Package 'LSAmitR'

July 21, 2025

Type Package

Title Daten, Beispiele und Funktionen zu 'Large-Scale Assessment mit R'

Version 1.0-3

Date 2022-05-17

Author Thomas Kiefer [aut, cre],
Alexander Robitzsch [aut],
Matthias Trendtel [aut],
Robert Fellinger [aut]

Maintainer Thomas Kiefer <thomas.kiefer@iqs.gv.at>

Description Dieses R-Paket stellt Zusatzmaterial in Form von Daten, Funktionen und R-Hilfe-Seiten für den Herausgeberband Breit, S. und Schreiner, C. (Hrsg.). (2016). ``Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung." Wien: facultas. (ISBN: 978-3-7089-1343-8, <https://www.iqs.gv.at/themen/bildungsforschung/publikationen/veroeffentlichte-publikationen>) zur Verfügung.

Depends R (>= 2.10)

Imports lme4, Hmisc

Suggests BIFIEsurvey, TAM, miceadds, sirt, mice, pls, WrightMap, irr, lavaan, difR, kerdiest, glmnet, mirt, car, mitml, matrixStats, combinat, xtable, tensor, gtools, plyr, prettyR, gridExtra, lattice

License GPL (>= 3)

URL https://www.iqs.gv.at/themen/bildungsforschung/publikationen/veroeffentlichte-publikationen

Language de

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2022-06-01 07:50:02 UTC

Contents

`LSAmitR-package` *Daten, Beispiele und Funktionen zu 'Large-Scale Assessment mit R'*

Description

Das Bundesinstitut für Bildungsforschung, Innovation und Entwicklung des österreichischen Schulwesens (BIFIE) führt die Überprüfung der Bildungsstandards (BIST-Ü) in Österreich durch. "Large-Scale Assessment mit R" ist ein Handbuch der grundlegenden Methodik, die bei diesen Überprüfungen zum Einsatz kommt. Angefangen bei der Testkonstruktion bis zu Aspekten der Rückmeldung werden die dabei eingesetzten methodischen Verfahren dargestellt und diskutiert sowie deren Anwendung in R anhand von Beispieldatensätzen, die in diesem R-Paket zur Verfügung gestellt werden, illustriert.

Beispiele, die sich durch den Band ziehen, lehnen sich an die BIST-Ü in Englisch im Jahr 2013 an. Die Daten, die den Ausführungen zugrunde liegen, sind jedoch keine Echtdaten und erlauben daher auch keine Rekonstruktion der in den Ergebnisberichten publizierten Kennwerte. Es handelt sich (mindestens) um partiell-synthetische Daten, die reale Kovarianzstrukturen zwischen Kovariaten und den Leistungsdaten abbilden sowie eine Mehrebenenstruktur simulieren, die in den

LSA-Erhebungen typischerweise auftreten. Die Datenmuster können weder als Einzelstücke noch als Ganzes auf tatsächliche Testpersonen, auf Klassen oder Schulen zurückgeführt werden. Ebenso führen Ergebnisse, die in den Ausführungen der einzelnen Kapitel erzielt werden, nicht zu den Datensätzen, die in späteren Kapiteln verwendet werden (z. B. entspricht die Stichprobe, die in Kapitel 2 gezogen wird, nicht jener, deren Testwerte in Kapitel 6 oder Kapitel 7 untersucht werden).

Author(s)

Thomas Kiefer [aut, cre], Alexander Robitzsch [aut], Matthias Trendtel [aut], Robert Fellinger [aut]

Maintainer: Thomas Kiefer <thomas.kiefer@iqs.gv.at>

References

Breit, S. & Schreiner, C. [HG.] (2016). *Large-Scale Assessment mit* R*: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung*. Wien: facultas.

https://www.iqs.gv.at/themen/bildungsforschung/publikationen/veroeffentlichte-publikationen

See Also

Zu Kapitel 0, Konzeption der Überprüfung der Bildungsstandards in Österreich.
Zu Kapitel 1, Testkonstruktion.
Zu Kapitel 2, Stichprobenziehung.
Zu Kapitel 3, Standard-Setting.
Zu Kapitel 4, Differenzielles Itemfunktionieren in Subgruppen.
Zu Kapitel 5, Testdesign.
Zu Kapitel 6, Skalierung und Linking.
Zu Kapitel 7, Statistische Analysen produktiver Kompetenzen.
Zu Kapitel 8, Fehlende Daten und Plausible Values.
Zu Kapitel 9, Fairer Vergleich in der Rückmeldung.
Zu Kapitel 10, Reporting und Analysen.
Zu Kapitel 11, Aspekte der Validierung.

Examples

```
## Not run:
install.packages("LSAmitR", dependencies = TRUE)
library(LSAmitR)
package?LSAmitR
?"Kapitel 7"

data(datenKapitel07)
names(datenKapitel07)
dat <- datenKapitel07$prodRat

## End(Not run)
```

`datenKapitel01` *Illustrationsdaten zu Kapitel 1, Testkonstruktion*

Description

Hier befindet sich die Dokumentation der in Kapitel 1, *Testkonstruktion*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel01)
```

Format

`datenKapitel01` ist eine Liste mit den vier Elementen `pilotScored`, `pilotItems`, `pilotRoh` und `pilotMM`, die einer fiktiven Pilotierung entstammen.

- **pilotScored**: Rekodierte Instrumentendaten der Pilotierung (vgl. `pilotItems`).
  - `sidstud`: Schüleridentifikator.
  - `female`: Geschlecht (`"w"` = weiblich, `"m"` = männlich).
  - `form`: das von der Schülerin/dem Schüler bearbeitete Testheft.
  - `E8RS*`: dichotom und polytom bewertete Itemantworten auf Items `E8RS13151` bis `E8RS7993` (`0:4` = Score der Itemantwort, `8` = Itemantwort "nicht bewertbar", `9` = "omitted response").

```
'data.frame': 2504 obs. of 163 variables:
$ sidstud : int 1052 1057 1058 1064 1068 1073 1074 1076 1078 1080 ...
$ female : chr "w" "w" "w" "w" ...
$ form : chr "PR019" "PR020" "PR021" "PR022" ...
$ E8RS13151: int NA NA NA NA NA NA NA NA NA NA ...
$ E8RS13171: int NA NA 1 NA NA NA NA 1 NA NA ...
$ E8RS13491: int NA NA NA NA 0 NA NA NA NA NA ...
$ E8RS13641: int 0 NA NA NA NA NA 0 NA NA NA ...
[...]
$ E8RS7929: int NA NA 0 NA NA NA NA NA NA NA ...
$ E8RS7940: int NA NA NA NA NA 0 NA NA NA NA ...
$ E8RS7955: int NA 0 NA NA NA NA NA NA 2 NA ...
$ E8RS7993: int NA NA NA 0 NA NA 2 NA NA NA ...
```

- **pilotItems**: Itembank der Pilotierung.
  - `testlet`: Testletname des Items (gleichbedeutend mit zugewiesenem Stimulus).
  - `item`: Itemname.
  - `format`: Antwortformat.
  - `focus`: Fokus des Testitems.

  - focusLabel: Bezeichnung des Fokus des Testitems.
  - topic: Themengebiet des Stimulus.
  - no.Words: Anzahl Wörter im Stimulus.
  - key: Indikator der richtigen Antwort (1:3 = korrekte Antwortoption bei Multiple-Choice Items, A:F = korrekt zuzuordnende Antwortoption bei Matching-Items, "" = korrekte Antworten für Items im Antwortformat "open gap-fill" werden in Form von Coding-Guides ausgebildeten Kodiererinnen/Kodierern vorgelegt; vgl. Buchkapite).
  - maxScore: Maximal zu erreichende Punkte.
  - PR*: Positionen der Items in den Testheften PR001 bis PR056.

```
'data.frame': 320 obs. of 65 variables:
$ testlet : chr "E8RS1315" "E8RS1317" "E8RS1340" "E8RS1349" ...
$ item : chr "E8RS13151" "E8RS13171" "E8RS13401" "E8RS13491" ...
$ format : chr "MC3" "MC3" "MC3" "MC3" ...
$ focus : int 1 1 1 1 1 1 1 1 1 1 ...
$ focusLabel: chr "RFocus1" "RFocus1" "RFocus1" "RFocus1" ...
$ topic : chr "Interkulturelle und landeskundliche Aspekte" "Familie und Freunde"
...
$ no.Words : int 24 24 29 32 10 33 22 41 10 37 ...
$ key : chr "1" "3" "2" "2" ...
$ maxScore : int 1 1 1 1 1 1 1 1 1 1 ...
$ PR001 : int NA NA NA 10 NA NA NA NA NA NA ...
$ PR002 : int 5 NA 6 NA 7 NA NA 8 NA NA ...
$ PR003 : int NA NA NA 6 NA NA NA NA NA NA ...
$ PR004 : int NA NA NA 10 NA NA NA NA NA NA ...
[...]
$ PR054 : int NA NA NA NA NA NA NA NA NA NA ...
$ PR055 : int NA 9 NA NA NA NA 10 NA NA 11 ...
$ PR056 : int NA NA NA NA NA NA NA NA 6 NA ...
```

- **pilotRoh**: Instrumentendaten der Pilotierung mit Roh-Antworten (vgl. pilotItems).
  - sidstud: Schüleridentifikator.
  - female: Geschlecht ("w" = weiblich, "m" = männlich).
  - form: das von der Schülerin/dem Schüler bearbeitete Testheft.
  - E8RS*: Roh-Antworten der Schülerin/des Schülers auf Items E8RS13151 bis E8RS37281 ((8, 9) = für alle Items, wie oben, nicht bewertbare bzw. ausgelassene Itemantwort, 1:3 = gewählte Antwortoption bei Multiple-Choice Items, A:F = zugeordnete Antwortoption bei Matching-Items, 0:1 = von Kodiererinnen/Kodierern bewertete Antworten für Items im Antwortformat "open gap-fill").

```
'data.frame': 2504 obs. of 323 variables:
$ sidstud : int 1052 1057 1058 1064 1068 1073 1074 1076 1078 1080 ...
$ female : chr "w" "w" "w" "w" ...
$ form : chr "PR019" "PR020" "PR021" "PR022" ...
$ E8RS13151: int NA NA NA NA NA NA NA NA NA NA ...
$ E8RS13171: int NA NA 3 NA NA NA NA 3 NA NA ...
$ E8RS13491: int NA NA NA NA 3 NA NA NA NA NA ...
$ E8RS13641: int 2 NA NA NA NA NA 2 NA NA NA ...
```

```
[...]
$ E8RS37163: chr "" "" "" "" ...
$ E8RS37164: chr "" "" "" "" ...
$ E8RS37165: chr "" "" "" "" ...
$ E8RS37281: chr "" "" "" "" ...
```

- **pilotMM**: Multiple-Marking-Datensatz der Pilotierung mit gemeinsamen Bewertungen einer itemweisen Auswahl von Schülerantworten durch alle Kodiererinnen/Kodierer (`0` = falsch, `1` = richtig, `(8, 9)` = wie oben, nicht bewertbare bzw. ausgelassene Itemantwort).
  - `sidstud`: Schüleridentifikator.
  - `item`: Itemnummer.
  - `Coder_1`: Bewertung der Schülerantwort von Kodiererin/Kodierer 1.
  - `Coder_2`: Bewertung der Schülerantwort von Kodiererin/Kodierer 2.
  - `Coder_3`: Bewertung der Schülerantwort von Kodiererin/Kodierer 3.

```
'data.frame': 1200 obs. of 5 variables:
$ sidstud: int 1185 1269 1311 1522 1658 1665 1854 1889 1921 2067 ...
$ item : chr "E8RS46051" "E8RS46051" "E8RS46051" "E8RS46051" ...
$ Coder_1: int 1 1 9 0 0 9 9 1 9 0 ...
$ Coder_2: int 1 1 9 0 0 9 9 1 9 0 ...
$ Coder_3: int 1 1 9 0 0 9 9 1 9 0 ...
```

References

Itzlinger-Bruneforth, U., Kuhn, J.-T. & Kiefer, T. (2016). Testkonstruktion. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 21–50). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe `Kapitel 1`.

`datenKapitel02` *Illustrationsdaten zu Kapitel 2, Stichprobenziehung*

Description

Hier befindet sich die Dokumentation der in Kapitel 2, *Stichprobenziehung*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel02)
```

Format

datenKapitel02 ist eine Liste mit den zwei Elementen schueler und schule, die auf Schulen- und Schülerebene alle für eine Stichprobenziehung und die Berechnung von Stichprobengewichten relevanten Informationen beinhalten.

Diese 51644 Schülerinnen und Schüler in 1327 Schulen – verteilt über vier Strata – stellen die Zielpopulation der im Band durchgeführten Analysen dar.

- **schueler**: Schülerdatensatz.
  - SKZ: Schulenidentifikator ("Schulkennzahl").
  - klnr: Nummer der Klasse innerhalb der Schule.
  - idclass: Klassenidentifikator.
  - idstud: Schüleridentifikator.
  - female: Geschlecht (1 = weiblich, 0 = männlich).
  - Stratum: Stratum der Schule. (1 : 4 = Stratum 1 bis Stratum 4; für eine genauere Beschreibung der Strata, siehe Buchkapitel).
  - teilnahme: Information über die Teilnahme der Schülerin/ des Schülers an der Erhebung (1 = nimmt teil, 0 = nimmt nicht teil). Information ist erst zum Zeitpunkt der Erhebung vorhanden (nicht schon bei der Stichprobenziehung) und wird zur Berechnung der Stichprobengewichte mit Ausfalladjustierung herangezogen (siehe Buchkapitel, Unterabschnitt 2.4.4).

```
'data.frame': 51644 obs. of 7 variables:
$ SKZ : int [1:51644] 10001 10001 10001 10001 10001 10001 10001 10001 10001 10001 ...
$ klnr : int [1:51644] 1 1 1 1 1 1 1 1 1 1 ...
$ idclass : int [1:51644] 1000101 1000101 1000101 1000101 1000101 1000101 1000101
1000101 ...
$ idstud : int [1:51644] 100010101 100010102 100010103 100010104 100010105 100010106
100010107 ...
$ female : int [1:51644] 1 0 0 0 0 1 0 1 0 1 ...
$ Stratum : int [1:51644] 1 1 1 1 1 1 1 1 1 1 ...
$ teilnahme: int [1:51644] 1 1 1 1 0 1 1 1 1 1 ...
```

- **schule**: Schulendatensatz.
  - index: Laufparameter.
  - SKZ: Schulenidentifikator ("Schulkennzahl").
  - stratum: Stratum der Schule. (1 : 4 = Stratum 1 bis Stratum 4; für eine genauere Beschreibung der Strata, siehe Buchkapitel).
  - NSchueler: Anzahl Schüler/innen in der 4. Schulstufe der Schule.
  - NKlassen: Anzahl Klassen in der 4. Schulstufe der Schule.

```
'data.frame': 1327 obs. of 5 variables:
$ index : int [1:1327] 1 2 3 4 5 6 7 8 9 10 ...
$ SKZ : int [1:1327] 10204 10215 10422 11017 10257 10544 10548 10846 11127 10126 ...
$ stratum : int [1:1327] 1 1 1 1 1 1 1 1 1 1 ...
$ NSchueler: int [1:1327] 8 9 9 9 10 10 10 10 10 11 ...
$ NKlassen : int [1:1327] 1 1 1 1 1 1 1 2 1 1 ...
```

References

George, A. C., Oberwimmer, K. & Itzlinger-Bruneforth, U. (2016). Stichprobenziehung. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 51–81). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 2.

`datenKapitel03` *Illustrationsdaten zu Kapitel 3, Standard-Setting*

Description

Hier befindet sich die Dokumentation der in Kapitel 3, *Standard-Setting*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel03)
```

Format

`datenKapitel03` ist eine Liste mit den vier Elementen `ratings`, `bookmarks`, `sdat` und `productive`, die Daten zu verschiedenen Methoden eines Standard-Settings beinhalten.

Normierte Personen- und Itemparameter entstammen einer Vorgängerstudie, in der die Parameter für das jeweils zu betrachtende Testinstrument auf die Berichtsmetrik transformiert wurden (vgl. Kapitel 5, *Testdesign*, und Kapitel 6, *Skalierung und Linking*, im Band).

- **ratings**: Daten aus der IDM-Methode (siehe Buchkapitel, Unterabschnitt 3.2.2).
  - `task`: Itemnummer.
  - `Norm_rp23`: Itemparameter auf der Berichtsmetrik.
  - `Seite_OIB`: Seitenzahl im OIB.
  - `R01...R12`: Von der jeweiligen Expertin/dem jeweiligen Experten (Rater/in) zugeordnete Kompetenzstufe des Items.

```
'data.frame': 60 obs. of 15 variables:
$ task : chr [1:60] "E8RS89991" "E8RS14021" "E8RS16031" "E8RS14171" ...
$ Norm_rp23: num [1:60] 376 396 396 413 420 ...
$ Seite_OIB: int [1:60] 1 2 3 4 5 6 7 8 9 10 ...
$ R01 : int [1:60] 1 1 1 1 1 1 1 1 1 1 ...
$ R02 : int [1:60] 1 1 1 2 1 2 2 1 2 2 ...
$ R03 : int [1:60] 1 2 1 2 1 2 2 1 1 2 ...
$ R04 : int [1:60] 1 1 1 1 2 1 1 1 2 1 ...
```

```
$ R05 : int [1:60] 2 2 1 2 1 1 2 1 2 2 ...
$ R06 : int [1:60] 1 1 1 1 2 1 2 1 2 2 ...
$ R07 : int [1:60] 1 1 1 1 1 1 1 1 1 2 ...
$ R08 : int [1:60] 2 2 1 2 2 2 2 1 2 2 ...
$ R09 : int [1:60] 2 1 1 1 1 1 2 1 2 2 ...
$ R10 : int [1:60] 1 2 1 1 1 1 1 1 2 1 ...
$ R11 : int [1:60] 2 2 1 1 2 2 2 1 2 1 ...
$ R12 : int [1:60] 1 2 1 2 3 2 2 1 1 2 ...
```

- **bookmarks**: Daten aus der Bookmark-Methode (siehe Buchkapitel, Unterabschnitt 3.2.3).
  - `Rater`: Rateridentifikator der Expertin/des Experten im Panel.
  - `Cut1`: Bookmark der Expertin/des Experten in Form einer Seite im OIB, wo ein Schüler an der Grenze zwischen der ersten und zweiten Stufe das Item nicht mehr sicher lösen könnte (für eine genauere Beschreibung der Stufen, siehe Buchkapitel).
  - `Cut2`: Entsprechender Bookmark für die Grenze zwischen zweiter und dritter Stufe.

```
'data.frame': 12 obs. of 3 variables:
$ Rater: chr [1:12] "R01" "R02" "R03" "R04" ...
$ Cut1 : int [1:12] 6 4 6 2 4 4 4 4 3 6 ...
$ Cut2 : int [1:12] 45 39 39 45 39 30 39 39 44 45 ...
```

- **sdat**: Plausible Values zum Berichten von Impact Data (siehe Buchkapitel, Unterabschnitt 3.2.4).
  - `pid`: Schüleridentifikator.
  - `studwgt`: Stichprobengewicht der Schülerin/des Schülers (vgl. Kapitel 2, *Stichprobenziehung*, im Band).
  - `TPV1...TPV10`: Plausible Values der Schülerin/des Schülers auf der Berichtsmetrik (vgl. Kapitel 8, *Fehlende Daten und Plausible Values*, im Band).

```
'data.frame': 3500 obs. of 12 variables:
$ pid : int [1:3500] 1 2 3 4 5 6 7 8 9 10 ...
$ studwgt: num [1:3500] 0.978 0.978 0.978 0.978 0.978 ...
$ TPV1 : num [1:3500] 635 562 413 475 427 ...
$ TPV2 : num [1:3500] 601 558 409 452 462 ...
$ TPV3 : num [1:3500] 512 555 383 444 473 ...
$ TPV4 : num [1:3500] 675 553 375 473 454 ...
$ TPV5 : num [1:3500] 595 553 384 471 457 ...
$ TPV6 : num [1:3500] 593 557 362 490 501 ...
$ TPV7 : num [1:3500] 638 518 292 460 490 ...
$ TPV8 : num [1:3500] 581 493 306 467 477 ...
$ TPV9 : num [1:3500] 609 621 333 448 462 ...
$ TPV10 : num [1:3500] 573 634 406 537 453 ...
```

- **productive**: Daten aus der Contrasting-Groups-Methode (siehe Buchkapitel, Unterabschnitt 3.3.2).
  - `Script`: Nummer des Schülertexts.
  - `Performance`: Personenparameter der Schülerin/des Schülers auf der Berichtsmetrik.

- `R01...R10`: Von der jeweiligen Expertin/dem jeweiligen Experten (Rater/in) zugeordnete Kompetenzstufe der Performanz (`0` = untere Stufe, `1` = obere Stufe; für eine genauere Beschreibung der Stufen, siehe Buchkapitel).

```
'data.frame': 45 obs. of 12 variables:
$ Script : int [1:45] 1 2 3 4 5 6 7 8 9 10 ...
$ Performance: num [1:45] 211 260 269 308 321 ...
$ R01 : int [1:45] 1 0 0 1 0 0 0 0 0 0 ...
$ R02 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R03 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R04 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R05 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R06 : int [1:45] 1 0 0 0 0 0 1 0 0 0 ...
$ R07 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R08 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R09 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
$ R10 : int [1:45] 0 0 0 0 0 0 0 0 0 0 ...
```

References

Luger-Bazinger, C., Freunberger, R. & Itzlinger-Bruneforth, U. (2016). Standard-Setting. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit* R*: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 83–110). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe `Kapitel 3`.

`datenKapitel04` *Illustrationsdaten zu Kapitel 4, Differenzielles Itemfunktionieren in Subgruppen*

Description

Hier befindet sich die Dokumentation der in Kapitel 4, *Differenzielles Itemfunktionieren in Subgruppen*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel04)
```

Format

datenKapitel04 ist eine Liste mit den drei Elementen dat, dat.th1 und ibank.

- **dat**: Dichotome Itemantworten von 9884 Schülerinnen und Schülern im Multiple-Matrix-Design mit Gruppierungsmerkmal.
  - idstud: Schüleridentifikator.
  - AHS: Besuch einer allgemeinbildenden höheren Schulen (AHS = 1), bzw. allgemeinbildenden Pflichtschule (AHS = 0).
  - E8RS*: dichotom bewertete Itemantworten zu Items E8RS01661 bis E8RS79931.

```
'data.frame': 9884 obs. of 52 variables:
$ idstud : int [1:9884] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ AHS : int [1:9884] 0 0 0 0 0 0 0 0 0 0 ...
$ E8RS01661: int [1:9884] 0 NA 1 NA 0 0 NA 1 NA 0 ...
$ E8RS02011: int [1:9884] 0 NA 0 NA 0 0 NA 0 NA 0 ...
$ E8RS02201: int [1:9884] NA 1 NA 1 NA NA 1 NA 1 NA ...
[...]
$ E8RS79641: int [1:9884] NA 0 0 0 0 0 NA NA 0 NA ...
$ E8RS79931: int [1:9884] 0 NA NA NA NA NA 0 1 NA 0 ...
```

- **dat.th1**: Teildatensatz mit Itemantworten der Subgruppe von 1636 Schülerinnen und Schülern, die das erste Testheft (vgl. ibank) bearbeitet haben.
  - idstud: Schüleridentifikator.
  - AHS: Besuch einer allgemeinbildenden höheren Schulen (AHS = 1), bzw. allgemeinbildenden Pflichtschule (AHS = 0).
  - E8RS*: dichotom bewertete Itemantworten zu Items E8RS01661 bis E8RS79931.

```
'data.frame': 1636 obs. of 27 variables:
$ idstud : int [1:1636] 10010109 10010111 10020101 10020113 10020114 10030110 ...
$ AHS : int [1:1636] 0 0 0 0 0 0 0 0 0 0 ...
$ E8RS01661: int [1:1636] 1 0 0 1 0 1 0 0 0 0 ...
$ E8RS02011: int [1:1636] 0 0 0 1 0 0 1 0 0 1 ...
$ E8RS02421: int [1:1636] 0 0 0 0 0 1 0 0 0 1 ...
[...]
$ E8RS28551: int [1:1636] 1 0 1 0 0 0 1 1 0 0 ...
$ E8RS79931: int [1:1636] 1 0 0 0 0 0 0 0 0 1 ...
```

- **ibank**: Beispielhafte Itembank mit klassifizierenden Item-Informationen (vgl. Kapitel 1, *Testkonstruktion*, im Band).
  - task: Itemname.
  - format: Antwortformat des Items.
  - focus: Fokuskategorie des Items.
  - itemnr: Itemidentifikator.

```
'data.frame': 50 obs. of 4 variables:
$ task : chr "E8RS01661" "E8RS02011" "E8RS02201" "E8RS02231" ...
$ format : chr "MC4" "MC4" "MC4" "MC4" ...
```

```
$ focus : int 0 0 0 0 0 0 0 0 0 0 ...
$ itemnr : int 1661 2011 2201 2231 2251 2421 2461 2891 2931 3131 ...
```

References

Trendtel, M., Schwabe, F. & Fellinger, R. (2016). Differenzielles Itemfunktionieren in Subgruppen. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 111–147). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 4.

`datenKapitel05` *Illustrationsdaten zu Kapitel 5, Testdesign*

Description

Hier befindet sich die Dokumentation der in Kapitel 5, *Testdesign*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel05)
```

Format

`datenKapitel05` ist eine Liste mit den sechs Elementen `tdItembank`, `tdBib2d`, `tdBibPaare`, `tdExItembank`, `tdExBib2d` und `tdExBibPaare`, die sowohl für die Umsetzung im Kapitel als auch für die Übungsaufgaben die relevanten Informationen auf Itemebene in Form einer Itembank und Zwischenergebnisse aus dem Blockdesign für die Weiterverarbeitung beinhalten.

- **tdItembank**: Itembank für den Testdesignprozess, bestehend aus 286 dichotomen und polytomen Items.
  - `testlet`: Testletname des Items (gleichbedeutend mit zugewiesenem Stimulus).
  - `itemnr`: Itemidentifikator.
  - `task`: Itemname.
  - `format`: Antwortformat.
  - `focus`: Fokuskategorie des Items.
  - `focus.label`: Bezeichnung des Fokus.
  - `topic`: Themengruppe des Inhalts des zum Item gehörenden Stimulus.
  - `audiolength`: Länge der Tonaufnahme in Sekunden.

  - `RelFreq`: Item-Schwierigkeit (genauer: aus Pilotierung gewonnener Erwartungswert gewichtet mit höchstem erreichbaren Punktewert bei dem Item; vgl. Kapitel 1, *Testkonstruktion*, im Band).
  - `rpb.WLE`: Item-Trennschärfe (genauer: Punktbiseriale Korrelation der Itemantworten mit dem Weighted Likelihood Personenschätzer (WLE); vgl. Kapitel 1 und Kapitel 6, *Skalierung und Linking*, im Band).
  - `uniformDIF`: Uniformes Differenzielles Itemfunktionieren (vgl. Kapitel 4, *Differenzielles Itemfunktionieren in Subgruppen*, im Band).
  - `DIF.ETS`: Klassifikation des uniform DIF nach ETS (vgl. Kapitel 4 im Band).
  - `IIF_380`: Wert der Fisher-Iteminformationsfunktionen am Skalenwert 380 (vgl. Kapitel 6 im Band).
  - `IIF_580`: Wert der Fisher-Iteminformationsfunktionen am Skalenwert 580.

```
'data.frame': 286 obs. of 14 variables:
$ testlet : chr [1:286] "E8LS0127" "E8LS0128" "E8LS0132" "E8LS0135" ...
$ itemnr : int [1:286] 127 128 132 135 139 141 142 144 145 147 ...
$ task : chr [1:286] "E8LS0127" "E8LS0128" "E8LS0132" "E8LS0135" ...
$ format : chr [1:286] "MC4" "MC4" "MC4" "MC4" ...
$ focus : int [1:286] 0 2 2 5 2 5 2 4 2 5 ...
$ focus.label: chr [1:286] "LFocus0" "LFocus2" "LFocus2" "LFocus5" ...
$ topic : chr [1:286] "Körper und Gesundheit" "Gedanken, Empfindungen und Gefühle"
...
$ audiolength: int [1:286] 47 46 39 62 51 30 44 28 50 23 ...
$ RelFreq : num [1:286] 0.71 0.314 0.253 0.847 0.244 ...
$ rpb.WLE : num [1:286] 0.516 0.469 0.285 0.54 0.352 ...
$ uniformDIF : num [1:286] 0.115726 0.474025 0.11837 0.083657 -0.000051 ...
$ DIF.ETS : chr [1:286] "A+" "B+" "A+" "A+" ...
$ IIF_380 : num [1:286] 0.4073 0.1542 0.0708 0.4969 0.0611 ...
$ IIF_580 : num [1:286] 0.157 0.508 0.277 0.26 0.148 ...
```

- **tdBib2d**: Vollständiges durch den BIB-Design-Algorithmus erzeugtes Itemblock-Design (vgl. Tabelle 5.3) in tabellarischer Aufstellung mit 30 Testheften (Zeilen), 6 Positionen (Spalten) und 30 Itemblöcken (Zelleneinträge).

