die drei Spalten, die wir dem Parameter `cols` übergeben, werden nun untereinander gestellt; die neue Spalte `Skill` gibt an, aus welcher Spalte die Messungen stammen; die neue Spalte `Score` enthält die Werte, die in den drei ursprünglichen Spalten standen.

```
skills_longer <- skills |>
  pivot_longer(cols = c("Reading", "Argumentation", "Narration"),
               names_to = "Skill", values_to = "Score")
skills_longer

# A tibble: 5,712 x 5
  Subject Time LanguageTested Skill      Score
  <chr>   <dbl> <chr>          <chr>    <dbl>
1 A_PLF_1     1 French      Reading    0.211
2 A_PLF_1     1 French      Argumentation 7
```

```

3 A_PLF_1      1 French      Narration      NA
4 A_PLF_1      1 Portuguese  Reading         0.579
5 A_PLF_1      1 Portuguese  Argumentation   9
6 A_PLF_1      1 Portuguese  Narration       6
7 A_PLF_1      2 French      Reading         0.684
# i 5,705 more rows

```

Jetzt, wo die Daten in diesem noch längeren Format vorliegen, können wir den Datensatz zu einem breiteren Format umgestalten, wo aber die Angaben zu den unterschiedlichen Zeitpunkten (statt zu den unterschiedlichen Fähigkeiten) nebeneinander stehen. Hierzu verwenden wir die Funktion `pivot_wider()`. Da die Werte in der Spalte `Time` numerisch sind, fügen wir ihnen mit dem Parameter `names_prefix` noch ein `T` hinzu:

```

skills_wider_time <- skills_longer |>
  pivot_wider(names_from = "Time", names_prefix = "T", values_from = "Score")
skills_wider_time

# A tibble: 2,100 x 6
  Subject LanguageTested Skill      T1      T2      T3
  <chr>      <chr>         <chr>    <dbl> <dbl> <dbl>
1 A_PLF_1    French          Reading  0.211  0.684  0.947
2 A_PLF_1    French          Argumentation 7      14      14
3 A_PLF_1    French          Narration  NA     10       8
4 A_PLF_1    Portuguese      Reading  0.579  0.737  0.842
5 A_PLF_1    Portuguese      Argumentation 9      13      13
6 A_PLF_1    Portuguese      Narration  6       9      NA
7 A_PLF_10   French          Reading  0.579  0.474  0.316
# i 2,093 more rows

```

Dieses Format wäre zum Beispiel praktisch, wenn wir die Unterschiede zwischen den T1-, T2- und T3-Messungen berechnen möchten. Diese können wir mit dem Befehl `mutate()` noch hinzufügen:

```

skills_wider_time |>
  mutate(ProgressT1_T2 = T2 - T1,
         ProgressT3_T2 = T3 - T2) |>
  select(Subject, LanguageTested, Skill, ProgressT1_T2, ProgressT3_T2)

# A tibble: 2,100 x 5
  Subject LanguageTested Skill ProgressT1_T2 ProgressT3_T2
  <chr>      <chr>         <chr>    <dbl>         <dbl>
1 A_PLF_1    French          Readin~  0.474         0.263
2 A_PLF_1    French          Argum~    7           0
3 A_PLF_1    French          Narra~   NA          -2
4 A_PLF_1    Portuguese      Readin~  0.158         0.105

```

```

5 A_PLF_1 Portuguese Argum~ 4 0
6 A_PLF_1 Portuguese Narra~ 3 NA
7 A_PLF_10 French Readi~ -0.105 -0.158
# i 2,093 more rows

```

Wir könnten auch die Angaben zu den unterschiedlichen Sprachen nebeneinander stellen. Die ersten Versuchspersonen waren alle Portugiesisch–Französisch-Zweisprachige, die nicht auf Deutsch getestet wurden. Daher enthält die letzte Spalte scheinbar nur NA (*not available*), aber mit `View()` können Sie sehen, dass diese Angaben für viele Versuchspersonen tatsächlich vorliegen.

```

skills_wider_language <- skills_longer |>
  pivot_wider(names_from = "LanguageTested", values_from = "Score")
skills_wider_language

# A tibble: 3,999 x 6
  Subject Time Skill      French Portuguese German
  <chr>   <dbl> <chr>      <dbl>      <dbl>   <dbl>
1 A_PLF_1 1 Reading    0.211      0.579     NA
2 A_PLF_1 1 Argumentation 7          9      NA
3 A_PLF_1 1 Narration  NA         6      NA
4 A_PLF_1 2 Reading    0.684      0.737     NA
5 A_PLF_1 2 Argumentation 14         13     NA
6 A_PLF_1 2 Narration  10         9      NA
7 A_PLF_1 3 Reading    0.947      0.842     NA
# i 3,992 more rows

```

Dieses Format wäre dann wieder praktischer, wenn wir pro Versuchsperson die Unterschiede zwischen den French-, Portuguese- und German-Messungen zu jedem Zeitpunkt berechnen möchten:

```

skills_wider_language |>
  mutate(DiffGer_Port = German - Portuguese,
         DiffFre_Port = French - Portuguese) |>
  select(Subject, Time, Skill, DiffGer_Port, DiffFre_Port)

# A tibble: 3,999 x 5
  Subject Time Skill      DiffGer_Port DiffFre_Port
  <chr>   <dbl> <chr>      <dbl>      <dbl>
1 A_PLF_1 1 Reading    NA        -0.368
2 A_PLF_1 1 Argumentation  NA        -2
3 A_PLF_1 1 Narration  NA         NA
4 A_PLF_1 2 Reading    NA       -0.0526
5 A_PLF_1 2 Argumentation  NA         1
6 A_PLF_1 2 Narration  NA         1

```

```
7 A_PLF_1      3 Reading      NA      0.105
# i 3,992 more rows
```

Wir können den Datensatz sogar noch breiter machen:

```
skills_wider_time_language <- skills_longer |>
  pivot_wider(names_from = c("LanguageTested", "Time"),
              values_from = "Score")
skills_wider_time_language

# A tibble: 1,410 x 11
  Subject Skill French_1 Portuguese_1 French_2 Portuguese_2
  <chr>    <chr>    <dbl>         <dbl>    <dbl>         <dbl>
1 A_PLF_1 Read~    0.211         0.579    0.684         0.737
2 A_PLF_1 Argu~     7           9        14          13
3 A_PLF_1 Narr~    NA           6        10           9
4 A_PLF_10 Read~    0.579         0.316    0.474         0.579
5 A_PLF_10 Argu~     5           6        10           7
6 A_PLF_10 Narr~    10           7         8          NA
7 A_PLF_12 Read~    0.895         NA         1         0.947
# i 1,403 more rows
# i 5 more variables: French_3 <dbl>, Portuguese_3 <dbl>,
#   German_1 <dbl>, German_2 <dbl>, German_3 <dbl>
```

Wenn dies nötig wäre, könnten wir diesen breiten Datensatz wieder zum langen Format konvertieren. Langsam wird der Code etwas schwieriger (bei `names_pattern` wird ein sog. regulärer Ausdruck verwendet) und für diesen Kurs ist es nicht so wichtig, dass Sie solche schwierigere Fälle bereits bewältigen können. Vielmehr soll dieser letzte Codeblock illustrieren, dass solche Konversionen möglich sind. Wenn man das weiss, kann man auf der Hilfeseite von `pivot_longer()` (dazu `?pivot_longer` eintippen) nachschauen, wie die Beispiele dort aussehen und diese ans eigene Problem anpassen.

```
skills_back <- skills_wider_time_language |>
  pivot_longer(cols = French_1:German_3,
               names_to = c("Language", "Time"),
               names_pattern = "(.*)_(.*)",
               values_to = "Score")
skills_back

# A tibble: 12,690 x 5
  Subject Skill Language Time Score
  <chr>    <chr>    <chr>   <chr> <dbl>
1 A_PLF_1 Reading French     1  0.211
2 A_PLF_1 Reading Portuguese 1  0.579
3 A_PLF_1 Reading French     2  0.684
```



```

4 A_PLF_1 Reading Portuguese 2      0.737
5 A_PLF_1 Reading French      3      0.947
6 A_PLF_1 Reading Portuguese 3      0.842
7 A_PLF_1 Reading German      1      NA
# i 12,683 more rows

```

Die Notation `French_1:German_3` selektiert übrigens alle Spalten zwischen `French_1` und `German_3` inklusive. Eine Alternative für wenn die Spalten nicht schön praktisch nebeneinander stehen, ist diese:

```

skills_back <- skills_wider_time_language |>
  pivot_longer(cols = starts_with(c("French", "Portuguese", "German")),
               names_to = c("Language", "Time"),
               names_pattern = "(.*)_(.*)",
               values_to = "Score")
skills_back

# A tibble: 12,690 x 5
  Subject Skill   Language   Time   Score
  <chr>   <chr>   <chr>     <chr> <dbl>
1 A_PLF_1 Reading French      1     0.211
2 A_PLF_1 Reading French      2     0.684
3 A_PLF_1 Reading French      3     0.947
4 A_PLF_1 Reading Portuguese  1     0.579
5 A_PLF_1 Reading Portuguese  2     0.737
6 A_PLF_1 Reading Portuguese  3     0.842
7 A_PLF_1 Reading German      1      NA
# i 12,683 more rows

```

2.6 Zusammenfassungen kreieren

Die `summarise()`-Funktion kann man verwenden, um etwa Durchschnitte von Variablen in einem Datensatz zu berechnen. So berechnet der nächste Codeblock die Mittel der Narration- und Argumentation-Variablen im `skills`-Datensatz. Bei der `mean()`-Funktion wird der Parameter `na.rm` noch auf `TRUE` gestellt. Dies bewirkt, dass beim Berechnen des Mittels fehlende Werte (NA) ignoriert werden; andernfalls wären beide Mittel nämlich auch NA.

```

skills |>
  summarise(mittel_narr = mean(Narration, na.rm = TRUE),
            mittel_arg = mean(Argumentation, na.rm = TRUE))

# A tibble: 1 x 2
  mittel_narr mittel_arg
      <dbl>      <dbl>
1      8.51      13.0

```

Mit `group_by()` können wir solche Zusammenfassungen auch für durch die Kombinationen der in dieser Funktion aufgeführten Variablen definierte Untergruppen generieren. Der Codeblock unten spaltet daher zunächst den Datensatz `skills` auf in 9 Untergruppen: eine pro Kombination der Werte von `Time` (1, 2, 3) und `LanguageTested` (French, German, Portuguese). Anschliessend werden die Durchschnitte für jede Untergruppe separat berechnet und in einem tibble zusammengefasst. Die Parametereinstellung `.groups = "drop"` bewirkt, dass der resultierende tibble sich nicht merken muss, wie die Gruppen definiert wurden, aber das ist nicht so wichtig; sie können dies auch weglassen.

```
skills |>
  group_by(Time, LanguageTested) |>
  summarise(mittel_narr = mean(Narration, na.rm = TRUE),
            mittel_arg = mean(Argumentation, na.rm = TRUE),
            .groups = "drop")

# A tibble: 9 x 4
   Time LanguageTested mittel_narr mittel_arg
  <dbl> <chr>          <dbl>    <dbl>
1     1 French        7.79      11.2
2     1 German        6.33      9.46
3     1 Portuguese    8.50     11.4
4     2 French        8.37     13.2
5     2 German        7.06     12.2
6     2 Portuguese    9.16     13.3
7     3 French       10.1     16.3
# i 2 more rows
```

Auch solche Zusammenfassungstibbles können Sie natürlich länger oder—wie hier—breiter machen:

```
skills |>
  group_by(Time, LanguageTested) |>
  summarise(mittel_narr = mean(Narration, na.rm = TRUE),
            .groups = "drop") |>
  pivot_wider(names_from = "Time", names_prefix = "T",
              values_from = "mittel_narr")

# A tibble: 3 x 4
  LanguageTested   T1   T2   T3
  <chr>          <dbl> <dbl> <dbl>
1 French        7.79  8.37 10.1
2 German        6.33  7.06  7.68
3 Portuguese    8.50  9.16 10.2
```

2.7 Viele Wege nach Rom

R bietet einem in der Regel mehrere Möglichkeiten, um das Gleiche zu bewirken. Gerade für die tidyverse-Funktionen `slice()` und `select()` existieren Alternativen, die durchaus praktisch sein können und im Verlauf dieses Skripts auftauchen werden.

Um Zeilen 1 bis 3 von `all_data` zu selektieren, kann man statt

```
all_data |> slice(1:3)
```

auch diese Notation verwenden:

```
all_data[1:3, ]

# A tibble: 3 x 9
  Versuchsperson Position Wort      Übersetzung Richtig
      <dbl>      <dbl> <chr>      <chr>      <dbl>
1         1034         1 söka      Socken         0
2         1034         2 försiktig  vorsichtig     1
3         1034         3 mjölk      Milch         1
# i 4 more variables: Geschlecht <chr>, Alter <dbl>,
#   Englisch <chr>, RichtigeÜbersetzung <chr>
```

Wenn man dahingegen die dritte Spalte auswählen möchte, kann man die Zahl 3 nach der Komma in den eckigen Klammern ausführen:

```
all_data[, 3]

# A tibble: 20 x 1
  Wort
  <chr>
1 söka
2 försiktig
3 mjölk
4 behärska
5 fiende
6 fiende
7 behärska
# i 13 more rows
```

Diese Spalte kann man auch mit seinem Namen (zwischen Anführungszeichen) auswählen:

```
all_data[, "Wort"]

# A tibble: 20 x 1
  Wort
  <chr>
1 söka
```

```

2 försiktig
3 mjölk
4 behärska
5 fiende
6 fiende
7 behärska
# i 13 more rows

```

Beide Ansätze können kombiniert werden. So werden mit dem folgenden Befehl die Zeilen 13 bis 15 der vierten Spalte ausgewählt:

```

all_data[13:15, 4]

# A tibble: 3 x 1
  Übersetzung
  <chr>
1 finden
2 suchen
3 vorsichtig

```

Um die siebte und die dreizehnte Zeile der Spalten namens `Geschlecht` und `Alter` auszuwählen:

```

all_data[c(7, 13), c("Geschlecht", "Alter")]

# A tibble: 2 x 2
  Geschlecht Alter
  <chr>      <dbl>
1 Frau      27
2 Frau      27

```

Mit dem Dollarzeichen kann man ebenso eine Spalte anhand ihres Namens auswählen. Das Ergebnis ist jedoch kein tibble, sondern ein Vektor, d.h., eine Art Liste, in der nur Daten des gleichen Typs vorhanden sind:

```

all_data$Wort

[1] "söka"      "försiktig" "mjölk"      "behärska"
[5] "fiende"    "fiende"     "behärska"   "försiktig"
[9] "mjölk"     "söka"       "mjölk"      "behärska"
[13] "fiende"    "söka"       "försiktig"  "behärska"
[17] "mjölk"     "försiktig" "fiende"     "söka"

```

Um auf das vierzehnte und das achtzehnte Element dieses Vektors zuzugreifen, können wir wieder die Klammernotation verwenden:

```
all_data$Wort[c(14, 18)]
[1] "söka"      "försiktig"
```

Aufgabe 2.8. Slavin et al. (2011) berichten die Ergebnisse einer mehrjährigen Evaluationsstudie, in der zwei Unterrichtsprogramme miteinander verglichen wurden. Schülern und Schülerinnen (SuS) in beiden Programmen wurden unter anderem ein spanischer und ein englischer Vokabeltest vorgelegt, und zwar in der 1., der 2., der 3. und der 4. Klasse. Tabellen 2.1 bis 2.4 auf Seite 54 zeigen einen Teil der Ergebnisse, die Slavin et al. (2011) berichten; es handelt sich dabei um die durchschnittlichen Vokabeltestergebnisse der getesteten SuS.

1. Tragen Sie diese Daten in ein Spreadsheet im langen Format ein. Jede Zeile soll das Ergebnis in einer einzigen Klasse, in einer einzigen Sprache und von einem einzigen Programm enthalten. Sie brauchen also 16 Zeilen mit Daten und eine Zeile mit passenden Spaltennamen.
2. Speichern Sie dieses Spreadsheet im CSV-Format. Lagern Sie diese CSV-Datei in dem Unterordner data in Ihrem Projektordner ab.
3. Lesen Sie das Spreadsheet in R ein.
4. Kontrollieren Sie, ob das Spreadsheet richtig eingelesen wurde.
5. Zeigen Sie in R nun nur die Englischergebnisse im *transitional bilingual*-Programm an.
6. Zeigen Sie die Spanischergebnisse in der 1. und 2. Klasse im *English immersion*-Programm an. ◇

Aufgabe 2.9.

1. Erklären Sie, was der folgende Codeblock bewirkt:

```
d1 <- all_data |>
  filter(Übersetzung == "vorsichtig")
d2 <- all_data |>
  filter(Übersetzung != "vorsichtig")
```

2. Wie viele Zeilen zählen d1 und d2? Wie viele Zeilen zählt all_data? Wie erklären Sie sich dies?
3. Kreieren Sie nun einen tibble d3, der tatsächlich alle Zeilen aus all_data enthält, wo die Versuchsperson das Wort nicht als vorsichtig übersetzt hat. ◇

Aufgabe 2.10.

1. Lesen Sie die Datensätze `helascot_background.csv` und `helascot_skills.csv` in R ein.
2. Nehmen Sie an, Sie bräuchten die Daten im folgenden Format:

- nur die französischen Lesetestergebnisse von Teilnehmenden, die einen Heimatsprache und -kulturkurs belegen (`HLC == "yes"`);
- Versuchspersonen ohne französisches Lesetestergebnis sollten nicht im resultierenden Datensatz vorkommen;
- die Lesetestergebnisse an den drei Zeitpunkten sollten in Spalten nebeneinander stehen;
- der resultierende Datensatz soll für die übrig gebliebenen Versuchspersonen auch noch die Angaben aus dem Datensatz `helascot_background.csv` enthalten.

Gestalten Sie bzw. kombinieren Sie die Datensätze so, dass das Resultat das gewünschte Format hat.

3. Verwenden Sie den umformatierten Datensatz aus Teil (b) und kreieren Sie eine Zusammenfassungstabelle, die den Median (`median()`) der Fortschritte von T1 zu T3 beim französischen Lesetest enthält.

Hinweis: Bei Aufgaben wie diesen ist es zielführender, sich zunächst zu überlegen, welche Schritte auszuführen sind bzw. wie die Zwischenergebnisse auszusehen haben, als sofort in R loszulegen. ◇

2.8 Weiterführende Literatur

Zum Verwalten von Spreadsheets, siehe meinen Blogeintrag zu *Some tips on preparing your data for analysis* (18.6.2015). Broman & Woo (2017) haben weitere nützliche Hinweise.

Das Referenzwerk schlechthin für die Arbeit mit dem tidyverse ist Wickham et al.'s *R for Data Science* und ist gratis verfügbar unter <https://r4ds.hadley.nz/>.

Tabelle 2.1: Vokabeltestergebnisse 1. Klasse.

	Transitional bilingual	English immersion
English	74.98	79.90
Spanish	99.85	90.19

Tabelle 2.2: Vokabeltestergebnisse 2. Klasse.

	Transitional bilingual	English immersion
English	80.40	81.13
Spanish	92.94	87.54

Tabelle 2.3: Vokabeltestergebnisse 3. Klasse.

	Transitional bilingual	English immersion
English	84.76	85.45
Spanish	92.86	85.64

Tabelle 2.4: Vokabeltestergebnisse 4. Klasse.

	Transitional bilingual	English immersion
English	88.07	90.36
Spanish	91.00	86.27

Kapitel 3

Eine einzelne numerische Variable beschreiben

In diesem Kapitel wollen wir illustrieren, wie man eine endliche Liste von Beobachtungen einer einzelnen numerischen Variablen grafisch und in Zahlen zusammenfassen kann. Dafür arbeiten wir mit einem kleinen, aber dafür übersichtlichen Datensatz, der meiner Bachelorarbeit zu Grunde lag. Für diese Arbeit habe ich 23 Studierenden im zweiten Jahr im Fach schwedische Sprach- und Literaturwissenschaft an der Universität Gent vier Leseverstehensaufgaben vorgelegt: einen auf Schwedisch, einen auf Dänisch, einen auf bokmål-Norwegisch und einen auf nynorsk-Norwegisch. Die Ergebnisse aus den unterschiedlichen Lesetests sind nicht miteinander vergleichbar und die nynorsk-Daten sind im Datensatz nicht vorhanden. Daneben gibt es Angaben zu den sonstigen Sprachkenntnissen der Teilnehmenden; diese ignorieren wir hier. Ich gehe davon aus, dass das tidyverse-Bündel und das here-Package geladen sind und dass Sie die Datei `jv_bachpap.csv` in den Ordner `data` in Ihrem R-Projekt abgelegt haben.

```
d <- read_csv(here("data", "jv_bachpap.csv"))
d |>
  slice_head(n = 3)

# A tibble: 3 x 9
  LvlFrench LvlEnglish LvlGerman LvlSpanish NoLanguages
    <dbl>      <dbl>      <dbl>      <dbl>      <dbl>
1         5         5         5         2         5
2         5         4         3         0         3
3         5         5         0         0         2
# i 4 more variables: Swedish <dbl>, Danish <dbl>,
#   Norwegian <dbl>, Participant <chr>
```

In diesem Kapitel widmen wir uns der grafischen und numerischen Beschreibung einer einzelnen numerischen Variablen: den Ergebnissen beim norwegischen Lesetest. Vorübergehend gehen wir davon aus, dass wir uns ausschliesslich für die 23 Ergebnisse im Datensatz interessieren und keine allgemeineren Aussagen machen möchten—z.B. über das Leseverständnis im Norwegischen von Studierenden im zweiten Jahr im Fach schwedische Sprach- und Literatur-

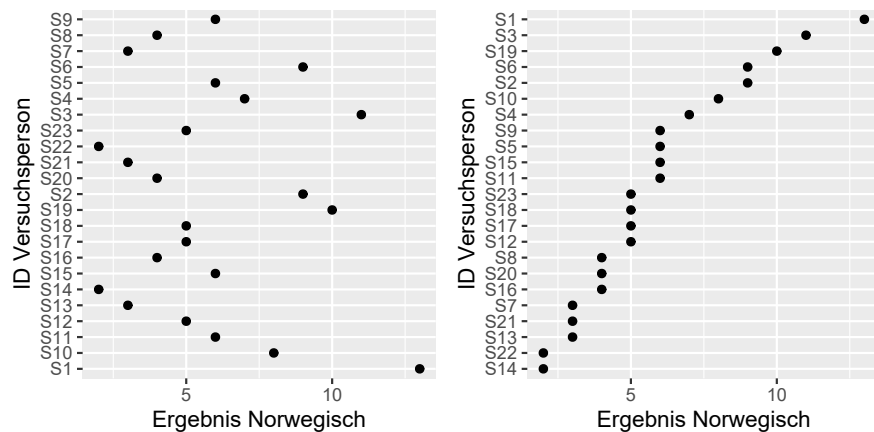


Abbildung 3.1: Ein Punktdiagramm sortiert nach den IDs der Versuchspersonen (links) und eins geordnet nach dem Ergebnis (rechts). Es gibt keine Werte, die weit von anderen liegen.

wissenschaft, die nicht im Datensatz vorhanden sind. Wir betrachten die Ergebnisse, die uns zur Verfügung stehen, also als die ganze, endliche **Population**, für die wir uns interessieren, und nicht als bloss eine **Stichprobe**, d.h., einen Teil der Population, die von Interesse wäre.

3.1 Das Punktdiagramm

Wenn wir über die Ergebnisse kommunizieren wollen und der Datensatz ganz klein ist, könnten wir ihn einfach direkt reproduzieren. Aber sogar für den relativ kleinen Datensatz hier—bloss 23 Beobachtungen—würde es sich lohnen, die Daten grafisch darzustellen und numerisch zusammenzufassen.

Eine erste grafische Darstellung ist das **Punktdiagramm** (oder **Cleveland dot chart**), siehe Abbildung 3.1. Anstatt lediglich die Zahlen im Datensatz aufzulisten, werden die Beobachtungen als Punkte auf separaten Linien entlang der x -Achse dargestellt. Jede Linie wird mit einer ID entlang der y -Achse vermerkt.

Um die linke Grafik zu zeichnen, können Sie den unten stehenden Befehl verwenden. Mittels des `#`-Zeichens habe ich diesem Befehl mit **Kommentaren** versehen. Diese erläutern, wie die Grafik aufgebaut ist. Am Anfang Ihrer R-Karriere empfehle ich Ihnen, Ihren Code reichlich mit Kommentaren auszustatten, sodass Sie ihn mehrere Wochen und Monate später noch verstehen können. Mit der Zeit werden Sie Ihren R-Code immer besser lesen können und dann reichen kargere Kommentare durchaus.

```
ggplot(data = d,                               # Datensatz mit den Variablen
       # aes() = aesthetics = welche Variable wie dargestellt werden soll
       aes(x = Norwegian,                       # Variable auf x-Achse
          y = Participant)) +                  # Variable auf y-Achse
```

```
geom_point() +           # Daten als Punkte darstellen
xlab("Ergebnis Norwegisch") + # insb. bei Arbeiten/Vorträgen/Artikeln:
ylab("ID Versuchsperson")    # Achsen beschriften
```

Achten Sie darauf, dass das tidyverse-Bündel geladen ist: Auch wenn Sie es installiert haben und in einer anderen Session verwendet haben, müssen Sie es in jeder Session erneut laden. Achten Sie weiter auf Gross- und Kleinschreibung, auf die Klammern und Kommas und auf die Pluszeichen. Mit Letzteren werden der Grafik zusätzliche Schichten hinzugefügt. Mit dem Befehl auf den ersten vier Zeilen (`ggplot(...)`) wird lediglich die 'Leinwand' der Grafik gezeichnet. Nach dem Pluszeichen folgt der Befehl `geom_point(...)`, der Punkte auf die Leinwand malt. Mit den Befehlen `xlab(...)` und `ylab(...)` werden Achsenbeschriftungen hinzugefügt bzw. überschrieben. In einem `ggplot()`-Befehl werden die unterschiedlichen Schichten mit einem `+`-Zeichen zusammengefügt, nicht mit einem `pipe` (`|>`).

Um die rechte Grafik zu zeichnen, ersetzen Sie in der 4. Zeile `y = Participant` durch `y = reorder(Participant, Norwegian)` (Klammer nicht vergessen!).

Bemerkung 3.1 (Code mit Stil). Mit dem folgenden Code können Sie ebenfalls die Grafik links zeichnen, denn Leerzeichen und Zeilenbrüche werden von R mehrheitlich ignoriert.

```
ggplot(data=d, aes(x=
Norwegian, y=Participant))+geom_point(
)+xlab("Ergebnis Norwegisch")+ylab("ID Versuchsperson")
```

Die erste Variante ist jedoch viel übersichtlicher, denn die Struktur des Codes (inkl. Einrückungen) widerspiegelt die logische Struktur des Befehls und die Leerzeichen machen den Code lesbarer. Versuchen Sie daher bereits am Anfang Ihrer R-Karriere, einen übersichtlichen und konsistenten Codierstil zu pflegen, etwa indem Sie meinen adoptieren oder sich an einer Gestaltungsrichtlinie (z.B. <https://style.tidyverse.org/>) orientieren. ◇

Bemerkung 3.2 (Grafiken speichern). Nachdem Sie eine Grafik mit `ggplot()` erzeugt haben, können Sie diese mit dem Befehl `ggsave()` speichern. Siehe hierzu `?ggsave`.

Es gibt aber eine allgemeinere Methode, die nicht nur bei von `ggplot()` erzeugten Grafiken funktioniert. Um die linke Grafik aus Abbildung 3.1 zu speichern, können Sie den `ggplot()`-Befehlen zwischen die Befehle `pdf()` und `dev.off()` zu stellen, wie folgt:

```
pdf(here("figs", "dotchart.pdf"),
    width = 5, height = 4) # Höhe und Breite in Zoll
ggplot(data = d,
      aes(x = Norwegian,
          y = Participant)) +
  geom_point() +
  xlab("Ergebnis Norwegisch") +
  ylab("ID Versuchsperson")
```



Abbildung 3.2: Beispiel eines Ausreissers. Hier müsste man kontrollieren, ob der Datenpunkt (17.6) richtig eingetragen wurde und nicht etwa ein Tippfehler ist (statt 1.76).

```
dev.off()
```

Die Abbildung finden Sie jetzt als eine PDF-Datei mit dem Namen `dotchart.pdf` im Unterordner `figs` in Ihrem Projektordner.

Wenn Sie die Grafik lieber in einem anderen Format speichern, können Sie statt `pdf()` auch `svg()`, `png()`, `tiff()` oder `bmp()` verwenden.

Für eine schnelle Grafik können Sie natürlich auch das Export-Menü in der Registerkarte `Plots` im Fenster rechts unten in RStudio verwenden. Aber ich empfehle Ihnen, den Gebrauch von `pdf()` zu umarmen, da Sie so in Ihrem Code dokumentieren, mit welchen Einstellungen die Grafik gespeichert wurde. Das ist nämlich sehr praktisch, wenn Sie später alle Grafiken mit leicht anderen Einstellungen neu zeichnen müssen. ◇

In diesem Beispiel ist das Punktdiagramm insbesondere nützlich aufgrund von dem, was es eben nicht aufzeigt. Es scheint nämlich keine Datenpunkte, oder Grüppchen von Datenpunkten, zu geben, die ziemlich weit von den anderen entfernt liegen. Solche Datenpunkte, die man **Ausreisser** nennt, können das Ergebnis einer Analyse stark beeinflussen, sodass es wichtig ist, zu wissen, dass es sie gibt. Ausreisser können (nicht müssen) auch auf technische Fehler hinweisen und sollten also nochmals kontrolliert werden. Abbildung 3.2 zeigt ein fiktives Beispiel, in dem die Anzahl morphologischer Fehler pro Textseite pro Lerner aufgeführt wird. Ein Datenpunkt liegt so weit von den anderen entfernt, dass man hier auf jeden Fall nochmals kontrollieren sollte, ob die Angabe tatsächlich stimmt, und nicht etwa auf einem Tippfehler bei der Dateneingabe beruht.

3.2 Das Histogramm

Eine zweite nützliche Grafik ist das **Histogramm**. Für ein Histogramm wird die Variable, die man darstellen möchte, in Intervalle (*bins*) aufgeteilt und es wird gezählt, wie viele Beobachtungen es in jedem *bin* gibt. Diese Anzahlen werden in der Grafik als Bälkchen dargestellt, siehe

Abbildung 3.3. Wie in diesem Beispiel sind die *bins* in den allermeisten Histogrammen gleich breit. Die Breite der *bins* muss man selber festlegen; wie die Befehle und Kommentare unten zeigen, ist dies eine Frage von Ausprobieren.

Verglichen mit dem Punktdiagramm ist ein Vorteil des Histogramms, dass auch sehr grosse Datensätze sinnvoll dargestellt werden können, während man in einem Punktdiagramm wohl schnell den Überblick verliert.

```
ggplot(data = d,
       aes(x = Norwegian)) +
  # Defaulteinstellungen fürs Histogramm (immer 30 bins)
  geom_histogram()

ggplot(data = d,
       aes(x = Norwegian)) +
  # Anzahl 'bins' definieren
  geom_histogram(bins = 10)

ggplot(data = d,
       aes(x = Norwegian)) +
  # Binbreite definieren
  geom_histogram(binwidth = 3)

ggplot(data = d,
       aes(x = Norwegian)) +
  # Grenzen selbst festlegen, hier etwa bei 0, 4, 8, 12, 16.
  # Kürzel: seq(from = 0, to = 16, by = 4).
  # Die Farben kann man selbst auswählen.
  geom_histogram(breaks = seq(from = 0, to = 16, by = 4),
                 fill = "lightgreen",
                 colour = "darkgreen") +
  # Achsenbezeichnungen
  xlab("Ergebnisse cloze-Test Norwegisch") +
  ylab("Anzahl Studierende")
```

Aufgabe 3.3. Schauen Sie sich das mittlere Histogramm in Abbildung 3.3 genauer an. Dieses Histogramm verwendet acht Intervalle. Es handelt sich hierbei um halboffene Intervalle, d.h., genau eines der Endpunkte ist jeweils im Intervall eingeschlossen. Handelt es sich aber um die Intervalle

$$(0, 2], (2, 4], \dots, (14, 16],$$

in denen der linke Endpunkt nicht zum Intervall gehört, oder um die Intervalle

$$[0, 2), [2, 4), \dots, [14, 16),$$

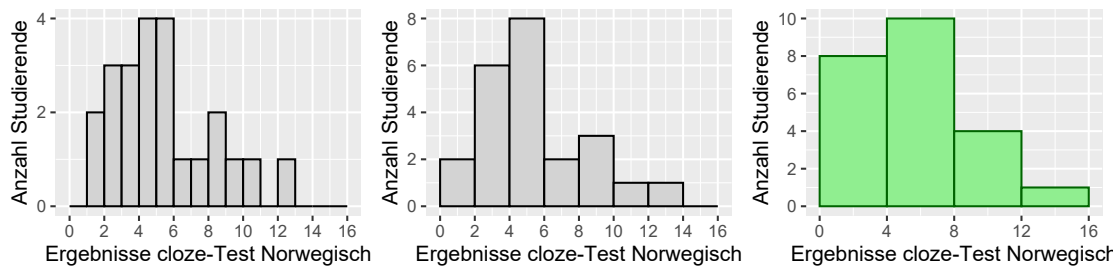


Abbildung 3.3: Drei Histogramme mit den Norwegian-Ergebnissen. *Links:* Die Grenzen zwischen den *bins* liegen bei 0, 1, 2, usw., 15, 16. *Mitte:* Grenzen bei 0, 2, 4, usw., 15, 16. *Rechts:* Grenzen bei 0, 4, 8, 12, 16. Sowohl die Grafik links als auch die in der Mitte halte ich hier für sinnvoll; die Grafik rechts ist nach meinem Geschmack ein bisschen zu grob. Eine goldene Regel für die Wahl der Breite der *bins* gibt es nicht. Daher gilt: Mit den Einstellungen herumspielen und eine nützliche Grafik auswählen.

in denen der rechte Endpunkt nicht zum Intervall gehört? (Hinweis: Schauen Sie sich auch Abbildung 3.1 nochmals an.)

Zeichnen Sie nun das Histogramm nochmals, aber so, dass der jeweils andere Endpunkt zum Intervall gehört. Dazu können Sie der `geom_histogram()`-Funktion einem weiteren Parameter `closed` übergeben. Schlagen Sie auf der Hilfeseite dieser Funktion nach, welche die möglichen Werte für diesen Parameter sind. ◇

In den Histogrammen bisher stand die Anzahl Beobachtungen pro *bin* auf der *y*-Achse. Wenn man unterschiedliche Histogramme (z.B. von unterschiedlichen Gruppen) vergleichen möchte, kann es sinnvoller sein, diese Zahlen zu einer Art relative Frequenz umzurechnen. Auch wenn die Bezeichnung in diesem Kontext nicht ganz stimmt, werden wir diese **Wahrscheinlichkeitsdichten** nennen. Diese werden so berechnet, dass die Gesamtfläche, die das Histogramm einnimmt, 1 beträgt. Anders gesagt: Man nimmt die Höhe jedes *bins*, also die Wahrscheinlichkeitsdichte, und multipliziert diese mit seiner Breite, um die Oberfläche der Bälkchen zu berechnen. Die Höhe wird so festgelegt, dass wenn man alle Oberflächen addiert, die Summe 1 beträgt. Die Form des Histogramms ist aber gleich, egal ob man mit den Anzahlen oder den Wahrscheinlichkeitsdichten arbeitet.

Abbildung 3.4 zeigt ein Beispiel. Die Breite jedes *bins* in der rechten Grafik beträgt 4. Die Höhen sind etwa 0.09, etwa 0.105, etwa 0.045 und fast 0.015, sodass $(4 \cdot 0.09) + (4 \cdot 0.105) + (4 \cdot 0.045) + (4 \cdot 0.015) \approx 1$.

Um die Grafiken in Abbildung 3.4 selber zu zeichnen, ersetzen Sie in den Befehlen oben

```
ggplot(data = d,
       aes(x = Norwegian))
```

durch

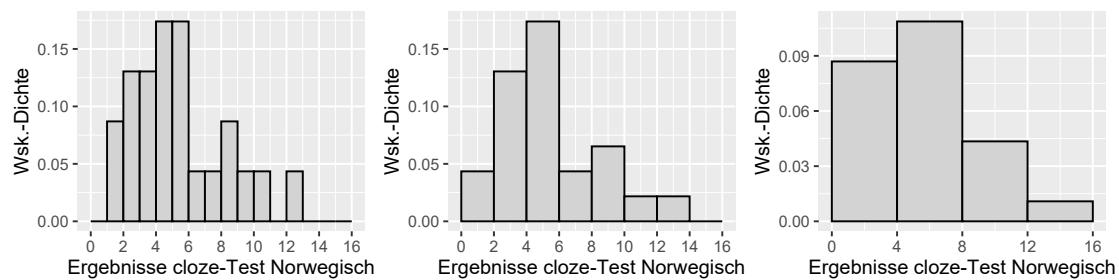


Abbildung 3.4: Drei Histogramme mit der Norwegian-Ergebnissen. Statt der absoluten Anzahl Beobachtungen pro *bin* stehen hier Wahrscheinlichkeitsdichten auf der y-Achse.

```
ggplot(data = d,
       aes(x = Norwegian,
          y = after_stat(density)))
```

Aufgabe 3.4 (Wahrscheinlichkeitsdichten). Nehmen Sie an, wir transformieren die Norwegian-Ergebnisse aus Abbildung 3.4, indem wir diese Ergebnisse zu Prozentsätzen umgestalten: Ein Ergebnis von 10 entspricht einem Prozentsatz von 50; ein Ergebnis von 14 einem Prozentsatz von 70. Wir zeichnen ein Histogramm auf der Basis dieser Prozentsätze und verwenden hierbei vier *bins*, genau wie im rechten Histogramm in Abbildung 3.4. Das heisst, wir verwenden vier Intervalle der Breite 25. Wie wird sich dies auf die Dichtewerte auswirken? Beantworten Sie diese Frage, ohne das Histogramm zu zeichnen. ◇

3.3 Mittelwerte

Es ist unpraktisch, in einem Bericht alle einzelnen beobachteten Werte aufzulisten.¹ Wenn möglich probiert man diese Informationen daher zu komprimieren, indem man berichtet, welcher Wert typisch für diese Beobachtungen ist und wie sehr die einzelnen Beobachtungen von diesem typischen Wert abweichen. Die Frage nach dem typischen Wert betrifft die **zentrale Tendenz** der Beobachtungen und wird anhand von einem Mittelwert beantwortet. Die Frage nach den Abweichungen betrifft die **Streuung** der Beobachtungen und wird anhand von Streuungsmassen beantwortet. Zuerst widmen wir uns den Mittelwerten.

3.3.1 Das arithmetische Mittel

Wenn man vom ‘Durchschnitt’ oder ‘Mittelwert’ spricht, meint man meistens das (arithmetische) Mittel. Eigentlich sind ‘Durchschnitt’ und ‘Mittelwert’ aber Hyperonyme, denn es gibt ausser dem Mittel noch andere Durchschnittsmasse.

¹Aber es ist sehr sinnvoll, den Datensatz online verfügbar zu stellen und im Bericht auf ihn zu verweisen; siehe Klein et al. (2018) und Levenstein & Lyle (2018)! Eine benutzerfreundliche Website, wo Sie dies machen können, ist <https://osf.io>; siehe Soderberg (2018) für eine Anleitung.

Um das Mittel einer endlichen Population zu berechnen, addiert man alle N Werte und teilt man die Summe durch die Anzahl Werte (N). Das Populationsmittel kürzt man in Formeln meistens als μ ab. In Gross-Sigmanotation schaut die Formel so aus:

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i.$$

Wir können eine R-Funktion schreiben, die diese Formel umsetzt; siehe auch Abschnitt 1.3:

```
my_mean <- function(x) {
  N <- length(x)
  my_sum <- 0
  for (i in 1:N) {
    my_sum <- my_sum + x[i]
  }
  return(my_sum / N)
}
my_mean(d$Norwegian)
[1] 5.913043
```

Für das Berechnen von Summen und Mitteln gibt es natürlich eingebaute R-Funktionen, sodass wir diese selbst geschriebene Funktion gar nicht brauchen:

```
sum(d$Norwegian)
[1] 136
mean(d$Norwegian)
[1] 5.913043
```

Die tidyverse-Lösung brauchen wir hier im Prinzip nicht, aber sie zeigt nochmals, wie die `summarise()`-Funktion funktioniert.

```
d |>
  summarise(mittel_norwegisch = mean(Norwegian))
# A tibble: 1 x 1
  mittel_norwegisch
              <dbl>
1                5.91
```

Das Mittel hat ein paar nützliche mathematische Eigenschaften, denen es seine Omnipräsenz verdankt. Eine erste praktische Eigenschaft ist seine **Linearität**: Seien X, Y zwei Vektoren der gleichen Länge N und mit Mitteln μ_X bzw. μ_Y und a, b zwei Konstanten. Wir definieren nun einen dritten Vektor Z der Länge N durch $Z := aX + bY$. Damit ist gemeint, dass $Z_i = aX_i + bY_i$

für $i = 1, \dots, N$. Das Mittel von Z ist nun gerade

$$\mu_Z = a\mu_X + b\mu_Y.$$

Eine zweite praktische Eigenschaft ist die folgende. Sei X ein Vektor der Länge N_X mit Mittel μ_X und Y ein Vektor der Länge N_Y mit Mittel μ_Y . Wir definieren einen dritten Vektor Z der Länge $N_X + N_Y$, indem wir X und Y aufreihen: $Z := (X_1, \dots, X_{N_X}, Y_1, \dots, Y_{N_Y})$. Das Mittel von Z ist nun das **gewichtete Mittel** der Mittel von X und Y :

$$\mu_Z = \frac{N_X\mu_X + N_Y\mu_Y}{N_X + N_Y}.$$

Eine dritte praktische Eigenschaft betrifft den zentralen Grenzwertsatz, den wir uns später anschauen. Ein wesentlicher Nachteil des Mittels ist aber, dass er stark von Ausreissern beeinflusst wird. Nimmt man zum Beispiel die Daten aus Abbildung 3.2 auf Seite 58 und berechnet man ihr Mittel, dann ist das Ergebnis 4.25—ein ziemlich untypischer Wert für die Beobachtungen, sind doch 6 der 7 Beobachtungen unter 2.6 und 1 über 17.5. Lässt man den Ausreisser weg, ist das Mittel 2.02, was der Tendenz des Hauptanteils der Beobachtungen besser entspricht.

3.3.2 Der Median

Ein anderer beliebter Mittelwert ist der **Median**. Es handelt sich hier buchstäblich um den Wert in der Mitte. Genauer ist m_X ein Median des Vektors X , wenn für mindestens der Hälfte der Werte x von X gilt, dass $m_X \leq x$, und wenn für mindestens der Hälfte der Werte in X gilt, dass $m_X \geq x$. Ein Vektor mit einer geraden Anzahl Einträgen kann mehrere (sogar unendlich viele) Mediane haben. In diesem Fall bezeichnet man als *den* Median das Mittel des grössten und des kleinsten solchen Wertes.

Zum Berechnen des Medians ordnet man die Daten von klein nach gross und nimmt man den mittleren Wert. Gibt es eine gerade Anzahl Beobachtungen, gibt es zwei mittlere Werte. Diese Werte, sowie auch alle Zahlen zwischen ihnen, sind Mediane. Um *den* Median zu berechnen, nimmt man einfach das Mittel beider mittlerer Werte.

Zuerst die komplizierte Berechnungsmethode, um den Vorgang zu illustrieren: Mit `arrange()` die Daten von klein nach gross ordnen und dann den mittleren Wert nehmen. Da es 23 Beobachtungen gibt, ist der 12. Wert der mittlere (es gibt 11 kleinere und 11 grössere):

```
d |>
  select(Norwegian) |>
  arrange(Norwegian) |>
  slice(12)

# A tibble: 1 x 1
  Norwegian
    <dbl>
```



```
1      5
```

Einfacher geht es mit `median()`:

```
median(d$Norwegian)

[1] 5
```

Mit der `summarise()`-Funktion können wir eine Zusammenfassungstabelle mit mehreren Werten aufstellen:

```
d |>
  summarise(mittel_norwegisch = mean(Norwegian),
            median_norwegisch = median(Norwegian))

# A tibble: 1 x 2
  mittel_norwegisch median_norwegisch
          <dbl>          <dbl>
1             5.91             5
```

Der Median ist weniger ausreisserempfindlich als das Mittel. Berechnet man für die Werte aus Abbildung 3.2 den Median mit dem Ausreisser, ist das Ergebnis 2.09; ohne ist es 2.04.

Der Median ist zwar linear, aber anhand der Mediane von zwei Vektoren kann man nicht den Median des zusammengelegten Vektors bestimmen. Für den Median gibt es unter bestimmten Umständen eine (schwierigere) Variante des zentralen Grenzwertsatzes.

Grosse Unterschiede zwischen dem Mittel und dem Median sind öfters Ausreissern oder asymmetrischen Verteilungen zuzuschreiben. So oder so gilt: **Keine Mittelwerte berechnen, ohne die Daten zuerst grafisch darzustellen!** Die Berechnung mag stimmen, aber unter Umständen ist sie nicht *sinnvoll*.

3.3.3 Der Modus

Den Modus trifft man weniger oft an, aber er ist eine ganz einfache Art und Weise, um den 'typischen Wert' zu definieren: Es handelt sich schlicht und einfach um den Wert, der am häufigsten vorkommt. Eine Modusfunktion gibt es nicht, aber wir können mit `count()` für jeden Wert zählen, wie oft er vorkommt. Mit `arrange(desc(n))` sortieren wir diese Anzahlen in absteigender Reihenfolge:

```
d |>
  count(Norwegian) |>
  arrange(desc(n))

# A tibble: 11 x 2
  Norwegian      n
      <dbl> <int>
1          5      4
```

```

2      6      4
3      3      3
4      4      3
5      2      2
6      9      2
7      7      1
# i 4 more rows

```

Zwei Werte kommen am häufigsten vor: 5 und 6 kommen beide 4 Mal vor. Die Modi der Norwegischdaten sind also 5 und 6.

Ein wesentlicher Nachteil des Modus ist, dass bei feinkörnigen Daten jede Beobachtung eh nur ein oder zwei Mal vorkommt, sodass es nicht sinnvoll ist, ihn zu berechnen.

3.3.4 Andere Mittelwerte

Es existieren noch andere Mittelwerte z.B. das **harmonische Mittel**, das **geometrische Mittel**, das **winsorisierte Mittel**, das **getrimmte Mittel** und das **gewichtete Mittel**. Diese seien hier bereits der Vollständigkeit halber erwähnt, werden aber noch nicht weiter erläutert. Gerade das winsorisierte und das getrimmte Mittel dienen vor allem dazu, das Mittel einer Population zu schätzen, wenn man bloss mit einer Stichprobe arbeitet.

Aufgabe 3.5. Die Datei `stocker2017.csv` enthält einen Teil der Daten aus einer on-line-Studie von Stocker (2017). 160 Versuchspersonen wurden gebeten, die Glaubwürdigkeit von Aussagen von SprecherInnen mit unterschiedlichen Akzenten (Englisch, Französisch, Deutsch und Italienisch) mithilfe eines *sliders* auf einer Skala von 0 bis 100 zu bewerten. Diese Daten stehen in der `score`-Spalte.

1. Lesen Sie diese Datei in R ein. Kontrollieren Sie, ob dies geklappt hat.
2. Berechnen Sie das Mittel und den Median der `score`-Daten. Sind sich diese Mittelwerte ähnlich?
3. Stellen Sie die `score`-Daten in einem Histogramm mit 10 *bins* dar. Beschreiben Sie die Form dieses Histogramms.
4. Zeichnen Sie ein Histogramm mit 100 bins. Beschreiben Sie dieses Histogramm. Sind das Mittel und der Median gute Masse für die zentrale Tendenz in diesen Daten?
5. Welcher Wert ist der dritthäufigste? Warum, denken Sie?
6. Was ist bei den viert-, fünft-, sechst- usw. -häufigsten Werten auffällig? ◇

Aufgabe 3.6 (harmonisches Mittel). Es sei $x = (x_1, x_2, \dots, x_n)$ ein Vektor mit strikt positiven Zahlen. Das **harmonische Mittel** H dieser Zahlen ist nun definiert als

$$H = \frac{n}{\sum_{i=1}^n \frac{1}{x_i}}.$$

Schreiben Sie eine eigene R-Funktion `harmonic_mean()`, welche einen Vektor mit einer beliebigen Anzahl strikt positiver Zahlen als Parameter erhält und sein harmonisches Mittel ausspuckt.

Hinweis: Wenn Sie den ersten Kilometer gegen 5 Kilometer/Stunde überbrücken, den zweiten mit 10 Kilometern/Stunde und den dritten mit 2 Kilometern/Stunde, dann haben Sie insgesamt 3 Kilometer in 48 Minuten überbrückt. (Man rechne nach.) Dies entspricht einer durchschnittlichen Geschwindigkeit von 3.75 Kilometern/Stunde. Diese durchschnittliche Geschwindigkeit können Sie mit dem harmonischen Mittel berechnen. Wenn Ihre Funktion gut geschrieben ist, sollte der folgende Befehl die Antwort 3.75 ergeben:

```
harmonic_mean(c(5, 10, 2))
```

```
[1] 3.75
```



3.4 Streuungsmasse

Wenn man nur einen oder ein paar Mittelwerte berichtet, bleibt die Frage unbeantwortet, wie stark die einzelnen Beobachtungen davon abweichen. Mit Streuungsmassen versucht man diese Abweichung numerisch auszudrücken.

3.4.1 Spannweite

Ein einfaches Streuungsmass ist die **Spannweite**. Man berechnet lediglich den niedrigsten und den höchsten Wert und berichtet diese oder den Unterschied zwischen ihnen:

```
min(d$Norwegian)
```

```
[1] 2
```

```
max(d$Norwegian)
```

```
[1] 13
```

```
range(d$Norwegian)
```

```
[1] 2 13
```

Dieses Mass wird aus gutem Grund selten verwendet: Es ist extrem ausreisserempfindlich. Ausserdem unterschätzt die Spannweite einer Stichprobe systematisch die Spannweite der Population, aus der sie stammt.² (Mit Stichproben beschäftigen wir uns in späteren Kapiteln.)

²Diese Tatsache scheint übrigens in der Zweitspracherwerbsforschung nicht allen Forschenden bekannt zu sein, die diesem Streuungsmass eine zentrale Rolle in ihrer Forschung zuteilen (Vanhove, 2020).

3.4.2 Summe der Quadrate

Wenn wir alle Beobachtungen ins Streuungsmass einfließen lassen wollen, scheint es auf den ersten Blick sinnvoll, die Unterschiede zwischen den beobachteten Werten und dem Mittel zu berechnen und diese Unterschiede beieinander aufzuzählen: $(x_1 - \mu) + (x_2 - \mu) + \dots$. Diese Summe ist aber immer 0. Um das zu sehen, überlege man sich Folgendes:

$$(x_1 - \mu) + (x_2 - \mu) + \dots + (x_N - \mu) = (x_1 + x_2 + \dots + x_N) - N\mu.$$

Nun wird μ berechnet als $\frac{1}{N}(x_1 + x_2 + \dots + x_N)$, sodass $N\mu = x_1 + x_2 + \dots + x_N$. Daher gilt

$$(x_1 + x_2 + \dots + x_N) - N\mu = (x_1 + x_2 + \dots + x_N) - (x_1 + x_2 + \dots + x_N) = 0.$$

Man kann auch direkt die Linearität des Mittels anwenden, um zum gleichen Schluss zu kommen.

Um dieses Problem zu lösen, werden die Unterschiede zwischen den beobachteten Werten und dem Mittel quadriert, bevor sie beieinander aufgezählt werden. Dadurch werden sie alle positiv, sodass ihre Summe nicht länger 0 ist. Diese **Summe der Quadrate** wird in Formeln als d^2 oder *SS* (*sum of squares*) abgekürzt:

$$d^2 = \sum_{i=1}^N (x_i - \mu)^2.$$

```
sum((d$Norwegian - mean(d$Norwegian))^2)
```

```
[1] 187.8261
```

Achten Sie auf die Stelle der Klammern und der Quadrierung: Die Unterschiede müssen quadriert werden, nicht die Summe oder das Mittel.

```
# Falsch: Hier wird die Summe quadriert.
```

```
sum((d$Norwegian - mean(d$Norwegian)))^2
```

```
[1] 7.888609e-29
```

```
# Falsch: Hier wird das Mittel quadriert.
```

```
sum((d$Norwegian - mean(d$Norwegian)^2))
```

```
[1] -668.1739
```

Vielleicht fragen Sie sich, wieso man hier mit quadrierten Unterschieden arbeitet. Wäre es nicht einfacher, mit den absoluten Unterschieden zu rechnen? Streuungsmasse, die auf den absoluten Unterschieden basieren, gibt es tatsächlich, aber das Arbeiten mit quadrierten Unterschieden bietet mathematische Vorteile (z.B. zentralen Grenzwertsatz).

Bemerkung 3.7 (Gleitkommazahlen). Wir haben zwar gezeigt, dass immer

$$\sum_{i=1}^N (x_i - \mu) = 0$$

gilt. Aber wenn wir diese Berechnung in R kontrollieren, erhalten wir ein anderes Ergebnis:

```
sum(d$Norwegian - mean(d$Norwegian))
[1] 8.881784e-15
```

8.8818e-15 bedeutet $8.8818 \cdot 10^{-15}$, also 0.0000000000000088818. Aber eigentlich sollte das Ergebnis genau 0 sein, nicht *fast* 0. Das Problem liegt bei der Genauigkeit, mit der ein Computer mit Zahlen umgeht. Wir werden auf dieses Problem hier nicht näher eingehen, da es für uns nicht von riesiger Bedeutung ist. ◇

3.4.3 Varianz

Ein Problem mit d^2 ist, dass Datensätze unterschiedlicher Grösse nicht vergleichbar sind: Je mehr Beobachtungen es gibt, desto grösser ist d^2 . Das Mass d^2 drückt also sowohl die Grösse des Datensatzes als auch die Streuung der Beobachtungen aus, was unerwünscht ist. Die Lösung liegt auf der Hand: d^2 teilen durch die Anzahl Beobachtungen. Dies ergibt die **Populationsvarianz** (σ^2):

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2. \quad (3.1)$$

In der Regel müssen wir die Varianz einer Stichprobe, nicht jene einer Population berechnen. Diese wird leicht anders berechnet; siehe Kapitel 5.

Aufgabe 3.8 (Funktion für die Populationsvarianz). In R gibt es keine eingebaute Funktion, um die Populationsvarianz einer endlichen Population zu berechnen. Schreiben Sie eine Funktion `pop_var()`, um diese Lücke zu füllen. ◇

3.4.4 Standardabweichung

Varianzen sind nicht einfach zu interpretieren, da sie aufgrund der Quadrierung in der Berechnung in quadrierten Einheiten ausgedrückt werden (z.B. quadrierte Testergebnisse, quadrierte Sprecher per Sprache). Wir können aber die Wurzel der Varianz nehmen, was die Populationsstandardabweichung ergibt (σ):

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}.$$

In R mit der selbst geschriebenen `pop_var()`-Funktion (Aufgabe 3.8):

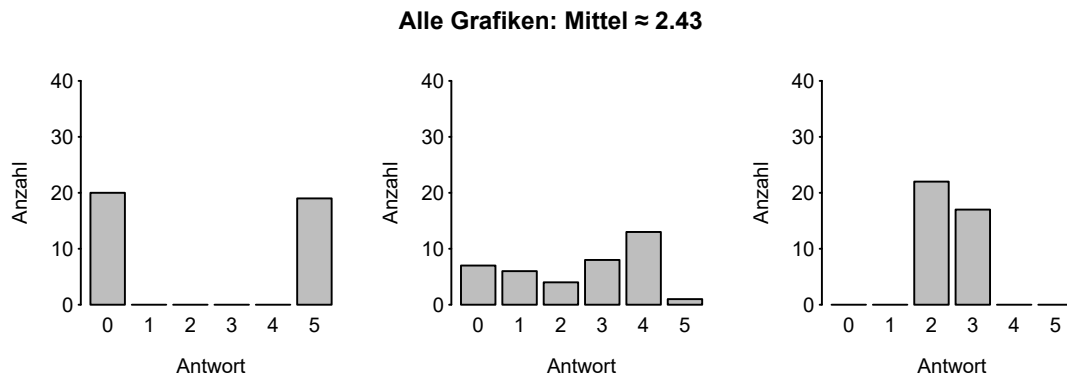


Abbildung 3.5: Hinter dem gleichen Mittelwert kann sich eine Vielzahl von unterschiedlichen Mustern verstecken. Diese drei Grafiken zeigen alle 39 Beobachtungen einer Variablen mit einem Mittel von 2.43 (nach Rundung).

```
pop_var(d$Norwegian) |>
  sqrt()
[1] 2.857683
```

Standardabweichungen und Varianzen kann man (wie Mittelwerte) nicht absolut interpretieren: Eine Standardabweichung von 0.4 ist je nach der Art von Daten klein, gross oder unauffällig, und dies gilt auch für Standardabweichungen von 8'000. So wäre etwa eine Standardabweichung von 13 unauffällig, wenn es sich um in Zentimetern gemessenen Körpergrössen von Menschen handeln würde; erstaunlich klein, wenn die Körpergrössen in Millimetern ausgedrückt wären; und ziemlich gross, wenn sie in Zoll ausgedrückt wären.

In der Regel müssen wir die Standardabweichung einer Stichprobe, nicht jene einer Population berechnen. Diese wird leicht anders berechnet; siehe Kapitel 5.

Bemerkung 3.9 (Daten nicht nur numerisch zusammenfassen!). Stellen Sie sich vor, dass Sie in einer Studie lesen, dass 39 Versuchspersonen eine Frage auf einer 6er-Skala von 0 bis 5 beantwortet haben und das Mittel der Antworten 2.43 betrug. Vielleicht stellen Sie sich dann darunter vor, dass die meisten Versuchspersonen sich für '2' oder '3' entschieden. Dies muss aber nicht der Fall sein: Hinter diesem Mittelwert können sich viele andere Datenmuster verstecken, die zu anderen Schlussfolgerungen führen sollten. Vielleicht sind sich die Versuchspersonen einig in ihrer Gleichgültigkeit, weshalb sie alle Antworten in der Mitte der Skala wählen. Oder vielleicht handelt es sich um ein sehr kontroverses Thema mit überzeugten Gegnern und Befürwortern aber ohne eine moderate Mitte. Oder vielleicht sind alle Arten von Meinung etwas vertreten; siehe Abbildung 3.5.

Wenn ein Streuungsmass berichtet wird, schränkt sich die Anzahl möglicher Muster zwar ein, aber trotzdem können sich hinter einem Mittel und einer Standardabweichung mehrere

Alle Grafiken: Mittel ≈ 2.43 , SD ≈ 1.05

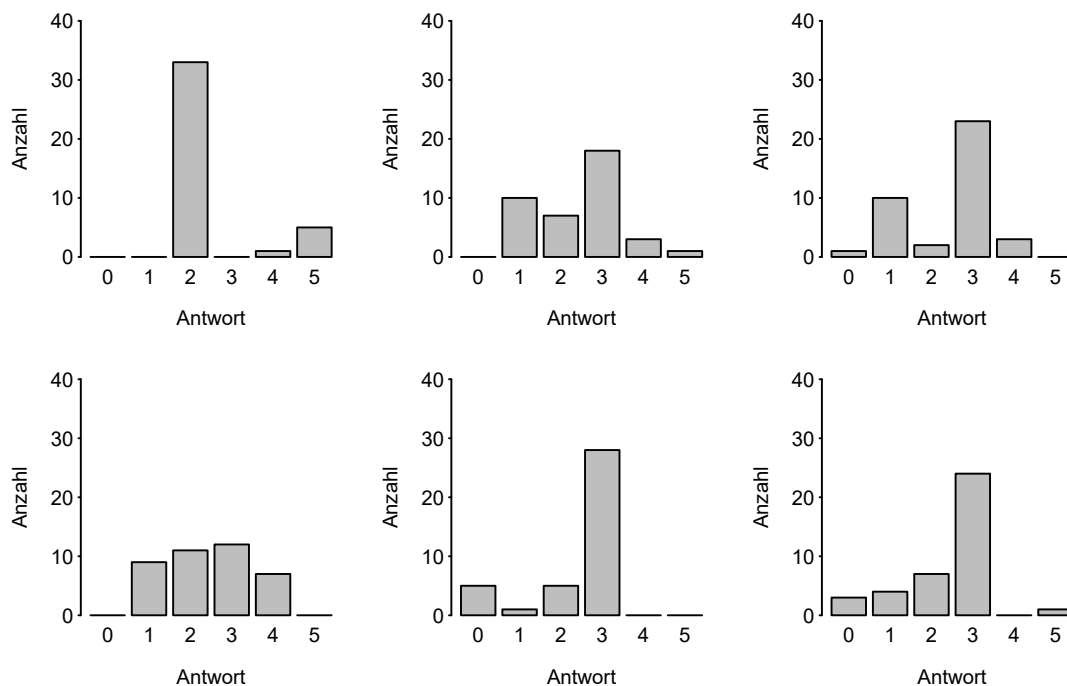


Abbildung 3.6: Sogar wenn man das Mittel und die Standardabweichung kennt, weiss man noch nicht, welches Muster sich hinter diesen Zahlen versteckt. In all diesen Grafiken beträgt das Mittel nach Rundung 2.43 und die Standardabweichung 1.05.

Verteilungen verstecken (Abbildung 3.6).³



Merksatz! Stellen Sie Ihre Daten auch grafisch dar, sodass Sie und Ihre Leserschaft wissen, wie diese überhaupt aussehen. Mittelwerte und Streuungsmasse erzählen nicht die ganze Geschichte.

3.5 Kerndichteschätzungen

In unserem Norwegischbeispiel gibt es es nur eine geringe Anzahl Beobachtungen und ausserdem ist die Variable nicht sehr feinkörnig. Eine sehr feinkörnige Variable wäre eine Variable mit sehr vielen möglichen Ergebnissen und höchstens einem Beleg pro möglichen Wert.

Was würde nun passieren, wenn wir eine grosse Anzahl Beobachtungen (z.B. 100'000) von einer sehr feinkörnigen Variablen erheben würden, diese Beobachtungen in einem Histogramm mit Wahrscheinlichkeitsdichten (siehe Abschnitt 3.2) darstellen würden und die Anzahl bins

³Diese Verteilungen wurden generiert anhand des R-Codes unter <http://bayesfactor.blogspot.ch/2016/03/how-to-check-likert-scale-summaries-for.html>.

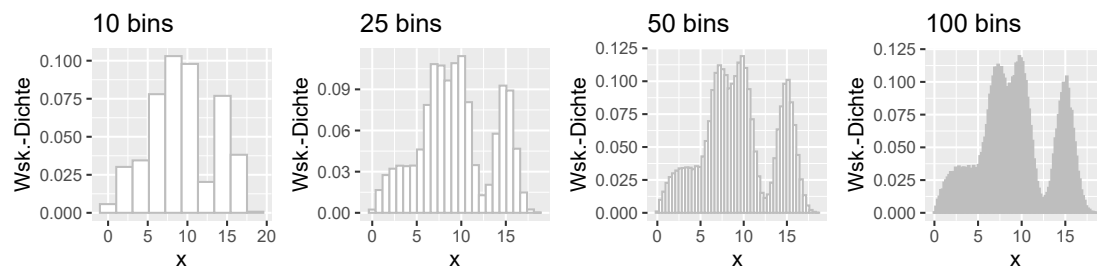


Abbildung 3.7: Vier Histogramme der gleichen feinkörnigen Variablen.

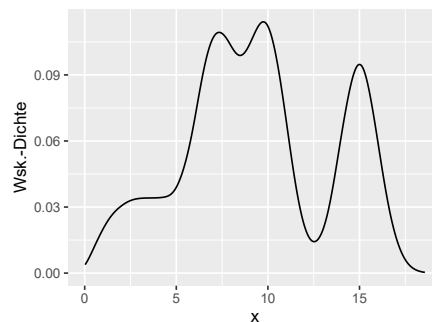


Abbildung 3.8: Eine Kerndichteschätzung der gleichen Variablen wie in Abbildung 3.7.

immer vergrössern würden? Wenn die Anzahl *bins* gross wird, können wir sie kaum mehr mehr voneinander unterscheiden, wie Abbildung 3.7 zeigt. Ausserdem werden wir irgendwann nur noch *bins*, die entweder eine oder keine einzige Beobachtung beinhalten, haben.

In solchen Fällen arbeitet man stattdessen mit Kerndichteschätzungen. Die Berechnungsmethode braucht uns hier nicht zu interessieren (es gibt mehrere); grundsätzlich handelt es sich um ein geglättetes Histogramm, bei dem die Wahrscheinlichkeitsdichten jedes Bälkchens mit einer Kurve verbunden werden und die Bälkchen selber nicht mehr dargestellt werden; siehe Abbildung 3.8.