```
'data.frame': 30 obs. of 6 variables:
$ V1: int [1:30] 12 5 6 7 3 1 17 4 18 13 ...
$ V2: int [1:30] 2 11 9 4 10 8 15 17 7 26 ...
$ V3: int [1:30] 7 6 10 12 1 5 20 15 17 8 ...
$ V4: int [1:30] 11 9 3 2 8 4 13 5 22 7 ...
$ V5: int [1:30] 10 7 2 9 4 12 6 18 13 1 ...
$ V6: int [1:30] 3 8 1 5 6 11 16 27 14 24 ...
```

- **tdBibPaare**: Ergebnis des BIB-Design-Algorithmus als Blockpaare, wobei die Zelleneinträge die paarweisen Auftretenshäufigkeiten des Zeilenblocks mit dem Spaltenblock im Design anzeigen.

```
'data.frame': 30 obs. of 30 variables:
$ V1 : int [1:30] 6 1 2 2 1 2 1 3 1 2 ...
$ V2 : int [1:30] 1 6 2 2 1 1 3 0 3 2 ...
$ V3 : int [1:30] 2 2 6 1 0 3 1 2 2 5 ...
```

```
[...]
$ V29: int [1:30] 0 0 1 1 0 1 0 0 1 2 ...
$ V30: int [1:30] 1 1 1 0 0 0 0 1 0 1 ...
```

- **tdExItembank**: Beispiel-Itembank für den Testdesignprozess in den Übungsaufgaben zum Kapitel.
  - task: Itemname.
  - format: Antwortformat.
  - focus: Fokuskategorie des Items.
  - p: Item-Leichtigkeit (genauer: in der Pilotierung beobachtete relative Lösungshäufigkeit für dichotome Items).
  - p_cat: Dreistufige Kategorisierung der Schwierigkeit.
  - itemdiff: Rasch-kalibrierte Itemparameter.
  - bearbeitungszeit: Geschätzte mittlere Bearbeitungszeit des Items.

```
'data.frame': 250 obs. of 7 variables:
$ task : chr [1:250] "M80003" "M80004" "M80006" "M80007" ...
$ format : chr [1:250] "ho" "MC4" "MC4" "ho" ...
$ focus : int [1:250] 1 4 4 2 3 4 1 2 3 3 ...
$ p : num [1:250] 0.84 0.56 0.34 0.45 0.2 0.42 0.77 0.42 0.34 0.71 ...
$ p_cat : chr [1:250] "leicht" "mittel" "mittel" "mittel" ...
$ itemdiff : int [1:250] 404 570 676 622 761 636 457 636 676 494 ...
$ bearbeitungszeit: int [1:250] 90 60 90 120 90 150 90 30 120 90 ...
```

- **tdExBib2d**: Vollständiges Itemblock-Design zur Weiterverarbeitung in den Übungsaufgaben zum Kapitel in tabellarischer Aufstellung mit 10 Testheften (Zeilen), 4 Positionen (Spalten) und 10 Itemblöcken (Zelleneinträge).

```
'data.frame': 10 obs. of 4 variables:
$ V1: int [1:10] 1 9 8 2 10 4 7 3 5 6
$ V2: int [1:10] 10 6 7 8 4 1 9 5 3 2
$ V3: int [1:10] 6 10 9 1 5 2 3 8 4 7
$ V4: int [1:10] 7 8 4 3 9 6 1 10 2 5
```

- **tdExBibPaare**: Itemblock-Design zur Weiterverarbeitung in den Übungsaufgaben in der Darstellung als Blockpaare, wobei die Zelleneinträge die paarweisen Auftretenshäufigkeiten des Zeilenblocks mit dem Spaltenblock im Design anzeigen.

```
'data.frame': 10 obs. of 10 variables:
$ V1 : int [1:10] 4 2 2 1 0 2 2 1 1 1
$ V2 : int [1:10] 2 4 2 2 2 2 1 1 0 0
$ V3 : int [1:10] 2 2 4 1 2 0 1 2 1 1
$ V4 : int [1:10] 1 2 1 4 2 1 1 1 2 1
$ V5 : int [1:10] 0 2 2 2 4 1 1 1 1 2
$ V6 : int [1:10] 2 2 0 1 1 4 2 1 1 2
$ V7 : int [1:10] 2 1 1 1 1 2 4 1 2 1
$ V8 : int [1:10] 1 1 2 1 1 1 1 4 2 2
$ V9 : int [1:10] 1 0 1 2 1 1 2 2 4 2
$ V10: int [1:10] 1 0 1 1 2 2 1 2 2 4
```

References

Kiefer, T., Kuhn, J.-T. & Fellinger, R. (2016). Testdesign. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 149–184). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 5.

`datenKapitel06` *Illustrationsdaten zu Kapitel 6, Skalierung und Linking*

Description

Hier befindet sich die Dokumentation der in Kapitel 6, *Skalierung und Linking*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel06)
```

Format

`datenKapitel06` ist eine Liste mit den fünf Elementen `dat`, `itembank`, `datTH1`, `itembankTH1` und `normdat`.

- **dat**: Dichotome und polytome Itemantworten von 9885 Schülerinnen und Schülern im Multiple-Matrix-Design mit Stichprobengewichten und Testheftinformation.
  - `index`: Laufindex.
  - `idstud`: Schüleridentifikator.
  - `wgtstud`: Stichprobengewicht der Schülerin/des Schülers (vgl. Kapitel 2, *Stichprobenziehung*, im Band).
  - `th`: Bearbeitetes Testheft.
  - `I1...I50`: Itemantworten.

```
'data.frame': 9885 obs. of 54 variables:
$ index : int [1:9885] 1 2 3 4 5 6 7 8 9 10 ...
$ idstud : int [1:9885] 10010101 10010102 10010103 10010105 10010106 10010107 10010108
...
$ wgtstud: num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
$ th : chr [1:9885] "ER04" "ER03" "ER05" "ER02" ...
$ I1 : int [1:9885] 0 NA 1 NA 0 0 NA 1 NA 0 ...
$ I2 : int [1:9885] 0 NA 0 NA 0 0 NA 0 NA 0 ...
$ I3 : int [1:9885] NA 1 NA 1 NA NA 1 NA 1 NA ...
[...]
```

```
$ I49 : int [1:9885] 0 NA NA 4 NA NA 3 NA 3 NA ...
$ I50 : int [1:9885] NA 0 0 NA 1 2 NA 0 NA 2 ...
```

- **itembank**: Den Instrumentendaten zugrundeliegende Itembank mit klassifizierenden Item-Informationen (vgl. Kapitel 1, Testkonstruktion, im Band).
  - `Item`: Itemname.
  - `format`: Antwortformat des Items.
  - `focus`: Fokuskategorie des Items.
  - `itemnr`: Itemidentifikator.
  - `N.subI`: Anzahl Subitems.

```
'data.frame': 50 obs. of 5 variables:
$ Item : chr [1:50] "I1" "I2" "I3" "I4" ...
$ format : chr [1:50] "MC4" "MC4" "MC4" "MC4" ...
$ focus : int [1:50] 0 0 0 0 0 0 0 0 0 0 ...
$ itemnr : int [1:50] 1661 2011 2201 2231 2251 2421 2461 2891 2931 3131 ...
$ N.subI : int [1:50] 1 1 1 1 1 1 1 1 1 1 ...
```

- **datTH1**: Teildatensatz mit Itemantworten der Subgruppe von 1637 Schülerinnen und Schülern, die das erste Testheft bearbeitet haben.
  - `index`: Laufindex.
  - `idstud`: Schüleridentifikator.
  - `wgtstud`: Stichprobengewicht der Schülerin/des Schülers (vgl. Kapitel 2, *Stichprobenziehung*, im Band).
  - `th`: Bearbeitetes Testheft.
  - `I1...I50`: Itemantworten.

```
'data.frame': 1637 obs. of 29 variables:
$ index : int [1:1637] 8 10 12 23 24 34 41 46 54 57 ...
$ idstud : int [1:1637] 10010109 10010111 10020101 10020113 10020114 10030110 10040103
...
$ wgtstud: num [1:1637] 34.5 34.5 29.2 29.2 29.2 ...
$ th : chr [1:1637] "ER01" "ER01" "ER01" "ER01" ...
$ I1 : int [1:1637] 1 0 0 1 0 1 0 0 0 0 ...
$ I2 : int [1:1637] 0 0 0 1 0 0 1 0 0 1 ...
$ I6 : int [1:1637] 0 0 0 0 0 1 0 0 0 1 ...
[...]
$ I47 : int [1:1637] 0 2 0 2 0 0 2 1 0 1 ...
$ I50 : int [1:1637] 0 2 0 2 0 0 1 1 0 1 ...
```

- **itembankTH1**: Itembank zum Testheft 1.
  - `Item`: Itemname.
  - `format`: Antwortformat des Items.
  - `focus`: Fokuskategorie des Items.
  - `itemnr`: Itemidentifikator.
  - `N.subI`: Anzahl Subitems.

```
'data.frame': 25 obs. of 5 variables:
$ Item : chr [1:25] "I1" "I2" "I6" "I9" ...
$ format : chr [1:25] "MC4" "MC4" "MC4" "MC4" ...
$ focus : int [1:25] 0 0 0 0 0 1 1 1 1 1 ...
$ itemnr : int [1:25] 1661 2011 2421 2931 3131 3641 4491 4681 5621 5761 ...
$ N.subI : int [1:25] 1 1 1 1 1 1 1 1 1 1 ...
```

- **normdat**: Instrumentendaten einer Normierungsstudie (vgl. Kapitel 3, *Standard-Setting*, und Kapitel 5, *Testdesign*, im Band) mit Ankeritems für die Illustration von Linkingmethoden.
  - `idstud`: Schüleridentifikator.
  - `wgtstud`: Stichprobengewicht der Schülerin/des Schülers in der Normierungsstudie (es wird von einer vollständig randomisierten Stichprobe ausgegangen, weshalb die Gewichte konstant 1 sind).
  - `th`: Testheft.
  - `I*`: Itemantworten zu Items, die in der zu linkenden Studie auch eingesetzt werden.
  - `J*`: Itemantworten zu Items, die in der zu linkenden Studie nicht verwendet werden.

```
'data.frame': 3000 obs. of 327 variables:
$ idstud : int [1:3000] 1000 1005 1011 1014 1021 1024 1025 1026 1027 1028 ...
$ wgtstud: int [1:3000] 1 1 1 1 1 1 1 1 1 1 ...
$ th : chr [1:3000] "E8R01" "E8R02" "E8R03" "E8R04" ...
$ J1 : int [1:3000] NA NA NA NA NA NA NA NA 0 NA ...
$ J2 : int [1:3000] NA NA 0 NA NA NA NA NA NA NA ...
$ J3 : int [1:3000] NA NA NA NA NA NA NA NA 0 NA ...
[...]
$ I39 : int [1:3000] NA NA NA NA NA NA NA NA NA NA ...
$ I40 : int [1:3000] NA NA NA NA NA NA NA NA 0 NA ...
```

References

Trendtel, M., Pham, G. & Yanagida, T. (2016). Skalierung und Linking. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 185–224). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe `Kapitel 6`.

`datenKapitel07` *Illustrationsdaten zu Kapitel 7, Statistische Analysen produktiver Kompetenzen*

Description

Hier befindet sich die Dokumentation der in Kapitel 7, *Statistische Analysen produktiver Kompetenzen*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel07)
```

Format

`datenKapitel07` ist eine Liste mit den fünf Elementen `prodRat`, `prodPRat`, `prodRatL`, `prodRatEx` und `prodRatLEx`, zu unterschiedlichen Darstellungen von Ratings zu Schreib-Performanzen für das Kapitel wie auch die darin gestellten Übungsaufgaben.

- **prodRat**: Bewertung der Texte von 9736 Schülerinnen und Schülern zu einer von 3 "long prompts" durch einen (oder mehrere) der 41 Raters.
  - `idstud`: Schüleridentifikator.
  - `aufgabe`: 3 lange Schreibaufgaben.
  - `rater`: 41 Raters.
  - `TA`: Bewertung des Schülertexts auf der Dimension *Task Achievement* anhand einer 8-stufigen Ratingskala.
  - `CC`: Bewertung des Schülertexts auf der Dimension *Coherence and Cohesion* anhand einer 8-stufigen Ratingskala.
  - `GR`: Bewertung des Schülertexts auf der Dimension *Grammar* anhand einer 8-stufigen Ratingskala.
  - `VO`: Bewertung des Schülertexts auf der Dimension *Vocabulary* anhand einer 8-stufigen Ratingskala.

```
'data.frame': 10755 obs. of 7 variables:
$ idstud : int [1:10755] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ aufgabe: chr [1:10755] "E8W014" "E8W014" "E8W014" "E8W014" ...
$ rater : chr [1:10755] "R141" "R143" "R191" "R191" ...
$ TA : int [1:10755] 0 0 0 3 4 4 0 0 2 4 ...
$ CC : int [1:10755] 0 0 0 3 5 2 0 0 1 3 ...
$ GR : int [1:10755] 0 0 0 3 5 3 0 0 1 4 ...
$ VO : int [1:10755] 0 0 0 3 5 2 0 0 1 3 ...
```

- **prodPRat**: Bewertung der Schülertexte von 841 Schülerinnen und Schülern durch *Pseudo-raters*.
  Die Mehrfachkodierungen der Schülertexte werden auf zwei zufällige Raters reduziert (siehe Unterabschnitt 7.1 für eine Erläuterung).
  - `idstud`: Schüleridentifikator.
  - `aufgabe`: 3 lange Schreibaufgaben.
  - `TA_R1...VO_R1`: Bewertung des Schülertexts auf den Dimension *Task Achievement* (`TA_*`), *Coherence and Cohesion* (`CC_*`), *Grammar* (`GR_*`) und *Vocabulary* (`VO_*`) anhand einer 8-stufigen Ratingskala durch Pseudorater/in 1.
  - `TA_R2...VO_R2`: Bewertung des Schülertexts auf den Dimension *Task Achievement* (`TA_*`), *Coherence and Cohesion* (`CC_*`), *Grammar* (`GR_*`) und *Vocabulary* (`VO_*`) anhand einer 8-stufigen Ratingskala durch Pseudorater/in 2.

```
'data.frame': 841 obs. of 10 variables:
$ idstud : int [1:841] 10010108 10010112 10030106 10030110 10030112 10050105 ...
```

```
$ aufgabe: chr [1:841] "E8W006" "E8W006" "E8W010" "E8W006" ...
$ TA_R1 : int [1:841] 0 1 5 2 4 6 2 4 0 5 ...
$ CC_R1 : int [1:841] 0 1 5 2 6 5 2 6 0 3 ...
$ GR_R1 : int [1:841] 0 0 5 1 5 5 2 6 0 1 ...
$ VO_R1 : int [1:841] 0 2 4 1 5 5 3 6 0 2 ...
$ TA_R2 : int [1:841] 0 0 3 4 4 6 5 2 0 5 ...
$ CC_R2 : int [1:841] 0 0 2 2 4 5 2 3 0 2 ...
$ GR_R2 : int [1:841] 0 0 2 1 5 5 3 4 0 2 ...
$ VO_R2 : int [1:841] 0 0 3 2 5 6 4 3 0 2 ...
```

- **prodRatL**: Bewertung der Schülertexte im *Long Format*.
  - `idstud`: Schüleridentifikator.
  - `aufgabe`: 3 lange Schreibaufgaben.
  - `rater`: 41 Raters.
  - `item`: Dimension.
  - `response`: Rating zur Aufgabe in jeweiliger Dimension.

```
'data.frame': 43020 obs. of 5 variables:
$ idstud : int [1:43020] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ aufgabe : chr [1:43020] "E8W014" "E8W014" "E8W014" "E8W014" ...
$ rater : chr [1:43020] "R141" "R143" "R191" "R191" ...
$ item : Factor w/ 4 levels "TA","CC","GR",..: 1 1 1 1 1 1 1 1 1 1 ...
$ response: int [1:43020] 0 0 0 3 4 4 0 0 2 4 ...
```

- **prodRatEx**: Übungsdatensatz: Bewertung der Texte von 9748 Schülerinnen und Schülern zu einer von 3 "short prompts" durch einen (oder mehrere) der 41 Raters.
  - `idstud`: Schüleridentifikator.
  - `aufgabe`: 3 Schreibaufgaben.
  - `rater`: 41 Raters.
  - `TA`: Bewertung des Schülertexts auf der Dimension *Task Achievement* anhand einer 8-stufigen Ratingskala.
  - `CC`: Bewertung des Schülertexts auf der Dimension *Coherence and Cohesion* anhand einer 8-stufigen Ratingskala:
  - `GR`: Bewertung des Schülertexts auf der Dimension *Grammar* anhand einer 8-stufigen Ratingskala.
  - `VO`: Bewertung des Schülertexts auf der Dimension *Vocabulary* anhand einer 8-stufigen Ratingskala.

```
'data.frame': 10643 obs. of 7 variables:
$ idstud : int [1:10643] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ aufgabe: chr [1:10643] "E8W001" "E8W011" "E8W001" "E8W011" ...
$ rater : chr [1:10643] "R123" "R132" "R132" "R113" ...
$ TA : int [1:10643] 0 3 0 4 3 2 0 1 1 5 ...
$ CC : int [1:10643] 0 3 0 4 2 2 0 1 2 3 ...
$ GR : int [1:10643] 0 3 0 4 3 1 0 1 3 1 ...
$ VO : int [1:10643] 0 3 0 4 3 2 0 1 3 1 ...
```

- **prodRatLEx**: Übungsdatensatz: Bewertung der Schülertexte im *Long Format*.
  - `idstud`: Schüleridentifikator.
  - `aufgabe`: 3 kurze Schreibaufgaben.
  - `rater`: 41 Raters.
  - `item`: Dimension.
  - `response`: Rating zur Aufgabe in jeweiliger Dimension.

```
'data.frame': 42572 obs. of 5 variables:
$ idstud : int [1:42572] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ aufgabe : chr [1:42572] "E8W001" "E8W011" "E8W001" "E8W011" ...
$ rater : chr [1:42572] "R123" "R132" "R132" "R113" ...
$ item : Factor w/ 4 levels "TA","CC","GR",..: 1 1 1 1 1 1 1 1 1 1 ...
$ response: int [1:42572] 0 3 0 4 3 2 0 1 1 5 ...
```

References

Freunberger, R., Robitzsch, A. & Luger-Bazinger, C. (2016). Statistische Analysen produktiver Kompetenzen. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 225–258). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 7.

`datenKapitel08` *Illustrationsdaten zu Kapitel 8, Fehlende Daten und Plausible Values*

Description

Hier befindet sich die Dokumentation der in Kapitel 8, *Fehlende Daten und Plausible Values*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel08)
```

Format

`datenKapitel08` ist eine Liste mit den vier Elementen `data08H`, `data08I`, `data08J` und `data08K`, die Kontextinformationen mit fehlenden Daten zur Imputation sowie Instrumentendaten im Multiple-Matrix-Design für die Plausible-Value-Ziehung enthalten.

- **data08H**: Roh-Datensatz mit Leistungsschätzern und Kontextinformationen für 2507 Schüler/innen in 74 Schulen.

  - `idstud`: Schüleridentifikator.
  - `idschool`: Schulenidentifikator.
  - `wgtstud`: Stichprobengewicht der Schülerin/des Schülers (vgl. Kapitel 2, *Stichprobenziehung*, im Band).
  - `wgtstud`: Stichprobengewicht der Schule (vgl. Kapitel 2 im Band).
  - `Stratum`: Stratum der Schule. (`1:4` = Stratum 1 bis Stratum 4; für eine genauere Beschreibung der Strata, siehe Kapitel 2 im Band).
  - `female`: Geschlecht (1 = weiblich, `0` = männlich).
  - `migrant`: Migrationsstatus (1 = mit Migrationshintergrund, `0` = ohne Migrationshintergrund).
  - `HISEI`: Sozialstatus (vgl. Kapitel 10, *Reporting und Analysen*, im Band).
  - `eltausb`: Ausbildung der Eltern.
  - `buch`: Anzahl der Bücher zu Hause.
  - `SK`: Fragebogenskala "Selbstkonzept".
  - `LF`: Fragebogenskala "Lernfreude".
  - `NSchueler`: Anzahl Schüler/innen in der 4. Schulstufe (vgl. Kapitel 2 im Band).
  - `NKlassen`: Anzahl Klassen in der 4. Schulstufe (vgl. Kapitel 2 im Band).
  - `SES_Schule`: Auf Schulebene erfasster Sozialstatus (siehe Buchkapitel).
  - `E8WWLE`: WLE der Schreibkompetenz (vgl. Kapitel 7, *Statistische Analysen produktiver Kompetenzen*, im Band).
  - `E8LWLE`: WLE der Hörverstehenskompetenz (vgl. Kapitel 6, *Skalierung und Linking*, im Band).

```
'data.frame': 2507 obs. of 17 variables:
$ idstud : int [1:2507] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ idschool : int [1:2507] 1001 1001 1001 1001 1001 1001 1001 1001 1001 1001 ...
$ wgtstud : num [1:2507] 34.5 34.5 34.5 34.5 34.5 ...
$ wgtschool : num [1:2507] 31.2 31.2 31.2 31.2 31.2 ...
$ stratum : int [1:2507] 1 1 1 1 1 1 1 1 1 1 ...
$ female : int [1:2507] 0 0 0 0 1 0 1 1 1 1 ...
$ migrant : int [1:2507] 0 0 0 0 0 NA 0 0 0 0 ...
$ HISEI : int [1:2507] 31 NA 25 27 27 NA NA 57 52 58 ...
$ eltausb : int [1:2507] 2 NA 2 2 2 NA 2 1 2 1 ...
$ buch : int [1:2507] 1 1 1 1 3 NA 4 2 5 4 ...
$ SK : num [1:2507] 2.25 2.25 3 3 2.5 NA 2.5 3.25 3.5 2.5 ...
$ LF : num [1:2507] 1.25 1.5 1 1 4 NA 2 3.5 3.75 2.25 ...
$ NSchueler : int [1:2507] 69 69 69 69 69 69 69 69 69 69 ...
$ NKlassen : int [1:2507] 1 1 1 1 1 1 1 1 1 1 ...
$ SES_Schule: num [1:2507] 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57 ...
$ E8WWLE : num [1:2507] -3.311 -0.75 -3.311 0.769 1.006 ...
$ E8LWLE : num [1:2507] -1.175 -1.731 -1.311 0.284 0.336 ...
```

- **data08I**: Datensatz zur Illustration der Bedeutung einer geeigneten Behandlung fehlender Werte und von Messfehlern.
  - `index`: Laufindex.
  - `x`: Vollständig beobachteter Sozialstatus.

  - theta: Kompetenzwert.
  - WLE: WLE-Personenschätzer (vgl. Kapitel 6 im Band).
  - SEWLE: Messfehler ("standard error") des WLE-Personenschätzers.
  - X: Sozialstatus mit teilweise fehlenden Werten.

```
'data.frame': 1500 obs. of 6 variables:
$ index: int [1:1500] 1 2 3 4 5 6 7 8 9 10 ...
$ x : num [1:1500] 0.69 0.15 -0.13 -0.02 0.02 0.02 -0.56 0.14 -0.06 -1.41 ...
$ theta: num [1:1500] 2.08 -1.56 -0.65 -0.62 0.76 -1 1.12 0.08 0 -0.6 ...
$ WLE : num [1:1500] 1.22 -2.9 -2.02 0.03 0.8 0.93 0.28 -0.77 -0.31 -1.76 ...
$ SEWLE: num [1:1500] 0.83 0.82 0.8 0.8 0.8 0.81 0.81 0.8 0.8 0.8 ...
$ X : num [1:1500] 0.69 0.15 NA NA 0.02 0.02 -0.56 NA -0.06 -1.41 ...
```

- **data08J**: Datensatz data08H nach Imputation der fehlenden Werte. Für die Beschreibung der Variablen, siehe data08H.

```
'data.frame': 2507 obs. of 14 variables:
$ idstud : int [1:2507] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ wgtstud : num [1:2507] 34.5 34.5 34.5 34.5 34.5 ...
$ female : int [1:2507] 0 0 0 0 1 0 1 1 1 1 ...
$ migrant : num [1:2507] 0 0 0 0 0 ...
$ HISEI : num [1:2507] 31 56.8 25 27 27 ...
$ eltausb : num [1:2507] 2 1.04 2 2 2 ...
$ buch : num [1:2507] 1 1 1 1 3 ...
$ SK : num [1:2507] 2.25 2.25 3 3 2.5 ...
$ LF : num [1:2507] 1.25 1.5 1 1 4 ...
$ E8LWLE : num [1:2507] -1.175 -1.731 -1.311 0.284 0.336 ...
$ E8WWLE : num [1:2507] -3.311 -0.75 -3.311 0.769 1.006 ...
$ NSchueler : num [1:2507] 69 69 69 69 69 69 69 69 69 69 ...
$ NKlassen : int [1:2507] 1 1 1 1 1 1 1 1 1 1 ...
$ SES_Schule: num [1:2507] 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57 ...
```

- **data08K**: Datensatz mit Itemantworten der Schüler/innen zu den Testinstrumenten zu Hörverstehen und Schreiben.
  - idstud: Schüleridentifikator.
  - wgtstud: Stichprobengewicht der Schülerin/des Schülers (vgl. Kapitel 2 im Band).
  - E8LS*: Itemantworten für Hörverstehen (vgl. Kapitel 6).
  - E8W*: Itemantworten für Schreiben (vgl. Kapitel 7).

```
'data.frame': 2507 obs. of 99 variables:
$ idstud : int [1:2507] 10010101 10010102 10010103 10010105 10010106 10010107 ...
$ wgtstud : num [1:2507] 34.5 34.5 34.5 34.5 34.5 ...
$ E8LS0158 : int [1:2507] NA NA NA NA NA NA 0 0 NA NA ...
$ E8LS0165 : int [1:2507] 0 1 1 0 1 0 NA NA 1 0 ...
$ E8LS0166 : int [1:2507] 0 0 1 1 0 1 NA NA 1 1 ...
[...]
$ E8W014CC : int [1:2507] 0 0 0 3 5 2 NA NA NA NA ...
$ E8W014GR : int [1:2507] 0 0 0 3 5 3 NA NA NA NA ...
$ E8W014VOC: int [1:2507] 0 0 0 3 5 2 NA NA NA NA ...
```

References

Robitzsch, A., Pham, G. & Yanagida, T. (2016). Fehlende Daten und Plausible Values. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 259–293). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 8.

`datenKapitel09` *Illustrationsdaten zu Kapitel 9, Fairer Vergleich in der Rueckmeldung*

Description

Hier befindet sich die Dokumentation der in Kapitel 9, *Fairer Vergleich in der Rückmeldung*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel09)
```

Format

`datenKapitel09` ist ein singulärer vollständiger Datensatz.

- **datenKapitel09**: Datensatz mit sieben Kontextinformationen und 43 im Fairen Vergleich daraus abgeleiteten und berechneten Kenngrößen zu 244 Schulen (Kapitel 9).
  - `idschool`: Schulenidentifikator.
  - `Stratum`: Stratum der Schule. (`1:4` = Stratum 1 bis Stratum 4; für eine genauere Beschreibung der Strata, siehe Kapitel 2, *Stichprobenziehung*, im Band).
  - `groesse`: Logarithmierte Schulgröße.
  - `TWLE`: Aggregierte Leistungsschätzer der Schüler in der Schule (abhängige Variable im Fairen Verlgeich).
  - `female`: Anteil an Mädchen in der Schule.
  - `mig`: Anteil an Schülerinnen und Schülern mit Migrationshintergrund.
  - `sozstat`: Mittlerer sozioökonomischer Status (SES).
  - `zgroesse...zsozzsoz`: z-Standardisierte Werte der entsprechenden Variablen und Interaktionen.
  - `expTWLE.*`: Nach den jeweiligen Modellen erwartete (aggregierte) Leistungswerte der Schulen unter Berücksichtigung des Schulkontexts.
  - `*.eb*`: Untere und obere Grenzen der Erwartungsbereiche (EB) der Schulen und Indikator der Lage der Schule zum Bereich (`-1` = unter dem EB, `0` = im EB, `1` = über dem EB).

```
'data.frame': 244 obs. of 50 variables:
$ idschool : int 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 ...
$ stratum : int 1 1 1 1 1 1 1 1 1 1 ...
$ groesse : num 2.48 2.64 2.71 2.83 2.89 ...
$ TWLE : num 449 447 495 482 514 ...
$ female : num 0.545 0.462 0.571 0.529 0.389 ...
$ mig : num 0.0168 0.0769 0 0 0 ...
$ sozstat : num -1.034 -0.298 -0.413 -0.259 -0.197 ...
$ zgroesse : num -2.86 -2.54 -2.4 -2.14 -2.02 ...
[...]
$ expTWLE.OLS1 : num 431 475 481 489 485 ...
$ expTWLE.OLS2 : num 439 463 483 490 471 ...
$ expTWLE.Lasso1 : num 430 472 475 484 482 ...
$ expTWLE.Lasso2 : num 434 470 481 486 476 ...
[...]
$ expTWLE.np : num 422 478 479 490 465 ...
[...]
$ OLS1.eblow31 : num 415 460 465 474 470 ...
$ OLS1.ebupp31 : num 446 491 496 505 501 ...
$ OLS1.pos.eb31 : int 1 -1 0 0 1 -1 -1 -1 0 0 ...
[...]
```

References

Pham, G., Robitzsch, A., George, A. C. & Freunberger, R. (2016). Fairer Vergleich in der Rückmeldung. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 295–332). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 9.

`datenKapitel10` *Illustrationsdaten zu Kapitel 10, Reporting und Analysen*

Description

Hier befindet sich die Dokumentation der in Kapitel 10, *Reporting und Analysen*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung, verwendeten Daten. Die Komponenten der Datensätze werden knapp erläutert und deren Strukturen dargestellt.

Usage

```
data(datenKapitel10)
```

Format

datenKapitel10 ist eine Liste mit den vier Elementen, dat, dat.roh, dat.schule und dat.schule.roh. Die Elemente dat und dat.schule enthalten jeweils zehn imputierte Datensätze (vgl. Kapitel 8, *Fehlende Daten und Plausible Values*, im Band). Zum Vergleich stehen denen die Rohdatensätze dat.roh bzw. dat.schule.roh gegenüber.

- **dat** und **dat.roh**: Roh-Datensatz bzw. Liste mit zehn imputierten Datensätzen für 9885 Schülerinnen und Schüler.
  - idschool: Schulenidentifikator.
  - idstud: Schüleridentifikator.
  - idclass: Klassenidentifikator.
  - wgtstud: Stichprobengewicht der Schülerin/des Schülers (vgl. Kapitel 2, *Stichprobenziehung*, im Band).
  - female: Geschlecht (1 = weiblich, 0 = männlich).
  - migrant: Migrationsstatus (1 = mit Migrationshintergrund, 0 = ohne Migrationshintergrund).
  - HISEI: Sozialstatus (vgl. Kapitel 10, *Reporting und Analysen*, im Band).
  - eltausb: Ausbildung der Eltern.
  - buch: Anzahl der Bücher zu Hause.
  - SK: Fragebogenskala "Selbstkonzept".
  - LF: Fragebogenskala "Lernfreude".
  - E8RTWLE: WLE der Lesekompetenz (vgl. Kapitel 1, *Testkonstruktion*, und Kapitel 6, *Skalierung und Linking*, im Band).
  - E8RPV: Plausible Values für die Leistung in Englisch Lesen (vgl. Kapitel 8 im Band).
  - jkzone: Jackknife-Zone im Jackknife-Repeated-Replication-Design (vgl. Kapitel 2).
  - jkrep: Jackknife-Replikationsfaktor im Jackknife-Repeated-Replication-Design (vgl. Kapitel 2).
  - w_fstr*: Jackknife-Replikationsgewichte (vgl. Kapitel 2).

```
List of 10
$ :'data.frame': 9885 obs. of 151 variables:
..$ idschool : int [1:9885] 1001 1001 1001 1001 1001 1001 1001 ...
..$ idstud : int [1:9885] 10010101 10010102 10010103 10010105 ...
..$ idclass : int [1:9885] 100101 100101 100101 100101 100101 ...
..$ wgtstud : num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
..$ female : int [1:9885] 0 0 0 0 1 0 1 1 1 1 ...
..$ migrant : int [1:9885] 0 0 0 0 0 0 0 0 0 0 ...
..$ HISEI : int [1:9885] 31 28 25 27 27 76 23 57 52 58 ...
..$ eltausb : int [1:9885] 2 1 2 2 2 2 2 1 2 1 ...
..$ buch : int [1:9885] 1 1 1 1 3 3 4 2 5 4 ...
..$ SK : num [1:9885] 2.25 2.25 3 3 2.5 3.25 2.5 3.25 3.5 2.5 ...
..$ LF : num [1:9885] 1.25 1.5 1 1 4 3 2 3.5 3.75 2.25 ...
..$ E8RTWLE : num [1:9885] 350 438 383 613 526 ...
..$ E8RPV : num [1:9885] 390 473 380 599 509 ...
..$ jkzone : int [1:9885] 37 37 37 37 37 37 37 37 37 37 ...
..$ jkrep : int [1:9885] 0 0 0 0 0 0 0 0 0 0 ...
```

```
..$ w_fstr1 : num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
..$ w_fstr2 : num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
..$ w_fstr3 : num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
[...]
..$ w_fstr83 : num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
..$ w_fstr84 : num [1:9885] 34.5 34.5 34.5 34.5 34.5 ...
$ :'data.frame': 9885 obs. of 151 variables:
[...]
```

- **dat.schule** und **dat.schule.roh**: Roh-Datensatz bzw. Liste mit zehn imputierten Datensätzen als Liste für 244 Schulen. Es handelt sich hierbei – wie bei allen Datensätzen im Band – um fiktive (höchstens partiell-synthetische) Daten!
  - `idschool`: Schulenidentifikator.
  - `Schultyp`: Schultyp (`AHS` = allgemeinbildende höhere Schule, bzw. `APS` = allgemeinbildende Pflichtschule).
  - `Strata`: Stratum der Schule. (`1:4` = Stratum 1 bis Stratum 4, für eine genauere Beschreibung der Strata; siehe Kapitel 2 im Band).
  - `Strata.label`: Bezeichnung des Stratums.
  - `NSchueler`: Anzahl Schüler/innen in der 4. Schulstufe (vgl. Kapitel 2 im Band).
  - `NKlassen`: Anzahl Klassen in der 4. Schulstufe (vgl. Kapitel 2 im Band).
  - `gemgroesse`: Gemeindegröße.
  - `SCFRA04x02`: Fragebogenvariable aus Schulleiterfragebogen zur Schulgröße (vgl. https://www.bifie.at/node/2119).
  - `SCFO05a*`: Fragebogenvariable aus Schulleiterfragebogen zu "Schwerpunktschule für ..." (`*a01` = Informatik, `*a02` = Mathematik, `*a03` = Musik, `*a04` = Naturwissenschaften, `*a05` = Sport, `*a06` = Sprachen, `*a07` = Technik, `*a081` = Anderes; vgl. https://www.bifie.at/node/2119).
  - `HISEI`: Auf Schulenebene aggregierte HISEI.
  - `E8RPV`: Auf Schulenebene aggregierte Plausible Values für die Leistung in Englisch Lesen.

```
List of 10
$ :'data.frame': 244 obs. of 18 variables:
..$ idschool : int [1:244] 1001 1002 1003 1004 1005 1006 1007 1010 ...
..$ Schultyp : chr [1:244] "HS" "HS" "HS" "HS" ...
..$ Strata : int [1:244] 1 1 1 1 1 1 1 1 1 1 ...
..$ Strata.label: chr [1:244] "HS/Land" "HS/Land" "HS/Land" "HS/Land" ...
..$ NSchueler : int [1:244] 12 14 15 17 18 19 20 20 21 22 ...
..$ NKlassen : int [1:244] 1 1 1 1 2 1 2 1 2 2 ...
..$ gemgroesse : int [1:244] 5 4 4 5 3 4 5 4 4 5 ...
..$ SCFRA04x02 : int [1:244] 45 63 47 81 95 80 66 86 104 126 ...
..$ SCFO05a01 : int [1:244] 1 0 0 0 0 0 0 1 1 0 ...
..$ SCFO05a02 : int [1:244] 0 0 0 0 0 0 0 0 0 0 ...
..$ SCFO05a03 : int [1:244] 1 1 0 0 0 0 0 0 0 0 ...
..$ SCFO05a04 : int [1:244] 1 0 0 0 0 1 0 0 0 0 ...
..$ SCFO05a05 : int [1:244] 0 0 0 0 1 0 1 0 0 0 ...
..$ SCFO05a06 : int [1:244] 0 1 1 0 0 1 0 0 1 0 ...
..$ SCFO05a07 : int [1:244] 0 0 0 0 0 0 0 0 0 0 ...
```

```
..$ SCFO05a081 : int [1:244] 0 0 1 0 0 1 1 0 0 0 ...
..$ HISEI : num [1:244] 33.5 48.6 41.1 43.5 46.9 ...
..$ E8RPV : num [1:244] 471 463 513 494 525 ...
$ :'data.frame': 244 obs. of 18 variables:
[...]
```

References

Bruneforth, M., Oberwimmer, K. & Robitzsch, A. (2016). Reporting und Analysen. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit* R*: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 333–362). Wien: facultas.

See Also

Für die Verwendung der Daten, siehe Kapitel 10.

`Kapitel 0` *Kapitel 0: Konzeption der Ueberpruefung der Bildungsstandards in Oesterreich*

Description

Das ist die Nutzerseite zum Kapitel 0, *Konzeption der Überprüfung der Bildungsstandards in Österreich*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der öster\-reichischen Bildungsstandardüberprüfung. Hier werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert, dokumentiert und gegebenenfalls erweitert.

Details

Dieses Kapitel enthält keine Beispiele mit R.

Author(s)

Claudia Schreiner und Simone Breit

References

Schreiner, C. & Breit, S. (2016). Konzeption der Überprüfung der Bildungsstandards in Österreich. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit* R*: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 1–20). Wien: facultas.

See Also

Zu Kapitel 1, Testkonstruktion.
Zur Übersicht.

Kapitel 1 *Kapitel 1: Testkonstruktion*

Description

Das ist die Nutzerseite zum Kapitel 1, *Testkonstruktion*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Author(s)

Ursula Itzlinger-Bruneforth, Jörg-Tobias Kuhn, und Thomas Kiefer

References

Itzlinger-Bruneforth, U., Kuhn, J.-T. & Kiefer, T. (2016). Testkonstruktion. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 21–50). Wien: facultas.

See Also

Zu datenKapitel01, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 0, Konzeption.
Zu Kapitel 2, Stichprobenziehung.
Zur Übersicht.

Examples

```
## Not run:
library(TAM)
library(miceadds)
library(irr)
library(gtools)
library(car)

set.seed(1337)
data(datenKapitel01)
pilotScored <- datenKapitel01$pilotScored
pilotItems <- datenKapitel01$pilotItems
pilotRoh <- datenKapitel01$pilotRoh
pilotMM <- datenKapitel01$pilotMM

## ----------------------------------------------------------------
## Abschnitt 1.5.5: Aspekte empirischer Güteüberprüfung
## ----------------------------------------------------------------

# ----------------------------------------------------------------
```

```
# Abschnitt 1.5.5, Listing 1: Vorbereitung
#

# Rekodierter Datensatz pilotScored
dat <- pilotScored
items <- grep("E8R", colnames(dat), value = TRUE)
dat[items] <- recode(dat[items], "9=0;8=0")
# Itembank im Datensatz pilotItems
dat.ib <- pilotItems
items.dich <- dat.ib$item[dat.ib$maxScore == 1]

# Berechne erreichbare Punkte je TH
# aus Maximalscore je Item in Itembank
ind <- match(items, dat.ib$item)
testlets.ind <- ! items %in% items.dich
ind[testlets.ind] <- match(items[testlets.ind], dat.ib$testlet)
maxscores <- dat.ib$maxScore[ind]
max.form <- 1 * (!is.na(dat[, items])) %*% maxscores

# Erzielter Score ist der Summenscore dividiert durch
# Maximalscore
sumscore <- rowSums(dat[, items], na.rm = TRUE)
relscore <- sumscore/max.form
mean(relscore)

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 2: Omitted Response
#

library(TAM)
# Bestimme absolute und relative Häufigkeit der Kategorie 9 (OR)
ctt.omit <- tam.ctt2(pilotScored[, items])
ctt.omit <- ctt.omit[ctt.omit$Categ == 9, ]
# Übersicht der am häufigsten ausgelassenen Items
tail(ctt.omit[order(ctt.omit$RelFreq), -(1:4)])

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 3: Not Reached
#

not.reached <- rep(0, length(items))
names(not.reached) <- items

# Führe die Bestimmung in jedem Testheft durch
forms <- sort(unique(dat$form))
for(ff in forms){
  # (1) Extrahiere Itempositionen
  order.ff <- order(dat.ib[, ff], na.last = NA,
                    decreasing = TRUE)
  items.ff <- dat.ib$item[order.ff]
  testlets.ff <- dat.ib$testlet[order.ff]

  # (2) Sortiere Items und Testlets nach den Positionen
```

```
  testlets.ind <- ! items.ff %in% items.dich
  items.ff[testlets.ind] <- testlets.ff[testlets.ind]
  items.order.ff <- unique(items.ff)

  # (3) Bringe Testhefte in Reihenfolge und
  #     zähle von hinten aufeinanderfolgende Missings
  ind.ff <- pilotScored$form == ff
  dat.order.ff <- pilotScored[ind.ff, items.order.ff]
  dat.order.ff <- dat.order.ff == 9
  dat.order.ff <- apply(dat.order.ff, 1, cumsum)

  # (4) Vergleiche letzteres mit theoretisch möglichem
  #     vollständigen NR
  vergleich <- cumsum(rep(1, length(items.order.ff)))
  dat.order.ff[dat.order.ff != vergleich] <- 0

  # (5) Erstes NR kann auch OR sein
  erstes.NR <- apply(dat.order.ff, 2, which.max)
  ind <- cbind(erstes.NR, 1:ncol(dat.order.ff))
  dat.order.ff[ind] <- 0

  # (6) Zähle, wie oft für ein Item NR gilt
  not.reached.ff <- rowSums(dat.order.ff > 0)
  not.reached[items.order.ff] <- not.reached.ff[items.order.ff] +
    not.reached[items.order.ff]
}

tail(not.reached[order(not.reached)])

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 4: Itemschwierigkeit
#

# Statistik der relativen Lösungshäufigkeiten
p <- colMeans(dat[, items], na.rm = TRUE) / maxscores
summary(p)

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 5: Trennschärfe
#

discrim <- sapply(items, FUN = function(ii){
  if(var(dat[, ii], na.rm = TRUE) == 0) 0 else
    cor(dat[, ii], relscore, use = "pairwise.complete.obs")
})

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 6: Eindeutigkeit der Lösung
#

dat.roh <- pilotRoh
items <- grep("E8R", colnames(dat.roh), value = TRUE)
vars <- c("item", "Categ", "AbsFreq", "RelFreq", "rpb.WLE")
```

```
# Wähle nur geschlossene Items (d. h., nicht Open gap-fill)
items.ogf <- dat.ib$item[dat.ib$format == "Open gap-fill"]
items <- setdiff(items, items.ogf)

# Bestimme absolute und relative Häufigkeit der Antwortoptionen
# und jeweilige punktbiseriale Korrelationen mit dem Gesamtscore
ctt.roh <- tam.ctt2(dat.roh[, items], wlescore = relscore)

# Indikator der richtigen Antwort
match.item <- match(ctt.roh$item, dat.ib$item)
rohscore <- 1 * (ctt.roh$Categ == dat.ib$key[match.item])

# Klassifikation der Antwortoptionen
ist.antwort.option <- (!ctt.roh$Categ %in% c(8,9))
ist.distraktor <- rohscore == 0 & ist.antwort.option
ist.pos.korr <- ctt.roh$rpb.WLE > 0.05
ist.bearb <- ctt.roh$AbsFreq >= 10

# Ausgabe
ctt.roh[ist.distraktor & ist.pos.korr & ist.bearb, vars]

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 7: Plausible Distraktoren
#

# Ausgabe
head(ctt.roh[ist.distraktor & ctt.roh$RelFreq < 0.05, vars],4)

# ------------------------------------------------------------
# Abschnitt 1.5.5, Listing 8: Kodierbarkeit
#

library(irr)
dat.mm <- pilotMM

# Bestimme Modus der Berechnung: bei 3 Kodierern
# gibt es 3 paarweise Vergleiche
vars <- grep("Coder", colnames(dat.mm))
n.vergleiche <- choose(length(vars), 2)
ind.vergleiche <- upper.tri(diag(length(vars)))

# Berechne Statistik für jedes Item
coder <- NULL
for(ii in unique(dat.mm$item)){
  dat.mm.ii <- dat.mm[dat.mm$item == ii, vars]

  # Relative Häufigkeit der paarweisen Übereinstimmung
  agreed <- apply(dat.mm.ii, 1, function(dd){
    sum(outer(dd, dd, "==")[ind.vergleiche]) / n.vergleiche
  })

  # Fleiss Kappa
```

```
  kappa <- kappam.fleiss(dat.mm.ii)$value

  # Ausgabe
  coderII <- data.frame("item" = ii,
                        "p_agreed" = mean(agreed),
                        "kappa" = round(kappa, 4))
  coder <- rbind(coder, coderII)
}


## End(Not run)
```

Kapitel 2 *Kapitel 2: Stichprobenziehung*

Description

Das ist die Nutzerseite zum Kapitel 2, *Stichprobenziehung*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Details

Vorbereitungen: Zunächst werden die Datensätze `schule` mit den 1.327 Schulen der Population und `schueler` mit den 51.644 Schüler/innen dieser Schulen geladen. Durch das Setzen eines festen Startwerts für den Zufallszahlengenerator (`set.seed(20150506)`) wird erreicht, dass wiederholte Programmdurchläufe immer wieder zur selben Stichprobe führen.

Abschnitt 4.1: Stratifizierung - Schichtung einer Stichprobe: Die für die explizite Stratifizierung notwendige Information der Anzahl der Schüler/innen pro Stratum wird durch Aggregierung (Summe) aus dem Schuldatensatz in das Objekt `strata` extrahiert. Die entsprechende Spalte wird aus Gründen der Eindeutigkeit noch in `NSchuelerStratum` umbenannt.

```
> strata <- aggregate(schule[,"NSchueler", drop = FALSE],
+ by=schule[,"stratum", drop = FALSE], sum)
> colnames(strata)[2] <- "NSchuelerStratum" #Ergänzung zum Buch
```

Abschnitt 4.2: Schulenziehung, Listing 1: Im Schuldatensatz wird eine Dummyvariable `Klassenziehung` angelegt, die indiziert, in welchen Schulen mehr als drei Klassen sind, aus denen in Folge gezogen werden muss.

```
> schule$Klassenziehung <- 0
> schule[which(schule$NKlassen>3), "Klassenziehung"] <- 1
```

Abschnitt 4.2: Schulenziehung, Listing 2: Dann wird der unter Beachtung der Klassenziehung erwartete Beitrag der Schulen (d. h. die Anzahl ihrer Schülerinnen bzw. Schüler) zur Stichprobe in der Spalte `NSchueler.erw` errechnet.

```
> schule$NSchueler.erw <- schule$NSchueler
> ind <- which(schule$Klassenziehung == 1)
> schule[ind, "NSchueler.erw"] <-
+ schule[ind, "NSchueler"]/schule[ind, "NKlassen"]*3
```

Abschnitt 4.2: Schulenziehung, Listing 3: Berechnet man aus der erwarteten Anzahl von Lernenden pro Schule ihren relativen Anteil (Spalte `AnteilSchueler`) an der Gesamtschülerzahl im Stratum, so kann per Mittelwertbildung die mittlere Anzahl (Spalte `NSchueler/Schule.erw`) von Lernenden einer Schule pro Stratum bestimmt werden. Die mittlere Anzahl der Schulen im Stratum wird zusätzlich mit den einfachen Ziehungsgewichten der Schulen gewichtet, da große Schulen mit höherer Wahrscheinlichkeit für die Stichprobe gezogen werden.

```
> temp <- merge(schule[, c("SKZ","stratum","NSchueler")],
+ strata[, c("stratum","NSchuelerStratum")])
> schule$AnteilSchueler <-
+ temp$NSchueler/temp$NSchuelerStratum
> strata$"NSchueler/Schule.erw" <-
+ rowsum(apply(schule, 1, function(x)
+ x["NSchueler.erw"]*x["AnteilSchueler"]), schule$stratum)
```

Abschnitt 4.2: Schulenziehung, Listing 4: Schließlich erfolgt die Berechnung der Anzahl an Schulen (`Schulen.zu.ziehen`), die in jedem Stratum gezogen werden müssen, um einen Stichprobenumfang von 2500 Schülerinnen bzw. Schülern in etwa einzuhalten.

```
> strata$Schulen.zu.ziehen <-
+ round(2500/strata[,"NSchueler/Schule.erw"])
```

Abschnitt 4.2: Schulenziehung, Listing 5: Die Schulenliste wird vorab nach expliziten und impliziten Strata sortiert.

```
> schule <- schule[order(schule$stratum, schule$NSchueler),]
```

Abschnitt 4.2: Schulenziehung, Listing 6: Das Sampling-Intervall pro Stratum wird bestimmt (`Samp.Int`).

```
> strata$Samp.Int <-
+ strata$NSchuelerStratum/strata$Schulen.zu.ziehen
```

Abschnitt 4.2: Schulenziehung, Listing 7: Ein zufälliger Startwert aus dem Bereich 1 bis `Samp.Int` wird für jedes Stratum bestimmt (`Startwert`). Zur Festlegung eines festen Ausgangswertes des Zufallszahlengenerators siehe oben unter "Vorbereitungen".

```
> set.seed(20150506)
> strata$Startwert <-
+ sapply(ceiling(strata$Samp.Int), sample, size = 1)
```

Abschnitt 4.2: Schulenziehung, Listing 8: Die Listenpositionen der Lernenden, deren Schulen gezogen werden, werden vom Startwert ausgehend im Sampling-Intervall (pro Stratum) ermittelt. Die Positionen werden im Objekt `tickets` abgelegt.

```
> tickets <- sapply(1:4, function(x)
+ trunc(0:(strata[strata$stratum==x,"Schulen.zu.ziehen"]-1)
+ * strata[strata$stratum==x, "Samp.Int"] +
+ strata$Startwert[x]))
```

Abschnitt 4.2: Schulenziehung, Listing 9: Um die Auswahl der Schulen (entsprechend den Tickets der Lernenden) direkt auf der Schulliste durchführen zu können wird in `NSchuelerKum` die kumulierte Anzahl an Schülerinnen und Schülern nach Sortierung (siehe oben Abschnit 4.2, Listing 5) berechnet.

```
> schule$NSchuelerKum <-
+ unlist(sapply(1:4, function(x)
+ cumsum(schule[schule$stratum==x, "NSchueler"])))
```

Abschnitt 4.2: Schulenziehung, Listing 10: Durch die Dummy-Variable `SInSamp` werden nun jene Schulen als zugehörig zur Stichprobe markiert, von denen wenigstens eine Schülerin oder ein Schüler in Listing 8 dieses Abschnitts ein Ticket erhalten hat.

```
> schule$SInSamp <- 0
> for(s in 1:4) {
+ NSchuelerKumStrat <-
+ schule[schule$stratum==s, "NSchuelerKum"]
+ inds <- sapply(tickets[[s]], function(x)
+ setdiff(which(NSchuelerKumStrat <= x),
+ which(NSchuelerKumStrat[-1] <= x)))
+ schule[schule$stratum==s, "SInSamp"][inds] <- 1 }
```

Abschnitt 4.2: Schulenziehung, Listing 11: Die Ziehungswahrscheinlichkeiten der Schulen (`Z.Wsk.Schule`) werden für die später folgende Gewichtung berechnet.

```
> temp <- merge(schule[, c("stratum", "AnteilSchueler")],
+ strata[, c("stratum", "Schulen.zu.ziehen")])
> schule$Z.Wsk.Schule <-
+ temp$AnteilSchueler*temp$Schulen.zu.ziehen
```

Abschnitt 4.3: Klassenziehung, Listing 1: Im Objekt `schukla` werden zunächst notwendige Informationen für die Klassenziehung zusammengetragen. Die Dummy-Variable `KlInSamp` darin indiziert schließlich gezogene Klassen (aus bereits gezogenen Schulen), wobei aus Schulen mit drei oder weniger Klassen alle Klassen gezogen werden. Daher wird der Aufruf von `sample.int` mit `min(3, length(temp))` parametrisiert.

```
> schukla <- unique(merge(
+ schule[, c("SKZ","NKlassen", "Klassenziehung",
+ "Z.Wsk.Schule", "SInSamp")],
+ schueler[, c("SKZ", "idclass")], by="SKZ"))
> schukla$KlInSamp <- 0
> for(skz in unique(schukla[schukla$SInSamp==1,"SKZ"])) {
+ temp <- schukla[schukla$SKZ==skz, "idclass"]
+ schukla[schukla$idclass %in% temp[sample.int
+ (min(3, length(temp)))], "KlInSamp"] <- 1 }
```

Abschnitt 4.3: Klassenziehung, Listing 2: Die Ziehungswahrscheinlichkeit einer Klasse (`Z.Wsk.Klasse`) kann entsprechend der Dummy-Variable `Klassenziehung` (siehe Abschnitt 4.2, Listing 1) berechnet werden. Man beachte, dass entweder der erste oder der zweite Term der Addition Null ergeben muss, sodass die Fallunterscheidung direkt ausgedrückt werden kann.

```
> schukla$Z.Wsk.Klasse <- ((1 - schukla$Klassenziehung) * 1 +
+ schukla$Klassenziehung * 3 / schukla$NKlassen)
```

Abschnitt 4.4: Gewichtung, Listing 1: Nachdem das Objekt `schueler` um die Informationen zur Klassenziehung sowie den Ziehungswahrscheinlichkeiten von Schule und Klasse ergänzt wird, kann die Ziehungswahrscheinlichkeit einer Schülerin bzw. eines Schülers (`Z.Wsk.Schueler`) berechnet werden.

```
> schueler <- merge(schueler, schukla[, c("idclass", "KlInSamp", "Z.Wsk.Schule",
+ "Z.Wsk.Klasse")],
+ by="idclass", all.x=T)
> schueler$Z.Wsk.Schueler <-
+ schueler$Z.Wsk.Schule * schueler$Z.Wsk.Klasse
```

Abschnitt 4.4: Gewichtung, Listing 2: Nach Reduktion des Objekts `schueler` auf die gezogenen Lernenden, werden in `temp` die nonresponse-Raten (Variable `x`) bestimmt.

```
> schueler <- schueler[schueler$KlInSamp==1,]
> temp <- merge(schueler[, c("idclass",
+ "Z.Wsk.Schueler")],
+ aggregate(schueler$teilnahme,
+ by=list(schueler$idclass),
+ function(x) sum(x)/length(x)),
+ by.x="idclass", by.y="Group.1")
```

Abschnitt 4.4: Gewichtung, Listing 3: Mittels der Ziehungswahrscheinlichkeiten der Schülerinnen und Schüler sowie der nonresponse-Raten (siehe vorangegangenes Listing) werden die (nicht normierten) Schülergewichte (`studwgt`) bestimmt.

```
> schueler$studwgt <- 1/temp$x/temp$Z.Wsk.Schueler
```

Abschnitt 4.4: Gewichtung, Listing 4: Schließlich werden die Schülergewichte in Bezug auf die Anzahl an Schülerinnen und Schülern im jeweiligen Stratum normiert (`NormStudwgt`), sodass sie in Summe dieser Anzahl entsprechen.

```
> Normierung <- strata$NSchuelerStratum /
+ rowsum(schueler$studwgt * schueler$teilnahme,
+ group = schueler$Stratum)
> schueler$NormStudwgt <-
+ schueler$studwgt * Normierung[schueler$Stratum]
```

Abschnitt 5.3: Anwendung per Jackknife-Repeated-Replication, Listing 1:

Die im Folgenden genutzte Hilfsfunktion `zones.within.stratum` erzeugt ab einem Offset einen Vektor mit jeweils doppelt vorkommenden IDs zur Bildung der Jackknife-Zonen. Nachdem die Schulliste zunächst auf die gezogenen Schulen und nach expliziten und impliziten Strata* sortiert wurde, werden die Strata in Pseudo-Strata mit zwei (oder bei ungerader Anzahl drei) Schulen unterteilt. Dies führt zur Variable `jkzone`. Basierend auf `jkzone` wird für jeweils eine der Schulen im Pseudo-Stratum der Indikator `jkrep` auf Null gesetzt, um diese in der jeweiligen Replikation von der Berechnung auszuschließen. Ergänzend zum Buch wird hier eine Fallunterscheidung getroffen, ob in einem Pseudo-Stratum zwei oder drei Schulen sind (s.o): Bei drei Schulen wird zufällig ausgewählt, ob bei ein oder zwei Schulen `jkrep=0` gesetzt wird.

* Die Sortierung nach dem impliziten Strata Schulgröße erfolgt hier absteigend, nachzulesen im Buch-Kapitel.