Eine Kerndichteschätzung können Sie mit dem Befehl `geom_density()` zeichnen; siehe die Beispiele unter https://ggplot2.tidyverse.org/reference/geom_density.html. Den Beispielen auf dieser Seite kann man entnehmen, dass man die Kerndichte einer Variablen auf mehrere Arten schätzen kann, sodass es nicht *die* Kerndichteschätzung einer bestimmten Variablen gibt. Vergleichen Sie dazu die erste, dritte und vierte Grafik, die alle Kerndichteschätzungen der gleichen Variablen darstellen.

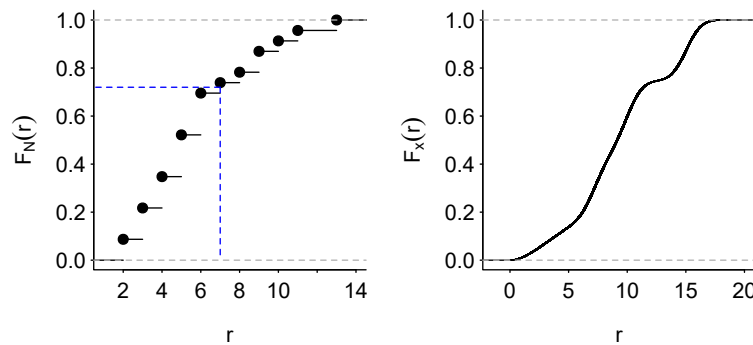


Abbildung 3.9: Die Verteilungsfunktionen der norwegischen Lesedaten (links) und der feinkörnigen Variablen aus Abbildung 3.8.

3.6 Verteilungs- und Quantilfunktionen

Für einen numerischen Vektor X drückt die **Verteilungsfunktion** F_X für jede Zahl aus, welche Proportion der Einträge von X nicht grösser als dieser Wert ist. Das heisst,

$$F_X(r) := \frac{1}{N} \sum_{i=1}^N \mathbb{1}(X_i \leq r),$$

wobei N die Anzahl Einträge in X ist und $\mathbb{1}(X_i \leq r) = 1$, falls $X_i \leq r$ und $\mathbb{1}(X_i \leq r) = 0$ sonst.

Abbildung 3.9 zeigt die Verteilungsfunktion der norwegischen Lesedaten (F_N , links) und der feinkörnigen Variablen x aus dem letzten Abschnitt (F_x , rechts). Diese Abbildung wurde mit der Funktion `ecdf()` gezeichnet. Obwohl man ähnliche Grafiken mit `ggplot2` zeichnen kann, zeichne ich Verteilungsfunktionen lieber ohne, da `ggplot2` defaultmässig Verteilungsfunktionen stetig (lückenlos) aussehen lässt. Die linke Verteilungsfunktion ist aber klar nicht-stetig; auch die rechte Grafik ist nicht-stetig, nur sind die Lücken kleiner.

Die **Quantilfunktion** F_X^{-1} beantwortet die umgekehrte Frage. Für eine Proportion $p \in (0, 1)$ ist $F_X^{-1}(p)$ der kleinste Wert q , sodass $F_X(q) \geq p$. Die blauen Strichellinien in Abbildung 3.9 (links) zeigen, dass der kleinste Wert, für den gilt, dass mindestens 72% der Lesetestergebnisse unter ihm liegen, 7 ist. Also $F_N^{-1}(0.72) = 7$.

In R gibt es zwar eine `quantile()`-Funktion, aber diese ist eigentlich dafür gedacht, dass man anhand einer Stichprobe die Quantile der Population schätzt. Wie Sie der Hilfeseite dieser Funktion entnehmen können, gibt es verschiedene Möglichkeiten, wie diese Schätzung ausgeführt werden kann. Der Vollständigkeit halber gebe ich hier eine R-Funktion an, die das p -te Quantil einer endlichen Population x berechnet:

```
pop_quantile <- function(x, p) {
  x <- x[!is.na(x)]
  cum_prop <- cumsum(table(x)) / length(x)
```

```
  return(which(cum_prop > p)[1] |> names() |> as.numeric())
}
pop_quantile(d$Norwegian, 0.43)
[1] 5
pop_quantile(d$Norwegian, 0.72)
[1] 7
```

3.7 Weiterführende Literatur

Huff (1954) (*How to lie with statistics*) ist ein kurzes und sehr lesbares Büchlein. Es behandelt unter anderem die unterschiedlichen Mittelwerte und wie diese manipulativ eingesetzt werden.

In diesem Skript werden zwar mehrere nützliche Arten von Grafiken vorgestellt, aber eine ausführlichere Behandlung finden Sie bei Healy (2019). Dieses Buch ist auch kostenlos verfügbar unter <https://socviz.co/>.

Kapitel 4

Grundrisse der Stochastik

In diesem Kapitel schauen wir uns einige der für die quantitative Datenanalyse wichtigsten mathematischen Konzepte an.

4.1 Wahrscheinlichkeitsräume

4.1.1 Ereignisse und Wahrscheinlichkeiten

Jass ist eine Familie von Kartenspielen, die mit einem Blatt von 36 Karten gespielt werden: jeweils 6 bis Ass in jeder der vier Farben Herz, Schaufel, Ecke und Kreuz. Wir stellen uns nun zwei einfache **Zufallsexperimente** vor:

1. Im ersten Versuch ziehen wir zufällig eine Karte aus dem Blatt.
2. Im zweiten Versuch ziehen wir zufällig zwei Karten aus dem Blatt, und zwar die eine nach der anderen. Wir merken uns dabei, welche Karte als erste gezogen wurde.

Es ist klar, dass der erste Versuch 36 mögliche Ergebnisse hat. Der zweite Versuch hat $36 \cdot 35 = 1260$ mögliche Ergebnisse, denn die erste Karte ist eine von 36 möglichen und die zweite eine der 35 übrigen.

In beiden Versuchen können wir nun Ja-/Nein-Fragen zum Ergebnis stellen. Im ersten Versuch wären dies einige der möglichen Ja-/Nein-Fragen:

- Handelt es sich um $7\clubsuit$?
- Ist das Ergebnis in der folgenden Liste enthalten: $7\heartsuit, D\spadesuit, A\spadesuit$?
- Steht eine gerade Zahl auf der Karte?
- Steht eine gerade Zahl auf der Karte oder ist die Karte schwarz (\spadesuit, \clubsuit)?

Mit 'oder' ist übrigens immer das inklusive 'oder' gemeint. Für das exklusive 'oder' verwenden wir Ausdrücke wie 'A oder B, aber nicht beide'.

Im zweiten Versuch wären dies einige der möglichen Ja-/Nein-Fragen:

- Ist die erste Karte $7\clubsuit$ und die zweite $8\heartsuit$?
- Ist eine der beiden Karten schwarz?
- Ist die Zahl auf der ersten Karte niedriger als die Zahl auf der zweiten?
- Ist die Summe der Zahlen auf den Karten gerade?
- Handelt es sich um zwei Asse?

Wenn wir eine solche Ja-/Nein-Frage mit 'ja' beantworten, so sagen wir, dass das entsprechende Ereignis eingetreten ist. In der Wahrscheinlichkeitsrechnung werden Ergebnisse und Ereignisse mithilfe der Mengentheorie formalisiert.

Definition 4.1 (Ergebnisse und Ereignisse). Ein **Ergebnis** ist ein mögliches Resultat eines Zufallsexperiments. (Den Begriff Zufallsexperiment lassen wir hier undefiniert.)

Die Menge aller Ergebnisse eines Zufallsexperiments nennen wir den **Grundraum** dieses Zufallsexperiments. Dieser Grundraum wird oft als Ω bezeichnet.

Ein **Ereignis** ist eine Teilmenge des Grundraums. Anders gesagt ist ein Ereignis eine Menge von Ergebnissen, wobei diese Menge nicht alle Ergebnisse enthalten muss und sogar leer sein darf. Das leere Ereignis bezeichnet man als \emptyset oder $\{\}$.

Ein Ereignis der Form $\{\omega\}$, wo $\omega \in \Omega$ ein Ergebnis ist, bezeichnet man als **Elementarereignis**.

◇

In unserem ersten Versuch enthält der Grundraum Ω_1 36 Elemente:

$$\begin{aligned}\Omega_1 = \{ & 6\clubsuit, 7\clubsuit, 8\clubsuit, 9\clubsuit, 10\clubsuit, B\clubsuit, D\clubsuit, K\clubsuit, A\clubsuit, \\ & 6\diamondsuit, 7\diamondsuit, 8\diamondsuit, 9\diamondsuit, 10\diamondsuit, B\diamondsuit, D\diamondsuit, K\diamondsuit, A\diamondsuit, \\ & 6\heartsuit, 7\heartsuit, 8\heartsuit, 9\heartsuit, 10\heartsuit, B\heartsuit, D\heartsuit, K\heartsuit, A\heartsuit, \\ & 6\spadesuit, 7\spadesuit, 8\spadesuit, 9\spadesuit, 10\spadesuit, B\spadesuit, D\spadesuit, K\spadesuit, A\spadesuit \}.\end{aligned}$$

Die Ja-/Nein-Frage *Steht eine gerade Zahl auf der Karte?* entspricht dem Ereignis

$$A := \{6\clubsuit, 8\clubsuit, 10\clubsuit, 6\diamondsuit, 8\diamondsuit, 10\diamondsuit, 6\spadesuit, 8\spadesuit, 10\spadesuit, 6\heartsuit, 8\heartsuit, 10\heartsuit\},$$

das 12 Elemente zählt. Die Ja-/Nein-Frage *Ist die Karte schwarz?* entspricht dem Ereignis

$$B := \{6\clubsuit, 7\clubsuit, \dots, A\clubsuit, 6\spadesuit, 7\spadesuit, \dots, A\spadesuit\},$$

das 18 Elemente zählt. Die Ja-/Nein-Frage *Steht eine gerade Zahl auf der Karte oder ist die Karte*

schwarz entspricht nun der mengentheoretischen **Vereinigung** von A und B :

$$\begin{aligned} A \cup B &= A \text{ oder } B \\ &= \{\omega \in \Omega_1 : \omega \in A \text{ oder } \omega \in B\} \\ &= \{6\clubsuit, 7\clubsuit, \dots, A\clubsuit, 6\spadesuit, 7\spadesuit, \dots, A\spadesuit, 6\diamondsuit, 8\diamondsuit, 10\diamondsuit, 6\heartsuit, 8\heartsuit, 10\heartsuit\}. \end{aligned}$$

Das Ereignis $A \cup B$ enthält 24 Ergebnisse. Die Ja-/Nein-Frage *Steht eine gerade Zahl auf der Karte und ist die Karte schwarz* entspricht dahingegen dem mengentheoretischen **Schnitt** von A und B :

$$\begin{aligned} A \cap B &= A \text{ und } B \\ &= \{\omega \in \Omega_1 : \omega \in A \text{ und } \omega \in B\} \\ &= \{6\clubsuit, 8\clubsuit, 10\clubsuit, 6\spadesuit, 8\spadesuit, 10\spadesuit\}. \end{aligned}$$

Die Ja-/Nein-Frage *Steht keine gerade Zahl auf der Karte?* entspricht dann wieder dem mengentheoretischen **Komplement** von A :

$$\begin{aligned} A^c &= \text{nicht } A \\ &= \{\omega \in \Omega_1 : \omega \notin A\}. \end{aligned}$$

Wir beantworten die Ja-/Nein-Frage genau dann mit ‘ja’, wenn das tatsächliche Ergebnis ω im entsprechenden Ereignis E enthalten ist, also wenn $\omega \in E$.

Im zweiten Zufallsexperiment besteht der Grundraum Ω_2 aus den 1260 Tupeln, die aus zwei ungleichen Karten bestehen. Die Ja-/Nein-Frage *Handelt es sich um zwei Asse?* entspricht nun dem Ereignis

$$\begin{aligned} C = \{ & (A\clubsuit, A\diamondsuit), (A\clubsuit, A\heartsuit), (A\clubsuit, A\spadesuit), (A\diamondsuit, A\clubsuit), (A\diamondsuit, A\heartsuit), (A\diamondsuit, A\spadesuit), \\ & (A\heartsuit, A\clubsuit), (A\heartsuit, A\diamondsuit), (A\heartsuit, A\spadesuit), (A\spadesuit, A\clubsuit), (A\spadesuit, A\diamondsuit), (A\spadesuit, A\heartsuit) \}. \end{aligned}$$

Definition 4.2 (diskrete Wahrscheinlichkeitsräume). Ein **diskreter Wahrscheinlichkeitsraum** besteht aus einem ‘auflistbaren’¹ Grundraum Ω und einer Abbildung (Funktion) \mathbb{P} , die jedem Ereignis eine Zahl zwischen 0 und 1 (beide inklusive) zuordnet. Diese Abbildung legt für jedes Ereignis fest, zu welcher Wahrscheinlichkeit es eintritt. Dazu muss die Abbildung \mathbb{P} folgende Bedingungen (Axiome) genügen:

1. Für jedes Ereignis E gilt $\mathbb{P}(E) \geq 0$. Das heisst, Wahrscheinlichkeiten sind nicht-negativ.

¹In der Mathematik spricht man von ‘abzählbar’. Jede endliche Menge kann aufgelistet werden und ist damit abzählbar. Auch bestimmte unendliche Mengen können aufgelistet werden. Beispielsweise kann man die Menge der natürlichen Zahlen $\mathbb{N} = \{0, 1, 2, \dots\}$ problemlos auflisten. Auch die Ganzzahlen \mathbb{Z} können aufgelistet werden: $\mathbb{Z} = \{0, 1, -1, 2, -2, \dots\}$. Sogar die rationalen Zahlen (Bruchzahlen) \mathbb{Q} können (etwas mühsamer) aufgelistet werden:

$$\begin{aligned} \mathbb{Q} = \{ & 0/1, 1/1, -1/1, 1/2, -1/2, 2/1, -2/1, 1/3, -1/3, 3/1, -3/1, \\ & 1/4, -1/4, 2/3, -2/3, 4/1, -4/1, 3/2, -3/2, \dots \}. \end{aligned}$$

Die reellen Zahlen \mathbb{R} können jedoch nachweisbar nicht aufgelistet werden.

2. $\mathbb{P}(\Omega) = 1$. Mit anderen Worten: Die Wahrscheinlichkeit, *irgendein* Ergebnis zu beobachten, liegt bei 1.
3. Sei A_1, A_2, A_3, \dots eine (endliche oder unendliche) Liste von Ereignissen. Wenn jedes Ergebnis in höchstens einem der aufgelisteten Ereignisse vorkommt, so ist die Wahrscheinlichkeit, dass *irgendeines* der aufgelisteten Ereignisse eintritt, gerade die Summe der Wahrscheinlichkeiten der einzelnen Ereignisse. In Symbolen: Wenn für jedes $i \neq j$ gilt, dass $A_i \cap A_j = \emptyset$, so gilt

$$\mathbb{P}\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathbb{P}(A_i).$$

◇

Lemma 4.3 (Rechenregeln für Wahrscheinlichkeiten). Aus den Bedingungen an der Abbildung \mathbb{P} können ein paar wichtige Rechenregeln abgeleitet werden.

1. Für jedes Ereignis A gilt $A \cup A^c = \Omega$ und $A \cap A^c = \emptyset$. Daher gilt

$$1 = \mathbb{P}(\Omega) = \mathbb{P}(A \cup A^c) = \mathbb{P}(A) + \mathbb{P}(A^c),$$

also

$$\mathbb{P}(A^c) = 1 - \mathbb{P}(A).$$

Da $\Omega^c = \emptyset$, gilt insbesondere $\mathbb{P}(\emptyset) = 0$.

2. Für zwei Ereignisse A, B bezeichnen wir mit $A \setminus B$ das Ereignis, dass das Ergebnis in A aber nicht in B liegt:

$$A \setminus B = \{\omega \in \Omega : \omega \in A, \omega \notin B\}.$$

Dann gilt

$$A \cup B = (A \setminus B) \cup (B \setminus A) \cup (A \cap B),$$

wobei $A \setminus B, B \setminus A, A \cap B$ nicht überlappen. Somit gilt

$$\mathbb{P}(A \cup B) = \mathbb{P}((A \setminus B) \cup (B \setminus A) \cup (A \cap B)) = \mathbb{P}(A \setminus B) + \mathbb{P}(B \setminus A) + \mathbb{P}(A \cap B).$$

Nun gilt auch $A = (A \setminus B) \cup (A \cap B)$ und $B = (B \setminus A) \cup (A \cap B)$. Folglich

$$\begin{aligned} \mathbb{P}(A \cup B) &= \mathbb{P}(A \setminus B) + \mathbb{P}(B \setminus A) + \mathbb{P}(A \cap B) \\ &= \mathbb{P}(A) - \mathbb{P}(A \cap B) + \mathbb{P}(B) - \mathbb{P}(A \cap B) + \mathbb{P}(A \cap B) \\ &= \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cap B) \\ &\leq \mathbb{P}(A) + \mathbb{P}(B). \end{aligned}$$

In Worten: Die Wahrscheinlichkeit, dass irgendeines von zwei Ereignissen eintritt, ist die Summe der Wahrscheinlichkeiten der beiden Ereignisse minus die Wahrscheinlichkeit, dass sie beide eintreffen.

3. In diskreten Wahrscheinlichkeitsräumen lässt sich jedes Ereignis schreiben als die Vereinigung einer Liste von Elementarereignissen. Diese Elementarereignisse überlappen nicht. Also wird die Wahrscheinlichkeit jedes Ereignisses bereits festgelegt von den Wahrscheinlichkeiten der Elementarereignissen. \diamond

Beispiel 4.4 (Diskrete Gleichverteilung auf Ω_1). Wir nehmen das erste Zufallsexperiment wieder auf und nehmen an, dass jede Karte die gleiche Wahrscheinlichkeit hat, gezogen zu werden. Das heisst, $\mathbb{P}(\{\omega\}) = 1/36$ für jedes $\omega \in \Omega_1$. Wir sagen in diesem Fall, dass \mathbb{P} eine **Gleichverteilung** auf Ω_1 ist.

Das Ereignis A ('Auf der Karte steht eine gerade Zahl') zählt 12 Ergebnisse und tritt folglich zu einer Wahrscheinlichkeit von $12 \cdot 1/36 = 1/3$ ein. Das Ereignis B ('Die Karte ist schwarz') zählt 18 Ergebnisse, sodass $\mathbb{P}(B) = 18/36 = 1/2$. Das Ereignis $A \cap B$ zählt 6 Ergebnisse, sodass $\mathbb{P}(A \cap B) = 6/36 = 1/6$. Folglich gilt $\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cap B) = 2/3$. \diamond

Beispiel 4.5 (Zipfverteilung). Wir betrachten eine Sprache mit einem endlichen Wortschatz Ω . Die Wörter bezeichnen wir ihrer Frequenz absteigend nach mit $\omega_1, \omega_2, \dots, \omega_n$. Hierbei gehen wir davon aus, dass diese Frequenzen paarweise unterschiedlich sind. Das Zipfmodell besagt, dass die relative Häufigkeit eines Wortes in einer Sprache (in etwa) umgekehrt proportional zu seinem Frequenzrang ist, also dass

$$\mathbb{P}(\{\omega_k\}) = C_n \frac{1}{k}$$

für irgendeine von n abhängige Konstante C_n und für alle $k = 1, \dots, n$. Es muss gelten, dass

$$1 = \mathbb{P}(\Omega) = \mathbb{P}\left(\bigcup_{k=1}^n \{\omega_k\}\right) = \sum_{k=1}^n C_n \frac{1}{k} = C_n \sum_{k=1}^n \frac{1}{k}.$$

Also

$$C_n = \frac{1}{\sum_{k=1}^n \frac{1}{k}} = \frac{1}{H_n}$$

wo H_n die n -te harmonische Zahl ist. Für $n = 10$ erhalten wir laut diesem Modell folgende relative Häufigkeiten:

```
n <- 10
woerter <- 1:n
Hn <- sum(1/woerter)
tibble(Wort = woerter,
       rel.Häufigkeit = 1/Hn * 1/woerter)

# A tibble: 10 x 2
  Wort rel.Häufigkeit
<int>         <dbl>
1     1           0.341
2     2           0.171
```

3	3	0.114
4	4	0.0854
5	5	0.0683
6	6	0.0569
7	7	0.0488

i 3 more rows

Dieses Modell geht übrigens deswegen von einem endlichen Wortschatz aus, weil $\sum_{k=1}^{\infty} \frac{1}{k}$ gegen unendlich divergiert. \diamond

Beispiel 4.6 (unendlicher diskreter Wahrscheinlichkeitsraum). Wir stellen uns einen Zufallsgenerator vor, der jede Zahl $k = 1, 2, 3, \dots$ zu einer Wahrscheinlichkeit von $1/2^k$ ausspuckt. Der Grundraum besteht nun aus allen natürlichen Zahlen $k \geq 1$. Da

$$\sum_{k=1}^{\infty} \mathbb{P}(\{k\}) = \sum_{k=1}^{\infty} \frac{1}{2^k} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots = 1,$$

handelt es sich hier um einen gültigen diskreten Wahrscheinlichkeitsraum. Die Wahrscheinlichkeit, dass man eine gerade Zahl generiert, beträgt

$$\sum_{k=1}^{\infty} \mathbb{P}(\{2k\}) = \sum_{k=1}^{\infty} \frac{1}{2^{2k}} = \sum_{k=1}^{\infty} \frac{1}{4^k} = \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \dots = \frac{1}{3}.$$

\diamond

Diskrete Wahrscheinlichkeitsräume lassen sich relativ einfach beschreiben, denn sie werden komplett charakterisiert von den Elementarereignissen und den Wahrscheinlichkeiten, die wir ihnen zuordnen. Für viele Anwendungen sind diskrete Wahrscheinlichkeitsräume jedoch ungeeignet, da sich der Grundraum besser als die Menge der reellen Zahlen (\mathbb{R}) auffassen lässt. Diese Menge kann jedoch nicht aufgelistet werden. Es stellt sich nun heraus, dass wir in solchen Fällen nicht länger beliebige Teilmengen des Grundraums als Ereignisse betrachten können und sinnvolle Wahrscheinlichkeitsaussagen über diese machen können. Auf die Gründe dafür wollen wir hier nicht eingehen. Praktisch gesehen hält sich das Problem glücklicherweise in Grenzen, denn die Teilmengen, die man nicht als Ereignis betrachten kann, sind recht schwierig zu konstruieren und für uns nicht relevant. In der nächsten Definition verzichten wir daher auf eine nähere Beschreibung der Eigenschaften, welche die Familie von Ereignissen, über die man Wahrscheinlichkeitsaussagen machen kann, erfüllen muss.

Definition 4.7 (allgemeine Wahrscheinlichkeitsräume). Ein **allgemeiner Wahrscheinlichkeitsraum** besteht aus (1) einer (auflistbaren oder nicht-auflistbaren) Grundmenge Ω , (2) einer Familie von Ereignissen, über die man Wahrscheinlichkeitsaussagen machen kann, und (3) einer Abbildung \mathbb{P} , die jedem dieser Ereignisse eine Wahrscheinlichkeit zuordnet. Die Abbildung \mathbb{P} hat die gleichen Eigenschaften wie in Definition 4.2. \diamond

Beispiel 4.8 (kontinuierliche Gleichverteilung). Die Kreislinie eines Glücksrads ist wie in

Abbildung 4.1 mit Zahlen von 0 bis 360 (exklusive) vermerkt. Jedes Mal, wenn der Pfeil gedreht wird, bleibt er an einer zufälligen Stelle auf der Kreislinie stehen. Wir können mit beliebiger Genauigkeit ablesen, bei welchem Zahlenwert der Pfeil stehen bleibt.

Der Grundraum in diesem Beispiel besteht aus allen reellen Zahlen im Intervall $[0, 360)$. Diese Menge kann zwar nicht aufgelistet werden, aber über alles, was wir sinnvollerweise als Ereignis betrachten möchten, können wir dennoch Wahrscheinlichkeitsaussagen machen. So ist die Wahrscheinlichkeit, dass der Pfeil irgendwo zwischen den Zahlen a und b ($a, b \in [0, 360), a \leq b$) landet, proportional zur Länge des Intervalls $[a, b)$, also $b - a$. Da $\mathbb{P}([0, 360)) = 1$ gelten muss, folgt

$$\mathbb{P}([a, b)) = \frac{b - a}{360}.$$

Beispielsweise beträgt die Wahrscheinlichkeit, dass der Pfeil zwischen 45 und 93 stehen bleibt $\frac{93-45}{360} \approx 0.133$.

Etwas kontraintuitiv beträgt in diesem Beispiel die Wahrscheinlichkeit, dass der Pfeil an einem bestimmten Wert $a \in [0, 360)$ stehen bleibt, genau 0, und dies für alle $a \in [0, 360)$. Dies liegt daran, dass ein einzelner Punkt die Länge 0 hat. Aus der dritten Eigenschaft von \mathbb{P} folgt nun, dass sogar die Wahrscheinlichkeit, dass der Pfeil an irgendeinem Punkt, der in einer unendlich langen Liste vorkommt, stehen bleibt, genau 0 beträgt. Beispielsweise gilt

$$\mathbb{P}(\{1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \dots\}) = 0.$$

Dies beisst sich nicht mit der Bedingung, dass $\mathbb{P}([0, 360)) = 1$ gelten soll: Das Intervall $[0, 360)$ kann nicht als eine solche Liste geschrieben werden.

Eine Konsequenz von der Tatsache, dass in diesem Beispiel individuelle Punkte Wahrscheinlichkeit 0 haben, ist, dass man ohne Bedenken die Randpunkte den Intervallen hinzufügen darf oder sie weglassen kann:

$$\mathbb{P}([a, b]) = \mathbb{P}((a, b)) = \mathbb{P}([a, b)) = \mathbb{P}((a, b]).$$

◇

4.1.2 Unabhängigkeit

Definition 4.9 (Unabhängigkeit (1)). Wir sagen, dass zwei Ereignisse A und B **unabhängig** sind, falls

$$\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$$

gilt. Salopper sagt man in diesem Fall auch, dass A und B unabhängig *voneinander* sind oder dass A unabhängig von B ist (oder umgekehrt). ◇

Beispiel 4.10. Das Ereignis \emptyset ist unabhängig von jedem anderen Ereignis, denn für jedes

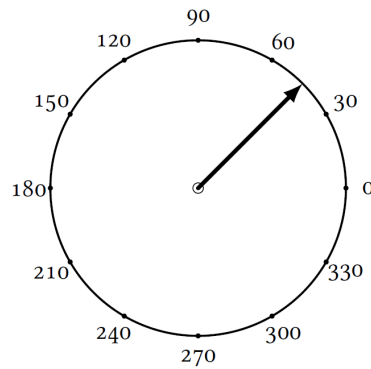


Abbildung 4.1: Ein Glücksrad.

Ereignis B gilt $\emptyset \cap B = \emptyset$. Somit

$$\mathbb{P}(\emptyset \cap B) = \mathbb{P}(\emptyset) = 0 = 0 \cdot \mathbb{P}(B) = \mathbb{P}(\emptyset)\mathbb{P}(B).$$

Ebenso ist das Ereignis Ω unabhängig von jedem anderen Ereignis, denn für jedes Ereignis B gilt $\Omega \cap B = \Omega$. Somit

$$\mathbb{P}(\Omega \cap B) = \mathbb{P}(B) = 1 \cdot \mathbb{P}(B) = \mathbb{P}(\Omega)\mathbb{P}(B).$$

◇

Beispiel 4.11. Die Ereignisse A ('Auf der Karte steht eine gerade Zahl') und B ('Die Karte ist schwarz') aus dem ersten Zufallsexperiment sind unabhängig, denn

$$\mathbb{P}(A \cap B) = \frac{6}{36} = \frac{12}{36} \cdot \frac{18}{36} = \mathbb{P}(A)\mathbb{P}(B).$$

Die Ereignisse $A \cup B$ und $A \cap B$ sind jedoch nicht unabhängig, denn

$$\mathbb{P}((A \cup B) \cap (A \cap B)) = \mathbb{P}(A \cap B) = \frac{1}{6} \neq \frac{2}{3} \cdot \frac{1}{6} = \mathbb{P}(A \cup B)\mathbb{P}(A \cap B).$$

◇

Aufgabe 4.12. Betrachten wir das zweite Zufallsexperiment. Sind die Ereignisse 'Die erste Karte ist ein Ass' und 'Die zweite Karte ist ein Ass' unabhängig? ◇

Aufgabe 4.13. Es seien F und G irgendwelche Ereignisse mit $\mathbb{P}(F) = 0.6$, $\mathbb{P}(G) = 0.2$ und $\mathbb{P}(F \cup G) = 0.72$. Sind F und G unabhängig? ◇

Definition 4.14 (Unabhängigkeit (2)). Wir sagen, dass n Ereignisse A_1, \dots, A_n **paarweise unabhängig** sind, falls jedes Paar von Ereignissen aus diesen n Ereignissen unabhängig ist.

Wir sagen, dass n Ereignissen A_1, \dots, A_n **unabhängig** sind, falls für *jede* Untermenge $I \subset \{1, \dots, n\}$ gilt, dass

$$\mathbb{P}(\cap_{i \in I} A_i) = \prod_{i \in I} \mathbb{P}(A_i).$$

In Worten: Für jede Auswahl von höchstens n Ereignissen ist die Wahrscheinlichkeit, dass all die ausgewählten Ereignisse eintreten, gleich dem Produkt der Wahrscheinlichkeiten, dass sie einzeln eintreten. Unabhängigkeit impliziert paarweise Unabhängigkeit. Dies sieht man, wenn man alle Untermengen I mit zwei Ereignissen überprüft. \diamond

Beispiel 4.15. Wir definieren den Grundraum

$$\Omega := \{(0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 0)\}$$

versehen mit einer diskreten Gleichverteilung, also $\mathbb{P}(\{\omega\}) = 1/4$ für alle $\omega \in \Omega$. Wir betrachten die Ereignisse $A :=$ ‘Die erste Zahl ist eine 1’, $B :=$ ‘Die zweite Zahl ist eine 1’ und $C :=$ ‘Die dritte Zahl ist eine 1’. Sie können nun überprüfen, dass A, B, C paarweise unabhängig sind. Jedoch gilt

$$\mathbb{P}(A \cap B \cap C) = 0 \neq \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \mathbb{P}(A)\mathbb{P}(B)\mathbb{P}(C).$$

Somit sind A, B, C nicht unabhängig. \diamond

Die nächste Definition brauchen wir nur, um später die Gesetze der grossen Zahlen zu verstehen.

Definition 4.16 (Unabhängigkeit (3)). Haben wir eine Familie von unendlich vielen Ereignissen, so sagen wir, dass diese unabhängig ist, wenn jede endliche Teilfamilie unabhängig ist (im Sinne der vorigen Definition). \diamond

4.1.3 Bedingte Wahrscheinlichkeiten

Definition 4.17 (bedingte Wahrscheinlichkeit). Seien A und B Ereignisse mit $\mathbb{P}(B) > 0$. Dann bezeichnen wir

$$\mathbb{P}(A|B) := \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$$

als die **bedingte Wahrscheinlichkeit von A gegeben B** . Die bedingte Wahrscheinlichkeit von A gegeben B drückt die Wahrscheinlichkeit aus, dass A eintritt, wenn man bereits weiss, dass B eintritt. \diamond

Bemerkung 4.18. Sind A und B unabhängig mit $\mathbb{P}(B) > 0$, so gilt

$$\begin{aligned} \mathbb{P}(A|B) &= \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} && [\text{Definition bedingte Wsk.}] \\ &= \frac{\mathbb{P}(A)\mathbb{P}(B)}{\mathbb{P}(B)} && [\text{Definition Unabhängigkeit}] \\ &= \mathbb{P}(A). \end{aligned}$$

Dies entspricht der Intuition, dass bei unabhängigen Ereignissen A und B das Wissen darüber, ob B eintritt oder nicht, einem keine Information darüber gibt, ob auch A eintritt. \diamond

Beispiel 4.19. Im Glücksradbeispiel können wir uns fragen, wie gross die Wahrscheinlichkeit ist, dass der Pfeil zwischen 90 und 270 stehen bleibt (' A '), angenommen, dass er zwischen 0 und 120 stehen bleibt (' B '). Es gilt $\mathbb{P}(B) = (120 - 0)/360 = 1/3$ und $\mathbb{P}(A \cap B) = (120 - 90)/(360) = 1/12$. Also

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} = \frac{1/12}{1/3} = \frac{1}{4}.$$

Da $\mathbb{P}(A) = (270 - 90)/360 = 1/2$, gilt $\mathbb{P}(A|B) \neq \mathbb{P}(A)$. Also sind A, B nicht unabhängig. \diamond

Manchmal nützlich ist der Satz der totalen Wahrscheinlichkeit, der es uns erlaubt, ein komplexes Ereignis zu zerlegen in handlicheren Ereignissen.

Satz 4.20 (totale Wahrscheinlichkeit). Seien E_1, E_2, \dots nicht-überlappende Ereignisse, die zusammen den ganzen Grundraum umfassen, also $E_1 \cup E_2 \cup \dots = \Omega$. Es gelte weiter $\mathbb{P}(E_k) > 0$ für alle $k = 1, 2, \dots$. Dann gilt für jedes Ereignis A :

$$\mathbb{P}(A) = \sum_{k=1}^{\infty} \mathbb{P}(A|E_k)\mathbb{P}(E_k).$$

Dies sieht man so:

$$\begin{aligned} \mathbb{P}(A) &= \mathbb{P}(A \cap \Omega) \\ &= \mathbb{P}\left(A \cap \bigcup_{k=1}^{\infty} E_k\right) \\ &= \mathbb{P}\left(\bigcup_{k=1}^{\infty} (A \cap E_k)\right) \\ &= \sum_{k=1}^{\infty} \mathbb{P}(A \cap E_k) \\ &= \sum_{k=1}^{\infty} \mathbb{P}(A|E_k)\mathbb{P}(E_k). \end{aligned}$$

\diamond

Beispiel 4.21. Betrachten wir wieder das zweite Zufallsexperiment. Die diskrete Gleichverteilung auf Ω_2 weist jedem Elementarereignis (ω_1, ω_2) in Ω_2 eine Wahrscheinlichkeit von $1/1260$ zu. Die Wahrscheinlichkeit, dass die erste Karte $6\clubsuit$ ist und die zweite Karte keine 6 ist, beträgt

$$\frac{1}{36} \cdot \frac{36 - 4}{35} \approx 0.0254.$$

Folglich beträgt die Wahrscheinlichkeit, dass die erste Karte irgendeine 6 ist und die zweite

nicht

$$4 \left(\frac{1}{36} \cdot \frac{36-4}{35} \right) \approx 0.1016.$$

Analog berechnet man die Wahrscheinlichkeit, dass die erste Karte irgendeine 7 ist und die zweite grösser ist:

$$4 \left(\frac{1}{36} \cdot \frac{36-8}{35} \right) \approx 0.0889.$$

Die Ereignisse 'Die erste Karte ist eine 6', 'Die erste Karte ist eine 7', ..., überlappen nicht und decken den ganzen Grundraum ab. Also können wir den Satz der totalen Wahrscheinlichkeit anwenden, um die Wahrscheinlichkeit zu berechnen, dass die erste Karte niedriger als die zweite Karte ist:

$$\sum_{k=1}^9 4 \left(\frac{1}{36} \cdot \frac{36-4k}{35} \right) = \frac{4}{36 \cdot 35} \sum_{k=1}^9 (36-4k) = \frac{4}{36 \cdot 35} (9 \cdot 36 - (4+8+\dots+36)).$$

Das Resultat berechnen wir in R:

```
4/(36*35) * (9*36 - sum(4*seq(1, 9)))
```

```
[1] 0.4571429
```

Das heisst, etwa 46%.

Etwas eleganter kann man dies auch so lösen: Die Wahrscheinlichkeit, dass beide Karte den gleichen Rang haben, beträgt $3/35$. Also beträgt die Wahrscheinlichkeit, dass die beiden Karten unterschiedlichen Ranges sind $32/35$. Es ist dann klar, dass die Wahrscheinlichkeiten, dass die erste niedriger als die zweite ist und dass die zweite niedriger als die erste ist, gleich sind. Daher beträgt die Wahrscheinlichkeit, dass die erste Karte niedrigeren Ranges als die zweite ist $(32/35)/2 \approx 0.4571$. \diamond

Aufgabe 4.22 (M&Ms). M&Ms kommen in sechs Farben vor; in Tabelle 4.1 auf der nächsten Seite werden ihre relativen Frequenzen aufgelistet. Wir gehen davon aus, dass die Population der M&Ms unendlich gross ist, sodass wir unterschiedliche Auswahlen als unabhängig auffassen dürfen.

1. Wie wahrscheinlich ist es, dass ein zufällig ausgewähltes M&M rot *oder* orange ist?
2. Wie wahrscheinlich ist es, dass von zwei zufällig ausgewählten M&Ms *beide* rot oder orange (also zwei rote, zwei orange oder ein rotes und ein oranges) sind?
3. Wie wahrscheinlich ist es, dass von zwei zufällig ausgewählten M&Ms eines rot und eines orange ist?
4. Wie wahrscheinlich ist es, dass wenn 5 M&Ms zufällig ausgewählt werden, alle blau sind?
5. Wie wahrscheinlich ist es, dass wenn 5 M&Ms zufällig ausgewählt werden, kein einziges blau ist? \diamond

Tabelle 4.1: Relative Frequenzen von M&Ms nach Farbe.

Farbe	relative Frequenz
blau	23%
orange	23%
gelb	15%
grün	15%
braun	12%
rot	12%

Sowohl in wissenschaftlichen als auch in gesellschaftlichen Diskussionen kommt es leider recht häufig vor, dass die bedingten Wahrscheinlichkeiten $\mathbb{P}(A|B)$ und $\mathbb{P}(B|A)$ miteinander verwechselt werden. Das nächste klassische Beispiel und der darauf folgende Satz zeigen, wie man $\mathbb{P}(A|B)$ und $\mathbb{P}(B|A)$ richtig miteinander in Bezug zu setzt.

Beispiel 4.23 (medizinische Tests). Man stelle sich vor, dass bei allen Neugeborenen einen medizinischen Test durchgeführt wird, um eine seltene genetische Krankheit aufzuspüren, von der schätzungsweise 0.01% der Neugeborenen betroffen ist. Der Test stuft tatsächlich Betroffene zu einer Wahrscheinlichkeit von 97% als krank ein. Jedoch werden auch 1% der Neugeborenen, die nicht von der Krankheit betroffen sind, trotzdem als krank eingestuft. Wie wahrscheinlich ist es nun, dass ein Neugeborener, der als krank eingestuft wird, tatsächlich die Krankheit hat?

Um diese Frage zu beantworten, können wir uns zunächst eine grosse Anzahl von Neugeborenen vorstellen, zum Beispiel eine Million. Tatsächlich krank wären dann etwa $0.0001 \cdot 10^6 = 100$ unter ihnen; die restlichen 999'900 wären nicht betroffen. Von den 100 kranken Kindern werden 97 als krank eingestuft; bei drei wird die Krankheit nicht sofort entdeckt. Von den 999'900 nicht-betroffenen Kindern werden $0.01 \cdot 999900 = 9999$ trotzdem als krank eingestuft. Wenn wir nun aus den insgesamt $97 + 9999 = 10096$ als krank eingestuften Kindern zufällig ein Kind auswählen, so beträgt die Wahrscheinlichkeit, dass dieses tatsächlich von der Krankheit betroffen ist, bloss $97/10096 \approx 0.0096$, also nicht einmal 1%. \diamond

Das Vorgehen aus dem vorigen Beispiel ist allgemeingültig, wie der berühmte Satz von Bayes zeigt.

Satz 4.24 (Bayes). Seien A, B Ereignisse mit $\mathbb{P}(A), \mathbb{P}(B) > 0$. Dann gilt

$$\begin{aligned}
 \mathbb{P}(B|A) &= \frac{\mathbb{P}(B \cap A)}{\mathbb{P}(A)} && \text{[Definition bedingte Wsk.]} \\
 &= \frac{\mathbb{P}(A|B)\mathbb{P}(B)}{\mathbb{P}(A)} && [\mathbb{P}(A|B) = \mathbb{P}(A \cap B)/\mathbb{P}(B)] \\
 &= \frac{\mathbb{P}(A|B)\mathbb{P}(B)}{\mathbb{P}(A|B)\mathbb{P}(B) + \mathbb{P}(A|B^c)(1 - \mathbb{P}(B))}. && \text{[totale Wahrscheinlichkeit]}
 \end{aligned}$$

\diamond

Angewandt auf Beispiel 4.23 hiesse B 'Kind ist betroffen' und A 'Kind wird als krank eingestuft'. Dann $\mathbb{P}(A|B) = 0.97$, $\mathbb{P}(B) = 0.0001$, $\mathbb{P}(A|B^c) = 0.01$. Also

$$\mathbb{P}(B|A) = \frac{0.97 \cdot 0.0001}{0.97 \cdot 0.0001 + 0.01 \cdot (1 - 0.0001)} \approx 0.0096.$$

4.2 Zufallsvariablen

Beim Analysieren quantitativer Daten nehmen wir häufig an, dass diese Daten Beobachtungen von Zufallsvariablen sind.

Definition 4.25 (Zufallsvariablen). Sei Ω der Grundraum eines Wahrscheinlichkeitsraums. Eine **Zufallsvariable** X assoziiert jedes Ereignis $\omega \in \Omega$ mit einer reellen Zahl.

Ist die Menge der möglichen Werte von X auflistbar, so nennen wir X eine **diskrete Zufallsvariable**. \diamond

Beispiel 4.26. Beim Jassen können wir die Zufallsvariable X betrachten, die jede Karte mit ihrem Wert im *Obenabe*-Spiel assoziiert, d.h.,

$$X(\omega) := \begin{cases} 11, & \text{falls } \omega \text{ ein Ass ist,} \\ 4, & \text{falls } \omega \text{ ein König ist,} \\ 3, & \text{falls } \omega \text{ eine Dame ist,} \\ 2, & \text{falls } \omega \text{ ein Bub ist,} \\ 10, & \text{falls } \omega \text{ eine 10 ist,} \\ 8, & \text{falls } \omega \text{ eine 8 ist,} \\ 0, & \text{sonst.} \end{cases}$$

Diese Zufallsvariable ist bloss eine von beliebig vielen, die man auf dem Grundraum Ω_1 konstruieren kann.

Wenn wir zwei Karten ziehen, könnten wir beispielsweise eine Zufallsvariable definieren, die den Gesamtwert der beiden Karten darstellt. \diamond

Wir können die Werte, die eine Zufallsvariable annimmt, auch als eigene Ergebnisse betrachten. So erhalten wir einen neuen Grundraum

$$\tilde{\Omega} := \{X(\omega) : \omega \in \Omega_1\} = \{0, 2, 3, 4, 8, 10, 11\}.$$

Die Wahrscheinlichkeit, dass X den Wert 11 annimmt, ist hier $4/36 = 1/9$; wir schreiben $\mathbb{P}(X = 11) = 1/9$. Auch gilt $\mathbb{P}(X = 0) = 1/3$. Auch wenn wir Zufallsvariablen betrachten, können wir über Ereignisse reden, beispielsweise das Ereignis, dass $X \geq 10$ ist ($\mathbb{P}(X \geq 10) = 2/9$).

Wir widmen uns nun den wichtigsten Darstellungen und Eigenschaften von Zufallsvariablen.

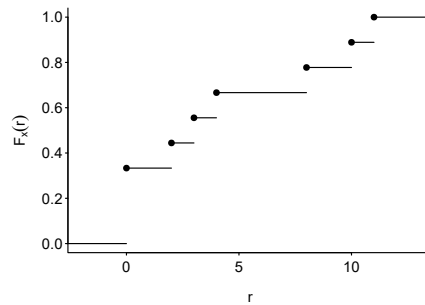


Abbildung 4.2: Verteilungsfunktion der Punktzahl einer Jasskarte beim Obenabe.

4.2.1 Die Verteilungs- und Quantilfunktion

Zufallsvariablen werden vollständig von ihrer **Verteilungsfunktion** charakterisiert (vgl. Abschnitt 3.6). Die Verteilungsfunktion F_X einer Zufallsvariablen X ist wie folgt definiert:

$$F_X(r) := \mathbb{P}(X \leq r)$$

für jede reellen Zahl r .

Greifen wir das obige Beispiel wieder auf, so können wir eine Tabelle mit den kumulativen Wahrscheinlichkeiten zusammenbasteln:

Wert	0	2	3	4	8	10	11
Wahrscheinlichkeit	0.333	0.111	0.111	0.111	0.111	0.111	0.111
kumulative Wahrscheinlichkeit	0.333	0.444	0.555	0.666	0.777	0.888	1

Folglich gelten etwa $F_X(-1) = 0$, $F_X(2.5) = 0.444$ und $F_X(1.2) = 1$. Abbildung 4.2 stellt die Verteilungsfunktion von X dar.

Kennen wir die Verteilungsfunktion einer Zufallsvariablen, so können wir Wahrscheinlichkeitsaussagen darüber machen, ob die Zufallsvariable in einem bestimmten Intervall liegt:

$$\begin{aligned}\mathbb{P}(X \in (-\infty, b]) &= \mathbb{P}(X \leq b) = F_X(b), \\ \mathbb{P}(X \notin (-\infty, a]) &= \mathbb{P}(X > a) = 1 - F_X(a)\end{aligned}$$

und

$$\mathbb{P}(X \in (a, b]) = F_X(b) - F_X(a).$$

Etwas mehr Vorsicht ist geboten bei Intervallen der Form $[a, b]$, $[a, b)$ und (a, b) , da die Endpunkte möglicherweise eine positive Wahrscheinlichkeit haben. Dazu definieren wir noch die Funktion

$$F_X(r-) := \lim_{s \rightarrow r} F_X(s) = \mathbb{P}(X \in (-\infty, r)) = F_X(r) - \mathbb{P}(X = r).$$

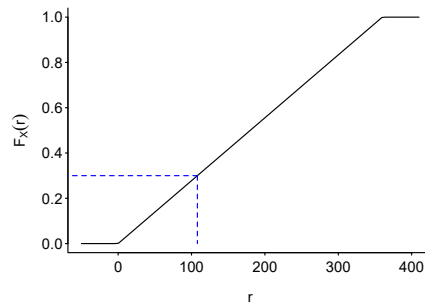


Abbildung 4.3: Verteilungsfunktion der kontinuierlichen Gleichverteilung auf $[0, 360)$.

Dann gilt

$$\begin{aligned}\mathbb{P}(X \in [a, b]) &= F_X(b) - F_X(a-) = F_X(b) - F_X(a) + \mathbb{P}(X = a), \\ \mathbb{P}(X \in [a, b)) &= F_X(b-) - F_X(a-) = F_X(b) - F_X(a) - \mathbb{P}(X = b) + \mathbb{P}(X = a), \\ \mathbb{P}(X \in (a, b)) &= F_X(b-) - F_X(a) = F_X(b) - F_X(a) - \mathbb{P}(X = b).\end{aligned}$$

Während die Verteilungsfunktion uns sagt, wie wahrscheinlich es ist, dass eine Zufallsvariable nicht grösser als ein bestimmter Wert ist, liefert uns die **Quantilfunktion** F_X^{-1} die umgekehrte Information: Gegeben eine Zahl $p \in (0, 1)$ sagt sie uns, was der niedrigste Wert q ist, sodass $F_X(q) \geq p$. In Formeln:

$$F_X^{-1}(p) := \min\{q \in \mathbb{R} : F_X(q) \geq p\}.$$

Abbildung 4.2 kann man beispielsweise entnehmen, dass $F_X^{-1}(0.4) = 3$ und $F_X^{-1}(7/9) = 8$.

Beispiel 4.27 (kontinuierliche Gleichverteilung). Beim Glücksrad aus Beispiel 4.8 können wir die Zufallsvariable X betrachten, die ganz einfach darstellt, bei welchem Wert der Pfeil stehen bleibt. Die Verteilungsfunktion von X (Abbildung 4.3) ist diesmal **stetig** (ohne Lücken). Das 0.3-Quantil dieser Verteilung liegt bei 108. \diamond

4.2.2 Wahrscheinlichkeitsdichten

Die Verteilung einer Zufallsvariablen X mit einer stetigen Verteilungsfunktion F_X kann auch mittels einer **Wahrscheinlichkeitsdichte** f_X dargestellt werden. Abbildung 4.4 zeigt, was die Idee ist. Wir wollen die Verteilung von X so darstellen, dass die Wahrscheinlichkeit, dass X in einem bestimmten Intervall liegt, gerade die Fläche unter der Wahrscheinlichkeitsdichte in diesem Intervall ist. Die Kerndichteschätzungen aus Kapitel 3 sind eine Methode, um anhand von Beobachtungen einer Zufallsvariablen die Wahrscheinlichkeitsdichte ihrer Verteilung zu schätzen.

Für die kontinuierliche Gleichverteilung auf $[0, 360)$ ist eine² Wahrscheinlichkeitsdichte f_X

²Wahrscheinlichkeitsdichten sind nicht eindeutig definiert, aber zwei unterschiedliche Wahrscheinlichkeitsdichten

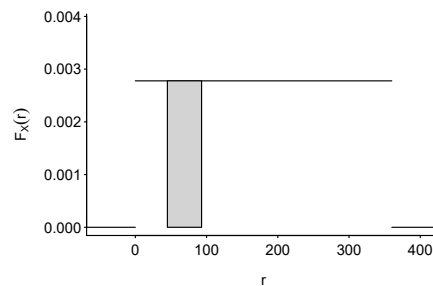


Abbildung 4.4: Wahrscheinlichkeitsdichte einer kontinuierlichen Gleichverteilung mit Bereich 0 bis 360. Die Wahrscheinlichkeit, dass wir einen Wert zwischen 45 und 93 beobachten, entspricht der Fläche unter der Wahrscheinlichkeitsdichte in diesem Intervall.

gegeben durch

$$f_X(x) = \begin{cases} \frac{1}{360}, & \text{falls } x \in [0, 360), \\ 0, & \text{sonst.} \end{cases}$$

Wie Sie sich aus der Schule erinnern dürften, entspricht diese Fläche dem Integral der Funktion über dieses Intervall. Tatsächlich gilt

$$\mathbb{P}(X \in [45, 93]) = \int_{45}^{93} f_X(x) \, dx = \int_{45}^{93} \frac{1}{360} \, dx = \frac{93 - 45}{360}.$$

Zufallsvariablen, deren Verteilung eine Wahrscheinlichkeitsdichte besitzt, nennen wir **kontinuierlich**. Wir können auch sagen, dass ihre Verteilung (absolut-)stetig ist.

Bemerkung 4.28. Eine kontinuierliche Zufallsvariable kann nicht diskret sein und umgekehrt: Falls X eine Wahrscheinlichkeitsdichte hat, so gilt

$$\mathbb{P}(X = r) = \int_r^r f_X(x) \, dx = 0$$

für jedes r . Ist X diskret, so muss es jedoch ein r geben mit $\mathbb{P}(X = r) > 0$.

Es existieren jedoch Zufallsvariablen, die weder kontinuierlich noch diskret sind. Ein Beispiel einer solchen Zufallsvariablen ist die Niederschlagsmenge an einem bestimmten Tag in einem bestimmten Gebiet. So kann es beispielsweise eine Wahrscheinlichkeit von 70% geben, dass es nicht regnet ($\mathbb{P}(X = 0) = 0.7$), während die Niederschlagsmenge, wenn es tatsächlich regnet, schon kontinuierlich verteilt ist. \diamond

zur gleichen Verteilung stimmen ‘fast überall’ überein. ‘Fast überall’ hat hierbei eine genaue mathematische Bedeutung, die uns jedoch zu weit führen würde.

4.2.3 Der Erwartungswert und die Varianz

Bevor wir uns ein paar klassische Wahrscheinlichkeitsverteilungen genauer anschauen, wollen wir die zwei am häufigst verwendeten numerischen Merkmale von Wahrscheinlichkeitsverteilungen genauer anschauen. Das erste Merkmal ist der **Erwartungswert**. Dieser drückt aus, welcher Wert Zufallsvariablen, die einer bestimmten Verteilung folgen, im Schnitt annimmt. Der Erwartungswert ist die Verallgemeinerung des arithmetischen Mittels auf beliebig grosse Grundräume bzw. Populationen.

Für eine diskrete Zufallsvariable X auf einem Grundraum Ω kann der Erwartungswert $\mathbb{E}(X)$ wie folgt berechnet werden,

$$\mathbb{E}(X) = \sum_{x \in X(\Omega)} x \mathbb{P}(X = x),$$

insofern diese Summe tatsächlich existiert. (Nicht alle Verteilungen haben einen Erwartungswert. Mit solchen pathologischen Verteilungen werden wir uns aber nicht auseinandersetzen.) So gilt im Obenabe-Beispiel (Beispiel 4.26)

$$\mathbb{E}(X) = 0 \cdot \frac{1}{3} + 2 \cdot \frac{1}{9} + 3 \cdot \frac{1}{9} + 4 \cdot \frac{1}{9} + 8 \cdot \frac{1}{9} + 10 \cdot \frac{1}{9} + 11 \cdot \frac{1}{9} = 4.22.$$

Ein weiteres Beispiel: Sei G die Zufallsvariable, die vom Zufallsgenerator aus Beispiel 4.6 generiert wird. Ihr Erwartungswert beträgt

$$\mathbb{E}(G) = \sum_{k=1}^{\infty} k \mathbb{P}(G = k) = \sum_{k=1}^{\infty} \frac{k}{2^k} = 2.$$

Erwartungswerte wie in diesem unendlichen Fall brauchen Sie für diesen Kurs nicht selber berechnen zu können. Der Zweck des Beispiels ist lediglich, zu zeigen, dass Erwartungswerte auch berechnet werden können, wenn die Anzahl möglicher Werte unendlich gross ist.

Allgemeiner gilt übrigens, dass

$$\mathbb{E}(g(X)) = \sum_{x \in X(\Omega)} g(x) \mathbb{P}(X = x)$$

für (grundsätzlich) jede Abbildung g , insofern diese Summe existiert.³

Hat die Verteilung der Zufallsvariablen X eine Wahrscheinlichkeitsdichte f_X , so berechnet man ihren Erwartungswert als

$$\mathbb{E}(X) = \int_{-\infty}^{\infty} x f_X(x) dx,$$

³Die Ausnahmen sind für uns in der Praxis nicht relevant.

insofern dieses Integral tatsächlich existiert. So gilt im Glücksradbeispiel

$$\begin{aligned}\mathbb{E}(R) &= \int_{-\infty}^{\infty} x f_X(x) \, dx \\ &= \int_0^{360} \frac{x}{360} \, dx \\ &= \frac{1}{360} \left[\frac{1}{2} x^2 \right]_0^{360} \\ &= \frac{360^2}{2 \cdot 360} \\ &= 180.\end{aligned}$$

Auch hier gilt allgemeiner, dass

$$\mathbb{E}(g(X)) = \int_{-\infty}^{\infty} g(x) f_X(x) \, dx,$$

für (grundsätzlich) jede Abbildung g , insofern dieses Integral existiert.⁴ (Nicht alle Verteilungen haben eine Varianz. Mit solchen pathologischen Verteilungen werden wir uns aber nicht auseinandersetzen.)

Für Wahrscheinlichkeitsverteilungen, die weder kontinuierlich noch diskret sind, kann man allenfalls den Erwartungswert separat für ihren kontinuierlichen und ihren diskreten Teil berechnen, und dann mit einer geeigneten Gewichtung kombinieren. Wir werden uns jedoch nur mit diskreten und kontinuierlichen Verteilungen auseinandersetzen.

Lemma 4.29 (Eigenschaften des Erwartungswerts). Der Erwartungswert einer Konstanten c ist gleich c , also $\mathbb{E}(c) = c$.

Der Erwartungswert ist wie bereits erwähnt **linear**. Dies heisst, dass $\mathbb{E}(aX + bY) = a\mathbb{E}(X) + b\mathbb{E}(Y)$ für Konstanten a, b , insofern $\mathbb{E}(X)$ und $\mathbb{E}(Y)$ existieren. Diese Eigenschaft folgt aus der Linearität von endlichen und unendlichen Summen und von Integralen. \diamond

Der Erwartungswert drückt aus, welcher Wert eine Zufallsvariable im Schnitt annimmt. Es wäre jedoch nützlich, auch eine numerisches Mass zu haben, das ausdrückt, wie stark die Werte, die die Zufallsvariable annimmt, im Schnitt von diesem Erwartungswert abweichen. Eine erste Idee dürfte sein, dass wir dazu den Erwartungswert der Abweichungen berechnen, also

$$\mathbb{E}(X - \mathbb{E}(X)).$$

Der Erwartungswert ist linear, also gilt

$$\mathbb{E}(X - \mathbb{E}(X)) = \mathbb{E}(X) - \mathbb{E}(\mathbb{E}(X)) = \mathbb{E}(X) - \mathbb{E}(X) = 0,$$

⁴Diese Formel bezeichnet man übrigens manchmal als *law of the unconscious statistician*.

und dies für jede Zufallsvariable mit einem Erwartungswert. Also liefert uns diese Grösse keine Information. Sinnvoller wäre, den Erwartungswert der absoluten Unterschiede zu berechnen, also

$$\mathbb{E}(|X - \mathbb{E}(X)|).$$

Diese Grösse wird durchaus verwendet, ist aber eher schwierig zu hantieren. Stattdessen arbeitet man meistens mit dem Erwartungswert der quadrierten Unterschiede, also mit

$$\mathbb{E}((X - \mathbb{E}(X))^2).$$

Diese Grösse nennt man die **Varianz** der Verteilung und stellt eine Verallgemeinerung der Varianz aus Kapitel 3 auf beliebige Grundräume bzw. Populationen dar. Es gilt

$$\begin{aligned} \text{Var}(X) &= \mathbb{E}((X - \mathbb{E}(X))^2) && [\text{Definition}] \\ &= \mathbb{E}(X^2 - 2X\mathbb{E}(X) + \mathbb{E}(X)^2) && [(a - b)^2 = a^2 - 2ab + b^2] \\ &= \mathbb{E}(X^2) - 2\mathbb{E}(X(\mathbb{E}(X))) + \mathbb{E}(X)^2 && [\text{Linearität von } \mathbb{E}] \\ &= \mathbb{E}(X^2) - 2\mathbb{E}(X)^2 + \mathbb{E}(X)^2 && [\mathbb{E}(X) \text{ ist konstant}] \\ &= \mathbb{E}(X^2) - \mathbb{E}(X)^2. \end{aligned}$$

Die Varianz einer Zufallsvariablen, die in einer bestimmten Einheit ausgedrückt wird (z.B. Sekunden), wird selber in quadrierten Einheiten ausgedrückt (z.B. quadrierten Sekunden). Die Grösse $\sqrt{\text{Var}(X)} =: \text{Std}(X)$ nennt man die **Standardabweichung** von X und wird in den gleichen Einheiten wie die Variable selbst ausgedrückt.

Beispiel 4.30. Im Obenabe-Beispiel haben wir

$$\mathbb{E}(X^2) = 0^2 \cdot \frac{1}{3} + 2^2 \cdot \frac{1}{9} + 3^2 \cdot \frac{1}{9} + 4^2 \cdot \frac{1}{9} + 8^2 \cdot \frac{1}{9} + 10^2 \cdot \frac{1}{9} + 11^2 \cdot \frac{1}{9} = 34.89.$$

Daher beträgt die Varianz im Obenabe-Beispiel

$$\mathbb{E}(X^2) - \mathbb{E}(X)^2 = 34.89 - 4.22^2 = 17.08.$$

Die Standardabweichung beträgt also etwa 4.13.

Für den Zufallsgenerator aus Beispiel 4.6 gilt

$$\mathbb{E}(G^2) = \sum_{k=1}^n \frac{k^2}{2^k} = 6.$$

Also beträgt die Varianz von G

$$\text{Var}(G) = \mathbb{E}(G^2) - \mathbb{E}(G)^2 = 6 - 2^2 = 2.$$

Im Glücksradbeispiel gilt

$$\mathbb{E}(R^2) = \int_0^{360} \frac{x^2}{360} dx = \frac{360^3}{3 \cdot 360} = 43200.$$

Daher beträgt die Varianz

$$\mathbb{E}(R^2) - \mathbb{E}(R)^2 = 43200 - 180^2 = 10800.$$

Die Standardabweichung beträgt etwa 103.9. \diamond

Lemma 4.31 (Eigenschaften der Varianz). Für die Varianz gilt

$$\text{Var}(aX + b) = a^2 \text{Var}(X),$$

für Konstanten a, b , falls $\text{Var}(X)$ besteht.

Im Allgemeinen gilt jedoch *nicht*, dass $\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y)$. Dies gilt aber schon, wenn X, Y unabhängig im Sinne des nächsten Abschnitts sind. \diamond

Beispiel 4.32. Auf der Basis des Zufallsgenerators G aus Beispiel 4.6 definieren wir die Zufallsvariable $\tilde{G} := 3G + 4$. Dann

$$\mathbb{E}(\tilde{G}) = \mathbb{E}(3G + 4) = 3\mathbb{E}(G) + 4 = 10$$

und

$$\text{Var}(\tilde{G}) = \text{Var}(3G + 4) = 3^2 \text{Var}(G) = 18. \quad \diamond$$

Beispiel 4.33 (Varianz einer Summe \neq Summe der Varianzen). Auf der Basis des Zufallsgenerators G aus Beispiel 4.6 definieren wir die Zufallsvariable $\bar{G} := -G$. Dann gilt $G + \bar{G} = G - G = 0$. Also

$$\text{Var}(G + \bar{G}) = 0 \neq 4 = \text{Var}(G) + \text{Var}(\bar{G}).$$

Die Zufallsvariablen G und \bar{G} sind klar nicht unabhängig. \diamond

4.2.4 Unabhängigkeit

Definition 4.34 (Unabhängigkeit von Zufallsvariablen). Seien X, Y zwei Zufallsvariablen. Wir sagen, dass X und Y unabhängig sind, falls für jedes Paar von Zahlenmengen E_1, E_2 gilt,⁵ dass

$$\mathbb{P}(X \in E_1, Y \in E_2) = \mathbb{P}(X \in E_1) \mathbb{P}(Y \in E_2).$$

⁵Im Prinzip muss es überhaupt möglich sein, über die Ereignisse $X \in E_1, Y \in E_2$ Wahrscheinlichkeitsaussagen machen zu können – vgl. die Bemerkung auf Seite 79. Für unsere Zwecke ist diese Randbemerkung jedoch bloss eine Spitzfindigkeit, da es recht schwierig ist, eine Zahlenmenge E zu konstruieren, sodass wir über das Ereignis $X \in E$ keine Wahrscheinlichkeitsaussagen machen können.

Die Definition von Unabhängigkeit von mehr als zwei Ereignissen lässt sich sinngemäss auf die Unabhängigkeit von mehr als zwei Zufallsvariablen übertragen. \diamond

Konzeptuell bedeutet die Unabhängigkeit von X und Y , dass einem der Wert von X einem keinerlei Information darüber gibt, was der Wert von Y ist und umgekehrt.

Lemma 4.35. Sind X, Y unabhängige Zufallsvariablen mit existierender Varianz. Dann gilt

$$\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y).$$

Auf den Beweis verzichten wir hier.

Die entsprechende Gleichung für den Erwartungswert gilt immer ($\mathbb{E}(X + Y) = \mathbb{E}(X) + \mathbb{E}(Y)$), bei der Varianz im Allgemeinen jedoch nur für unabhängige Variablen. \diamond

4.3 Beispiele von diskreten Wahrscheinlichkeitsverteilungen

Wir wollen uns nun einige klassische Wahrscheinlichkeitsverteilungen näher anschauen.

4.3.1 Die diskrete Gleichverteilung

Sei n eine natürliche Zahl und $\Omega := \{1, \dots, n\}$. Gilt für jedes $k \in \Omega$, dass $\mathbb{P}(X = k) = 1/n$, so hat X eine diskrete Gleichverteilung über Ω . Wir schreiben $X \sim \text{Unif}(\Omega)$.

Das klassische Beispiel ist ein Würfelwurf mit einem sechsseitigen Würfel. Es gilt $\mathbb{P}(X = k) = 1/6$ für $k \in \{1, 2, 3, 4, 5, 6\}$ und $\mathbb{P}(X = k) = 0$ für alle andere k .

Der Erwartungswert einer Zufallsvariablen X mit einer diskreten Gleichverteilung mit Grundraum $\{1, 2, \dots, n\}$ beträgt

$$\mathbb{E}(X) = \frac{1}{n}(1 + 2 + \dots + n) = \frac{n(1+n)}{2n} = \frac{1+n}{2},$$

was wohl intuitiv einleuchtet. Die Varianz dieser Zufallsvariablen beträgt

$$\text{Var}(X) = \frac{n^2 - 1}{12},$$

wobei wir uns die Herleitung dieser Tatsache schenken.

Allgemeiner gilt für eine Zufallsvariable X mit einer diskreten Gleichverteilung auf

$$\Omega := \{a, a+1, a+2, \dots, b-2, b-1, b\},$$

dass

$$\mathbb{E}(X) = \frac{a+b}{2}, \text{Var}(X) = \frac{b^2 - a^2}{12}.$$

Aufgabe 4.36. Sei X eine Zufallsvariable mit einer diskreten Gleichverteilung auf

$$\{1, 2, \dots, n-1, n\}.$$

Wir definieren $Y := X/2 + 1/2$. Dann hat Y eine diskrete Gleichverteilung auf

$$\{1, 1.5, 2, \dots, n/2 - 0.5\}.$$

Verwenden Sie die Eigenschaften des Erwartungswerts und der Varianz, um $\mathbb{E}(Y)$, $\text{Var}(Y)$ zu bestimmen.