```
> ### Ergänzung zum Buch: Hilfsfunktion zones.within.stratum
> zones.within.stratum <- function(offset,n.str) {
+ maxzone <- offset-1+floor(n.str/2)
+ zones <- sort(rep(offset:maxzone,2))
+ if (n.str %% 2 == 1) zones <- c(zones,maxzone)
+ return(zones) }
> ### Ende der Ergänzung
>
> # Sortieren der Schulliste (explizite und implizite Strata)
> schule <- schule[schule$SInSamp==1,]
> schule <- schule[order(schule$stratum,-schule$NSchueler),]
>
> # Unterteilung in Pseudostrata
> cnt.strata <- length(unique(schule$stratum))
> offset <- 1
> jkzones.vect <- integer()
> for (i in 1:cnt.strata) {
```

```
+ n.str <- table(schule$stratum)[i]
+ jkzones.vect <-
+ c(jkzones.vect,zones.within.stratum(offset,n.str))
+ offset <- max(jkzones.vect)+1 }
> schule$jkzone <- jkzones.vect
>
> # Zufällige Auswahl von Schulen mit Gewicht 0
> schule$jkrep <- 1
> cnt.zones <- max(schule$jkzone)
> jkrep.rows.null <- integer()
> for (i in 1:cnt.zones) {
+ rows.zone <- which(schule$jkzone==i)
+ ### Ergänzung zum Buch: Fallunterscheidung je nach Anzahl Schulen in der Zone
+ if (length(rows.zone)==2) jkrep.rows.null <-
+ c(jkrep.rows.null,sample(rows.zone,size=1))
+ else {
+ num.null <- sample(1:2,size=1)
+ jkrep.rows.null <-
+ c(jkrep.rows.null,sample(rows.zone,size=num.null))
+ } }
> schule[jkrep.rows.null,]$jkrep <- 0
```

Abschnitt 5.3: Anwendung per Jackknife-Repeated-Replication, Listing 2: Die Anwendung von Jackknife-Repeated-Replication zur Abschätzung der Stichprobenvarianz wird im folgenden am Schülerdatensatz demonstriert, weswegen `jkzone` und `jkrep` zunächst auf diese Aggregatsebene übertragen werden. In einer Schleife werden replicate weights mittels `jkzone` und `jkrep` generiert. Diese beziehen sich auf das normierte Schülergewicht `NormStudwgt`. Man beachte: Es gilt entweder `in.zone==0` oder `(in.zone-1)==0`, sodass Formel 5 aus dem Buch-Kapitel direkt in einer Addition ausgedrückt werden kann. Es entstehen so viele replicate weights (`w_fstr1` usw.) wie Jackknife-Zonen existieren.

```
> # Übertragung auf Schülerebene
> schueler <-
+ merge(schueler,schule[,c("SKZ","jkzone","jkrep")],all.x=TRUE)
> # Schleife zur Generierung von Replicate Weights
> for (i in 1:cnt.zones) {
+ in.zone <- as.numeric(schueler$jkzone==i)
+ schueler[paste0("w_fstr",i)] <- # vgl. Formel 5
+ in.zone * schueler$jkrep * schueler$NormStudwgt * 2 +
+ (1-in.zone) * schueler$NormStudwgt }
```

Abschnitt 5.3: Anwendung per Jackknife-Repeated-Replication, Listing 3: Als einfaches Beispiel wird der Anteil Mädchen (`perc.female`) in der Population aus der Stichprobe heraus geschätzt. Die Schätzung selbst erfolgt als Punktschätzung über das normierte Schülergewicht. Zur Bestimmung der Stichprobenvarianz `var.jrr` wird der Anteil wiederholt mit allen replicate weights berechnet und die quadrierte Differenz zur Punktschätzung einfach aufsummiert (Formel 6 aus dem Buch-Kapitel).

```
> # Schätzung mittels Gesamtgewicht
> n.female <- sum(schueler[schueler$female==1,]$NormStudwgt)
> perc.female <- n.female / sum(schueler$NormStudwgt)
> # wiederholte Berechnung und Varianz
> var.jrr = 0
> for (i in 1:cnt.zones) {
+ n.female.rep <-
+ sum(schueler[schueler$female==1,paste0("w_fstr",i)])
+ perc.female.rep <-
+ n.female.rep / sum(schueler[paste0("w_fstr",i)])
+ var.jrr <- # vgl. Formel 6
+ var.jrr + (perc.female.rep - perc.female) ^ 2.0 }
```

Author(s)

Ann Cathrice George, Konrad Oberwimmer, Ursula Itzlinger-Bruneforth

References

George, A. C., Oberwimmer, K. & Itzlinger-Bruneforth, U. (2016). Stichprobenziehung. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 51–81). Wien: facultas.

See Also

Zu datenKapitel02, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 1, Testkonstruktion.
Zu Kapitel 3, Standard-Setting.
Zur Übersicht.

Examples

```
## Not run:
data(datenKapitel02)
schueler <- datenKapitel02$schueler
schule <- datenKapitel02$schule
set.seed(20150506)

## ------------------------------------------------------------
## Abschnitt 4.1: Stratifizierung
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 4.1, Listing 1

# Information in Strata
strata <- aggregate(schule[,"NSchueler", drop = FALSE],
                    by=schule[,"stratum", drop = FALSE], sum)
colnames(strata)[2] <- "NSchuelerStratum"
```

```
## ------------------------------------------------------------
## Abschnitt 4.2: Schulenziehung
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 1

# Dummyvariable Klassenziehung
schule$Klassenziehung <- 0
schule[which(schule$NKlassen>3), "Klassenziehung"] <- 1

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 2

# erwarteter Beitrag zur Stichprobe pro Schule
schule$NSchueler.erw <- schule$NSchueler
ind <- which(schule$Klassenziehung == 1)
schule[ind, "NSchueler.erw"] <-
  schule[ind, "NSchueler"]/schule[ind, "NKlassen"]*3

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 3

# relativer Anteil Schüler pro Schule
temp <- merge(schule[, c("SKZ","stratum","NSchueler")],
              strata[, c("stratum","NSchuelerStratum")])
schule$AnteilSchueler <-
  temp$NSchueler/temp$NSchuelerStratum
# mittlere Anzahl von Schülern pro Schule
strata$"NSchueler/Schule.erw" <-
  rowsum(apply(schule, 1, function(x)
    x["NSchueler.erw"]*x["AnteilSchueler"]), schule$stratum)

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 4

# Bestimmung Anzahl zu ziehender Schulen
strata$Schulen.zu.ziehen <-
  round(2500/strata[,"NSchueler/Schule.erw"])

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 5

# Schulenliste nach Stratum und Groesse ordnen
schule <-
  schule[order(schule$stratum, schule$NSchueler),]

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 6

# Berechnung Sampling-Intervall
strata$Samp.Int <-
```

```
  strata$NSchuelerStratum/strata$Schulen.zu.ziehen

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 7

# Startwerte bestimmen
strata$Startwert <-
  sapply(ceiling(strata$Samp.Int), sample, size = 1)

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 8

# Schüler-Tickets
tickets <- sapply(1:4, function(x)
  trunc(0:(strata[strata$stratum==x,"Schulen.zu.ziehen"]-1)
  * strata[strata$stratum==x, "Samp.Int"] +
    strata$Startwert[x]))

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 9

# kummulierte Schüleranzahl pro Stratum berechnen
schule$NSchuelerKum <-
  unlist(sapply(1:4, function(x)
    cumsum(schule[schule$stratum==x, "NSchueler"])))

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 10

# Schulen ziehen
schule$SInSamp <- 0
for(s in 1:4) {
  NSchuelerKumStrat <-
    schule[schule$stratum==s, "NSchuelerKum"]
  inds <- sapply(tickets[[s]], function(x)
    setdiff(which(NSchuelerKumStrat <= x),
            which(NSchuelerKumStrat[-1] <= x)))
  schule[schule$stratum==s, "SInSamp"][inds] <- 1 }

# ------------------------------------------------------------
# Abschnitt 4.2, Listing 11

# Berechnung Ziehungswahrscheinlichkeit Schule
temp <- merge(schule[, c("stratum", "AnteilSchueler")],
  strata[, c("stratum", "Schulen.zu.ziehen")])
schule$Z.Wsk.Schule <-
  temp$AnteilSchueler*temp$Schulen.zu.ziehen

## ------------------------------------------------------------
## Abschnitt 4.3: Klassenziehung
## ------------------------------------------------------------

# ------------------------------------------------------------
```

```
# Abschnitt 4.3, Listing 1

### Klassenziehung (Alternative 2)
schukla <- unique(merge(
  schule[, c("SKZ","NKlassen", "Klassenziehung",
    "Z.Wsk.Schule", "SInSamp")],
    schueler[, c("SKZ", "idclass")], by="SKZ"))
schukla$KlInSamp <- 0
for(skz in unique(schukla[schukla$SInSamp==1,"SKZ"])) {
  temp <- schukla[schukla$SKZ==skz, "idclass"]
  schukla[schukla$idclass%in%temp[sample.int(
    min(3, length(temp)))], "KlInSamp"] <- 1 }

# ------------------------------------------------------------
# Abschnitt 4.3, Listing 2

# Ziehungswahrscheinlichkeit Klasse
schukla$Z.Wsk.Klasse <- ((1 - schukla$Klassenziehung) * 1 +
     schukla$Klassenziehung * 3 / schukla$NKlassen)

## ------------------------------------------------------------
## Abschnitt 4.4: Gewichtung
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 4.4, Listing 1

### Gewichte
schueler <- merge(schueler, schukla[, c("idclass", "KlInSamp", "Z.Wsk.Schule",
                                        "Z.Wsk.Klasse")],
                  by="idclass", all.x=T)
# Ziehungswahrscheinlichkeiten Schueler
schueler$Z.Wsk.Schueler <-
  schueler$Z.Wsk.Schule * schueler$Z.Wsk.Klasse

# ------------------------------------------------------------
# Abschnitt 4.4, Listing 2

schueler <- schueler[schueler$KlInSamp==1,]
# Nonresponse Adjustment
temp <- merge(schueler[, c("idclass", "Z.Wsk.Schueler")],
  aggregate(schueler$teilnahme,
    by=list(schueler$idclass),
    function(x) sum(x)/length(x)),
  by.x="idclass", by.y="Group.1")

# ------------------------------------------------------------
# Abschnitt 4.4, Listing 3

# Schülergewichte
schueler$studwgt <- 1/temp$x/temp$Z.Wsk.Schueler

# ------------------------------------------------------------
```

```
# Abschnitt 4.4, Listing 4

# Normierung
Normierung <- strata$NSchuelerStratum /
  rowsum(schueler$studwgt * schueler$teilnahme,
         group = schueler$Stratum)
schueler$NormStudwgt <-
  schueler$studwgt * Normierung[schueler$Stratum]

## ------------------------------------------------------------
## Abschnitt 5.3: Jackknife-Repeated-Replication
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 5.3, Listing 1

### Ergänzung zum Buch: Hilfsfunktion zones.within.stratum
zones.within.stratum <- function(offset,n.str) {
  maxzone <- offset-1+floor(n.str/2)
  zones <- sort(rep(offset:maxzone,2))
  if (n.str %% 2 == 1) zones <- c(zones,maxzone)
  return(zones) }
### Ende der Ergänzung

# Sortieren der Schulliste (explizite und implizite Strata)
schule <- schule[schule$SInSamp==1,]
schule <- schule[order(schule$stratum,-schule$NSchueler),]

# Unterteilung in Pseudostrata
cnt.strata <- length(unique(schule$stratum))
offset <- 1
jkzones.vect <- integer()
for (i in 1:cnt.strata) {
  n.str <- table(schule$stratum)[i]
  jkzones.vect <-
    c(jkzones.vect,zones.within.stratum(offset,n.str))
  offset <- max(jkzones.vect)+1 }
schule$jkzone <- jkzones.vect

# Zufällige Auswahl von Schulen mit Gewicht 0
schule$jkrep <- 1
cnt.zones <- max(schule$jkzone)
jkrep.rows.null <- integer()
for (i in 1:cnt.zones) {
  rows.zone <- which(schule$jkzone==i)
### Ergänzung zum Buch: Fallunterscheidung je nach Anzahl Schulen in der Zone
  if (length(rows.zone)==2) jkrep.rows.null <-
    c(jkrep.rows.null,sample(rows.zone,size=1))
  else {
      num.null <- sample(1:2,size=1)
      jkrep.rows.null <-
        c(jkrep.rows.null,sample(rows.zone,size=num.null))
    } }
```

```
schule[jkrep.rows.null,]$jkrep <- 0

# ------------------------------------------------------------
# Abschnitt 5.3, Listing 2

# Übertragung auf Schülerebene
schueler <-
  merge(schueler,schule[,c("SKZ","jkzone","jkrep")],all.x=TRUE)
# Schleife zur Generierung von Replicate Weights
for (i in 1:cnt.zones) {
  in.zone <- as.numeric(schueler$jkzone==i)
  schueler[paste0("w_fstr",i)] <-   # vgl. Formel 5
    in.zone * schueler$jkrep * schueler$NormStudwgt * 2 +
    (1-in.zone) * schueler$NormStudwgt }

# ------------------------------------------------------------
# Abschnitt 5.3, Listing 3

# Schätzung mittels Gesamtgewicht
n.female <- sum(schueler[schueler$female==1,]$NormStudwgt)
perc.female <- n.female / sum(schueler$NormStudwgt)
# wiederholte Berechnung und Varianz
var.jrr = 0
for (i in 1:cnt.zones) {
  n.female.rep <-
    sum(schueler[schueler$female==1,paste0("w_fstr",i)])
  perc.female.rep <-
    n.female.rep / sum(schueler[paste0("w_fstr",i)])
  var.jrr <-   # vgl. Formel 6
    var.jrr + (perc.female.rep - perc.female) ^ 2.0 }

## End(Not run)
```

`Kapitel 3` *Kapitel 3: Standard-Setting*

Description

Das ist die Nutzerseite zum Kapitel 3, *Standard-Setting*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Details

Übersicht über die verwendeten Daten: Für dieses Kapitel werden drei Datensätze verwendet. Der Datensatz `ratings` ist das Ergebnis der IDM-Methode, darin enthalten sind für alle Items die Einstufung jedes Raters auf eine der drei Kompetenzstufen (1, 2, 3), sowie Item-Nummer und

Schwierigkeit. Der Datensatz `bookmarks` ist das Ergebnis der Bookmark-Methode, darin enthalten sind pro Rater und pro Cut-Score jeweils die gewählte Bookmark als Seitenzahl im OIB (die ein bestimmtes Item repräsentiert). In `sdat` sind Personenparameter von 3500 Schülerinnen und Schülern enthalten, diese dienen zur Schätzung von Impact Data. Der Datensatz `productive` ist für die Illustration der Contrasting-Groups-Methode gedacht: Dieser enthält die Ratings aus der Contrasting-Groups-Methode, pro Rater die Information, ob der entsprechende Text auf die Stufe unter- oder oberhalb des Cut-Scores eingeteilt wurde, sowie Nummer des Textes und Personenfähigkeit.

Abschnitt 3.2.2: Daten aus der IDM-Methode:

Listing 1: Feedback: Hier wird der Datensatz `ratings` verwendet. Er ist das Ergebnis der IDM-Methode, darin enthalten sind für alle Items die Einstufung jedes Raters auf eine der drei Kompetenzstufen (1, 2, 3). Zunächst werden die Rater und die Items aus dem Datensatz ausgewählt, dann wird pro Item die prozentuelle Verteilung der Ratings auf die drei Stufen berechnet.

```
> raterID <- grep("R", colnames(ratings), value = TRUE)
> nraters <- length(raterID)
> nitems <- nrow(ratings)
> itemID <- ratings[, 1]
> itemdiff <- ratings[, 2]
> stufen <- c(1, 2, 3) # Anzahl der Kompetenzstufen
> item.freq <- data.frame()
> # Berechne Prozentuelle Zuteilungen auf Stufen pro Item
> tabelle.ii <- data.frame()
> for(ii in 1:nitems){
+ tabelle.ii <- round(table(factor(as.numeric(ratings[ii,
+ raterID]), levels = stufen)) / nraters * 100, digits = 2)
+ item.freq <- rbind(item.freq, tabelle.ii) }
> colnames(item.freq) <- paste0("Level_", stufen)
> item.freq <- data.frame(ratings[, 1:2], item.freq)
> head(item.freq, 3)
> # Anmerkung: Item 3 zu 100% auf Stufe 1, Item 2 aufgeteilt
> # auf Stufe 1 und 2
```

Listing 1a: Ergänzung zum Buch: Hier wird eine Grafik erzeugt, in der das Rating-Verhalten sichtbar wird: Pro Item wird angezeigt, wieviele Prozent der Raters es auf eine der drei Stufen eingeteilt haben. Zunächst werden drei verschiedene Farben definiert, anschließend werden drei Barplots erstellt, die zusammen auf einer Seite dargestellt werden. Die Grafik wird zur Orientierung bei Diskussionen verwendet, da so schnell ersichtlich ist, bei welchen Items sich das Experten-Panel einig oder uneinig war. Für die Grafik gibt es die Möglichkeit, diese in Schwarz-Weiss zu halten oder in Farbe zu gestalten.

```
> # Farben für die Grafik definieren - falls eine bunte Grafik gewünscht ist,
> # kann barcol <- c(c1, c2, c3) definiert werden
> c1 <- rgb(239/255, 214/255, 67/255)
> c2 <- rgb(207/255, 151/255, 49/255)
> c3 <- rgb(207/255, 109/255, 49/255)
```

```
>
> # Aufbereitung Tabelle für Grafik
> freq.dat <- t(as.matrix(item.freq[1:nitems,(3:(2+length(stufen)))]))
> barcol <- c("black", "gray", "white")
>
> #Grafik wird erzeugt
> par(mfcol=c(3,1), oma=c(0,0,3,0)) # Angeben der Plot-Anzahl
> perplot <- round(nitems/3)
> a <- perplot + 1
> b <- perplot*2
> c <- b + 1
> d <- perplot*3
> barplot(freq.dat[,1 : perplot], col = barcol, beside = T,
+ names.arg = seq(1 , perplot), xlab = "Itemnummer (Seitenzahl im OIB)",
+ ylab = "% Zuteilung auf Stufe", horiz = F, ylim = range(1:100))
> barplot(freq.dat[, a:b], col = barcol, beside = T, names.arg = seq(a, b),
+ xlab = "Itemnummer (Seitenzahl im OIB)",
+ ylab = "% Zuteilung auf Stufe",
+ horiz = F, ylim = range(1:100))
> barplot(freq.dat[, c:d], col = barcol, beside = T, names.arg = seq(c, d),
+ xlab = "Itemnummer (Seitenzahl im OIB)",
+ ylab = "% Zuteilung auf Stufe",
+ horiz = F, ylim = range(1:100))
> title("Feedback für das Experten-Panel aus der IDM-Methode", outer = T)
```

Listing 2: Cut-Score Berechnung: Hier wird der Cut-Score aus den Daten der IDM-Methode mithilfe logistischer Regression für den ersten Rater im Experten-Panel berechnet. Dafür wird der zweite Cut-Score herangezogen. Zunächst müssen die entsprechenden Ratings für die logistische Regression umkodiert werden (2 = 0, 3 = 1). Anschließend wird die logistische Regression berechnet, als unabhängige Variable dient die Einstufung durch den jeweiligen Experten (0, 1), als abhängige Variable die Itemschwierigkeit. Anhand der erhaltenen Koeffizienten kann der Cut-Score berechnet werden.

```
> library(car)
> # Rekodieren
> rate.i <- ratings[which(ratings$R01 %in% c(2, 3)),
+ c("MB_Norm_rp23", "R01")]
> rate.i$R01 <- recode(rate.i$R01, "2=0; 3=1")
> coef(cut.i <- glm(rate.i$R01 ~ rate.i$MB_Norm_rp23 ,
+ family = binomial(link="logit")))
> # Berechnung des Cut-Scores laut Formel
> cut.R01 <- (-cut.i$coefficients[1])/ cut.i$coefficients[2]
```

Listing 3: Rater-Analysen: Als ersten Schritt in den Rater-Analysen wird das mittlere Cohen's Kappa eines Raters mit allen anderen Raters berechnet. Dafür werden zunächst die Ratings ausgewählt und dann für jeden Rater die Übereinstimmung mit jedem anderen Rater paarweise berechnet. Anschließend werden diese Werte gemittelt und auch die Standard-Abweichung berechnet.

```
> library(irr)
> # Auswahl der Ratings
> rater.dat <- ratings[ ,grep("R", colnames(ratings))]
> # Kappa von jeder Person mit allen anderen Personen wird berechnet
> kappa.mat <- matrix(NA, nraters, nraters)
> for(ii in 1:nraters){
+ rater.eins <- rater.dat[, ii]
+ for(kk in 1:nraters){
+ rater.zwei <- rater.dat[ ,kk]
+ dfr.ii <- cbind(rater.eins, rater.zwei)
+ kappa.ik <- kappa2(dfr.ii)
+ kappa.mat[ii, kk] <- kappa.ik$value
+ }}
> diag(kappa.mat) <- NA
> # Berechne Mittleres Kappa für jede Person
> MW_Kappa <- round(colMeans(kappa.mat, na.rm=T), digits=2)
> SD_Kappa <- round(apply(kappa.mat, 2, sd, na.rm=T), digits=2)
> (Kappa.Stat <- data.frame("Person"= raterID, MW_Kappa, SD_Kappa))
```

Listing 4: Berechnung Fleiss' Kappa: Fleiss' Kappa gibt die Übereinstimmung innerhalb des gesamten Experten-Panels an. Wird das Standard-Setting über mehrere Runden durchgeführt, kann Fleiss' Kappa auch für jede Runde berechnet werden.

```
> kappam.fleiss(rater.dat)
```

Listing 5: Modalwerte: Auch die Korrelation zwischen dem Modalwert jedes Items (d.h., ob es am häufigsten auf Stufe 1, 2 oder 3 eingeteilt wurde) und der inviduellen Zuordnung durch einen Rater kann zu Rater-Analysen herangezogen werden. Zunächst wird der Modal-Wert eines jeden Items berechnet. Hat ein Item zwei gleich häufige Werte, gibt es eine Warnmeldung und es wird für dieses Item NA anstatt eines Wertes vergeben (für diese Analyse sind aber nur Items von Interesse, die einen eindeutigen Modalwert haben). Danach wird pro Rater die Korrelation zwischen dem Modalwert eines Items und der entsprechenden Einteilung durch den Rater berechnet, und dann in aufsteigender Höhe ausgegeben.

```
> library(prettyR)
> # Berechne Modalwert
> mode <- as.numeric(apply(rater.dat, 1, Mode))
> # Korrelation für die Ratings jeder Person im Panel mit den
> # Modalwerten der Items
> corr <- data.frame()
> for(z in raterID){
+ rater.ii <- rater.dat[, (grep(z, colnames(rater.dat)))]
+ cor.ii <- round(cor(mode, rater.ii, use = "pairwise.complete.obs",
+ method = "spearman"), digits = 2)
+ corr <- rbind(corr, cor.ii)
+ }
> corr[, 2] <- raterID
```

```
> colnames(corr) <- c("Korrelation", "Rater")
> # Aufsteigende Reihenfolge
> (corr <- corr[order(corr[, 1]),])
```

Listing 5a: Ergänzung zum Buch: Die Korrelation zwischen Modalwerten und individueller Zuordnung kann auch zur besseren Übersicht graphisch gezeigt werden. Dabei werden die Korrelationen der Raters aufsteigend dargestellt.

```
> # Grafik
> plot(corr$Korrelation, xlab = NA, ylab = "Korrelation",
+ ylim = c(0.5, 1), xaxt = "n", main = \"Korrelation zwischen
+ Modalwert und individueller Zuordnung der Items pro Rater/in\")
> text(seq(1:nraters), corr$Korrelation - 0.02, labels = corr[, 2],
+ offset = 1, cex = 1)
> title(xlab = "Raters nach aufsteigender Korrelation gereiht")
```

Listing 6: ICC: Hier wird der ICC als Ausdruck der Übereinstimmung (d.h., Items werden auf dieselbe Stufe eingeteilt) und der Konsistenz (d.h., Items werden in dieselbe Reihenfolge gebracht) zwischen Raters berechnet. Falls es mehrere Runden gibt, kann der ICC auch wiederholt berechnet und verglichen werden.

```
> library(irr)
> (iccdat.agree <- icc(rater.dat, model = "twoway", type = "agreement",
+ unit = "single", r0 = 0, conf.level=0.95))
> (iccdat.cons <- icc(rater.dat, model = "twoway", type = "consistency",
+ unit = "single", r0 = 0, conf.level=0.95))
```

Abschnitt 3.2.3: Daten aus der Bookmark-Methode:

Listing 1: Feedback: Auch in der Bookmark-Methode wird dem Experten-Panel Feedback angeboten, um die Diskussion zu fördern. Hier wird pro Cut-Score Median, Mittelwert und Standard-Abweichung der Bookmarks (Seitenzahl im OIB) im Experten-Panel berechnet.

```
> head(bookmarks)
> statbookm <- data.frame("Stats"=c("Md","Mean","SD"),
+ "Cut1"=0, "Cut2"=0)
> statbookm[1,2] <- round(median(bookmarks$Cut1), digits=2)
> statbookm[1,3] <- round(median(bookmarks$Cut2), digits=2)
> statbookm[2,2] <- round(mean(bookmarks$Cut1), digits=2)
> statbookm[2,3] <- round(mean(bookmarks$Cut2), digits=2)
> statbookm[3,2] <- round(sd(bookmarks$Cut1), digits=2)
> statbookm[3,3] <- round(sd(bookmarks$Cut2), digits=2)
> (statbookm)
> table(bookmarks$Cut1)
> table(bookmarks$Cut2)
```

Listing 2: Cut-Score Berechnung: Jede Bookmark repräsentiert ein Item, das eine bestimmte

Itemschwierigkeit hat. Die Cut-Scores lassen sich berechnen, in dem man die unterliegenden Itemschwierigkeiten der Bookmarks mittelt.

```
> bm.cut <- NULL
> bm.cut$cut1 <- mean(ratings$MB_Norm_rp23[bookmarks$Cut1])
> bm.cut$cut2 <- mean(ratings$MB_Norm_rp23[bookmarks$Cut2])
> bm.cut$cut1sd <- sd(ratings$MB_Norm_rp23[bookmarks$Cut1])
> bm.cut$cut2sd <- sd(ratings$MB_Norm_rp23[bookmarks$Cut2])
```

Listing 3: Standardfehler des Cut-Scores: Der Standardfehler wird berechnet, um eine mögliche Streuung des Cut-Scores zu berichten.

```
> se.cut1 <- bm.cut$cut1sd/sqrt(nraters)
> se.cut2 <- bm.cut$cut2sd/sqrt(nraters)
```

Listing 4: Impact Data: Mithilfe von Impact Data wird auf Basis von pilotierten Daten geschätzt, welche Auswirkungen die Cut-Scores auf die Schülerpopulation hätten (d.h., wie sich die Schülerinnen und Schüler auf die Stufen verteilen würden). Für diese Schätzung werden die Personenparameter herangezogen. Anschließend wird die Verteilung der Personenparameter entsprechend der Cut-Scores unterteilt. Die Prozentangaben der Schülerinnen und Schüler, die eine bestimmte Stufe erreichen, dienen dem Experten-Panel als Diskussionsgrundlage.

```
> Pers.Para <- sdat[, "TPV1"]
> cuts <- c(bm.cut$cut1, bm.cut$cut2)
> # Definiere Bereiche: Minimaler Personenparameter bis Cut-Score 1,
> # Cut-Score 1 bis Cut-Score 2, Cut-Score 2 bis maximaler
> # Personenparameter
> Cuts.Vec <- c(min(Pers.Para)-1, cuts, max(Pers.Para)+1)
> # Teile Personenparameter in entsprechende Bereiche auf
> Kum.Cuts <- cut(Pers.Para, breaks = Cuts.Vec)
> # Verteilung auf die einzelnen Bereiche
> Freq.Pers.Para <- xtabs(~ Kum.Cuts)
> nstud <- nrow(sdat)
> # Prozent-Berechnung
> prozent <- round(as.numeric(Freq.Pers.Para / nstud * 100),
+ digits = 2)
> (Impact.Data <- data.frame("Stufe" = c("A1", "A2", "B1"),
+ "Prozent" = prozent))
```

Abschnitt 3.3.3: Daten aus der Contrasting-Groups-Methode:

Listing 1: Cut-Scores: Hier wird der Cut-Score für den produktiven Bereich Schreiben berechnet, die Basis ist dabei die Personenfähigkeeit. Dabei wird pro Rater vorgegangen. Für jeden Rater werden dabei zwei Gruppen gebildet - Texte, die auf die untere Stufe eingeteilt wurden und Texte, die auf die obere Stufe eingeteilt wurden. Von beiden Gruppen wird jeweils der Mittelwert der Personenfähigkeit berechnet und anschließend der Mittelwert zwischen diesen beiden Gruppen. Wurde das für alle Raters durchgeführt, können die individuell gesetzten Cut-Scores

wiederum gemittelt werden und die Standard-Abweichung sowie der Standardfehler berechnet werden.

```
> raterID <- grep("R", colnames(productive), value = TRUE)
> nraters <- length(raterID)
> nscripts <- nrow(productive)
> # Berechne Cut-Score für jeden Rater
> cutscore <- data.frame("rater"=raterID, "cut1.ges"=NA)
> for(ii in 1:length(raterID)){
+ rater <- raterID[ii]
+ rates.ii <- productive[ ,grep(rater, colnames(productive))]
+ mean0.ii <- mean(productive$Performance[rates.ii == 0], na.rm = T)
+ mean1.ii <- mean(productive$Performance[rates.ii == 1], na.rm = T)
+ mean.ii <- mean(c(mean1.ii, mean0.ii), na.rm = T)
+ cutscore[ii, "cut1.ges"] <- mean.ii }
> # Finaler Cut-Score
> cut1 <- mean(cutscore$cut1.ges)
> sd.cut1 <- sd(cutscore$cut1.ges)
> se.cut1 <- sd.cut1/sqrt(nraters)
```

Appendix: Abbildungen im Buch: Hier ist der R-Code für die im Buch abgedruckten Grafiken zu finden.

Abbildung 3.1: In einem nächsten Schritt wird anhand des mittleren Kappa und der dazugehörigen Standard-Abweichung eine Grafik erstellt, um die Übereinstimmung eines Raters mit allen anderen Ratern dazustellen. Dafür wird zunächst ein Boxplot des mittleren Kappa pro Rater erzeugt. In einem zweiten Schritt werden die mittleren Kappas mit der dazugehörigen Standard-Abweichung abgetragen. Linien markieren 1.5 Standard-Abweichungen vom Mittelwert. Raters, die über oder unter dieser Grenze liegen, werden gekennzeichnet.

```
> # GRAFIK
> # 1. Grafik
> par(fig=c(0, 1, 0, 0.35), oma=c(0,0,3,0), cex = 0.85)
> boxplot(Kappa.Stat$MW_Kappa, horizontal = T, ylim=c(0.42,0.66),
+ axes = F, xlab = "MW Kappa")
> # 2. Grafik wird hinzugefügt
> par(fig=c(0, 1, 0.2, 1), new=TRUE)
> sd.factor <- 1.5
> mmw <- mean(Kappa.Stat$MW_Kappa)
> sdmw <- sd(Kappa.Stat$MW_Kappa)
> #Grenzwerte für MW und SD werden festgelegt
> mwind <- c(mmw-(sd.factor*sdmw), mmw+(sd.factor*sdmw))
> plot(Kappa.Stat$MW_Kappa, Kappa.Stat$SD_Kappa, xlab = "",
+ ylab = "SD Kappa", type = "n", xlim = c(0.42, 0.66),
+ ylim = c(0, 0.2))
> abline(v = mwind, col="grey", lty = 2)
> # Rater mit 1.5 SD Abweichung vom MW werden grau markiert
> abw.rater <- which(Kappa.Stat$MW_Kappa < mwind[1] |
+ Kappa.Stat$MW_Kappa > mwind[2])
```

```
> points(Kappa.Stat$MW_Kappa[-abw.rater],
+ Kappa.Stat$SD_Kappa[-abw.rater],
+ pch = 19)
> points(Kappa.Stat$MW_Kappa[abw.rater],
+ Kappa.Stat$SD_Kappa[abw.rater],
+ pch = 25, bg = "grey")
> text(Kappa.Stat$MW_Kappa[abw.rater],
+ Kappa.Stat$SD_Kappa[abw.rater],
+ Kappa.Stat$Person[abw.rater],
+ pos = 3)
> title("Rater-Analysen: MW und SD Kappa aus der IDM-Methode",
+ outer = TRUE)
```

Abbildung 3.2: Um das Feedback über die Setzung der Bookmarks an das Experten-Panel einfacher zu gestalten, wird eine Grafik erstellt. Darin sieht man pro Cut-Score, wo die Raters ihre Bookmarks (d.h. Seitenzahl im OIB) gesetzt haben, sowie Info über den Mittelwert dieser Bookmarks. Diese Grafik soll die Diskussion fördern.

```
> nitems <- 60
>
> library(lattice)
> library(gridExtra)
> #Erster Plot mit Mittelwert
> plot.Cut1 <- dotplot(bookmarks$Rater ~ bookmarks$Cut1, col = "black",
+ panel = function(...){
+ panel.dotplot(...)
+ panel.abline(v = mean(bookmarks$Cut1), lty = 5)
+ },
+ xlab = "Bookmarks für Cut-Score 1 (Seite im OIB)",
+ ylab = "Raters", cex = 1.3)
> #Zweiter Plot mit Mittelwert
> plot.Cut2 <- dotplot(bookmarks$Rater ~ bookmarks$Cut2, col = "black",
+ panel = function(...){
+ panel.dotplot(...)
+ panel.abline(v = mean(bookmarks$Cut2), lty = 5)
+ },
+ xlab = "Bookmarks für Cut-Score 2 (Seite im OIB)",
+ ylab = "Raters", cex = 1.3)
> #Plots nebeneinander anordnen
> grid.arrange(plot.Cut1, plot.Cut2, nrow = 1, top = "Bookmarks pro Rater/in")
```

Author(s)

Claudia Luger-Bazinger, Roman Freunberger, Ursula Itzlinger-Bruneforth

References

Luger-Bazinger, C., Freunberger, R. & Itzlinger-Bruneforth, U. (2016). Standard-Setting. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit* R*: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 83–110). Wien: facultas.

See Also

Zu datenKapitel03, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 2, Stichprobenziehung.
Zu Kapitel 4, Differenzielles Itemfunktionieren in Subgruppen.
Zur Übersicht.

Examples

```
## Not run:
library(car)
library(irr)
library(prettyR)
library(lattice)
library(gridExtra)

data(datenKapitel03)
ratings <- datenKapitel03$ratings
bookmarks <- datenKapitel03$bookmarks
sdat <- datenKapitel03$sdat
productive <- datenKapitel03$productive

## ------------------------------------------------------------
## Abschnitt 3.2.2: Daten aus der IDM-Methode
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 1: Feedback
#

raterID <- grep("R", colnames(ratings), value = TRUE)
nraters <- length(raterID)
nitems <- nrow(ratings)
itemID <- ratings[, 1]
itemdiff <- ratings[, 2]
stufen <- c(1, 2, 3) # Anzahl der Kompetenzstufen
item.freq <- data.frame()
# Berechne Prozentuelle Zuteilungen auf Stufen pro Item
tabelle.ii <- data.frame()
for(ii in 1:nitems){
  tabelle.ii <- round(table(factor(as.numeric(ratings[ii,
    raterID]), levels = stufen)) / nraters * 100, digits = 2)
  item.freq <- rbind(item.freq, tabelle.ii) }
colnames(item.freq) <- paste0("Level_", stufen)
item.freq <- data.frame(ratings[, 1:2], item.freq)
head(item.freq, 3)
```

```
# Anmerkung: Item 3 zu 100% auf Stufe 1, Item 2 aufgeteilt
# auf Stufe 1 und 2

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 1a: Ergänzung zum Buch
# GRAFIK-Erzeugung
#

# Farben für die Grafik definieren
c1 <- rgb(239/255, 214/255, 67/255)
c2 <- rgb(207/255, 151/255, 49/255)
c3 <- rgb(207/255, 109/255, 49/255)

# Aufbereitung Tabelle für Grafik
freq.dat <- t(as.matrix(item.freq[1:nitems,(3:(2+length(stufen)))]))
barcol <- c("black", "gray", "white")

#Grafik wird erzeugt
par(mfcol=c(3,1), oma=c(0,0,3,0)) # Angeben der Plot-Anzahl
perplot <- round(nitems/3)
a <- perplot + 1
b <- perplot*2
c <- b + 1
d <- perplot*3
barplot(freq.dat[,1 : perplot], col = barcol, beside = T,
        names.arg = seq(1 , perplot), xlab = "Itemnummer (Seitenzahl im OIB)",
        ylab = "% Zuteilung auf Stufe", horiz = F, ylim = range(1:100))
barplot(freq.dat[, a:b], col = barcol, beside = T, names.arg = seq(a, b),
        xlab = "Itemnummer (Seitenzahl im OIB)",
        ylab = "% Zuteilung auf Stufe",
        horiz = F, ylim = range(1:100))
barplot(freq.dat[, c:d], col = barcol, beside = T, names.arg = seq(c, d),
        xlab = "Itemnummer (Seitenzahl im OIB)",
        ylab = "% Zuteilung auf Stufe",
        horiz = F, ylim = range(1:100))
title("Feedback für das Experten-Panel aus der IDM-Methode", outer = T)

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 2: Cut-Score Berechnung
#

library(car)
# Rekodieren
rate.i <- ratings[which(ratings$R01 %in% c(2, 3)),
                  c("Norm_rp23", "R01")]
rate.i$R01 <-  recode(rate.i$R01, "2=0; 3=1")
coef(cut.i <- glm(rate.i$R01  ~ rate.i$Norm_rp23 ,
                  family = binomial(link="logit")))
# Berechnung des Cut-Scores laut Formel
cut.R01 <- (-cut.i$coefficients[1])/ cut.i$coefficients[2]

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 3: Rater-Analysen
```

```
#

library(irr)
# Auswahl der Ratings
rater.dat <- ratings[ ,grep("R", colnames(ratings))]
# Berechne Kappa von jeder Person mit allen anderen Personen
kappa.mat <- matrix(NA, nraters, nraters)
for(ii in 1:nraters){
  rater.eins <- rater.dat[, ii]
  for(kk in 1:nraters){
    rater.zwei <- rater.dat[ ,kk]
    dfr.ii <- cbind(rater.eins, rater.zwei)
    kappa.ik <- kappa2(dfr.ii)
    kappa.mat[ii, kk] <- kappa.ik$value }}
diag(kappa.mat) <- NA
# Berechne Mittleres Kappa für jede Person
MW_Kappa <- round(colMeans(kappa.mat, na.rm=T), digits=2)
SD_Kappa <- round(apply(kappa.mat, 2, sd, na.rm=T), digits=2)
(Kappa.Stat <- data.frame("Person"= raterID, MW_Kappa,
  SD_Kappa))

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 4: Berechnung Fleiss' Kappa
#

kappam.fleiss(rater.dat)

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 5: Modalwerte
#

library(prettyR)
# Berechne Modalwert
mode <- as.numeric(apply(rater.dat, 1, Mode))
# Korrelation für die Ratings jeder Person im Panel mit den
# Modalwerten der Items
corr <- data.frame()
for(z in raterID){
  rater.ii <- rater.dat[, (grep(z, colnames(rater.dat)))]
  cor.ii <- round(cor(mode, rater.ii, method = "spearman",
    use = "pairwise.complete.obs"), digits = 2)
  corr <- rbind(corr, cor.ii)
}
corr[, 2] <- raterID
colnames(corr) <- c("Korrelation", "Rater")
# Aufsteigende Reihenfolge
(corr <- corr[order(corr[, 1]),])

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 5: Ergänzung zum Buch
# GRAFIK-Erzeugung und ICC
#
```

```
# Grafik
plot(corr$Korrelation, xlab = NA, ylab = "Korrelation",
     ylim = c(0.5, 1), xaxt = "n", main = "Korrelation zwischen
     Modalwert und individueller Zuordnung der Items pro Rater/in")
text(seq(1:nraters), corr$Korrelation - 0.02, labels = corr[, 2],
     offset = 1, cex = 1)
title(xlab = "Raters nach aufsteigender Korrelation gereiht")

# ------------------------------------------------------------
# Abschnitt 3.2.2, Listing 6: ICC
#

library(irr)
(iccdat.agree <- icc(rater.dat, model = "twoway",
  type = "agreement", unit = "single", r0 = 0, conf.level=0.95))
(iccdat.cons <- icc(rater.dat, model = "twoway",
  type = "consistency", unit = "single", r0 = 0, conf.level=0.95))


## ------------------------------------------------------------
## Abschnitt 3.2.3: Daten aus der Bookmark-Methode
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 3.2.3, Listing 1: Feedback
#

head(bookmarks)
statbookm <- data.frame("Stats"=c("Md","Mean","SD"),
                        "Cut1"=0, "Cut2"=0)
statbookm[1,2] <- round(median(bookmarks$Cut1), digits=2)
statbookm[1,3] <- round(median(bookmarks$Cut2), digits=2)
statbookm[2,2] <- round(mean(bookmarks$Cut1), digits=2)
statbookm[2,3] <- round(mean(bookmarks$Cut2), digits=2)
statbookm[3,2] <- round(sd(bookmarks$Cut1), digits=2)
statbookm[3,3] <- round(sd(bookmarks$Cut2), digits=2)
(statbookm)
table(bookmarks$Cut1)
table(bookmarks$Cut2)

# ------------------------------------------------------------
# Abschnitt 3.2.3, Listing 2: Cut-Score Berechnung
#

bm.cut <- NULL
bm.cut$cut1 <- mean(ratings$Norm_rp23[bookmarks$Cut1])
bm.cut$cut2 <- mean(ratings$Norm_rp23[bookmarks$Cut2])
bm.cut$cut1sd <- sd(ratings$Norm_rp23[bookmarks$Cut1])
bm.cut$cut2sd <- sd(ratings$Norm_rp23[bookmarks$Cut2])

# ------------------------------------------------------------
# Abschnitt 3.2.3, Listing 3: Standardfehler des Cut-Scores
#
```

```
se.cut1 <- bm.cut$cut1sd/sqrt(nraters)
se.cut2 <- bm.cut$cut2sd/sqrt(nraters)

# ------------------------------------------------------------
# Abschnitt 3.2.3, Listing 4: Impact Data
#

Pers.Para <- sdat[, "TPV1"]
cuts <- c(bm.cut$cut1, bm.cut$cut2)
# Definiere Bereiche: Minimaler Personenparameter bis Cut-Score 1,
#   Cut-Score 1 bis Cut-Score 2, Cut-Score 2 bis maximaler
#   Personenparameter
Cuts.Vec <- c(min(Pers.Para)-1, cuts, max(Pers.Para)+1)
# Teile Personenparameter in entsprechende Bereiche auf
Kum.Cuts <- cut(Pers.Para, breaks = Cuts.Vec)
# Verteilung auf die einzelnen Bereiche
Freq.Pers.Para <- xtabs(~ Kum.Cuts)
nstud <- nrow(sdat)
# Prozent-Berechnung
prozent <- round(as.numeric(Freq.Pers.Para / nstud * 100),
                 digits = 2)
(Impact.Data <- data.frame("Stufe" = c("A1", "A2", "B1"),
                           "Prozent" = prozent))


## ------------------------------------------------------------
## Abschnitt 3.3.2: Daten aus der Contrasting-Groups-Methode
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 3.3.2, Listing 1: Cut-Scores
#

raterID <- grep("R", colnames(productive), value = TRUE)
nraters <- length(raterID)
nscripts <- nrow(productive)
# Berechne Cut-Score für jeden Rater
cutscore <- data.frame("rater"=raterID, "cut1.ges"=NA)
for(ii in 1:length(raterID)){
  rater <- raterID[ii]
  rates.ii <- productive[ ,grep(rater, colnames(productive))]
  mean0.ii <- mean(productive$Performance[rates.ii == 0],
    na.rm = TRUE)
  mean1.ii <- mean(productive$Performance[rates.ii == 1],
    na.rm = TRUE)
  mean.ii <- mean(c(mean1.ii, mean0.ii), na.rm = TRUE)
  cutscore[ii, "cut1.ges"] <- mean.ii }
# Finaler Cut-Score
cut1 <- mean(cutscore$cut1.ges)
sd.cut1 <- sd(cutscore$cut1.ges)
se.cut1 <- sd.cut1/sqrt(nraters)
```

```
## ------------------------------------------------------------
## Appendix: Abbildungen
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abbildung 3.1
#

# 1. Grafik
par(fig=c(0, 1, 0, 0.35), oma=c(0,0,3,0), cex = 0.85)
boxplot(Kappa.Stat$MW_Kappa, horizontal = T, ylim=c(0.42,0.66),
        axes = F, xlab = "MW Kappa")
# 2. Grafik wird hinzugefügt
par(fig=c(0, 1, 0.2, 1), new=TRUE)
sd.factor <- 1.5
mmw <- mean(Kappa.Stat$MW_Kappa)
sdmw <- sd(Kappa.Stat$MW_Kappa)
#Grenzwerte für MW und SD werden festgelegt
mwind <- c(mmw-(sd.factor*sdmw), mmw+(sd.factor*sdmw))
plot(Kappa.Stat$MW_Kappa, Kappa.Stat$SD_Kappa, xlab = "",
     ylab = "SD Kappa", type = "n", xlim = c(0.42, 0.66),
     ylim = c(0, 0.2))
abline(v = mwind, col="grey", lty = 2)
# Rater mit 1.5 SD Abweichung vom MW werden grau markiert
abw.rater <- which(Kappa.Stat$MW_Kappa < mwind[1] |
                     Kappa.Stat$MW_Kappa > mwind[2])
points(Kappa.Stat$MW_Kappa[-abw.rater],
       Kappa.Stat$SD_Kappa[-abw.rater],
       pch = 19)
points(Kappa.Stat$MW_Kappa[abw.rater],
       Kappa.Stat$SD_Kappa[abw.rater],
       pch = 25, bg = "grey")
text(Kappa.Stat$MW_Kappa[abw.rater],
     Kappa.Stat$SD_Kappa[abw.rater],
     Kappa.Stat$Person[abw.rater],
     pos = 3)
title("Rater-Analysen: MW und SD Kappa aus der IDM-Methode",
      outer = TRUE)

# ------------------------------------------------------------
# Abbildung 3.2
#

nitems <- 60

library(lattice)
library(gridExtra)
#Erster Plot mit Mittelwert
plot.Cut1 <- dotplot(bookmarks$Rater ~ bookmarks$Cut1, col = "black",
                     panel = function(...){
                       panel.dotplot(...)
                       panel.abline(v = mean(bookmarks$Cut1), lty = 5)
```

```
                    },
                    xlab = "Bookmarks für Cut-Score 1 (Seite im OIB)",
                    ylab = "Raters", cex = 1.3)
#Zweiter Plot mit Mittelwert
plot.Cut2 <- dotplot(bookmarks$Rater ~ bookmarks$Cut2, col = "black",
                    panel = function(...){
                      panel.dotplot(...)
                      panel.abline(v = mean(bookmarks$Cut2), lty = 5)
                    },
                    xlab = "Bookmarks für Cut-Score 2 (Seite im OIB)",
                    ylab = "Raters", cex = 1.3)
#Plots nebeneinander anordnen
grid.arrange(plot.Cut1, plot.Cut2, nrow = 1, top = "Bookmarks pro Rater/in")


## End(Not run)
```

Kapitel 4 *Kapitel 4: Differenzielles Itemfunktionieren in Subgruppen*

Description

Das ist die Nutzerseite zum Kapitel 4, *Differenzielles Itemfunktionieren in Subgruppen*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Author(s)

Matthias Trendtel, Franziska Schwabe, Robert Fellinger

References

Trendtel, M., Schwabe, F. & Fellinger, R. (2016). Differenzielles Itemfunktionieren in Subgruppen. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 111–147). Wien: facultas.

See Also

Zu datenKapitel04, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 3, Standard-Setting.
Zu Kapitel 5, Testdesign.
Zur Übersicht.

Examples

```
## Not run:
library(difR)
library(mirt)
library(sirt)
library(TAM)
set.seed(12345)

data(datenKapitel04)
dat <- datenKapitel04$dat
dat.th1 <- datenKapitel04$dat.th1
ibank <- datenKapitel04$ibank

## ------------------------------------------------------------
## Abschnitt 4.4.1 DIF-Analysen für vollständige Daten
## ------------------------------------------------------------

items.th1 <- grep("E8R", colnames(dat.th1), value=T)
resp <- dat.th1[, items.th1]
AHS <- dat.th1$AHS

# ------------------------------------------------------------
# Abschnitt 4.4.1, Listing 1: Mantel-Haenszel
#

difMH(Data = resp, group = AHS, correct = F, focal.name = 0)

# ------------------------------------------------------------
# Abschnitt 4.4.1, Listing 2: Standardisierte p-Wert Differenzen
#

difStd(Data = resp, group = AHS, focal.name = 0)

# ------------------------------------------------------------
# Abschnitt 4.4.1, Listing 3: SIBTEST
#

SIBTEST(dat = resp, group = AHS, focal_name = 0,
        focal_set = grep("E8RS03131", items.th1))
SIBTEST(dat = resp, group = AHS, focal_name=0,
        focal_set = grep("E8RS15621", items.th1))

# ------------------------------------------------------------
# Abschnitt 4.4.1, Listing 4: Methode nach Lord
#

difLord(Data = resp, group = AHS, focal.name = 0,
        model = "1PL")

# ------------------------------------------------------------
# Abschnitt 4.4.1, Listing 5: Zusammenschau
#
```

```
dichoDif(Data = resp, group = AHS, correct = F, focal.name = 0,
         method = c("MH", "Std", "Lord"), model = "1PL")


## ------------------------------------------------------------
## Abschnitt 4.4.2 DIF-Analysen für unvollständige Daten
## ------------------------------------------------------------

items <- grep("E8R", colnames(dat), value = T)
resp <- dat[ ,items]
AHS <- dat$AHS

# ------------------------------------------------------------
# Abschnitt 4.4.2, Listing 1: Matching-Variable setzen
#

score <- rowSums(resp, na.rm=T)

# ------------------------------------------------------------
# Abschnitt 4.4.2, Listing 2: Durchführung Logistische Regression
#

difLR <- dif.logistic.regression(resp, group = AHS, score = score)

# ------------------------------------------------------------
# Abschnitt 4.4.2, Listing 3: Durchführung Logistische Regression
#                             mit angepasster Referenzgruppe
#

difLR <- dif.logistic.regression(resp, AHS==0, score)

# ------------------------------------------------------------
# Abschnitt 4.4.2, Listing 4: Ausgabe erster Teil
#

cbind(item = difLR$item, round(difLR[, 4:13], 3))

# ------------------------------------------------------------
# Abschnitt 4.4.2, Listing 5: Ausgabe zweiter Teil
#

cbind(difLR[, c(3,14:16)], sign = difLR[, 17], ETS = difLR[, 18])

# ------------------------------------------------------------
# Abschnitt 4.4.2, Listing 6: DIF-Größen
#

table(difLR[, 17], difLR[, 18])

difLR[c(10, 18), c(3, 14, 17:18)]

# ------------------------------------------------------------
```

```
# Abschnitt 4.4.2, Listing 7: Ausgabe dritter Teil
#

cbind(difLR[, c(3, 21:23)], sign=difLR[, 24])


## ------------------------------------------------------------
## Abschnitt 4.4.3 Hypothesenprüfung mit GLMM
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 1: Itemauswahl
#

HO.items <- ibank[ibank$format == "ho", "task"]

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 2: Facettenidentifikation
#

facets <- data.frame(AHS = dat$AHS)
form <- formula( ~ item * AHS)

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 3: Initiierung des Designs
#

design <- designMatrices.mfr(resp = dat[, items],
                             formulaA = form, facets = facets)

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 4: Übergabe der Designmatrix und des
#                             erweiterten Responsepatterns
#

A <- design$A$A.3d[, , 1:(length(items) + 2)]
dimnames(A)[[3]] <- c(items, "AHS", "HO:AHS")
resp <- design$gresp$gresp.noStep

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 5: Ausgabe der ersten Zeilen des
#                             Responsepatterns
#

head(resp)

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 6: Identifikation Itemformat X Gruppe
#

HO.AHS0 <- paste0(HO.items, "-AHS0")
HO.AHS1 <- paste0(HO.items, "-AHS1")
```

```
# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 7: Spezifizierung des Designs
#

A[, , "HO:AHS"] <- 0
A[HO.AHS0, 2, "HO:AHS"] <- -1; A[HO.AHS1, 2, "HO:AHS"] <-  1

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 8: Ausgabe der Designmatrix für
#                             Itemkategorie 'richtig beantwortet'
#

A[,2,c("AHS", "HO:AHS")]

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 9: Schätzen des Modells
#

mod <- tam.mml(resp = resp, A=A)

# ------------------------------------------------------------
# Abschnitt 4.4.3, Listing 10: Ausgabe der Parameterschätzer
#

summary(mod)

## End(Not run)
```

Kapitel 5 *Kapitel 5: Testdesign*

Description

Das ist die Nutzerseite zum Kapitel 5, *Testdesign*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Author(s)

Thomas Kiefer, Jörg-Tobias Kuhn, Robert Fellinger

References

Kiefer, T., Kuhn, J.-T. & Fellinger, R. (2016). Testdesign. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 149–184). Wien: facultas.

See Also

Zurück zu Kapitel 4, Differenzielles Itemfunktionieren in Subgruppen.
Zu Kapitel 6, Skalierung und Linking.
Zur Übersicht.

Examples

```
## Not run:
library(tensor)
set.seed(1337)

data(datenKapitel05)
dat.ib <- datenKapitel05$tdItembank
dat.bib <- datenKapitel05$tdBib2d
dat.bibPaare <- datenKapitel05$tdBibPaare

## ------------------------------------------------------------
## Abschnitt 5.3.2: ATA Methode für das Blockdesign
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 1: Initialisierung
#

library(tensor)

nTh <- 30
nPos <- 6
nBl <- 30
inc <- array(0, dim = c(nTh, nPos, nBl))

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 2: Startdesign
#

for(tt in 1:nTh){
  inc[tt, , sample(1:nBl, nPos)] <- diag(1, nPos)
}

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 3: Zielfunktion
#
des <- inc
desAllePos <- tensor(des, rep(1, nPos), 2, 1)

blockPaarInd <- upper.tri(diag(nrow = nBl))
blockPaar <- crossprod(desAllePos)[blockPaarInd]

err.bb <- blockPaar
err.bb[blockPaar >= 2] <- blockPaar[blockPaar >= 2] - 2
err.bb[blockPaar <= 1] <- 1 - blockPaar[blockPaar <= 1]
```

```
objective <- sum(err.bb) / length(err.bb)
objWgt <- 2^0

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 4: Studienzuweisung
#

blMatching <- seq(6, nBl, 6)

nbStatus <- list(
  (desAllePos[1:6, -(1:12)] > 0) / (6 * 18),      # 1
  (desAllePos[25:30, -(19:30)] > 0) / (6 * 18),   # 2
  (rowSums(desAllePos[, blMatching]) != 1) / nTh  # 3
)
nbStatus <- unlist(lapply(nbStatus, sum))

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 5: Erweiterung Positionsbalancierung
#

# 4
nbPos <- sum((colSums(des) != 1) / (nPos * nBl))
# 5
nbPos.pLSA <- list(
  (colSums(des[1:6, 1:2, 1:12], dims = 2) != 1) / 12,
  (colSums(des[1:6, 3:4, 1:12], dims = 2) != 1) / 12,
  (colSums(des[1:6, 5:6, 1:12], dims = 2) != 1) / 12
)
nbPos.pLSA <- sum(unlist(lapply(nbPos.pLSA, sum)) / 3)
# 6
nbPos.link <- list(
  (colSums(des[25:30, 1:2, 19:30], dims = 2) != 1) / 12,
  (colSums(des[25:30, 3:4, 19:30], dims = 2) != 1) / 12,
  (colSums(des[25:30, 5:6, 19:30], dims = 2) != 1) / 12
)
nbPos.link <- sum(unlist(lapply(nbPos.link, sum)) / 3)

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 6: Zusammenfügen
#

nb <- c(nbStatus, nbPos, nbPos.pLSA, nbPos.link)
nbWgt <- c(
  rep(2^5, length(nbStatus)),
  rep(2^6, length(nbPos)),
  rep(2^4, length(nbPos.pLSA)),
  rep(2^3, length(nbPos.link))
)

nbWgt.norm <- nbWgt / (sum(nbWgt) + objWgt)
objWgt.norm <- objWgt / (sum(nbWgt) + objWgt)
oDes <- objWgt.norm %*% objective + nbWgt.norm %*% nb
```