Betrachten wir nun eine Zufallsvariable $Z \sim \text{Unif}(\Omega)$, wo

$$\Omega = \{1, 1.5, 2, \dots, n/2 - 0.5, n/2\}.$$

Bestimmen Sie $\mathbb{E}(Z)$, $\text{Var}(Z)$.

◇

Bemerkung 4.37 (Daten aus einer diskreten Gleichverteilung generieren). In späteren Kapiteln werden wir häufig Simulationen verwenden, um Konzepte zu verstehen und Daten auszuwerten. Für solche Situationen müssen wir Daten aus vorgegebenen Verteilungen generieren. Mit dem folgenden Vorgehen generieren wir `n_obs` unabhängige Datenpunkte aus einer diskreten Gleichverteilung auf $\{1, \dots, n\}$.

```
# Grundraum definieren
n <- 6
Omega <- 1:n
# n_obs Datenpunkte aus Gleichverteilung auf {1, ..., n} generieren
n_obs <- 20
daten <- sample(Omega, n_obs, replace = TRUE)
daten

[1] 1 2 1 2 6 4 2 6 6 1 4 3 1 2 5 5 3 3 1 6
```

◇

4.3.2 Die Bernoulliverteilung

Die **Bernoulliverteilung** beschreibt den Ausgang X eines Zufallsexperiments mit zwei möglichen Ergebnissen, die wir als 0 und 1 bezeichnen. Sie hat einen Parameter $p \in [0, 1]$, die die Wahrscheinlichkeit einer 1 ausdrückt. Also gelten $\mathbb{P}(X = 1) = p$ und $\mathbb{P}(X = 0) = 1 - p$. Folgt eine Zufallsvariable einer Bernoulliverteilung mit Parameter p , so schreiben wir $X \sim \text{Bernoulli}(p)$.

Der Erwartungswert einer $\text{Bernoulli}(p)$ -verteilten Zufallsvariablen X lässt sich einfach berechnen:

$$\mathbb{E}(X) = 0 \cdot (1 - p) + 1 \cdot p = p.$$

Ebenso gilt

$$\mathbb{E}(X^2) = 0^2 \cdot (1 - p) + 1^2 \cdot p = p.$$

Daher

$$\text{Var}(X) = p - p^2 = p(1 - p).$$

Beispiel 4.38. Ziehen wir eine Karte zufällig aus einem Blatt Jasskarten, so beträgt die Wahrscheinlichkeit, dass es sich um ein Ass handelt $p = 4/36 = 1/9$. Das Zufallsexperiment ‘Ass ziehen’ können wir als Bernoulliexperiment mit Parameter p modellieren, wobei wir das Ziehen eines Asses als 1 bezeichnen und das Nicht-Ziehen eines Asses als 0. \diamond

4.3.3 Die Binomialverteilung

Seien X_1, \dots, X_n unabhängige Zufallsvariablen, die alle Bernoulli(p)-verteilt sind. Dann folgt $X := X_1 + \dots + X_n$ einer **Binomialverteilung** mit Parametern n und p . Wir schreiben $X \sim \text{Binomial}(n, p)$. Die Binomialverteilung erfasst also, wie viele von n unabhängigen und identischen Bernoulliexperimenten erfolgreich ausgehen.

Wegen der Linearität des Erwartungswerts gilt

$$\mathbb{E}(X) = \mathbb{E}(X_1) + \dots + \mathbb{E}(X_n) = np.$$

Wegen der Unabhängigkeit von X_1, \dots, X_n gilt weiter

$$\text{Var}(X) = np(1 - p).$$

Die Wahrscheinlichkeit, dass genau die ersten k von n unabhängigen und identischen Bernoulliexperimenten Erfolg sind, beträgt

$$\mathbb{P}(X_1 = 1)\mathbb{P}(X_2 = 1) \dots \mathbb{P}(X_k = 1)\mathbb{P}(X_{k+1} = 0) \dots \mathbb{P}(X_n = 0) = p^k(1 - p)^{n-k}.$$

Es gibt $n!$ Möglichkeiten, die Bernoulliexperimente umzuordnen; siehe Beispiel 1.11. Da die Ausgänge von k bzw. $n - k$ Bernoulliexperimente nicht voneinander unterschieden werden können, gibt es

$$\frac{n!}{k!(n - k)!} =: \binom{n}{k}$$

unterschiedliche mögliche Reihenfolgen, in denen die Ausgänge beobachtet werden können. Insgesamt beträgt die Wahrscheinlichkeit, dass genau k von n unabhängigen und identischen Bernoulliexperimenten Erfolge sind

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

Die Zahl $\binom{n}{k}$ nennen wir den Binomialkoeffizienten von n und k oder auch ‘ n tief k ’ oder ‘ n choose k ’. Für $k < 0$ oder $k > n$ definieren wir $\binom{n}{k} = 0$. Die Verteilungsfunktion einer Binomial(n ,

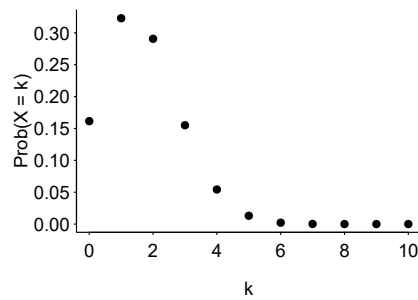


Abbildung 4.5: Wahrscheinlichkeit der Anzahl Erfolg bei einer Binomial(10, 1/6)-Verteilung.

p)-Verteilung ist folglich gegeben durch

$$F(r) = \sum_{k=0}^{\lfloor r \rfloor} \binom{n}{k} p^k (1-p)^{n-k}.$$

Der Binomialkoeffizient $\binom{n}{k}$ kann mit dem R-Befehl `choose(n, k)` berechnet werden. Auch eingebaut in R sind einige Funktionen, mit denen man Informationen über die Binomialverteilung abfragen kann und binomialverteilte Zufallsdaten generieren kann. Als Beispiel können wir uns hier vorstellen, dass wir 10 Mal mit einem 6-seitigen Würfel werfen und dabei zählen, wie oft wir eine 6 würfeln. Dies entspricht einer Binomial(10, 1/6)-Verteilung. Mit der Funktion `dbinom()` können wir $\mathbb{P}(X = k)$ abfragen. Beispielsweise beträgt die Wahrscheinlichkeit, dass wir genau fünf Mal eine 6 würfeln ungefähr 1.3%.

```
dbinom(5, 10, 1/6)
[1] 0.01302381
```

Abbildung 4.5 zeigt diese Wahrscheinlichkeit für $k = 0, \dots, 10$.

```
k <- 0:10
plot(k, dbinom(k, 10, 1/6), pch = 16,
     xlab = "k", ylab = "Prob(X = k)")
```

Mit `pbinom()` können wir $\mathbb{P}(X \leq k)$ abfragen. So finden wir heraus, dass die Wahrscheinlichkeit, dass wir höchstens zwei Mal eine 6 würfeln, etwa 78% beträgt.

```
pbinom(2, 10, 1/6)
[1] 0.7752268
```

Die Wahrscheinlichkeit, dass wir mehr als zwei Sechsen würfeln, beträgt also etwa 22%. Die Wahrscheinlichkeit, dass wir mindestens zwei Sechsen würfeln beträgt etwa 52%.

```
1 - pbinom(1, 10, 1/6)
[1] 0.5154833
pbinom(1, 10, 1/6, lower.tail = FALSE)
[1] 0.5154833
```

Mit `qbinom()` können wir die Quantile der Binomial(n, p)-Verteilung abfragen:

```
qbinom(0.70, 10, 1/6)
[1] 2
qbinom(0.75, 10, 1/6)
[1] 2
qbinom(0.80, 10, 1/6)
[1] 3
```

Mit anderen Worten: Wenn zig Leute je 10 Mal würfeln, werden mindestens 70% unter ihnen höchstens zwei Sechsen würfeln. Sogar werden mindestens 75% unter ihnen höchstens zwei Sechsen würfeln. Mindestens 80% unter ihnen werden höchstens drei Sechsen würfeln. Tatsächlich werden ungefähr 77.5% höchstens zwei Sechsen würfeln, wie wir mit `pbinom()` ausrechnen können:

```
k <- 0:10
tibble(k = k,
       "Prob(X <= k)" = pbinom(k, 10, 1/6))

# A tibble: 11 x 2
   k 'Prob(X <= k)'
  <int>          <dbl>
1     0          0.162
2     1          0.485
3     2          0.775
4     3          0.930
5     4          0.985
6     5          0.998
7     6          1.00
# i 4 more rows
```

Mit `rbinom()` können wir unabhängige Beobachtungen aus einer Binomialverteilung generieren. Beispielsweise können wir uns vorstellen, dass 20 Personen alle 10 Mal würfeln und zählen, wie oft sie eine 6 bekommen:

```
rbinom(20, 10, 1/6)
```

```
[1] 1 0 1 1 2 0 2 1 4 2 0 2 0 2 1 0 0 1 1 1
```

Die Bernoulli(p)-Verteilung ist die gleiche Verteilung wie die Binomial(1, p)-Verteilung.

4.4 Beispiele von kontinuierlichen Wahrscheinlichkeitsverteilungen

4.4.1 Die kontinuierliche Gleichverteilung

Die kontinuierliche Gleichverteilung, auch Uniformverteilung genannt, haben wir bereits im Glücksradbeispiel kennengelernt. Allgemeiner bezeichnen wir mit $\text{Unif}([a, b])$ die kontinuierliche Gleichverteilung auf dem Intervall $[a, b]$, $a \leq b$. Die Endpunkte kann man zum Intervall rechnen oder auch nicht; die Verteilung ändert sich dadurch nicht. Für $X \sim \text{Unif}([a, b])$ gelten

$$\mathbb{E}(X) = \frac{b-a}{2}$$

und

$$\text{Var}(X) = \frac{(b-a)^2}{12}.$$

Eine Wahrscheinlichkeitsdichte einer Gleichverteilung auf $[a, b]$ ist gegeben durch

$$f_U(x) = \begin{cases} \frac{1}{b-a}, & \text{falls } x \in [a, b], \\ 0, & \text{sonst.} \end{cases}$$

Die Verteilungsfunktion ist

$$F_U(r) = \begin{cases} 0, & \text{falls } r < a, \\ \frac{r-a}{b-a}, & \text{falls } r \in [a, b], \\ 1, & \text{falls } r > b. \end{cases}$$

Die Wahrscheinlichkeitsdichte können wir mit `dunif()` abfragen, hier für eine Gleichverteilung auf $[-\pi, \pi]$:

```
dunif(2, -pi, pi) # = 1/(2*pi)
```

```
[1] 0.1591549
```

```
dunif(4, -pi, pi)
```

```
[1] 0
```

Die Verteilungsfunktion ist in `punif()` implementiert:

```
punif(2, -pi, pi)
```

```
[1] 0.8183099
```

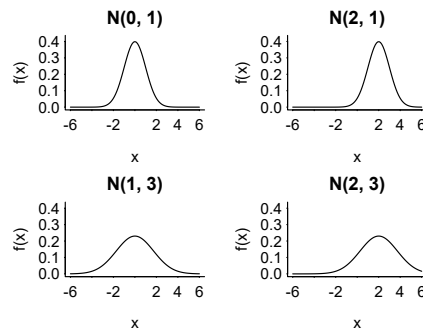


Abbildung 4.6: Wahrscheinlichkeitsdichten von vier Normalverteilungen.

Das heisst, bei einer $\text{Unif}([-\pi, \pi])$ -Verteilung beträgt die Wahrscheinlichkeit, einen Wert unter (oder nicht grösser als) 2 zu beobachten, etwa 82%. Mit `qunif()` können wir die Quantile von kontinuierlichen Gleichverteilungen berechnen und `runif()` können unabhängige Beobachtung aus kontinuierlichen Gleichverteilungen generiert werden:

```
runif(10, -pi, pi)
```

```
[1] 0.8533392 -2.9717588 -2.8516817 2.1348476 1.0033936
[6] 0.2730563 -1.8166756 -0.4526675 -1.4096288 -2.0959474
```

4.4.2 Die Normalverteilung

Von grosser praktischer Bedeutung sind **Normalverteilungen**, wie wir im nächsten Abschnitt sehen werden. Normalverteilungen werden durch zwei Parameter definiert: ihr Erwartungswert μ und ihre Varianz σ^2 . Hat eine Zufallsvariable X eine Normalverteilung mit Parametern μ, σ^2 , so schreibt man $X \sim \text{Normal}(\mu, \sigma^2)$ oder $X \sim \mathcal{N}(\mu, \sigma^2)$. Abbildung 4.6 zeigt die Wahrscheinlichkeitsdichten von vier Normalverteilungen. Links oben sehen Sie die **Standardnormalverteilung**, das heisst, die Normalverteilung mit Mittel 0 und Varianz 1.

Die Wahrscheinlichkeitsdichte f einer $\mathcal{N}(\mu, \sigma^2)$ -Verteilung sieht anfangs wie ein Schrecksgepenst aus:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp \left\{ \frac{-(x - \mu)^2}{2\sigma^2} \right\},$$

wobei $\exp\{\cdot\}$ die Exponentialfunktion ist. Wichtig ist diese Formel für uns jedoch nicht; es reicht, dass Sie einsehen, dass der Parameter μ die zentrale Lage der Normalverteilung regelt und σ^2 wie hoch und breit sie ist. Normalverteilungen haben keine Verteilungsfunktion, die man analytisch darstellen kann.

In R stehen uns die Funktionen `dnorm()`, `pnorm()` und `rnorm()` zur Verfügung für die Wahrscheinlichkeitsdichte und Verteilungsfunktion bzw. fürs Generieren von unabhängigen Beobachtungen aus Normalverteilungen. Dabei ist jedoch zu beachten, dass Normalverteilungen in R nicht durch ihre Varianz, sondern durch ihre Standardabweichung parametrisiert werden. Sei beispielsweise $X \sim \mathcal{N}(3, 16)$. So können wir $\mathbb{P}(X \leq 0)$ wie folgt berechnen; siehe auch

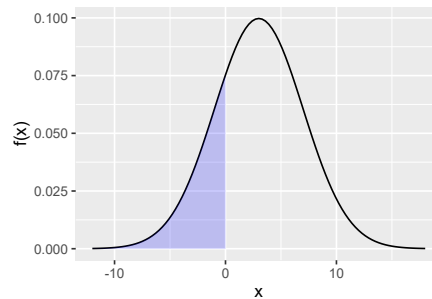


Abbildung 4.7: Die $\mathcal{N}(3, 16)$ -Verteilung. Die Wahrscheinlichkeit, einen Wert unter 0 anzutreffen, entspricht der Fläche unter der Wahrscheinlichkeitsdichte bis 0.

Abbildung 4.7:

```
pnorm(0, mean = 3, sd = sqrt(16))
[1] 0.2266274
```

Aufgabe 4.39 (IQ). Die Verteilung der IQ-Werten in der Gesamtpopulation wird durch eine $\mathcal{N}(100, 15^2)$ -Verteilung modelliert. Verwenden Sie die `pnorm()`- und `qnorm()`-Funktionen, um folgende Fragen zu beantworten. Für die letzten drei Fragen könnten Sie zusätzlich auch noch die Binomialverteilung heranziehen, aber Sie können diese Aufgaben auch 'zu Fuss' lösen.

1. Wie wahrscheinlich ist es, dass eine zufällig ausgewählte Person einen IQ niedriger als 90 hat?
2. Wie wahrscheinlich ist es, dass eine zufällig ausgewählte Person einen IQ grösser als 85 hat?
3. Wie wahrscheinlich ist es, dass eine zufällig ausgewählte Person einen IQ zwischen 110 und 120 hat?
4. Wie wahrscheinlich ist es, dass eine zufällig ausgewählte Person einen IQ hat, der mindestens eine Standardabweichung (also 15 IQ-Punkte) unter dem Mittel liegt?
5. Wie wahrscheinlich ist es, dass eine zufällig ausgewählte Person einen IQ hat, der mindestens eine Standardabweichung (also 15 IQ-Punkte) vom Mittel entfernt liegt?
6. Für welchen IQ-Wert gilt, dass 25% der Bevölkerung einen niedrigeren IQ-Wert hat?
7. Durchschnittliche Intelligenz ist definiert als der IQ der mittleren 45% der Bevölkerung. Zwischen welchen zwei Werten liegt er?
8. Wie wahrscheinlich ist es, dass, wenn zwei Personen zufällig und unabhängig voneinander ausgewählt werden, keine der beiden einen IQ niedriger als 105 hat?

Tipp: Wie wahrscheinlich ist es, dass eine einzige Person einen IQ höher als 105 hat?

9. Wie wahrscheinlich ist es, dass, wenn drei Personen zufällig ausgewählt werden, *genau* eine Person einen IQ niedriger als 90 hat?
10. Wie wahrscheinlich ist es, dass, wenn drei Personen zufällig ausgewählt werden, *mindestens* eine Person einen IQ niedriger als 90 hat? \diamond

Aufgabe 4.40. Wie wahrscheinlich ist es, bei einer normalverteilten Variable (*egal welcher!*) einen zufällig ausgewählten Wert, der weniger als 1; 1,5; und 2 Standardabweichungen vom Mittel entfernt ist, anzutreffen?

Tipp: Beantworten Sie diese Frage für ein paar Normalverteilungen mit unterschiedlichen Mitteln und Standardabweichungen und ziehen Sie eine allgemeine Schlussfolgerung. \diamond

4.5 Der zentrale Grenzwertsatz

Wir greifen das Glücksradbeispiel wieder auf. Nehmen wir an, dass n Leute unabhängig voneinander je ein Mal mit diesem Glücksrad spielen. Dann erhalten wir unabhängige und identisch verteilte Zufallsvariablen $X_1, X_2, \dots, X_n \sim \text{Unif}([0, 360])$. Können wir nun etwas Vernünftiges über ihr durchschnittliches Resultat, also über

$$\bar{X} = \frac{X_1 + \dots + X_n}{n},$$

sagen?

Die Antwort ist 'ja'. Aufgrund der Linearität des Erwartungswerts gilt erstens

$$\begin{aligned}\mathbb{E}(\bar{X}) &= \mathbb{E}\left(\frac{X_1 + \dots + X_n}{n}\right) \\ &= \frac{1}{n}\mathbb{E}(X_1 + \dots + X_n) \\ &= \frac{1}{n}(\mathbb{E}(X_1) + \dots + \mathbb{E}(X_n)) \\ &= \frac{n}{n}\mathbb{E}(X_1) \\ &= \mathbb{E}(X_1).\end{aligned}$$

In diesem konkreten Beispiel also $\mathbb{E}(\bar{X}) = 180$.

Zweitens gilt wegen der Unabhängigkeit, dass

$$\begin{aligned}\text{Var}(\bar{X}) &= \text{Var}\left(\frac{X_1 + \cdots + X_n}{n}\right) \\ &= \frac{1}{n^2} \text{Var}(X_1 + \cdots + X_n) \\ &= \frac{1}{n^2} (\text{Var}(X_1) + \cdots + \text{Var}(X_n)) \\ &= \frac{n}{n^2} \text{Var}(X_1) \\ &= \frac{\text{Var}(X_1)}{n}.\end{aligned}$$

In diesem konkreten Beispiel also $\sigma_{\bar{X}}^2 := \text{Var}(\bar{X}) = \frac{360^2}{12n}$.

Diese Erkenntnisse sind allgemeingültig, wenn wir n unabhängige und identisch verteilte Zufallsvariablen aus einer Verteilung mit existierendem Erwartungswert und endlicher Varianz mitteln:

$$\mu_{\bar{X}} = \mathbb{E}(\bar{X}) = \mathbb{E}(X_1)$$

und

$$\sigma_{\bar{X}}^2 = \text{Var}(\bar{X}) = \frac{1}{n} \text{Var}(X_1).$$

Die Grösse $\sigma_{\bar{X}} := \text{Std}(\bar{X}) = \sqrt{\text{Var}(\bar{X})}$ nennt man den **Standardfehler** (des Mittels). Es gilt also

$$\sigma_{\bar{X}} = \frac{\sigma}{\sqrt{n}}.$$

Aufgabe 4.41. Stellen Sie sich vor, dass 10 Leute unabhängig voneinander eine Zahl mit dem Zufallsgenerator aus Beispiel 4.6 generieren. Was ist der Erwartungswert der Zahl, die sie im Schnitt generieren? Was ist der Standardfehler dieses Mittels?

Hinweis: Den Erwartungswert und die Varianz einer einzelnen generierten Zahl haben wir bereits berechnet. \diamond

Was können wir noch über die Verteilung von \bar{X} sagen? Eine Möglichkeit, diese besser zu verstehen, besteht darin, das Zufallsexperiment mehrmals am Computer zu **simulieren**. Dazu definieren wir zunächst eine Funktion, die das Zufallsexperiment ein Mal simuliert:

```
wheel_one_run <- function(n, min = 0, max = 360) {
  x <- runif(n, min, max)
  return(mean(x))
}
```

Diese Funktion führen wir nun 10'000 Mal aus, hier mit $n = 2$:

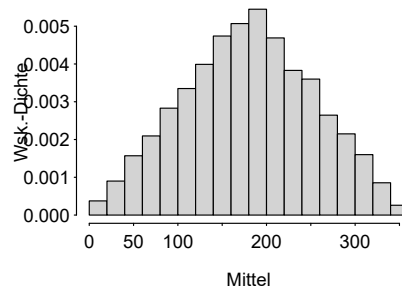


Abbildung 4.8: Verteilung des Mittels von zwei Drehen am Glücksrad.

```
simulation <- replicate(10000, wheel_one_run(n = 2))
```

Mit `hist()` können wir schnell ein Histogramm von `simulation` zeichnen, ohne dass wir zuerst noch ein `tibble` zusammenbasteln müssen. Das Resultat ist Abbildung 4.8.

```
hist(simulation, freq = FALSE, xlim = c(0, 360),
     xlab = "Mittel", ylab = "Wsk.-Dichte", main = "")
```

Aufgabe 4.42 (n vergrößern). Führen Sie die gleiche Simulation durch, aber diesmal mit $n = 5, 25, 100$. Wie sehen die Histogramme aus? ◇

Aufgabe 4.43 (Obenabe-Spiel). Schauen wir uns ein weiteres Beispiel an: Nehmen wir an, n Leute ziehen je eine Karte aus je einem Jassblatt von 36 Karten. Wir bezeichnen mit \bar{X} die durchschnittliche Punktzahl, die die gezogenen Karten beim Obenabe-Spiel wert sind; vgl. Beispiel 4.26. Die entsprechende Simulation können wir wie folgt durchführen:

```
werte <- c(11, 4, 3, 2, 10, 8, 0)
wsk <- c(1/9, 1/9, 1/9, 1/9, 1/9, 1/9, 1/3)
jass_one_run <- function(n, values = wert, probs = wsk) {
  x <- sample(values, n, replace = TRUE, prob = probs)
  mean(x)
}
# Simulation für n = 2
simulation <- replicate(10000, jass_one_run(2))
hist(simulation, freq = FALSE, xlim = c(0, 11),
     xlab = "Mittel", ylab = "Wsk.-Dichte", main = "")
```

Führen Sie diese Simulation durch für $n = 4, 8, 16$. Wie sehen die Histogramme aus? ◇

Das Fazit, das man aus diesen beiden Aufgaben ziehen kann, ist als der zentrale Grenzwertsatz bekannt. ‘Zentral’ bezieht sich hier auf die Wichtigkeit des Satzes, nicht auf den Grenzwert; ‘zentrale Grenzwerte’ gibt es nicht.

Satz 4.44 (zentraler Grenzwertsatz). Seien X_1, \dots, X_n unabhängige und identisch verteilte Zufallsvariablen mit existierendem Erwartungswert μ und endlicher Varianz σ^2 . Dann gilt: Für $n \rightarrow \infty$ nähert sich die Verteilung von $\bar{X}_n = \frac{1}{n}(X_1 + \dots + X_n)$ der Normalverteilung $\mathcal{N}(\mu, \sigma^2/n)$. \diamond

Das Mittel \bar{X}_n von n unabhängigen Beobachtungen einer nicht-pathologischen Zufallsvariablen X ist also selber eine Zufallsvariable, deren Verteilung ungefähr normalverteilt aussieht, wenn n ‘gross genug’ ist. Dies gilt auch dann, wenn die Verteilung von X selber nicht normal ist. Die praktische Bedeutung hiervon ist, dass wir die Normalverteilung verwenden können, um approximative Aussagen über die Wahrscheinlichkeit, bestimmte Mittel zu beobachten, zu machen.

Bemerkung 4.45 (‘gross genug’). Der Ausdruck ‘gross genug’ wirkt zunächst ungenau. Tatsächlich ist die Aussage des zentralen Grenzwertsatzes *nicht*, dass das Mittel von n Beobachtungen normalverteilt ist. Im Glücksradbeispiel können wir kein Mittel niedriger als 0 oder grösser als 360 erhalten, während jede Normalverteilung dieser Möglichkeit eine vielleicht äusserst kleine, aber dennoch positive Wahrscheinlichkeit zuordnen würde. Im Obenabe-Beispiel kann das Mittel von n Werten nur eine Bruchzahl sein, während eine Zufallsvariable mit einer Normalverteilung jeden Wert annehmen kann. Was stattdessen gemeint ist, ist dies: Für jede positive Fehlermarge $\varepsilon > 0$ existiert laut dem zentralen Grenzwertsatz eine natürliche Zahl N , sodass für jede natürliche Zahl $n \geq N$ gilt, dass

$$\mathbb{P}\left(\frac{1}{n}(X_1 + \dots + X_n) \leq r\right)$$

um höchstens ε abweicht von

$$\mathbb{P}(\mu + \sigma_{\bar{X}} Z \leq r),$$

wobei $\mu + \sigma_{\bar{X}} Z$ normalverteilt ist mit Mittel $\mu = \mathbb{E}(X_1)$ und Varianz $\sigma_{\bar{X}}^2 = \sigma^2/n$; dies für jede Zahl r .⁶ Folglich können wir Wahrscheinlichkeiten wie

$$\mathbb{P}(\bar{X} \in [a, b])$$

anhand einer Normalverteilung schätzen. Für grössere n ist diese Schätzung genauer. \diamond

Beispiel 4.46 (Würfelwurf). Die Anzahl Augen, die man beim Würfeln mit einer 6-seitigen Würfel erhält, folgt einer diskreten Gleichverteilung auf $\{1, 2, 3, 4, 5, 6\}$. Ein Wurf hat Erwartungswert $\mu = 3.5$ und Varianz $\sigma^2 \approx 2.92$. Wenn wir 7 Mal würfeln, beträgt die Varianz des durchschnittlichen Wurfs $\sigma^2/7 \approx 0.42$. Die Wahrscheinlichkeit, dass der durchschnittliche Wurf kleiner als 3 ist, beträgt mit dem zentralen Grenzwertsatz ungefähr 22%.

```
pnorm(3, mean = 3.5, sd = sqrt(0.42))
```

```
[1] 0.2202003
```

⁶Die Aussage ist übrigens, dass ein solche Zahl N existiert – nicht, dass wir sie auch einfach finden können.

Die Wahrscheinlichkeit, dass der Durchschnitt grösser als 4.5 ist, beträgt etwa 6%.

```
1 - pnorm(4.5, 3.5, sd = sqrt(0.42))
```

```
[1] 0.06141132
```

Die Wahrscheinlichkeit, dass der Durchschnitt zwischen 3 und 4.5 liegt, ist also etwa $1 - 0.22 - 0.06 \approx 0.72$, also 72%.

Die möglichen Mittel aus sieben Würfeln können alle als Bruchzahl mit Nenner 7 geschrieben werden. Insbesondere ist es nicht möglich, dass das Mittel zwischen $3 + \frac{1}{3.7}$ und $3 + \frac{2}{3.7}$ liegt. Mit der normalen Annäherung fänden wir jedoch, dass diese Wahrscheinlichkeit bei etwa 2.4% und nicht bei 0% liegt.

```
pnorm(3 + 2/21, 3.5, sd = sqrt(0.42)) - pnorm(3 + 1/21, 3.5, sd = sqrt(0.42))
```

```
[1] 0.02355314
```



Aufgabe 4.47. Wir generieren n unabhängige Zahlen mit dem Zufallsgenerator aus Beispiel 4.6. Es gilt also $\mu = 2, \sigma^2 = 2$.

1. Sei $n = 2$. Wie gross ist die Wahrscheinlichkeit, dass die durchschnittliche Zahl kleiner als $1/2$ ist? Was wäre unsere Schätzung für diese Wahrscheinlichkeit, wenn wir den zentralen Grenzwertsatz anwenden würden?
2. Gleiche Fragen für $n = 20$ und $n = 200$.
3. Sei $n = 30$. Wie gross ist laut dem zentralen Grenzwertsatz die Wahrscheinlichkeit, dass die durchschnittliche Zahl zwischen 1.8 und 2.2 liegt?



Teil II

Schätzungen

Kapitel 5

Zufallsstichproben

In Kapitel 3 sind wir davon ausgegangen, dass die Daten, die uns zur Verfügung standen, die ganze Population, für die wir uns interessierten, darstellten. Eine Population kann eine endliche Menge von tatsächlichen oder potenziellen Beobachtungen sein, wie zum Beispiel die Wahlpräferenz aller AmerikanerInnen, die vorhaben, zur Urne zu gehen. Öfters sollte man die Population von Interesse aber eher als einen abstrakteren datengenerierenden Mechanismus verstehen. Ein simples Beispiel hierfür ist die Kreisscheibe aus Abbildung 4.1, die Daten aus einer kontinuierlichen Gleichverteilung generiert.

In der Regel stellen die Daten, die gesammelt wurden, nur einen kleinen Teil der Population von Interesse da bzw. sie besteht aus endlich vielen Belegen, die von einem Mechanismus generiert wurden, der theoretisch unendlich viele solche Belege generieren könnte. Man sagt, dass solche Daten eine **Stichprobe** der Population von Interesse bilden. Diese Stichprobe gibt einem notwendigerweise ein unvollständiges Bild der Population, aus der sie stammt. Das Ziel ist es dann, anhand der Stichprobe Rückschlüsse über die Population, aus der die Stichprobe stammt, zu ziehen. Von Interesse sind also nicht sosehr etwa die zentrale Tendenz und Streuung in der Stichprobe, sondern die zentrale Tendenz und Streuung in der Population.

Im Folgenden gehen wir davon aus, dass uns eine (**einfache**) **Zufallsstichprobe** (*(simple) random sample*) zur Verfügung steht und dass die relevante Population als unendlich gross aufgefasst werden kann. Genauer sei P die Verteilung der Population, über die wir Aussagen machen möchten. Dann besteht eine einfache Zufallsstichprobe aus Beobachtungen X_1, X_2, \dots, X_n , die identisch und unabhängig nach P verteilt sind.¹ Dieses Kapitel widmet sich der Frage, wie wir anhand einer (einfachen) Zufallsstichprobe Rückschlüsse über die Population machen können. Insbesondere die **Schätzung** der zentralen Tendenz (vor allem des Mittels) und der Streuung

¹Eine etwas kompliziertere Art Stichprobe wäre die **geschichtete Zufallsstichprobe** (*stratified random sample*). Hierzu teilt man die Population von Interesse (z.B. Studierende an der Universität Freiburg) in Teilpopulationen auf (z.B. Studierende an der Philosophischen Fakultät, an der Theologischen Fakultät, an der Naturwissenschaftlichen Fakultät usw.). Dann zieht man zufällige Stichproben innerhalb jeder Gruppe. Somit hat man in der Stichprobe garantiert aus jeder Gruppe einige Beobachtungen (z.B. würde die Stichprobe sowieso Studierende an der Theologischen Fakultät beinhalten), aber ist es dennoch möglich, Aussagen über das Mittel und die Streuung in der Gesamtpopulation zu machen. Letzteres tut man grundsätzlich, indem man die Gruppenergebnisse nach der Gesamtgruppengröße gewichtet. Geschichtete Zufallsstichproben werden wir in diesem Skript nicht behandeln.

(Varianz und Standardabweichungen) steht dabei im Fokus.

Fünf Sekunden kritisches Überlegen zeigen aber, dass wir es in der Praxis nie wirklich mit Zufallsstichproben zu tun haben. Auf dieses Problem wird am Ende dieses Kapitels näher eingegangen. Bis dahin bitte ich um etwas *willing suspension of disbelief*.

5.1 Stichprobenfehler

Zufallsstichproben widerspiegeln nicht perfekt jeden Aspekt der Population, aus der sie stammen. Um dies besser einzusehen, lohnt es sich, Zufallsstichproben aus Populationen zu ziehen, deren Eigenschaften wir kennen. So können wir sehen, wie stark diese von der Population und voneinander abweichen. Dies können wir tun, indem wir am Computer Stichproben simulieren.

Aufgabe 5.1 (Daten aus Normalverteilung). Mit dem R-Code unten können Sie einschätzen, wie Stichproben aus einer normalverteilten Population mit Mittel 3 und Standardabweichung 7 aussehen. Zunächst habe ich hier die Stichprobengrösse auf 20 festgelegt, aber mit dieser Zahl sollten Sie selber herumspielen. Führen Sie diese Befehle aus und zwar nicht ein Mal, sondern mehrmals. Bemerken Sie dabei, wie (un)ähnlich sich die Histogramme von Stichproben aus einer Normalverteilung sind.

```
groesse <- 20
x <- rnorm(n = groesse, mean = 3, sd = 7)
hist(x, col = "lightgrey", xlim = c(3 - 4*7, 3 + 4*7))
```



Aufgabe 5.2 (Daten aus Gleichverteilung). Passen Sie den Code oben so an, dass er Stichproben aus einer Gleichverteilung mit Bereich $[-5, 5]$ statt aus einer Normalverteilung generiert. Die Funktion, die Sie dazu brauchen, ist `runif()`. Neben dem `n`-Parameter hat diese Funktion einen `min`- und `max`-Parameter, mit denen der Bereich der Gleichverteilung eingestellt wird. Passen Sie die `xlim`-Werte beim Histogramm entsprechend an. Lassen Sie den angepassten Code dann mehrmals laufen. Sehen die einzelnen Histogramme wie Gleichverteilungen aus? Was ist, wenn Sie die Stichprobengrösse vergrössern?



Fazit: Zufallsstichproben sind imperfekte Abbildungen der Population, aus der sie stammen. Diese Gegebenheit bezeichnet man als **Stichprobenfehler** (*sampling error*). Rückschlüsse über die Population verstehen sich also als **Schätzungen**. Sowohl das Schätzen selbst als auch das Quantifizieren ihrer Genauigkeit sind das Ziel der **Inferenzstatistik**.

5.2 Die zentrale Tendenz und Streuung schätzen

In Kapiteln 3 und 4 wurden das Mittel (Erwartungswert) und die Varianz als Masse der zentralen Tendenz und der Streuung einer Population eingeführt. Wenn uns nun eine Stichprobe aus dieser Population P (also Beobachtungen X_1, \dots, X_n mit Verteilung P) zur Verfügung steht, so

ist es naheliegend, die **empirische Verteilung** \hat{P} durch die empirische Verteilungsfunktion \hat{F} zu definieren, wo

$$\hat{F}(r) := \frac{1}{n} \sum_{i=1}^n \mathbb{1}\{X_i \leq r\}$$

für alle reellen Zahlen r . In Worten: Wir betrachten die Stichprobe als eine eigene, endliche Verteilung. Wir können nun das Mittel (den Erwartungswert) und die Varianz und Standardabweichung von \hat{P} mit den Formeln aus Kapitel 3 berechnen und diese als unsere Schätzung für die entsprechenden Merkmale von P verwenden.

Hierbei stellt sich die Frage, wie gut solche sogenannten *plug-in*-Schätzer sind.

Definition 5.3 (Estimand, Schätzung und Schätzer). Ein **Estimand** μ ist eine numerische Eigenschaft einer Population, die wir anhand von Daten schätzen wollen.

Eine **Schätzung** ist eine in der Regel auf Daten und Vorwissen basierte Zahl, die eine Annäherung des Estimanden darstellt.

Ein **Schätzer** ist eine Vorschrift (Funktion), die festlegt, wie Daten und Vorwissen zu einer Schätzung kombiniert werden sollen. \diamond

Beispielsweise wäre das (unbekannte) Mittel der Population, aus der die Daten generiert wurden, ein Estimand. Ein Schätzer wäre eine Vorschrift wie ‘summiere alle Beobachtungen und teile durch die Anzahl Beobachtungen’. Wird dieser Schätzer auf eine konkrete Stichprobe angewandt, so erhält man eine konkrete Zahl – die Schätzung.

Über einzelne Schätzungen können wir keine allgemeinen Aussagen machen, aber wir können schon die Eigenschaften von Schätzern untersuchen. Beim Vergleichen von Schätzern sind die wichtigsten Gütekriterien die **Verzerrung**, die **Konsistenz** und die **Varianz** der Schätzer.

Definition 5.4 (Verzerrung). Sei g ein Schätzer und sei θ ein Estimand einer Population P . Die Verzerrung (oder das **Bias**) von g als Schätzer von θ ist

$$\mathbb{E}(g(X_1, \dots, X_n)) - \theta,$$

wo X_1, \dots, X_n unabhängig und nach P verteilt sind.

Gilt $\mathbb{E}(g(X_1, \dots, X_n)) = \theta$ für jeden möglichen Wert von θ , so nennen wir g einen **unverzerrten** oder **erwartungstreuen** Schätzer. Anders gesagt: Ein Schätzer ist unverzerrt, wenn er, gemittelt über zahllose Zufallsstichproben der gleichen Grösse, die richtige Antwort liefert. \diamond

Eine genaue Definition von Konsistenz ist etwas schwieriger zu geben, ohne weitere mathematische Konzepte einzuführen. Daher soll eine konzeptuelle Definition hier genügen.

Definition 5.5 (Konsistenz). Ein Schätzer g heisst konsistent (als Schätzer von θ), falls die Grösse

$$\mathbb{E}(|g(X_1, \dots, X_n) - \theta|)$$

beliebig klein wird für grosse n . Anders gesagt ist ein Schätzer konsistent für θ , wenn die Unterschiede zwischen den resultierenden Schätzungen und θ beliebig klein gemacht werden, wenn die Stichprobengrösse wächst. \diamond

Definition 5.6 (Varianz eines Schätzers). Die Varianz eines Schätzers g ist definiert als

$$\text{Var}(g(X_1, \dots, X_n)).$$

Sie erfasst, wie sehr die resultierenden Schätzungen je nach Zufallsstichprobe voneinander abweichen. \diamond

Im Idealfall sind Schätzer unverzerrt und konsistent; zudem sollte ihre Varianz niedrig sein. Wie wir (spätestens im Ausblick) sehen werden, kann man jedoch oft die Varianz eines Schätzers senken, indem man bei der Verzerrung Abstriche macht (**bias–variance tradeoff**).

Beispiel 5.7 (unverzerrt, aber nutzlos). Unverzerrtheit ist kein ausreichendes Kriterium, denn in manchen Situationen ist der einzige unverzerrte Schätzer ziemlich nutzlos. Sei beispielsweise $P = \text{Binomial}(n, p)$ und definiere $\theta := (1 - p)^n$ (Dümbgen, 2016, S. 49). Wir beobachten $X \sim P$. Man kann nun zeigen, dass es nur einen erwartungstreuen Schätzer $g(X)$ von θ gibt, und zwar

$$g(X) := \begin{cases} 1, & \text{falls } X = 0, \\ 0, & \text{sonst.} \end{cases}$$

Während θ je nach p alle Werte im Intervall $[0, 1]$ haben kann, liefert g immer 0 oder 1 als Schätzung. Folglich ist dieser erwartungstreue Schätzer nicht konsistent. \diamond

5.2.1 Das Stichprobenmittel

Das **Stichprobenmittel** ist der plug-in-Schätzer des Populationsmittels μ :

$$\hat{\mu} := \bar{X} := \frac{1}{N}(X_1 + \dots + X_n).$$

In Abschnitt 4.5 haben wir gezeigt, dass

$$\mathbb{E}(\bar{X}) = \mu.$$

Also ist das Stichprobenmittel ein unverzerrter Schätzer des Populationsmittels. Ausserdem wissen wir, dass die Varianz von \bar{X} gleich $\frac{\sigma^2}{n}$ ist, mit σ^2 der Varianz von P . Also wird die Varianz von \bar{X} für grosse n beliebig klein. Folglich ist \bar{X} auch ein konsistenter Schätzer von μ . Laut dem Satz von Gauss ist das Stichprobenmittel unter allen unverzerrten Schätzern des Populationsmittels ausserdem der Schätzer mit der niedrigsten Varianz.

Satz 5.8 (Gauss). Sei X_1, \dots, X_n eine einfache Stichprobe aus der Verteilung P . Dann gilt für

jeden unverzerrten Schätzer $g(X_1, \dots, X_n)$ von μ , dass

$$\text{Var}(\bar{X}) \leq \text{Var}(g(X_1, \dots, X_n)).$$

Es ist unter Umständen aber möglich, unverzerrte Schätzer mit niedrigerer Varianz zu konstruieren, wenn die Stichprobe keine einfache Stichprobe ist. \diamond

Beispiel 5.9. Seien X_1, \dots, X_n unabhängige Bernoulli(p)-verteilte Zufallsvariablen. Dann ist $\frac{1}{n}(X_1 + \dots + X_n)$ ein unverzerrter und konsistenter Schätzer von p . Unter allen unverzerrten Schätzern von p hat er ausserdem die geringste Varianz. \diamond

Beispiel 5.10 (gewichtetes Mittel). Wir nehmen an, dass P_1 und P_2 Normalverteilungen mit gleichem, aber unbekanntem Mittel und ungleichen, aber bekannten Varianzen sind: $P_1 = \mathcal{N}(\mu, 1)$, $P_2 = \mathcal{N}(\mu, 4)$. Wir beobachten nun die unabhängigen Zufallsvariablen $X_1 \sim P_1$, $X_2 \sim P_2$ und wollen anhand dieser μ schätzen.

Eine erste Möglichkeit ist, μ durch das Mittel von X_1 und X_2 zu schätzen. Dieser Schätzer ist unverzerrt, denn

$$\mathbb{E}\left(\frac{X_1 + X_2}{2}\right) = \frac{1}{2}(\mathbb{E}(X_1) + \mathbb{E}(X_2)) = \mu.$$

Dank der Unabhängigkeit von X_1, X_2 lässt sich seine Varianz leicht berechnen:

$$\text{Var}\left(\frac{X_1 + X_2}{2}\right) = \frac{1}{4}(\text{Var}(X_1) + \text{Var}(X_2)) = \frac{5}{4}.$$

Eine zweite Möglichkeit ist, X_2 einfach zu ignorieren und den Wert von X_1 als Schätzung von μ zu verwenden. Auch dieser Schätzer ist unverzerrt, denn $\mathbb{E}(X_1) = \mu$. Ausserdem ist seine Varianz niedriger, denn $\text{Var}(X_1) = 1 < 5/4$.

Noch besser wäre jedoch, die Beobachtungen X_1, X_2 passend zu gewichten, etwa mit der Inversen ihrer Varianz:

$$g(X_1, X_2) := \frac{1}{1 + \frac{1}{4}} \left(X_1 + \frac{1}{4} X_2 \right) = \frac{4}{5} \left(X_1 + \frac{1}{4} X_2 \right).$$

Auch dieser Schätzer ist erwartungstreu, denn

$$\mathbb{E}(g(X_1, X_2)) = \frac{4}{5} \left(\mathbb{E}(X_1) + \frac{1}{4} \mathbb{E}(X_2) \right) = \mu.$$

Seine Varianz ist aber niedriger:

$$\text{Var}(g(X_1, X_2)) = \frac{4^2}{5^2} \left(\text{Var}(X_1) + \frac{1}{4^2} \text{Var}(X_2) \right) = \frac{4}{5}.$$

Der Satz von Gauss trifft hier nicht zu, da X_1, X_2 keine einfache Zufallsstichprobe darstellt: Die Beobachtungen stammen aus unterschiedlichen Verteilungen. \diamond

Aufgabe 5.11 (Vergleich von Schätzern von μ). Sei X_1, \dots, X_n eine einfache Zufallsstichprobe aus der Verteilung P und sei μ das Mittel (Erwartungswert) von P . Entscheiden Sie für die folgenden Schätzer, ob sie μ unverzerrt schätzen, ob sie konsistent sind und ob ihre Varianz kleiner oder grösser als jene von \bar{X} ist. Sie dürfen der Einfachheit halber annehmen, dass $\mathbb{P}(X_1 = X_2) = 0$.

$$\begin{aligned} g_1(X_1, \dots, X_n) &:= X_1; \\ g_2(X_1, \dots, X_n) &:= \bar{X} + \frac{1}{n}; \\ g_3(X_1, \dots, X_n) &:= \frac{n-1}{n} \bar{X}; \\ g_4(X_1, \dots, X_n) &:= \begin{cases} \bar{X} - 1, & \text{falls } X_1 < X_2, \\ \bar{X}, & \text{falls } X_1 = X_2, \\ \bar{X} + 1, & \text{falls } X_1 > X_2. \end{cases} \end{aligned}$$

Definieren Sie ausserdem einen Schätzer g_5 , der μ unverzerrt schätzt und konsistent ist, aber eine höhere Varianz als das Stichprobenmittel hat. \diamond

5.2.2 Die Stichprobenvarianz

Bei der Schätzung des Populationsmittels schneidet der plug-in-Schätzer hervorragend ab. Wir überprüfen nun, ob man die Populationsvarianz σ^2 ebenso mit diesem Ansatz schätzen kann. Wir verzichten hier auf formale Beweise und gehen dieser Frage mittels einer Simulation nach. Dazu betrachten wir eine Zufallsstichprobe X_1, \dots, X_n aus der $\text{Unif}([-5, 5])$ -Verteilung. Diese Varianz dieser Verteilung beträgt

$$\sigma^2 = \frac{(5 - (-5))^2}{12} \approx 8.33.$$

Wie in Abschnitt 4.5 definieren wir zunächst eine R-Funktion, die eine Stichprobe der Grösse n aus der $\text{Unif}([-5, 5])$ -Verteilung generiert und die selbst geschriebene `pop_var()`-Funktion (siehe Aufgabe 3.8) auf sie anwendet. Dann führen wir diese Funktion 10'000 Mal aus, zunächst mit $n = 25$, und ermitteln den Schnitt der berechneten Varianzschätzungen:

```
var_one_run <- function(n, min = -5, max = 5) {
  x <- runif(n, min = min, max = max)
  pop_var(x)
}
simulation <- replicate(10000, var_one_run(n = 25))
mean(simulation)
[1] 7.99864
```

Aufgabe 5.12 (unterschiedliche n). Führen Sie die gleiche Simulation für $n = 16, 9, 4, 1$ durch. Vergleichen Sie jeweils die durchschnittliche Varianzschätzung mit dem Populationswert $\sigma^2 \approx$

8.33.

◇

Wie diese Simulation belegt, ist der plug-in-Schätzer der Populationsvarianz *nicht* erwartungstreu: Im Schnitt liefert er zu tiefe Schätzungen. Diese Verzerrung ist ausgeprägter für kleinere Stichproben und lässt sich intuitiv so verstehen: Wenn wir Zufallsstichproben von je nur einer Beobachtung aus einer Population ziehen, dann gibt es keine Streuung innerhalb der Stichprobe – die Beobachtung kann ja nicht von sich selbst abweichen. Bei der kleinst möglichen Stichprobe ist die Unterschätzung der Populationsvarianz also maximal. In grösseren Stichproben ist dieses Problem in stets geringerem Ausmass vorhanden.² Der plug-in-Schätzer ist jedoch schon konsistent.

Die Unterschätzung der Populationsvarianz lässt sich einfach korrigieren. Wie Sie anhand von Aufgabe 5.12 ergibt die plug-in-Schätzung von σ^2 im Schnitt den Wert $\frac{n-1}{n}\sigma^2$. Um diese Unterschätzung zu korrigieren, wird die **Stichprobenvarianz** (Kürzel: S^2) nicht wie die Populationsvarianz (σ^2), sondern folgendermassen berechnet:

$$S^2 := \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2 = \frac{n}{n-1} \left(\frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2 \right).$$

Wegen des Korrekturfaktors kann S^2 nur berechnet werden, wenn $n \geq 2$; sonst wird eben durch 0 geteilt. So erhält man einen erwartungstreuen und konsistenten Schätzer der Populationsvarianz. Den Konsistenznachweis schenken wir uns.

Die Stichprobenvarianz kann mit der R-Funktion `var()` berechnet werden.

Bemerkung 5.13 (Verteilung der Stichprobenvarianz im Falle normalverteilter Daten). Der zentrale Grenzwertsatz (Abschnitt 4.5) erlaubt es uns, ungefähre Wahrscheinlichkeitsaussagen über Stichprobenmittel von genügend grossen Stichproben zu machen. Für die Varianz besteht ein ähnliches Resultat, worauf wir hier aber nicht näher eingehen. Für den Spezialfall, dass die Zufallsstichprobe X_1, \dots, X_n aus unabhängigen Beobachtungen aus einer $\mathcal{N}(\mu, \sigma^2)$ -Verteilung stammen, kann man jedoch genaue Aussagen über die Stichprobenvarianz S^2 machen. Es gilt nämlich, dass die Grösse

$$(n-1) \frac{S^2}{\sigma^2}$$

die gleiche Verteilung hat wie

$$Z_1^2 + \dots + Z_{n-1}^2,$$

wo Z_1, \dots, Z_{n-1} unabhängige Zufallsvariablen mit Verteilung $\mathcal{N}(0, 1)$ sind. Die Verteilung von $Z_1^2 + \dots + Z_{n-1}^2$ nennt man die **Chi-Quadrat-Verteilung** mit $n-1$ Freiheitsgraden (χ_{n-1}^2). Abbildung 5.1 zeigt die Wahrscheinlichkeitsdichten einiger χ^2 -Verteilungen.

Diese Tatsache können wir verwenden, um Wahrscheinlichkeitsaussagen über die Stichprobenvarianz normalverteilter Daten zu machen. Wenn wir beispielsweise $n = 10$ unabhängige

²Etwas genauer ist der Grund des Problems, dass wir mit der `pop_var()`-Funktion zuerst das Mittel der Population durch das Stichprobenmittel schätzen. Wenn wir das Populationsmittel als bekannt voraussetzen würden (hier: $\mu = 0$), so würde die Formel schon einen unverzerrten Schätzer darstellen.

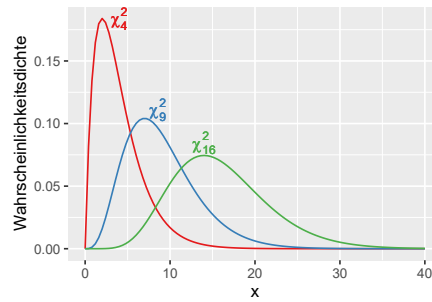


Abbildung 5.1: Wahrscheinlichkeitsdichten der χ^2 -Verteilungen mit 4, 9 und 16 Freiheitsgraden.

Beobachtungen aus einer $\mathcal{N}(3, 12)$ -Verteilung generieren und wissen wollen, wie gross die Wahrscheinlichkeit ist, dass $S^2 \leq 14$, so bemerken wir, dass

$$\begin{aligned} \mathbb{P}(S^2 \leq 14) &= \mathbb{P}\left(\frac{10-1}{12}S^2 \leq \frac{(10-1)14}{12}\right) \\ &= \mathbb{P}\left(\frac{n-1}{\sigma^2}S^2 \leq 10.5\right) \\ &= \mathbb{P}(H^2 \leq 10.5), \end{aligned}$$

wo $H^2 \sim \chi_9^2$. Mit R berechnen wir den genauen Wert:

```
pchisq(10.5, df = 9)
[1] 0.6884577
```

Also etwa 69%. ◇

5.2.3 Die Stichprobenstandardabweichung

Aus dem gleichen Grund, weshalb die Stichprobenvarianz nicht wie die Populationsvarianz berechnet wird, wird die **Stichprobenstandardabweichung** (S) nicht wie die Populationsstandardabweichung berechnet, sondern wie folgt:

$$S := \sqrt{S^2} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}.$$

In R kann hierfür die `sd()`-Funktion verwendet werden. Die Stichprobenstandardabweichung ist ein konsistenter Schätzer der Populationsstandardabweichung. Sie ist im Gegensatz zur Stichprobenvarianz jedoch *kein* erwartungstreuer Schätzer: Die Stichprobenstandardabweichung unterschätzt die Populationsstandardabweichung ganz leicht. Diese Schätzung zu korrigieren, ist im besten Fall schwierig und meistens unmöglich, weshalb man sie einfach in Kauf nimmt.³

³Nur weil $g(X_1, \dots, X_n)$ einen Estimanden θ unverzerrt schätzt, heisst das nicht, dass $h(g(X_1, \dots, X_n))$ den Estiman-

Es kommt eigentlich quasi nie vor, dass man für einen Datensatz die Populationsvarianz und -standardabweichung berechnet. Spricht man in diesem Kontext von der Varianz und Standardabweichung, meint man also die Stichprobenvarianz und die Stichprobenstandardabweichung. Bei grossen Populationen oder Stichproben ergeben beide Berechnungsmethoden ohnehin nahezu das gleiche Resultat.

Aufgabe 5.14 (Verteilung der Stichprobenstandardabweichung normalverteilter Daten). Sei X_1, \dots, X_{10} eine Zufallsstichprobe aus einer $\mathcal{N}(-4, 8)$ -Verteilung. Berechnen Sie die Wahrscheinlichkeit, dass die Stichprobenstandardabweichung S mindestens 3.5 beträgt. Dazu können Sie wie in Bemerkung 5.13 vorgehen. \diamond

5.3 Nicht-zufällige Stichproben

Zwei grosse Vorteile von Zufallsstichproben sind, dass sie unverzerrte Schätzungen des Populationsmittels und der Populationsvarianz liefern und dass der zentrale Grenzwertsatz auf sie zutrifft. In der Praxis ist es jedoch schwierig, eine Zufallsstichprobe aus einer einigermaßen interessanten Population zu ziehen. Wenn wir etwa anhand einer Zufallsstichprobe die Englischkenntnisse bei Erwachsenen kosovarischer Herkunft im Kanton Sankt-Gallen charakterisieren möchten, brauchen wir zuerst eine vollständige Liste aller Erwachsenen kosovarischer Herkunft in SG. Dann müssten wir zufällig eine Stichprobe von ihnen auswählen und die Ausgewählten alle von einer Teilnahme an der Studie überzeugen: Sobald sich eine Person weigert, mitzumachen, hätten wir keine Zufallsstichprobe aus der ursprünglichen Population mehr. Stattdessen hätten wir eine Stichprobe aus der Population der in SG wohnhaften Erwachsenen kosovarischer Herkunft, die bei einer solchen Studie mitmachen möchten. Unsere Schätzungen würden sich in erster Linie auf diese neue Population beziehen, nicht auf die Population, für die wir uns anfangs interessierten. Ausserdem bestünde die Stichprobe nicht aus unabhängigen Beobachtungen, denn eine Person kann nur ein Mal ausgewählt werden.

Das Beispiel macht auch klar, was die Konsequenz hiervon ist: Während eine Zufallsstichprobe eine unverzerrte Schätzung des Mittels der Population, die eigentlich von Interesse ist, liefert, wäre es bei einigen Weigerungen möglich, dass einige der ausgewählten Personen nicht zur Teilnahme bereit sind, gerade weil sie ihre Englischkenntnisse als ungenügend einschätzen oder weil sie sprachwissenschaftliche Forschung für uninteressant halten. Die Übrigen dürften also tendenziell eher gut im Englischen sein oder sich eher für Sprachen interessieren. Das Mittel dieser Stichprobe dürfte entsprechend das Mittel der Population, die ursprünglich von Interesse war, eher über- als unterschätzen.

Fazit: Statt Zufallsstichproben werden in den Sozialwissenschaften meistens nicht-zufällige Stichproben verwendet. Die Konsequenz davon ist, dass man sich bei der Interpretation der Ergebnisse mehr Gedanken machen muss, wenn man Rückschlüsse über eine Population ziehen möchte, als wenn die Stichprobe zufällig ausgewählt worden wäre.

den $h(\theta)$ unverzerrt schätzt. Beispielsweise schätzt das Stichprobenmittel \bar{X} das Populationsmittel μ unverzerrt, aber die Grösse $(\bar{X})^2$ überschätzt μ^2 . Hiervon kann man sich überzeugen, indem man eine Population mit $\mu = 0$ betrachtet. Im Allgemeinen handelt es sich um eine Überschätzung, wenn h streng konvex ist, und um eine Unterschätzung, wenn h streng konkav ist.

- Eine Meinungsumfrage auf Twitter erreicht tendenziell Menschen ähnlicher Meinung. Aber sogar die angesehensten Meinungsforschungsinstitute können keine Zufallsstichproben organisieren: Bei Telefonumfragen in den USA nehmen nur etwa 10% der Ausgewählten teil.
- Wer ohne Entgelt einen langen Fragebogen zu seinem mehrsprachigen Verhalten ausfüllt, findet Mehrsprachigkeit tendenziell wichtiger als jemand, der nach der dritten Frage das Browserfenster schliesst oder den Fragebogen nicht einmal erhalten hat (bei einer Erhebung nach dem Schneeballprinzip).
- Muster in einer gut ausgebildeten Stichprobe mit überdurchschnittlichem sozioökonomischem Status (z.B. Universitätsstudierende) dürften nicht auf Populationen mit niedrigerem Bildungsniveau oder sozioökonomischen Status generalisieren lassen. Dies ist natürlich vor allem relevant, wenn Bildung und der sozioökonomische Status wichtig für den Forschungsgegenstand sind. Wenn man bereit ist, anzunehmen, dass diese Faktoren nur einen minimalen Effekt auf die Befunde haben, kann man zuversichtlicher generalisieren. Ob eine solche Annahme berechtigt ist, ist eine sachlogische – keine statistische – Frage.

Soll man jetzt verzweifeln, dieses Skriptum bereits in der Mitte des Semesters ins Lagerfeuer werfen? Meines Erachtens sollte man damit bis zum Semesterende warten. Einerseits liefert die Betrachtung des idealisierten Falles der Zufallsstichprobe ein *Modell* für den Umgang mit Zufallsdaten. Entspricht der *data-generating mechanism* in einer spezifischen Studie diesem Idealfall nicht, so kann es trotzdem möglich sein, dieses Modell zu verfeinern, sodass es besser zum konkreten Fall passt. Andererseits stellt es sich heraus, dass sich unsere Erkenntnisse recht gut auf einige häufige Situationen übertragen lassen – beispielsweise auf Experimente, in denen die Versuchspersonen zwar keine Zufallsstichprobe aus irgendeiner Population darstellen, aber in denen diese schon zufällig den Konditionen des Experiments zugeordnet werden.

Kapitel 6

Die Ungenauigkeit von Schätzungen schätzen

Eine unumgängliche Gegebenheit beim Arbeiten mit Stichproben ist, dass wir Eigenschaften von Populationen nur *schätzen* können. Die Frage stellt sich, wie genau diese Schätzungen denn sind. Interessanterweise wissen wir dies in der Regel auch nicht genau, weshalb diese Ungenauigkeit *auch* anhand der Stichprobe geschätzt werden muss.

Das Ziel dieses Kapitels ist es, anhand eines Beispiels zu illustrieren, wie man mit einer Stichprobe die Unsicherheit in einer Parameterschätzung einschätzen kann. Dazu introduziert dieses Kapitel den sog. **Bootstrap**, ein flexibles, mechanistisches Verfahren, um Ungenauigkeit einzuschätzen. Danach wird gezeigt, wie man anhand des zentralen Grenzwertsatzes (vgl. Abschnitt 4.5) das Gleiche machen kann.

Im Folgenden arbeiten wir mit einem Datensatz aus der Studie von DeKeyser et al. (2010). Diese untersuchten, wie das Alter, in dem Migrant:innen angefangen haben, eine Zweitsprache zu lernen (*age of acquisition*, AOA), mit ihrer Leistung bei einer Grammatikaufgabe zusammenhängt (*grammaticality judgement task*, GJT). Die Teilnehmenden waren russische Migrant:innen in Israel und in Nordamerika. Die Grammatikaufgabe bestand aus 204 richtig/falsch-Items. In den nächsten Kapiteln werden wir uns mit dem Zusammenhang zwischen AOA und GJT befassen; hier verwenden wir den Datensatz von DeKeyser et al. (2010), um zu zeigen, wie man die Unsicherheit bei Stichprobenschätzungen quantifizieren kann.

Aufgabe 6.1. Der Datensatz `dekeyser2010.csv` enthält die AOA- und GJT-Daten der russischen MigrantInnen in Nordamerika. Lesen Sie diesen Datensatz in R ein. Zeichnen Sie die Grafik in Abbildung 6.1 selbst. Berechnen Sie zudem das Mittel der GJT-Werte. ◇

6.1 Stichprobenmittel variieren

Gehen wir davon aus, dass die GJT-Daten in der ganzen Population genau so verteilt wären wie in Abbildung 6.1. Dies ist nur eine Annahme für didaktische Zwecke. Als Forschende haben wir keinen Zugriff zur ganzen Population, d.h., wir wissen eigentlich nicht, wie dieses Populationsverteilung aussieht. Stattdessen müssen wir uns mit Stichproben begnügen. Aber

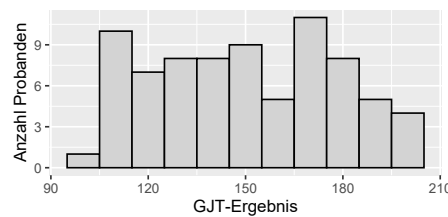


Abbildung 6.1: Histogramm der GJT-Daten aus der Nordamerika-Studie von DeKeyser et al. (2010). Diese Grafik sollten Sie selber zeichnen (Aufgabe 6.1).

nehmen wir vorübergehend an, dass die Daten in der Population genau so verteilt wären wie in dieser Stichprobe.

Wie schon in Kapitel 5 besprochen, bilden die Mittel von Zufallsstichproben mit der gleichen Grösse eine Stichprobenmittelverteilung, deren Mittel gleich dem Populationsmittel ist ($\mu_{\bar{x}} = \mu$). Abbildung 6.2 zeigt exemplarisch fünf Stichproben mit Grösse 20 aus dieser GJT-Population und die Verteilung der Mittel von 20'000 Stichproben mit je 20 Beobachtungen aus der Population; bereits gezogene GJT-Werte konnten dabei nochmals gezogen werden (*sampling with replacement*). Die Standardabweichung der Stichprobenmittelverteilung, der Standardfehler (siehe Kapitel 5), beträgt 6.07 Punkte. 2.5% der Stichprobenmittel sind kleiner als 138.95; 2.5% sind grösser als 162.60. 95% aller Stichprobenmittel liegen also in einem Intervall von $162.60 - 138.95 = 23.65$ Punkten. Der Standardfehler oder die Breite eines solchen Intervalls wären sinnvolle Masse für die Genauigkeit, mit der man mit einer Stichprobe einen Populationsparameter (hier: das Mittel) schätzen kann.

Unser Problem ist aber, dass wir die Stichprobenmittelverteilung nur generieren können, wenn wir Zugriff zur ganzen Population haben. Wenn wir nur über eine Stichprobe verfügen, müssen wir den Standardfehler bzw. die Breite solcher Intervalle anhand der Stichprobe schätzen.

6.2 Das plug-in-Prinzip und der Bootstrap

Enter the plug-in principle. Abbildung 6.2 zeigt zwar, dass jede einzelne Stichprobe die Population nur imperfekt widerspiegelt. Aber gleichzeitig ist diese Widerspiegelung das Beste, was wir in der Praxis haben.¹ Um den Standardfehler bzw. die Form der Stichprobenmittelverteilung zu schätzen, können wir die Stichprobe als Stellvertreter der Population betrachten.²

Beispiel 6.2 (rote Stichprobe). Abbildung 6.3 auf Seite 121 zeigt das Vorgehen. Zur Verfügung steht uns die erste (rote) Stichprobe aus Abbildung 6.2. Wir tun nun, als ob die GJT-Population genau so wie diese Stichprobe verteilt wäre, denn wir haben keine besseren Anknüpfungspunkte. Um die Stichprobenmittelverteilung zu generieren, ziehen wir Zufallsstichproben mit Grösse

¹Sogenannte bayessche Methoden erlauben es einem aber, auch Informationen, die man nicht aus den Daten selber ableiten kann, in der Analyse zu berücksichtigen.

²Dieser Abschnitt wurde von Hesterberg (2015) inspiriert.