```
# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 6a: Ergänzung zum Buch
#
#

fit <- function(des){
  desAllePos <- tensor(des, rep(1, nPos), 2, 1)

  #
  blockPaarInd <- upper.tri(diag(nrow = nBl))
  blockPaar <- crossprod(desAllePos)[blockPaarInd]

  err.bb <- blockPaar
  err.bb[blockPaar >= 2] <- blockPaar[blockPaar >= 2] - 2
  err.bb[blockPaar <= 1] <- 1 - blockPaar[blockPaar <= 1]

  objective <- sum(err.bb) / length(err.bb)
  objWgt <- 2^0

  #
  nbStatus <- list(
    (desAllePos[1:6, -(1:12)] > 0) / (6 * 18),      # 1
    (desAllePos[25:30, -(19:30)] > 0) / (6 * 18),   # 2
    (rowSums(desAllePos[, blMatching]) != 1) / nTh  # 3
  )
  nbStatus <- unlist(lapply(nbStatus, sum))

  # 4
  nbPos <- sum((colSums(des) != 1) / (nPos * nBl))
  # 5
  nbPos.pLSA <- list(
    (colSums(des[1:6, 1:2, 1:12], dims = 2) != 1) / 12,
    (colSums(des[1:6, 3:4, 1:12], dims = 2) != 1) / 12,
    (colSums(des[1:6, 5:6, 1:12], dims = 2) != 1) / 12
  )
  nbPos.pLSA <- sum(unlist(lapply(nbPos.pLSA, sum)) / 3)
  # 6
  nbPos.link <- list(
    (colSums(des[25:30, 1:2, 19:30], dims = 2) != 1) / 12,
    (colSums(des[25:30, 3:4, 19:30], dims = 2) != 1) / 12,
    (colSums(des[25:30, 5:6, 19:30], dims = 2) != 1) / 12
  )
  nbPos.link <- sum(unlist(lapply(nbPos.link, sum)) / 3)

  #
  nb <- c(nbStatus, nbPos, nbPos.pLSA, nbPos.link)
  nbWgt <- c(
    rep(2^5, length(nbStatus)),
    rep(2^6, length(nbPos)),
    rep(2^4, length(nbPos.pLSA)),
    rep(2^3, length(nbPos.link))
  )
  nbWgt.norm <- nbWgt / (sum(nbWgt) + objWgt)
```

```
  objWgt.norm <- objWgt / (sum(nbWgt) + objWgt)
  oDes <- objWgt.norm %*% objective + nbWgt.norm %*% nb

  return(oDes)
}

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 7: Initialisierung des Algorithmus
#

# t <- 1; t.min <- 1e-5; c <- 0.7; L <- 10000; l <- 1
t <- 1; tMin <- 1e-5; c <- 0.9; L <- 100000; l <- 1

fitInc <- fit(inc)

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 8: Störung
#

thisTh <- (l - 1) %% nTh + 1
child <- inc

bloeckeTh <- which(colSums(child[thisTh, , ]) == 1)
raus <- sample(bloeckeTh, 1)
rein <- sample(setdiff(1:nBl, bloeckeTh), 1)

child[thisTh, , rein] <- child[thisTh, , raus]
child[thisTh, , raus] <- 0


# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 9: Survival
#

fitChild <- fit(child)

behalte <- fitChild < fitInc
if(!behalte){
  pt <- exp(-(fitChild - fitInc) / t)
  behalte <- runif(1) <= pt
}

if(behalte){
  inc <- child
  fitInc <- fitChild
}

# ------------------------------------------------------------
# Abschnitt 5.3.2, Listing 9a: Ergänzung zum Buch
#

# Achtung: Algorithmus benötigt einige Zeit.
# Je nach Wahl der Lauf-Parameter in Abschnitt 5.3.2, Listing 7, kann der
```

```
# folgende Prozess bis zu ein paar Stunden dauern.

start <- Sys.time()
best <- list(inc, fitInc)
while(t > tMin){
  while(l < L){
    thisTh <- (l - 1) %% nTh + 1
    child <- inc

    # Perturbation
    bloeckeTh <- which(colSums(child[thisTh, , ]) == 1)
    raus <- sample(bloeckeTh, 1)
    rein <- sample(setdiff(1:nBl, bloeckeTh), 1)

    child[thisTh, , rein] <- child[thisTh, , raus]
    child[thisTh, , raus] <- 0

    # Fit und Survival
    fitChild <- fit(child)

    behalte <- fitChild < fitInc
    if(!behalte){
      pt <- exp(-(fitChild - fitInc) / t)
      behalte <- runif(1) <= pt
    }

    if(behalte){
      inc <- child
      fitInc <- fitChild
    }

    # Kontroll-Ausgaben
    if(fitInc < best[[2]]){
      best <- list(inc, fitInc)
    }

    if (l %% 500 == 0) {
      cat("\r")
      cat(paste("l=", l),
          paste("t=", as.integer(log(t) / log(c) + 1)),
          paste("fit=", round(fitInc, 4)),
          paste("pt=", round(pt, 5)),
          sep=";   ")
      cat("                     ")
      flush.console()
    }
    l <- l + 1
  }
  l <- 1
  t <- t * c
}
end <- Sys.time()
```

```
tdBib2d <- apply(inc, 1, function(bb){
  this <- which(colSums(bb) > 0)
  this[order((1:nrow(bb) %*% bb)[this])]
})

## ------------------------------------------------------------
## Abschnitt 5.3.3: ATA Methode für die Item-zu-Block-Zuordnung
## ------------------------------------------------------------

set.seed(1338)

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 1: Initialisierung
#

nTh <- nrow(dat.bib)
nPos <- ncol(dat.bib)
nBl <- length(unique(unlist(dat.bib)))
blMatching <- seq(6, nBl, 6)

nI <- nrow(dat.ib)
itemsMatching <- which(dat.ib$format == "Matching")
itemsSonst <- which(dat.ib$format != "Matching")

# ------------------------------------------------------------
# Abschnitt 3.3, Listing 2: Startdesign
#

inc <- array(0, dim = c(nI, nBl))
for(bb in blMatching){
  inc[sample(itemsMatching, 2), bb] <- 1
}
for(bb in setdiff(1:nBl, blMatching)){
  inc[sample(itemsSonst, 7), bb] <- 1
}

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 3: Testheftebene
#

des <- inc
desTh <- des[, dat.bib[, 1]] + des[, dat.bib[, 2]] +
  des[, dat.bib[, 3]] + des[, dat.bib[, 4]] +
  des[, dat.bib[, 5]] + des[, dat.bib[, 6]]

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 4: IIF
#

theta <- c(380, 580)
InfoItem <- dat.ib[,grep("IIF", colnames(dat.ib))]
TIF <- (t(InfoItem) %*% desTh) / 37
```

```
objective <- - sum(TIF) / prod(dim(TIF))
objWgt <- 2^0

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 5: KEY
#

nbKey <- list(
  (colSums(desTh > 1) > 0) / nTh,              # 7
  ((rowSums(desTh[, 1:6]) > 0) +               # 8
     (rowSums(desTh[, 25:30]) > 0) > 1) / nI
)
nbKey <- unlist(lapply(nbKey, sum))
nbWgt <- 2^c(7, 6)

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 6: Kategorial
#

# 9
zFocus.block <- c(0, 1, 1, 1, 1, 2, 0)
gFocus.block <- rowsum(des[, -blMatching], dat.ib$focus) -
  zFocus.block
# 10
zFocus.form <- c(2, 6, 6, 6, 6, 13, 1)
gFocus.form <- rowsum(desTh, dat.ib$focus) - zFocus.form
# 11
gTopic.form <- rowsum(desTh, dat.ib$topic) - 4

nbKonstrukt <- list(
  colSums(gFocus.block < 0) / prod(dim(gFocus.block)),
  colSums(gFocus.form > 0) / prod(dim(gFocus.form)),
  colSums(gTopic.form > 0) / 30
)
nbKonstrukt <- unlist(lapply(nbKonstrukt, sum))
nbWgt <- c(nbWgt, 2^c(4, 4, 3))

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 7: Stetig
#

length.form <- ((dat.ib$audiolength + 13) %*% desTh) / 60
nbStetig <- list(
  (length.form > 32) / length(length.form),
  (length.form < 28) / length(length.form)
)
nbStetig <- unlist(lapply(nbStetig, sum))
nbWgt <- c(nbWgt, 2^c(3, 2))

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 8: Perturbation
#
```

```
thisBl <- 1
child <- inc

items.raus <- which(child[, thisBl] == 1)
raus <- sample(items.raus, 1)

bibPaar.bl <- dat.bibPaare[thisBl, ] != 0
items.bibPaare <- rowSums(child[, bibPaar.bl]) > 0
rein <- which(!items.bibPaare)

if(thisBl %in% blMatching){
  rein <- sample(intersect(rein, itemsMatching), 1)
}else{
  rein <- sample(intersect(rein, itemsSonst), 1)
}

child[c(raus, rein), thisBl] <- c(0, 1)

# ------------------------------------------------------------
# Abschnitt 5.3.3, Listing 8a: Ergänzung zum Buch
#                              Vollständige Umsetzung
#

# Achtung: Algorithmus benötigt einige Zeit.
# Je nach Wahl der Lauf-Parameter im nachfolgenden Abschnitt, kann der
# Prozess bis zu einigen Stunden dauern.

fit <- function(des, dat.ib, dat.bib){
  desTh <- des[, dat.bib[, 1]] + des[, dat.bib[, 2]] +
    des[, dat.bib[, 3]] + des[, dat.bib[, 4]] +
    des[, dat.bib[, 5]] + des[, dat.bib[, 6]]

  #
  TIF <- (t(InfoItem) %*% desTh) / 37

  objective <- - sum(TIF) / prod(dim(TIF))
  objWgt <- 2^0

  #
  nbKey <- list(
    (colSums(desTh > 1) > 0) / nTh,              # 7
    ((rowSums(desTh[, 1:6]) > 0) +               # 8
       (rowSums(desTh[, 25:30]) > 0) > 1) / nI
  )
  nbKey <- unlist(lapply(nbKey, sum))
  nbWgt <- 2^c(7, 6)

  # 9
  zFocus.block <- c(0, 1, 1, 1, 1, 2, 0)
  gFocus.block <- rowsum(des[, -blMatching], dat.ib$focus) -
    zFocus.block
  # 10
  zFocus.form <- c(2, 6, 6, 6, 6, 13, 1)
```

```
  gFocus.form <- rowsum(desTh, dat.ib$focus) - zFocus.form
  # 11
  gTopic.form <- rowsum(desTh, dat.ib$topic) - 4

  nbKonstrukt <- list(
    colSums(gFocus.block < 0) / prod(dim(gFocus.block)),
    colSums(gFocus.form > 0) / prod(dim(gFocus.form)),
    colSums(gTopic.form > 0) / 30
  )
  nbKonstrukt <- unlist(lapply(nbKonstrukt, sum))
  nbWgt <- c(nbWgt, 2^c(4, 4, 3))

  #
  length.form <- ((dat.ib$audiolength + 13) %*% desTh) / 60
  nbStetig <- list(
    (length.form > 32) / length(length.form),
    (length.form < 28) / length(length.form)
  )
  nbStetig <- unlist(lapply(nbStetig, sum))
  nbWgt <- c(nbWgt, 2^c(3, 2))

  #
  nb <- c(nbKey, nbKonstrukt, nbStetig)

  nbWgt.norm <- nbWgt / (sum(nbWgt) + objWgt)
  objWgt.norm <- objWgt / (sum(nbWgt) + objWgt)
  oDes <- objWgt.norm %*% objective + nbWgt.norm %*% nb

  return(oDes)
}

#
# t <- 1; tMin <- 1e-5; c <- 0.7; L <- 10000; l <- 1
# t <- 1; tMin <- 1e-5; c <- 0.8; L <- 25000; l <- 1
# t <- 1; tMin <- 1e-5; c <- 0.9; L <- 50000; l <- 1
t <- 1; tMin <- 1e-7; c <- 0.9; L <- 100000; l <- 1

#
fitInc <- fit(inc, dat.ib, dat.bib)
best <- list(inc, fitInc)
vers <- versBest <- 1
#
start <- Sys.time()
while(t > tMin){
  while(l < L){
    thisBl <- (l - 1) %% nBl + 1

    # Perturbation
    child <- inc

    items.raus <- which(child[, thisBl] == 1)
    raus <- sample(items.raus, 1)
```

```
    bibPaar.bl <- dat.bibPaare[thisBl, ] != 0
    items.bibPaare <- rowSums(child[, bibPaar.bl]) > 0
    rein <- which(!items.bibPaare)

    if(thisBl %in% blMatching){
      rein <- sample(intersect(rein, itemsMatching), 1)
    }else{
      rein <- sample(intersect(rein, itemsSonst), 1)
    }

    child[c(raus, rein), thisBl] <- c(0, 1)

    # Fit und Survival
    fitChild <- fit(child, dat.ib, dat.bib)

    behalte <- fitChild < fitInc
    if(!behalte){
      pt <- exp((fitInc - fitChild) / t)
      behalte <- runif(1) <= pt
    }

    if(behalte){
      inc <- child
      fitInc <- fitChild
    }

    if(fitInc < best[[2]]){
      best <- list(inc, fitInc)
      versBest <- versBest + 1
    }

    # Kontroll-Ausgaben; ggf. löschen
    if (identical(inc, child)) vers <- vers + 1
    if (l %% 500 == 0) {
      cat("\r")
      cat(paste("l=", l),
          paste("t=", as.integer(log(t) / log(c) + 1)),
          paste("versionen=", vers),
          paste("versionenBest=", versBest),
          paste("fit=", round(fitInc, 4)),
          paste("fitBest=", round(best[[2]], 4)),
          paste("pt=", round(pt, 5)),
          sep=";   ")
      cat("                     ")
      flush.console()
    }
    l <- l + 1
  }
  l <- 1
  t <- t * c
}
end <- Sys.time()
```

```
## End(Not run)
```

Kapitel 6 *Kapitel 6: Skalierung und Linking*

Description

Das ist die Nutzerseite zum Kapitel 6, *Skalierung und Linking*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Author(s)

Matthias Trendtel, Giang Pham, Takuya Yanagida

References

Trendtel, M., Pham, G. & Yanagida, T. (2016). Skalierung und Linking. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 185–224). Wien: facultas.

See Also

Zu datenKapitel06, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 5, Testdesign.
Zu Kapitel 7, Statistische Analysen produktiver Kompetenzen.
Zur Übersicht.

Examples

```
## Not run:
library(TAM)
library(sirt)
library(WrightMap)
library(miceadds)
library(plyr)
set.seed(20150528)

dat <- data(datenKapitel06)
# Hauptstudie
dat <- datenKapitel06$dat
ue <- datenKapitel06$itembank
items <- grep("I", colnames(dat), value=TRUE)

# Nur TH1
datTH1 <- datenKapitel06$datTH1
ueTH1 <- datenKapitel06$itembankTH1
```

```
rownames(ueTH1) <- ueTH1$Item
itemsTH1 <- grep("I", colnames(datTH1), value=TRUE)
respTH1 <- datTH1[, -(1:4)]; wTH1 <- datTH1$wgtstud

# Normierungsstudie
normdat <- datenKapitel06$normdat

## ----------------------------------------------------------------
## Abschnitt 6.3.4 Das Partial Credit Model (PCM)
## ----------------------------------------------------------------

# ----------------------------------------------------------------
# Abschnitt 6.3.4, Listing 1: Leistungsdaten und Stich-
#                             probengewichte Objekten zuweisen
#

resp <- dat[, grep("I", colnames(dat))]; w <- dat$wgtstud

# ----------------------------------------------------------------
# Abschnitt 6.3.4, Listing 2: Anpassen eines PCMs
#

mod.1PL <- tam.mml(resp = resp, irtmodel = "1PL", pweights = w)

# ----------------------------------------------------------------
# Abschnitt 6.3.4, Listing 2a: Ergänzung zum Buch
# Runden zur besseren Darstellung im Buch
#

mod.1PL$item$M <- round(mod.1PL$item$M, 2)

# ----------------------------------------------------------------
# Abschnitt 6.3.4, Listing 3: Darstellung des letzen Items
#

tail(mod.1PL$item, 1)

# ----------------------------------------------------------------
# Abschnitt 6.3.4, Listing 4: Umparametrisierung
#

b_ih <- mod.1PL$item[, grep("AXsi_", colnames(mod.1PL$item))]
delta.tau <- pcm.conversion(b_ih)

# ----------------------------------------------------------------
# Abschnitt 6.3.4, Listing 5: Berechnung der Thursonian
#                             Threshods und Lokations Indizes
#

thurst.thres <- IRT.threshold(mod.1PL)
LI <- IRT.threshold(mod.1PL, type="item")
```

```
## ------------------------------------------------------------
## Abschnitt 6.3.5 Itemtrennschärfen polytomer Items und
##                 Rateparameter
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 1: Anpassen eines Generalized
#                             Partial Credit Models
#

mod.GPCM <- tam.mml.2pl(resp, irtmodel = "GPCM", pweights = w)

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 2: Anpassen eines
#                             Nominal Item Response Models
#

mod.NIRM <- tam.mml.2pl(resp, irtmodel="2PL", pweights = w)

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 3: Anpassen eines Generalized
#                             Partial Credit Models mit festen
#                             Itemgewichten (Trennschärfen)
#

tammodel <- "
  LAVAAN MODEL:
  F =~ a1__a50*I1__I50;
  # Trait-Varianz auf 1 fixieren
  F ~~ 1*F
  MODEL CONSTRAINT:
  # Gewichtung für die Items festlegen
  a1__a40 == 1*a # dichotome Items
  a41__a44 == .3333*a # T/F Items mit max. Score von 3
  a45__a50 == .25*a # M56 Items mit max. Score von 4
  "
mod.GPCMr <- tamaan(tammodel, resp, pweights = w)

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 4: Itemtrennschärfevergleich
#

## Itemparameter im Vergleich
rbind(GPCM = mod.GPCM$item[50, 9:12],
      NIRM = mod.NIRM$item[50, 9:12],
      GPCMr = mod.GPCMr$item[50, 10:13]) / rep(c(1:4), each=3)

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 5: Itemtrennschärfen eines
#                             dichotomen und eines polytomen
#                             Items

rbind(I40 = mod.GPCMr$item[40, 10:13],
```

```
      I50 = mod.GPCMr$item[50, 10:13])

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 6: Anpassen eines 1PL-G Modells
#


## Das 1PL-G Modell
tammodel <- "
  LAVAAN MODEL:
  F =~ 1*I1__I50
  F ~~ F
  # Rateparameter für MC4 Items
  I1__I10 ?= gMC4*g1
  # Rateparameter für MC3 Items
  I11__I20 + I31__I40 ?= gMC3*g1
  "
mod.1PL_G <- tamaan(tammodel, resp, pweights = w,
                    control = list(Msteps = 15))

# ------------------------------------------------------------
# Abschnitt 6.3.5, Listing 7: Ausgabe geschätzter Rateparameter
#                             für MC3 und MC4 Items
#

mod.1PL_G$item[c(10,11), c(1,4,5)]


## ------------------------------------------------------------
## Abschnitt 6.3.6 Bookleteffekte
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 6.3.6, Listing 1: Anpassen eines Bookletmodells
#

mod.1PL_Book <- tam.mml.mfr(resp, facets = cbind(th = dat$th),
                 formulaA= ~ item + item:step + th, pweights = w)

# ------------------------------------------------------------
# Abschnitt 6.3.6, Listing 2: Ausgabe der Bookleteffekte der einzelnen
#                             Testhefte
#

rbind((tmp <- mod.1PL_Book$xsi[paste0("thER0", 1:5),]),
      thER06 = - c(sum(tmp[,1]), NA))


## ------------------------------------------------------------
## Abschnitt 6.3.7 Personenfähigkeitsschätzer
## ------------------------------------------------------------

# ------------------------------------------------------------
```

```
# Abschnitt 6.3.7, Listing 1: WLEs
#

WLE.1PL <- as.data.frame(tam.wle(mod.1PL))
round(head(WLE.1PL, 2), 4)

# ------------------------------------------------------------
# Abschnitt 6.3.7, Listing 2: WLE Reliabilität
#

WLErel(WLE.1PL$theta, WLE.1PL$error, w)

# ------------------------------------------------------------
# Abschnitt 6.3.7, Listing 3: EAPs
#

round(head(mod.1PL$person, 2), 4)

# ------------------------------------------------------------
# Abschnitt 6.3.7, Listing 4: EAP Reliabilität
#

EAPrel(mod.1PL$person$EAP, mod.1PL$person$SD.EAP, w)

# ------------------------------------------------------------
# Abschnitt 6.3.7, Listing 4a: Ergänzung zum Buch
# Alternative Berechnung der EAP-Reliabilität
#

1 - weighted.mean(mod.1PL$person$SD.EAP^2, w)/mod.1PL$variance


# ------------------------------------------------------------
# Abschnitt 6.3.7, Listing 5: PVs
#

PV.1PL <- tam.pv(mod.1PL)$pv
round(head(PV.1PL, 2), 4)

# ------------------------------------------------------------
# Abschnitt 6.3.7, Listing 6: Statistische Kennwerte der einzelnen
#                             Personenfähigkeitsschätzer
#

cbind(WLEs = c(M = weighted.mean(WLE.1PL$theta, w),
               SD = weighted_sd(WLE.1PL$theta, w)),
      EAPs = c(M = weighted.mean(mod.1PL$person$EAP, w),
               SD = weighted_sd(mod.1PL$person$EAP, w)),
      PVs = c(M = mean(apply(PV.1PL[, -1], 2, weighted.mean, w)),
              SD=mean(apply(PV.1PL[, -1], 2, weighted_sd, w))))


## ------------------------------------------------------------
```

```
## Abschnitt 6.3.8 Mehrdimensionale Modelle
## ------------------------------------------------------------

# Achtung: Algorithmen benötigen einige Zeit
# Zur schnelleren Konvergenz werden nur Daten aus Testheft 1 verwendet

# ------------------------------------------------------------
# Abschnitt 6.3.8, Listing 1: Verteilung der Items auf Foki
#

table(paste("Fokus", ue$focus[ue$Item %in% colnames(datTH1)]))
table(paste("Fokus", ueTH1$focus))

# ------------------------------------------------------------
# Abschnitt 6.3.8, Listing 2: Spezifizierung der Q-Matrix und
#                             Anpassung des Modells
#                             Achtung: Schätzung benötigt > 300 Iterationen
#

Q <- array(0, c(25, 5), list(items[items %in% colnames(datTH1)]))
for(i in 1:25) Q[i, ueTH1$focus[i] + 1] <- 1
mod.1PL_multi <- tam(resp = respTH1, pweights = wTH1,
                     Q = Q, control = list(snodes = 1500))

# ------------------------------------------------------------
# Abschnitt 6.3.8, Listing 3: Anpassen eines Bifaktormodells
#                             Achtung: Schätzung benötigt > 350 Iterationen
#

mod.1PL_bi <- tam.fa(respTH1, irtmodel = "bifactor1",
                dims = ueTH1$format, pweights = wTH1,
                control = list(snodes = 1500))

# ------------------------------------------------------------
# Abschnitt 6.3.8, Listing 4: Darstellung der Varianzen des
#                             Hauptfaktors und der Störfaktoren
#

nams <- c("I26", "I45", "I12", "I1", "I41")
dfr <- data.frame(mod.1PL_bi$B.stand[nams,],
                  row.names=ueTH1[nams, "format"])
dfr

# ------------------------------------------------------------
# Abschnitt 6.3.8, Listing 5: Darstellung der Reliabilitätsschätzer
#

mod.1PL_bi$meas


## ------------------------------------------------------------
## Abschnitt 6.3.9 Modellpassung
## ------------------------------------------------------------
```

```
# ------------------------------------------------------------
# Abschnitt 6.3.9, Listing 1: Berechnung und Darstellungen von
#                             Itemfitstatistiken
#

itemfit <- tam.fit(mod.1PL)
summary(itemfit)

# ------------------------------------------------------------
# Abschnitt 6.3.9, Listing 2: Berechnung und Darstellungen von
#                             Modellfitstatistiken
#

modfit <- tam.modelfit(mod.1PL)
modfit$fitstat

# ------------------------------------------------------------
# Abschnitt 6.3.9, Listing 3: LRT für Modelltestung
#

anova(mod.1PL, mod.GPCM)


## ------------------------------------------------------------
## Abschnitt 6.4.1 Simultane Kalibrierung
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 6.4.1, Listing 1: Daten vorbereiten
#

vars <- c("idstud", "wgtstud", "th")
# Daten der Hauptstudie
tmp1 <- cbind("Hauptstudie" = 1, dat[,c(vars, items)])
# Daten der Normierungsstudie
n.items <- grep("I|J",names(normdat),value=T)
tmp2 <- cbind("Hauptstudie" = 0, normdat[, c(vars, n.items)])
# Schülergewichte der Normierungsstudie sind konstant 1
# Datensätze zusammenfügen
dat.g <- rbind.fill(tmp1,tmp2)
all.items <- grep("I|J",names(dat.g),value=T)

# ------------------------------------------------------------
# Abschnitt 6.4.1, Listing 2: Simultane Kalibrierung
#                             Achtung: Schätzung benötigt > 450 Iterationen
#

# 2-Gruppenmodell
linkmod1 <-  tam.mml(resp=dat.g[, all.items], pid=dat.g[, 2],
              group = dat.g$Hauptstudie, pweights=dat.g$wgtstud)
summary(linkmod1)
```

```
# ------------------------------------------------------------
# Abschnitt 6.4.1, Listing 2a: Ergänzung zum Buch
# Berechnung von Verteilungsparametern
#

set.seed(20160828)

# PVs
PV_linkmod1 <- tam.pv(linkmod1, nplausible = 20)

# Personendatensatz
dfr_linkmod1 <- linkmod1$person
dfr_linkmod1 <- merge( x = dfr_linkmod1, y = PV_linkmod1$pv, by = "pid" , all=T)
dfr_linkmod1 <- dfr_linkmod1[ order(dfr_linkmod1$case) , ]

# Leistungsskala transformieren
vars.pv <- grep("PV",names(dfr_linkmod1),value=T)
# Mittlere Fähigkeit der Normierungsgruppe
p0 <- which(dat.g$Hauptstudie == 0)
M_PV <- mean(apply(dfr_linkmod1[p0,vars.pv],2,Hmisc::wtd.mean,
                   weights = dfr_linkmod1[p0,"pweight"]))
SD_PV <- mean(sqrt(apply(dfr_linkmod1[p0,vars.pv],2,Hmisc::wtd.var,
                         weights = dfr_linkmod1[p0,"pweight"])))
# Tranformationsparameter
a <- 100/SD_PV; b <- 500 - a*M_PV

# Verteilungsparameter der Hauptstudie
p1 <- which(dat.g$Hauptstudie == 1)
M1_PV <- mean(apply(dfr_linkmod1[p1,vars.pv],2,Hmisc::wtd.mean,
                    weights = dfr_linkmod1[p1,"pweight"]))
SD1_PV <- mean(sqrt(apply(dfr_linkmod1[p1,vars.pv],2,Hmisc::wtd.var,
                          weights = dfr_linkmod1[p1,"pweight"])))
TM_PV <- M1_PV*a + b; TSD_PV <- SD1_PV*a

# Ergebnisse
trafo_linkmod1 <- data.frame(M_Norm = 500, SD_Norm = 100, a = a, b = b,
                             M = TM_PV, SD = TSD_PV)


## ------------------------------------------------------------
## Abschnitt 6.4.2 Separate Kalibrierung mit fixiertem
##                 Itemparameter
## ------------------------------------------------------------


# Vorgehensweise 1:
# Daten der Normierungsstudie frei kalibrieren und skalieren
# Skalierung der Hauptstudie-Daten mit fixiertem Itemparameter

# ------------------------------------------------------------
# Abschnitt 6.4.2, Listing 1: Daten der Normierungsstudie frei
#                             kalibrieren und skalieren
#
```

```
normmod <- tam.mml(resp = normdat[, n.items],
                   pid = normdat[, "idstud"])

# ------------------------------------------------------------
# Abschnitt 6.4.2, Listing 1a: Ergänzung zum Buch
# Berechnung von Verteilungsparametern
#

summary(normmod)

set.seed(20160828)

# Personenfähigkeitsschätzer
PV_normmod <- tam.pv(normmod, nplausible = 20)
# In Personendatensatz kombinieren
dfr_normmod <- normmod$person
dfr_normmod <- merge( x = dfr_normmod, y = PV_normmod$pv, by = "pid" , all=T)
dfr_normmod <- dfr_normmod[ order(dfr_normmod$case) , ]

M_norm <- mean(apply(dfr_normmod[,vars.pv],2,Hmisc::wtd.mean,
                     weights = dfr_normmod[,"pweight"]))
SD_norm <- mean(sqrt(apply(dfr_normmod[,vars.pv],2,Hmisc::wtd.var,
                           weights = dfr_normmod[,"pweight"])))
# Tranformationsparameter
a_norm <- 100/SD_norm; b_norm <- 500 - a_norm*M_norm