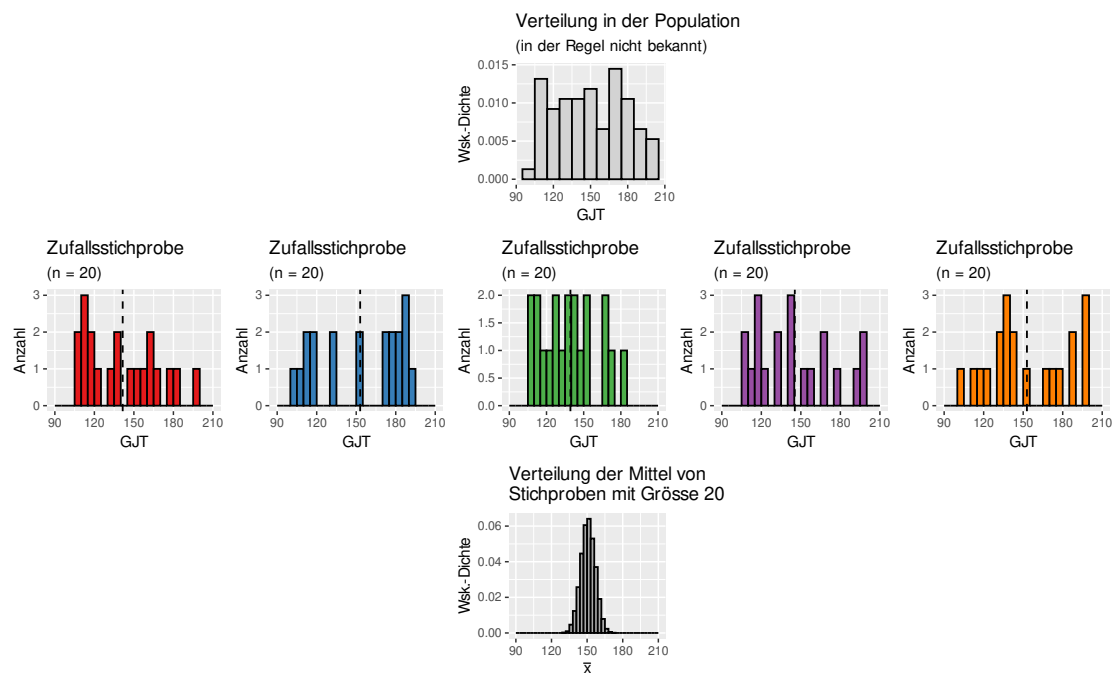


Abbildung 6.2: Wenn eine grosse Anzahl Zufallsstichproben der gleichen Grösse aus der Population gezogen werden und je ihr Mittel berechnet wird (senkrechte Linie), ergibt sich die Stichprobenmittelverteilung. In diesem Fall ist diese in etwa normalverteilt, aber dies ist nicht unbedingt der Fall. Exemplarisch werden fünf der Stichproben gezeigt.

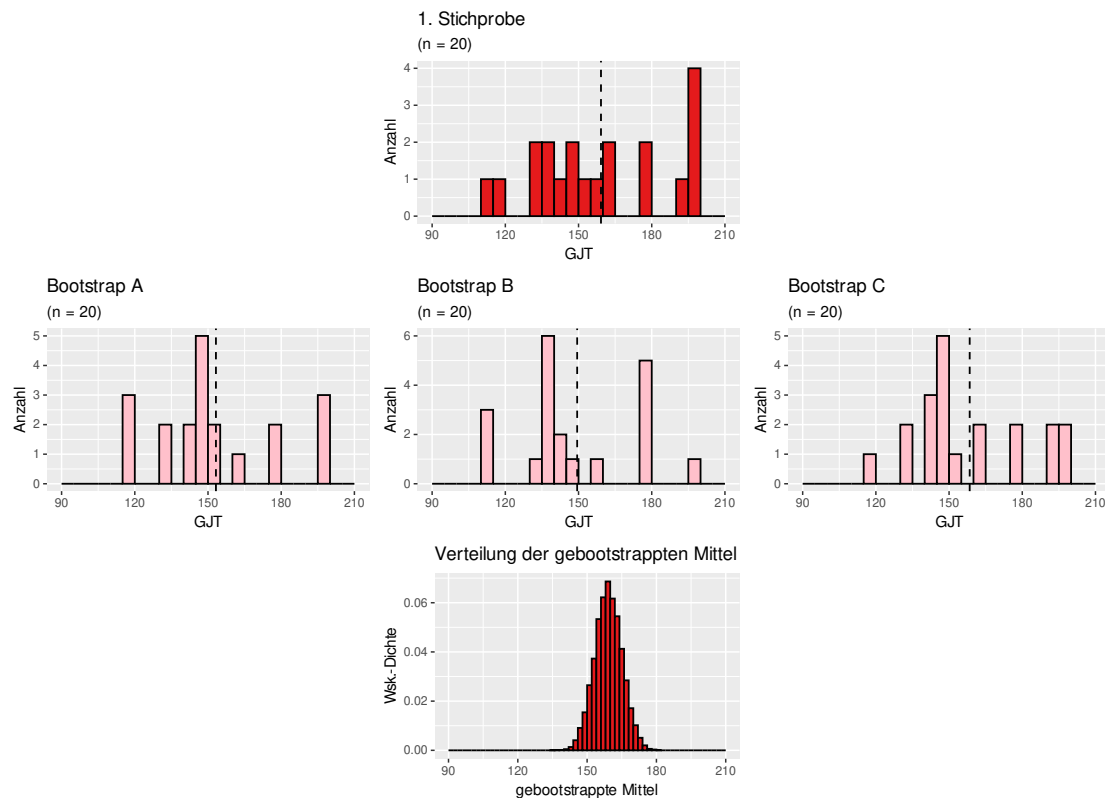


Abbildung 6.3: Die erste Stichprobe aus Abbildung 6.2 dient hier als Stellvertreter der GJT-Population. Exemplarisch werden drei Bootstrap-Stichproben mit Grösse 20 gezeigt. Wenn man 20'000 solche Bootstrap-Stichproben generiert, bilden ihre Mittel die Verteilung in der unteren Grafik. Diese hier schaut normalverteilt aus, aber dies ist nicht zwingend der Fall.

20 aus dieser Stichprobe.³ Diese Stichproben werden *Bootstrap*-Stichproben (oder *bootstrap replicates*) genannt. Abbildung 6.3 zeigt exemplarisch drei solche *Bootstrap*-Stichproben. Für jede Bootstrap-Stichprobe können wir das Mittel berechnen; die Verteilung von 20'000 dieser Mittel steht in der unteren Grafik.

Das Mittel der gebootstrappten Mittel ist gleich dem Mittel der Stichprobe (141.5). Ihre Standardabweichung beträgt etwa 6.0. 2.5% der gebootstrappten Mittel sind kleiner als 130.10; 2.5% sind grösser als 153.55; die Breite dieses Intervalls beträgt also etwa 23.45 Punkte. ◇

Beispiel 6.3 (blaue Stichprobe). Abbildung 6.4 auf der nächsten Seite zeigt das Verfahren noch einmal, diesmal mit der zweiten (blauen) Stichprobe aus Abbildung 6.2 als Ausgangspunkt. Das Mittel der gebootstrappten Mittel ist gleich dem Mittel der Stichprobe (153). Ihre Standardabweichung beträgt etwa 7.05 Punkte. 2.5% der gebootstrappten Mittel sind kleiner als 139.05;

³Ein Detail: Eine Beobachtung darf mehrmals in der gleichen Stichprobe vorkommen. Dies nennt man *sampling with replacement* und man macht es, weil man davon ausgeht, dass die Population (praktisch gesehen) unendlich gross ist.

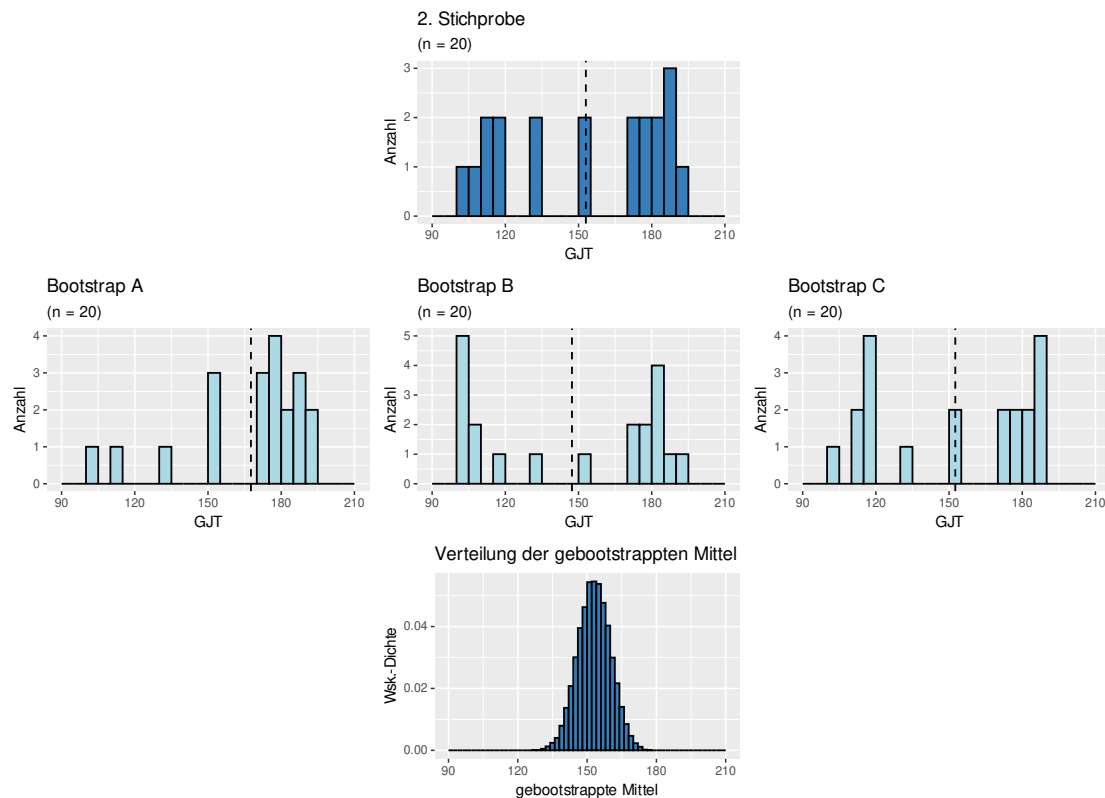


Abbildung 6.4: Die zweite Stichprobe aus Abbildung 6.2 dient hier als Stellvertreter der GJT-Population. Exemplarisch werden drei Bootstrap-Stichproben mit Grösse 20 gezeigt. Wenn man 20'000 solche Bootstrap-Stichproben generiert, bilden ihre Mittel die Verteilung in der unteren Grafik. Diese hier schaut normalverteilt aus, aber dies ist nicht zwingend der Fall.

2.5% sind grösser als 166.60; die Breite dieses Intervalls ist also 27.55 Punkte. ◇

Der Bootstrap ist eine Technik, um die Ungenauigkeit bzw. Unsicherheit von Parameterschätzungen zu quantifizieren (Efron, 1979; Efron & Tibshirani, 1993). Um die *tatsächliche* Ungenauigkeit einer Parameterschätzung zu berechnen, könnten wir eine grosse Anzahl Stichproben aus der gleichen Population ziehen und feststellen, wie die Schätzungen zwischen den Stichproben variieren:

- Population definieren,
 - Stichproben ziehen,
 - Verteilung von Schätzungen in Stichproben generieren,
 - Variabilität in Schätzung berechnen.

Mangels einer grossen Anzahl Stichproben aus der gleichen Population, verlässt man sich auf das *plug-in*-Prinzip: Die Stichprobe tritt stellvertretend für die Population auf, und geschaut wird, wie gut Stichproben einer bestimmten Grösse aus dieser Stichprobe den untersuchten

Tabelle 6.1: Standardabweichung, Perzentile und der Unterschied zwischen den Perzentilen für die eigentliche Stichprobenmittelverteilung und die fünf Verteilungen der gebootstrappten Mittel. Die Perzentile und der Unterschied zwischen ihnen stimmen wegen Rundung nicht komplett miteinander überein.

Stichprobenmittelverteilung	<i>SD</i>	2.5. Perzentil	97.5. Perzentil	Unterschied
Tatsächlich (unbekannt)	6.1	139	163	24
Bootstrap Stichprobe 1	6.0	130	154	23
Bootstrap Stichprobe 2	7.0	139	167	28
Bootstrap Stichprobe 3	4.9	130	149	19
Bootstrap Stichprobe 4	6.5	133	158	25
Bootstrap Stichprobe 5	6.6	140	166	26

Parameter schätzen können.

- Stichprobe definieren,
→ Bootstrap-Stichproben ziehen,
→ Verteilung von Schätzungen in Bootstrap-Stichproben generieren,
→ Variabilität in Schätzung *schätzen*.

Der Bootstrap ergibt eine *Schätzung* der Ungenauigkeit einer Parameterschätzung. Dies wird klar, wenn man sich Abbildung 6.5 auf der nächsten Seite und Tabelle 6.1 anschaut. Abbildung 6.5 zeigt die Verteilung der gebootstrappten Mittel für die fünf Stichproben; Tabelle 6.1 fasst ihre Standardabweichung, ihre 2.5. und 97.5. Perzentile, und den Unterschied zwischen diesen Perzentilen zusammen. Die Standardabweichungen und die Breite der Intervalle zwischen dem 2.5. und dem 97.5. Perzentil sind in keinem der fünf Beispiele dem entsprechenden tatsächlichen, aber unbekannten Wert, gleich. Aber im Schnitt sind sie ihm recht ähnlich. Insbesondere wenn man über keine weiteren Anknüpfungspunkte verfügt (z.B. vorherige Studien, sachlogische Überlegungen), kann der Bootstrap also nützliche, wenn auch imperfekte, Informationen über die Unsicherheit einer Parameterschätzung liefern.

Bemerkung 6.4 (Vorteile des Bootstraps). Die wichtigsten Vorteile des Bootstraps sind die folgenden.

- Der Bootstrap ist didaktisch wertvoll (hoffe ich). Die mathematischen Anforderungen sind beim Bootstrappen gering. Dies erlaubt uns, wichtige Konzepte unabhängig von ihrer üblichen mathematischen Umsetzung zu besprechen.
- Der Bootstrap ist flexibel. Hier haben wir uns mit der Unsicherheit eines Stichprobenmittels befasst. Diese kann man auch mit einer relativ einfachen analytischen Methode ausdrücken (siehe unten). Den Bootstrap kann man aber auch verwenden, um die Unsicherheit vieler anderer Schätzungen auszudrücken, zum Beispiel eines getrimmten oder winsorisierten Mittels, eines Medians, einer Standardabweichung, eines bestimmten Perzentils, oder irgendwelcher anderen Masse. Dies wird in den Aufgaben gemacht.

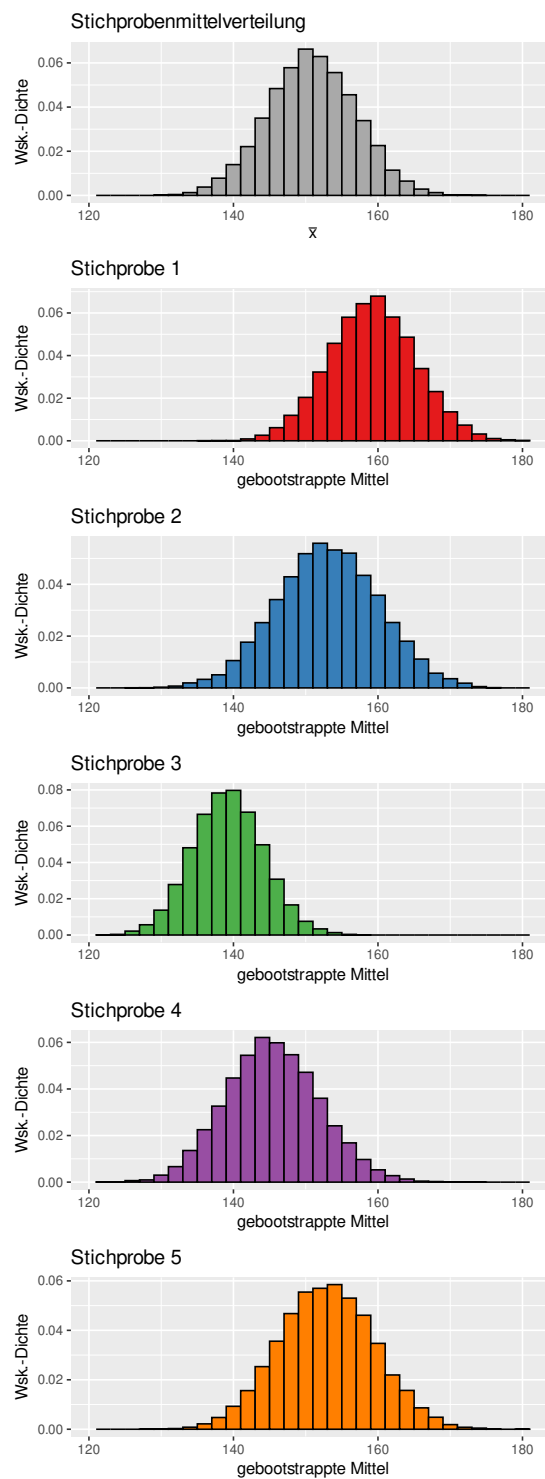


Abbildung 6.5: Die Verteilung der gebootstrappten Mittel auf der Basis von fünf Stichproben.

Ausserdem kann der Bootstrap auch bei komplexeren Modellen (z.B., wenn wir den Zusammenhang zwischen verschiedenen Variablen untersuchen) verwendet werden.

- Je nachdem sind die Annahmen des Bootstraps plausibler als die Annahmen anderer geläufiger Verfahren. Dieser Punkt wird später in diesem Kapitel deutlicher werden, aber hier sei bereits darauf hingewiesen, dass wir in den obigen Beispielen nirgends davon ausgegangen sind, dass die Stichprobenmittelverteilung normalverteilt ist. In den Beispielen sind die Verteilungen der gebootstrappten Mittel zwar annähernd normalverteilt, aber hiervon sind wir nicht *a priori* *ausgegangen*. Wir sind insbesondere nicht davon ausgegangen, dass die Population, aus der die Stichproben stammen, normalverteilt ist. ◇

Bemerkung 6.5 (Nachteile des Bootstraps). Selbstverständlich ist der Bootstrap keine perfekte Lösung.

- “Bootstrapping does not overcome the **weakness of small samples** as a basis for inference.” (Hesterberg, 2015, S. 379) Einerseits ist die *tatsächliche* Unsicherheit einer Parameterschätzung bei einer kleinen Stichprobe natürlich grösser als bei einer grösseren (siehe auch den zentralen Grenzwertsatz). Aber andererseits ist unsere *Schätzung* dieser Unsicherheit bei kleineren Stichproben auch weniger genau als bei grösseren. Dies ist aber nicht sosehr ein Nachteil des Bootstraps, sondern von kleinen Stichproben im Allgemeinen: Andere Verfahren bieten hier keine bessere Lösung.⁴
- Die Implementierung des Bootstraps, die oben illustriert wurde, tendiert dazu, die Unsicherheit einer Parameterschätzung eher zu unter- als zu überschätzen. Dies ist umso mehr der Fall bei kleinen Stichproben. Der Grund dafür ist, dass die Streuung einer Stichprobe die Streuung der Population eher unter- als überschätzt; deswegen wird die Varianz einer Stichprobe ja leicht anders berechnet als jene einer Population (siehe Abschnitt 5.2.2 auf Seite 113). Beim Bootstrap tritt die Stichprobe aber stellvertretend für die Population auf. Insofern die Stichprobe die Streuung in der Population unterschätzt, unterschätzt der Bootstrap die Unsicherheit der Parameterschätzung. Es gibt ein paar Möglichkeiten, diese Verzerrung zu korrigieren (siehe Efron & Tibshirani, 1993), aber pädagogisch sind diese hier nicht so interessant.
- Da der Bootstrap so flexibel ist, ist es schwierig, eine allgemeine, benutzerfreundliche Funktion für ihn zu schreiben. Daher muss man den Bootstrap meistens selber programmieren. ◇

Die obigen Beispiele dienten nur einem pädagogischen Zweck: Wenn wir eine Stichprobe von 76 Versuchspersonen haben, ist es ja kaum sinnvoll, kleinere Stichproben aus ihr zu ziehen. Stattdessen werden hier zuerst die Befehle gezeigt, mit denen Sie die Unsicherheit von DeKeyser et al.’s ursprünglichem Stichprobenmittel schätzen können. Übrigens befinden wir uns dabei in der komischen aber noch recht üblichen Situation, dass wir nicht wirklich wissen, über welche

⁴Ausser sie machen striktere Annahmen oder sie berücksichtigen Informationen, die man nicht aus den Daten selber ableiten kann.

Population wir genau Aussagen treffen können. Dann folgen zwei Übungen, die die Flexibilität des Bootstrap illustrieren.

Beispiel 6.6. Der erste Codeabschnitt definiert, wie viele bootstrap replicates generiert werden sollen, generiert diese dann anhand eines *for-loops* und berechnet jeweils das Mittel. In diesem Codeabschnitt wird davon ausgegangen, dass Sie den Datensatz *d* genannt haben. Wenn dies nicht der Fall ist, müssen Sie überall noch *d* durch den richtigen Objektnamen ersetzen oder eben den Datensatz umbenennen.

```
# Anzahl bootstrap replicates
n_bootstraps <- 20000
bootstraps <- vector(length = n_bootstraps)

for (i in 1:n_bootstraps) {
  # Sampling with replacement, daher 'replace = TRUE'.
  bootstrap_sample <- sample(d$GJT, replace = TRUE)

  # Mittel des bootstrap replicates berechnen und speichern.
  bootstraps[[i]] <- mean(bootstrap_sample)
}
```

Hesterberg (2015) empfiehlt, 20'000 bootstrap replicates zu generieren, sodass das Ergebnis nur minimal vom Zufallsfaktor im Bootstrap selber beeinflusst wird. Um eine grobe Idee zu erhalten, würden 1'000 replicates reichen, aber im Prinzip sollte diese Berechnung nicht sehr lange dauern. Ein schnelles Histogramm (ohne ggplot2) zeigt die Wirkung des zentralen Grenzwertsatzes.

```
hist(bootstraps)
```

Als Schätzung des Standardfehlers dient die Standardabweichung der gebootstrappten Mittel.

```
# Standardfehler des Mittels schätzen.
sd(bootstraps)

[1] 3.135237
```

Berichten würde ich die Schätzung des Mittels und die Unsicherheit über diese Schätzung als 150.8 ± 3.1 oder sogar als 151 ± 3 . 150.7763 ± 3.1352 wären aber zu viele Zahlen, über die es zu viel Unsicherheit gibt. In Vanhove (2021) habe ich versucht, ein paar Richtschnuren fürs Abrunden von Schätzungen zu formulieren.

Etwa 95% der gebootstrappten Mittel liegen zwischen 145 und 157. Dieses Intervall stellt ein **Konfidenzintervall** dar, aber darüber später mehr.

```
quantile(bootstraps, probs = c(0.025, 0.975))

      2.5%      97.5%
144.5395 156.8951
```

Da die Verteilung der gebootstrappten Mittel in etwa normalverteilt aussieht, können wir diese Perzentile auch mithilfe der Eigenschaften von Normalverteilungen berechnen. Das 2.5. Perzentil jeder Normalverteilung liegt etwa 1.96 Standardabweichungen unter dem Mittel:

```
qnorm(0.025)
[1] -1.959964
```

Und das 97.5. Perzentil liegt genauso weit über dem Mittel:

```
qnorm(0.975)
[1] 1.959964
```

Diese Berechnungsmethode ergibt daher grundsätzlich die gleiche Lösung:

```
mean(d$GJT) + c(-1.96, 1.96) * sd(bootstraps)
[1] 144.6313 156.9214
```

Dies gilt natürlich nur, wenn die Verteilung der gebootstrappten Mittel normalverteilt ist; die Perzentilmethode ist allgemeiner gültig. ◇

Aufgabe 6.7. Verglichen mit den Angaben in Tabelle 6.1 sind der geschätzte Standardfehler und die Breite des Intervalls kleiner. Wieso? ◇

Aufgabe 6.8 (getrimmtes Mittel). Das $100\alpha\%$ **getrimmte Mittel** ist eine weitere Methode, um anhand einer Stichprobe das Mittel einer Population zu schätzen. Um es zu berechnen, legen wir zunächst eine Zahl $\alpha \in (0, 0.5)$ fest. Für eine Stichprobengrösse n berechnen wir dann die Zahl $\lfloor \alpha n \rfloor$. Wir entfernen dann die $\lfloor \alpha n \rfloor$ niedrigsten und die $\lfloor \alpha n \rfloor$ höchsten Beobachtungen aus der Stichprobe und berechnen das arithmetische Mittel der restlichen Beobachtungen.

Konkret wollen wir hier das 20% getrimmte Mittel der GJT-Daten berechnen, d.h., $\alpha = 0.2$. Also

$$\lfloor \alpha n \rfloor = \lfloor 0.2 \cdot 76 \rfloor = \lfloor 15.2 \rfloor = 15.$$

Mit der `rank()`-Funktion bestimmen wir die **Ränge** der GJT-Beobachtungen: Die niedrigste GJT-Beobachtung hat Rang 1, die höchste Rang 76. Falls mehrere GJT-Beobachtungen den gleichen Rang haben, werden ihre Ränge zufällig zugeteilt (`ties.method = "random"`). Dann wird das Mittel derjenigen GJT-Beobachtungen mit einem Rang grösser als 15 und kleiner als 72 (bzw. kleiner-gleich 71) berechnet:

```
n <- length(d$GJT)
alpha <- 0.2
discard <- floor(alpha * n)
ranks <- rank(d$GJT, ties.method = "random")
d$GJT[ranks > discard & ranks <= n - discard] |> mean()
[1] 150.6739
```


Viel einfacher geht es direkt mit der `mean()`-Funktion:

```
mean(d$GJT, trim = 0.2)
[1] 150.6739
```

Das getrimmte Mittel stellt eine Zwischenlösung zwischen dem üblichen Mittel ($\alpha \approx 0$) und dem Median ($\alpha \approx 0.5$) dar und kann nützlich sein, wenn die Stichprobe möglicherweise verunreinigt wurde.

Schätzen Sie den Standardfehler des 20% getrimmten Mittels der GJT-Daten anhand des Bootstraps. Dazu brauchen Sie lediglich eine Zeile aus dem Code unter Beispiel 6.6 zu ändern. Zeichnen Sie auch ein Histogramm der Bootstrap-Schätzungen. ◇

Aufgabe 6.9 (Ungenauigkeit der Standardabweichung). Der Bootstrap ist nicht nur nützlich, um die Unsicherheit in der Schätzung eines Mittels zu quantifizieren. Berechnen Sie die Standardabweichung der GJT-Daten und verwenden Sie den Bootstrap, um die Ungenauigkeit dieser Schätzung zu quantifizieren. Zeichnen Sie auch das entsprechende Histogramm. ◇

Aufgabe 6.10 (Median). Berechnen Sie den Median der GJT-Daten und verwenden Sie den Bootstrap, um die Ungenauigkeit dieser Schätzung zu quantifizieren. Was fällt Ihnen verglichen mit den vorigen Übungen auf? Wenn Ihnen nichts auffällt, sollten Sie die Anzahl *bins* im Histogramm vergrößern: `hist(bootstraps, breaks = 100)`. Wie erklären Sie sich Ihren Befund? ◇

6.3 Das *plug-in*-Prinzip und der zentrale Grenzwertsatz

In der angewandten Linguistik wird der Bootstrap eher selten verwendet, um die Ungenauigkeit eines Stichprobenmittels zu quantifizieren. Stattdessen verlässt man sich meistens auf den zentralen Grenzwertsatz (siehe Abschnitt 4.5 auf Seite 102). Zur Erinnerung: Der zentrale Grenzwertsatz besagt, dass die Verteilung der Stichprobenmittel (\bar{X}) zu einer Normalverteilung neigt, wenn die Stichproben gross genug sind. Das Mittel der Stichprobenmittelverteilung ist gleich dem Populationsmittel ($\mu_{\bar{X}} = \mu$); ihre Standardabweichung (der Standardfehler) beträgt:

$$SE = \sigma_{\bar{X}} = \frac{\sigma}{\sqrt{n}}.$$

Wenn wir die Standardabweichung der Population kennen, so können wir diesen Standardfehler direkt berechnen. Wenn uns die Standardabweichung der Population nicht bekannt ist, können wir wieder das *plug-in*-Prinzip anwenden: Die Stichprobenstandardabweichung S ist die beste Schätzung der Populationsstandardabweichung σ , die wir haben, weshalb wir diese Schätzung stellvertretend in die Formel eintragen. So erhalten wir wiederum eine Schätzung \widehat{SE} des Standardfehlers SE :

$$SE \approx \widehat{SE} = \frac{S}{\sqrt{n}}.$$

Bei den GJT-Daten beträgt die Stichprobenstandardabweichung etwa 27.23. Daher beträgt der geschätzte Standardfehler $\frac{27.32}{\sqrt{76}} = 3.13$.

Anhand des zentralen Grenzwertsatzes können wir auch ein 95%-Konfidenzintervall konstruieren:

```
mean(d$GJT) + qnorm(c(0.025, 0.975)) * sd(d$GJT)/sqrt(76)
[1] 144.6347 156.9180
```

Dieses Konfidenzintervall ist dem Konfidenzintervall, das mit dem Bootstrap konstruiert wurde, sehr ähnlich, was natürlich beruhigend ist. Diesmal sind wir aber davon *ausgegangen*, dass die Stichprobenmittel aus der Population normalverteilt sind; diese Annahme haben wir beim Bootstrap nicht gemacht. Bei sehr schiefen oder anderen asymmetrischen Verteilungen ist es durchaus möglich, dass der zentrale Grenzwertsatz auch bei Stichproben von 76 Beobachtungen noch nicht gegriffen hat. Wenn dieser Verdacht besteht, dürfte der Bootstrap also geeigneter sein. (Man sollte sich bei solchen Verteilungen aber ohnehin einmal überlegen, ob man sich wirklich für ihr Mittel interessieren sollte; siehe *Before worrying about model assumptions, think about model relevance* (11.04.2019).) Ausserdem gilt der zentrale Grenzwertsatz nur für das Mittel, nicht für andere Parameterschätzungen. Für ein paar Parameterschätzungen gibt es andere Formeln, aber der Bootstrap ist wesentlich flexibler.

6.4 Die *t*-Verteilungen

Die Stichprobenstandardabweichung (S) ist bloss eine Schätzung der Populationsstandardabweichung (σ). Insofern S σ unterschätzt, wird die Unsicherheit in der Parameterschätzung unterschätzt; überschätzt S σ , dann wird die Unsicherheit in der Parameterschätzung überschätzt. Damit könnte man sich abfinden, wenn sich Unter- und Überschätzungen ausgleichen würden. In Abschnitt 5.2.3 wird diesbezüglich jedoch auf ein Problem hingewiesen: S tendiert dazu, σ zu unterschätzen, insbesondere bei kleinen Stichproben. Entsprechend ist die Stichprobenmittelverteilung eher breiter als schmaler als eine Normalverteilung mit S/\sqrt{n} als Standardabweichung. Meistens ist es unmöglich, diese Verzerrung in S zu beheben. Die Ausnahme ist, wenn die Stichprobe aus einer normalverteilten Population stammt.

Satz 6.11 (Gosset (Student)). Sei X_1, \dots, X_n eine Zufallsstichprobe unabhängiger Beobachtungen aus einer $\mathcal{N}(\mu, \sigma^2)$ -Verteilung. Seien \bar{X} und S das Stichprobenmittel bzw. die Stichprobenstandardabweichung dieser Stichprobe. Dann hat die Grösse

$$T := \frac{\bar{X} - \mu}{S/\sqrt{n}}$$

die gleiche Verteilung wie

$$\frac{Z_0}{\sqrt{\frac{1}{n-1} (Z_1^2 + \dots + Z_{n-1}^2)}},$$

wo $Z_0, \dots, Z_{n-1} \sim \mathcal{N}(0, 1)$ unabhängig sind. Diese Verteilung nennt man die **Student-*t***-

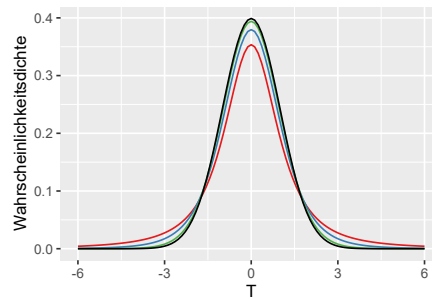


Abbildung 6.6: Wahrscheinlichkeitsdichten der Student-*t*-Verteilung mit 2 (rot), 5 (blau) und 20 (grün) Freiheitsgraden. Die schwarze Kurve ist die Wahrscheinlichkeitsdichte der Standardnormalverteilung.

Verteilung mit $n - 1$ Freiheitsgraden (Anzahl Zufallsvariablen, die im Nenner auftauchen). Man schreibt $T \sim t_{n-1}$ oder $T \sim t(n - 1)$. \diamond

Zum Vergleich: Ist σ bekannt, so gilt

$$\frac{\bar{X} - \mu}{\sigma / \sqrt{n}} \sim \mathcal{N}(0, 1)$$

oder, äquivalent damit,

$$\bar{X} \sim \mathcal{N}(\mu, \sigma^2 / n).$$

Abbildung 6.6 zeigt *t*-Verteilungen mit 2, 5 und 20 Freiheitsgraden sowie eine Standardnormalverteilung. Je mehr Freiheitsgrade, desto ähnlicher ist die Verteilung einer Standardnormalverteilung. Dies entspricht der Tatsache, dass grössere Stichproben σ tendenziell weniger unterschätzen als kleinere Stichproben. ‘Student’ war übrigens das Pseudonym von William S. Gosset.

Um das 95%-Konfidenzintervall um ein Stichprobenmittel mithilfe der *t*-Verteilungen zu finden, sucht man zuerst mit `qt()` das 2.5. und das 97.5. Perzentil der *t*-Verteilung mit $n - 1$ Freiheitsgraden (hier: $76 - 1 = 75$). Dann multipliziert man den geschätzten Standardfehler mit diesen Perzentilen. Dies ist komplett analog zur Berechnung auf der Basis des zentralen Grenzwertsatzes, nur wird mit einer *t*- statt einer Normalverteilung gearbeitet.

```
qt(0.025, df = 75)
```

```
[1] -1.992102
```

```
qt(0.975, df = 75)
```

```
[1] 1.992102
```

```
mean(d$GJT) + qt(c(0.025, 0.975), df = 75) * sd(d$GJT) / sqrt(76)
```

```
[1] 144.5340 157.0187
```

In diesem Beispiel ergeben alle Berechnungsmethoden ein recht ähnliches Ergebnis. Insbesondere bei kleinen Stichproben oder bei Stichproben, die den Verdacht nahelegen, dass die Population sehr schräg verteilt ist, ist dies aber nicht unbedingt der Fall.

Von den drei besprochenen Methoden macht die t -Methode die meisten Annahmen: Sie geht nicht nur davon aus, dass man anhand der Stichprobe sinnvolle Aussagen über die Unsicherheit machen kann und dass die Population irgendwelche Verteilung hat, für die den zentralen Grenzwertsatz bei dieser Stichprobengrösse greift, sondern auch, dass die Population *selber* normalverteilt ist. Wenn all diese Annahmen aber stimmen, ist diese Methode auch die genaueste. Der Bootstrap dahingegen ist sozusagen das Schweizer Sackmesser unter den Schätzungsmethoden: Er kann in vielen Situationen angewandt werden, aber je nach Situation gibt es spezialisierte Methoden, die schon besser funktionieren.⁵

Bemerkung 6.12 (pathologische Verteilungen). Die $t(1)$ -Verteilung wird auch die **Cauchy-Verteilung** genannt. Ihre Wahrscheinlichkeitsdichte liegt zwar symmetrisch um die y -Achse, aber sie hat trotzdem keinen Erwartungswert. Das Mittel einer grossen Anzahl unabhängiger $t(1)$ -verteilter Zufallsvariablen konvergiert entsprechend nicht gegen irgendeinen Wert. Um dies zu sehen, können Sie die folgenden Befehle mehrmals laufen lassen. Sie können auch n erhöhen.

```
n <- 1000
stichprobe <- rt(n, df = 1)
mean(stichprobe)

[1] 2.193034
```

Die $t(2)$ -Verteilung hat zwar ein Mittel, nämlich 0, aber ihre Varianz ist unendlich gross. ◇

6.5 Konfidenzintervalle

Im Laufe dieses Kapitels haben wir ein paar Konfidenzintervalle konstruiert, sodass es nun die höchste Zeit ist, zu erklären, was diese überhaupt sind. Eine leider schwierige Definition ist die folgende.

Definition 6.13 (Vertrauensschranken und Konfidenzintervall). Sei X_1, \dots, X_n eine Stichprobe aus einer Population und sei θ ein zu schätzender Parameter (Estimand). Eine Abbildung (Funktion) $a(\alpha, X_1, \dots, X_n)$ liefert eine **untere** $100(1 - \alpha)\%$ -**Vertrauensschranke**, wenn

$$\mathbb{P}(a(\alpha, X_1, \dots, X_n) \leq \theta) \geq 1 - \alpha \quad (6.1)$$

für alle $\alpha \in (0, 1)$. Eine Abbildung $b(X_1, \dots, X_n)$ liefert eine **obere** $100(1 - \alpha)\%$ -**Vertrauensschranke**, wenn

$$\mathbb{P}(\theta \leq b(X_1, \dots, X_n)) \geq 1 - \alpha.$$

⁵Tatsächlich heisst der Vorläufer des Bootstraps das *jackknife*.

für alle $\alpha \in (0, 1)$. Zwei Abbildungen $\tilde{a}(\alpha, X_1, \dots, X_n)$ und $\tilde{b}(\alpha, X_1, \dots, X_n)$ liefern ein $100(1 - \alpha)\%$ -**Vertrauens- oder Konfidenzintervall**, wenn

$$\mathbb{P}(\tilde{a}(\alpha, X_1, \dots, X_n) \leq \theta \leq \tilde{b}(\alpha, X_1, \dots, X_n)) \geq 1 - \alpha$$

für alle $\alpha \in (0, 1)$. ◇

Die Idee ist wie folgt. Wir wollen anhand einer Zufallsstichprobe einen Parameter θ schätzen. Statt lediglich einen Punktschätzer anzugeben, möchten wir einen Bereich möglicher Werte für θ angeben, die mit den beobachteten Daten kompatibel sind. Dazu brauchen wir ein Verfahren, das garantiert, dass die untere bzw. obere Vertrauensschränke zu einer Wahrscheinlichkeit von höchstens α grösser bzw. kleiner als θ ist, wobei α die Fehlerwahrscheinlichkeit ist, die wir uns erlauben. Für ein Konfidenzintervall brauchen wir ein Verfahren, das ein Intervall konstruiert, das zu einer Wahrscheinlichkeit von mindestens $1 - \alpha$ den Parameter θ enthält.

Beispiel 6.14 (Exakte Vertrauensschränken für das Mittel einer Normalverteilung mit bekannter Varianz). Sei X_1, \dots, X_n eine Zufallsstichprobe unabhängiger Beobachtungen aus der Verteilung $\mathcal{N}(\mu, \sigma^2)$, wobei σ^2 bekannt ist. Wir wissen, dass

$$\bar{X} \sim \mathcal{N}(\mu, \sigma^2/n)$$

oder, äquivalent,

$$\frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \sim \mathcal{N}(0, 1).$$

Somit gilt für das $(1 - \alpha)$ -Quantil der Standardnormalverteilung $q_{1-\alpha}$

$$\begin{aligned} 1 - \alpha &= \mathbb{P}\left(\frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq q_{1-\alpha}\right) \\ &= \mathbb{P}\left(\bar{X} - q_{1-\alpha} \frac{\sigma}{\sqrt{n}} \leq \mu\right) \\ &= \mathbb{P}\left(\bar{X} + q_{\alpha} \frac{\sigma}{\sqrt{n}} \leq \mu\right), \end{aligned}$$

denn $-q_{1-\alpha} = q_{\alpha}$. Entsprechend definieren wir

$$a(\alpha, X_1, \dots, X_n) := \bar{X} + q_{\alpha} \frac{\sigma}{\sqrt{n}},$$

wo q_{α} das α -Quantil der Standardnormalverteilung ist. Dann ist $a(\alpha, X_1, \dots, X_n)$ eine untere $100(1 - \alpha)\%$ -Vertrauensschränke für μ .

Diese Tatsache ist eine Konsequenz von dem, was wir in Abschnitt 4.5 besprochen haben, und soll nun anhand einer Simulation illustriert werden, hier mit $n = 5, \mu = -34, \sigma^2 = 9^2$ und $\alpha = 0.13$:

```
known_sigma_one_run <- function(n, mu, sigma, alpha) {
  x <- rnorm(n, mu, sigma)
  mean(x) + qnorm(alpha) * sigma / sqrt(n)
}
simulation <- replicate(
  20000, known_sigma_one_run(n = 5, mu = -34, sigma = 9, alpha = 0.13))
mean(simulation <= -34)

[1] 0.87125
```

Für 87% der simulierten Stichproben liegen die untere 87%-Vertrauensschranke also unter dem zu schätzenden Parameter μ . Tatsächlich gilt für dieses Verfahren, dass die Ungleichheit in (6.1) mit Gleichheit gilt.

Generieren wir nun eine konkrete Stichprobe aus $\mathcal{N}(12, 17^2)$ und konstruieren wir eine konkrete untere 80%-Vertrauensschranke für μ :

```
mu <- 12
sigma <- 17
n <- 7
alpha <- 0.2
x <- rnorm(n, mu, sigma)
x # Stichprobe

[1] 28.693480 19.968156 10.164487 8.381071 31.687674
[6] 33.970032 21.089001

mean(x) + qnorm(alpha) * sigma / sqrt(n)

[1] 16.58566
```

A priori beträgt die Wahrscheinlichkeit, dass eine Zufallsstichprobe eine ‘richtige’ untere $100(1 - \alpha)\%$ -Vertrauensschranke liefert (also eine Schranke, die niedriger als θ ist), mindestens $1 - \alpha$. Aber es kann natürlich, wie in diesem Fall, vorkommen, dass die untere Vertrauensschranke grösser als θ ist. Dieses Beispiel zeigt auch, dass sich die Wahrscheinlichkeitsgarantien auf das Konstruktionsverfahren beziehen und nicht auf individuelle Schranken: Es wäre Quatsch, zu sagen, dass

$$\mathbb{P}(16.58 \leq \mu) \geq 0.80,$$

denn im Ausdruck ‘ $16.58 \leq \mu$ ’ kommt keine Zufälligkeit vor.⁶

Eine obere $100(1 - \alpha)\%$ -Vertrauensschranke ist gegeben durch

$$b(\alpha, X_1, \dots, X_n) := \bar{X} + q_{1-\alpha} \frac{\sigma}{\sqrt{n}}.$$

⁶Ein anderes Beispiel: Bevor wir mit einem sechsseitigen Würfel würfeln, beträgt die Wahrscheinlichkeit, dass wir mindestens eine 4 würfeln, 50%. Wenn wir aber eine 2 beobachten, so ergibt es nicht länger Sinn, zu behaupten, dass zu einer Wahrscheinlichkeit von 50% 2 mindestens gleich gross 4 ist.

Ein $100(1 - \alpha)\%$ -Konfidenzintervall ist nun gegeben durch

$$[a(\alpha/2, X_1, \dots, X_n), b(\alpha/2, X_2, \dots, X_n)].$$

Das heisst, wir verwenden zwei $100(1 - \alpha/2)\%$ -Vertrauensschranken. Andere Konstruktionsverfahren sind möglich; diese ergeben dann andere konkrete Werte, bieten aber dieselben Wahrscheinlichkeitsgarantien. Die hier beschriebene Methode ist die geläufigste. \diamond

Bemerkung 6.15 (Gütekriterien für Konfidenzintervalle). Es gibt nicht für jedes Schätzproblem *das* Verfahren, um Konfidenzintervalle zu konstruieren. Das wichtigste Gütekriterium bei der Wahl eines Verfahrens ist, dass die Wahrscheinlichkeitsgarantien (wie in 6.1) respektiert werden. Werden diese Garantien perfekt eingehalten, so spricht man von **exakten** Vertrauensschranken. Oft werden sie jedoch nur annäherungsweise eingehalten. Die Methode aus Beispiel 6.14 ist exakt; die Bootstrap-Intervallen weiter oben sind approximativ.

Wenn zwei Verfahren dieses Kriterium gleichermaßen erfüllen, so bevorzugt man in der Regel das Verfahren, das tendenziell die schmalsten Intervalle liefert, oder wo die Intervallbreite für $n \rightarrow \infty$ gegen 0 strebt. Tatsächlich ist es möglich, sich Verfahren einfallen zu lassen, die zwar die nötigen Wahrscheinlichkeitsgarantien bieten, aber die entweder komplett triviale Vertrauensschranken liefern oder wo sich das Intervall für $n \rightarrow \infty$ nicht verengt. Mit diesen wollen wir uns hier aber nicht auseinandersetzen.

Ein weiteres relevantes Gütekriterium für Konfidenzschranken und -intervalle ist ihre **Robustheit**: Wenn wir nur unter recht spezifischen Annahmen an die Daten nachweisen können, dass ein Verfahren die Wahrscheinlichkeitsgarantien exakt einhält, so ist es aus praktischen Gründen erwünscht, dass dieses Verfahren die Wahrscheinlichkeitsgarantien unter weniger spezifischen Annahmen immerhin in etwa einhält. Siehe hierzu Aufgabe 6.21. \diamond

Bemerkung 6.16 (Wahl der Schranken bzw. des Intervalls). Eine untere Vertrauensschranke bietet sich an, wenn sich dafür interessiert, ob θ mindestens (und mit gewisser Fehlerwahrscheinlichkeit) einen bestimmten Wert hat. Eine obere Vertrauensschranke bietet sich an, wenn man sich dafür interessiert, ob θ höchstens einen bestimmten Wert hat. Ein Konfidenzintervall bietet sich an, wenn man beide Fragen gleichzeitig beantworten möchte. Für die Fragen, die uns interessieren, sind meistens Konfidenzintervalle angebracht.

Meistens setzt man defaultmässig $\alpha = 0.05$, d.h., man berechnet 95%-Vertrauensschranken oder 95%-Konfidenzintervalle. Dies ist jedoch bloss eine Konvention. \diamond

Die Konzepte der Vertrauensschranken und Konfidenzintervalle sind schwieriger als was man auf den ersten Blick denken würde – auch für erfahrene Forschende (Hoekstra et al., 2014). Oft interpretiert man ein 95%-Konfidenzintervall als jene zwei Werte, zwischen denen der Populationsparameter (hier: μ) mit 95% Wahrscheinlichkeit liegt. Dies stimmt aber nicht, wie das Beispiel zeigt (siehe auch Morey et al., 2016). Nichtsdestoweniger schreibt Ehrenberg (1982) zur Interpretation von Konfidenzintervallen Folgendes:

“[This] rough-and-ready interpretation of confidence limits ... will be close to the

truth. The choice is between making a statement which is true but so complex that it is almost unactionable, and making one which is much simpler but not quite correct. Fortunately, the effective content of the two kinds of statement is generally similar.” (S. 125)

Statt Konfidenzintervallen empfehlen Morey et al. (2016) den Gebrauch von ‘Kredibilitätsintervallen’. Diese sind in der bayesschen Statistik angesiedelt und kommen momentan in unserer Forschungsliteratur kaum vor, weshalb ich sie hier nicht bespreche. Albers et al. (2018) bemerken, dass Konfidenz- und Kredibilitätsintervalle einander üblicherweise sehr ähnlich sind; Nalborczyk et al. (2019) ziehen diese Schlussfolgerung aber in Frage.

Dieser Bemerkung zum Trotz sind meines Erachtens insbesondere die folgenden Punkte wichtig:

- Konfidenzintervalle heben hervor, dass Schätzungen inhärent unsicher sind.
- Bei grossen Stichproben oder bei Stichproben aus Populationen, in denen es wenig Variation gibt, sind Konfidenzintervalle tendenziell schmaler.
- Rein durch Zufall kann eine Stichprobe die Streuung in der Population unterschätzen und daher kann das Konfidenzintervall die Unsicherheit in der Schätzung ebenso unterschätzen.
- Genauere Unsicherheitseinschätzungen erhält man mit grösseren Stichproben oder indem man weitere nützliche Annahmen über die Daten macht (wie in der bayesschen Statistik).

Im Folgenden beschäftigen wir uns noch mit der Konstruktion von Vertrauensintervallen für drei realistischere Szenarien.

Beispiel 6.17 (Exakte Vertrauensschranken für das Mittel einer Normalverteilung mit unbekannter Varianz). Haben wir eine Zufallsstichprobe mit unabhängigen Beobachtungen $X_1, \dots, X_n \sim \mathcal{N}(\mu, \sigma^2)$, wo sowohl μ als auch σ^2 unbekannt sind, so liefert die Methode aus Beispiel 6.14 nicht länger exakte Vertrauensschranken. Mit Satz 6.11 können wir aber ausnutzen, dass

$$\frac{\bar{X} - \mu}{S/\sqrt{n}} \sim t(n-1).$$

Eine analoge Herleitung wie in Beispiel 6.14 liefert eine untere Vertrauensschranke von

$$a(\alpha, X_1, \dots, X_n) := \bar{X} + q_{n-1, \alpha} \frac{S}{\sqrt{n}},$$

wo $q_{n-1, \alpha}$ das α -Quantil der $t(n-1)$ -Verteilung ist. Obere Vertrauensschranken und Konfidenzintervalle konstruiert man analog.

Mit der Funktion `t.test()` können Sie diese Konfidenzschranken schnell berechnen. Den restlichen Output dieser Funktion behandeln wir erst später:

```
# 80%-Konfidenzintervall fürs Mittel der GJT-Daten
t.test(d$GJT, conf.level = 0.80)$conf.int
```



```
[1] 146.7248 154.8278
attr(,"conf.level")
[1] 0.8

# untere 95%-Vertrauensschranke für die gleichen Daten
t.test(d$GJT, conf.level = 0.95, alternative = "greater")$conf.int

[1] 145.5576      Inf
attr(,"conf.level")
[1] 0.95
```

Für obere Vertrauensschranken verwende man `alternative = "less"`. ◇

Bemerkung 6.18. Identisch aufgegleiste Zufallsexperimente können recht unterschiedliche Konfidenzintervalle ergeben. Mit dem unten stehenden Code können Sie dies selber aufzeigen. Es wird `n_sim` Mal das gleiche Zufallsexperiment ausgeführt: Jeweils `n_obs` unabhängige Beobachtungen werden aus einer $\mathcal{N}(\text{mittel}, \text{stabw}^2)$ -Verteilung generiert. Auf der Basis dieser Stichprobe wird mit `t.test()` ein 95%-Konfidenzintervall konstruiert. Die Konfidenzintervalle werden, nach ihrer Breite sortiert, in Abbildung 6.7 dargestellt. In jeweils etwa 5% der gezeichneten Intervalle ist der Wert von `mittel` nicht enthalten und auch die Breite der Intervalle variiert zwischen den Stichproben.

```
n_sim <- 100
n_obs <- 23
stabw <- 2.5
mittel <- -7

cis <- matrix(nrow = n_sim, ncol = 2)

for (i in 1:n_sim) {
  x <- rnorm(n_obs, mittel, stabw)
  cis[i, ] <- t.test(x)$conf.int
}

im_intervall <- cis[, 1] <= mittel & mittel <= cis[, 2]

resultate <- tibble(min = cis[, 1],
                    max = cis[, 2],
                    im_intervall,
                    sim = 1:n_sim) |>
  mutate(breite = max - min)

ggplot(resultate,
       aes(xmin = min, xmax = max,
```

```

    y = reorder(sim, breite),
    colour = im_intervall)) +
geom_errorbarh() +
geom_vline(xintercept = mittel,
           linetype = "dashed") +
scale_y_discrete(breaks = NULL) +
scale_colour_manual("Mittel im Intervall?",
                    limits = c(FALSE, TRUE),
                    labels = c("nein", "ja"),
                    values = c("red", "black")) +
theme(legend.position = "bottom") +
labs(y = "Simulation",
     title = "100 95%-Konfidenzintervalle fürs gleiche Mittel")

```

◇

Aufgabe 6.19. Angenommen, zwei Stichproben haben identische Stichprobenstandardabweichungen: $S_1 = S_2$. Stichprobe 1 bestehe aus 16 Datenpunkten; Stichprobe 2 aus nur vier. Aus welchen *zwei* Gründen wird das 95%-Konfidenzintervall bei Stichprobe 1 schmäler sein als bei Stichprobe 2, wenn Sie diese Intervalle mit t -Verteilungen konstruieren? ◇

Beispiel 6.20 (Approximative Vertrauensschranken fürs Mittel). Wenn die Zufallsstichprobe nicht aus einer Normalverteilung stammt, so kann man die Vertrauensschranken und Konfidenzintervalle aus Beispiel 6.17 immer noch als Annäherungen einsetzen. Eine bereits besprochene Alternative ist, dass man Bootstrap-Intervalle verwendet. ◇

Aufgabe 6.21 (Robustheit der t -Methode). Wir wollen untersuchen, wie gut die Vertrauensintervalle, welche die Methode aus Beispiel 6.17 liefert, sind, wenn die Beobachtungen nicht aus einer Normalverteilung, sondern aus einer Gleichverteilung stammen. Dazu definieren wir zunächst eine Funktion, die eine Stichprobe generiert und anhand dieser ein Konfidenzintervall konstruiert. Sie gibt aus, ob das Mittel der Gleichverteilung im Konfidenzintervall liegt.

```

uniform_one_run <- function(n, min, max, conf_level) {
  x <- runif(n, min, max)
  ci <- t.test(x, conf.level = conf_level)$conf.int
  mittel <- (min + max)/2
  ci[1] <= mittel & mittel <= ci[2]
}

```

Diese Funktion führen wir 20'000 Mal mit bestimmten Parameterwerten aus. Anschliessend berechnen wir die Proportion von Stichproben, für die das eigentliche Mittel im Konfidenzintervall liegt:

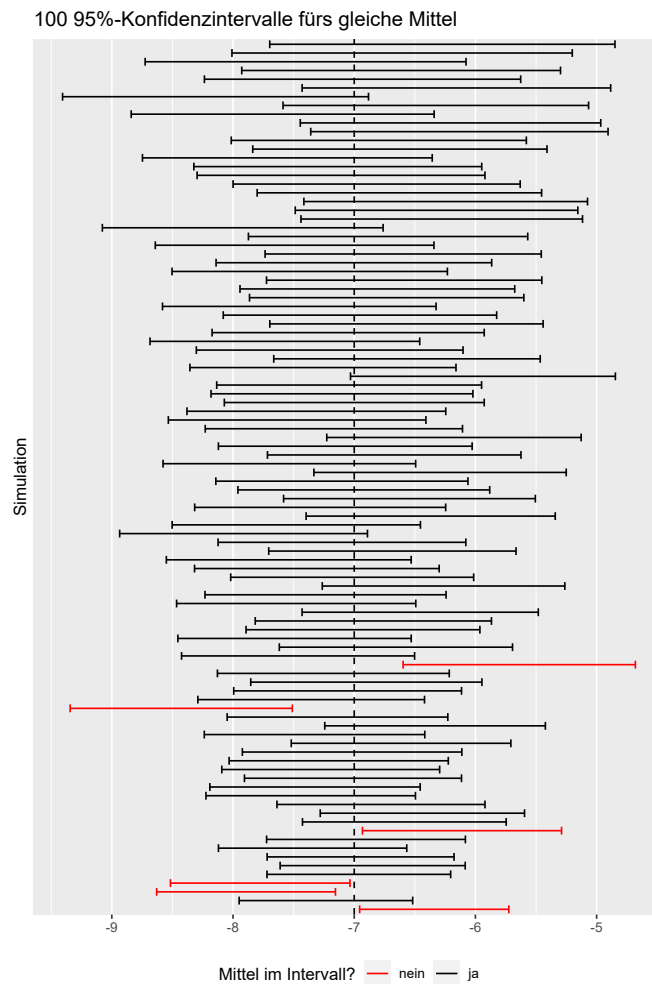


Abbildung 6.7: Konfidenzintervalle basiert auf 100 Stichproben von je 23 Beobachtungen aus einer $\mathcal{N}(-7, 2.5^2)$ -Verteilung. Die Konfidenzintervalle sind ihrer Breite nach sortiert (kürzer unten).

```
simulation <- replicate(20000, uniform_one_run(9, -12, 2, 0.7))
mean(simulation)

[1] 0.70735
```

In diesem Fall enthalten also etwa 70.5% der 70%-Konfidenzintervalle den tatsächlichen Parameterwert.

1. Wiederholen Sie diese Simulation mit kleineren und grösseren Stichproben. Variieren Sie auch den Wert von `conf_level`. Was schlussfolgern Sie?
2. Gleichverteilungen sind, wie auch Normalverteilungen, symmetrisch um ihr Mittel. χ^2 -Verteilungen sind dahingegen rechtsschief: Es gibt mehr Wahrscheinlichkeitsmasse unter als über dem Mittel, vor allem für geringe Freiheitsgrade. Wir wollen nun überprüfen, ob die t -Methode auch in etwa für solche rechtsschiefen Verteilungen funktioniert. Passen Sie den obigen Code so an, dass die Stichproben aus einer χ^2 -Verteilung mit `df` Freiheitsgraden stammen (`rchisq(n, df)`); auch anderswo sollte der Code sinngemäss angepasst werden. Spielen Sie mit den Werten für `n`, `df` und `conf_level` herum und ziehen Sie ein Fazit.

Hinweis: Das Mittel einer χ^2 -Verteilung ist gleich ihrer Anzahl Freiheitsgrade. \diamond

Beispiel 6.22 (Exakte Vertrauensschranken für Binomialparameter). Wir runden dieses Kapitel mit einem klassischen Beispiel ab. Sei $X \sim \text{Binomial}(n, p)$. Ein erwartungstreuer Schätzer von p ist

$$\hat{p} := \frac{X}{n}.$$

Wir wollen nun untere und obere exakte Vertrauensschranken für p konstruieren. Dazu betrachten wir die Verteilungsfunktion $F_{n,p}$ der $\text{Binomial}(n, p)$ -Verteilung. Ein nützliches Resultat, das wir nicht beweisen werden, lautet, dass für alle $\alpha \in (0, 1)$ gilt, dass

$$\mathbb{P}(F_{n,p}(X) \leq \alpha) = 1 - \mathbb{P}(F_{n,p}(X) > \alpha) \leq \alpha.$$

(Dies gilt im Übrigen nicht nur für Binomialverteilungen, sondern für alle Variablen X mit Verteilungsfunktion F .) Folglich

$$\mathbb{P}(F_{n,p}(X) > \alpha) \geq 1 - \alpha. \quad (*)$$

Wir fassen den Ausdruck $F_{n,p}(r)$ diesmal jedoch nicht auf als eine Funktion von r , sondern als eine Funktion von p . Für r setzen wir die beobachtete Anzahl Erfolge X ein. Die Funktion, die wir so erhalten, ist streng monoton fallend in p : Für eine feste Anzahl Erfolge ist die Wahrscheinlichkeit, dass eine binomialverteilte Zufallsvariable höchstens so viele Erfolge generiert, grösser für kleine p als für grosse p . Als obere $100(1 - \alpha)\%$ -Vertrauensschranke wählen wir nun

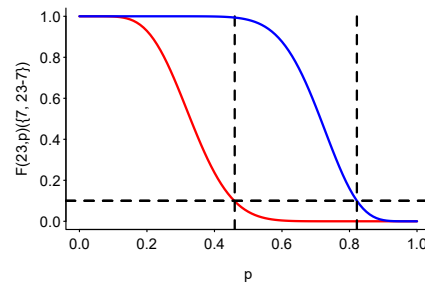


Abbildung 6.8: Konstruktion eines 80%-Konfidenzintervalls für den Binomialparameter p bei $n = 23, X = 7$. Die rote Kurve zeigt die Grösse $F_{23,p}(7)$ in Abhängigkeit von p ; die blaue Kurve zeigt die Grösse $F_{23,p}(23 - 7)$, ebenso in Abhängigkeit von p . Die Strichellinien heben die obere 90%-Vertrauensschranken für p (rot) und $1 - p$ hervor. Die untere Vertrauensschranke für p kann aus der oberen Vertrauensschranke für $1 - p$ berechnet werden.

das kleinstmögliche b mit $F_{n,b}(X) > \alpha$. Dann gilt nämlich

$$\begin{aligned} \mathbb{P}(p \leq b) &= \mathbb{P}(F_{n,p}(X) \geq F_{n,b}(X)) && [\text{streng monoton fallend}] \\ &\geq \mathbb{P}(F_{n,p}(X) > \alpha) && [\text{Auswahl von } b] \\ &\geq 1 - \alpha. && [(*)] \end{aligned}$$

Nehmen wir als konkretes Beispiel ein Experiment mit 23 Versuchen und 7 Erfolgen. Dann $\hat{p} \approx 0.304$. Die rote Kurve in Abbildung 6.8 zeigt, wie $F_{23,p}(7)$ mit p variiert. Die obere 90%-Vertrauensschranke für p (also mit $\alpha = 0.1$) liegt bei etwa 0.46.

Für die untere Vertrauensschranke betrachten wir die Erfolge als Misserfolge und umgekehrt. Das heisst, statt X betrachten wir die Zufallsvariable $Y := n - X$. Diese ist $\text{Binomial}(n, 1 - p)$ -verteilt. Wir berechnen mit der gleichen Methode die obere Vertrauensschranke \tilde{b} für $1 - p$. Dann erhalten wir eine untere Vertrauensschranke $a := 1 - \tilde{b}$ für p . Die blaue Kurve in Abbildung 6.8 zeigt, wie $F_{23,p}(23 - 7)$ mit p variiert. Die obere 90%-Vertrauensschranke für $1 - p$ liegt bei etwa 0.82. Entsprechend liegt die untere 90%-Vertrauensschranke für p bei etwa 0.18. Ein 80%-Konfidenzintervall für p ist also $[0.18, 0.46]$.

Mit `binom.test()` geht es natürlich einfacher:

```
binom.test(7, 23, conf.level = 0.80)$conf.int
[1] 0.1781578 0.4585561
attr(,"conf.level")
[1] 0.8
```



Teil III

Das allgemeine lineare Modell

Kapitel 7

Ein anderer Blick aufs Mittel

In diesem und den folgenden Kapiteln behandeln wir das sog. **allgemeine lineare Modell** (*general linear model*).¹ Das allgemeine lineare Modell ist eine Methode, um zu ausdrücken, wie ein oder mehrere **Prädiktoren** mit dem *outcome* zusammenhängen. Oft redet man statt von Prädiktoren und outcome von unabhängigen bzw. abhängigen Variablen, aber ich finde die Begriffe Prädiktor und outcome deutlicher.

In diesem Kapitel werden einige Schlüsselkonzepte des allgemeinen linearen Modells erläutert, indem wir das Mittel einer Population auf eine andere Art und Weise schätzen als wir es bisher gemacht haben. In den darauf folgenden Kapiteln werden die Modelle graduell komplexer, aber die Basisprinzipien aus diesem Kapitel werden noch immer zutreffen.

7.1 Ein Modell für die GJT-Daten

Lasst uns kurz alles über Mittelwerte vergessen. Wir erhalten Daten (hier: die GJT-Werte von DeKeyser et al. (2010), siehe Kapitel 6) und müssen diese sinnvoll beschreiben. Sinnvoller als einfach alle Datenpunkte aufzulisten, wäre, die Datenpunkte in zwei Teile zu zerlegen: einen systematischen Teil, der die Gemeinsamkeiten zwischen allen Werten ausdrückt, und einen unsystematischen Teil, der die individuellen Unterschiede zwischen diesen Gemeinsamkeiten und den Werten ausdrückt:

$$\text{Wert einer Beobachtung} = \text{Gemeinsamkeit} + \text{Abweichung}.$$

Um die Notation übersichtlich zu halten, wird diese Gleichung meistens so geschrieben:

$$y_i = \beta_0 + \varepsilon_i, \tag{7.1}$$

für $i = 1, \dots, n$. Hier ist y_i die i . Beobachtung im Datensatz, β_0 stellt die Gemeinsamkeit zwischen allen Werten in der Population dar und ε_i drückt aus, wie stark die i . Beobachtung

¹Nicht zu verwechseln mit dem **verallgemeinerten linearen Modell** (*generalized linear model*). Dieses ist eine Erweiterung des allgemeinen linearen Modells, mit der wir uns in diesem Kurs nur kurz befassen; siehe Kapitel ??.

von diesem Populationswert abweicht. ε_i nennt man auch die **Residuen** oder den **Restfehler**. Wir schreiben β_0 statt einfach β , weil wir nachher die Gemeinsamkeit zwischen den y -Werten mithilfe mehrerer β s ausdrücken werden.

In der Regel interessieren wir uns mehr für die β s als für die ε s. Da uns aber nicht die ganze Population zur Verfügung steht, müssen wir uns mit einer Schätzung von β_0 begnügen. In Gleichung 7.1 ist β_0 ein Parameter mit einem bestimmten, aber in der Regel unbekannten Wert; für Schätzungen dieses Parameters wird die Notation $\widehat{\beta}_0$ verwendet. Da $\widehat{\beta}_0$ bloss eine Schätzung ist, wird der Restfehler ebenfalls bloss geschätzt:

$$y_i = \widehat{\beta}_0 + \widehat{\varepsilon}_i,$$

für $i = 1, \dots, n$.

Wie können wir nun $\widehat{\beta}_0$ berechnen (bzw. β_0 schätzen)? Im Prinzip geht die Gleichung für jeden β_0 -Wert auf, denn wir können uns die $\widehat{\varepsilon}$ -Werte eben so aussuchen, wie es uns passt. Die ersten beiden GJT-Werte im Datensatz sind 151 und 182. Wenn wir nun beispielsweise für β_0 einen beliebigen Wert, z.B. 1823, wählen, wählen wir $\widehat{\varepsilon}_1 = -1672$ und $\widehat{\varepsilon}_2 = -1641$ und die Gleichung geht auf:

$$y_1 = 151 = 1823 - 1672,$$

$$y_2 = 182 = 1823 - 1641.$$

Wenn wir nun für β_0 den Wert 14 wählen, wählen wir $\widehat{\varepsilon}_1 = 137$ und $\widehat{\varepsilon}_2 = 168$ und die Gleichung geht wiederum auf:

$$y_1 = 151 = 14 + 137,$$

$$y_2 = 182 = 14 + 168.$$

Wir brauchen also eine prinzipielle Methode, um β_0 zu schätzen.

7.2 Die Methode der kleinsten Quadrate

Die Frage stellt sich, was die optimale Art und Weise ist, um $\widehat{\beta}_0$ zu bestimmen. Die wenig überraschende Antwort lautet: Es hängt davon ab, was man unter 'optimal' versteht. Eine sinnvolle Definition von 'optimal' ist, dass β_0 so geschätzt werden soll, dass die Summe der absoluten Restfehler ($\sum_{i=1}^n |\widehat{\varepsilon}_i|$) möglichst klein ist. Wenn wir für $\widehat{\beta}_0$ den Wert 135 wählen, beträgt die Summe der absoluten Restfehler 1993. Hier gehen wir davon aus, dass der Datensatz `dekeyser2010.csv` als den Objektnamen `d` hat.

```
sum(abs(d$GJT - 135))
```

```
[1] 1993
```

(Es gilt $y_i = \widehat{\beta}_0 + \widehat{\varepsilon}_i$, also $\widehat{\varepsilon}_i = y_i - \widehat{\beta}_0$.)

Wenn wir stattdessen den Wert 148 wählen, beträgt die Summe der absoluten Restfehler 1799.

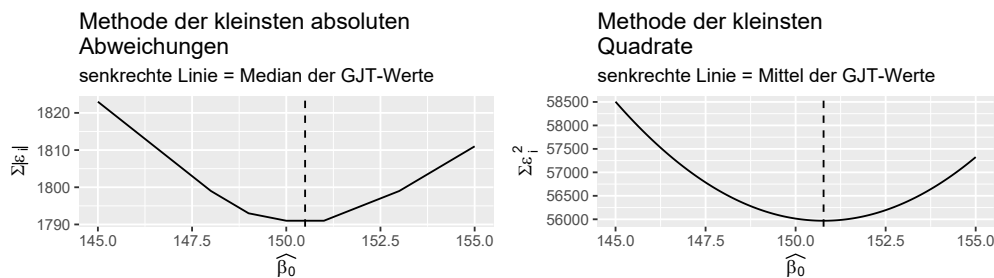


Abbildung 7.1: Links: Wenn der Parameter mit der Methode der kleinsten absoluten Abweichungen geschätzt wird, ist die Lösung gleich dem Median der Stichprobe. Rechts: Wenn der Parameter mit der Methode der kleinsten Quadrate geschätzt wird, ist die Lösung gleich dem Mittel der Stichprobe.

```
sum(abs(d$GJT - 148))
[1] 1799
```

Wenn wir 'optimal' so definieren, ist 148 also die bessere Schätzung von β_0 . Diese Übung können wir für jede Menge Kandidatwerte durchprobieren und dann den optimalen Wert wählen. Dies ist die **Methode der kleinsten absoluten Abweichungen**. Wie Abbildung 7.1 (links) zeigt, sind β_0 -Schätzungen zwischen 150 und 151 in diesem Sinne optimal. Nicht zufällig sind alle Werte im Intervall $[150, 151]$ Mediane der GJT-Werte; der Median ist entsprechend 150.5. Tatsächlich: Wenn man β_0 mit der Methode der kleinsten absoluten Abweichungen schätzt, ist das Ergebnis der Median der Stichprobe. Dass wir es hier mit einer Schätzung zu tun haben, wird klar, wenn man sich überlegt, dass diese Methode ein anderes Ergebnis liefern könnte, wenn man eine neue Stichprobe aus der gleichen Population zieht.

Eine andere sinnvolle Definition von 'optimal' ist, dass β_0 so geschätzt werden soll, dass die Summe der quadrierten Restfehler ($\sum_{i=1}^n \hat{e}_i^2$) möglichst klein ist. Dies ist die **Methode der kleinsten Quadrate**. Verglichen mit der Methode der kleinsten absoluten Abweichungen fallen grosse Residuen noch mehr ins Gewicht. Anders gesagt: Grosse Abweichungen (darunter auch Ausreisser) üben einen stärkeren Einfluss auf das Ergebnis aus. Wie Abbildung 7.1 (rechts) zeigt, ist die optimal β_0 -Schätzung laut der Methode der kleinsten Quadrate 150.78—nicht zufällig das Mittel der Stichprobe!

Während wir in Kapitel 3 die Summe der Quadrate als eine Funktion des Mittels betrachtet haben, kann man die Rollen auch umkehren: Das Mittel ist jener Wert, der die Summe der Quadrate minimiert! Ebenso ist der Median jener Wert, der die Summe der absoluten Abweichungen minimiert. Der Modus ist übrigens der Wert, der die Summe der binären Abweichungen minimiert. Sind y_i und $\hat{\beta}_0$ einander gleich, beträgt die binäre Abweichung 0, sonst 1.

Rechnerisch ist die Methode der kleinsten Quadrate am einfachsten, und die Parameter in allgemeinen linearen Modellen werden daher meistens mit dieser Methode geschätzt (*ordinary*

least squares, OLS). Aber dies ist keine Notwendigkeit. In bestimmten Bereichen trifft man ab und zu andere Optimierungskriterien an. (Eventuell werden in den Ausblickskapiteln einige dieser weiteren Kriterien angesprochen.)

7.3 Lineare Modelle in R

Mit der `lm()`-Funktion können lineare Modelle aufgebaut werden. Ihre Parameter werden anhand der Methode der kleinsten Quadrate geschätzt. Innerhalb der Funktion braucht es eine Formel mit dem outcome vor und den Prädiktoren nach der Tilde. In diesem Fall gibt es keinen Prädiktor, stattdessen wird 1 verwendet.

```
mod.lm <- lm(GJT ~ 1, data = d)
```

Die geschätzten β s kann man abrufen, indem man den Namen des Modells (hier: `mod.lm`) eintippt.

```
mod.lm

Call:
lm(formula = GJT ~ 1, data = d)

Coefficients:
(Intercept)
      150.8
```

Auch mit `coef()` erhält man die β -Schätzungen (hier nur β_0).

```
coef(mod.lm)

(Intercept)
      150.7763
```

Dass mal 150.8 und mal 150.7763 angezeigt wird, liegt lediglich daran, dass das der Output der letzten zwei Befehle strenger bzw. lockerer gerundet wird.

Mit `predict()` erhält man einen Vektor mit n Werten (hier: $n = 76$), der die ‘vorhergesagten’ y -Werte enthält. (Ich mag den Begriff ‘vorhergesagt’ hier nicht.) Es handelt sich um die y -Werte abzüglich der $\hat{\varepsilon}$ -Werte: $\hat{y}_i = y_i - \hat{\varepsilon}_i$.

```
predict(mod.lm)

      1      2      3      4      5      6
150.7763 150.7763 150.7763 150.7763 150.7763 150.7763
      7      8      9     10     11     12
150.7763 150.7763 150.7763 150.7763 150.7763 150.7763
     13     14     15     16     17     18
150.7763 150.7763 150.7763 150.7763 150.7763 150.7763
```

19	20	21	22	23	24
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
25	26	27	28	29	30
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
31	32	33	34	35	36
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
37	38	39	40	41	42
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
43	44	45	46	47	48
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
49	50	51	52	53	54
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
55	56	57	58	59	60
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
61	62	63	64	65	66
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
67	68	69	70	71	72
150.7763	150.7763	150.7763	150.7763	150.7763	150.7763
73	74	75	76		
150.7763	150.7763	150.7763	150.7763		

In unserem Fall sind dies lediglich 76 Wiederholungen von $\widehat{\beta}_0$. Der Grund ist dieser:

$$y_i = \widehat{\beta}_0 + \widehat{\varepsilon}_i.$$

Also

$$\widehat{\varepsilon}_i = y_i - \widehat{\beta}_0,$$

also

$$\widehat{y}_i = y_i - \widehat{\varepsilon}_i = y_i - (y_i - \widehat{\beta}_0) = \widehat{\beta}_0.$$

Die Residuen kann man mit der `resid()`-Funktion abfragen.

```
# Output hier nicht gezeigt
resid(mod.lm)
```

7.4 Unsicherheit in einem allgemeinen linearen Modell quantifizieren

7.4.1 Der Bootstrap

Genauso wie im letzten Kapitel können wir den Bootstrap verwenden, um die Unsicherheit in der Schätzung von β_0 zu quantifizieren. Da in diesem Fall $\widehat{\beta}_0$ gleich dem Stichprobenmittel ist, ergibt dies natürlich die gleiche Lösung wie vorher. Aber es gibt mir die Gelegenheit, zu zeigen, wie man auch für komplexere lineare Modelle den Bootstrap verwenden kann.

Die Logik ist wiederum, dass wir die Stichprobe stellvertretend für die Population einsetzen. Aber anstatt Bootstrap-Stichproben aus der Stichprobe zu ziehen, ziehen wir diesmal Bootstrap-Stichproben aus den Residuen (ϵ). Diese kombinieren wir dann mit $\widehat{\beta}_0$, um die Bootstrap-Stichproben zu generieren. Wenn wir uns nur fürs Mittel interessieren, hat diese Methode überhaupt keinen Mehrwert, denn sie ist mathematisch der zuerst besprochenen Bootstrap-Methode gleich. Aber sie ist pädagogisch wertvoller (und manchmal auch statistisch besser), wenn wir später mehrere β s haben werden. Konkret:

1. Man berechnet $\widehat{\beta}_0$ und erhält dazu auch noch einen Vektor $\hat{\epsilon}$, der die Werte $\hat{\epsilon}_1$ bis $\hat{\epsilon}_n$ enthält.
2. Man zieht eine Bootstrap-Stichprobe aus $\hat{\epsilon}$ (*sampling with replacement*). Diese kann man als $\hat{\epsilon}^*$ bezeichnen. Dieser Vektor enthält ebenso n Werte, wobei bestimmte $\hat{\epsilon}_i$ eventuell nicht vorkommen, andere dafür mehrmals.
3. Man kombiniert $\widehat{\beta}_0$ und $\hat{\epsilon}^*$. Dies ergibt eine neue Reihe von y -Werten: $y_i^* = \widehat{\beta}_0 + \hat{\epsilon}_i^*$.
4. Man schätzt nun auf der Basis von y^* erneut den Parameter von Interesse ($\widehat{\beta}_0^*$).
5. Man führt Schritte 2–4 ein paar tausend Mal aus und erhält so die Verteilung der gebootstrappten β_0 -Schätzungen.

Der unten stehende R-Code implementiert diese Schritte.

```
n_bootstrap <- 20000
bs_b0 <- vector(length = n_bootstrap)
residuals <- resid(mod.lm)
predictions <- predict(mod.lm)

for (i in 1:n_bootstrap) {
  # Residuen bootstrappen
  bs_residual <- sample(residuals, replace = TRUE)

  # neuen Outcome kreieren
  bs_outcome <- predictions + bs_residual

  # Modell neu berechnen mit diesem Outcome
  bs_mod <- lm(bs_outcome ~ 1)

  # Schätzung speichern
  bs_b0[[i]] <- coef(bs_mod)[[1]]
}
```

Wir können jetzt wieder die Verteilung der gebootstrappten Parameterschätzungen visualisieren, und ihre Standardabweichung und 2.5. und 97.5. Perzentile berechnen. Die Ergebnisse sind natürlich identisch mit jenen aus Kapitel 6.

```
# Das Histogramm wird hier nicht gezeigt.
hist(bs_b0)

sd(bs_b0)

[1] 3.106529

quantile(bs_b0, probs = c(0.025, 0.975))

      2.5%      97.5%
144.6576 156.9079
```

7.4.2 Ein anderer Bootstrap

Beim Bootstrap, den wir soeben besprochen haben, sind wir davon ausgegangen, dass die Residuen in der Stichprobe genau so verteilt sind wie die Residuen in der Population. Ein Nachteil dieser Annahme ist, dass wir dadurch die Feinkörnigkeit der Residuen in der Population wohl unterschätzen. Beispielsweise gibt es für das `mod.lm`-Modell ein Residuum von -14.78 und ein Residuum von -12.78 , aber keines von -13.78 . Ein Residuum von -14.78 entspricht einer Beobachtung von $150.78 - 14.78 = 136$; ein Residuum von -13.78 entspräche einer Beobachtung von $150.78 - 13.78 = 137$. Nach unserer Annahme gäbe es in der Population also keine Versuchspersonen mit einem GJT-Ergebnis von 137.

Dies ist natürlich eine etwas komische Annahme; in der Regel hat sie aber kaum einen Einfluss auf die Inferenzen. Aber eine Alternative wäre, dass wir davon ausgehen, dass die Residuen in der Population normalverteilt sind. Normalverteilungen sind unendlich feinkörnig, sodass diese Annahme sozusagen den Gegenpol der ersten Annahme darstellt. Das Mittel der Residuen beträgt 0, sodass wir nur die Standardabweichung der normalverteilten Restfehler in der Population finden müssen. Diese wird durch die Standardabweichung der Restfehler in der Stichprobe geschätzt. Dafür können wir hier zwei Funktionen verwenden:

```
sd(resid(mod.lm))

[1] 27.31769

sigma(mod.lm)

[1] 27.31769
```

In diesem Beispiel (lineares Modell ohne Prädiktoren) sind diese Werte identisch, aber sobald Prädiktoren im Spiel sind, liefert `sigma()` die bessere Schätzung der Standardabweichung der Residuen. Sie wird so berechnet:

$$\hat{\sigma}_\varepsilon = \sqrt{\frac{1}{n-p} \sum_{i=1}^n \hat{\varepsilon}_i^2}, \quad (7.2)$$

wo p die Anzahl geschätzten β s ist. In diesem Fall ist $p = 1$, sodass die Gleichung die Stichprobenstandardabweichung der Residuen ergibt:

```
sqrt(sum(resid(mod.lm)^2)/(length(resid(mod.lm)) - length(coef(mod.lm))))
[1] 27.31769
```

Hier teilt man durch $n - p$ aus dem gleichen Grund, weshalb man bei der Standardabweichung der Beobachtungen in der einer Stichprobe durch $n - 1$ teilt: um eine systematische Unterschätzung zum grössten Teil entgegenzuwirken.

Anstatt für jede Bootstrapstichprobe die Residuen durch *sampling with replacement* zu generieren, werden sie hier zufällig aus einer Normalverteilung mit $\mu = 0$ und $\sigma = \hat{\sigma}_\varepsilon$ generiert:

```
n_bootstrap <- 20000
bs_b0 <- vector(length = n_bootstrap)
sd_residuals <- sigma(mod.lm)
predictions <- predict(mod.lm)

for (i in 1:n_bootstrap) {
  # Neue Residuen aus Normalverteilung generieren
  bs_residual <- rnorm(n = 76, sd_residuals)

  # neuen Outcome kreieren
  bs_outcome <- predictions + bs_residual

  # Modell neu berechnen mit diesem Outcome
  bs_mod <- lm(bs_outcome ~ 1)

  # Schätzung speichern
  bs_b0[[i]] <- coef(bs_mod)[1]
}
```

```
# Histogramm (nicht gezeigt)
hist(bs_b0)
```

```
# Geschätzter Standardfehler
sd(bs_b0)

[1] 0.1147176

# 95% Konfidenzintervall
quantile(bs_b0, probs = c(0.025, 0.975))

      2.5%      97.5%
177.8677 178.3179
```

Diese Art von Bootstrap—bei der wir davon ausgehen, dass die Residuen eine bestimmte Verteilung haben, und wir die relevanten Parameter dieser Verteilung anhand der Stichprobe schätzen—nennt man einen **parametrischen Bootstrap**. Den Bootstrap aus dem letzten

Abschnitt—bei der man Bootstrapstichproben der Modellresiduen mit den Modellvorhersagen kombiniert und die relevanten Parameter anhand dieser neuen Werte schätzt—nennt man einen **semiparametrischen Bootstrap**. Den Bootstrap aus dem letzten Kapitel—bei der man Bootstrapstichproben aus dem ursprünglichen Datensatz generiert—nennt man einen **nichtparametrischen Bootstrap**.

Man bemerke hier übrigens, dass sowohl die Annahme, dass die Residuen in der Population genau so wie in der Stichprobe verteilt sind, als auch die Annahme, dass sie normalverteilt und unendlich feinkörnig sind, hier gar nicht stimmen können, da bei dieser Studie nur Ganzzahlen zwischen 0 und 204 hätten vorkommen können. Die Tatsache, dass wir unter unterschiedlichen Annahmen zum gleichen Ergebnis kommen, deutet bereits darauf hin, dass bei dieser Stichprobengröße und bei dieser Art von Datenverteilung die Schätzung der Unsicherheit eines Mittels nicht massgeblich von Annahmen über die genaue Verteilung der Residuen in der Population abhängt.

7.4.3 Mit *t*-Verteilungen

Wenn man ohnehin davon ausgehen will, dass die Residuen normalverteilt sind, kann man den geschätzten Standardfehler und das Konfidenzintervall auch analytisch herleiten. Die Formeln, die man dazu braucht, werden hier nicht gezeigt, denn sie haben kaum einen didaktischen Mehrwert. In R kann man die `summary()`-Funktion verwenden, um den geschätzten Standardfehler zu berechnen (Std. Error):

```
summary(mod.lm)

Call:
lm(formula = GJT ~ 1, data = d)

Residuals:
    Min       1Q   Median       3Q      Max
-46.776 -23.026  -0.276  23.224  47.224

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   150.776      3.134   48.12  <2e-16

Residual standard error: 27.32 on 75 degrees of freedom
```

β_0 heisst hier (Intercept). Was `t value` und `Pr(>|t|)` bedeuten, werden wir erst in einem späteren Kapitel besprechen.

Das 95%-Konfidenzintervall kann mit `confint()` berechnet werden.

```
confint(mod.lm)

              2.5 %    97.5 %
```

```
(Intercept) 144.534 157.0187
```

Natürlich kann man auch gerne andere Intervalle berechnen, z.B. 80%-Konfidenzintervalle:

```
confint(mod.lm, level = 0.80)

      10 %      90 %
(Intercept) 146.7248 154.8278
```

Aufgabe 7.1. Warum wird bei der `summary()`-Funktion schon der Median aber nicht das Mittel der Residuen gezeigt? ◇

7.5 Fazit

- Datenpunkte kann man als eine Kombination einer Gemeinsamkeit aller Datenpunkte in der Population und einer spezifischen Abweichung verstehen.
- In der Regel ist die Gemeinsamkeit von Interesse; diese wird aber anhand der Abweichungen und eines Optimierungskriteriums geschätzt.
- Das am meisten verwendete Optimierungskriterium ist die Methode der kleinsten Quadrate, die im 'univariaten' Fall (= wenn man nur mit einer Variablen arbeitet) das Mittel ergibt, aber es existieren auch andere Methoden.
- Die Unsicherheit in der Parameterschätzung kann mithilfe des Bootstraps oder anhand weiterer Annahmen geschätzt werden.

Kapitel 8

Einen Prädiktor hinzufügen

In diesem Kapitel beschäftigen wir uns mit der Frage, wie der Zusammenhang zwischen einem kontinuierlichen Prädiktor und einem kontinuierlichen Outcome erfasst werden kann. Als Beispiel dienen uns dabei wieder die GJT- und AOA-Daten im Datensatz von DeKeyser et al. (2010), die wir als kontinuierlich auffassen werden.

In den letzten zwei Kapiteln haben wir uns mit der zentralen Tendenz der GJT-Daten beschäftigt. Eigentlich interessierten sich DeKeyser et al. (2010) aber nicht sosehr für diese, sondern für den Zusammenhang zwischen GJT und AOA. Diesen Zusammenhang schauen wir uns hier an. Wie immer ist es eine gute Idee, die Daten zunächst grafisch darzustellen. Wenn man sich für den Zusammenhang zwischen zwei kontinuierlichen Variablen interessiert, bietet sich das **Streudiagramm** (*scatterplot*) an; siehe Abbildung 8.1. Beim Zeichnen eines Streudiagramms muss man spezifizieren, welche Variable entlang der x -Achse und welche entlang der y -Achse dargestellt wird. Wenn es naheliegender ist, dass Variable A Variable B beeinflusst als umgekehrt, empfiehlt es sich, Variable A entlang der x -Achse darzustellen und Variable B entlang der y -Achse. Hier ist es unmöglich, dass die Grammatikalitätsurteile das Erwerbsalter beeinflussen, aber sehr wohl, dass das Erwerbsalter die Grammatikalitätsurteile beeinflussen.

```
ggplot(data = d,  
       aes(x = AOA,  
           y = GJT)) +  
  geom_point(shape = 1) + # shape = 1 für leere Kreischen; siehe ?points  
  xlab("Erwerbsalter (Jahre)") +  
  ylab("Ergebnis\nGrammatikalitätsurteile")
```

Das Streudiagramm zeigt, dass die Leistung beim GJT mit steigendem Erwerbsalter allmählich abnimmt. Diese Senkung scheint auch ungefähr linear zu sein; zum Vergleich zeigt Abbildung 8.2 vier deutliche Beispiele von nicht-linearen Zusammenhängen. Ausserdem gibt es keine einzelnen Punkte, die sehr weit von der Punktwolke entfernt liegen, und alle Daten sind plausibel: Es gibt keine 207-Jährigen und in DeKeyser et al. (2010) kann man lesen, dass das GJT-Instrument aus 204 binären Items bestand. Das höchstmögliche Ergebnis von 204 wird hier

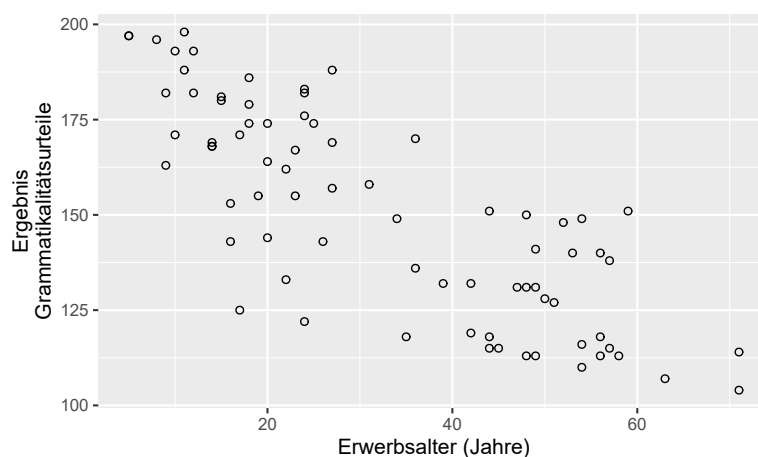


Abbildung 8.1: Zusammenhang zwischen AOA und GJT in der Studie von DeKeyser et al. (2010). Die AOA-Werte stehen entlang der x -Achse und die GJT-Werte entlang der y -Achse, da es plausibler ist, dass das Erwerbsalter die Grammatikalitätsurteile beeinflusst als umgekehrt.

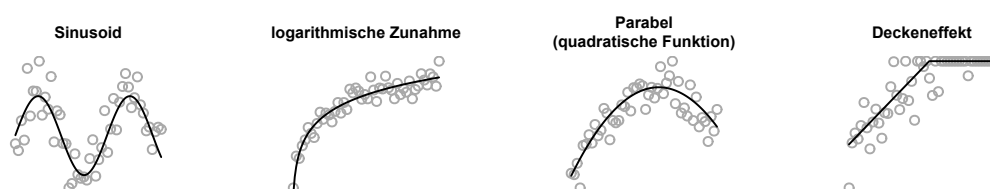


Abbildung 8.2: Beispiele von nicht-linearen Zusammenhängen.

nicht überschritten.

Im Folgenden werden wir eine Antwort auf diese beiden Fragen geben:

1. *Wie perfekt* ist der Zusammenhang zwischen den GJT- und AOA-Variablen? Wie genau 'perfekt' in diesem Kontext zu verstehen ist, sollte in Kürze klar werden. Um diese Frage zu beantworten, verwendet man oft **Korrelationsanalyse**.
2. *Was* ist der Zusammenhang zwischen den beiden Variablen? Anders gesagt, wenn wir den Wert einer Variablen kennen, *wie* können wir dann den Wert der anderen Variablen schätzen? (**Regressionsanalyse**)

Beide Fragen werden oft miteinander verwechselt, was manchmal zu Verwirrungen führt (siehe Vanhove, 2013). Zwei Beispiele sollten den Unterschied klar machen.

Beispiel 8.1 (Temperatur). Wenn man die Temperatur in Grad Celsius kennt, kann man die Temperatur in Grad Fahrenheit perfekt 'schätzen'. Die Korrelation ist also äusserst stark (Frage

1). Damit wissen wir aber noch nicht, wie wir die Temperatur in Grad Fahrenheit berechnen können, wenn wir die Temperatur in Grad Celsius kennen. Eine Regressionsanalyse würde zeigen, dass wir dazu die folgende Formel anwenden müssten:

$$\text{Grad Fahrenheit} = 32 + \frac{9}{5} \cdot \text{Grad Celsius.} \quad (8.1)$$

◇

Beispiel 8.2 (Körpergröße und -gewicht). Wenn man die Körpergröße eines Menschen kennt, kann man sein Gewicht wesentlich besser schätzen, als wenn man die Körpergröße nicht kennt. Die Schätzung ist aber nicht perfekt, denn Menschen mit der gleichen Körpergröße variieren ja in ihrem Gewicht. Die Korrelation ist folglich zwar positiv, aber nicht so hoch wie im letzten Beispiel (Frage 1). Um zu wissen, wie man das Gewicht am besten anhand der Größe schätzt (z.B. Gewicht in kg = 0.6 kg/cm · Größe in cm – 35 kg für Frauen zwischen 145 und 185 cm), braucht es Regressionsanalyse. ◇

Die zweite Frage ist m.E. in der Regel (nicht immer!) viel sinnvoller. In unserem Beispiel wäre das Ziel also, eine Gleichung wie Gleichung 8.1 zu finden, anhand derer man Unterschiede im GJT-Ergebnis mit Unterschieden im AOA verknüpfen kann. Um diese Gleichung zu finden, brauchen wir zuerst aber eine Antwort auf die erste Frage.

Bemerkung 8.3 (nicht-lineare Zusammenhänge). Korrelation- und Regressionsanalyse können sinnvoll sein, um lineare Zusammenhänge zu untersuchen. Ist der Zusammenhang zwischen den Variablen nicht *ungefähr* gerade, dann kann man die Berechnung noch immer ausführen. Diese würde dann aber sinnlose (nicht ‘falsche’!) Ergebnisse liefern. Bei einem verantwortungsvollen Umgang mit quantitativen Forschungsdaten sollte die Frage der **Relevanz** immer im Vordergrund stehen.

Manchmal kann man übrigens Daten sinnvoll transformieren, sodass der Zusammenhang linear wird (Beispiele in etwa Baayen, 2008 und Gelman & Hill, 2007). Ist dies nicht möglich, dann dürften komplexere Verfahren geeignet sein. Siehe hierzu <https://m-clark.github.io/generalized-additive-models/>, Wieling (2018) und Baayen & Linke (2020). ◇

Empfehlung: Fragen und Werkzeuge. Korrelations-, Regressions- und sonstige Analysen, Modelle und Tests sind lediglich Werkzeuge. Je nach Fragestellung sind diese Werkzeuge nützlich oder nutzlos. Anstatt sich etwa vorzunehmen, eine Korrelationsanalyse oder einen *t*-Test durchzuführen (vielleicht, weil dies in einer bestimmten Forschungsliteratur gang und gäbe ist), ist es sinnvoller, die Frage ohne ablenkenden technischen Wortschatz (z.B. *Korrelation*, *signifikant*, *Interaktion*) zu formulieren und sich dann zu überlegen, welches Werkzeug für deren Beantwortung am nützlichsten ist. Das Ziel einer Datenerhebung und einer Analyse ist es, eine Frage zu beantworten – nicht ein bestimmtes Werkzeug zu benutzen.

8.1 Antwort auf Frage 1: Kovarianz und Korrelation

Um numerisch zu beschreiben, wie stark zwei Zufallsvariablen miteinander zusammenhängen, brauchen wir ein Mass, dessen absoluter Wert gross ist, wenn kleine Unterschiede in x mit kleinen Unterschieden in y zusammenhängen und grosse Unterschiede in x mit grossen Unterschieden in y , und dessen absoluter Wert klein ist, wenn grosse Unterschiede in der einen Variablen mit nur kleinen Unterschieden in der anderen Variablen zusammenhängen. Ein solches Mass ist die **Kovarianz**.

Definition 8.4 (Populationskovarianz). Seien X und Y zwei Zufallsvariablen mit endlicher Varianz auf einem Wahrscheinlichkeitsraum Ω . Dann definieren wir die Kovarianz von X und Y als

$$\text{Cov}(X, Y) := \mathbb{E}((X - \mathbb{E}(X))(Y - \mathbb{E}(Y))).$$

◇

Beispiel 8.5. Wir greifen das Jass-Beispiel aus Kapitel 4 wieder auf. In Beispiel 4.26 wurde die Variable X durch die Punktzahl der Karten im *Obenabe*-Spiel definiert:

$$X(\omega) := \begin{cases} 11, & \text{falls } \omega \text{ ein Ass ist,} \\ 4, & \text{falls } \omega \text{ ein König ist,} \\ 3, & \text{falls } \omega \text{ eine Dame ist,} \\ 2, & \text{falls } \omega \text{ ein Bub ist,} \\ 10, & \text{falls } \omega \text{ eine 10 ist,} \\ 8, & \text{falls } \omega \text{ eine 8 ist,} \\ 0, & \text{sonst.} \end{cases}$$

Wir definieren nun die zusätzliche Variable Y durch die Punktzahl der Karten im *Undenufe*-Spiel:

$$Y(\omega) := \begin{cases} 11, & \text{falls } \omega \text{ eine 6 ist,} \\ 4, & \text{falls } \omega \text{ ein König ist,} \\ 3, & \text{falls } \omega \text{ eine Dame ist,} \\ 2, & \text{falls } \omega \text{ ein Bub ist,} \\ 10, & \text{falls } \omega \text{ eine 10 ist,} \\ 8, & \text{falls } \omega \text{ eine 8 ist,} \\ 0, & \text{sonst.} \end{cases}$$

Wie Sie kontrollieren können, gilt $\mathbb{E}(Y) = \mathbb{E}(X) = 4.22$. Zur Berechnung der Kovarianz holen

wir tief Atem:

$$\begin{aligned}
 \text{Cov}(X, Y) &= \mathbb{E}((X - \mathbb{E}(X))(Y - \mathbb{E}(Y))) \\
 &= \frac{1}{36}(X(6\clubsuit) - \mathbb{E}(X))(Y(6\clubsuit) - \mathbb{E}(Y)) + \\
 &\quad \frac{1}{36}(X(6\diamond) - \mathbb{E}(X))(Y(6\diamond) - \mathbb{E}(Y)) + \\
 &\quad \dots + \\
 &\quad \frac{1}{36}(X(A\spadesuit) - \mathbb{E}(X))(Y(A\spadesuit) - \mathbb{E}(Y)) \\
 &= \frac{1}{9}(0 - 4.22)(11 - 4.22) + \frac{2}{9}(0 - 4.22)(0 - 4.22) + \\
 &\quad \frac{1}{9}(8 - 4.22)(8 - 4.22) + \frac{1}{9}(10 - 4.22)(10 - 4.22) + \\
 &\quad \frac{1}{9}(2 - 4.22)(2 - 4.22) + \frac{1}{9}(3 - 4.22)(3 - 4.22) + \\
 &\quad \frac{1}{9}(4 - 4.22)(4 - 4.22) + \frac{1}{9}(11 - 4.22)(0 - 4.22) \\
 &\approx 0.55.
 \end{aligned}$$

◇

Lemma 8.6 (Eigenschaften der Kovarianz). Seien X, Y, Z drei Zufallsvariablen auf einem gemeinsamen Wahrscheinlichkeitsraum und seien a, b Konstanten. Dann gilt Folgendes:

1. $\text{Cov}(X, X) = \text{Var}(X)$.
2. $\text{Cov}(X, Y) = \text{Cov}(Y, X)$.
3. $\text{Cov}(X + Y, Z) = \text{Cov}(X, Z) + \text{Cov}(Y, Z)$.
4. $\text{Cov}(aX, bY) = ab\text{Cov}(X, Y)$.
5. Sind X, Y unabhängig, so gilt $\text{Cov}(X, Y) = 0$.

Anhand dieser Eigenschaften können wir eine allgemeine Formel für die Varianz der Summe von zwei Zufallsvariablen herleiten:

$$\begin{aligned}
 \text{Var}(X + Y) &= \text{Cov}(X + Y, X + Y) && [(1)] \\
 &= \text{Cov}(X, X + Y) + \text{Cov}(Y, X + Y) && [(3)] \\
 &= \text{Cov}(X + Y, X) + \text{Cov}(X + Y, Y) && [(2)] \\
 &= \text{Cov}(X, X) + \text{Cov}(Y, X) + \text{Cov}(X, Y) + \text{Cov}(Y, Y) && [(3)] \\
 &= \text{Var}(X) + 2\text{Cov}(X, Y) + \text{Var}(Y). && [(1, 2)]
 \end{aligned}$$

Insbesondere gilt: Falls $\text{Cov}(X, Y) = 0$, so $\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y)$.

◇

Beispiel 8.7 (keine Kovarianz, aber nicht unabhängig). Sind zwei Zufallsvariablen unabhängig,

so gilt $\text{Cov}(X, Y) = 0$. Das Umgekehrte gilt jedoch nicht. Dazu betrachten wir den Grundraum

$$\Omega := \{(0, 0), (1, -1), (1, 1)\}$$

versehen mit einer Gleichverteilung. Die Zufallsvariable X bildet (ω_1, ω_2) auf ω_1 ab; die Zufallsvariable Y bildet (ω_1, ω_2) auf ω_2 ab. Es gilt $\mathbb{E}(X) = 2/3, \mathbb{E}(Y) = 0$. Daher

$$\text{Cov}(X, Y) = \frac{1}{3}(0 - 2/3)(0 - 0) + \frac{1}{3}(1 - 1/3)(-1 - 0) + \frac{1}{3}(1 - 1/3)(1 - 0) = 0.$$

Jedoch gilt auch

$$\mathbb{P}(X = 1, Y = 1) = \frac{1}{3} \neq \frac{2}{3} \cdot \frac{1}{3} = \mathbb{P}(X = 1)\mathbb{P}(Y = 1).$$

Also sind X, Y nicht unabhängig. ◇

Die Kovarianz zwischen zwei Zufallsvariablen wird in der Regel anhand der **Stichprobenkovarianz** geschätzt.

Definition 8.8 (Stichprobenkovarianz). Seien $(X_1, Y_1), \dots, (X_n, Y_n)$ unabhängige und identisch verteilte Paare von Beobachtungen, so definieren wir die Stichprobenkovarianz

$$\widehat{\text{Cov}}_{XY} := \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y}). \quad (8.2)$$

Hier sind \bar{X}, \bar{Y} die Stichprobenmittel von X, Y . ◇

Die Summe der Produkte $(\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y}))$ wird durch $n - 1$ statt durch n geteilt aus dem gleichen Grund, weshalb dies bei der Varianzberechnung gemacht wird. Eine intuitivere Erklärung ist, dass man ja nicht über den Zusammenhang von zwei Variablen sprechen kann, wenn man nur eine Beobachtung pro Variable hat. Das Streudiagramm würde dann nur einen Punkt zeigen. Wenn $n = 1$, ist $n - 1 = 0$ und dann ergibt die Gleichung keine Antwort, denn durch 0 kann nicht geteilt werden.

In R:

```
# kompliziert:
sum((d$AOA - mean(d$AOA)) * (d$GJT - mean(d$GJT))) / (nrow(d) - 1)

[1] -394.9311

# einfacher:
cov(d$AOA, d$GJT)

[1] -394.9311
```

Ist die Kovarianz positiv, dann besteht ein positiver linearer Zusammenhang zwischen den beiden Variablen (je grösser x , desto grösser y); ist die Kovarianz negativ, dann gibt es einen negativen linearen Zusammenhang (je grösser x , desto kleiner y). Abgesehen von diesen zwei Richtschnuren ist das Kovarianzmass schwierig zu interpretieren, weshalb Sie es in der Literatur

nur selten antreffen werden. Aber Kovarianz ist ein wichtiges Konzept in der Mathe hinter komplexeren Verfahren, weshalb es sich trotzdem lohnt, zumindest zu wissen, dass es besteht.

Da das Kovarianzmass nicht einfach zu interpretieren ist, wird meistens Pearsons **Produkt-Moment-Korrelationskoeffizient** (oder einfach Pearsons Korrelation) verwendet. Diese Zahl drückt aus, wie gut der Zusammenhang durch eine gerade Linie beschrieben werden kann. Es wird ähnlich zum Kovarianzmass berechnet, aber die Variablen werden in Standardabweichungen zum Stichprobemittel ausgedrückt. Dies ergibt dann immer eine Zahl zwischen -1 und 1 :

$$\rho_{XY} := \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y}.$$

Wird die Korrelation anhand einer Stichprobe geschätzt, so wird diese Formel sinngemäss angepasst:

$$r_{XY} := \hat{\rho}_{XY} := \frac{\widehat{\text{Cov}}(X, Y)}{S_X S_Y} = \frac{1}{n-1} \sum_{i=1}^n \frac{X_i - \bar{X}}{S_X} \frac{Y_i - \bar{Y}}{S_Y}.$$

```
# kompliziert:
cov(d$AOA, d$GJT) / (sd(d$AOA) * sd(d$GJT))

[1] -0.8028533

# einfach:
cor(d$AOA, d$GJT)

[1] -0.8028533
```

Sobald irgendein Wert fehlt, ergibt die `cor()`-Funktion das Ergebnis 'NA' (*not available*). Eine Möglichkeit ist dann, die Beobachtungen mit einem oder zwei fehlenden Werten zu ignorieren:

```
cor(d$AOA, d$GJT, use = "pairwise.complete.obs")

[1] -0.8028533
```

Ist $r = 1$, dann liegen alle Datenpunkte perfekt auf einer geraden, steigenden Linie. Dies deutet fast ausnahmslos auf eine Tautologie hin. Zum Beispiel sind Körpergrößen in Zentimetern und in Zoll perfekt korreliert, aber dieser Zusammenhang ist nicht spektakulär sondern höchst langweilig. Ist $r = -1$, dann liegen alle Datenpunkte auf einer geraden, senkenden Linie. Dies deutet wohl darauf hin, dass die beiden Variablen perfekt komplementär sind. Zum Beispiel wird die Anzahl richtiger Antworten bei einem Test oft zu $r = -1$ mit der Anzahl falscher Antworten korrelieren; auch dies ist wenig spektakulär. Ist $r = 0$, dann ist die Linie perfekt senkrecht, d.h., es gibt überhaupt keinen linearen Zusammenhang zwischen den beiden Variablen. Je grösser der absolute Wert von r , desto näher befinden sich die Datenpunkte bei der geraden Linie. Anders ausgedrückt: Je grösser der absolute r -Wert, desto präziser kann man y anhand einer linearen Gleichung bestimmen, wenn man x schon kennt (und umgekehrt) als wenn man x nicht gekannt hätte. Die Korrelation zwischen X und Y ist gleich

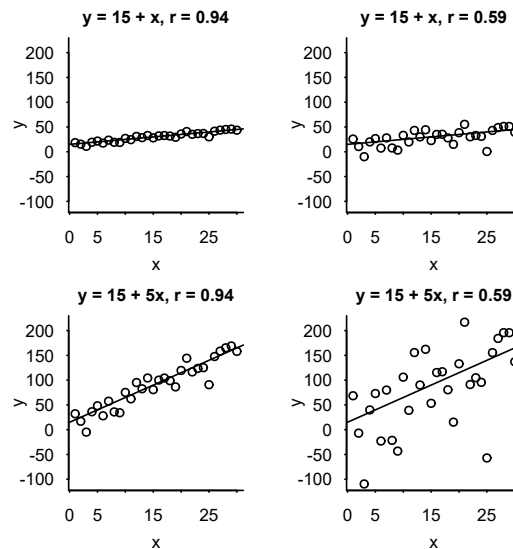


Abbildung 8.3: Korrelationskoeffizienten erzählen einem wenig über die Form eines Zusammenhangs.

der Korrelation zwischen Y und X . Es macht also nichts aus, ob man `cor(datAOA, datGJT)` oder `cor(datGJT, datAOA)` eintippt.

Beispiel 8.9 (Form vs. Stärke eines Zusammenhangs). Abbildung 8.3 zeigt vier Zusammenhänge, um die Bedeutung von Pearsons r besser zu illustrieren.

- *Oben links:* Es gibt wenig Streuung entlang der y -Achse. Die Streuung, die es gibt, wird grösstenteils von einer Geraden erfasst. r ist daher sehr hoch.
- *Oben rechts:* Es gibt nun mehr Streuung entlang der y -Achse; diese wird aber weniger gut von einer Geraden erfasst, daher der niedrigere Korrelationskoeffizient. Die Form der Geraden ist zwar gleich wie in der linken Grafik, der Korrelationskoeffizient aber nicht.
- *Unten links:* Es gibt zwar sehr viel Streuung entlang der y -Achse, aber diese wird grösstenteils von einer Geraden erfasst. r ist daher wiederum sehr hoch. Der Korrelationskoeffizient ist zwar gleich wie in der Grafik oberhalb, die Form der Geraden aber nicht.
- *Unten rechts:* Die gleiche Gerade erfasst die Streuung entlang der y -Achse weniger gut, daher ist die Form der Geraden zwar gleich, der Korrelationskoeffizient aber niedriger. ◇

Bemerkung 8.10 (Datenmuster hinter Korrelationskoeffizienten). Wie wir gerade gesehen haben, drückt Pearsons r aus, welcher Anteil der Streuung der Datenpunkte in einer Punktwolke durch eine *gerade Linie* erfasst wird. Es gibt keine direkte Antwort auf die Frage, wie diese Linie aussieht (ausser: steigend oder senkend); siehe die vier obigen Beispiele.

Ausserdem ist es möglich, dass es einen sehr starken (nicht-linearen) Zusammenhang zwischen

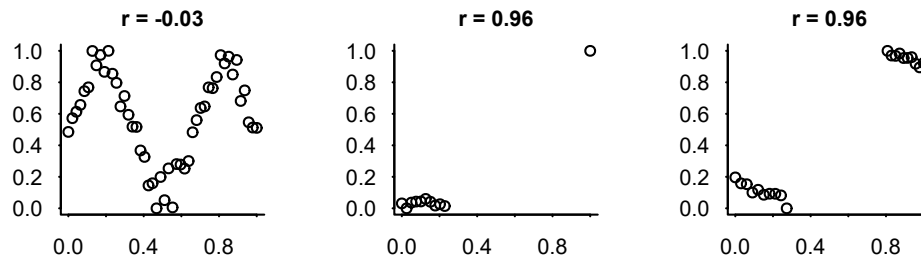


Abbildung 8.4: Ein Korrelationskoeffizient nahe 0 muss nicht heißen, dass es keinen Zusammenhang zwischen den Variablen gibt, und ein Korrelationskoeffizient nahe 1 muss nicht heißen, dass das Muster in den Daten am besten durch einen starken positiven Zusammenhang beschrieben wird.

zwei Variablen gibt, dieser aber in Pearsons r nicht zum Ausdruck kommt (Abbildung 8.4, links). Umgekehrt kann r den Eindruck geben, dass es sich um einen ziemlich starken linearen Zusammenhang handelt, während ein solcher Zusammenhang für die meisten Datenpunkte kaum vorliegt (Mitte), oder während der Zusammenhang sogar eigentlich in die umgekehrte Richtung geht. So gibt es in der rechten Grafik zwei Gruppen, in denen der Zusammenhang negativ ist. Der Koeffizient ist jedoch positiv, wenn die beiden Gruppen gleichzeitig betrachtet werden. Das Problem ist hier nicht, dass r falsch berechnet wird, sondern, dass die Berechnung von r hier kaum Sinn ergibt. Bevor man sich mit der Richtigkeitsfrage auseinandersetzt, sollte man sich eben zuerst mit der Relevanzfrage befassen.

Mit der `plot_r()`-Funktion im `cannonball`-Paket können Sie selber Streudiagramme zeichnen, die alle anders aussehen, aber den gleichen Korrelationskoeffizienten haben. Unter <https://github.com/janhove/cannonball> finden Sie Anweisungen, wie Sie dieses Paket installieren können. Der Blogeintrag *What data patterns can lie behind a correlation coefficient?* (21.11.2016) beschreibt die `plot_r()`-Funktion. Abbildung 8.5 zeigt 16 Zusammenhänge mit je 50 Beobachtungen, die alle eine Korrelation von $r = -0.72$ aufzeigen:

```
library(cannonball)
plot_r(n = 50, r = -0.72)
```

Spielen Sie mit der `plot_r()`-Funktion herum, um besser zu verstehen, was ein Korrelationskoeffizient eben alles nicht bedeutet und was der Einfluss von nicht-linearen Zusammenhängen und Ausreißern sein kann:

```
plot_r(n = 15, r = 0.9)
plot_r(n = 80, r = 0.0)
plot_r(n = 30, r = 0.4)
```

Die Funktionsdokumentation können Sie wie gehabt abrufen:

All correlations: $r(50) = -0.72$

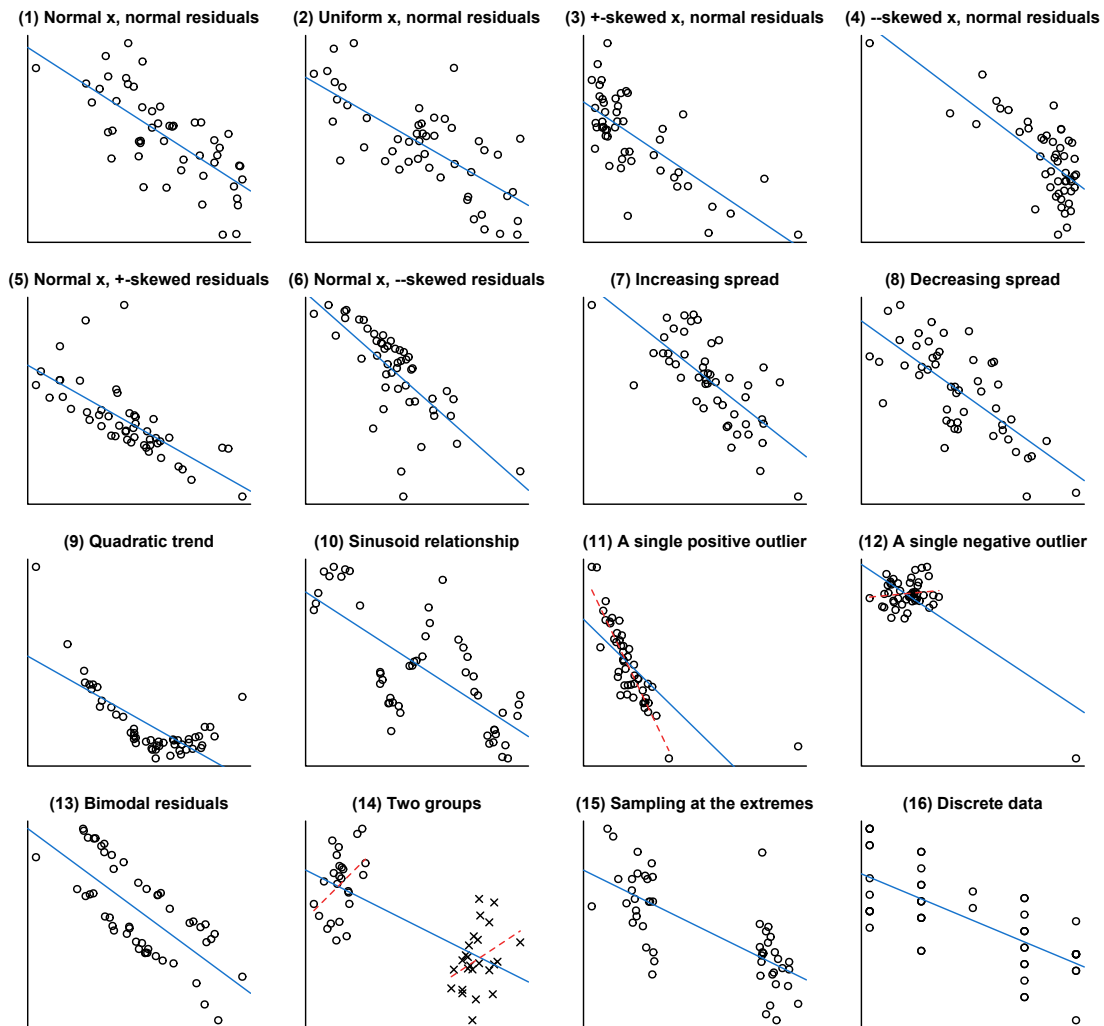


Abbildung 8.5: Alle sechzehn Zusammenhänge zeigen eine Korrelation von -0.72 auf, sehen jedoch zum Teil ganz unterschiedlich aus.

?plot_r




Merksatz: Ein Korrelationskoeffizient kann einer Vielzahl von Zusammenhängen entsprechen! Schauen Sie sich, bevor Sie Korrelationskoeffizienten berechnen, immer die Daten **grafisch** (Streudiagramm) an. Nehmen Sie diese Streudiagramme in Ihre Papers, Arbeiten und Vorträge auf. Ein Korrelationskoeffizient ohne Streudiagramm ist m.E. wertlos.

Neben dem Korrelationskoeffizienten nach Pearson trifft man in der Forschungsliteratur ab und zu andere Masse für die Assoziation zwischen zwei Variablen an.

Bemerkung 8.11 (Spearman's ρ). Um Spearman's ρ zu berechnen, drückt man die Beobachtungen zunächst in Rängen aus, d.h., man ordnet die Daten von klein nach gross und schaut, auf welchem Platz die einzelnen Datenpunkte stehen. Dann berechnet man einfach die Pearsonkorrelation für die Ränge statt für die Rohwerte:

```
cor(rank(d$AOA), rank(d$GJT))
[1] -0.7887659

# einfacher:
cor(d$AOA, d$GJT, method = "spearman")
[1] -0.7887659
```

Spearman's ρ kann nützlich sein, wenn der Zusammenhang zwischen zwei Variablen monoton aber nicht-linear ist (Monoton heisst tendenziell steigend oder tendenziell senkend; nicht etwa zuerst steigend und dann senkend.) oder wenn ein Ausreisser das Globalbild zerstört, aber man ihn aus irgendwelchem Grund nicht aus dem Datensatz entfernen kann. Wenn Spearman's $\rho = 1$, dann ist der Zusammenhang perfekt monoton steigend (höhere Werte in x entsprechen immer höheren Werten in y), wenn Spearman's $\rho = -1$, dann ist der Zusammenhang perfekt monoton senkend, und wenn $\rho = 0$, dann gibt es keinen monotonen Zusammenhang in den Daten. Bemerken Sie, dass man mit ρ eine andere Frage beantwortet als mit r : *Wie perfekt ist der monotone Zusammenhang?* vs. *Wie perfekt ist der lineare Zusammenhang?* 

Bemerkung 8.12 (Kendalls τ). Das Assoziationsmass Kendalls τ wird recht selten verwendet. Die Berechnung ist konzeptuell relativ einfach (siehe Noether, 1981), aber schaut in R-Code schwierig aus, weshalb ich sie hier nur in Worten zusammenfasse:

1. Vergleiche jeden x -Wert mit jedem anderen x -Wert und notiere, ob Ersterer grösser oder kleiner als Letzterer ist. Wenn die x -Werte zum Beispiel 5, 3, 8 und 7 sind, erhält man folgende Vergleiche:
 - 5 vs. 3: grösser,
 - 5 vs. 8: kleiner,

- 5 vs. 7: kleiner,
 - 3 vs. 8: kleiner,
 - 3 vs. 7: kleiner,
 - 8 vs. 7: grösser.
2. Vergleiche jeden y -Wert mit jedem anderen y -Wert. Für die y -Werte 8, -2, -4, -3 erhält man: grösser, grösser, grösser, grösser, grösser, kleiner.
3. Zähle, wie viele Vergleiche in die gleiche Richtung gehen:
- 1. Vergleich: grösser–grösser: gleich,
 - 2. Vergleich: kleiner–grösser: anders,
 - 3. Vergleich: kleiner–grösser: anders,
 - 4. Vergleich: kleiner–grösser: anders,
 - 5. Vergleich: kleiner–grösser: anders,
 - 6. Vergleich: grösser–kleiner: anders.
- Also 1 Vergleich, der in die gleiche Richtung geht ('konkordant'), und 5, die in die andere Richtung gehen ('diskordant').
4. Schätze jetzt Kendalls τ wie folgt:

$$\hat{\tau} = \frac{\text{Anzahl konkordant} - \text{Anzahl diskordant}}{\text{Anzahl Vergleiche}}.$$

Das Hütchen zeigt, dass wir es mit einer Schätzung auf der Basis einer Stichprobe zu tun haben. Also:

$$\hat{\tau} = \frac{1 - 5}{6} = -0.67.$$

```
# Für unser kleines Beispiel
x <- c(5, 3, 8, 7)
y <- c(8, -2, -4, -3)
cor(x, y, method = "kendall")

[1] -0.6666667

# Für die AOA-GJT-Daten
cor(d$AOA, d$GJT, method = "kendall")

[1] -0.6035606
```

Kendalls $\hat{\tau}$ schätzt den Unterschied zwischen der Proportion konkordanter Vergleiche und der Proportion diskordanter Vergleiche. Diese Interpretation finde ich selber schwierig, aber es gibt eine einfachere Interpretation: Nimm zwei beliebige (x, y) -Paare (also (x_1, y_1) und (x_2, y_2)).

Wenn x_2 grösser ist als x_1 , dann ist es $\frac{1+\hat{\tau}}{1-\hat{\tau}}$ Mal wahrscheinlicher, dass auch y_2 grösser als y_1 ist als dass er kleiner ist. Für die AOA–GJT-Daten: Wenn eine Person einen höheren AOA-Wert als eine andere hat, dann ist es $\frac{1+(-0.60)}{1-(-0.60)} = 0.25$ Mal wahrscheinlicher, dass sie auch einen höheren GJT-Wert als einen kleineren hat. Oder anders gesagt: Es ist 4 Mal wahrscheinlicher, dass sie einen kleineren GJT-Wert als einen grösseren hat. \diamond

In der Praxis ist die Anwendung von Spearmans ρ und Kendalls τ ist eher beschränkt. Statt automatisch auf ρ oder τ zurückzugreifen, wenn ein Zusammenhang nicht-linear ist oder wenn man einen Ausreisser vermutet, lohnt es sich m.E. eher, darüber nachzudenken, ob (a) man sich tatsächlich für Frage 1 (Stärke des Zusammenhangs) interessiert (die Relevanzfrage), (b) man eine oder beide Variablen nicht sinnvoll transformieren kann, sodass sich ein linearerer Zusammenhang ergibt, oder (c) der vermutete Ausreisser überhaupt ein legitimer Datenpunkt ist.

Bemerkung 8.13 ('starke' und 'schwache' Korrelationen). Korrelationskoeffizienten werden oft—ohne Berücksichtigung der Forschungsfrage oder des Kontextes—als klein bzw. schwach, mittelgross oder gross bzw. stark eingestuft. Ich halte dies für wenig sinnvoll, weshalb ich diese Einstufungen hier nicht reproduziere. Selber finde ich, dass Korrelationskoeffizienten überverwendet werden. Blogeinträge zu diesem Thema:

- *Why I don't like standardised effect sizes* (5.2.2015)
- *More on why I don't like standardised effect sizes* (16.3.2015)
- *Abandoning standardised effect sizes and opening up other roads to power* (14.7.2017)

Siehe weiter auch Baguley (2009). \diamond

Da sie auf der Basis von Stichproben berechnet werden, sind auch Korrelationskoeffizienten vom Stichprobenfehler betroffen: Andere Stichproben aus der gleichen Population werden Korrelationskoeffizienten ergeben, die mehr oder weniger voneinander abweichen. Die Ungenauigkeit bzw. die Variabilität eines auf einer Stichprobe basierenden Korrelationskoeffizienten kann in einem Konfidenzintervall ausgedrückt werden. Besprochen werden hier eine Bootstrap-Methode und eine Methode, die auf t -Verteilungen basiert.

Bemerkung 8.14 (Konfidenzintervall mit Bootstrap). Das Vorgehen ist analog zum Bootstrap aus Kapitel 6: Aus der Stichprobe werden neue Bootstrap-Stichproben generiert und für jede Stichprobe wird die Statistik von Interesse (hier: die Korrelation zwischen AOA und GJT) berechnet. Die Streuung der Schätzungen in den Bootstrap-Stichproben gibt uns ein Indiz über die Variabilität des Korrelationskoeffizienten in Stichproben dieser Grösse.

```
n_bootstraps <- 20000
bootstraps <- vector(length = n_bootstraps)

for (i in 1:n_bootstraps) {
  # Sampling with replacement aus der beobachteten Stichprobe
```

```

bootstrap_sample <- d[sample(1:nrow(d), replace = TRUE), ]

# Korrelation im bootstrap sample berechnen und speichern
bootstraps[[i]] <- cor(bootstrap_sample$GJT, bootstrap_sample$AOA)
}

# Histogramm mit den Bootstrap-Schätzungen
# (nicht gezeigt)
hist(bootstraps, breaks = 50)

```

Da das Histogramm nicht normalverteilt ist, verzichten wir hier auf die Berechnung eines Standardfehlers. Anhand der Perzentile der Verteilung können wir aber durchaus ein Konfidenzintervall konstruieren. Hier berechne ich ein 90%-Konfidenzintervall:

```

quantile(bootstraps, probs = c(0.05, 0.95))

      5%      95%
-0.8648265 -0.7291892

```



Bemerkung 8.15 (Konfidenzintervall mit t -Verteilungen). Die Formel, mit der man anhand von einer t -Verteilung ein Konfidenzintervall um einen Korrelationskoeffizienten konstruiert, werde ich hier nicht reproduzieren, da sie erstens abschreckt und zweitens keinen konzeptuellen Mehrwert bietet. Sie basiert auf der Annahme, dass die Population, aus der die beiden Variablen gezogen wurden, 'bivariat normal' ist. Grundsätzlich heisst dies, dass—insofern es einen Zusammenhang zwischen den Variablen gibt—dieser Zusammenhang linear ist und beide Variablen normalverteilt sind. Wenn diese Annahmen plausibel sind, kann das Konfidenzintervall (hier wiederum ein 90%-Konfidenzintervall) mit der `cor.test()`-Funktion berechnet werden:

```

cor.test(d$AOA, d$GJT, conf.level = 0.9)$conf.int

[1] -0.8614924 -0.7230816
attr(,"conf.level")
[1] 0.9

```

Mit dieser Methode erhalten wir ein 90%-Konfidenzintervall von etwa $[-0.86, -0.72]$. Dieses unterscheidet sich nur minimal vom Konfidenzintervall, das wir mit dem Bootstrap berechnet haben. Beim Bootstrap sind wir jedoch nicht davon ausgegangen, dass die Population, aus der die Stichprobe stammt, bivariat normalverteilt ist. Insbesondere bei kleineren Stichproben können die Ergebnisse der Bootstrap- und der t -Methode erheblich voneinander abweichen. Wenn ihre Annahmen stimmen, ist die t -Methode in solchen Fällen zweifellos besser, aber gerade bei kleinen Stichproben sind diese Annahmen schwierig zu überprüfen. Die Leistung der Bootstrappmethode kann noch etwas verbessert werden, indem die Konfidenzintervalle anders konstruiert werden (siehe hierzu DiCiccio & Efron, 1996), aber diese Konstruktionsmethoden sind schwieriger und weniger intuitiv. Für unsere Zwecke reicht es m.E., die Warnung von

Hesterberg (2015) zu wiederholen: “Bootstrapping does not overcome the weakness of small samples as a basis for inference.” (S. 379) ◇

Aufgabe 8.16. Es gibt mittlerweile eine ausführliche Literatur zur Frage, inwieweit Zweisprachigkeit zu kognitiven Vorteilen führt. Ein kognitives **Konstrukt**, das oft in diesem Zusammenhang erwähnt wird, ist die kognitive Kontrolle. Dieses Konstrukt lässt sich nur indirekt messen, nämlich mithilfe von kognitiven Tests: Die Leistung beim Test ist nicht die kognitive Kontrolle einer Person, sondern lediglich ein imperfekter **Indikator** hierfür. Wenn unterschiedliche Indikatoren von kognitiver Kontrolle stark miteinander korrelieren, ist es aber wahrscheinlicher, dass sich Befunde, die auf dem einen Indikator basieren, auch zu anderen Indikatoren generalisieren lassen.

Die Datei `poarch2018.csv` enthält Angaben zu zwei kognitiven Tests, von denen angenommen wird, dass sie Indikatoren von kognitiver Kontrolle sind: dem Flanker-Test (Eriksen & Eriksen, 1974) und dem Simon-Test (Simon, 1969). In beiden Tests müssen die Versuchspersonen manchmal irrelevante Informationen ignorieren. Die Daten stammen aus einer kleinen Studie von Poarch et al. (2019), in der den Probanden beide Tests vorgelegt wurden. Die Ergebnisse stellen dar, wie viel schneller die Versuchspersonen reagierten, wenn die irrelevante Information ‘kongruent’ mit der relevanten Information ist als, wenn die irrelevante Information ‘inkongruent’ mit der relevanten Information ist. Ausgedrückt werden die Angaben in Stimuli pro Sekunde; ein Wert von 0.5 heisst also, dass die Versuchsperson in einer kongruenten Testsituation 5 Stimuli mehr bewältigen kann pro 10 Sekunden als bei einer inkongruenten Testsituation.

1. Lesen Sie diesen Datensatz ein.
2. Stellen Sie den Zusammenhang zwischen den Variablen `Flanker` und `Simon` grafisch dar.
3. Berechnen Sie den Pearson-Korrelationskoeffizienten, insofern Sie dies für sinnvoll halten.
4. Berechnen Sie gegebenenfalls das 90%-Konfidenzintervall, und zwar sowohl anhand des Bootstraps als auch mit der t -Verteilung.
5. Fassen Sie Ihre Befunde kurz schriftlich zusammen. ◇

Bemerkung 8.17 (Konfidenzintervalle rekonstruieren). Wird um einen Korrelationskoeffizienten ein Konfidenzintervall mit der t -Verteilungsmethode konstruiert, so hängt das Ergebnis von nur drei Faktoren ab:

- ob das Konfidenzintervall ein 50%-, 80%-, 87%- usw.-Konfidenzintervall sein soll;
- dem Korrelationskoeffizienten selber;
- der Anzahl beobachteter Paare.

Wenn man weiss, was der Korrelationskoeffizient ist, und wie gross die Stichprobe war, kann man also selber das t -verteilungsbasierte Konfidenzintervall berechnen. Selber finde ich dies nützlich, wenn das Konfidenzintervall um r in einer Studie nicht berichtet wurde. Mit der

Funktion `r.test()` aus dem `psych`-Package ist dies ein Kinderspiel, allerdings werden nur 95%-Konfidenzintervalle berechnet. Gegebenenfalls müssen Sie das `psych`-Paket noch installieren.

```
psych::r.test(r12 = -0.80, n = 76)

Correlation tests
Call:psych::r.test(n = 76, r12 = -0.8)
Test of significance of a correlation
t value -11.47 with probability < 0
and confidence interval -0.87 -0.7
```

Übrigens: Mit der Notation `psych::r.test()` brauchen Sie das `psych`-Package nicht mit dem Befehl `library(psych)` zu laden. Dies ist nützlich, wenn man aus einem Paket eh nur eine Funktion braucht.

Das 95%-Konfidenzintervall eines Korrelationskoeffizienten von $r = -0.80$ in einer Stichprobe mit 76 Datenpunkten ist also $[-0.87, -0.70]$. Dabei gehen wir davon aus, dass die Stichprobe aus einer bivariaten Normalverteilung stammt.