TM_norm <- M_norm * a_norm + b_norm
TSD_norm <- SD_norm * a_norm

# ------------------------------------------------------------
# Abschnitt 6.4.2, Listing 2: Parameter aus Normierungsstudie
#                             für die Skalierung der Haupt-
#                             studie bei deren Skalierung
#                             fixieren
#

# Itemschwierigkeit aus der Normierungsstudie
norm.xsi <- normmod$xsi.fixed.estimated
# Hauptstudie: xsi-Matrix aus mod.1PL
xsi.fixed <- mod.1PL$xsi.fixed.estimated
# nur Parameter von Items in Hauptstudie
norm.xsi <- norm.xsi[
  rownames(norm.xsi) %in% rownames(xsi.fixed), ]
# Setzen der Parameter in richtiger Reihenfolge
xsi.fixed <- cbind(match(rownames(norm.xsi),
                         rownames(xsi.fixed)), norm.xsi[, 2])
# Skalierung der Hauptstudie-Daten mit fixierten Itemparameter
mainmod.fixed <- tam.mml(resp = resp, xsi.fixed = xsi.fixed,
                         pid = dat$MB_idstud, pweights = w)

# ------------------------------------------------------------
# Abschnitt 6.4.2, Listing 2a: Ergänzung zum Buch
```

```
# Berechnung von Verteilungsparametern
#

summary(mainmod.fixed)

set.seed(20160828)

# Personenfähigkeitsschätzer
WLE_mainmod.fixed <- tam.wle(mainmod.fixed)
PV_mainmod.fixed <- tam.pv(mainmod.fixed, nplausible = 20)
# In Personendatensatz kombinieren
dfr_mainmod.fixed <- mainmod.fixed$person
dfr_mainmod.fixed <- merge( x = dfr_mainmod.fixed, y = WLE_mainmod.fixed, by = "pid" , all=T)
dfr_mainmod.fixed <- merge( x = dfr_mainmod.fixed, y = PV_mainmod.fixed$pv, by = "pid" , all=T)
dfr_mainmod.fixed <- dfr_mainmod.fixed[ order(dfr_mainmod.fixed$case) , ]

M_main <- mean(apply(dfr_mainmod.fixed[,vars.pv],2,Hmisc::wtd.mean,
                     weights = dfr_mainmod.fixed[,"pweight"]))
SD_main <- mean(sqrt(apply(dfr_mainmod.fixed[,vars.pv],2,Hmisc::wtd.var,
                           weights = dfr_mainmod.fixed[,"pweight"])))

TM_main <- M_main * a_norm + b_norm
TSD_main <- SD_main * a_norm

trafo.fixed1 <- data.frame(M_norm = M_norm, SD_norm = SD_norm,
                           a = a_norm, b = b_norm,
                           TM_norm = TM_norm, TSD_norm = TSD_norm,
                           M_PV = M_main, SD_PV = SD_main,
                           M_TPV = TM_main, SD_TPV = TSD_main)

# Vorgehensweise 2:
# Daten der Hauptstudie frei kalibrieren und skalieren
# Skalierung der Hauptstudie-Daten mit fixierten Itemparameter

# ------------------------------------------------------------
# Abschnitt 6.4.2, Listing 2b: Ergänzung zum Buch
# Analoges Vorgehen mit fixierten Parametern aus der
# Hauptstudie für die Skalierung der Normierungsstudie
#

# Daten der Hauptstudie kalibrieren und skalieren
mainmod <- tam.mml(resp=dat[, items], irtmodel="1PL",
                   pid=dat$MB_idstud, pweights=dat[,"wgtstud"])
summary(mainmod)

set.seed(20160828)

# Personenfähigkeitsschätzer
WLE_mainmod <- tam.wle(mainmod)
PV_mainmod <- tam.pv(mainmod, nplausible = 20)
# In Personendatensatz kombinieren
dfr_mainmod <- mainmod$person
dfr_mainmod <- merge( x = dfr_mainmod, y = WLE_mainmod, by = "pid" , all=T)
```

```
dfr_mainmod <- merge( x = dfr_mainmod, y = PV_mainmod$pv, by = "pid" , all=T)
dfr_mainmod <- dfr_mainmod[order(dfr_mainmod$case),]

M_main <- mean(apply(dfr_mainmod[,vars.pv],2,Hmisc::wtd.mean,
                     weights = dfr_mainmod[,"pweight"]))
SD_main <- mean(sqrt(apply(dfr_mainmod[,vars.pv],2,Hmisc::wtd.var,
                           weights = dfr_mainmod[,"pweight"])))


# Itemschwierigkeit aus der Hauptstudie
main.xsi <- mod.1PL$xsi.fixed.estimated
# Hauptstudie: xsi-Matrix aus normmod
xsi.fixed <- normmod$xsi.fixed.estimated
# nur Parameter von Items in Hauptstudie
main.xsi <- main.xsi[
  rownames(main.xsi) %in% rownames(xsi.fixed), ]
# Setzen der Parameter in richtiger Reihenfolge
xsi.fixed <- cbind(match(rownames(main.xsi),
                         rownames(xsi.fixed)), main.xsi[, 2])

# Skalierung der Hauptstudie-Daten mit fixiertem Itemparameter
normmod.fixed <- tam.mml(resp=normdat[, n.items], irtmodel="1PL",
                         xsi.fixed = xsi.fixed,
                         pid=normdat$MB_idstud, pweights=normdat[,"wgtstud"])
summary(normmod.fixed)

set.seed(20160828)

# Personenfähigkeitsschätzer
PV_normmod.fixed <- tam.pv(normmod.fixed, nplausible = 20)
dfr_normmod.fixed <- normmod.fixed$person
dfr_normmod.fixed <- merge( x = dfr_normmod.fixed, y = PV_normmod.fixed$pv, by = "pid" , all=T)
dfr_normmod.fixed <- dfr_normmod.fixed[ order(dfr_normmod.fixed$case) , ]

M_norm <- mean(apply(dfr_normmod.fixed[,vars.pv],2,Hmisc::wtd.mean,
                     weights = dfr_normmod.fixed[,"pweight"]))
SD_norm <- mean(sqrt(apply(dfr_normmod.fixed[,vars.pv],2,Hmisc::wtd.var,
                           weights = dfr_normmod.fixed[,"pweight"])))

# Tranformationsparameter
a_norm <- 100/SD_norm; b_norm <- 500 - a_norm*M_norm

TM_norm <- M_norm * a_norm + b_norm
TSD_norm <- SD_norm * a_norm

TM_main <- M_main * a_norm + b_norm
TSD_main <- SD_main * a_norm

trafo.fixed2 <- data.frame(M_PV = M_main, SD_PV = SD_main,
                           M_Norm.fixed = M_norm, SD_Norm.fixed = SD_norm,
                           a = a_norm, b = b_norm,
                           TM_norm = TM_norm, TSD_norm = TSD_norm,
                           M_TPV = TM_main, SD_TPV = TSD_main)
```

```
## ------------------------------------------------------------
## Abschnitt 6.4.3 Separate Kalibrierung mit Linking durch
##                 Transformationsfunktion
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 6.4.3, Listing 1: equating.rasch()
#

# Freigeschätzte Itemparameter der Normierung- und Hauptstudie
norm.pars <- normmod$item[,c("item","xsi.item")]
main.pars <- mainmod$item[,c("item","xsi.item")]
# Linking mit equating.rasch
mod.equate <- equating.rasch(x = norm.pars, y = main.pars)
mod.equate$B.est
#   Mean.Mean    Haebara Stocking.Lord
#  -0.1798861 -0.1788159    -0.1771145
head(mod.equate$anchor,2)

# ------------------------------------------------------------
# Abschnitt 6.4.3, Listing 1a: Ergänzung zum Buch
# Berechnung Linkingfehler
#
linkitems <- intersect(n.items, items)

head(mod.equate$transf.par,2)
mod.equate$descriptives

# Linkingfehler: Jackknife unit ist Item
pars <- data.frame(unit = linkitems,
                   study1 = normmod$item$xsi.item[match(linkitems, normmod$item$item)],
                   study2 = mainmod$item$xsi.item[match(linkitems, mainmod$item$item)],
                   item = linkitems)
# pars <- as.matrix(pars)
mod.equate.jk <- equating.rasch.jackknife(pars,se.linkerror = T)
mod.equate.jk$descriptives

# ------------------------------------------------------------
# Abschnitt 6.4.3, Listing 2: Linking nach Haberman
#

# Itemparameter der Normierungsstudie
M1 <- mean( apply(dfr_normmod[,vars.pv], 2, mean ) )
SD1 <- mean( apply(dfr_normmod[,vars.pv], 2, sd ) )
a1 <- 1/SD1; b1 <- 0-a1*M1
A <- normmod$item$B.Cat1.Dim1/a1
B <- (normmod$item$xsi.item + b1/a1)
# Itemparameter der Normierungsstudie fuer haberman.linking
tab.norm <- data.frame(Studie = "1_Normierung",
                       item = normmod$item$item,
                       a = A, b = B/A)
```

```
# Itemparameter der Hauptstudie
A <- mainmod$item$B.Cat1.Dim1
B <- mainmod$item$xsi.item
tab.main <- data.frame(Studie = "2_Hauptstudie",
                       item = mainmod$item$item,
                       a = A, b = B/A)
# Itemparameter aller Studien
itempars <- rbind(tab.norm, tab.main)
# Personenparameter
personpars <- list(PV_normmod$pv*a1+b1, PV_mainmod$pv)
# Linking nach Habermans Methode
linkhab <- linking.haberman(itempars = itempars,
                            personpars = personpars)

# ------------------------------------------------------------
# Abschnitt 6.4.3, Listing 2a: Ergänzung zum Buch
# Ergebnisdarstellung, Transformation und Berechnung
# von Verteilungsparametern
#

# Ergebnisse
# Transformationsparameter der Itemparameter
linkhab$transf.itempars
# Transformationsparameter der Personenparameter
linkhab$transf.personpars

# Itemparameter
dfr.items <- data.frame(linkhab$joint.itempars,
                        linkhab$b.orig, linkhab$b.trans)
names(dfr.items)[-1] <- c("joint_a","joint_b",
                          "orig_b_norm","orig_b_main",
                          "trans_b_norm","trans_b_main")
head(round2(dfr.items[,-1],2),2)

# Transformierte Personenparameter der Hauptstudie
dfr_main_transpv <- linkhab$personpars[[2]]
names(dfr_main_transpv)[-1] <- paste0("linkhab_",vars.pv)
dfr_main_transpv <- cbind(dfr_mainmod,dfr_main_transpv[,-1])
round2(head(dfr_main_transpv[,c("PV1.Dim1","linkhab_PV1.Dim1","PV2.Dim1","linkhab_PV2.Dim1")],2),2)

# Aufgeklärte und Fehlvarianz des Linkings
linkhab$es.invariance

# Transformationsparameter der Normierungsstudie auf Skala 500,100
# trafo.fixed1
a <- 100/mean( apply(dfr_normmod[,vars.pv]*a1+b1, 2, sd ) )
b <- 500 - a*mean( apply(dfr_normmod[,vars.pv]*a1+b1, 2, mean ) )

# trafo.fixed2
M_PV <- mean( apply(linkhab$personpars[[2]][vars.pv], 2,
                    Hmisc::wtd.mean, weights = dfr_mainmod$pweight ) )
SD_PV <- mean( sqrt(apply(linkhab$personpars[[2]][vars.pv], 2,
                          Hmisc::wtd.var, weights = dfr_mainmod$pweight )) )
```

```
M_TPV <- M_PV*a + b
SD_TPV <- SD_PV * a

trafo.linkhab <- data.frame(trafo.fixed1[,1:2],
                            a1 = a1, b1 = b1,
                            M_norm_trans = 0,
                            SD_norm_trans = 1,
                            a = 100, b = 500,
                            trafo.fixed2[,1:2],
                            linkhab_M_PV = M_PV,
                            linkhab_SD_PV = SD_PV,
                            linkhab_M_TPV = M_TPV,
                            linkhab_SD_TPV = SD_TPV)


## ------------------------------------------------------------
## Abschnitt 6.4.4 Ergebnisse im Vergleich und Standardfehler
##                 des Linkings
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 6.4.4, Listing 3a: Ergänzung zum Buch
# Berechnung von Standardfehlern Ergebnisvergleiche
#

# Gemeinsame Skalierung mit fixiertem Itemparameter aus Hauptstudie
# Standardfehler bzgl. Itemstichprobenfehler

# Matrix für fixerte Itemparameter vorbereiten
xsi.fixed <- normmod.fixed$xsi.fixed.estimated
npar <- length(xsi.fixed[,"xsi"])
mat.xsi.fixed <- cbind(index=1:npar,par = dimnames(xsi.fixed)[[1]])
sequence <- match(mat.xsi.fixed[,"par"],dimnames(main.xsi)[[1]])
mat.xsi.fixed <- cbind(index=as.numeric(mat.xsi.fixed[,1]),
                       par = mat.xsi.fixed[,2],
                       xsi.fixed = as.numeric(main.xsi[sequence,"xsi"]))
# Nicht fixierte Itemparameter löschen
del <- which(is.na(mat.xsi.fixed[,"xsi.fixed"]))
mat.xsi.fixed <- mat.xsi.fixed[-del,]
head(mat.xsi.fixed,3)

dfr <- data.frame(elim = "none",growth=trafo.fixed2$M_TPV-500)
# Jedes Mal ein Ankeritem weniger
# Schleife über alle Ankeritems

set.seed(20160828)

for(ii in linkitems){
  # ii <- linkitems[1]
  del <- grep(paste0(ii,"_"), mat.xsi.fixed[,2])
  tmp <- mat.xsi.fixed[-del,c(1,3)]
  tmp <- data.frame(index = as.numeric(tmp[,1]),xsi.fixed = as.numeric(tmp[,2]))
```

```
  # Skalierung der Hauptstudie-Daten mit fixiertem Itemparameter
  normmod.tmp <- tam.mml(resp=normdat[, n.items], irtmodel="1PL",
                         xsi.fixed = tmp,
                         pid=normdat$MB_idstud, pweights=normdat[,"wgtstud"])

  # Personenfähigkeitsschätzer
  # WLE_normmod.tmp <- tam.wle(normmod.tmp)
  PV_normmod.tmp <- tam.pv(normmod.tmp, nplausible = 20)
  # In Personendatensatz kombinieren

  M_norm.tmp <- mean(apply(PV_normmod.tmp$pv[,vars.pv],2,mean))
  SD_norm.tmp <- mean(apply(PV_normmod.tmp$pv[,vars.pv],2,sd))

  # Tranformationsparameter
  a_norm.tmp <- 100/SD_norm.tmp
  b_norm.tmp <- 500 - a_norm.tmp*M_norm.tmp

  TM_main.tmp <- M_main * a_norm.tmp + b_norm.tmp
  dfr.tmp <- data.frame(elim = ii,growth=TM_main.tmp-500)
  dfr <- rbind(dfr,dfr.tmp)

}

dfr$diff2 <- (dfr$growth-dfr$growth[1])^2
sum <- sum(dfr$diff2)
Var <- sum*28/29
SE <- sqrt(Var)

quant <- 1.96
low <- trafo.fixed2$M_TPV - quant*SE
upp <- trafo.fixed2$M_TPV + quant*SE

dfr$SE <- SE; dfr$quant <- quant
dfr$low <- low; dfr$upp <- upp


## End(Not run)
```

Kapitel 7 *Kapitel 7: Statistische Analysen produktiver Kompetenzen*

Description

Das ist die Nutzerseite zum Kapitel 7, *Statistische Analysen produktiver Kompetenzen*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Details

Abschnitt 1: Beispieldatensätze:

Der zur Illustration verwendete Datensatz `prodRat` beinhaltet die beurteilten Schreibkompetenzen im Fach Englisch auf der 8. Schulstufe von 9836 Schüler/innen (`idstud`) die von insgesamt 41 Ratern (`rater`) beurteilt wurden. Die sechs Schreibaufgaben (`aufgabe`) wurden auf sechs Testhefte (`th`) aufgeteilt, wobei jede Aufgabe in genau zwei Testheften vorkommt.

Zur weiteren Analyse verwenden wir auch den Datensatz `prodPRat` mit sogenannten Pseudoratern.

Für die Analyse von Varianzkomponenten mittels Linear Mixed Effects (LME) Modellen verwenden wir den ursprünglichen Datensatz im Long Format (`prodRatL`).

Abschnitt 2: Beurteilerübereinstimmung:

Listing 1: Berechnen von Häufigkeitstabellen: Hier werden die Datensätze `prodRat` und `prodPRat` verwendet. Die R-Funktion `apply()` ermöglicht eine Anwendung einer beliebigen Funktion z.B. `prop.table()` über alle Zeilen (1) oder Spalten (2) eines `data.frame`.

```
> library(irr)
> data(datenKapitel07)
> prodRat <- datenKapitel07$prodRat
>
> # Items auswählen
> items <- c("TA", "CC", "GR", "VO")
> # Tabelle erzeugen
> tab <- apply(prodRat[, items], 2,
+ FUN=function(x){
+ prop.table(table(x))*100})
> print(tab, digits = 2)
>
> # Mittelwert der Ratings berechnen
> round(apply(prodRat[, items], 2, mean), 2)
```

Listing 2: Beurteilerübereinstimmung berechnen: Wir verwenden den Datensatz mit Pseudoratern `prodPRat`. Die Analysen werden mit dem Paket `irr` durchgeführt.

```
> prodRat <- datenKapitel07$prodRat
>
> items <- c("TA", "CC", "GR", "VO")
> dfr <- data.frame(items, agree = NA,
+ kappa = NA, wkappa = NA, korr = NA)
> for(i in 1:length(items)){
+ dat.i <- prodPRat[, grep(items[i], colnames(prodPRat))]
+ dfr[i, "agree"] <- agree(dat.i, tolerance = 1)["value"]
+ dfr[i, "kappa"] <- kappa2(dat.i)["value"]
+ dfr[i, "wkappa"] <- kappa2(dat.i, weight = "squared")["value"]
+ dfr[i, "korr"] <- cor(dat.i[,1], dat.i[,2])
+ dfr[i, "icc"] <- icc(dat.i, model = "twoway")["value"]
+ }
```

```
> print(dfr, digits = 3)
```

Abschnitt 3: Skalierungsmodelle:

Listing 1: Skalierungsmodell mit TAM:
Der Funktion `tam.mm.mfr()` muss ein `data.frame` für die Facetten übergeben werden. Zusätzlich können Einstellungen in einer Liste für das Argument `control = list()` übergeben werden. Hier verwenden wir die Einstellung `xsi.start0 = 1`, was dazu führt, dass alle Startwerte auf 0 gesetzt werden. Mit `fac.oldxsi = 0.1` setzen wir das Gewicht der Parameterwerte aus der vorigen Iteration etwas über 0. Damit kann der Algorithmus stabilisiert und Konvergenzprobleme (deviance increase) verhindert werden. Wir definieren noch `increment.factor = 1.05` etwas über dem default-Wert von 1 um mögliche Konvergenzprobleme abzufangen. Dieser Wert definiert das Ausmaß der Abnahme des maximalen Zuwachs der Parameter pro Iteration (s. TAM-Hilfe).
Die Personenparameter werden mit der Funktion `tam.wle()` geschätzt.
Gibt man in der Funktion `summary()` das Argument `file` an, so wird der Output direkt in ein Textfile geschrieben.

```
> set.seed(1234)
> library(TAM)
>
> prodRat <- datenKapitel07$prodRat
>
> # Rater-Facette definieren
> facets <- prodRat[, "rater", drop = FALSE]
>
> # Response Daten definieren
> vars <- c("TA", "CC", "GR", "VO")
> resp <- prodRat[, vars]
>
> # Personen-ID definieren
> pid <- prodRat$idstud
>
> # Formel für Modell
> formulaA <- ~item*step+item*rater
>
> # Modell berechnen
> mod <- tam.mml.mfr(resp = resp, facets = facets, formulaA = formulaA,
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
```

Listing 1b (Ergänzung zum Buch): Skalierungsmodell mit TAM:
Hier werden alle im Buch besprochenen Modelle berechnet und anschließend ein Modellvergleich durchgeführt.

```
> f1 <- ~item * rater * step
> mod1 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f1,
```

```
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
> f2 <- ~item*step+item*rater
> mod2 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f2,
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
> f3 <- ~item * step + rater
> mod3 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f3,
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
> f4 <- ~item + step + rater
> mod4 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f4,
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
> mod4$xsi.facets
> IRT.compareModels(mod1, mod2, mod3, mod4)
```

Listing 1c (Ergänzung zum Buch): Wright-Map:
Mit dem Paket `WrightMap` können die Ergebnisse für die einzelnen Facetten dargestellt werden. Wir machen dies für Items und Rater.

```
> library(WrightMap)
>
> item.labs <- vars
> rater.labs <- unique(prodRat$rater)
> item.labs <- c(item.labs, rep(NA, length(rater.labs) -
+ length(item.labs)))
>
> pars <- mod$xsi.facets$xsi
> facet <- mod$xsi.facets$facet
> item.par <- pars[facet == "item"]
> rater.par <- pars[facet == "rater"]
> item_rat <- pars[facet == "item:rater"]
> len <- length(item_rat)
> item.long <- c(item.par, rep(NA, len - length(item.par)))
> rater.long <- c(rater.par, rep(NA, len - length(rater.par)))
>
> wrightMap(persons.mod$theta, rbind(item.long, rater.long),
+ label.items = c("Items", "Rater"),
+ thr.lab.text = rbind(item.labs, rater.labs),
+ axis.items = "", min.l=-3, max.l=3,
+ axis.persons = "Personen")
```

Listing 2: Fit-Indices berechnen:

Die Fit-Indices werden mit der Funktion `msq.itemfitWLE` für die Raterparameter und Itemparameter gesondert berechnet. Der Funktion muss ein Vektor mit Parameterbezeichnungen übergeben werden so wie sie im Modell-Objekt vorkommen. Im Paket `TAM` gibt es noch die Funktion `tam.fit()`, diese basiert auf einer Simulation der individuellen Posterior-Verteilung. Die Funktion `msq.itemfitWLE` wertet dagegen die individuelle Posterior-Verteilung direkt aus (s. `TAM`-Hilfe für weitere Beispiele) und führt keine Simulation durch.

```
> # Infit/Outfit berechnen
> pseudo_items <- colnames(mod$resp)
> pss <- strsplit(pseudo_items , split="-")
> item_parm <- unlist(lapply(pss, FUN = function(ll){ll[1]}))
> rater_parm <- unlist(lapply(pss, FUN = function(ll){ll[2]}))
>
> # Fit Items
> res.items <- msq.itemfitWLE(mod, item_parm)
> summary(res.items)
>
> # Fit Rater
> res.rater <- msq.itemfitWLE(mod, rater_parm)
> summary(res.rater)
```

Listing 2a (Ergänzung zum Buch): Abbildung Fit-Indices:

```
> # Abbildung: Histogramm, Rohscores
>
> par(mfcol=c(1,2))
>
> hist(persons.mod$theta, col="grey", breaks=40,
+ main = "",
+ xlab = "Theta (logits)",
+ ylab = "Häufigkeit")
> with(persons.mod, scatter.smooth(raw.score, theta,
+ pch = 1, cex = .6, xlab = "Roscores",
+ ylab = "Theta (logits)",
+ lpars = list(col = "darkgrey", lwd = 2, lty = 1)))
>
> # Abbildung: Fit-Statistik
> par(mfcol=c(1,2))
> fitdat <- res.rater$fit_data
> fitdat$var <- factor(substr(fitdat$item, 1, 2))
> boxplot(Outfit~var, data=fitdat,
+ ylim=c(0,2), main="Outfit")
> boxplot(Infit~var, data=fitdat,
+ ylim=c(0,2), main="Infit")
```

Listing 2b (Ergänzung zum Buch): Korrelationen:
Pearson und Spearman Korrelationskoeffizient wird zwischen Rohscores und Theta berechnet.

```
> korr <- c(with(persons.mod, cor(raw.score, theta,
+ method = "pearson")),
+ with(persons.mod, cor(raw.score, theta,
+ method = "spearman")))
> print(korr)
```

Listing 3: Q3-Statistik berechnen:
Die Q3-Statistik für lokale stochastische Unabhängigkeit wird mit der Funktion `tam.modelfit()` berechnet. Der Output enthält eine Vielzahl an Fit-Statistiken, für weitere Details sei hier auf die TAM-Hilfeseite verwiesen. Die adjustierte aQ3-Statistik berechnet sich aus den Q3-Werten abzüglich des Gesamtmittelwerts von allen Q3-Werten.
Mit `tam.modelfit()` werden Fit-Statistiken für alle Rater x Item Kombinationen berechnet. Diese werden im Code unten anschließend aggregiert um eine Übersicht zu erhalten. Dazu werden zuerst nur Paare gleicher Rater ausgewählt, somit wird die aggregierte Q3-Statistik nur Rater-spezifisch berechnet. Das Objekt `rater.q3` beinhaltet eine Zeile pro Rater x Item Kombination. Kombinationen ergeben sich nur für einen Rater, nicht zwischen unterschiedlichen Ratern.
Anschließend kann man mit `aggregate()` separat über Rater und Kombinationen mitteln und diese als Dotplot darstellen (Paket `lattice`).

```
> # Q3 Statistik
> mfit.q3 <- tam.modelfit(mod)
> rater.pairs <- mfit.q3$stat.itempair
>
> # Nur Paare gleicher Rater wählen
> unique.rater <- which(substr(rater.pairs$item1, 4,12) ==
+ substr(rater.pairs$item2, 4,12))
> rater.q3 <- rater.pairs[unique.rater, ]
>
> # Spalten einfügen: Rater, Kombinationen
> rater.q3$rater <- substr(rater.q3$item1, 4, 12)
> rater.q3 <- rater.q3[order(rater.q3$rater),]
> rater.q3$kombi <- as.factor(paste(substr(rater.q3$item1, 1, 2),
+ substr(rater.q3$item2, 1, 2), sep="_"))
>
> # Statistiken aggregieren: Rater, Kombinationen
> dfr.raterQ3 <- aggregate(rater.q3$aQ3, by = list(rater.q3$rater), mean)
> colnames(dfr.raterQ3) <- c("Rater", "Q3")
> dfr.itemsQ3 <- aggregate(rater.q3$aQ3, by = list(rater.q3$kombi), mean)
> colnames(dfr.itemsQ3) <- c("Items", "Q3")
> dfr.itemsQ3
```

Listing 3 (Ergänzung zum Buch): Lattice Dotplot:

```
> library(lattice)
> library(grid)
>
> # Lattice Dotplot
```

```
> mean.values <- aggregate(rater.q3$aQ3, list(rater.q3$kombi), mean)[["x"]]
> dotplot(aQ3~kombi, data=rater.q3, main="Q3-Statistik", ylab="Q3 (adjustiert)",
+ col="darkgrey",
+ panel = function(x,...){
+ panel.dotplot(x,...)
+ panel.abline(h = 0, col.line = "grey", lty=3)
+ grid.points(1:6, mean.values, pch=17)
+ })
```

Abschnitt 4: Generalisierbarkeitstheorie:

Listing 1: Varianzkomponenten mit lme4 berechnen:
Mit der Funktion `lmer()` aus dem Paket `lme4` schätzen wir die Varianzkomponenten. In der Formel definieren wir dabei die Facetten als random effects.

```
> library(lme4)
>
> prodRatL <- datenKapitel07$prodRatL
>
> # Formel definieren
> formula1 <- response ~ (1|idstud) + (1|item) + (1|rater) +
+ (1|rater:item) + (1|idstud:rater) +
+ (1|idstud:item)
> # Modell mit Interaktionen
> mod.vk <- lmer(formula1, data=prodRatL)
>
> # Zusammenfassung der Ergebnisse
> summary(mod.vk)
```

Listing 1a (Ergänzung zum Buch): Summary-Funktion für Varianzkomponenten:
Wir generieren eine Funktion `summary.VarComp()`, die den Output des Modells `mod.vk` in einen ansprechenden `data.frame` schreibt. Hier werden auch die prozentualen Anteile der Varianzkomponenten berechnet.

```
> # Helper-Function um die Varianzkomponenten zu extrahieren
> summary.VarComp <- function(mod){
+ var.c <- VarCorr(mod)
+ var.c <- c(unlist(var.c) , attr(var.c , "sc")^2)
+ names(var.c)[length(var.c)] <- "Residual"
+ dfr1 <- data.frame(var.c)
+ colnames(dfr1) <- "Varianz"
+ dfr1 <- rbind(dfr1, colSums(dfr1))
+ rownames(dfr1)[nrow(dfr1)] <- "Total"
+ dfr1$prop.Varianz <- 100 * (dfr1$Varianz / dfr1$Varianz[nrow(dfr1)])
+ dfr1 <- round(dfr1,2)
+ return(dfr1)
+ }
> summary.VarComp(mod.vk)
```

Listing 2: Berechnung des G-Koeffizienten:
Den G-Koeffizienten berechnen wir nach der Formel im Buch.