Falls Sie lieber 80%- oder 90%-Konfidenzintervalle um Korrelationskoeffizienten berechnen, können Sie die unten stehende Funktion übernehmen. Sie kreiert mithilfe der `plot_r()`-Funktion aus dem `cannonball`-Package einen Datensatz mit den gewünschten Merkmalen und berechnet dann das Konfidenzintervall um den Korrelationskoeffizienten in diesem Datensatz. Auch die von dieser Funktion berechneten Konfidenzintervalle basieren auf der Annahme, dass die Daten aus einer bivariaten Normalverteilung stammen.

```
ci_r <- function(r, n, conf_level = 0.90) {
  dat <- cannonball::plot_r(r = r, n = n, showdata = 1, plot = FALSE)
  ci <- cor.test(dat$x, dat$y, conf.level = conf_level)$conf.int[1:2]
  ci
}

# 95%-Konfidenzintervall
ci_r(r = -0.80, n = 76, conf_level = 0.95)

[1] -0.8687618 -0.7009755

# 80%-Konfidenzintervall
ci_r(r = -0.80, n = 76, conf_level = 0.80)

[1] -0.8478924 -0.7391568

# 50%-Konfidenzintervall
ci_r(r = -0.80, n = 76, conf_level = 0.50)

[1] -0.8266792 -0.7697318
```

Bootstrapped Konfidenzintervalle können nur anhand der Daten selber rekonstruiert

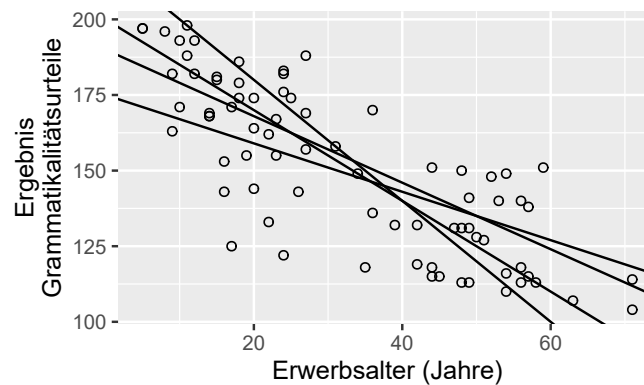


Abbildung 8.6: Wenn man von Hand eine gerade Linie durch die Punktwolke ziehen würde, zeichnet jeder die Linie wohl an einer anderen Stelle. Wir können aber Kriterien festlegen, die bewirken, dass alle die gleiche Linie zeichnen.

werden.



8.2 Antwort auf Frage 2: Regression

Es ist klar, dass es im Datensatz `dekeyser2010.csv` einen Zusammenhang zwischen AOA und GJT gibt. Eine senkende gerade Linie erfasst die Tendenz in den GJT-Daten schon ziemlich gut. Aber wie schaut diese Linie genau aus? Wir könnten zwar von Hand eine Gerade durch die Punktwolke ziehen, aber jeder zieht die Linie wohl an einer etwas anderen Stelle, siehe Abbildung 8.6. Eine prinzipiellere Herangehensweise wäre daher erwünscht.

Ähnlich wie im letzten Kapitel können wir eine Gleichung aufschreiben, die die Datenpunkte in zwei Teile zerlegt: einen systematischen Teil, der die Gemeinsamkeiten zwischen allen Werten ausdrückt, und einen unsystematischen Teil, der die individuellen Unterschiede zwischen diesen Gemeinsamkeiten und den Werten ausdrückt. Diesmal können wir den Zusammenhang zwischen AOA und GJT als eine Gemeinsamkeit im Datensatz betrachten. Dieser Zusammenhang scheint linear, weshalb er als eine gerade Linie modelliert werden kann:

$$\text{Wert} = \text{Gemeinsamkeit (inkl. AOA-Zusammenhang)} + \text{Abweichung.}$$

Eine gerade Linie wird definiert durch einen Schnittpunkt (β_0 ; dies ist der y -Wert, wenn $x = 0$) und eine Steigung (β_1 ; diese sagt, um wie viele Punkte y steigt, wenn x um eine Einheit erhöht wird); siehe Abbildung 8.7.

Egal, wie wir β_0 und β_1 wählen: Die Linie $y_i = \beta_0 + \beta_1 x_i$ wird die Daten nicht perfekt beschreiben: Es wird noch einen Restfehler (ϵ) geben. Jeder y -Wert (y_1, y_2 etc.) kann also umschrieben werden als die Kombination eines systematischen Teils ($\beta_0 + \beta_1 x_i$) und eines

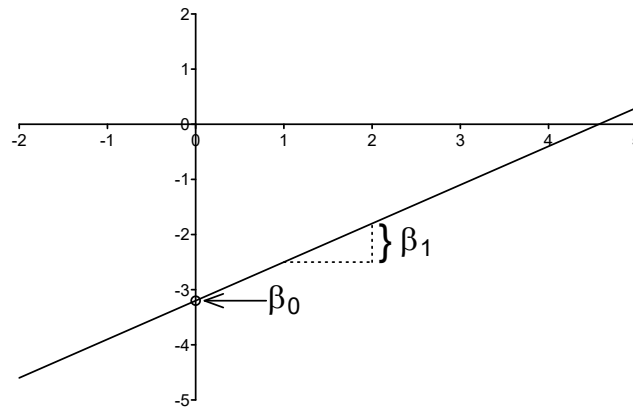


Abbildung 8.7: Schnittpunkt und Steigung einer Geraden.

Restfehlers:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad (8.3)$$

für $i = 1, \dots, n$. Diese mathematische Beschreibung ist ein **einfaches lineares Modell**: ‘einfach’, weil y nur eine Funktion einer (statt mehrerer) Variablen (x) ist, und ‘linear’, weil y als eine Summe (und nicht etwa ein Produkt oder etwas Komplexeres) verschiedener Terme modelliert wird. Der Begriff ‘einfach’ kontrastiert hier also mit ‘mehrfach’, nicht mit ‘schwierig’.

β_0 und β_1 sind die Parameter der einfachen Regressionsgleichung, und unsere nächste Aufgabe ist es, diese Parameter so gut wie möglich zu schätzen. Im Prinzip gibt es unendlich viele Möglichkeiten, β_0 und β_1 auszuwählen, sodass die Gleichung aufgeht (wie Abbildung 8.6 illustriert), aber uns interessieren nur die β_0 - und β_1 -Werte der optimalen Geraden. Um diese zu schätzen, müssen wir definieren, was ‘optimal’ in diesem Kontext heißt. Wie im letzten Kapitel besprochen wurde, ist das am meisten verwendete Optimierungskriterium die Methode der kleinsten Quadrate, wonach die optimale Linie jene Gerade ist, die die Summe der quadrierten Restfehler ($\sum_{i=1}^n \hat{\varepsilon}_i^2$) minimiert. Wir können diese Summe für unterschiedliche Kombinationen von $\hat{\beta}_0$ - und $\hat{\beta}_1$ -Werten berechnen.

- Sind $\hat{\beta}_0 = 185$ und $\hat{\beta}_1 = -2$, dann ist

$$\sum_{i=1}^{76} \hat{\varepsilon}_i^2 = \sum_{i=1}^{76} (y_i - (185 - 2x_i))^2 = 107121.$$

In R berechnet man dies so:

```
sum((d$GJT - (185 - 2*d$AOA))^2)
[1] 107121
```

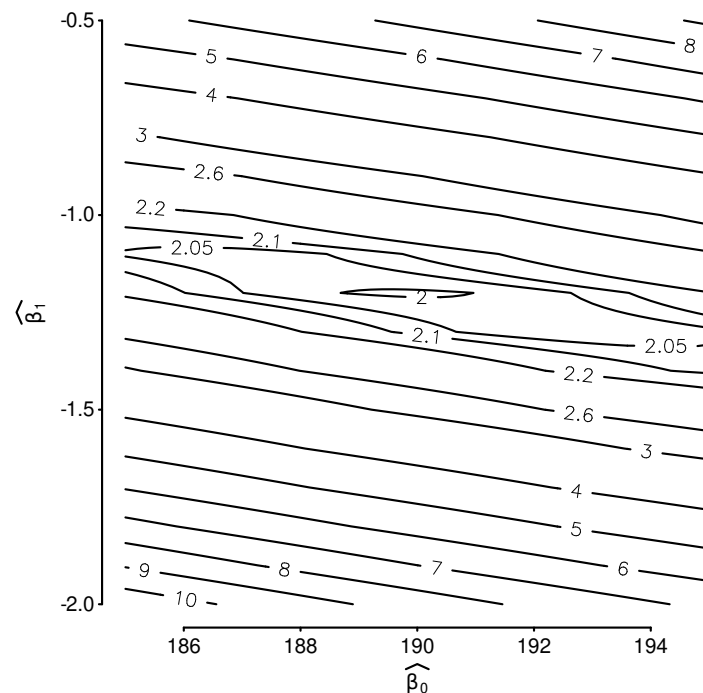


Abbildung 8.8: Die Summe der quadrierten Restfehler (geteilt durch 10'000) der GJT-Daten für unterschiedliche Parameterschätzungen. Diese Grafik kann wie eine topografische Karte gelesen werden; die Linien sind sozusagen Höhenlinien. Für Schnittpunkte nahe bei 190 und Steigungen nahe bei -1.2 wird diese Summe minimiert; bei diesen Koordinaten gibt es sozusagen einen Kessel.

- Sind $\hat{\beta}_0 = 190$ und $\hat{\beta}_1 = -1.5$, dann ist

$$\sum_{i=1}^{76} \hat{\varepsilon}_i^2 = \sum_{i=1}^{76} (y_i - (190 - 1.5x_i))^2 = 28810.$$

In R:

```
sum((d$GJT - (190 - 1.5*d$AOA))^2)
[1] 28810.25
```

$\hat{\beta} = (190, -1.5)$ ist also optimaler als $\hat{\beta} = (180, -2)$. Abbildung 8.8 zeigt die Summe der quadrierten Restfehler für unterschiedliche $(\hat{\beta}_0, \hat{\beta}_1)$ -Kombinationen; Kombinationen nahe bei $(190, -1.2)$ haben die kleinste Summe der quadrierten Restfehler.

In der Praxis ackert man nicht zig Parameterkombinationen durch, um die optimale zu finden.

Der Vorteil der Methode der kleinsten Quadrate ist, dass man die optimalen Parameterschätzungen schnell analytisch finden kann, und zwar so:

$$\begin{aligned}\widehat{\beta}_1 &= r_{xy} \frac{s_y}{s_x}, \\ \widehat{\beta}_0 &= \bar{y} - \widehat{\beta}_1 \bar{x}.\end{aligned}\tag{8.4}$$

Hier steht r_{xy} für die Pearsonkorrelation zwischen x und y in der Stichprobe. s_x, s_y stehen für die Stichprobenstandardabweichungen von x bzw. y . \bar{x} ist das Stichprobenmittel von x . Der Beweis für diese Formeln wird hier nicht reproduziert. Für die Daten von DeKeyser et al. (2010) sieht das Ergebnis dieser Berechnungen so aus:

```
beta1_hat <- cor(d$AOA, d$GJT) * sd(d$GJT) / sd(d$AOA)
beta1_hat

[1] -1.217977

beta0_hat <- mean(d$GJT) - beta1_hat * mean(d$AOA)
beta0_hat

[1] 190.4086
```

Einfacher geht es mit der `lm()`-Funktion:

```
aoa.lm <- lm(GJT ~ AOA, data = d)
aoa.lm

Call:
lm(formula = GJT ~ AOA, data = d)

Coefficients:
(Intercept)          AOA
    190.409         -1.218
```

(Intercept) ist hier $\widehat{\beta}_0$, also der Schnittpunkt der Regressionsgeraden mit der y -Achse. AOA ist $\widehat{\beta}_1$ und zeigt, um wie viele Einheiten die Gerade steigt oder senkt, wenn man entlang der AOA-Achse eine Einheit nach rechts geht.

Bemerkung 8.18 (andere Optimalitätskriterien). Neben der Methode der kleinsten Quadrate gibt es weitere Methoden, um die Regressionsparameter zu schätzen. Eine Alternative ist, β_0, β_1 mit der Methode der kleinsten absoluten Abweichungen zu schätzen, also derart, dass $\sum_{i=1}^n |\varepsilon_i|$ minimiert wird (**Median-Regression**). Es gibt auch Methoden, in denen gleichzeitig versucht wird, die quadrierten Abweichungen und die Grösse der β -Schätzungen zu minimieren. Je nach dem genauen Kriterium spricht man hier von **Lasso-Regularisierung**, **Ridge-Regularisierung** oder vom **elastischen Netz**. Die Grundidee bei diesen drei Ansätzen ist, die β -Parameter absichtlich mit einer gewissen Verzerrung zu schätzen, dabei aber die Varianz der Schätzungen

zu senken. ◇

Aufgabe 8.19. Führen Sie folgende Analyse auf die `dekeyser2010.csv`-Daten aus:

```
plot(AOA ~ GJT, data = d)
gjt.lm <- lm(AOA ~ GJT, data = d)
summary(gjt.lm)
```

1. Erklären Sie, was Sie gerade berechnet haben. Wieso ist das Intercept so gross? Was bedeuten die geschätzten Parameter? ◇
2. Welches Modell finden Sie am sinnvollsten: `aoa.lm` oder `gjt.lm`? Warum? ◇

Aufgabe 8.20. Die Datei `vanhove2014_cognates.csv` enthält eine Zusammenfassung der Daten meiner Dissertation (Vanhove, 2014). 163 Deutschschweizer Versuchspersonen wurden gebeten, 45 geschriebene und 45 (andere) gesprochene schwedische Wörter ins Deutsche zu übersetzen. Die Anzahl richtiger Antworten steht in den Spalten `CorrectWritten` für geschriebene Wörter bzw. `CorrectSpoken` für gesprochene Wörter. Die Datei `vanhove2014_background.csv` enthält Angaben zur Leistung der Versuchspersonen bei weiteren Sprach- und kognitiven Tests. Fügen Sie die beiden Datensätze zusammen. Beantworten Sie dann folgende Fragen. Ob Sie dies mit Korrelationsanalyse, Regressionsanalyse oder auf eine andere Art machen, liegt in Ihrem Ermessen.

1. `DS.Span` enthält die Leistung der Versuchspersonen bei einem Arbeitsgedächtnistest. Wie hängt die Leistung bei `DS.Span` mit `CorrectSpoken` zusammen? ◇
2. Wie hängt die Leistung bei einem Englischtest (`English.Overall`) mit der Übersetzungsleistung in der geschriebenen Modalität zusammen? ◇
3. Wie variiert die Übersetzungsleistung in den beiden Modalitäten mit dem Alter (`Age`) der Versuchspersonen? ◇

8.3 Regressionsgeraden zeichnen

Das Ergebnis einer Regressionsanalyse kann auch grafisch dargestellt werden, indem man dem Streudiagramm die Regressionsgerade hinzufügt (Abbildung 8.9):

```
ggplot(data = d,
       aes(x = AOA,
          y = GJT)) +
  geom_point(shape = 1) +
  geom_abline(intercept = 190.41, slope = -1.218)
```

Eine alternative Methode ist die folgende. Wenn man bei `geom_smooth()` als Methode "lm" einstellt, wird das Regressionsmodell von `ggplot()` berechnet. Im Code unten habe ich den Parameter `se` auf `FALSE` gestellt; weiter unten wird klar warum.

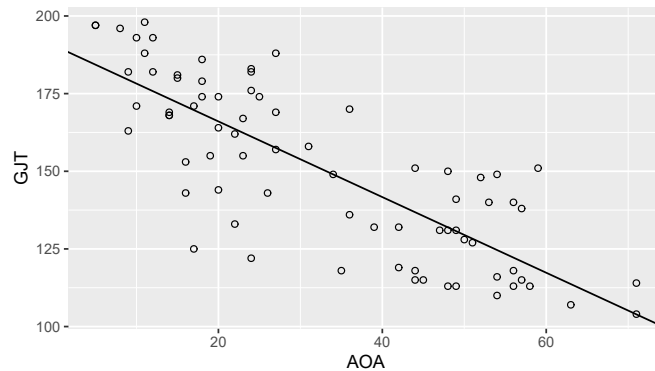


Abbildung 8.9: Ein Streudiagramm mit einer Regressionsgeraden. Die Regressionsgerade erfasst die modellierte zentrale Tendenz (genauer: das GJT-Mittel) für die unterschiedlichen AOA-Werte.

```
# Nicht gezeichnet
ggplot(data = d,
       aes(x = AOA, y = GJT)) +
  geom_point(shape = 1) +
  geom_smooth(method = "lm", se = FALSE)
```

Aber was stellt diese Gerade genau dar? Mit dem Regressionsmodell versuchten wir die GJT-Werte (y_i) als eine Funktion der AOA-Werte (x_i) und eines Restfehlers (ε_i) zu modellieren:

$$y_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + \hat{\varepsilon}_i, \quad i = 1, \dots, n.$$

Auf der Regressionsgeraden ($\hat{\beta}_0 + \hat{\beta}_1 x_i$) liegen die y_i -Werte abzüglich des Restfehlers, also

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i, \quad i = 1, \dots, n.$$

Wie man diese \hat{y}_i -Werte konzeptuell interpretieren kann, ist einfacher zu erklären, wenn wir uns zuerst anschauen, wie man die Unsicherheit in den Parameterschätzungen quantifizieren kann. Die Parameterschätzungen $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1)$ werden aufgrund des Stichprobenfehlers von Stichprobe zu Stichprobe mehr oder weniger voneinander abweichen. Da $\hat{\beta}$ aus Schätzungen besteht, ist auch die Regressionsgerade nur eine Schätzung. Um die Unsicherheit in den Parameterschätzungen und in der Regressionsgeraden zu quantifizieren, können wir uns wiederum auf den Bootstrap oder auf analytische Methoden verlassen.

8.3.1 Unsicherheit in Parameterschätzungen schätzen

Mit dem semi-parametrischen Bootstrap

Das Bootstrappen eines linearen Modells mit einem Prädiktor verläuft analog zum Bootstrappen eines linearen Modells ohne Prädiktor. Zuerst wird hier die Methode aus Abschnitt 7.4.1 (semi-

parametrischer Bootstrap) aufs lineare Modell mit einem Prädiktor angewandt:

1. Man berechnet $\hat{\beta}$ (also $\hat{\beta}_0$ und $\hat{\beta}_1$) und erhält dazu auch noch $\hat{\varepsilon}$.
2. Man zieht eine Bootstrap-Stichprobe aus $\hat{\varepsilon}$. Nenne diese $\hat{\varepsilon}^*$.
3. Man kombiniert $\hat{\beta}_0$, $\hat{\beta}_1 x_i$ und $\hat{\varepsilon}^*$. Dies ergibt eine neue Reihe von y -Werten: $y_i^* = \hat{\beta}_0 + \hat{\beta}_1 x_i + \hat{\varepsilon}_i^*$, $i = 1, \dots, n$.
4. Auf der Basis von y_i^* wird $\hat{\beta}$ erneut geschätzt.
5. Schritte 2–4 werden ein paar tausend Mal ausgeführt, sodass man die Verteilung der gebootstrappten β -Schätzungen erhält.

In R-Code:

```
runs <- 20000
residuals <- resid(aoa.lm)
n_obs <- length(residuals)
predictions <- predict(aoa.lm)

# 20'000 Bootstrapschätzungen für 2 Parameter
bs_beta <- matrix(nrow = runs, ncol = 2)

for (i in 1:runs) {
  bs_residuals <- sample(residuals, n_obs, replace = TRUE)
  bs_GJT <- predictions + bs_residuals
  resampled.lm <- lm(bs_GJT ~ d$AOA)
  bs_beta[i, ] <- coef(resampled.lm)
}

head(bs_beta)

      [,1]      [,2]
[1,] 191.8229 -1.289491
[2,] 184.2666 -1.053764
[3,] 187.7258 -1.161843
[4,] 185.9599 -1.181903
[5,] 197.2874 -1.446598
[6,] 182.4883 -1.085926
```

Die Verteilungen der Bootstrapschätzungen können wie gehabt mit Histogrammen gezeichnet werden; siehe Abbildung 8.10.

```
# Ergebnisse in tibble gießen
bs_beta_tbl <- tibble(Schnittpunkt = bs_beta[, 1],
                     Steigung = bs_beta[, 2])
```

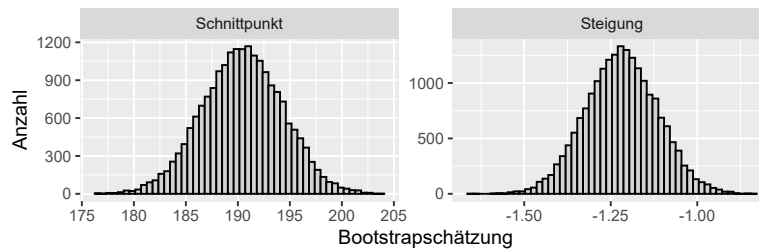


Abbildung 8.10: Verteilung der Bootstrap-Schätzungen der Parameter im Regressionsmodell `aoa.lm`.

```
bs_beta_tbl |>
  # Schätzungen alle in gleiche Spalte
  pivot_longer(cols = everything(),
               names_to = "Parameter",
               values_to = "Estimate") |>
  ggplot(aes(x = Estimate)) +
  geom_histogram(fill = "lightgrey", col = "black", bins = 50) +
  # facet_wrap zeichnet separate Grafiken je nach (hier) Parameter
  facet_wrap(vars(Parameter), scales = "free") +
  xlab("Bootstrapschätzung") +
  ylab("Anzahl")
```

Da diese Verteilungen normalverteilt aussehen, können die Konfidenzintervalle (hier: 90%) sowohl mit der `quantile()`- als auch mit der `qnorm()`-Funktion berechnet werden; die Ergebnisse sind einander nahezu identisch.

```
# Perzentilmethode
quantile(bs_beta_tbl$Schnittpunkt, probs = c(0.05, 0.95)) # Schnittpunkt

      5%      95%
184.0892 196.7159

quantile(bs_beta_tbl$Steigung, probs = c(0.05, 0.95)) # Steigung

      5%      95%
-1.388537 -1.049464

# Kürzer mit apply(). Die '2' heisst, dass die Funktion
# 'quantile' pro Spalte und nicht pro Zeile ausgeführt werden soll.
apply(bs_beta, 2, quantile, probs = c(0.05, 0.95))

      [,1]      [,2]
5%  184.0892 -1.388537
95%  196.7159 -1.049464
```


`coef(aoa.lm)` gibt zwei Werte aus: die Schätzung des Schnittpunkts und die Schätzung der Steigung:

```
coef(aoa.lm)

(Intercept)      AOA
190.408634    -1.217977
```

Um den ersten Wert zu selektieren, kann auch `coef(aoa.lm)[1]` verwendet werden, daher:

```
# qnorm
coef(aoa.lm)[1] + qnorm(c(0.05, 0.95)) * sd(bs_beta_tbl$Schnittpunkt)

[1] 184.0646 196.7527

coef(aoa.lm)[2] + qnorm(c(0.05, 0.95)) * sd(bs_beta_tbl$Steigung)

[1] -1.388678 -1.047275
```

Die Standardabweichungen der Bootstrapschätzungen schätzen den relevanten Standardfehler:

```
apply(bs_beta, 2, sd)

[1] 3.856910 0.103779
```

Die Bootstrapschätzungen können auch verwendet werden, um die Unsicherheit in der Regressionsgeraden grafisch darzustellen. Wir können zum Beispiel die Bootstrapschätzung des Schnittpunktes und der Steigung abrufen und anhand derer Regressionsgeraden zeichnen. Der Übersichtlichkeit halber werden hier nur die ersten vier Bootstrapschätzungen gezeigt. Bei Ihnen werden diese aufgrund der Zufälligkeit im Bootstrap natürlich anders aussehen. Die entsprechenden Regressionsgeraden sieht man in Abbildung 8.11:

```
head(bs_beta, 4)

      [,1]      [,2]
[1,] 191.8229 -1.289491
[2,] 184.2666 -1.053764
[3,] 187.7258 -1.161843
[4,] 185.9599 -1.181903
```

In Abbildung 8.12 mache ich nochmals das Gleiche, aber diesmal mit 100 Schätzungen. Für die Interessierten zeige ich diesmal auch den Code. Wie Sie sehen können, kommen die Regressionsgeraden für durchschnittliche AOA-Werte einander ziemlich nahe, aber nahe den Minimum- und Maximumwerten fächern sie sich auf.

```
plot_konfidenzband <- ggplot(data = d,
                             aes(x = AOA, y = GJT)) +
  geom_point(shape = 1)
```

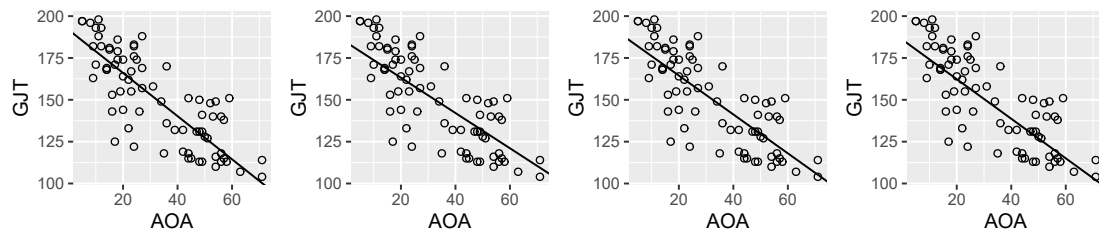


Abbildung 8.11: Regressionsgeraden, die anhand der 1., 2., 3. und 4. Bootstrapschätzungen des Schnittpunktes und der Steigung gezeichnet wurden. Die Grafiken sind einander recht ähnlich, aber nicht identisch.

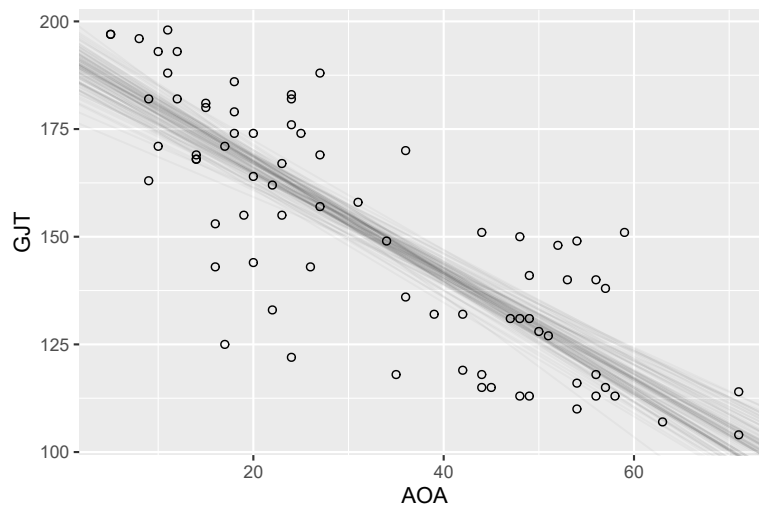


Abbildung 8.12: Regressionsgeraden, die auf der Basis von 100 Bootstrapschätzungen des Schnittpunktes und der Steigung gezeichnet wurden.

```
for (i in 1:100) {
  plot_konfidenzband <- plot_konfidenzband +
    geom_abline(intercept = bs_beta[i, 1],
               slope = bs_beta[i, 2],
               alpha = 1/30)
}
plot_konfidenzband
```

Statt diese Regressionsgeraden einzeln darzustellen, färbt man in der Regel das Band, in das diese Geraden mehrheitlich fallen, ein. Analog zum Konfidenzintervall nennt man dieses dann ein **Konfidenzband**. Die nächste Bemerkung erklärt, wie man Konfidenzbänder mittels Bootstrapping konstruieren kann, aber diese können Sie gerne überspringen.

Bemerkung 8.21 (Konfidenzbänder mit dem Bootstrap konstruieren). Die Idee ist die folgende.

Man definiert eine Reihe von x -Werten, an denen man das Konfidenzband zeichnen möchte. In unserem Fall handelt es sich einfach um eine Handvoll Zahlen zwischen dem AOA-Minimum und dem AOA-Maximum. Es spielt hier keine grosse Rolle, wie viele Werte man festlegt.

```
neue_aoa <- seq(from = min(d$AOA), to = max(d$AOA), by = 1)
```

Man nimmt die Bootstrapschätzungen des Schnittpunkts ($\hat{\beta}_0^*$) und der Steigung ($\hat{\beta}_1^*$) und man berechnet für jedes Paar von Schätzungen den \hat{y} -Wert für jeden x -Wert. Zum Beispiel ist (in meinem Fall) das erste Paar Bootstrapschätzungen:

```
bs_beta[1, ]
[1] 191.822911 -1.289491
```

Der Vektor von \hat{y} -Werten für dieses Paar von Bootstrapschätzungen ist daher:

```
# 191.82 + (-1.29) * 5, 191.82 + (-1.29) * 6, usw.
bs_beta[1, 1] + bs_beta[1, 2] * neue_aoa

[1] 185.3755 184.0860 182.7965 181.5070 180.2175 178.9280
[7] 177.6385 176.3490 175.0595 173.7700 172.4805 171.1910
[13] 169.9016 168.6121 167.3226 166.0331 164.7436 163.4541
[19] 162.1646 160.8751 159.5856 158.2961 157.0066 155.7172
[25] 154.4277 153.1382 151.8487 150.5592 149.2697 147.9802
[31] 146.6907 145.4012 144.1117 142.8222 141.5327 140.2433
[37] 138.9538 137.6643 136.3748 135.0853 133.7958 132.5063
[43] 131.2168 129.9273 128.6378 127.3483 126.0589 124.7694
[49] 123.4799 122.1904 120.9009 119.6114 118.3219 117.0324
[55] 115.7429 114.4534 113.1639 111.8744 110.5850 109.2955
[61] 108.0060 106.7165 105.4270 104.1375 102.8480 101.5585
[67] 100.2690
```

Aufgrund der Zufälligkeit, die dem Bootstrap inhärent ist, wird das Ergebnis bei Ihnen natürlich anders aussehen. Diese Übung machen wir für alle Paare von Bootstrapschätzungen – in unserem Fall also 20'000 Mal. Mit einer *for*-Schleife kann man dies übersichtlich tun. Da das Ergebnis jeder Iteration aber einen Vektor mit so vielen Elementen wie (hier) `neue_aoa` ist, ist es praktischer, diese Werte in einer Matrix zu speichern als in 20'000 Vektoren. Diese Schritte kann man auch mit Matrizenalgebra ausführen, aber ich vermute, dass ein *for*-Schleife das Verfahren transparenter macht.

```
bs_y_hat <- matrix(nrow = runs,
                   ncol = length(neue_aoa))

for (i in 1:runs) {
  bs_y_hat[i, ] <- bs_beta[i, 1] + bs_beta[i, 2]*neue_aoa
}
```

Sie können diese Matrix mit etwa `head(bs_y_hat)` inspizieren. Um das 95%-Konfidenzband zu konstruieren, schlagen wir nun das 2.5. und das 97.5. Perzentil jeder Spalte nach. Die 2.5. Perzentile bilden die untere Grenze des Konfidenzbandes; die 97.5. die obere. Dazu verwende ich hier die `apply()`-Funktion, mit der man eine Funktion (hier `quantile()`) mit dem Zusatzparameter `probs = 0.025`) bequem auf alle Spalten oder Zeilen einer Matrix (hier `bs_y_hat`) evaluieren kann. Die Zahl 2 spezifiziert, dass die Funktion pro Spalte evaluiert werden soll; 1 hiesse, dass sie pro Zeile zu evaluieren ist.

```
unten_95 <- apply(bs_y_hat, 2, quantile, probs = 0.025)
oben_95 <- apply(bs_y_hat, 2, quantile, probs = 0.975)
```

Das 80%-Konfidenzband würde man so konstruieren:

```
unten_80 <- apply(bs_y_hat, 2, quantile, probs = 0.10)
oben_80 <- apply(bs_y_hat, 2, quantile, probs = 0.90)
```

Man kann auch noch das Mittel jeder Spalte berechnen. Dies ergibt ungefähr die Regressionsgerade. Es ist jedoch sinnvoller, den genauen Wert der Regressionsgerade zu verwenden.

```
gerade <- predict(aoa.lm, newdata = data.frame(AOA = neue_aoa))
```

Wenn man die Grafik mit `ggplot()` zeichnen möchte, muss man diese Werte noch in ein tibble giessen:

```
konfidenzband_tbl <- tibble(neue_aoa, gerade,
                           unten_95, oben_95,
                           unten_80, oben_80)
```

Zeichnen kann man das Konfidenzband dann so; siehe Abbildung 8.13.

```
ggplot(data = konfidenzband_tbl,
       aes(x = neue_aoa)) +
  # 95% Konfidenzband leicht
  geom_ribbon(aes(ymin = unten_95,
                 ymax = oben_95),
            fill = "lightgrey") +
  # 80% Konfidenzband dunkel
  geom_ribbon(aes(ymin = unten_80,
                 ymax = oben_80),
            fill = "darkgrey") +
  # Regressionsgerade
  geom_line(aes(y = gerade)) +
  # ev. auch noch die Rohdaten plotten.
  # Diese stehen in einem anderen tibble.
  geom_point(data = d,
            aes(x = AOA, y = GJT),
```

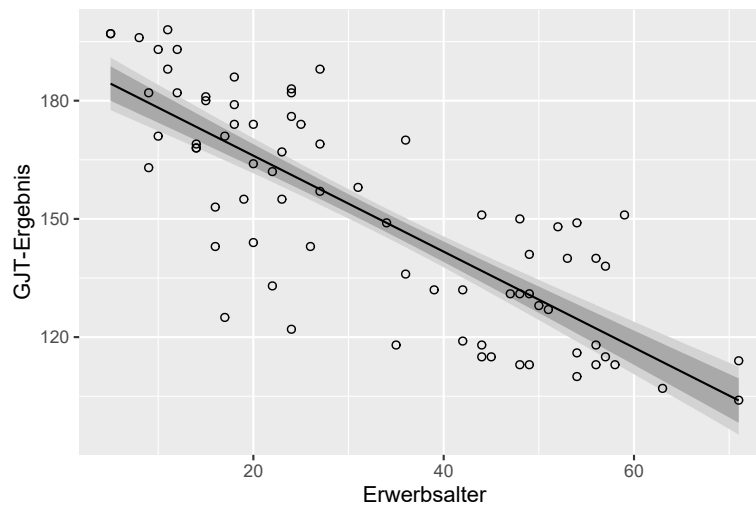


Abbildung 8.13: Regressionsgerade mit 80%- und 95%-Konfidenzbändern, die mittels Bootstrapping berechnet wurden.

```

shape = 1) +
xlab("Erwerbsalter") +
ylab("GJT-Ergebnis")

```



Bemerkung 8.22 (identisch und unabhängig verteilte Restfehler). Beim Einschätzen der Unsicherheit in den Parameterschätzungen und in der Regressionsgeraden haben wir eine grundlegende Annahme gemacht, die bisher noch nicht diskutiert wurde. Beim Bootstrappen haben wir auf der Basis der beobachteten Residuen zufällig neue Vektoren mit Residuen ($\hat{\epsilon}^*$) generiert und diese dann mit den \hat{y} -Werten kombiniert. Dieser Schritt ist nur verteidigbar, wenn zwei Bedingungen gleichzeitig erfüllt sind:

1. Die Verteilung des Restfehlers, inklusive seine Streuung, ist für alle \hat{y} -Werte gleich ('identisch verteilte Restfehler', 'Homoskedastizitätsannahme'). Zum Beispiel sollte es genauso plausibel sein, dass ein Restfehler von 25 auftaucht, wenn der \hat{y} -Wert 120 ist als wenn er 180 ist. Sonst wäre es ja nicht sinnvoll gewesen, die Restfehler beim Bootstrappen komplett zufällig durcheinander zu werfen.
2. Die Restfehler bilden keine Klumpen. Anders gesagt, wenn wir den Restfehler einer bestimmten Beobachtung kennen, liefert uns dies nicht mehr Informationen über gewisse weitere Restfehler als über andere ('unabhängig verteilte Restfehler', 'Unabhängigkeitsannahme'). Wiederum wäre es sonst ja nicht sinnvoll gewesen, die Restfehler komplett zufällig durcheinander zu werfen, sondern hätten wir die Restfehler gruppchenweise neu zuordnen müssen.

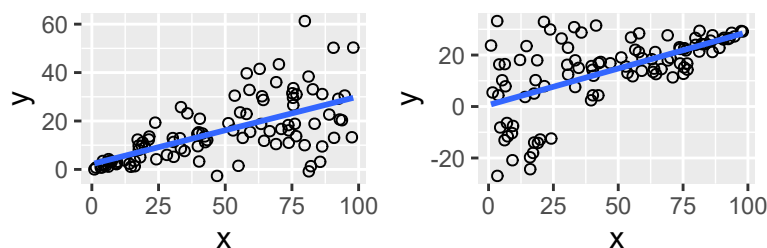


Abbildung 8.14: Die Streuung in beiden Streudiagrammen variiert erheblich je nach dem x - oder \hat{y} -Wert.

Ein paar Beispiele, um diese Bedingungen anschaulicher zu machen:

- Es kann durchaus vorkommen, dass die Streuung um die Regressionsgerade systematisch zu- oder abnimmt für grössere \hat{y} -Werte. Abbildung 8.14 zeigt zwei klare Beispiele.
- Jemand möchte die durchschnittliche Länge von [i]-Produktionen von Bernern schätzen und wählt zufällig 25 Berner aus. (So weit, so gut.) Jeder Sprecher liest 50 Wörter mit einem [i:] vor. Die Vokallängen eines beliebigen Sprechers sind sich aber denkbar ähnlicher als die Vokallängen unterschiedlicher Sprecher: Manche Sprecher werden eher überdurchschnittlich lange Vokale produzieren, manche eher unterdurchschnittlich. Die Restfehler ('über-/unterdurchschnittlich') der einzelnen Produktionen sind also nicht unabhängig voneinander, sondern bilden pro Sprecher Klumpen.
- Ausserdem ist es wahrscheinlich, dass das [i:] in bestimmten phonologischen Kontexten unterschiedlich schnell ausgesprochen wird. Die Restfehler bilden also auch pro phonologischen Kontext (oder pro Wort) Klumpen.

Wenn die sogenannte '**i.i.d.-Bedingung** (*identically and independently distributed*) nicht erfüllt ist, dann ist es möglich, dass es effizientere Arten und Weisen gibt, um die Modellparameter zu schätzen. Damit ist Folgendes gemeint: Wenn Regressionsparameter mit der Methode der kleinsten Quadrate geschätzt werden, dann sind diese Schätzungen weder tendenziell Überschätzungen noch tendenziell Unterschätzungen—die Schätzungen sind also unverzerrt. Es gibt aber auch andere Methoden, um diese Parameter unverzerrt zu schätzen. Wenn die 'i.i.d.-Bedingung erfüllt, ist es ausserdem so, dass die Methode der kleinsten Quadrate Schätzungen liefert, die von Stichprobe zu Stichprobe am wenigsten voneinander abweichen. Ist die 'i.i.d.-Bedingung nicht erfüllt, dann ist es möglich, dass eine andere unverzerrte Schätzungsmethode Schätzungen liefert, die von Stichprobe zu Stichprobe weniger variieren. Siehe hierzu noch den Satz von Gauss (Satz 5.8 auf Seite 111) und die darauf folgende Diskussion (Beispiele und Aufgabe).

Wichtiger ist aber, dass die Schätzung der Unsicherheit betroffen ist. Insbesondere bei einer Verletzung der Unabhängigkeitsannahme wird die Unsicherheit in den Parameterschätzungen in der Regel unterschätzt. Dies gilt nicht nur beim Bootstrappen, sondern auch beim Verwenden des zentralen Grenzwertsatzes oder von t -Verteilungen. Im Beispiel mit den [i:] könnten wir

also nicht den Standardfehler der durchschnittlichen Vokallänge berechnen, indem wir die Streuung in den Produktionen teilen durch die Wurzel von 1'250 ($25 \cdot 50$).

Typische Verletzungen der Unabhängigkeitsannahme können mit sog. **gemischten Modellen** behoben werden; siehe Kapitel ?? für Literaturvorschläge. Für Verletzungen der Homoskedastizitätsannahme bietet sich eine andere Schätzungsmethode als *ordinary least squares*, die sog. *generalised least squares* (siehe Zuur et al., 2009). Eine alternative Lösung ist, den Bootstrap so durchzuführen, dass die Abhängigkeitsstruktur der Daten beim Bootstrappen respektiert wird (siehe Efron & Tibshirani, 1993, Abschnitt 9.5). ◇

Bemerkung 8.23 (punktweise vs. simultane Konfidenzbänder). Die hier besprochenen Konfidenzbänder sind sog. **punktweise Konfidenzbänder**. Ein gut kalibriertes punktweises $100(1 - \alpha)\%$ -Konfidenzband stellt für jeden x -Wert ein $(100(1 - \alpha)\%$ -Konfidenzintervall für den entsprechenden Wert $\beta_0 + \beta_1 x$ dar.

Ab und zu trifft man auch **simultane Konfidenzbänder** an. Ein simultanes $100(1 - \alpha)\%$ Konfidenzband sollte für eine Menge von x -Werten x_1, x_2, \dots, x_n garantieren, dass das Konfidenzband zu einer Wahrscheinlichkeit von mindestens $100(1 - \alpha)\%$ *alle* entsprechenden $\beta_0 + \beta_1 x_i, i = 1, \dots, n$, umfasst. Dies ist eine viel strengere Bedingung als bei punktweisen Konfidenzbändern! Entsprechend sind simultane Konfidenzbänder breiter als punktweise Konfidenzbänder.

In diesem Skript werden nur punktweise Konfidenzbänder behandelt. Werden in der Literatur Konfidenzbänder ohne weitere Angaben berichtet, so sollten Sie davon ausgehen, dass es sich um die punktweise Variante handelt. ◇

Bootstrappen unter der Normalitätsannahme

Wenn wir annehmen wollen, dass die Residuen nicht nur identisch und unabhängig, sondern auch noch normalverteilt sind, können wir die $\hat{\varepsilon}^*$ -Vektoren auch mit der `rnorm()`-Funktion generieren. Das Vorgehen ist komplett analog zu der in Abschnitt 7.4.2 auf Seite 148 beschriebenen Methode. Unsere Annahmen können expliziter gemacht werden:

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_i + \varepsilon_i, \\ \varepsilon_i &\sim N(0, \sigma_\varepsilon^2), \end{aligned} \tag{8.5}$$

wobei die ε_i -Werte unabhängig sind für $i = 1, \dots, n$.

Der neue Teil $\varepsilon_i \sim N(0, \sigma_\varepsilon^2)$ macht klar, dass wir annehmen, dass die Restfehler aus einer Normalverteilung mit Mittel 0 und Varianz σ_ε^2 stammen. σ_ε^2 ist ein einziger (obgleich unbekannter) Wert, sodass klar ist, dass wir davon ausgehen, dass die Streuung der Residuen nicht von x (und somit \hat{y} abhängt). Sowohl die Unabhängigkeits- als auch die Homoskedastizitätsannahme werden von dieser Annahme umfasst.

σ_ε ist zwar unbekannt, aber wird anhand der Stichprobe geschätzt; siehe Gleichung 7.2 auf Seite 148. Der folgende Code zeigt, wie die Bootstrapschätzungen des Schnittpunkts und der Steigung berechnet werden können. Abgesehen von der Konstruktion der $\hat{\varepsilon}^*$ -Vektoren ist die

Herangehensweise aber identisch zu jener des Bootstraps ohne die Normalitätsannahme aus dem letzten Abschnitt, sodass die anderen Schritte hier nicht mehr wiederholt werden.

```
runs <- 20000

bs_beta <- matrix(nrow = runs, ncol = 2)
sigma_eps <- sigma(aoa.lm)
n_obs <- length(resid(aoa.lm))

for (i in 1:runs) {
  bs_residuals <- rnorm(n = n_obs, mean = 0, sd = sigma_eps)
  bs_GJT <- predict(aoa.lm) + bs_residuals
  resampled.lm <- lm(bs_GJT ~ d$AOA)
  bs_beta[i, ] <- coef(resampled.lm)
}
```

Wenn wir andere Annahmen über die Verteilung der Residuen treffen, können wir diese analog ins Bootstrapverfahren einbauen.

Mit *t*-Verteilungen

Wenn wir ohnehin davon ausgehen wollen, dass die Residuen (i.i.d.) normalverteilt sind, können wir den Standardfehler, die Konfidenzintervalle und das Konfidenzbänder auch algebraisch anhand der *t*-Verteilungen berechnen. Dadurch wird auch die Unterschätzung von σ_ε durch $\hat{\sigma}_\varepsilon$ mitberücksichtigt. Bei 76 Beobachtungen und bloss zwei Parameterschätzungen wird diese Unterschätzung aber kaum merkbar sein.

```
summary(aoa.lm)$coefficients
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	190.408634	3.9040275	48.77236	5.209293e-58
AOA	-1.217977	0.1051385	-11.58450	2.728150e-18

```
# 95%-Konfidenzintervalle nach t-Methode
confint(aoa.lm, level = 0.95)
```

	2.5 %	97.5 %
(Intercept)	182.62969	198.187578
AOA	-1.42747	-1.008484

Mit `geom_smooth()` können *t*-basierte Konfidenzbänder sofort gezeichnet werden, siehe Abbildung 8.15.

```
ggplot(data = d,
       aes(x = AOA,
          y = GJT)) +
  geom_point(shape = 1) +
```

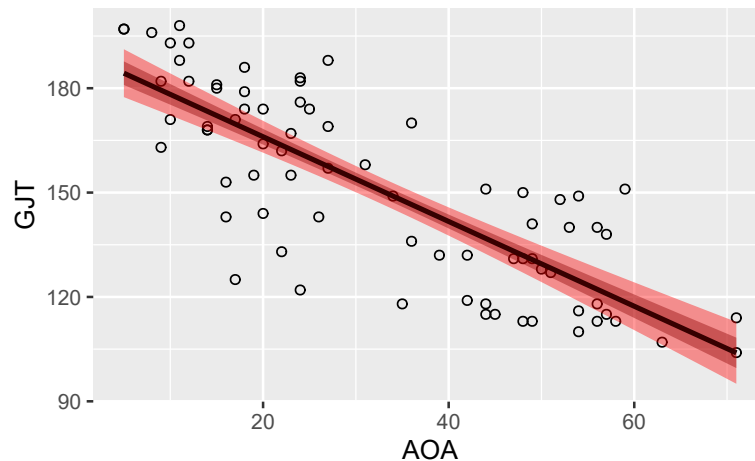



Abbildung 8.15: Regressionsgerade mit 67%- und 95%-Konfidenzbänder. Konfidenzbänder von Regressionsmodellen sind übrigens am schmalsten beim durchschnittlichen x -Wert.

```
geom_smooth(method = "lm", level = 0.95,
            fill = "red", col = "black") +
geom_smooth(method = "lm", level = 0.67,
            fill = "darkred", col = NA)
```

8.4 Regressionsgeraden interpretieren

Gleichung 8.5 auf Seite 182 ist nützlich, um die konzeptuelle Interpretation der Regressionsgeraden zu verstehen. Nach dieser Gleichung gehen wir davon aus, dass die Restfehler zufällig (und 'i.i.d.') aus einer Verteilung mit Mittel 0 stammen. Folglich liegen auf der Geraden $\beta_0 + \beta_1 x_i$ die **bedingten Erwartungswerte** der y -Verteilung gegeben x . Abbildung 8.16 stellt dieses Konzept grafisch dar.

Die Regressionsgerade stellt eine Schätzung dieser bedingten Erwartungswerte dar. Für einen festen x -Wert zeigt sie also eine Schätzung des Mittels der y -Verteilung für dieses x . Der Querschnitt des $100(1 - \alpha)\%$ -Konfidenzbands an einem bestimmten x -Wert stellt dementsprechend das $100(1 - \alpha)\%$ -Konfidenzintervall des y -Mittels für diesen x -Wert dar.

Beispiel 8.24 (unterschiedliche bedingte Mittel). Laut dem `aoa.lm`-Modell sind die geschätzten β -Parameter 190.4 und -1.22 . Laut dem Modell ist die beste Schätzung der durchschnittlichen (Mittel) GJT-Leistung von Versuchspersonen mit einem AOA von 15 also $190.4 - 1.22 \cdot 15 = 172.1$. Dieses Ergebnis erhält man auch mit `predict()`:

```
predict(aoa.lm, newdata = tibble(AOA = 15))
```

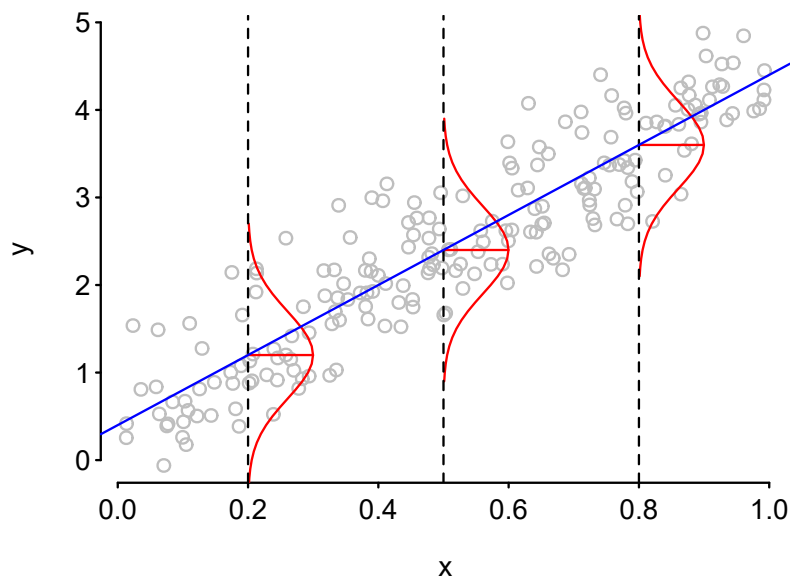


Abbildung 8.16: Wenn wir davon ausgehen, dass die Residuen i.i.d. verteilt sind, verbindet die Gerade die Mittel der y -Verteilungen für die unterschiedlichen x -Werte ('bedingter Erwartungswert'). In dieser Grafik sind die Residuen normalverteilt, aber dies ist keine Voraussetzung. Wenn die Residuen nicht normalverteilt sind, ist es jedoch möglich, dass das Mittel kein sehr relevantes Mass ist.

172.139

In unserem Datensatz gibt es zwei Versuchspersonen mit einem AOA von 15, aber ihr Durchschnittsergebnis ist nicht 172.1:

```
d |> filter(AOA == 15)

# A tibble: 2 x 2
  AOA  GJT
<dbl> <dbl>
1    15  180
2    15  181
```

Inwiefern unsere modellbasierte Schätzung eine zuverlässigere Schätzung des konditionellen Mittels darstellt als das Mittel dieser beiden Werte, hängt von der Gültigkeit unserer Annahmen ab. Die Annahme eines linearen Zusammenhangs scheint hier doch auf jeden Fall nicht wahnsinnig daneben zu liegen. Konzeptuell gesprochen erlaubt uns diese Annahme, y -Mittel für bestimmte x -Werte besser zu schätzen, indem wir auch Information über den x - y -Zusammenhang, die wir aus den restlichen Daten ableiten, mit einbeziehen. ◇

Bemerkung 8.25 (Intrapolation und Extrapolation). Versuchspersonen mit einem AOA von 21 gibt es in der Stichprobe nicht. Nach der Regressionsgleichung wäre aber das Durchschnittsergebnis von Versuchspersonen mit diesem AOA in der Population etwa 165 Punkte.

```
predict(aoa.lm, newdata = tibble(AOA = 21))

1
164.8311
```

Dies ist ein Beispiel von **Intrapolation**, denn es gibt sowohl Versuchspersonen mit niedrigeren als mit höheren AOA-Werten in der Stichprobe.

Versuchspersonen mit einem AOA von 82 gibt es in der Stichprobe auch nicht. Nach der Regressionsgleichung wäre aber das Durchschnittsergebnis von Versuchspersonen mit diesem AOA in der Population etwa 91 Punkte.

```
predict(aoa.lm, newdata = tibble(AOA = 82))

1
90.53455
```

Dies ist ein Beispiel von **Extrapolation**, denn der Maximum-AOA-Wert in der Stichprobe ist 71 Jahre.

Seien Sie vorsichtig mit Extrapolation: Wenn wir eine Stichprobe von Versuchspersonen zwischen 8 und 26 Jahren haben, ist es gefährlich, Aussagen über 5- oder 40-Jährige zu machen. Dies wird in der linken Grafik in Abbildung 8.17 illustriert: Eine Fähigkeit, die sich im Alter zwischen 10 und 35 entwickelt, hat nicht unbedingt die gleiche Entwicklung ausserhalb dieses

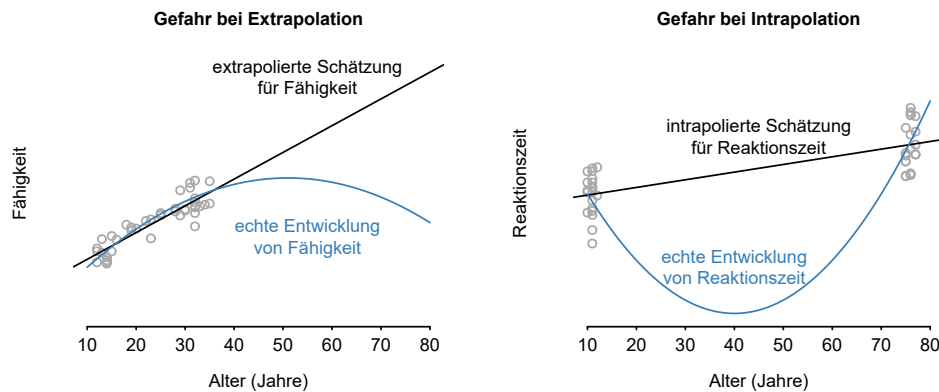


Abbildung 8.17: Die Gefahr bei Intrapolation und Extrapolation.

Bereichs. Eine Extrapolation auf der Basis der Regressionsgeraden ist hier irreführend. Auch bei Intrapolation ist Vorsicht geboten. Aus den Daten in der rechten Grafik könnte man zum Beispiel die Schlussfolgerung ziehen, dass sich Reaktionszeiten im Alter graduell verlängern. Auch diese Schlussfolgerung dürfte zu kurz greifen. ◇

Bemerkung 8.26. Das durchschnittliche (Mittel) AOA in der Stichprobe ist etwa 32.5 Jahre. Das geschätzte konditionelle GJT-Mittel für dieses Alter ist gleich dem Mittel der Stichprobe.

```
predict(aoa.lm, newdata = tibble(AOA = mean(d$AOA)))

1
150.7763
```

Dies ist natürlich kein Zufall, sondern ein allgemeines Phänomen. Wenn wir Gleichung 8.4 in die Regressionsgleichung einsetzen, erhalten wir ja Folgendes:

$$\hat{y}_i = \underbrace{\bar{y} - \hat{\beta}_1 \bar{x}}_{=\hat{\beta}_0} + \hat{\beta}_1 x_i.$$

Für $x_i = \bar{x}$ erhalten wir also $\hat{y}_i = \bar{y}$. ◇

Bemerkung 8.27. Konfidenzintervalle um konditionelle Mittel können mit den oben beschriebenen Bootstrappmethoden berechnet werden, indem man das Konfidenzband für nur einen x -Wert berechnet. Man kann aber auch `predict()` verwenden; dann wird das Konfidenzintervall auf der Basis der geeigneten t -Verteilung konstruiert.

```
predict(aoa.lm, newdata = tibble(AOA = 35),
        interval = "confidence", level = 0.80)

      fit      lwr      upr
1 147.7795 145.3246 150.2343
```



Bemerkung 8.28 (Schnittpunkt interpretierbarer machen). Der geschätzte Schnittpunkt hat nicht unbedingt eine nützliche Interpretation. In unserem Fall stellt er die geschätzte Durchschnittsleistung von Versuchspersonen mit AOA 0 dar. Solche gibt es in der Stichprobe nicht, sodass diese Zahl eine Art Extrapolation darstellt. Sie können den Schnittpunkt interpretierbarer machen, indem Sie das Stichprobenmittel des Prädiktors von den Prädiktorwerten abziehen und mit diesen neuen Werten arbeiten. Diese Technik heisst **zentrieren** (*centring*). Ein Vorteil des Zentrierens ist, dass der geschätzte Schnittpunkt einem jetzt sofort auch sagt, was das Stichprobenmittel der y -Variable ist.

```
# AOA zentrieren
d$c.AOA <- d$AOA - mean(d$AOA)

# Modell neu fitten
aoa.lm <- lm(GJT ~ c.AOA, data = d)

# Parameterschätzungen
summary(aoa.lm)$coefficients
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	150.776316	1.8807316	80.16897	1.120538e-73
c.AOA	-1.217977	0.1051385	-11.58450	2.728150e-18

Achten Sie aber darauf, dass eine Versuchsperson mit einem AOA von 35 jetzt für das Modell eine Versuchsperson mit einem c.AOA von $35 - \bar{x}_{AOA} = 2.46$ ist:

```
predict(aoa.lm, newdata = tibble(c.AOA = 35 - mean(d$AOA)),
        interval = "confidence", level = 0.80)
```

	fit	lwr	upr
1	147.7795	145.3246	150.2343



8.5 Modellannahmen überprüfen

Die Modellresiduen sollten grafisch dargestellt werden, um die Modellannahmen zu überprüfen. Leitfragen dabei sind unter anderem:

- Gibt es noch einen erkennbaren Zusammenhang zwischen den Residuen und den \hat{y} -Werten? Ein solcher Zusammenhang deutet darauf hin, dass der Zusammenhang zwischen einem oder mehreren Prädiktoren und dem outcome nicht-linear ist.
- Variiert die Streuung der Residuen mit \hat{y} oder mit den Prädiktoren? Systematische Unterschiede in der Streuung der Residuen deuten darauf hin, dass der Restfehler 'heteroskedastisch' ist.

- Sind die Residuen ungefähr normalverteilt? Nicht-normalverteilte Residuen lassen vermuten, dass die Annahme, dass der Restfehler aus einer Normalverteilung stammt, nicht stimmt. Dies hätte einerseits Konsequenzen für die auf t -Verteilungen basierten Konfidenzintervalle und Konfidenzbänder. Andererseits, und wichtiger, sind die bedingten Mittel, die die Regressionslinie darstellt, eventuell weniger relevant.
- Gibt es einzelne Datenpunkte, die einen viel stärkeren Einfluss aufs Regressionsmodell ausüben als die meisten? Das Problem mit einflussreichen Datenpunkten ist, dass sie etwa dazu führen können, dass das Modell einen leichten positiven Zusammenhang zwischen den Variablen findet, während für die meisten Datenpunkte ein starker negativer Zusammenhang vorliegt.

Abbildung 8.18 zeigt ein paar nützliche Grafiken, die man einfach mit `plot(aoa.lm)` generieren kann. Auf riesige Probleme in den Modellannahmen deuten diese Grafiken m.E. nicht hin. Solche Probleme würde man ohnehin nicht erwarten, wenn man sich das Streudiagramm am Anfang dieses Kapitels angeschaut hat. In meiner Erfahrung stösst man selten auf Überraschungen, wenn man die Daten bereits ausführlich grafisch dargestellt hat.

```
# 4 Grafiken in 2x2-Raster zeichnen.
# (Dies funktioniert nicht für ggplot!)
par(mfrow = c(2, 2))
# Modelldiagnosen darstellen
plot(aoa.lm)
# Ab jetzt wieder normal zeichnen.
par(mfrow = c(1, 1))
```

Aufgrund des Stichprobenfehlers wird man oft – rein durch Zufall – Zusammenhänge und nicht-normalverteilte Residuen finden, sodass man ein bisschen Erfahrung braucht, um unbedeutende Muster in den Residuen von potenziellen Problemen zu unterscheiden. Ausserdem sind Modellannahmen gerade bei kleineren Stichproben schwieriger zu überprüfen. Für mehr Informationen hierzu, siehe *Checking model assumptions without getting paranoid* (25.4.2018) und Vanhove (2018). Siehe ausserdem *Before worrying about model assumptions, think about model relevance* (11.4.2019).

Das Thema Modellkritik wird weiter behandelt von unter anderem Baayen (2008), Cohen et al. (2003, Kapitel 4), Faraway (2005, Kapitel 4), Weisberg (2005, Kapitel 8–9) und Zuur et al. (2009, Kapitel 2). Statt sich zu sehr in technischen Details zu verlieren, halte ich es aber sinnvoller, sich stets die Relevanzfrage zu stellen (vgl. Blogeintrag 11.4.2019).

Aufgabe 8.29. Obwohl die Modelldiagnose nicht auf grössere Probleme hindeutet, können die Modellannahmen eigentlich gar nicht stimmen. Erklären Sie. ◇

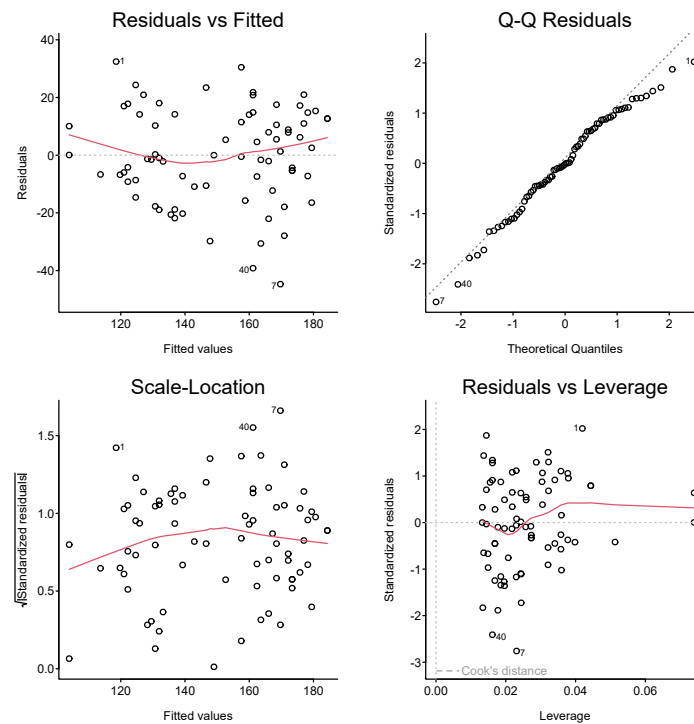


Abbildung 8.18: Grafische Modelldiagnose des `aoa.lm`-Modells mithilfe der `plot()`-Funktion. *Links oben:* Zusammenhang zwischen Residuen und \hat{y} . Die rote Trendlinie sollte ungefähr flach sein. Sonstige auffällige Muster wären auch unerwünscht. *Rechts oben:* Normalität der Residuen. Wenn die Residuen normalverteilt sind, liegen sie grundsätzlich auf der gestrichelten Diagonale. *Links unten:* Streuung in den Residuen. Eine flache rote Trendlinie deutet auf Homoskedastizität hin. *Rechts unten:* Manchmal gibt es in dieser Grafik ein paar gestrichelte rote Linien. Datenpunkte, die jenseits dieser Linien liegen, dürften viel einflussreicher als andere Datenpunkte sein. Bei allen Grafiken ist jedoch zu bemerken, dass die Muster aufgrund des Stichprobenfehlers unerwünscht aussehen können, auch wenn die Annahmen tatsächlich berechnigt sind.

Kapitel 9

Gruppenunterschiede

Im vorigen Kapitel haben wir uns mit der Frage beschäftigt, wie man den Zusammenhang zwischen einem kontinuierlichen Prädiktor und einem kontinuierlichen outcome modellieren kann. In diesem Kapitel widmen wir uns der Frage, wie Zusammenhänge zwischen **kategorischen** Prädiktoren und einem kontinuierlichen outcome modelliert werden können. Das typische Beispiel eines kategorischen Prädiktors ist die Konditionzugehörigkeit in einem Experiment: Wie stark unterscheiden sich die Ergebnisse von Teilnehmenden in der Experimentalgruppe im Schnitt von jenen von Teilnehmenden in der Kontrollgruppe? Das Vorgehen ist nahezu identisch mit dem aus dem letzten Kapitel.

9.1 Unterschiede zwischen zwei Gruppen

Das Beispiel, dem wir uns hier widmen, stammt nicht aus der Sprachwissenschaft, sondern aus der Sozialpsychologie. Caruso et al. (2013, Experiment 1) berichteten, dass amerikanische Versuchspersonen Aussagen, die das US-amerikanische Sozialsystem rechtfertigen, stärker zustimmen, wenn man sie an Geld erinnert (sogenanntes *currency priming*). Ihr Design sah wie folgt aus. Es gab acht Aussagen im Stil von *Everyone has a fair shot at wealth and happiness*. Die Teilnehmenden deuteten am Bildschirm ihre Zustimmung zu diesen Aussagen auf einer 7-stufigen Likertskala an (1 = überhaupt nicht einverstanden, 7 = vollständig einverstanden). Pro Versuchsperson wurden die acht Zustimmungswerte gemittelt. Die Hälfte der Versuchspersonen sah im Hintergrund ein verblasstes aber ersichtliches Bild einer Banknote; bei der anderen Hälfte war dieses Bild verwischt. Die Versuchspersonen, welche die Banknote im Hintergrund sahen, stimmten den Aussagen stärker zu, als jene, bei denen das Bild verwischt war. Klein et al. (2014) versuchten, dieses Ergebnis in 36 neuen Stichproben zu replizieren. In der Datei `Klein2014_money_abington.csv` finden Sie die Ergebnisse einer dieser Stichproben (84 Teilnehmende). Diese Daten werden wir analysieren.

Aufgabe 9.1. Lesen Sie diese Datei in R ein. Den Datensatz können Sie einfach `d` nennen. Vergessen Sie nicht zu kontrollieren, ob das Einlesen geklappt hat. ◇

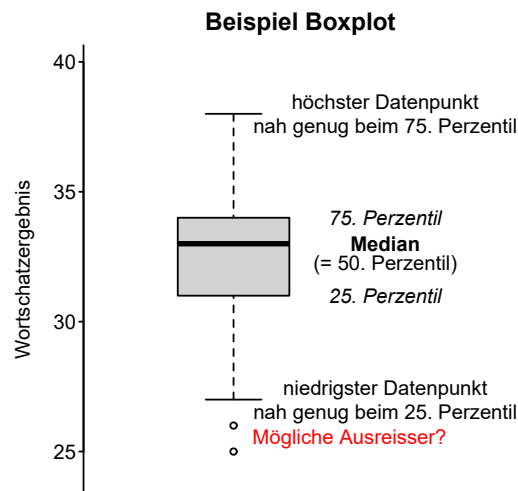


Abbildung 9.1: Erklärung Boxplot.

9.1.1 Grafische Darstellung und beschreibende Statistik

Eine nützliche grafische Darstellung, um unterschiedliche Gruppen hinsichtlich eines mehr oder weniger kontinuierlichen outcomes zu vergleichen, ist der **Boxplot** (zu Deutsch auch *Kastengrafik*). Abbildung 9.1 zeigt als Beispiel einen Boxplot der Ergebnisse bei einem Wortschatztest der 80 Versuchspersonen von Vanhove (2016). Die dickere Linie in der Mitte liegt beim Median und das Kästchen reicht vom 25. bis zum 75. Perzentil und umfasst somit die Hälfte der Datenpunkte. Manchmal gibt es (wie hier) auch Kreischen in einem Boxplot. Diese stellen Extremwerte dar, die mehr als 1.5 Mal die Distanz zwischen dem 25. und dem 75. Perzentil vom 25. oder 75. Perzentil entfernt liegen. Diese Extremwerte sind mögliche (!) Ausreisser. Mit einem Dotplot kann man besser überprüfen, ob sie tatsächlich auch Ausreisser sind. In diesem Beispiel liegen die zwei möglichen Ausreisser nicht sehr weit von anderen Datenpunkten entfernt, sodass sie nicht als Ausreisser gelten.

Mit `ggplot()` können solche Boxplots mithilfe des `geom_boxplot()`-Befehls erzeugt werden. Weiter ist zu bemerken, dass wir die Grafik zunächst einmal als ein Objekt namens `p_boxplot` in die Arbeitsumgebung speichern. Um die Grafik dann tatsächlich zu zeichnen, müssen wir lediglich diesen Objektamen eintippen. Das Ergebnis steht in Abbildung 9.2.

```
p_boxplot <- ggplot(data = d,
                    aes(x = MoneyGroup,
                       y = Sysjust)) +

  geom_boxplot() +
  xlab("Kondition") +
  ylab("Systemrechtfertigungsscore")
p_boxplot
```

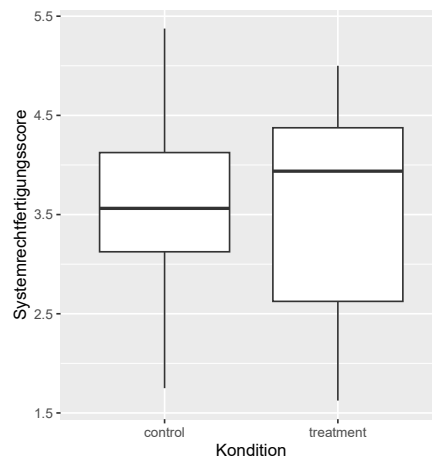


Abbildung 9.2: Vergleich der Systemrechtfertigungsscores in den beiden Konditionen in Klein et al.'s (2014) Replikation von Caruso et al. (2013, Experiment 1). Daten aus der Abington-Stichprobe.

Ich finde es oft eine gute Idee, dem Boxplot auch noch die einzelnen Datenpunkte hinzuzufügen (siehe auch Weissgerber et al., 2015). Insbesondere bei eher kleinen Datensätzen beeinträchtigt dies die Interpretierbarkeit der Grafik nicht und hilft es den Lesenden, einzuschätzen, wie die Daten tatsächlich verteilt sind. Boxplots können in dieser Hinsicht nämlich manchmal täuschen; siehe auch *Visualizing distributions with raincloud plots (and how to create them with ggplot2)* unter <https://www.cedricscherer.com/>.

Der Code unten zeigt Ihnen, wie Sie dies machen können. Dem `geom_boxplot()`-Befehl wird der Parameter `outlier.shape = NA` übergeben, womit verhindert wird, dass allfällige Extremwerte zwei Mal dargestellt werden: ein Mal als Kreischen beim Boxplot und ein Mal als einzelner Datenpunkt. Mit `geom_point()` werden die Datenpunkte einzeln dargestellt. Der Parameter `shape = 1` sorgt dafür, dass sie als leere Kreischen gezeichnet werden (siehe `?pch` → ‘pch’ values für andere mögliche Werte). Die Einstellung `position = position_jitter(width = ?, height = ?)` verschiebt die Punkte etwas horizontal und vertikal, damit überlappende Punkte sichtbar werden. Mit den Einstellungen unten werden die Punkte nur horizontal etwas verschoben, aber nicht vertikal. Zu guter Letzt werden die Defaultwerte auf der x-Achse (‘control’ und ‘treatment’; siehe Abbildung 9.2) mit dem Befehl `scale_x_discrete()` durch ‘ohne’ bzw. ‘mit’ ersetzt. Das Resultat zeigt Abbildung 9.3 auf der nächsten Seite.

```
p_boxplotdeluxe <- ggplot(data = d,
  aes(x = MoneyGroup,
    y = Sysjust)) +
  geom_boxplot(outlier.shape = NA) +
  geom_point(shape = 1,
    position = position_jitter(width = 0.2, height = 0)) +
  xlab("Banknote") +
```

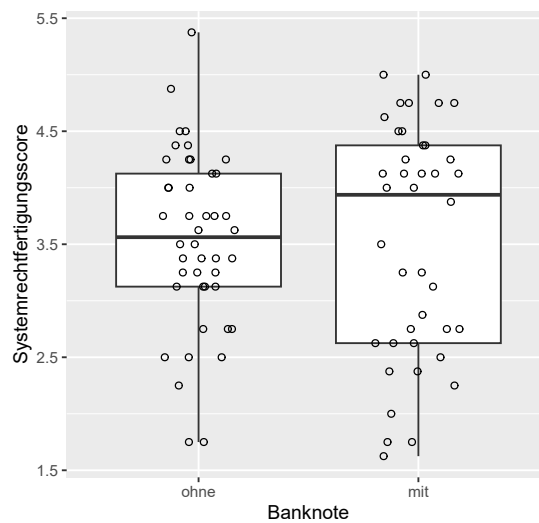


Abbildung 9.3: Nochmals die gleichen Daten, aber mit den einzelnen Datenpunkten.

```
scale_x_discrete(labels = c("ohne", "mit")) +
  ylab("Systemrechtfertigungsscore")
p_boxplotdeluxe
```

Mehr Informationen zu Befehlen wie `position_jitter()` und `scale_x_discrete()` finden Sie unter <https://ggplot2.tidyverse.org/reference/>. Siehe auch meinen Blogeintrag *Some alternatives to bar plots* (7.1.2015).

Eine Tabelle mit den üblichen beschreibenden Statistiken pro Gruppe können wir leicht mit `group_by()` und `summarise()` herstellen.

```
d |>
  group_by(MoneyGroup) |>
  summarise(AnzahlVpn = n(),
            Mittel = mean(Sysjust),
            Median = median(Sysjust),
            StdAbw = sd(Sysjust))
```

A tibble: 2 x 5

	MoneyGroup	AnzahlVpn	Mittel	Median	StdAbw
	<chr>	<int>	<dbl>	<dbl>	<dbl>
1	control	44	3.53	3.56	0.781
2	treatment	40	3.53	3.94	1.02

9.1.2 Modellierung

Die grafische Darstellung zeigt, dass der Median der ‘mit Banknote’-Kondition zwar höher ist als jener der ‘ohne Banknote’-Kondition, aber dass die Überlappung zwischen beiden Konditionen erheblich ist. Die numerische Zusammenfassung zeigt ausserdem, dass sich die Mittel kaum unterscheiden. Trotzdem können wir diese Daten—ähnlich wie im letzten Kapitel—in ein Modell giessen. Dieses Modell wird ebenfalls von Gleichung 8.3 auf Seite 169 beschrieben, die hier wiederholt wird:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad (9.1)$$

für $i = 1, \dots, n$. y_i stellt nun den Systemrechtfertigungsscore der i -ten Versuchsperson dar; x_i stellt die Gruppenzugehörigkeit dieser Versuchsperson dar, d.h., ob die Versuchsperson zu der ‘control’- oder ‘treatment’-Gruppe gehört. Genau wie vorher müssen β_0 und β_1 (und als Konsequenz davon auch ε_i) geschätzt werden.

Gleichungen wie diese können natürlich schwer mit Wörtern wie ‘control’ und ‘treatment’ umgehen, aber die Lösung ist erstaunlich einfach: Eine Gruppe bezeichnen wir als 0 und die andere als 1. Zum Beispiel können wir festlegen, dass $x_i = 0$, wenn die i -te Versuchsperson zur ‘control’-Kondition gehört, und dass $x_i = 1$, wenn sie zur ‘treatment’-Kondition gehört. (Wir könnten auch festlegen, dass $x_i = 1$ für Versuchspersonen in der Kontrollkondition und $x_i = 0$ für Versuchspersonen in der Experimentalkondition. Das macht eigentlich nichts aus.) Wenn wir dies gemacht haben, können wir den Vektor von Nullen und Einsen als Prädiktor in ein lineares Regressionsmodell aufnehmen. Wenn man kategorische Variablen (hier: Gruppenzugehörigkeit) als Zahlenreihen umschreibt, spricht man von **Dummy-Variablen**.

Gezeigt werden hier zwei Möglichkeiten, um die Dummy-Variable n.Kondition zu kreieren. Die erste funktioniert mit `ifelse()`:

```
d$n.Kondition <- ifelse(d$MoneyGroup == "treatment", yes = 1, no = 0)
```

Die zweite verwendet die tidyverse-Funktionen `mutate()` und `case_when()`. Beide Möglichkeiten liefern das gleiche Ergebnis; die Idee hier ist nur, die Funktion `case_when()` vorzustellen.

```
d <- d |>
  mutate(n.Kondition = case_when(
    MoneyGroup == "treatment" ~ 1, # falls MoneyGroup == "treatment"
    TRUE                       ~ 0 # sonst
  ))
```

Zur Kontrolle ist eine Kreuztabelle mit der ursprünglichen und der Dummy-Variablen nützlich:

```
xtabs(~ MoneyGroup + n.Kondition, d)

      n.Kondition
MoneyGroup  0  1
```

```
control  44  0
treatment 0 40
```

Jetzt können wir die Dummy-Variable als Prädiktor in einem linearen Modell verwenden:

```
money.lm <- lm(Sysjust ~ n.Kondition, data = d)
```

Wie gehabt können die geschätzten Parameter abgerufen werden, indem man den Namen des Modells eintippt.

```
money.lm

Call:
lm(formula = Sysjust ~ n.Kondition, data = d)

Coefficients:
(Intercept)  n.Kondition
  3.53409      -0.00597
```

Die zwei Parameterschätzungen sind $\hat{\beta}_0$ bzw. $\hat{\beta}_1$. Ihre Bedeutung kann aus der Regressionsgleichung hergeleitet werden:

$$y_i = 3.53 - 0.006 \cdot x_i + \hat{\varepsilon}_i,$$

für $i = 1, \dots, n$. Für Versuchspersonen in der Kontrollgruppe ist $x_i = 0$. Für solche Versuchspersonen wird die Gleichung zu

$$y_i = 3.53 - 0.006 \cdot 0 + \hat{\varepsilon}_i = 3.53 + \hat{\varepsilon}_i,$$

sodass $\hat{y}_i = 3.53$. $\hat{\beta}_0$ ist also das Gruppenmittel der Gruppe, die als 0 bezeichnet wurde. Für Versuchspersonen in der Experimentalgruppe ist $x_i = 1$. Für sie wird die Gleichung zu

$$y_i = 3.53 - 0.006 \cdot 1 + \hat{\varepsilon}_i,$$

sodass $\hat{y}_i = 3.53 - 0.006$. Durch Rundungsfehler ergibt dies eigentlich auch 3.53. $\hat{\beta}_1$ ist also der Unterschied zwischen den Mitteln der beiden Gruppen. Ist dieser Wert negativ, dann hat die Gruppe, die als 1 bezeichnet wurde, ein niedrigeres Mittel als die Gruppe, die als 0 bezeichnet wurde.

Aufgabe 9.2. Ändern Sie die Befehle oben, sodass nun die Kontrollgruppe als 1 bezeichnet wird und die Experimentalgruppe als 0. Was ändert sich im Output? ◇

Unsicherheit in den Parameterschätzungen quantifizieren

Den minimalen Unterschied zwischen den zwei Gruppenmitteln hätten wir auch einfach von Hand berechnen können. Der Mehrwert des allgemeinen linearen Modells besteht aber darin, dass wir auch die Unsicherheit in den Parameterschätzungen schätzen können; dies machen wir

hier. Ausserdem können dem allgemeinen linearen Modell mehrere Prädiktoren hinzugefügt werden; dies machen wir in einem nächsten Kapitel.

Bootstrappen ohne Normalitätsannahme. Die ‘neuen’ y -Werte (y^*) stellen sich aus den vom Modell ‘vorhergesagten’ y -Werten (\hat{y}) und einer Bootstrap-Stichprobe aus $\hat{\epsilon}$ zusammen (*sampling with replacement*).

```
runs <- 20000
bs_beta <- matrix(nrow = runs, ncol = 2)
predictions <- predict(money.lm)
residuals <- resid(money.lm)

for (i in 1:runs) {
  neu_Sysjust <- predictions + sample(residuals, replace = TRUE)
  bs_money.lm <- lm(neu_Sysjust ~ n.Kondition, data = d)
  bs_beta[i, ] <- coef(bs_money.lm)
}
```

Siehe Seite 174 für eine Erklärung; das Ergebnis dieser Befehle steht in Abbildung 9.4.

```
bs_beta_tbl <- tibble(Schnittpunkt = bs_beta[, 1],
                     Unterschied = bs_beta[, 2])

bs_beta_tbl |>
  pivot_longer(cols = everything(),
               names_to = "Parameter",
               values_to = "Estimate") |>
  ggplot(aes(x = Estimate)) +
  geom_histogram(fill = "lightgrey", col = "black", bins = 50) +
  facet_wrap(vars(Parameter), scales = "free") +
  xlab("Bootstrapschätzung") +
  ylab("Anzahl")
```

Die Standardabweichungen der Bootstrapverteilungen von $\hat{\beta}$ können wiederum als Schätzungen der Standardfehler dienen:

```
apply(bs_beta, 2, sd)

[1] 0.13451 0.19556
```

Ebenso können Konfidenzintervalle berechnet werden. In Fällen wie diesen interessiert man sich in der Regel hauptsächlich für den Standardfehler und das Konfidenzintervall um die Unterschiedsschätzung:

```
# 80% Konfidenzintervall
quantile(bs_beta[, 2], probs = c(0.1, 0.9))
```

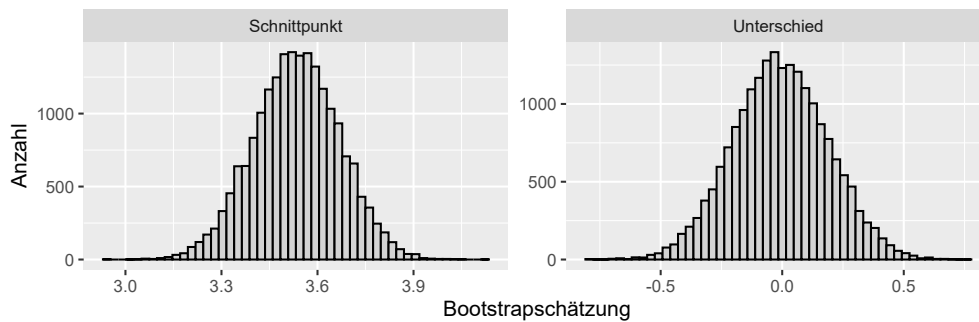


Abbildung 9.4: Verteilung der Bootstrap-Schätzungen der Parameter im Regressionsmodell `money.lm`.

10%	90%
-0.25539	0.24578

Bemerkung 9.3 (Alternative für heteroskedastische Daten). Im obigen Beispiel konnten $\hat{\varepsilon}$ -Werte aus der Kontrollkondition auch neu der Experimentkondition zugeordnet werden und umgekehrt. Dies entspricht der Homoskedastizitätsannahme (siehe Seite 180) des allgemeinen linearen Modells: Die Fehlervarianz ist überall gleich gross, sodass ein bestimmter Restfehler genau so gut in der anderen Kondition hätte vorkommen können. Man könnte den Bootstrap aber auch so durchführen, dass $\hat{\varepsilon}$ -Werte aus der Kontrollkondition nur der Kontrollkondition zugewiesen werden können und $\hat{\varepsilon}$ -Werte aus der Experimentalkondition nur der Experimentalkondition. Hiermit würde man die Möglichkeit berücksichtigen, dass die Fehlervarianz in der einen Gruppe von jener in der anderen Gruppe abweichen könnte.

In unserem Beispiel schaut diese Variante so aus:

```
runs <- 20000
bs_beta <- matrix(nrow = runs, ncol = 2)
predictions <- predict(money.lm)
residuals <- resid(money.lm)
bs_resid <- residuals

for (i in 1:runs) {
  # Residuen innerhalb jeder Kondition resampeln
  bs_resid[d$n.Kondition == 1] <- residuals[d$n.Kondition == 1] |>
    sample(replace = TRUE)
  bs_resid[d$n.Kondition == 0] <- residuals[d$n.Kondition == 0] |>
    sample(replace = TRUE)

  # Wie gehabt
  neu_Sysjust <- predictions + bs_resid
}
```

```
bs_money.lm <- lm(neu_Sysjust ~ n.Kondition, data = d)
bs_beta[i, ] <- coef(bs_money.lm)
}
```

Wie Sie selber kontrollieren können, ist das 80%-Konfidenzintervall jetzt etwas breiter. ◇

Aufgabe 9.4 (Konfidenzintervall für Unterschied zwischen Medianen). Wie würden Sie vorgehen, um mithilfe des Bootstraps ein Konfidenzintervall für den Unterschied zwischen den Medianen beider Konditionen zu konstruieren? Probieren Sie Ihren Vorschlag zu implementieren.

Hinweis: Die `lm()`-Funktion nutzt hier nichts. Verwenden Sie stattdessen `median()`. ◇

Bootstrappen mit Normalitätsannahme. Ähnlich wie in den letzten zwei Kapiteln kann man die Residuen auch aus einer Normalverteilung generieren.

```
runs <- 20000
bs_beta <- matrix(nrow = runs, ncol = 2)
predictions <- predict(money.lm)
sigma_eps <- sigma(money.lm)
n_obs <- length(predictions)
for (i in 1:runs) {
  neu_Sysjust <- predictions + rnorm(n_obs, sd = sigma_eps)
  bs_money.lm <- lm(neu_Sysjust ~ n.Kondition, data = d)
  bs_beta[i, ] <- coef(bs_money.lm)
}
```

Die Histogramme werden nicht nochmals gezeichnet.

```
apply(bs_beta, 2, sd)
[1] 0.13617 0.19909
quantile(bs_beta[, 2], probs = c(0.1, 0.9))
      10%      90%
-0.26239  0.24758
```

Mit *t*-Verteilungen. Wenn man ohnehin davon ausgehen will, dass der Restfehler aus einer Normalverteilung stammt, kann man wiederum die `summary()`-Funktion verwenden, um die Standardfehler abzurufen:

```
summary(money.lm)$coefficients
      Estimate Std. Error  t value Pr(>|t|)
(Intercept)  3.5340909    0.13633  25.923055 2.8981e-41
n.Kondition  -0.0059659    0.19756  -0.030198 9.7598e-01
```


Die Konfidenzintervalle um $\hat{\beta}$ können mit `confint()` abgerufen werden:

```
confint(money.lm, level = 0.8)

              10 %    90 %
(Intercept)  3.35796 3.71022
n.Kondition -0.26121 0.24928
```

Der minimale Unterschied zwischen den Gruppenmitteln von bloss -0.006 Punkten auf einer 7er-Skala hat also ein 80%-Konfidenzintervall von $[-0.26, 0.25]$ und könnte nach einer saloppen Interpretation von Konfidenzintervallen fast genau so gut positiv als auch negativ sein. In dieser Stichprobe bestätigt sich also das Ergebnis eines positiven Unterschiedes von Caruso et al. (2013) nicht. Die Bootstraphmethoden liefern ein nahezu identisches Ergebnis.

9.1.3 Treatment coding und sum-coding

Wenn man, wie oben, eine Gruppe als 0 und die andere als 1 bezeichnet, spricht man von **treatment coding**. Das Intercept stellt dann das Mittel der 0-Gruppe dar und die Steigung den Unterschied zwischen den Gruppenmitteln. Eine Alternative ist **sum-coding**. Hierzu wird die eine Gruppe als -0.5 und die andere als 0.5 bezeichnet:

```
d <- d |>
  mutate(n.Kondition = case_when(
    MoneyGroup == "treatment" ~ 0.5,
    TRUE                       ~ -0.5
  ))
money.lm <- lm(Sysjust ~ n.Kondition, data = d)
money.lm

Call:
lm(formula = Sysjust ~ n.Kondition, data = d)

Coefficients:
(Intercept)  n.Kondition
  3.53111      -0.00597
```

$\hat{\beta}_1$ stellt nach wie vor den Unterschied zwischen den beiden Gruppenmitteln dar, aber das Intercept ($\hat{\beta}_0$) stellt nun den **Gesamtmittelwert** (*grand mean*) dar. Dies ist das Mittel der Gruppenmittel. Achtung: Dies ist nicht unbedingt das Mittel sämtlicher Daten!

Aufgabe 9.5. Manche Forschende verwenden beim sum-coding lieber -1 und 1 als -0.5 und 0.5 . Was würde sich im Output ändern, wenn man dies machen würde? Was würde der geschätzte Parameter für `n.Kondition` jetzt bezeichnen? ◇

Bemerkung 9.6 (*Alabama first*). Eigentlich braucht man die Dummy-Variablen nicht selber zu kreieren: R macht dies automatisch, wenn Sie eine nicht-numerische Variable direkt dem Modell

hinzufügen:

```
money.lm2 <- lm(Sysjust ~ MoneyGroup, data = d)
money.lm2

Call:
lm(formula = Sysjust ~ MoneyGroup, data = d)

Coefficients:
      (Intercept)  MoneyGrouptreatment
           3.53409             -0.00597
```

Defaultmässig hantiert R treatment coding. Aber Achtung: Welche Gruppe als 0 bezeichnet wird und welche als 1, wird nach dem Alphabet festgelegt. 'treatment' kommt nach 'control', sodass 'control' die 0-Gruppe wird und 'treatment' die 1-Gruppe. Verlieren Sie dies bitte nicht aus dem Auge! Wenn Ihr Datensatz auf Deutsch zusammengestellt wurde, käme in einer L1-Variablen 'Deutsch' vor 'Französisch'; auf Englisch käme aber 'German' nach 'French'. Pflegen Sie besser die Gewohnheit, Ihre Dummy-Variablen selber zu kodieren, statt dies R zu überlassen. ◇

9.1.4 Annahmen überprüfen

Die wichtigsten Annahmen dieses Modells sind die folgenden.

1. Die Datenpunkte sind unabhängig voneinander. Ein klassisches Beispiel von Datensätzen mit *abhängigen* Datenpunkte sind Erhebungen in unterschiedlichen Schulklassen: Kinder aus derselben Klasse sind sich ähnlicher (aufgrund vorheriger Selektion, gemeinsamer Lehrkräfte, usw.) als Kinder aus unterschiedlichen Klassen. Die Konsequenz davon ist, dass Kinder aus derselben Klasse dem Modell keine vollständig neue Information hinzufügen. Das Modell 'weiss' dies aber nicht und würde deswegen fälschlicherweise davon ausgehen, dass jeder Eintrag den gleichen Informationswert hat. Dadurch würde der Standardfehler unterschätzt. Für weitere Diskussion und Lösungen, siehe Vanhove (2015).

Ein anderes Beispiel sind *within-subject*-Experimente, in denen Versuchspersonen in beiden/mehreren Konditionen getestet werden. *Within-subject*-Experimente bieten in der Regel mehr statistische Genauigkeit, sind aber schwieriger zu analysieren. Eine Option ist die Verwendung gemischter Modelle; siehe Kapitel ?? für Literaturempfehlungen. Wenn alle Versuchspersonen in zwei Konditionen getestet werden und es nur zwei Konditionen gibt, ist eine einfache Option, den Wert jeder Versuchsperson in der einen Kondition von ihren Wert in der anderen abzuziehen und diese Unterschiede zu analysieren.

Die Unabhängigkeitsannahme lässt sich meistens schwer überprüfen: Sogar kaum merkbare Abhängigkeiten können sich verheerend auf die statistischen Inferenzen auswirken. Die Gültigkeit der Unabhängigkeitsannahme muss auf der Basis von Sachwissen eingeschätzt werden: Man muss eben *wissen*, ob die Datenpunkte Klümpchen bilden oder nicht. In kontrollierten Experimenten kann man davon ausgehen, dass die Unabhängigkeit gegeben

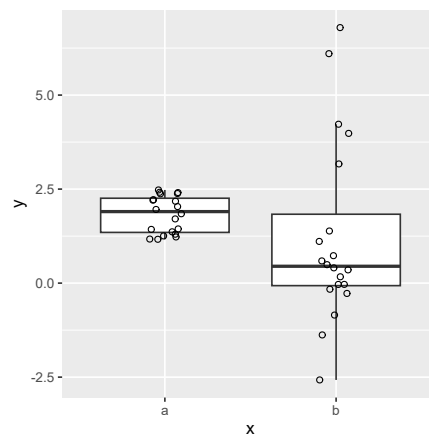


Abbildung 9.5: Beispiel von heteroskedastischen Daten.

ist, wenn die Teilnehmenden auf individueller Basis den Konditionen zufällig zugeordnet wurden.

2. Wenn die Konfidenzintervalle anhand von t -Verteilungen konstruiert werden, gehen wir auch davon aus, dass die Restfehler aus einer Normalverteilung stammen. In diesem Fall heisst dies, dass die outcome-Werte innerhalb jeder Kondition etwa normalverteilt sind, aber in komplexeren Modellen müsste man sich hierzu die Verteilung der Restfehler selber anschauen. Die Konstruktion der Konfidenzintervalle anhand von t -Verteilungen setzt normalverteilte Restfehler voraus, aber wie das Beispiel oben zeigt, kann eine Bootstrapmethode, die eben keine normalverteilten Restfehler voraussetzt, recht ähnliche Ergebnisse liefern, insbesondere, wenn die Datenmenge ausreichend ist. Meines Erachtens wichtiger ist aber, dass Restfehler, die nicht ungefähr normalverteilt sind, darauf hinweisen dürften, dass die Gruppenmittel, die man vergleicht, keine relevanten Masse der zentralen Tendenz sind. Siehe hierzu *Before worrying about model assumptions, think about model relevance* (11.4.2019).
3. Die Restfehler haben in jeder Gruppe die gleiche Streuung (Homoskedastizität). In etwa scheint dies in diesem Fall schon zu stimmen. Zum Vergleich: Die Boxplots in Abbildung 9.5 wären ein Grund, sich über diese Annahme Sorgen zu machen. Mögliche Lösungen finden sich bei Zuur et al. (2009, Kapitel 4). Eine andere Lösung wäre, dass man den Bootstrap so durchführt, dass die Residuen der einen Gruppe nicht der anderen Gruppe zugeordnet werden; siehe Bemerkung 9.3. Auch hier wiederhole ich mein Credo: *Before worrying about model assumptions, think about model relevance*.

Zur Überprüfung dieser Annahmen, siehe noch Vanhove (2018).

Aufgabe 9.7. Berthele (2012) spielte 155 angehenden Lehrpersonen eine Lautaufnahme vor, angeblich von einem Jungen, der Französisch als Fremdsprache spricht. Anschliessend wurden sie gebeten, das akademische Potenzial des Buben einzuschätzen (Skala von 1–6). In etwa der

Hälfte der Fälle enthielt die Lautaufnahme Codeswitches (also ein paar deutsche Wörter); in den anderen nicht. Ausserdem wurde der Hälfte der angehenden Lehrpersonen erzählt, der Junge hiesse Luca (ein gängiger Name in der Deutschschweiz); der anderen Hälfte wurde erzählt, er hiesse Dragan (ein Name, der man eher mit dem Balkan assoziiert). Die Frage war, ob dieses Labelling die Einschätzungen der angehenden Lehrpersonen beeinflusst, ggf. in Kombination mit den Codeswitches.

Hier fokussieren wir uns zunächst auf der Frage, wie gross der Unterschied in der durchschnittlichen Bewertung für Aufnahmen mit dem Dragan- und dem Luca-Label ist, wenn die Aufnahme Codeswitches enthält.

1. Lesen Sie die Datei `berthe1e2011.csv` in R ein.
2. Filtern Sie die Aufnahmen ohne Codeswitches heraus, sodass nur die Aufnahmen mit Codeswitches übrig bleiben.
3. Analysieren Sie die Bewertungen hinsichtlich der Frage, ob diese vom 'Luca'- vs. 'Dragan'-Label beeinflusst werden. Vergessen Sie nicht, die Daten grafisch darzustellen!
4. Fassen Sie Ihre Befunde in 2–3 Sätzen zusammen. ◇

Aufgabe 9.8. Ein anderer Befund, den Klein et al. (2014) zu replizieren versuchten, war der *gambler's fallacy*, der in einem Experiment von Oppenheimer & Monin (2009) belegt wurde. Klein et al. (2014) fassen dieses Experiment zusammen:

“Oppenheimer & Monin (2009) investigated whether the rarity of an independent, chance observation influenced beliefs about what occurred before that event. Participants imagined that they saw a man rolling dice in a casino. In one condition, participants imagined witnessing three dice being rolled and all came up 6's. In a second condition two came up 6's and one came up 3. In a third condition, two dice were rolled and both came up 6's. All participants then estimated, in an open-ended format, how many times the man had rolled the dice before they entered the room to watch him. Participants estimated that the man rolled dice more times when they had seen him roll three 6's than when they had seen him roll two 6's or two 6's and a 3. For the replication, the condition in which the man rolls two 6's was removed leaving two conditions.”

Die Daten der Replikationsstudie finden Sie in der Datei `Klein2014_gambler.csv`. Analysieren Sie den Datensatz hinsichtlich der Forschungsfrage, aber beschränken Sie sich dabei auf die Stichprobe der University of Florida (`uf1` in der Spalte `Sample`). Fassen Sie Ihre Befunde in 2–3 Sätzen zusammen. ◇

9.2 Unterschiede zwischen mehreren Gruppen

In Vanhove (2017) untersuchte ich, inwiefern die Genuszuordnungen im Deutschen ('Heisst es *der*, *die* oder *das Knie*?') bei flämischen Dialektsprecher:innen von ihrem Dialekt beeinflusst

werden. Zum Beispiel: Sagen Informant:innen, in deren Dialekt *knie* männlich ist, eher *der Knie*, verglichen mit Informant:innen, in deren Dialekt *knie* weiblich ist? Ich kam zum Schluss, dass dies kaum der Fall ist, sodass sich die Frage stellte, woran dies liegt. Eine Vermutung war, dass den Dialektsprecher:innen metalinguistische Kenntnisse über Genuszuordnungen in ihrem eigenen Dialekt fehlen, sodass sie nicht auf diese zurückgreifen können, wenn sie deutschen Nomen ein Genus zuordnen. In Vanhove (2019) überprüfte ich diese Vermutung, indem ich flämischen Dialektsprecher:innen metalinguistische Informationen über ihren Dialekt verschaffte. Konkret gab es drei Konditionen:

- Versuchspersonen in der ersten Kondition wurde eine Strategie erklärt, mit der sie das Genus eines Wortes in ihrem Dialekt erschliessen können.
- Versuchspersonen in der zweiten Kondition wurde mitgeteilt, dass ihr Dialekt (nicht aber das Standardniederländische) die gleiche Anzahl Genera wie das Deutsche hat. Ihnen wurde aber nicht erklärt, wie sie das Genus eines Wortes erschliessen können.
- Versuchspersonen in der dritten Kondition erhielten Informationen über einen irrelevanten Aspekt ihres Dialektes. Diese Kondition dient als Kontrollkondition.

Dann wurde geschaut, ob die Genuszuordnungen sich zwischen den Konditionen unterschieden. Getestet wurden 29 deutsche Nomen mit niederländischen Kognaten und es wurde gezählt, wie viele Genuszuordnungen im Deutschen pro Versuchsperson kongruent mit dem Genus des Kognats in ihrem Dialekt waren. Erwartet wurde insbesondere, dass Versuchspersonen in der ersten Kondition ('strategy') sich nach den Instruktionen eher am Dialekt orientieren als Versuchspersonen in den beiden anderen Konditionen ('information' und 'no information').

Aufgabe 9.9. Lesen Sie die Daten in der Datei `Vanhove2018.csv` ein; den Datensatz können Sie wieder `d` nennen. Die Spalte `ProportionCongruent` listet die Proportion kongruenter Genuszuordnungen pro Versuchsperson auf. ◇

9.2.1 Grafische Darstellung und beschreibende Statistik

Ob zwei oder mehrere Gruppen, die Boxplots kann man mit dem gleichen Befehl zeichnen. Das Einzige, was ich hier anders mache, ist, dass ich mit `scale_x_discrete()` die Konditionen in einer Reihenfolge aufliste, die m.E. sinnvoller ist als die alphabetische. Dazu verwende ich den Parameter `limits`.

```
ggplot(d,
  aes(x = Condition,
      y = ProportionCongruent)) +
  geom_boxplot(outlier.shape = NA) +
  geom_point(shape = 1,
             position = position_jitter(width = 0.2, height = 0)) +
  scale_x_discrete(limits = c("no information", "information", "strategy")) +
  xlab("Lernkondition") +
  ylab("Proportion L1-L2 kongruenter\nAntworten")
```

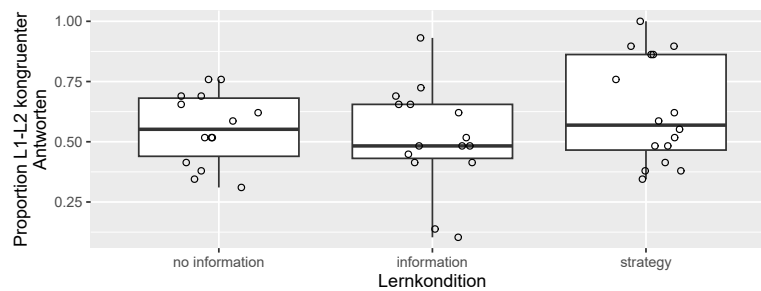


Abbildung 9.6: Boxplots der Daten von Vanhove (2019).

Die numerische Zusammenfassung machen wir wie gehabt:

```
d |>
  group_by(Condition) |>
  summarise(AnzahlVpn = n(),
            Mittel = mean(ProportionCongruent),
            Median = median(ProportionCongruent),
            StdAbw = sd(ProportionCongruent))

# A tibble: 3 x 5
  Condition      AnzahlVpn Mittel Median StdAbw
  <chr>          <int>   <dbl> <dbl> <dbl>
1 information         15  0.517  0.483  0.213
2 no information        14  0.554  0.552  0.150
3 strategy            16  0.627  0.569  0.219
```

9.2.2 Modellierung

Eine kategorische Variable mit zwei Ausprägungen konnten wir als eine binäre Dummy-Variable (0 vs. 1) umkodieren. Auch um eine kategorische Variable mit drei Ausprägungen in einem allgemeinen linearen Modell zu modellieren, brauchen wir Dummy-Variablen, aber wie viele? Man könnte die Zugehörigkeit zu jeder Kondition binär festlegen:

Kondition	Strategy?	Information?	(NoInformation?)
Strategy	1	0	(0)
Information	0	1	(0)
No information	0	0	(1)

Die letzte Spalte brauchen wir gar nicht: Wenn in der Strategy?- und in der Information?-Spalte eine Null steht, wissen wir schon, dass in der NoInformation?-Spalte eine Eins folgt. Wir brauchen also nur zwei Dummy-Variablen.

```
d$Strategy <- ifelse(d$Condition == "strategy", 1, 0)
d$Information <- ifelse(d$Condition == "information", 1, 0)
```

```
# Kontrolle:
d |> slice_head(n = 5)

# A tibble: 5 x 5
  SubjectID      Condition ProportionCongruent Strategy
  <chr>         <chr>          <dbl>         <dbl>
1 59b197ec251e0 strategy          0.586          1
2 59b3a2ae8fd4a no information      0.759          0
3 59b9482649f72 information        0.517          0
4 59b966ec6d23c strategy          0.862          1
5 59b9b20657c9f information        0.655          0
# i 1 more variable: Information <dbl>
```

Das allgemeine lineare Modell kann problemlos mit mehreren Prädiktoren umgehen, sodass wir beide Prädiktoren dem Modell hinzufügen können. Die Modellgleichung schaut so aus:

$$y_i = \beta_0 + \beta_1 \cdot x_{1,i} + \beta_2 \cdot x_{2,i} + \varepsilon_i,$$

für $i = 1, \dots, n$. Hier stellt $x_{1,i}$ den Wert der i -ten Versuchsperson bei der ersten Dummy-Variablen dar; $x_{2,i}$ stellt ihren Wert bei der zweiten Dummy-Variablen dar. So kann $x_{1,4} = 0$ sein, wenn die 4. Versuchsperson nicht der Information-Kondition zugeordnet wurde und 1, wenn sie schon dieser Kondition zugeordnet wurde.

```
mod.lm <- lm(ProportionCongruent ~ Information + Strategy, data = d)
mod.lm

Call:
lm(formula = ProportionCongruent ~ Information + Strategy, data = d)

Coefficients:
(Intercept)  Information      Strategy
    0.5542      -0.0369      0.0730
```

Aufgabe 9.10. Vergleichen Sie die geschätzten Parameter mit den Werten in der numerischen Zusammenfassung oben. Was stellt die Parameterschätzung für (Intercept) (also $\hat{\beta}_0$) dar? Was bedeuten die Parameterschätzungen für Information ($\hat{\beta}_1$) und Strategy ($\hat{\beta}_2$)? ◇

Aufgabe 9.11. Dem Modell kann die kategorische Variable der Konditionzugehörigkeit auch direkt hinzugefügt werden. Warum ergibt dies andere Parameterschätzungen?

```
mod.lm2 <- lm(ProportionCongruent ~ Condition, data = d)
mod.lm2
```

```
Call:
lm(formula = ProportionCongruent ~ Condition, data = d)

Coefficients:
      (Intercept) Conditionno information
              0.5172              0.0369
Conditionstrategy
              0.1099
```



9.2.3 Die Unsicherheit in den Parameterschätzungen quantifizieren

Die Bootstrapmethoden sollen mittlerweile ihrem didaktischen Zweck gedient haben, weshalb ich sie den interessierten Leser:innen als Übung überlasse. Die Standardfehler der Parameterschätzungen können wie gehabt mit `summary()` abgerufen werden:

```
summary(mod.lm)$coefficients
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	0.554187	0.053006	10.45526	2.9207e-13
Information	-0.036946	0.073701	-0.50129	6.1878e-01
Strategy	0.072968	0.072581	1.00533	3.2049e-01

Konfidenzintervalle lassen sich einfach mit `confint()` berechnen. Die Konfidenzintervalle um $\hat{\beta}_0$ sind in der Regel weniger interessant, aber werden automatisch mitberechnet.

```
confint(mod.lm, level = 0.90)
```

	5 %	95 %
(Intercept)	0.46503	0.643340
Information	-0.16091	0.087016
Strategy	-0.04911	0.195046

Wir könnten hier schlussfolgern, dass der Einfluss der metalinguistischen Instruktion eher ungewiss bleibt. Zwar gaben Versuchspersonen in der 'strategy'-Kondition mehr kongruente Antworten als jene in der 'no information'-kondition (7 Prozentpunkte mehr), aber die Unsicherheit in dieser Schätzung ist von der Grösse, dass auch negative Unterschiede oder Unterschiede nahe bei Null recht plausibel sind (vgl. das Konfidenzintervall).

9.2.4 Andere Kodierungssysteme

In dieser Analyse wurde treatment coding verwendet, sodass die Parameterschätzungen für β_1 und β_2 stets in Bezug zum Schnittpunkt (β_0) zu interpretieren sind. Dieser Schnittpunkt stellt den \hat{y} -Wert für Versuchspersonen in der Gruppe, die als (0,0) kodiert wurde, dar.

Manchmal sind aber andere Kodierungssysteme nützlich. Zum Beispiel könnte es sinnvoll

sein, die Parameter so schätzen zu lassen, dass $\widehat{\beta}_2$ den Unterschied zwischen der dritten und der zweiten Gruppe und nicht den Unterschied zwischen der dritten und der ersten Gruppe darstellt. Dies sei hier der Vollständigkeit halber erwähnt. Eine detaillierte (aber sehr nützliche!) Behandlung findet sich in Schad et al. (2020). Siehe auch den Blogeintrag *Tutorial: Obtaining directly interpretable regression coefficients by recoding categorical predictors* (5.5.2020).

Merksatz: Stellen Sie sicher, dass Sie wissen, worauf sich die Zahlen im Modelloutput überhaupt beziehen, bevor Sie diese theoretisch interpretieren!

Empfehlung: Probieren Sie, die Dummy-Variablen so zu kodieren, dass Sie die Antwort auf jede Forschungsfrage direkt aus einer Parameterschätzung (statt aus einer Kombination von mehreren) ablesen können. (Siehe dazu die Aufgaben.) Dann erhalten Sie bei jeder Antwort nämlich auch direkt ein Mass der Unsicherheit. Wenn Sie die Antwort aus mehreren Parameterschätzungen zusammenkombinieren müssen, ist dies nicht der Fall.

Aufgabe 9.12. Im Folgenden arbeiten wir mit Daten aus einer Längsschnittstudie mit drei Erhebungen. Im Projekt wurde u.a. die Lesefähigkeit Portugiesisch–Deutsch- und Portugiesisch–Französisch-zweisprachiger Kinder untersucht (Lambelet et al., 2017; Pestana et al., 2017). Die Datei `helascot_skills.csv` enthält die Ergebnisse der teilnehmenden Kinder bei mehreren Tests an den unterschiedlichen Erhebungen; `helascot_background.csv` enthält weitere Hintergrundinformationen zu den Kindern.

1. Lesen Sie die Datensätze `helascot_skills.csv` und `helascot_background.csv` ein und fügen Sie diese zusammen.
2. Kreieren Sie einen tibble, in dem nur die Angaben zu den Portugiesischtests (Variable `LanguageTested`) zur zweiten Erhebung (Variable `Time`) vorkommen.
3. Vergleichen Sie die Leistung von Kindern in Portugal (Variable `LanguageGroup: Control group Portuguese`) mit jener von Kindern in der Romandie (`Bilingual group French`) und in der Deutschschweiz (`Bilingual group German`) bei der portugiesischen Leseaufgabe (`Reading`) zur zweiten Erhebung. Tun Sie dies sowohl grafisch als auch in einem linearen Modell mit selbst kodierten Dummy-Variablen. Kodieren Sie die Dummy-Variablen dabei so, dass das Intercept das Mittel der Scores der Kinder aus Portugal zeigt und die beiden anderen Parameter jeweils den durchschnittlichen Unterschied zwischen den Kindern aus Portugal und den Kindern aus der Romandie bzw. aus der Deutschschweiz zeigen.
Achtung: Die Teilnehmenden in dieser Studie sind Schülerinnen und Schüler in Klassen. Die Beobachtungen verletzen somit vermutlich die Unabhängigkeitsannahme. Für diese Übung dürfen Sie diese Verletzung aber ignorieren.
4. Fakultativ: Kodieren Sie die Dummy-Variablen so, dass das Intercept das Mittel der Scores der Kinder aus der Deutschschweiz zeigt, der nächste Parameter den durchschnittlichen Unterschied zwischen Kindern aus der Deutschschweiz und Kindern aus der Romandie

zeigt, und der dritte Parameter den durchschnittlichen Unterschied zwischen Kindern aus der Romandie und Kindern aus Portugal zeigt. Rechnen Sie anhand des Modelloutputs kurz nach, ob die erhaltenen Schätzungen auch stimmen. (Tipp: Siehe Abschnitt 9.2.4.)

5. Fakultativ (schwierig): Kodieren Sie die Dummy-Variablen so, dass das Intercept das Mittel der Scores der Kinder aus Portugal zeigt, der nächste Parameter den durchschnittlichen Unterschied zwischen Kindern aus Portugal und Kindern aus der Schweiz (sowohl Romandie als auch Deutschschweiz) und der letzte Parameter den durchschnittlichen Unterschied zwischen der Romandie und der Deutschschweiz. Rechnen Sie anhand des Modelloutputs nach, ob die erhaltenen Schätzungen auch stimmen. (Tipp: Siehe Abschnitt 9.2.4, insbesondere den Blogeintrag oder Schad et al. (2020).)

(Eine solche Kodierung wäre geeignet, wenn Sie sich für diese zwei Forschungsfragen interessieren: (1) Wie stark unterscheidet sich die Leistung von Kindern in Portugal und portugiesischstämmigen Kindern in der Schweiz? (2) Wie stark unterscheidet sich die Leistung von portugiesischstämmigen Kindern in der Schweiz je nach Sprachregion?) ◇

9.2.5 Annahmen überprüfen

Die Annahmen beim Vergleichen mehrerer Gruppen sind identisch mit den Annahmen beim Vergleichen von zwei Gruppen.

In Vanhove (2019) wurden diese Daten übrigens anders analysiert, da der outcome eigentlich nicht kontinuierlich ist, sondern sich aus 29 binären Antworten pro Versuchsperson zusammensetzt. Diese Analyse führte aber zu den gleichen Schlussfolgerungen.

Kapitel 10

Interaktionen

Oft ist es nicht sosehr der Zusammenhang zwischen dem outcome und diesem oder jenem Prädiktor, der uns interessiert. Vielmehr sind wir am Zusammenspiel von zwei oder mehreren Prädiktoren interessiert. Zum Beispiel ist es nicht so interessant, ob man schneller auf hochfrequente als auf seltene Wörter reagiert – dieser Befund ist längst Gemeingut. Und es ist auch nicht so interessant, ob gute Lesende schneller auf bestehende Wörter reagieren als schlechte Lesende – auch das liegt auf der Hand. Interessanter wäre dahingegen die Frage, ob der Effekt von Wortfrequenz unterschiedlich gross ist je nach der Lesefähigkeit der Versuchspersonen. Dies ist eine Frage nach der **Interaktion** zwischen Lesefähigkeit und Wortfrequenz.

In Abbildung 10.1 auf der nächsten Seite werden drei von vielen möglichen Interaktionsmustern aufgeführt. Ihr gemeinsames Merkmal ist, dass die gezeichneten Linien nicht parallel laufen; bei der Absenz einer Interaktion ist dies schon der Fall. In der Grafik links oben liegt aber keine Interaktion zwischen Lesefähigkeit und Wortfrequenz vor: Beide Variablen hängen zwar mit der Lesegeschwindigkeit zusammen, aber der Zusammenhang zwischen Lesefähigkeit und Lesegeschwindigkeit unterscheidet sich nicht je nach Wortfrequenz (oder umgekehrt).

Ein mit 'Interaktion' verwandter Begriff ist **Haupteffekt**. Ein Haupteffekt eines Prädiktors, z.B. Wortfrequenz, auf Lesegeschwindigkeit liegt dann vor, wenn es, gemittelt über die Ausprägungen des anderen Prädiktors, z.B. Lesefähigkeit, einen Effekt des ersten Prädiktors gibt. In der Grafik links oben gibt es also einen Haupteffekt von Lesefähigkeit: Gemittelt über die Ausprägungen von Wortfrequenz lesen beschlagene Lesende schneller als schlechtere Lesende (gut: $\frac{4.5+6.5}{2} = 5.5$; schlecht: $\frac{3+5}{2} = 4$). In dieser Grafik gibt es auch einen Haupteffekt von Wortfrequenz: Gemittelt über die Ausprägungen von Lesefähigkeit werden hochfrequente Wörter schneller gelesen als Wörter mit niedriger Frequenz (hoch: $\frac{5+6.5}{2} = 5.75$; niedrig: $\frac{3+4.5}{2} = 3.75$).

Auch in der Grafik rechts oben gibt es Haupteffekte von sowohl Lesefähigkeit (gut: $\frac{4.5+9}{2} = 6.75$; schlecht: $\frac{3+5}{2} = 4$) als auch von Wortfrequenz auf Lesegeschwindigkeit (hoch: $\frac{5+9}{2} = 7$; niedrig: $\frac{3+4.5}{2} = 3.75$). Dies gilt auch für die Grafik links unten (gut: $\frac{6.5+6.5}{2} = 6.5$; schlecht: $\frac{3+5}{2} = 4$; hoch: $\frac{5+6.5}{2} = 5.75$; niedrig: $\frac{3+6.5}{2} = 4.75$).

In der letzten Grafik liegt ebenfalls einen Haupteffekt von Wortfrequenz vor (hoch: $\frac{2.5+5.5}{2} = 4$;

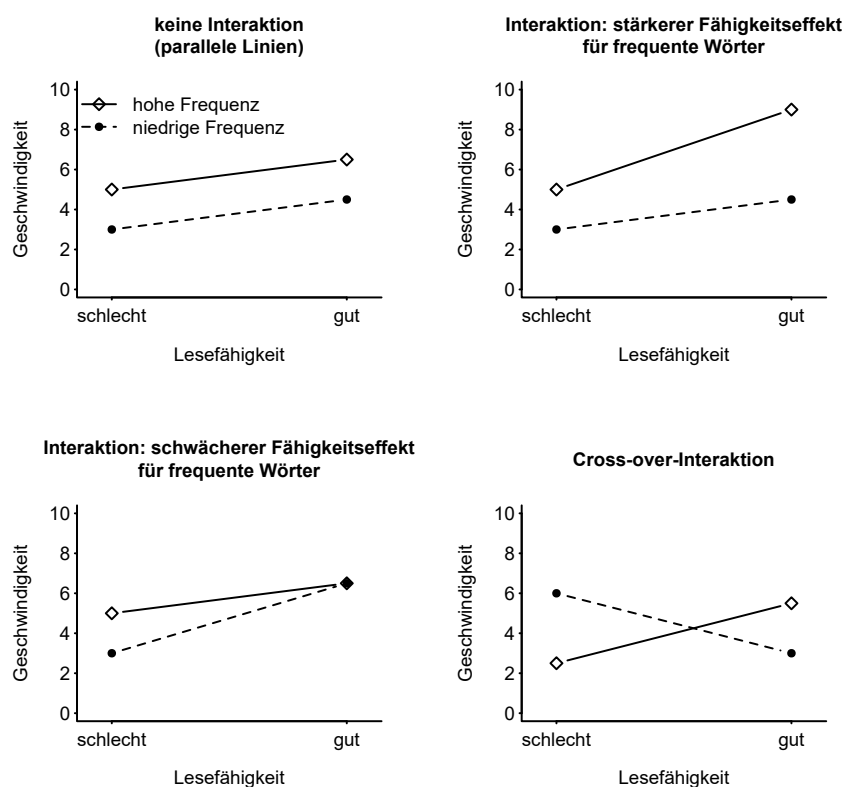


Abbildung 10.1: Wenn eine Interaktion zwischen Lesefähigkeit und Wortfrequenz auf Lesegeschwindigkeit vorliegt, dann unterscheidet sich der Effekt von Lesefähigkeit auf Lesegeschwindigkeit je nach Wortfrequenz. Umgekehrt gilt dann ebenfalls, dass sich der Effekt von Wortfrequenz auf Lesegeschwindigkeit je nach Lesefähigkeit unterscheidet. Dies zeigt sich in den nicht-parallelen Linien. (Die Einheiten entlang der y -Achse sind in diesem Beispiel arbiträr.)

niedrig: $\frac{6+3}{2} = 4.5$). Es gibt aber keinen Haupteffekt von Lesefähigkeit: Wenn man über die beiden Ausprägungen von Wortfrequenz mittelt, zeigt sich ein Nulleffekt (gut: $\frac{3+5.5}{2} = 4.25$; schlecht: $\frac{2.5+6}{2} = 4.25$). Die letzte Grafik ist ebenfalls ein Beispiel einer *cross-over*-Interaktion, denn die Effektslinien kreuzen sich.

10.1 Interaktionen zwischen zwei binären Prädiktoren

Eigentlich interessierte sich Berthele (2012) (Aufgaben letztes Kapitel) eher für die Interaktion zwischen dem Vorkommen (oder nicht) von Codeswitches und dem angeblichen Namen des Buben auf die Bewertungen seines akademischen Potenzials: Werden Codeswitches als gravierender betrachtet, wenn der Bub einen typischen Balkannamen hat als wenn er einen typischen schweizerischen Namen hat?

Aufgabe 10.1. Lesen Sie den Datensatz `berthele2011.csv` in R ein und nennen Sie ihn `d`. ◇

10.1.1 Grafische Darstellung und beschreibende Statistik

Wir können wieder Boxplots zeichnen, um die Muster in den Daten zu veranschaulichen. Mit `facet_grid(rows = vars(CS))` wird die Grafik vertikal aufgespaltet, und zwar in zwei Teile: einen für die Fälle, in denen Codeswitching vorlag, und einen für Fälle, in denen dies nicht der Fall war. Wie Abbildung 10.2 aber zeigt, dürften diese Daten etwas zu grobkörnig sein, um mit Boxplots dargestellt zu werden. Unten werden ein paar andere Grafiken gezeichnet, aber um diese zu zeichnen, müssen wir die Daten zuerst numerisch zusammenfassen.

```
ggplot(d,
  aes(x = Name,
      y = Potenzial)) +
  geom_boxplot(outlier.shape = NA) +
  geom_point(shape = 1,
    position = position_jitter(width = 0.2, height = 0)) +
  facet_grid(rows = vars(CS))
```

Mit `group_by()` kann man problemlos den Datensatz nach mehreren Variablen gruppieren, um so die Daten innerhalb jeder **Zelle** des Designs zusammenzufassen. Mit `.groups = "drop"` wird die vorgenommene Gruppierung wieder 'vergessen', auch wenn das hier eigentlich nicht wichtig ist; wenn Sie diesen Parameter weglassen, erhalten Sie eine harmlose Mitteilung.

```
summary_berthele <- d |>
  group_by(Name, CS) |>
  summarise(n = n(),
    Mittel = mean(Potenzial),
    StdAb = sd(Potenzial),
    .groups = "drop")
summary_berthele
```

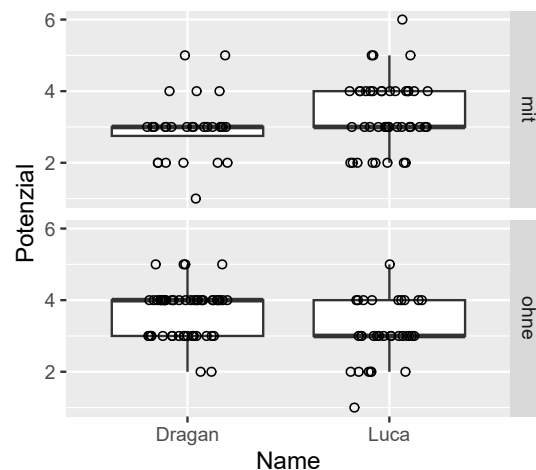


Abbildung 10.2: Boxplots der Bewertungen des akademischen Potenzials des Sprechers je nach Vorkommen von Codeswitches (mit vs. ohne) und seinem angeblichen Namen. Die Boxplots zeigen die Muster in den Daten hier nicht sehr deutlich, da die Daten nicht feinkörnig genug sind.

```
# A tibble: 4 x 5
  Name    CS      n Mittel StdAb
  <chr> <chr> <int> <dbl> <dbl>
1 Dragan mit      28  2.96 0.881
2 Dragan ohne     51  3.63 0.692
3 Luca   mit      44  3.36 0.942
4 Luca   ohne     32  3.09 0.856
```

Abbildung 10.3 stellt die soeben berechneten Gruppenmittel in einem Liniendiagramm dar. Manche Forschende mögen es nicht, wenn für kategoriale Prädiktoren Liniendiagramme gezeichnet werden, da diese implizieren würden, dass die Prädiktoren (hier etwa Name) kontinuierlich seien. Ich teile diese Meinung nicht: M.E. ist es klar, dass es zwischen Dragan und Luca keine Gradierung gibt. Die Zellenmittel selber werden ausserdem deutlich mit Punkten oder anderen Symbolen dargestellt, was dies nochmals betont. Die Linien, die diese Punkte verbinden, dienen nur der Deutlichkeit.

```
ggplot(summary_berthele,
  aes(x = Name,
    y = Mittel,
    linetype = CS, # unterschiedliche Linienarten je nach CS
    group = CS)) + # Manchmal braucht ggplot ein bisschen Hilfe,
  # um zu wissen, welche Punkte verknüpft werden
  # sollten. 'group' verschafft diese Hilfe.

  geom_point() +
```

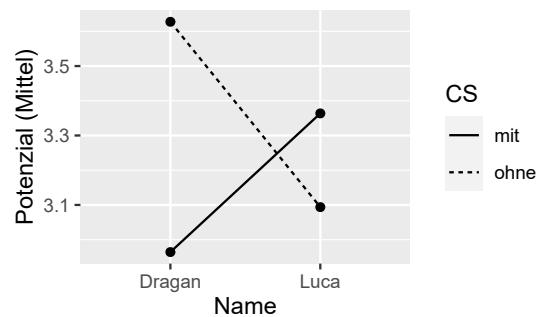


Abbildung 10.3: Liniendiagramm mit den Mitteln der Bewertungen des akademischen Potenzials des Sprechers je nach Vorkommen von Codeswitches (mit vs. ohne) und seinem angeblichen Namen. Die Muster sind hier deutlicher, aber dieser Grafik kann der Streuung der Daten nicht entnommen werden.

```
geom_line() +
ylab("Potenzial (Mittel)")
```

Es wäre jedoch nützlich, zusätzlich auch noch eine Idee über die Streuung der Daten innerhalb der Gruppen zu erhalten. Für Abbildung 10.4 wurden daher die Datenpunkte selber dargestellt und kombiniert mit den Mitteln aus `summary_berthele`. Es ist also möglich, Angaben aus unterschiedlichen Datensätzen in einer Grafik zu kombinieren.

```
ggplot(d,
  aes(x = Name,
      y = Potenzial)) +
geom_point(shape = 1, colour = "grey50",
  position = position_jitter(width = 0.2, height = 0)) +
geom_point(shape = 8, size = 3, colour = "blue",
  data = summary_berthele, # Daten aus anderem Datensatz
  aes(x = Name, y = Mittel)) +
geom_line(colour = "blue",
  data = summary_berthele, # Daten aus anderem Datensatz
  aes(x = Name, y = Mittel, group = CS)) +
facet_grid(rows = vars(CS))
```

Empfehlung: Probieren Sie mehrere Grafiken aus. Für Gruppenvergleiche sind Boxplots oft eine gute Wahl, aber eben nicht immer. Wenn eine Grafik Ihrer Meinung nach Verbesserungspotenzial hat, dann sollten Sie nicht zögern, andere Varianten auszuprobieren.

Übrigens kostet es oft Zeit, eine gute grafische Darstellung zu konstruieren. In diesem Skript finden Sie das Endergebnis meiner Bemühungen, aber insbesondere für die letzte Grafik

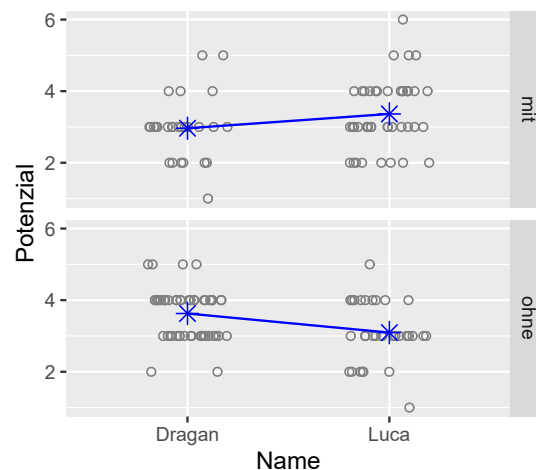


Abbildung 10.4: Bewertungen des akademischen Potenzials des Sprechers je nach Vorkommen von Codeswitches (mit vs. ohne) und seinem angeblichen Namen. Die Sternchen stellen die Gruppenmittel dar.

hat es eine Weile gedauert, bis ich herausgefunden habe, wie ich sie am besten zeichne. Die Entscheidung, die Datenpunkte grau zu färben, habe ich erst getroffen, nachdem ich feststellte, dass schwarze Datenpunkte die Zellenmittel nicht deutlich genug hervorhoben. Der Einstellung `size = 3` liegt eine ähnliche Beobachtung zu Grunde, usw.

10.1.2 Modellierung

Die Abbildungen zeigen, dass es innerhalb den Gruppen zwar ziemlich viel Variation gibt, aber dass der angebliche Name mit dem Vorkommen von Codeswitching interagieren dürfte: Liegen Codeswitches vor, dann wird das Potenzial von 'Luca' als besser eingestuft; liegen keine Codeswitches vor, dann scheinen die Lehrpersonen eher von der Leistung von Dragan als von jenen von Luca beeindruckt.

Mit der Modellierung möchten wir nicht nur die Fragen beantworten, welchen Zusammenhang der Name (Luca vs. Dragan) mit den Beurteilungen hat und welchen Zusammenhang Codeswitches mit ihnen haben, sondern auch inwiefern sich das Ausmass dieser Zusammenhänge je nach dem anderen Prädiktor unterscheidet. Dazu brauchen wir vier β -Parameter:

- β_0 , der Schnittpunkt, erfasst den Baseline der Beurteilungen.
- β_1 erfasst den Unterschied zwischen den Zellen je nach dem Namen (Dragan vs. Luca).
- β_2 erfasst den Unterschied zwischen den Zellen je nach dem Vorkommen von Codeswitches.
- β_3 passt β_1 und β_2 an: Wie viel grösser oder kleiner ist der Unterschied zwischen Dragan

und Luca, wenn Codeswitches vorliegen als wenn keine vorliegen?

Mathematisch schaut die Modellgleichung so aus:

$$y_i = \beta_0 + \beta_1 \cdot x_{1,i} + \beta_2 \cdot x_{2,i} + \beta_3 \cdot (x_{1,i} \cdot x_{2,i}) + \varepsilon_i, \quad i = 1, \dots, n.$$

- $x_{1,i}$ zeigt, ob der i -ten Versuchsperson erzählt wurde, dass der Bub Dragan (1) oder Luca (0) heisst.
- $x_{2,i}$ zeigt, ob der i -ten Versuchsperson der Text mit (1) oder ohne (0) Codeswitches vorgespielt wurde.¹
- Entsprechend ist β_0 die Durchschnittsbeurteilung von Versuchspersonen, die angeblich Luca (0) ohne Codeswitches (0) gehört haben.

Der Faktor $(x_{1,i} \cdot x_{2,i})$ mag erstaunen: Tatsächlich wird die Anpassung (Interaktion) berechnet, indem eine neue Variable kreiert wird, die das Produkt der beiden Prädiktoren enthält. Für die vier Zellen im Design ergibt dies die folgenden Werte:

- Luca (0) ohne Codeswitches (0): $x_{1,i} \cdot x_{2,i} = 0 \cdot 0 = 0$.
- Luca (0) mit Codeswitches (1): $x_{1,i} \cdot x_{2,i} = 0 \cdot 1 = 0$.
- Dragan (1) ohne Codeswitches (0): $x_{1,i} \cdot x_{2,i} = 1 \cdot 0 = 0$.
- Dragan (1) mit Codeswitches (1): $x_{1,i} \cdot x_{2,i} = 1 \cdot 1 = 1$.

Mit diesen Befehlen werden die Dummy-Variablen kreiert und ihr Produkt berechnet.

```
d$Dragan <- ifelse(d$Name == "Dragan", 1, 0)
d$MitCS <- ifelse(d$CS == "mit", 1, 0)
d$DraganMitCS <- d$Dragan * d$MitCS

# Kontrolle:
d |> slice_sample(n = 10)

# A tibble: 10 x 6
  CS      Name Potenzial Dragan MitCS DraganMitCS
<chr> <chr>      <dbl>  <dbl> <dbl>      <dbl>
1 mit   Dragan        3      1      1          1
2 ohne  Luca           3      0      0          0
3 mit   Luca           4      0      1          0
4 mit   Luca           4      0      1          0
5 mit   Luca           2      0      1          0
6 mit   Dragan          2      1      1          1
7 ohne  Dragan           3      1      0          0
# i 3 more rows
```

¹Diese Notation geht von treatment coding aus. Andere Kodierungssysteme sind möglich, aber schwieriger zu erklären.

Diese Variablen können dem allgemeinen linearen Modell als Prädiktoren hinzugefügt werden:

```
potenzial.lm <- lm(Potenzial ~ Dragan + MitCS + DraganMitCS, data = d)
potenzial.lm
```

Call:

```
lm(formula = Potenzial ~ Dragan + MitCS + DraganMitCS, data = d)
```

Coefficients:

(Intercept)	Dragan	MitCS	DraganMitCS
3.094	0.534	0.270	-0.933

Mit den geschätzten Koeffizienten können die Gruppenmittel rekonstruiert werden.²

- Nicht Dragan (also Luca), ohne Codeswitching:

$$\hat{y} = 3.09 + (0.53 \cdot 0) + (0.27 \cdot 0) + (-0.93 \cdot 0) = 3.09.$$

- Dragan, ohne Codeswitching:

$$\hat{y} = 3.09 + (0.53 \cdot 1) + (0.27 \cdot 0) + (-0.93 \cdot 0) = 3.63.$$

- Nicht Dragan (also Luca), mit Codeswitching:

$$\hat{y} = 3.09 + (0.53 \cdot 0) + (0.27 \cdot 1) + (-0.93 \cdot 0) = 3.36.$$

- Dragan, mit Codeswitching:

$$\hat{y} = 3.09 + (0.53 \cdot 1) + (0.27 \cdot 1) + (-0.93 \cdot 1) = 2.96.$$

Die nicht-numerischen Variablen können auch direkt dem Modell hinzugefügt werden. Um die Interaktion zwischen ihnen zu modellieren, kann die Name:CS-Notation verwendet werden, aber dann müssen die beiden Variablen auch noch einzeln eingetragen werden. Eine alternative Schreibweise ist die Name*CS-Notation: Diese wird von R intern zu Name + CS + Name:CS konvertiert.

```
potenzial.lm2 <- lm(Potenzial ~ Name + CS + Name:CS, data = d)
# Alternative Schreibweise
potenzial.lm2 <- lm(Potenzial ~ Name*CS, data = d)
potenzial.lm2
```

²In den Summen wurden die Rundungsfehler korrigiert. Die Klammern sind überflüssig, aber verdeutlichen die Struktur der Gleichungen.

```
Call:
lm(formula = Potenzial ~ Name * CS, data = d)

Coefficients:
      (Intercept)      NameLuca      CSohne
           2.964           0.399           0.663
NameLuca:CSohne
          -0.933
```

Aufgabe 10.2. Warum sind die Parameterschätzungen im Modell `potenzial.lm2` anders als diejenigen im Modell `potenzial.lm`? ◇

10.1.3 Unsicherheit einschätzen

Die Unsicherheit in den Parameterschätzungen kann wiederum mit `summary()` (für die Standardfehler) und mit `confint()` (für die Konfidenzintervalle) abgerufen werden. Die Konfidenzintervalle basieren wiederum auf *t*-Verteilungen und setzen also im Prinzip voraus, dass die Restfehler aus einer Normalverteilung stammen. In einem fakultativen Abschnitt werden wir diese Konfidenzintervalle nochmals berechnen mit einer Bootstrapmethode, die diese Voraussetzung nicht macht. Die Ergebnisse unterscheiden sich dank der Datenmenge und -verteilung aber kaum.

```
summary(potenzial.lm)$coefficients
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	3.09375	0.14796	20.9090	1.9477e-46
Dragan	0.53370	0.18876	2.8274	5.3289e-03
MitCS	0.26989	0.19446	1.3879	1.6722e-01
DraganMitCS	-0.93305	0.27672	-3.3719	9.4836e-04

```
confint(potenzial.lm, level = 0.95)
```

	2.5 %	97.5 %
(Intercept)	2.80141	3.38609
Dragan	0.16075	0.90665
MitCS	-0.11433	0.65410
DraganMitCS	-1.47979	-0.38632

Die Interpretation der Parameterschätzungen der Haupteffekte und ihre Unsicherheit ist etwas heikel: Wegen des treatment codings bezieht sich die Schätzung von 0.53 für *Dragan* *nur* auf Fälle, in denen kein Codeswitching vorliegt. Für die entsprechende Schätzung für Fälle mit Codeswitching muss man eben die Interaktion berücksichtigen: $0.53 - 0.93 = -0.40$. Analog bezieht sich die Schätzung von 0.27 für *MitCS* sich nur auf Fälle, in denen den Teilnehmenden gesagt wurde, die Aufnahme stamme von einem Buben namens Luca.

Die Interpretation der Interaktion und ihre Unsicherheit ist einfacher: Der Effekt von Codeswitching ist wesentlich gravierender für Dragan als für Luca, nämlich um 0.93 Punkte. Oder, äquivalent: Der Effekt der Absenz von Codeswitching ist 0.93 Punkte positiver für Dragan als für Luca. Es gibt zwar ziemlich viel Unsicherheit über diesen Effekt, aber nach einer saloppen Interpretation des Konfidenzintervalls (siehe Abschnitt 6.5 auf Seite 131) scheint die Richtung dieses Effekts klar zu sein, liegt das Intervall doch komplett im negativen Bereich.

In Kapitel 8 haben wir die Ergebnisse der Modellierung grafisch dargestellt und die Unsicherheit mit einem Konfidenzband veranschaulicht. Dies können wir für diese Modellierung (und übrigens auch für die Modellierungen aus dem letzten Kapitel) machen. Der nächste Abschnitt zeigt, wie man diesen fakultativen Schritt ausführen kann. Überspringen Sie es bei der ersten Lektüre.

10.1.4 Modelle visualisieren

Die Idee ist, dass die vom Modell ‘vorhergesagten’ Werte (\hat{y}) und die Unsicherheit um diese Werte dargestellt werden. Es gibt zwar ein paar R-Funktionen, mit denen man dies automatisch machen kann (siehe Fox, 2003), aber da es ein Ziel dieses Kurses ist, Statistik zu demystifizieren, wird hier gezeigt, wie man solche Grafiken selber erzeugen kann.

Schritt 1. Wir kreieren einen neuen Datensatz, der die Kombinationen der Prädiktorwerte enthält, für die wir die \hat{y} -Werte zeichnen wollen. Die `expand.grid()`-Funktion ist hierfür besonders praktisch, denn sie kreiert einen Datensatz, in dem alle Kombinationen der Prädiktorwerte vorkommen:

```
neue_daten <- expand.grid(Name = c("Dragan", "Luca"),
                          CS = c("mit", "ohne"))

neue_daten
```

	Name	CS
1	Dragan	mit
2	Luca	mit
3	Dragan	ohne
4	Luca	ohne

Schritt 2. Im Modell arbeiten wir mit Dummy-Variablen. Diese müssen wir dem neuen Datensatz hinzufügen. Geben Sie dabei den Dummy-Variablen den gleichen Namen wie bei der Modellierung.

```
neue_daten$Dragan <- ifelse(neue_daten$Name == "Dragan", 1, 0)
neue_daten$MitCS <- ifelse(neue_daten$CS == "mit", 1, 0)
neue_daten$DraganMitCS <- neue_daten$Dragan * neue_daten$MitCS
neue_daten
```

	Name	CS	Dragan	MitCS	DraganMitCS
1	Dragan	mit	1	1	1

2	Luca	mit	0	1	0
3	Dragan	ohne	1	0	0
4	Luca	ohne	0	0	0

Schritt 3. Fügen Sie mit der `predict()`-Funktion die modellierten \hat{y} -Werte hinzu. In diesem Fall sind diese \hat{y} -Werte den Zellenmitteln gleich.

```
neue_daten$Mittel <- predict(potenzial.lm, newdata = neue_daten)
neue_daten
```

	Name	CS	Dragan	MitCS	DraganMitCS	Mittel
1	Dragan	mit	1	1	1	2.9643
2	Luca	mit	0	1	0	3.3636
3	Dragan	ohne	1	0	0	3.6275
4	Luca	ohne	0	0	0	3.0938

Schritt 4. Fügen Sie mithilfe der `predict()`-Funktion die Limiten des erwünschten Konfidenzintervalls hinzu. Dazu muss man spezifizieren, dass man sich ein Konfidenzintervall wünscht und das Konfidenzniveau spezifizieren. Wie Sie sehen, generiert diese Funktion eine Matrix, deren zweite Spalte (`lwr`) die untere Grenze und deren dritte Spalte (`upr`) die obere Grenze enthält:

```
ki_grenzen <- predict(potenzial.lm, newdata = neue_daten,
                      interval = "confidence", level = 0.95)
ki_grenzen
```

	fit	lwr	upr
1	2.9643	2.6518	3.2768
2	3.3636	3.1143	3.6129
3	3.6275	3.3959	3.8590
4	3.0938	2.8014	3.3861

Diese Grenzen fügen wir dann `dummy_data` hinzu:

```
neue_daten$KI_unten <- ki_grenzen[, 2]
neue_daten$KI_oben <- ki_grenzen[, 3]
neue_daten
```

	Name	CS	Dragan	MitCS	DraganMitCS	Mittel	KI_unten
1	Dragan	mit	1	1	1	2.9643	2.6518
2	Luca	mit	0	1	0	3.3636	3.1143
3	Dragan	ohne	1	0	0	3.6275	3.3959
4	Luca	ohne	0	0	0	3.0938	2.8014

```
KI_oben
1 3.2768
2 3.6129
```

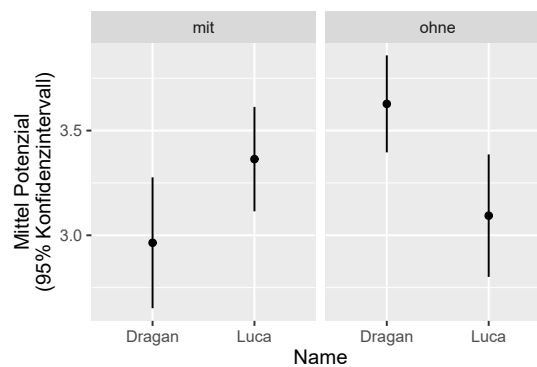


Abbildung 10.5: Zellenmittel und modellbasierte 95%-Konfidenzintervalle.

```
3 3.8590
4 3.3861
```

Schritt 5. Stellen Sie die Werte in Mittel als Punkte dar und zeichnen Sie anhand der Konfidenzlimiten Intervalle um sie, siehe Abbildung 10.5. Wenn Sie die vier #-Zeichen entfernen, werden auch noch die Rohdaten gezeichnet.

```
ggplot(neue_daten,
       aes(x = Name,
           y = Mittel)) +
  # geom_point(data = d,
  #           shape = 1, colour = "grey",
  #           aes(y = Potenzial),
  #           position = position_jitter(width = 0.2, height = 0)) +
  geom_point() +
  geom_linerange(aes(ymin = KI_unten, # untere Limite
                    ymax = KI_oben)) + # obere Limite
  facet_grid(cols = vars(CS)) +
  ylab("Mittel Potenzial\n(95% Konfidenzintervall)")
```

Wir hätten auch direkt auf der Basis der Zellenmittel und -standardabweichungen Konfidenzintervalle berechnen können, und zwar wie folgt:

```
summary_berthele <- summary_berthele |>
  mutate(
    SE = StdAb / sqrt(n),
    KI_unten = Mittel + qt(0.025, df = n - 1) * SE,
    KI_oben = Mittel + qt(0.975, df = n - 1) * SE
  )
summary_berthele
```

```
# A tibble: 4 x 8
  Name      CS      n Mittel StdAb      SE KI_unten KI_oben
  <chr> <chr> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
1 Dragan mit      28  2.96 0.881 0.167  2.62  3.31
2 Dragan ohne     51  3.63 0.692 0.0969 3.43  3.82
3 Luca mit      44  3.36 0.942 0.142  3.08  3.65
4 Luca ohne     32  3.09 0.856 0.151  2.79  3.40
```

Der Grund, weshalb die Konfidenzintervalle in `neue_daten` und in `summary_berthele` nicht identisch sind, ist, dass erstere von einem Modell abgeleitet wurden, das davon ausgeht, dass die Restfehler in jeder Zelle die gleiche Streuung hat. Letztere wurden direkt von den Standardabweichungen in den Zellen abgeleitet. Wenn der Restfehler in jeder Zelle tatsächlich (in der Population) die gleiche Streuung hat, liefert das Modell die besseren Konfidenzintervalle, da für die Schätzung der Streuung der Restfehler alle Datenpunkte zur Verfügung berücksichtigt wurden.

Weitere hoffentlich nützliche Links:

- Blogbeitrag *Tutorial: Plotting regression models* (23.4.2017)
- Blogbeitrag *Tutorial: Adding confidence bands to effect displays* (12.5.2017)
- <https://socviz.co/modeling.html#get-model-based-graphics-right>
- https://janhove.github.io/visualise_uncertainty/

10.2 Annahmen überprüfen

Da auch das Modell, mit dem wir uns gerade amüsiert haben, ein Beispiel eines allgemeinen linearen Modells ist, hat es die gleichen Annahmen wie die Modelle aus den letzten Kapiteln: Unabhängigkeit, Normalität und Homoskedastizität. Die Beschreibung von Berthele (2012) lässt vermuten, dass Unabhängigkeit gegeben ist. Mit `plot(potenzial.lm)` können diagnostische Plots erzeugt werden, mit denen die Normalitäts- und Homoskedastizitätsannahmen eingeschätzt werden können, siehe Abbildung 10.6.

Die ‘Treppchen’ in der Grafik rechts oben zeigen, was wir schon wussten, nämlich dass die Potenzialbeurteilungen ziemlich grobkörnig sind. Normalverteilte Variablen sind im Prinzip unendlich feinkörnig und können ausserdem theoretisch von $-\infty$ bis ∞ reichen. Diese Daten sind aber theoretisch beschränkt zwischen 1 und 6. Erfahrungsgemäss weiss ich, dass solche Abweichungen von den theoretischen Annahmen wenig ausmachen, wenn die Datenmenge ausreichend ist, aber im Zweifelsfall können Sie die Konfidenzintervalle mit einer Methode überprüfen, die andere Annahmen macht: dem Bootstrap. Die nächste fakultative Bemerkung zeigt, wie das geht.

Bemerkung 10.3 (Konfidenzintervalle mit dem Bootstrap überprüfen). Das Prinzip hinter dem Bootstrap wurde bereits mehrmals erklärt. Die letzten paar Male haben wir es erlaubt,

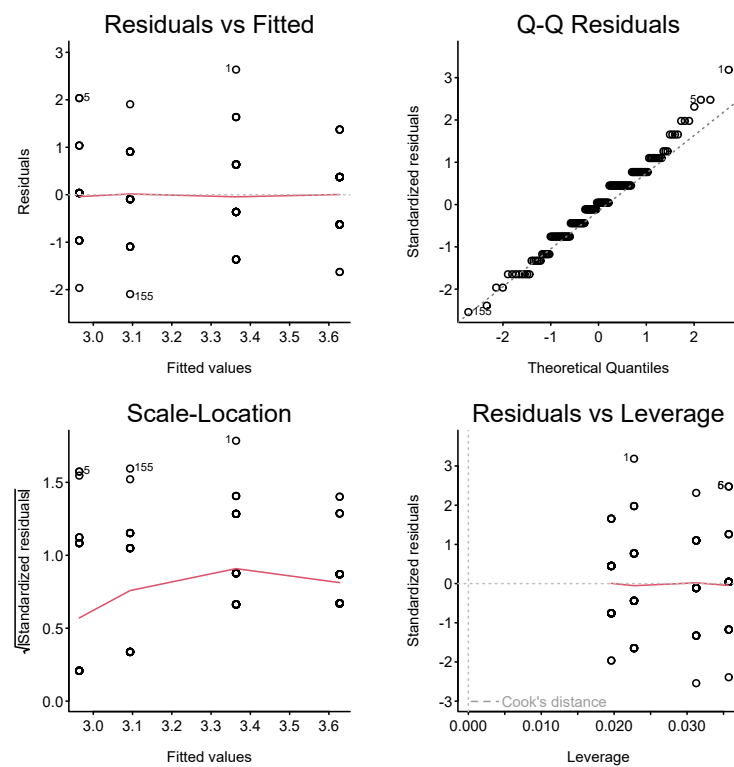


Abbildung 10.6: Diagnostische Plots für das potenzial.lm-Modell.

dass Restfehler aus der einen Zelle/Kondition in der Bootstrapstichprobe in der anderen Zelle/Kondition auftauchen. Dies entsprach der Homoskedastizitätsannahme des allgemeinen linearen Modells. Aber wir können dies auch verhindern, indem wir die Bootstrapstichprobe so gestalten, dass Datenpunkte in einer bestimmten Zelle nur in der gleichen Zelle 'rezykliert' werden. Jede Zelle sollte gleich gross sein wie in der ursprünglichen Stichprobe. Dies entspricht dem Vorgehen aus Bemerkung 9.3.

```
# Zellengrößen in der eigentlichen Stichprobe
```

```
xtabs(~ Dragan + MitCS, data = d)
```

```
      MitCS
Dragan  0  1
      0 32 44
      1 51 28
```

```
# Innerhalb jede Zelle bootstrappen
```

```
bs_dat <- d |>
```

```
  group_by(Dragan, MitCS) |>
```

```
  slice_sample(prop = 1, replace = TRUE)
```

```
# Zellengrößen in der Bootstrapstichprobe
```

```
xtabs(~ Dragan + MitCS, data = bs_dat)
```

```
      MitCS
Dragan  0  1
      0 32 44
      1 51 28
```

Den Bootstrap können wir wie gehabt durchführen.

```
# Anzahl Bootstrapstichproben definieren
```

```
runs <- 20000
```

```
# Matrix für Parameterschätzungen
```

```
bs_beta <- matrix(nrow = runs, ncol = 4)
```

```
# For-loop für Bootstraps
```

```
for (i in 1:runs) {
```

```
  bs_dat <- d |>
```

```
    group_by(Dragan, MitCS) |>
```

```
    slice_sample(prop = 1, replace = TRUE)
```

```
  bs_mod <- lm(Potenzial ~ Dragan + MitCS + DraganMitCS,
               data = bs_dat)
```

```
# Zeile der Matrix füllen.
```

```
# 1. Spalte = intercept; 2. Spalte, Dragan;
# 3. Spalte = MitCS; 4. Spalte: Interaktion
bs_beta[i, ] <- coef(bs_mod)
}
```

Die Verteilungen der Bootstrapschätzungen können grafisch dargestellt werden (Abbildung 10.7); siehe Seite 174 für eine Erklärung der Befehle. Bemerken Sie, dass die Verteilung der Schätzungen für den Schnittpunkt Lücken aufzeigt: Der Schnittpunkt erfasst das Mittel der 32 Beurteilungen für Luca ohne Codeswitches. Da diese Beurteilungen aber grobkörnig sind, gibt es Werte, die gar kein Mittel dieser Beurteilungen sein können.

```
bs_beta_tbl <- tibble(
  "B0: Schnittpunkt" = bs_beta[, 1],
  "B1: Dragan" = bs_beta[, 2],
  "B2: Codeswitches" = bs_beta[, 3],
  "B3: Interaktion" = bs_beta[, 4]
)

# Grafik zeichnen
bs_beta_tbl |>
  pivot_longer(cols = everything(),
               names_to = "Parameter",
               values_to = "Estimate") |>
  ggplot(aes(x = Estimate)) +
  geom_histogram(fill = "lightgrey", col = "black", bins = 50) +
  facet_wrap(vars(Parameter), scales = "free", ncol = 2) +
  xlab("Bootstrapschätzung") +
  ylab("Anzahl")
```

Auch ohne die Annahme, dass die Restfehler aus einer Normalverteilung stammen und dass sie in jeder Zelle die gleiche Streuung haben, erhalten wir grundsätzlich die gleichen Konfidenzintervalle um die geschätzten Parameter als mit `confint(potenzial.lm, level = 0.95)`;

```
apply(bs_beta, 2, quantile, probs = c(0.025, 0.975))

      [,1]      [,2]      [,3]      [,4]
2.5% 2.8125 0.19301 -0.12507 -1.47917
97.5% 3.3750 0.88174  0.67614 -0.38503
```



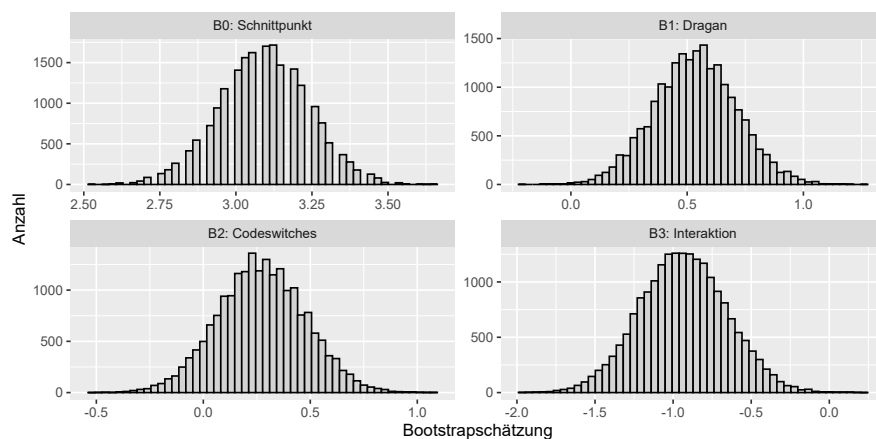


Abbildung 10.7: Bootstrapschätzungen der Variabilität der Modellparameter ohne Homoskedastizitätsannahme.

10.3 Interaktionen mit einem kontinuierlichen Prädiktor

Manchmal stösst man auf Studien, in denen untersucht wird, ob der Zusammenhang eines kontinuierlichen Prädiktors mit dem outcome von Gruppe zu Gruppe variiert. Auch diese Art Fragestellung betrifft die Interaktion von zwei Variablen: einer kontinuierlichen und einer kategorischen.

Bemerkung 10.4 (lieber ein grösseres Modell als viele kleine). Um den Zusammenhang zwischen einem kontinuierlichen Prädiktor und dem outcome in unterschiedlichen Gruppen zu vergleichen, analysieren viele Forschende ihre Daten in separaten Modellen (einem Modell pro Gruppe). Wenn wir später über Signifikanztests sprechen, wird klar werden, wieso dies in der Regel eine schlechte Idee ist (siehe auch Gelman & Stern, 2006; Nieuwenhuis et al., 2011). Es ist besser die unterschiedlichen Prädiktoren und ihre Interaktionen in *einem* Modell zu analysieren, denn so erhält man Parameterschätzungen für die Interaktionen *und* Indizien über die Unsicherheit dieser Schätzung. Wenn man die Gruppen in separaten Modellen analysiert, erhält man keine solchen Unsicherheitsmasse, und entsprechend wird die Unsicherheit über die Interaktionen in solchen Fällen fast ausnahmslos unterschätzt. ◇

Leider habe ich keinen Zugriff auf Datensätze, die eine solche Forschungsfrage behandeln und sinnvoll mit dem allgemeinen linearen Modell analysiert werden können. Um das Vorgehen zu illustrieren, versuche ich hier mit den Daten aus meiner Diss (siehe Übungen Kapitel 8) die etwas banale Frage, ob gute Englischkenntnisse beim Erkennen von gesprochenen Kognaten für Männer und Frauen unterschiedlich nützlich sind, zu beantworten.

```
# Daten einlesen und kombinieren; siehe Kapitel 8
cognates <- read_csv(here("data", "vanhove2014_cognates.csv"))
background <- read_csv(here("data", "vanhove2014_background.csv"))
d <- cognates |>
```

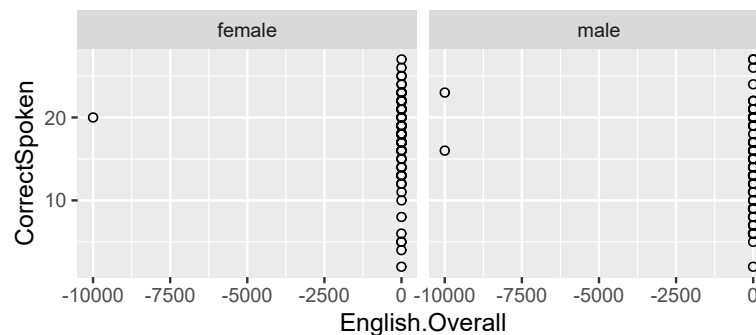


Abbildung 10.8: Ups.

```

left_join(background)
head(d)

# A tibble: 6 x 11
  Subject Sex      CorrectWritten CorrectSpoken FirstBlock
  <dbl> <chr>          <dbl>         <dbl> <chr>
1     64 female           25           23 Spoken
2    230 male             26           12 Spoken
3    527 male             15           9 Spoken
4    550 female           24           22 Spoken
5    552 female           18           17 Spoken
6    675 male             22           20 Spoken
# i 6 more variables: Age <dbl>, NrLang <dbl>,
#   DS.Span <dbl>, WST.Right <dbl>, Raven.Right <dbl>,
#   English.Overall <dbl>

```

10.3.1 Grafische Darstellung

Abbildung 10.8 hebt hervor, dass man nie blind herumrechnen sollte: Der Wert -9999 bei der Variablen `English.Overall` ist kein echter Wert, sondern will lediglich heissen, dass diese Angaben nicht vorliegen. Im Folgenden werden wir solche fehlenden Angaben einfach ignorieren.

```

ggplot(d,
  aes(x = English.Overall,
      y = CorrectSpoken)) +
  geom_point(shape = 1) +
  facet_grid(cols = vars(Sex))

```

Abbildung 10.9 zeigt, dass es sowohl für Männer als auch für Frauen einen positiven Zusammenhang zwischen der Kognatübersetzungsleistung und der Leistung beim Englischtest

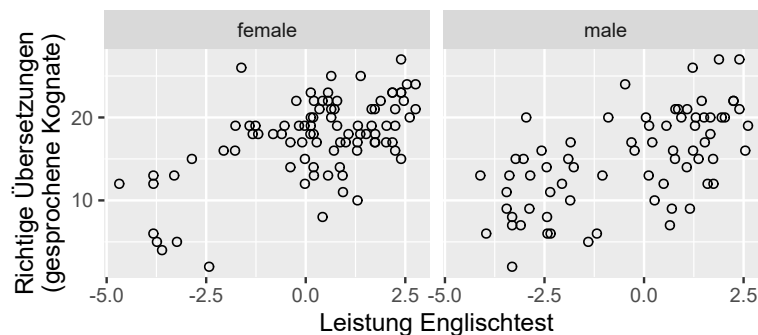


Abbildung 10.9: Für sowohl Männer als auch Frauen gibt es einen positiven Zusammenhang zwischen der Leistung beim Englischtest und bei der Kognatübersetzung.

gibt.

```
# Fehlende Daten löschen:
d <- d |>
  filter(English.Overall != -9999)

# Grafik zeichnen
ggplot(d,
  aes(x = English.Overall,
      y = CorrectSpoken)) +
  geom_point(shape = 1) +
  facet_grid(cols = vars(Sex)) +
  xlab("Leistung Englischtest") +
  ylab("Richtige Übersetzungen\n(gesprochene Kognate)")
```

10.3.2 Modellierung

Die Modellierung verläuft analog zur vorherigen. Da es in der Stichprobe mehr Frauen als Männer gibt, kodiere ich Frauen als 0 und Männer als 1, aber das macht eigentlich nichts aus. Die Variable `English.Overall` ist bereits zentriert um 0, sodass wir dies nicht mehr machen müssen.

```
d$Mann <- ifelse(d$Sex == "male", 1, 0)
```

Auch in diesem Fall bezieht sich der geschätzte Parameter für die Interaktion auf das Produkt der beiden Prädiktoren. Aber R wird dies automatisch machen.

```
kognat.lm <- lm(CorrectSpoken ~ English.Overall * Mann,
  data = d)
kognat.lm
```

```
Call:
lm(formula = CorrectSpoken ~ English.Overall * Mann, data = d)

Coefficients:
      (Intercept)      English.Overall
           17.145              1.592
           Mann  English.Overall:Mann
          -1.492              0.128
```

Aus der allgemeinen Regressionsgleichung, der wir bereits mehrmals begegnet sind, können wir herleiten, worauf sich diese Schätzungen beziehen:

- $\widehat{\beta}_0$: Die (modellierte) durchschnittliche Kognatübersetzungsleistung einer Frau mit einem Englischergebnis von 0 (hier: dem Stichprobenmittel). (Etwa 17 Punkte.)
- $\widehat{\beta}_1$: Wie viel besser schneidet (laut dem Modell) eine Frau im Schnitt ab, wenn sie ein Englischergebnis von 1 statt von 0 hat? (Etwa 1.6 Punkte besser.)
- $\widehat{\beta}_2$: Wie viel besser schneidet (laut dem Modell) ein Mann im Schnitt ab, wenn er ein Englischergebnis von 0 hat, verglichen mit einer Frau mit dem gleichen Englischergebnis? (Etwa 1.5 Punkte schlechter.)
- $\widehat{\beta}_3$: Wie viel 'nützlicher' ist ein Englischergebnis von 1 als eins von 0 für einen Mann als für eine Frau? (Etwa 0.1 Punkt beim Kognatsübersetzungstest nützlicher.)

10.3.3 Unsicherheit einschätzen

Die Standardfehler und Konfidenzintervalle können wir wie gehabt abfragen.

```
summary(kognat.lm)$coefficients
```

	Estimate	Std. Error	t value
(Intercept)	17.14534	0.45808	37.42903
English.Overall	1.59200	0.25399	6.26803
Mann	-1.49227	0.68876	-2.16660
English.Overall:Mann	0.12806	0.35709	0.35863

```
      Pr(>|t|)
```

(Intercept)	7.8339e-80
English.Overall	3.4164e-09
Mann	3.1783e-02
English.Overall:Mann	7.2036e-01

```
confint(kognat.lm, level = 0.9)
```

	5 %	95 %
(Intercept)	16.38737	17.90331
English.Overall	1.17173	2.01227

Mann	-2.63194	-0.35259
English.Overall:Mann	-0.46281	0.71893

Hinsichtlich unserer banalen Frage zeigt das Konfidenzintervall für die Interaktion, dass Englischkenntnisse in dieser Stichprobe zwar nützlicher für Männer scheinen als für Frauen, aber die Unsicherheit ist so gross, dass das Muster auch mit einem negativen oder ungefähren Nullergebnis kompatibel ist.

10.3.4 Annahmen überprüfen

Die Annahmen werden auf die gleiche Art und Weise überprüft als in den letzten Abschnitten und Kapiteln. Das Vorgehen wird hier nicht gezeigt, da die Frage derartig banal ist und ich in Vanhove (2014) die Daten ohnehin in einem angemessenen Modell analysiert habe (im Hinblick auf leicht weniger banale Fragen).

10.3.5 Modell visualisieren

Um das Modell zu visualisieren, kann die Technik, die oben erklärt wurde, verwendet werden. Diese wird hier im Telegrammstil wiederholt. Das Ergebnis ist Abbildung 10.10.

```
# Datensatz mit Prädiktorkombinationen generieren.
# Für English.Overall nehme ich einfach die einzelnen
# Werte, die in der Stichprobe beobachtet wurden (unique()).
neue_daten <- expand.grid(Sex = c("male", "female"),
                        English.Overall = unique(d$English.Overall))
# Der Datensatz wird nicht angezeigt, um Papier zu sparen.
# neue_daten

# Dummy-Variable kodieren
neue_daten$Mann <- ifelse(neue_daten$Sex == "male", 1, 0)

# y-hat-Werte hinzufügen
neue_daten$yhat <- predict(kognat.lm, newdata = neue_daten)

# Konfidenzlimiten hinzufügen: hier sowohl 80% als auch 95%
limiten80 <- predict(kognat.lm, newdata = neue_daten,
                    interval = "confidence", level = 0.80)
limiten95 <- predict(kognat.lm, newdata = neue_daten,
                    interval = "confidence", level = 0.95)
neue_daten$ki_unten80 <- limiten80[, 2]
neue_daten$ki_oben80 <- limiten80[, 3]
neue_daten$ki_unten95 <- limiten95[, 2]
neue_daten$ki_oben95 <- limiten95[, 3]
# Ergebnis inspizieren (nicht gezeigt)
```

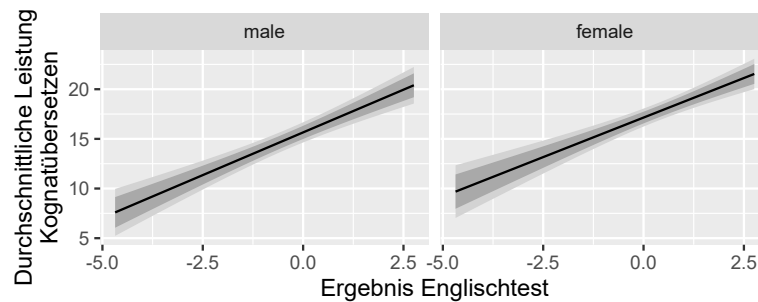


Abbildung 10.10: Visualisierung des `kognat.lm`-Modells mit 80%- und 95%-Konfidenzbändern.

```
# neue_daten |> slice_head(n = 4)

# Grafik zeichnen
ggplot(neue_daten,
       aes(x = English.Overall,
           y = yhat)) +
  geom_ribbon(aes(ymin = ki_unten95,
                 ymax = ki_oben95),
            fill = "lightgrey") +
  geom_ribbon(aes(ymin = ki_unten80,
                 ymax = ki_oben80),
            fill = "darkgrey") +
  geom_line() +
  ## Eventuell Datenpunkte hinzufügen
  # geom_point(data = d,
  #           shape = 1,
  #           aes(x = English.Overall,
  #               y = CorrectSpoken)) +
  facet_grid(cols = vars(Sex)) +
  xlab("Ergebnis Englischtest") +
  ylab("Durchschnittliche Leistung\Kognatübersetzen")
```

10.4 Komplexere Interaktionen

Im Prinzip ist es auch möglich, Interaktionen zwischen zwei kontinuierlichen Prädiktoren dem Modell hinzuzufügen. Diese Möglichkeit bespreche ich im Blogeintrag *Interactions between continuous variables* (26.6.2017). Für ein ausführlicheres Beispiel, siehe Vanhove & Berthele (2017).

Auch Interaktionen zwischen drei oder mehr Prädiktoren können modelliert werden. Solche Fälle werden hier nicht behandelt, da ich erstens keinen Zugriff auf einen geeigneten Datensatz

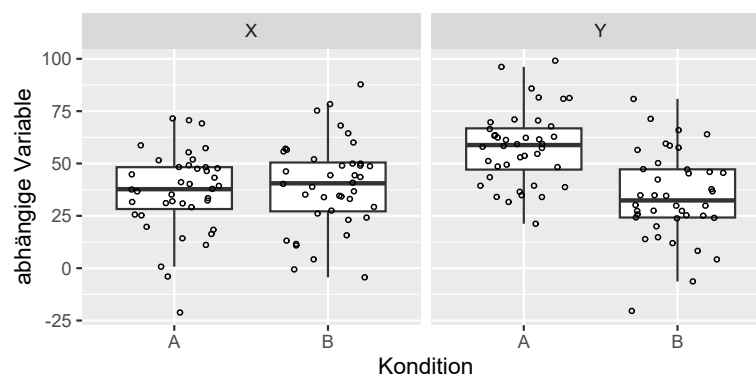


Abbildung 10.11: In diesem simulierten Datensatz gibt es zwar Haupteffekte von A vs. B und von X vs. Y, aber eigentlich schneidet nur die Gruppe mit A und Y wesentlich besser als die anderen ab.

habe und da solche dreifache, vierfache usw. Interaktionen oft schwierig zu erklären sind, wenn man sich nicht bereits mit der Forschungsliteratur auskennt. Für ein Beispiel einer dreifachen Interaktion, siehe Bialystok et al. (2004) (Interaktion zwischen Alter (jung–alt), Kongruenz (kongruent–inkongruent) und Sprachgruppe (monolingual–bilingual) auf Reaktionsgeschwindigkeit).³ Bialystok et al. analysierten ihre Daten übrigens nicht mit dem allgemeinen linearen Modell, da es Abhängigkeiten in den Daten gab: Die Versuchspersonen wurden sowohl in der kongruenten als auch in der inkongruenten Kondition getestet.

10.5 Interaktionen und Haupteffekte interpretieren

Es kann schwierig sein, Haupteffekte zu interpretieren, wenn eine Interaktion vorliegt; am besten basiert man sich hierbei auf einer Grafik. Bei etwa dem Datenmuster in Abbildung 10.11 wäre es vorschnell zu sagen, dass der outcome im Schnitt höher ist bei A als bei B (Haupteffekt von A vs. B) oder dass er höher ist bei Y als bei X (Haupteffekt von X vs. Y), auch wenn das Modell beide Haupteffekte belegen würde: Der Punkt ist ja dass es nur einen Unterschied zu geben scheint, wenn A und Y *gleichzeitig* vorkommen (Interaktion zwischen AB und XY).

Zur Interpretation von *non-cross-over interactions* sei hier auf Wagenmakers et al. (2012) verwiesen. Zusammengefasst: Eine Interaktion in der gemessenen Variablen (z.B. Reaktionsgeschwindigkeit) muss nicht zwingend darauf hindeuten, dass eine Interaktion im hinterliegenden Konstrukt (z.B. kognitiver Kontrolle) vorliegt.

Merksatz: Zur Bedeutung von Regressionsparameter. Das allgemeine lineare Modell ist in erster Linie eine Methode, um die Parameter einer Regressionsgleichung zu schätzen. Diese

³Wie Abelson (1995) erklärt, können Interaktionen oft weggerechnet werden. Zum Beispiel hätten Bialystok et al. (2004) für jede Versuchsperson den Unterschied zwischen den kongruenten und inkongruenten Reaktionszeiten berechnen können und dann die zweifachen Interaktion zwischen Alter und Sprachgruppe untersuchen können. Das Ergebnis wäre genau gleich gewesen, aber die Analyse einfacher und wohl verständlicher.

wiederum ist lediglich ein Hilfsmittel, um Muster in den Daten numerisch zu erfassen. Die Parameter in dieser Gleichung betrachten Sie daher am besten als Buchhaltungsmittel, nicht als den numerischen Ausdruck irgendeiner psychologischen, soziologischen usw. Wahrheit. Gerade bei Interaktionsparametern sollte man sich dessen bewusst sein, dass die Tatsache, dass man in einem Regressionmodell eine Interaktion zwischen zwei Variablen modellieren kann, *nicht* heisst, dass die Konstrukte, die hinter diesen Variablen stecken, miteinander interagieren in der alltäglichen Bedeutung des Wortes.

In diesem Kapitel gibt es keine praktischen Aufgaben, aber Ihr neu angeeignetes Wissen über Interaktionen werden Sie in Aufgabe ?? auf Seite ?? anwenden müssen. Falls es Ihnen sonst langweilig werden sollte, empfehle ich statt einer praktischen Übung die Lektüre von Wagenmakers et al. (2012).

Teil IV

Anhänge

Anhang A

Häufige Fehlermeldungen in R

Probleme immer der Reihe nach lösen! Wenn Ihr ganzer Bildschirm voller roter Fehlermeldungen steht, fangen Sie dann bei der allerersten Fehlermeldung an. Öfters sind die anderen Fehlermeldungen Konsequenzen der ersten.

Debugging. Wenn Sie jemanden um Hilfe bitten, sollten Sie ein reproduzierbares Beispiel, das das Problem illustriert, schreiben. Diese Beispiele sollten möglichst *kurz* sein, siehe <https://stackoverflow.com/q/5963269/1331521>. Oft findet man beim Herstellen eines möglichst kurzen reproduzierbaren Beispiels selber die Lösung. Eine weitere nützliche Technik, um Fehler in Computercode aufzudecken, ist *rubber duck debugging*, siehe <https://rubberduckdebugging.com/>. Und öfters hilft es auch, Feierabend zu machen oder einen Spaziergang zu machen, sodass Sie sich am nächsten Tag mit einem frischen Kopf mit dem Problem beschäftigen können.

object 'x' not found

Sie versuchen ein Objekt abzurufen, das im Arbeitsgedächtnis nicht vorhanden ist. Tippfehler sind hierfür ein typischer Auslöser. Es kann auch vorkommen, dass Sie ein Objekt nicht eingelesen haben oder dass das Objekt anders heisst, als Sie denken.

Beispiel:

```
x <- c(1, 5, 4)
mean(X)

[1] 7
```

In der ersten Zeile wird ein Objekt namens `x` kreiert, aber in der zweiten wird versucht, das Mittel eines Objektes namens `X` zu berechnen. Gross- und Kleinschreibung spielen eine Rolle!

Wenn Sie sich nicht mehr sicher sind, welche Objekte alles im Arbeitsgedächtnis vorhanden

sind, können Sie folgende Funktion verwenden; diese listet alle vorhandenen Objekte auf.

```
ls()
```

Die gleichen Infos können Sie in RStudio im Fenster rechts oben nachschlagen.

could not find function 'x'

Sie wollen eine Funktion verwenden, die nicht besteht oder nicht zugänglich ist. Kontrollieren Sie zuerst, ob Sie den Namen der Funktion richtig eingetippt haben. Wenn es sich um eine Funktion aus einem Erweiterungspaket handelt, sollten Sie ausserdem dieses Paket geladen haben.

Beispiel:

```
r.test(n = 50, r12 = 0.4)
```

```
Error in r.test(n = 50, r12 = 0.4): could not find function "r.test"
```

Die `r.test()`-Funktion ist Teil des Erweiterungspakets `psych`. Entweder sollten Sie das Paket zuerst mit `psych` laden oder Sie sollten dem Funktionsnamen noch `psych::` voranstellen.

Error in library(x) : there is no package called 'x'

Sie versuchen ein Erweiterungspaket zu laden, aber haben dieses noch nicht installiert.

Beispiel:

```
library(ggjoy)
```

```
Error in library(ggjoy): there is no package called 'ggjoy'
```

Lösung: Paket installieren:

```
install.packages("ggjoy")
```

unexpected symbol

Häufige Auslöser für diese Fehlermeldung sind vergessene Kommas.

Beispiel:

```
> library(ggplot2)
> ggplot(data = iris
+       aes(x = Sepal.Width,
Error: unexpected symbol in:
"ggplot(data = iris
      aes"
```

Nach `data = iris` fehlt eine Komma.

Beispiel:

```
> ggplot(data = iris)
+       aes(x = Sepal.Width,
+           y = Sepal.Length)) +
Error: unexpected ')' in:
"       aes(x = Sepal.Width,
           y = Sepal.Length))"
```

Die Klammer nach der ersten Zeile soll eine Komma sein.

Beispiel:

```
> ggplot(data = iris,
+       aes(x = Sepal.Width,
+           y = Sepal.Length))) +
Error: unexpected ')' in:
"       aes(x = Sepal.Width,
           y = Sepal.Length)))"
```

Auf der dritten Zeile steht eine Klammer zu viel.

Es funktioniert nicht und es gibt nicht mal eine Fehlermeldung!

Vermutlich fehlt irgendwo eine Klammer oder etwas Ähnliches.

Beispiel: Dieser Befehl ergibt keine Fehlermeldung, aber produziert auch keine Grafik.

```
> ggplot(data = iris,
+       aes(x = Sepal.Width,
+           y = Sepal.Length) +
+       geom_point()
+)
```

Wie Sie sehen, gibt es nach dem Befehl eine neue Zeile, die mit `+` anfängt. Das heisst, dass der Befehl noch nicht abgeschlossen ist: Nach abgeschlossenen Befehlen folgen Zeilen, die mit `>` anfangen. In diesem Fall ist der Auslöser eine fehlende Klammer auf der 3. Zeile: Die letzte Klammer schliesst den Befehl `aes()` ab, aber es fehlt noch eine Klammer, um den Befehl `ggplot()` abzuschliessen. So funktioniert es schon:

```
> ggplot(data = iris,
+       aes(x = Sepal.Width,
+           y = Sepal.Length)) +
+       geom_point()
>
```

Bemerken Sie, dass es nun eine neue Zeile mit > gibt: Der letzte Befehl wurde also ausgeführt.

Weiteres Beispiel: Dieser Befehl ergibt auch keine Fehlermeldung, aber produziert auch keine Grafik.

```
> ggplot(data = iris,
+       aes(x = Sepal.Width,
+       y = Sepal.Length))
> geom_point()
geom_point: na.rm = FALSE
stat_identity: na.rm = FALSE
position_identity
```

Problem: Das Plus-Zeichen nach der 3. Zeile fehlt.

Das Ergebnis einer Berechnung ist 'NA'

Sie wollen einen Mittelwert o.Ä. berechnen, aber die Datenreihe enthält NA-Angaben (*not available*). R betrachtet NA-Angaben als unbekannte Zahlen. Enthält eine Datenreihe eine unbekannte Zahl, dann ist ihr Mittel oder ihre Standardabweichung (usw.) auch unbekannt (NA).

```
x <- c(5, 4, 18, NA, 3)
mean(x)

[1] NA
```

Wenn Sie es für sinnvoll halten, können Sie das Mittel berechnen, ohne die unbekannte Zahl zu berücksichtigen:

```
mean(x, na.rm = TRUE)

[1] 7.5
```

Für mehr Informationen zum Umgang mit fehlenden Daten, siehe Graham (2009).

Anhang B

Softwareversionen

Ich arbeite auf zwei Computern an diesem Skript. Die hier aufgeführten Softwareversionen und die Systemeinstellungen variieren je nachdem, auf welchem Computer ich das Skript das letzte Mal kompiliert habe.

```
devtools::session_info()

- Session info -----
setting  value
version  R version 4.3.1 (2023-06-16 ucrt)
os       Windows 10 x64 (build 19045)
system   x86_64, mingw32
ui       RTerm
language (EN)
collate   English_United Kingdom.utf8
ctype     English_United Kingdom.utf8
tz        Europe/Zurich
date      2024-02-27
pandoc    NA

- Packages -----
package      * version date (UTC) lib source
bit           4.0.5   2022-11-15 [1] CRAN (R 4.3.1)
bit64         4.0.5   2020-08-30 [1] CRAN (R 4.3.1)
cachem        1.0.8   2023-05-01 [1] CRAN (R 4.3.1)
callr         3.7.3   2022-11-02 [1] CRAN (R 4.3.1)
cannonball    * 0.1.1   2024-02-26 [1] Github (janhove/cannonball@fe70eff)
cellranger    1.1.0   2016-07-27 [1] CRAN (R 4.3.1)
cli           3.6.1   2023-03-23 [1] CRAN (R 4.3.1)
colorspace    2.1-0   2023-01-23 [1] CRAN (R 4.3.1)
crayon        1.5.2   2022-09-29 [1] CRAN (R 4.3.1)
```


devtools	2.4.5	2022-10-11	[1]	CRAN	(R 4.3.1)
digest	0.6.33	2023-07-07	[1]	CRAN	(R 4.3.1)
dplyr	* 1.1.2	2023-04-20	[1]	CRAN	(R 4.3.1)
ellipsis	0.3.2	2021-04-29	[1]	CRAN	(R 4.3.1)
evaluate	0.21	2023-05-05	[1]	CRAN	(R 4.3.1)
fansi	1.0.4	2023-01-22	[1]	CRAN	(R 4.3.1)
farver	2.1.1	2022-07-06	[1]	CRAN	(R 4.3.1)
fastmap	1.1.1	2023-02-24	[1]	CRAN	(R 4.3.1)
forcats	* 1.0.0	2023-01-29	[1]	CRAN	(R 4.3.1)
fs	1.6.3	2023-07-20	[1]	CRAN	(R 4.3.1)
generics	0.1.3	2022-07-05	[1]	CRAN	(R 4.3.1)
ggplot2	* 3.4.2	2023-04-03	[1]	CRAN	(R 4.3.1)
glue	1.6.2	2022-02-24	[1]	CRAN	(R 4.3.1)
gridExtra	2.3	2017-09-09	[1]	CRAN	(R 4.3.1)
gtable	0.3.3	2023-03-21	[1]	CRAN	(R 4.3.1)
here	* 1.0.1	2020-12-13	[1]	CRAN	(R 4.3.1)
highr	0.10	2022-12-22	[1]	CRAN	(R 4.3.1)
hms	1.1.3	2023-03-21	[1]	CRAN	(R 4.3.1)
htmltools	0.5.5	2023-03-23	[1]	CRAN	(R 4.3.1)
htmlwidgets	1.6.2	2023-03-17	[1]	CRAN	(R 4.3.1)
httpuv	1.6.11	2023-05-11	[1]	CRAN	(R 4.3.1)
knitr	* 1.43	2023-05-25	[1]	CRAN	(R 4.3.1)
labeling	0.4.2	2020-10-20	[1]	CRAN	(R 4.3.0)
later	1.3.1	2023-05-02	[1]	CRAN	(R 4.3.1)
lattice	0.21-8	2023-04-05	[2]	CRAN	(R 4.3.1)
lifecycle	1.0.3	2022-10-07	[1]	CRAN	(R 4.3.1)
lubridate	* 1.9.2	2023-02-10	[1]	CRAN	(R 4.3.1)
magrittr	2.0.3	2022-03-30	[1]	CRAN	(R 4.3.1)
Matrix	1.5-4.1	2023-05-18	[2]	CRAN	(R 4.3.1)
memoise	2.0.1	2021-11-26	[1]	CRAN	(R 4.3.1)
mgcv	1.8-42	2023-03-02	[2]	CRAN	(R 4.3.1)
mime	0.12	2021-09-28	[1]	CRAN	(R 4.3.0)
miniUI	0.1.1.1	2018-05-18	[1]	CRAN	(R 4.3.1)
mnormt	2.1.1	2022-09-26	[1]	CRAN	(R 4.3.1)
munsell	0.5.0	2018-06-12	[1]	CRAN	(R 4.3.1)
nlme	3.1-162	2023-01-31	[2]	CRAN	(R 4.3.1)
pillar	1.9.0	2023-03-22	[1]	CRAN	(R 4.3.1)
pkgbuild	1.4.2	2023-06-26	[1]	CRAN	(R 4.3.1)
pkgconfig	2.0.3	2019-09-22	[1]	CRAN	(R 4.3.1)
pkgload	1.3.2.1	2023-07-08	[1]	CRAN	(R 4.3.1)
plotrix	3.8-4	2023-11-10	[1]	CRAN	(R 4.3.2)
prettyunits	1.1.1	2020-01-24	[1]	CRAN	(R 4.3.1)

processx	3.8.2	2023-06-30	[1]	CRAN	(R 4.3.1)
profvis	0.3.8	2023-05-02	[1]	CRAN	(R 4.3.1)
promises	1.2.0.1	2021-02-11	[1]	CRAN	(R 4.3.1)
ps	1.7.5	2023-04-18	[1]	CRAN	(R 4.3.1)
psych	2.4.1	2024-01-18	[1]	CRAN	(R 4.3.2)
purrr	* 1.0.1	2023-01-10	[1]	CRAN	(R 4.3.1)
R6	2.5.1	2021-08-19	[1]	CRAN	(R 4.3.1)
RColorBrewer	* 1.1-3	2022-04-03	[1]	CRAN	(R 4.3.0)
Rcpp	1.0.11	2023-07-06	[1]	CRAN	(R 4.3.1)
readr	* 2.1.4	2023-02-10	[1]	CRAN	(R 4.3.1)
readxl	* 1.4.3	2023-07-06	[1]	CRAN	(R 4.3.1)
remotes	2.4.2.1	2023-07-18	[1]	CRAN	(R 4.3.1)
rlang	1.1.1	2023-04-28	[1]	CRAN	(R 4.3.1)
rprojroot	2.0.3	2022-04-02	[1]	CRAN	(R 4.3.1)
rstudioapi	0.15.0	2023-07-07	[1]	CRAN	(R 4.3.1)
scales	1.2.1	2022-08-20	[1]	CRAN	(R 4.3.1)
sessioninfo	1.2.2	2021-12-06	[1]	CRAN	(R 4.3.1)
shiny	1.7.4.1	2023-07-06	[1]	CRAN	(R 4.3.1)
stringi	1.7.12	2023-01-11	[1]	CRAN	(R 4.3.0)
stringr	* 1.5.0	2022-12-02	[1]	CRAN	(R 4.3.1)
tibble	* 3.2.1	2023-03-20	[1]	CRAN	(R 4.3.1)
tidyr	* 1.3.0	2023-01-24	[1]	CRAN	(R 4.3.1)
tidyselect	1.2.0	2022-10-10	[1]	CRAN	(R 4.3.1)
tidyverse	* 2.0.0	2023-02-22	[1]	CRAN	(R 4.3.1)
timechange	0.2.0	2023-01-11	[1]	CRAN	(R 4.3.1)
tzdb	0.4.0	2023-05-12	[1]	CRAN	(R 4.3.1)
urlchecker	1.0.1	2021-11-30	[1]	CRAN	(R 4.3.1)
usethis	2.2.3	2024-02-19	[1]	CRAN	(R 4.3.2)
utf8	1.2.3	2023-01-31	[1]	CRAN	(R 4.3.1)
vctrs	0.6.3	2023-06-14	[1]	CRAN	(R 4.3.1)
vroom	1.6.3	2023-04-28	[1]	CRAN	(R 4.3.1)
withr	2.5.0	2022-03-03	[1]	CRAN	(R 4.3.1)
xfun	0.39	2023-04-20	[1]	CRAN	(R 4.3.1)
xtable	1.8-4	2019-04-21	[1]	CRAN	(R 4.3.1)

[1] C:/Users/VanhoveJ/AppData/Local/R/win-library/4.3

[2] C:/Program Files/R/R-4.3.1/library

Literaturverzeichnis

- Abelson, Robert P. 1995. *Statistics as principled argument*. New York, NY: Psychology Press.
- Albers, Casper J., Henk A. L. Kiers & Don van Ravenzwaaij. 2018. Credible confidence: A pragmatic view on the frequentist vs Bayesian debate. *Collabra: Psychology* 4(1). 31. doi: 10.1525/collabra.149.
- Baayen, R. Harald. 2008. *Analyzing linguistic data: A practical introduction to statistics using R*. Cambridge: Cambridge University Press.
- Baayen, R. Harald & Maja Linke. 2020. Generalized additive mixed models. In Magali Paquot & Stefan Th. Gries (eds.), *A practical handbook of corpus linguistics*, 563–591. Cham, Switzerland: Springer Nature. doi:10.1107/978-3-030-46216-1_23.
- Baguley, Thom. 2009. Standardized or simple effect size: What should be reported? *British Journal of Psychology* 100(3). 603–617. doi:10.1348/000712608X377117.
- Berthele, Raphael. 2012. The influence of code-mixing and speaker information on perception and assessment of foreign language proficiency: An experimental study. *International Journal of Bilingualism* 16(4). 453–466. doi:10.1177/1367006911429514.
- Bialystok, Ellen, Fergus I. M. Craik, Raymond Klein & Mythili Viswanathan. 2004. Bilingualism, aging, and cognitive control: Evidence from the Simon task. *Psychology and Aging* 19(2). 290–303. doi:10.1037/0882-7974.19.2.290.
- Broman, Karl W. & Kara H. Woo. 2017. Data organization in spreadsheets. *The American Statistician* doi:10.1080/00031305.2017.1375989.
- Caruso, Eugene M., Kathleen D Vohs, Brittani Baxter & Adam Waytz. 2013. Mere exposure to money increases endorsement of free-market systems and social inequality. *Journal of Experimental Psychology: General* 142(2). 301–306. doi:10.1037/a0029288.
- Cohen, Jacob, Patricia Cohen, Stephen G. West & Leona S. Aiken. 2003. *Applied multiple regression/correlation analysis for the behavioral sciences*. Mahwah, NJ: Erlbaum.
- DeKeyser, Robert, Iris Alfi-Shabtay & Dorit Ravid. 2010. Cross-linguistic evidence for the nature of age effects in second language acquisition. *Applied Psycholinguistics* 31. 413–438. doi:10.1017/S0142716410000056.

- Desgrippes, Magalie, Amelia Lambelet & Jan Vanhove. 2017. The development of argumentative and narrative writing skills in Portuguese heritage speakers in Switzerland (HELASCOT project). In Raphael Berthele & Amelia Lambelet (eds.), *Heritage and school language literacy development in migrant children: Interdependence or independence?*, 83–96. Bristol: Multilingual Matters. doi:10.21832/9781783099054-006.
- DiCiccio, Thomas J. & Bradley Efron. 1996. Bootstrap confidence intervals. *Statistical Science* 11(3). 189–212. doi:10.1214/ss/1032280214.
- Dümbgen, Lutz. 2016. *Einführung in die Statistik*. Basel: Birkhäuser. doi:10.1007/978-3-0348-0004-4.
- Efron, Bradley. 1979. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics* 7(1). 1–26. doi:10.1214/aos/1176344552.
- Efron, Bradley & Robert J. Tibshirani. 1993. *An introduction to the bootstrap*. Boca Raton, FL: Chapman & Hall/CRC.
- Ehrenberg, Andrew S. C. 1982. *A primer in data reduction: An introductory statistics textbook*. Chichester: Wiley.
- Eriksen, Barbara A. & Charles W. Eriksen. 1974. Effects of noise letters upon the identification of a target letter in a nonsearch task. *Perception & Psychophysics* 16. 143–149. doi:10.3758/BF03203267.
- Faraway, Julian J. 2005. *Linear models with R*. Boca Raton, FL: Chapman & Hall/CRC.
- Fox, John. 2003. Effect displays in R for generalised linear models. *Journal of Statistical Software* 8. 1–27. doi:10.18637/jss.v008.i15.
- Gelman, Andrew & Jennifer Hill. 2007. *Data analysis using regression and multilevel/hierarchical models*. New York, NY: Cambridge University Press.
- Gelman, Andrew & Hal Stern. 2006. The difference between “significant” and “not significant” is not itself statistically significant. *The American Statistician* 60(4). 328–331. doi:10.1198/000313006X152649.
- Graham, John W. 2009. Missing data analysis: Making it work in the real world. *Annual Review of Psychology* 60. 549–576. doi:10.1146/annurev.psych.58.110405.085530.
- Healy, Kieran. 2019. *Data visualization: A practical introduction*. Princeton, NJ: Princeton University Press. Also freely available online from <https://socviz.co/>.
- Hesterberg, Tim C. 2015. What teachers should know about the bootstrap: Resampling in the undergraduate statistics curriculum. *The American Statistician* 69(4). 371–386. doi:10.1080/00031305.2015.1089789.
- Hoekstra, Rink, Richard D. Morey, Jeffrey N. Rouder & Eric-Jan Wagenmakers. 2014. Robust misinterpretation of confidence intervals. *Psychonomic Bulletin & Review* 21(5). 1157–1164. doi:10.3758/s13423-013-0572-3.

- Huff, Darrell. 1954. *How to lie with statistics*. New York: Norton.
- Klein, Olivier, Tom E. Hardwicke, Frederik Aust, Johannes Breuer, Henrik Danielsson, Alicia Hofelich Mohr, Hans IJzerman, Gustav Nilsson, Wolf Vanpaemel & Michael C. Frank. 2018. A practical guide for transparency in psychological science. *Collabra: Psychology* 4(1). 20. doi:10.1525/collabra.158.
- Klein, Richard A., Kate A. Ratliff, Michelangelo Vianello, Reginald B. Adams Jr., Štěpán Bahník, Michael J. Bernstein, Konrad Bocian et al. 2014. Investigating variation in replicability: A “many labs” replication project. *Social Psychology* 45(3). 142–152. doi:10.1027/1864-9335/a000178.
- Lambelet, Amelia, Raphael Berthele, Magalie Desgrippes, Carlos Pestana & Jan Vanhove. 2017. Testing interdependence in Portuguese heritage speakers in Switzerland: the HELASCOT project. In Raphael Berthele & Amelia Lambelet (eds.), *Heritage and school language literacy development in migrant children: Interdependence or independence?*, 26–33. Bristol: Multilingual Matters. doi:10.21832/9781783099054-003.
- Levenstein, Margaret C. & Jared A. Lyle. 2018. Data: Sharing is caring. *Advances in Methods and Practices in Psychological Science* 1(1). 95–103. doi:10.1177/2515245918758319.
- Morey, Richard D., Rink Hoekstra, Jeffrey N. Rouder, Michael D. Lee & Eric-Jan Wagenmakers. 2016. The fallacy of placing confidence in confidence intervals. *Psychonomic Bulletin & Review* 23(1). 103–123. doi:10.3758/s13423-015-0947-8.
- Nalborczyk, Ladislav, Paul-Christian Bürkner & Donald R. Williams. 2019. Pragmatism should not be a substitute for statistical literacy: A commentary on Albers, Kiers, and van Ravenzwaaij (2018). *Collabra: Psychology* 5(1). 13. doi:10.1525/collabra.197.
- Nieuwenhuis, Sander, Birte U. Forstmann & Eric-Jan Wagenmakers. 2011. Erroneous analyses of interactions in neuroscience: A problem of significance. *Nature Neuroscience* 14. 1105–1107. doi:10.1038/nn.2886.
- Noether, G. E. 1981. Why Kendall tau? *Teaching Statistics* 3(2). 41–43. doi:10.1111/j.1467-9639.1981.tb00422.x.
- Oppenheimer, Daniel M & Benoît Monin. 2009. The retrospective gambler’s fallacy: Unlikely events, constructing the past, and multiple universes. *Judgment and Decision Making* 4(5). 326–334.
- Pestana, Carlos, Amelia Lambelet & Jan Vanhove. 2017. Reading comprehension in Portuguese heritage speakers in Switzerland (HELASCOT project). In Raphael Berthele & Amelia Lambelet (eds.), *Heritage and school language literacy development in migrant children: Interdependence or independence?*, 58–82. Bristol: Multilingual Matters. doi:10.21832/9781783099054-005.
- Poarch, Gregory J., Jan Vanhove & Raphael Berthele. 2019. The effect of bidialectalism on executive function. *International Journal of Bilingualism* 23(2). 612–628. doi:10.1177/1367006918763132.

- Schad, Daniel J., Shravan Vasishth, Sven Hohenstein & Reinhold Kliegl. 2020. How to capitalize on a priori contrasts in linear (mixed) models: A tutorial. *Journal of Memory and Language* 110. doi:10.1016/j.jml.2019.104038.
- Simon, J. Richard. 1969. Reactions toward the source of stimulation. *Journal of Experimental Psychology* 81. 174–176. doi:10.1037/h0027448.
- Slavin, Robert E., Nancy Madden, Margarita Calderón, Anne Chamberlain & Megan Hennessy. 2011. Reading and language outcomes of a multiyear randomized evaluation of transitional bilingual education. *Educational Evaluation and Policy Analysis* 33(1). 47–58. doi:10.3102/0162373711398127.
- Soderberg, Courtney K. 2018. Using OSF to share data: A step-by-step guide. *Advances in Methods and Practices in Psychological Science* 1(1). 115–120. doi:10.1177/2515245918757689.
- Stocker, Ladina. 2017. The impact of foreign accent on credibility: An analysis of cognitive statement ratings in a Swiss context. *Journal of Psycholinguistic Research* 46(3). 617–628. doi:10.1007/s10936-016-9455-x.
- Vanhove, Jan. 2013. The critical period hypothesis in second language acquisition: A statistical critique and a reanalysis. *PLOS ONE* 8. e69172. doi:10.1371/journal.pone.0069172.
- Vanhove, Jan. 2014. *Receptive multilingualism across the lifespan: Cognitive and linguistic factors in cognate guessing*: University of Fribourg dissertation. <http://doc.rero.ch/record/210293>.
- Vanhove, Jan. 2015. Analyzing randomized controlled interventions: Three notes for applied linguists. *Studies in Second Language Learning and Teaching* 5. 135–152. doi:10.14746/ssllt.2015.5.1.7.
- Vanhove, Jan. 2016. The early learning of interlingual correspondence rules in receptive multilingualism. *International Journal of Bilingualism* 20(5). 580–593. doi:10.1177/1367006915573338.
- Vanhove, Jan. 2017. The influence of standard and substandard Dutch on gender assignment in second language German. *Language Learning* 67(2). 431–460. doi:10.1111/lang.12230.
- Vanhove, Jan. 2018. Checking the assumptions of your statistical method without getting paranoid. doi:10.31234/osf.io/zvawb.
- Vanhove, Jan. 2019. Metalinguistic knowledge about the native language and language transfer in gender assignment. *Studies in Second Language Learning and Teaching* 9(2). 397–419. doi:10.14746/ssllt.2019.9.2.7.
- Vanhove, Jan. 2020. When labeling L2 users as nativelike or not, consider classification errors. *Second Language Research* 36(4). 709–724. doi:10.1177/0267658319827055.
- Vanhove, Jan. 2021. Towards simpler and more transparent quantitative research reports. *ITL - International Journal of Applied Linguistics* 172(1). 3–25. doi:10.1075/itl.20010.van.

- Vanhove, Jan & Raphael Berthele. 2017. Interactions between formal distance and participant-related variables in receptive multilingualism. *International Review of Applied Linguistics in Language Teaching* 55(1). 23–40. doi:10.1515/iral-2017-0007.
- Wagenmakers, Eric-Jan, Angelos-Miltiadis Krypotos, Amy H. Criss & Geoff Iverson. 2012. On the interpretation of removable interactions: A survey of the field 33 years after Loftus. *Memory & Cognition* 40(2). 145–160. doi:10.3758/s13421-011-0158-0.
- Weisberg, Sanford. 2005. *Applied linear regression*. Hoboken, NJ: Wiley.
- Weissgerber, Tracey L., Natasa M. Milic, Stacey J. Winham & Vesna D. Garovic. 2015. Beyond bar and line graphs: Time for a new data presentation paradigm. *PLOS Biology* 13(4). e1002128. doi:10.1371/journal.pbio.1002128.
- Wickham, Hadley. 2014. Tidy data. *Journal of Statistical Software* 59. doi:10.18637/jss.v059.i10.
- Wieling, Martijn. 2018. Analyzing dynamic phonetic data using generalized additive mixed modeling: A tutorial focusing on articulatory differences between L1 and L2 speakers of English. *Journal of Phonetics* 70. 86–116.
- Zuur, Alain F., Elena N. Ieno, Neil J. Walker, Anatoly A. Saveliev & Graham M. Smith. 2009. *Mixed effects models and extensions in ecology with R*. New York, NY: Springer.