```
> vk <- summary.VarComp(mod.vk)
> n.p <- length(unique(prodRatL$idstud)) # Anzahl Schüler
> n.i <- 4 # Anzahl Items
> n.r <- c(1,2,5) # Anzahl Rater
>
> # Varianzkomponenten extrahieren
> sig2.p <- vk["idstud", "Varianz"]
> sig2.i <- vk["item", "Varianz"]
> sig2.r <- vk["rater", "Varianz"]
> sig2.ri <- vk["rater:item", "Varianz"]
> sig2.pr <- vk["idstud:rater", "Varianz"]
> sig2.pi <- vk["idstud:item", "Varianz"]
> sig2.pir <- vk["Residual", "Varianz"]
>
> # Fehlervarianz berechnen
> sig2.delta <- sig2.pi/n.i + sig2.pr/n.r + sig2.pir/(n.i*n.r)
>
> # G-Koeffizient berechnen
> g.koeff <- sig2.p / (sig2.p + sig2.delta)
> print(data.frame(n.r, g.koeff), digits = 3)
```

Listing 2a (Ergänzung zum Buch): Phi-Koeffizient berechnen:

```
> sig2.D <- sig2.r/n.r + sig2.i/n.i + sig2.pi/n.i + sig2.pr/n.r +
+ sig2.ri/(n.i*n.r) + sig2.pir/(n.i*n.r)
> phi.koeff <- sig2.p / (sig2.p + sig2.D)
> print(data.frame(n.r, phi.koeff), digits = 3)
>
> # Konfidenzintervalle
> 1.96*sqrt(sig2.D)
```

Listing 2c (Ergänzung zum Buch): Variable Rateranzahl:
Für eine variable Rateranzahl (hier 1 bis 10 Rater) werden die G-Koeffizienten berechnet.

```
> n.i <- 4 # Anzahl Items
> dn.r <- seq(1,10) # 1 bis 10 mögliche Rater
> delta.i <- sig2.pi/n.i + sig2.pr/dn.r + sig2.pir/(n.i*dn.r)
> g.koeff <- sig2.p / (sig2.p + delta.i)
> names(g.koeff) <- paste("nR", dn.r, sep="_")
> print(g.koeff[1:4])
>
> plot(g.koeff, type = "b", pch = 19, lwd = 2, bty = "n",
+ main = "G-Koeffizient: Raters",
+ ylab = "G-Koeffizient",
```

```
+ xlab = "Anzahl Raters", xlim = c(0,10))
> abline(v=2, col="darkgrey")
```

Abschnitt 5: Strukturgleichungsmodelle:

In R setzen wir das Struktugleichungsmodell mit dem Paket `lavaan` um. Das Modell wird als Textvariable definiert, welche anschließend der Funktion `sem()` übergeben wird. Latente Variablen im Messmodell werden in `lavaan` mit der Form `latente Variable =~ manifeste Variable(n)` definiert, die Ladungen werden dabei auf den Wert 1 fixiert, was mittels der Multiplikation der Variable mit dem Wert 1 umgesetzt werden kann (z.B. `1*Variable`). Varianzen und Kovarianzen werden mit `Variable ~~ Variable` gebildet, wobei hier die Multiplikation mit einem Label einerseits den berechneten Parameter benennt, andererseits, bei mehrmaligem Auftreten des Labels, Parameterschätzungen von verschiedenen Variablen restringiert bzw. gleichstellt (z.B. wird für die Within-Varianz von `TA` über beide Rater nur ein Parameter geschätzt, nämlich `Vta_R12`). Die ICC wird für jede Dimension separat direkt im Modell spezifiziert, dies geschieht durch abgeleitete Variablen mit der Schreibweise `Variable := Berechnung`. Die Modellspezifikation und der Aufruf der Funktion `sem()` ist wie folgt definiert:

Listing 1 (mit Ergänzung zum Buch): SEM:

```
> library(lavaan)
>
> prodPRat <- datenKapitel07$prodPRat
>
> # SEM Modell definieren
> lv.mod <- \"
+ # Messmodell
+ TA =~ 1*TA_R1 + 1*TA_R2
+ CC =~ 1*CC_R1 + 1*CC_R2
+ GR =~ 1*GR_R1 + 1*GR_R2
+ VO =~ 1*VO_R1 + 1*VO_R2
+
+ # Varianz Between (Personen)
+ TA ~~ Vta * TA
+ CC ~~ Vcc * CC
+ GR ~~ Vgr * GR
+ VO ~~ Vvo * VO
+
+ # Varianz Within (Rater X Personen)
+ TA_R1 ~~ Vta_R12 * TA_R1
+ TA_R2 ~~ Vta_R12 * TA_R2
+ CC_R1 ~~ Vcc_R12 * CC_R1
+ CC_R2 ~~ Vcc_R12 * CC_R2
+ GR_R1 ~~ Vgr_R12 * GR_R1
+ GR_R2 ~~ Vgr_R12 * GR_R2
+ VO_R1 ~~ Vvo_R12 * VO_R1
+ VO_R2 ~~ Vvo_R12 * VO_R2
+
+ # Kovarianz Within
```

```
+ TA_R1 ~~ Kta_cc * CC_R1
+ TA_R2 ~~ Kta_cc * CC_R2
+ TA_R1 ~~ Kta_gr * GR_R1
+ TA_R2 ~~ Kta_gr * GR_R2
+ TA_R1 ~~ Kta_vo * VO_R1
+ TA_R2 ~~ Kta_vo * VO_R2
+ CC_R1 ~~ Kcc_gr * GR_R1
+ CC_R2 ~~ Kcc_gr * GR_R2
+ CC_R1 ~~ Kcc_vo * VO_R1
+ CC_R2 ~~ Kcc_vo * VO_R2
+ GR_R1 ~~ Kgr_vo * VO_R1
+ GR_R2 ~~ Kgr_vo * VO_R2
+
+ # ICC berechnen
+ icc_ta := Vta / (Vta + Vta_R12)
+ icc_cc := Vcc / (Vcc + Vcc_R12)
+ icc_gr := Vgr / (Vgr + Vgr_R12)
+ icc_vo := Vvo / (Vvo + Vvo_R12)
+ \"
> # Schätzung des Modells
> mod1 <- sem(lv.mod, data = prodPRat)
> summary(mod1, standardized = TRUE)
> # Inspektion der Ergebnisse
> show(mod1)
> fitted(mod1)
> inspect(mod1,"cov.lv")
> inspect(mod1, "free")
```

Listing 2: Kompakte Darstellung der Ergebnisse:

```
> parameterEstimates(mod1, ci = FALSE,
+ standardized = TRUE)
```

Listing 2a (Ergänzung zum Buch): Schreibe Ergebnisse in Latex-Tabelle:

```
> library(xtable)
>
> xtable(parameterEstimates(mod1, ci = FALSE,
+ standardized = TRUE), digits = 3)
```

Abschnitt 7: Übungen:

Übung 1: MFRM M3 und M4 umsetzen und Vergleichen: Wir setzen die Modelle separat in TAM um und lassen uns mit `summary()` die Ergebnisse anzeigen. Einen direkten Zugriff auf die geschätzen Parameter bekommt man mit `mod$xsi.facets`. Dabei sieht man, dass im Modell 4 keine generalized items gebildet werden, da hier kein Interaktionsterm vorkommt. Den Modellvergleich machen wir mit `IRT.compareModels(mod3, mod4)`. Modell 3 weist hier kleinere AIC-Werte auf und scheint etwas besser auf die Daten zu passen als Modell 4. Dies

zeigt auch der Likelihood-Ratio Test, demnach sich durch das Hinzufügen von Parametern die Modellpassung verbessert.

```
> library(TAM)
> prodRatEx <- datenKapitel07$prodRatEx
>
> # Rater-Facette definieren
> facets <- prodRatEx[, "rater", drop = FALSE]
>
> # Response Daten definieren
> vars <- c("TA", "CC", "GR", "VO")
> resp <- prodRatEx[, vars]
>
> # Personen-ID definieren
> pid <- prodRatEx$idstud
>
> # Modell 3
> f3 <- ~item * step + rater
> mod3 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f3,
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
> # Modell 4
> f4 <- ~item + step + rater
> mod4 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f4,
+ pid = pid, control=list(xsi.start0 = 1,
+ fac.oldxsi = 0.1,
+ increment.factor = 1.05))
> summary(mod3, file = "TAM_MFRM")
> summary(mod4, file = "TAM_MFRM")
>
>
> mod3$xsi.facets
> mod4$xsi.facets
>
> IRT.compareModels(mod3, mod4)
$IC
' Model loglike Deviance Npars Nobs AIC BIC AIC3 AICc CAIC'
1 mod3 -60795.38 121590.8 69 9748 121728.8 122224.5 121797.8 121729.8 122293.5
2 mod4 -61041.46 122082.9 51 9748 122184.9 122551.3 122235.9 122185.5 122602.3

$LRtest
' Model1 Model2 Chi2 df p'
1 mod4 mod3 492.1549 18 0

attr(,"class")
[1] "IRT.compareModels"
```

Übung 2: Varianzkomponentenmodell: Das Varianzkomponentenmodell setzen wir für die short prompts nach den Vorgaben im Buchkapitel um. Dabei verändern wir die Anzahl der möglichen Rater durch `n.r <- c(2,10,15)`. Der Phi-Koeffizient kann laut Gleichung 6.9 und 6.10 berechnet werden. Die Ergebnisse zeigen einen prozentuellen Anteil der Interaktion Person und Rater von ca. 15%, dieser scheint auf Halo-Effekte hinzuweisen.

```
> # Formel definieren
> formula1 <- response ~ (1|idstud) + (1|item) + (1|rater) +
+ (1|rater:item) + (1|idstud:rater) +
+ (1|idstud:item)
> # Modell mit Interaktionen
> mod.vk <- lmer(formula1, data=prodRatLEx)
>
> summary(mod.vk)
> print(vk <- summary.VarComp(mod.vk))
' Varianz prop.Varianz'
idstud:item 0.10 2.45
idstud:rater 0.64 15.21
idstud 2.88 67.94
rater:item 0.01 0.22
rater 0.19 4.39
item 0.00 0.02
Residual 0.41 9.78
Total 4.24 100.00
>
>
> # Verändern der Rateranzahl
> n.p <- length(unique(prodRatLEx$idstud)) # Anzahl Schüler
> n.i <- 4 # Anzahl Items
> n.r <- c(2,10,15) # Anzahl Rater
>
> # Varianzkomponenten extrahieren
> sig2.p <- vk["idstud", "Varianz"]
> sig2.i <- vk["item", "Varianz"]
> sig2.r <- vk["rater", "Varianz"]
> sig2.ri <- vk["rater:item", "Varianz"]
> sig2.pr <- vk["idstud:rater", "Varianz"]
> sig2.pi <- vk["idstud:item", "Varianz"]
> sig2.pir <- vk["Residual", "Varianz"]
>
> # Phi-Koeffizient berechnen
> sig2.D <- sig2.r/n.r + sig2.i/n.i + sig2.pi/n.i + sig2.pr/n.r +
+ sig2.ri/(n.i*n.r) + sig2.pir/(n.i*n.r)
> phi.koeff <- sig2.p / (sig2.p + sig2.D)
> print(data.frame(n.r, phi.koeff), digits = 3)
>
> # Konfidenzintervalle
> 1.96*sqrt(sig2.D)
```

Author(s)

Roman Freunberger, Alexander Robitzsch, Claudia Luger-Bazinger

References

Freunberger, R., Robitzsch, A. & Luger-Bazinger, C. (2016). Statistische Analysen produktiver Kompetenzen. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 225–258). Wien: facultas.

See Also

Zu datenKapitel07, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 6, Skalierung und Linking.
Zu Kapitel 8, Fehlende Daten und Plausible Values.
Zur Übersicht.
Zur Hilfeseite von TAM

Examples

```
## Not run:
library(irr)
library(TAM)
library(WrightMap)
library(lattice)
library(grid)
library(lme4)
library(lavaan)
library(xtable)

summary.VarComp <- function(mod){
  var.c <- VarCorr(mod)
  var.c <- c(unlist(var.c) , attr(var.c , "sc")^2)
  names(var.c)[length(var.c)] <- "Residual"
  dfr1 <- data.frame(var.c)
  colnames(dfr1) <- "Varianz"
  dfr1 <- rbind(dfr1, colSums(dfr1))
  rownames(dfr1)[nrow(dfr1)] <- "Total"
  dfr1$prop.Varianz <- 100 * (dfr1$Varianz / dfr1$Varianz[nrow(dfr1)])
  dfr1 <- round(dfr1,2)
  return(dfr1)
}

data(datenKapitel07)
prodRat <- datenKapitel07$prodRat
prodRatL <- datenKapitel07$prodRatL
prodPRat <- datenKapitel07$prodPRat

## ----------------------------------------------------------------
## Abschnitt 7.2: Beurteilerübereinstimmung
```

```
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 7.2, Listing 1: Berechnen der Häufigkeitstabellen
#

# Items auswählen
items <- c("TA", "CC", "GR", "VO")
# Tabelle erzeugen
tab <- apply(prodRat[, items], 2,
             FUN=function(x){
               prop.table(table(x))*100})
print(tab, digits = 2)

# Mittelwert der Ratings berechnen
round(apply(prodRat[, items], 2, mean), 2)

# ------------------------------------------------------------
# Abschnitt 7.2, Listing 2: Beurteilerübereinstimmung berechnen
#

items <- c("TA", "CC", "GR", "VO")
dfr <- data.frame(items, agree = NA,
                  kappa = NA, wkappa = NA, korr = NA)
for(i in 1:length(items)){
  dat.i <- prodPRat[, grep(items[i], colnames(prodPRat))]
  dfr[i, "agree"] <- agree(dat.i, tolerance = 1)["value"]
  dfr[i, "kappa"] <- kappa2(dat.i)["value"]
  dfr[i, "wkappa"] <- kappa2(dat.i, weight = "squared")["value"]
  dfr[i, "korr"] <- cor(dat.i[,1], dat.i[,2])
  dfr[i, "icc"] <- icc(dat.i, model = "twoway")["value"]
}
print(dfr, digits = 3)


## ------------------------------------------------------------
## Abschnitt 7.3: Skalierungsmodelle
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 1: Skalierungsmodell mit TAM
#

set.seed(1234)

# Rater-Facette definieren
facets <- prodRat[, "rater", drop = FALSE]
# Response Daten definieren
vars <- c("TA", "CC", "GR", "VO")
resp <- prodRat[, vars]
# Personen-ID definieren
pid <- prodRat$idstud
```

```
# Formel für Modell
formulaA <- ~item*step+item*rater
# Modell berechnen
mod <- tam.mml.mfr(resp = resp, facets = facets, formulaA = formulaA,
                   pid = pid, control=list(xsi.start0 = 1,
                                           fac.oldxsi = 0.1,
                                           increment.factor = 1.05))
summary(mod, file="TAM_MFRM")

# Personenparameter und Rohscores
persons.mod <- tam.wle(mod)
persons.mod$raw.score <- persons.mod$PersonScores / (persons.mod$N.items)

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 1b: Ergänzung zum Buch
# Modellvergleich aller besprochenen Modelle
#

f1 <- ~item * rater * step
mod1 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f1,
                    pid = pid, control=list(xsi.start0 = 1,
                                            fac.oldxsi = 0.1,
                                            increment.factor = 1.05))
f2 <- ~item*step+item*rater
mod2 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f2,
                    pid = pid, control=list(xsi.start0 = 1,
                                            fac.oldxsi = 0.1,
                                            increment.factor = 1.05))
f3 <- ~item * step + rater
mod3 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f3,
                    pid = pid, control=list(xsi.start0 = 1,
                                            fac.oldxsi = 0.1,
                                            increment.factor = 1.05))
f4 <- ~item + step + rater
mod4 <- tam.mml.mfr(resp = resp, facets = facets, formulaA = f4,
                    pid = pid, control=list(xsi.start0 = 1,
                                            fac.oldxsi = 0.1,
                                            increment.factor = 1.05))
mod4$xsi.facets
IRT.compareModels(mod1, mod2, mod3, mod4)

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 1c: Ergänzung zum Buch
# Wright-Map: Items und Rater
#

item.labs <- vars
rater.labs <- unique(prodRat$rater)
item.labs <- c(item.labs, rep(NA, length(rater.labs) -
                                length(item.labs)))

pars <- mod$xsi.facets$xsi
facet <- mod$xsi.facets$facet
```

```
item.par <- pars[facet == "item"]
rater.par <- pars[facet == "rater"]
item_rat <- pars[facet == "item:rater"]
len <- length(item_rat)
item.long <- c(item.par, rep(NA, len - length(item.par)))
rater.long <- c(rater.par, rep(NA, len - length(rater.par)))

wrightMap(persons.mod$theta, rbind(item.long, rater.long),
          label.items = c("Items",  "Rater"),
          thr.lab.text = rbind(item.labs, rater.labs),
          axis.items = "", min.l=-3, max.l=3,
          axis.persons = "Personen")

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 2: Fit-Indices berechnen
#

# Infit/Outfit berechnen
pseudo_items <- colnames(mod$resp)
pss <- strsplit(pseudo_items , split="-")
item_parm <- unlist(lapply(pss, FUN = function(ll){ll[1]}))
rater_parm <- unlist(lapply(pss, FUN = function(ll){ll[2]}))

# Fit Items
res.items <- msq.itemfitWLE(mod, item_parm)
summary(res.items)

# Fit Rater
res.rater <- msq.itemfitWLE(mod, rater_parm)
summary(res.rater)

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 2a: Ergänzung zum Buch
# Abbildung: Histogramm, Rohscores
#

dev.off()
par(mfcol=c(1,2))

hist(persons.mod$theta, col="grey", breaks=40,
     main = "",
     xlab = "Theta (logits)",
     ylab = "Häufigkeit")
with(persons.mod, scatter.smooth(raw.score, theta,
                                 pch = 1, cex = .6, xlab = "Rohscores",
                                 ylab = "Theta (logits)",
                                 lpars = list(col = "darkgrey", lwd = 2,
                                              lty = 1)))

# Abbildung: Fit-Statistik
par(mfcol=c(1,2))
fitdat <- res.rater$fit_data
fitdat$var <- factor(substr(fitdat$item, 1, 2))
```

```
boxplot(Outfit~var, data=fitdat,
        ylim=c(0,2), main="Outfit")
boxplot(Infit~var, data=fitdat,
        ylim=c(0,2), main="Infit")

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 2b: Ergänzung zum Buch
# Korrelationen
#

korr <- c(with(persons.mod, cor(raw.score, theta,
                                method = "pearson")),
          with(persons.mod, cor(raw.score, theta,
                                method = "spearman")))
print(korr)

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 3: Q3-Statistik berechnen
#

# Q3 Statistik
mfit.q3 <- tam.modelfit(mod)
rater.pairs <- mfit.q3$stat.itempair

# Nur Paare gleicher Rater wählen
unique.rater <- which(substr(rater.pairs$item1, 4,12) ==
                        substr(rater.pairs$item2, 4,12))
rater.q3 <- rater.pairs[unique.rater, ]

# Spalten einfügen: Rater, Kombinationen
rater.q3$rater <- substr(rater.q3$item1, 4, 12)
rater.q3 <- rater.q3[order(rater.q3$rater),]
rater.q3$kombi <- as.factor(paste(substr(rater.q3$item1, 1, 2),
                                  substr(rater.q3$item2, 1, 2), sep="_"))

# Statistiken aggregieren: Rater, Kombinationen
dfr.raterQ3 <- aggregate(rater.q3$aQ3, by = list(rater.q3$rater), mean)
colnames(dfr.raterQ3) <- c("Rater", "Q3")
dfr.itemsQ3 <- aggregate(rater.q3$aQ3, by = list(rater.q3$kombi), mean)
colnames(dfr.itemsQ3) <- c("Items", "Q3")
dfr.itemsQ3

# ------------------------------------------------------------
# Abschnitt 7.3, Listing 3a: Ergänzung zum Buch
# Lattice Dotplot
#

# Lattice Dotplot
mean.values <- aggregate(rater.q3$aQ3, list(rater.q3$kombi), mean)[["x"]]
dotplot(aQ3~kombi, data=rater.q3, main="Q3-Statistik", ylab="Q3 (adjustiert)",
        col="darkgrey",
        panel = function(x,...){
          panel.dotplot(x,...)
```

```
          panel.abline(h = 0, col.line = "grey", lty=3)
          grid.points(1:6, mean.values, pch=17)
        })


## ------------------------------------------------------------
## Abschnitt 7.4: Generalisierbarkeitstheorie
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 7.4, Listing 1: Varianzkomponenten mit lme4 berechnen
#

# Formel definieren
formula1 <- response ~ (1|idstud) + (1|item) + (1|rater) +
  (1|rater:item) + (1|idstud:rater) +
  (1|idstud:item)
# Modell mit Interaktionen
mod.vk <- lmer(formula1, data=prodRatL)

# Zusammenfassung der Ergebnisse
summary(mod.vk)

# ------------------------------------------------------------
# Abschnitt 7.4, Listing 1a: Ergänzung zum Buch
# Helper-Function um die Varianzkomponenten zu extrahieren
#

summary.VarComp <- function(mod){
  var.c <- VarCorr(mod)
  var.c <- c(unlist(var.c) , attr(var.c , "sc")^2)
  names(var.c)[length(var.c)] <- "Residual"
  dfr1 <- data.frame(var.c)
  colnames(dfr1) <- "Varianz"
  dfr1 <- rbind(dfr1, colSums(dfr1))
  rownames(dfr1)[nrow(dfr1)] <- "Total"
  dfr1$prop.Varianz <- 100 * (dfr1$Varianz / dfr1$Varianz[nrow(dfr1)])
  dfr1 <- round(dfr1,2)
  return(dfr1)
}
summary.VarComp(mod.vk)

# ------------------------------------------------------------
# Abschnitt 7.4, Listing 2: Berechnung des G-Koeffizienten
#

vk <- summary.VarComp(mod.vk)
n.p <- length(unique(prodRatL$idstud)) # Anzahl Schüler
n.i <- 4  # Anzahl Items
n.r <- c(1,2,5) # Anzahl Rater

# Varianzkomponenten extrahieren
sig2.p <- vk["idstud", "Varianz"]
```

```
sig2.i <- vk["item", "Varianz"]
sig2.r <- vk["rater", "Varianz"]
sig2.ri <- vk["rater:item", "Varianz"]
sig2.pr <- vk["idstud:rater", "Varianz"]
sig2.pi <- vk["idstud:item", "Varianz"]
sig2.pir <- vk["Residual", "Varianz"]

# Fehlervarianz berechnen
sig2.delta <- sig2.pi/n.i + sig2.pr/n.r + sig2.pir/(n.i*n.r)

# G-Koeffizient berechnen
g.koeff <- sig2.p / (sig2.p + sig2.delta)
print(data.frame(n.r, g.koeff), digits = 3)

# ------------------------------------------------------------
# Abschnitt 7.4, Listing 2a: Ergänzung zum Buch
# Phi-Koeffizient berechnen
#

sig2.D <- sig2.r/n.r + sig2.i/n.i + sig2.pi/n.i + sig2.pr/n.r +
  sig2.ri/(n.i*n.r) + sig2.pir/(n.i*n.r)
phi.koeff <- sig2.p / (sig2.p + sig2.D)
print(data.frame(n.r, phi.koeff), digits = 3)

# Konfidenzintervalle
1.96*sqrt(sig2.D)

# ------------------------------------------------------------
# Abschnitt 7.4, Listing 2c: Ergänzung zum Buch
# Variable Rateranzahl
#

dev.off()
n.i <- 4  # Anzahl Items
dn.r <- seq(1,10)# 1 bis 10 mögliche Rater
delta.i <- sig2.pi/n.i + sig2.pr/dn.r + sig2.pir/(n.i*dn.r)
g.koeff <- sig2.p / (sig2.p + delta.i)
names(g.koeff) <- paste("nR", dn.r, sep="_")
print(g.koeff[1:4])

# Abbildung variable Rateranzahl
plot(g.koeff, type = "b", pch = 19, lwd = 2, bty = "n",
     main = "G-Koeffizient: Raters",
     ylab = "G-Koeffizient",
     xlab = "Anzahl Raters",  xlim = c(0,10))
abline(v=2, col="darkgrey")


## ------------------------------------------------------------
## Abschnitt 7.5: Strukturgleichungsmodelle
## ------------------------------------------------------------

# ------------------------------------------------------------
```

```
# Abschnitt 7.5, Listing 1: SEM
#

# SEM Modell definieren
lv.mod <- "
  # Messmodell
  TA =~ 1*TA_R1 + 1*TA_R2
  CC =~ 1*CC_R1 + 1*CC_R2
  GR =~ 1*GR_R1 + 1*GR_R2
  VO =~ 1*VO_R1 + 1*VO_R2

  # Varianz Personen
  TA ~~ Vta * TA
  CC ~~ Vcc * CC
  GR ~~ Vgr * GR
  VO ~~ Vvo * VO

  # Varianz Rater X Personen
  TA_R1 ~~ Vta_R12 * TA_R1
  TA_R2 ~~ Vta_R12 * TA_R2
  CC_R1 ~~ Vcc_R12 * CC_R1
  CC_R2 ~~ Vcc_R12 * CC_R2
  GR_R1 ~~ Vgr_R12 * GR_R1
  GR_R2 ~~ Vgr_R12 * GR_R2
  VO_R1 ~~ Vvo_R12 * VO_R1
  VO_R2 ~~ Vvo_R12 * VO_R2

  # Kovarianz
  TA_R1 ~~ Kta_cc * CC_R1
  TA_R2 ~~ Kta_cc * CC_R2
  TA_R1 ~~ Kta_gr * GR_R1
  TA_R2 ~~ Kta_gr * GR_R2
  TA_R1 ~~ Kta_vo * VO_R1
  TA_R2 ~~ Kta_vo * VO_R2
  CC_R1 ~~ Kcc_gr * GR_R1
  CC_R2 ~~ Kcc_gr * GR_R2
  CC_R1 ~~ Kcc_vo * VO_R1
  CC_R2 ~~ Kcc_vo * VO_R2
  GR_R1 ~~ Kgr_vo * VO_R1
  GR_R2 ~~ Kgr_vo * VO_R2

  # ICC berechnen
  icc_ta := Vta / (Vta + Vta_R12)
  icc_cc := Vcc / (Vcc + Vcc_R12)
  icc_gr := Vgr / (Vgr + Vgr_R12)
  icc_vo := Vvo / (Vvo + Vvo_R12)
  "
# Schätzung des Modells
mod1 <- sem(lv.mod, data = prodPRat)
summary(mod1, standardized = TRUE)

# Inspektion der Ergebnisse
show(mod1)
```

```
fitted(mod1)
inspect(mod1,"cov.lv")
inspect(mod1, "free")

# ------------------------------------------------------------
# Abschnitt 7.5, Listing 2: Kompakte Darstellung der Ergebnisse
#

parameterEstimates(mod1, ci = FALSE,
                   standardized = TRUE)

# ------------------------------------------------------------
# Abschnitt 7.5, Listing 2a: Ergänzung zum Buch
# Schreibe Ergebnisse in Latex-Tabelle:
#

xtable(parameterEstimates(mod1, ci = FALSE,
                          standardized = TRUE), digits = 3)

## End(Not run)
```

Kapitel 8 *Kapitel 8: Fehlende Daten und Plausible Values*

Description

Das ist die Nutzerseite zum Kapitel 8, *Fehlende Daten und Plausible Values*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Details

Vorbereitungen: Zur Illustration der Konsequenzen fehlender Daten und der Messfehlerbehaftetheit von Variablen soll zunächst ein Illustrationsdatensatz (`data08I`) mit N=1500 simuliert werden. Dabei sollen zwei Variablen vorliegen: Der Sozialstatus `X` soll teilweise fehlende Werte aufweisen und die zu erfassende Kompetenz liegt sowohl als wahrer Wert θ als auch als messfehlerbehaftete Variable $\hat{\theta}$ vor. Im Datensatz `data08I` liegt sowohl der vollständig beobachtete Sozialstatus (`x`) als auch derselbe Sozialstatus mit teilweise fehlenden Variablen (`X`) vor. Neben dem Illustrationsdatensatz werden in diesem Kapitel Datensätze der österreichischen Bildungsstandards im Fach Englisch verwendet. Der Datensatz `data08H` enthält Kovariaten (d.h. Variablen aus Fragebögen oder administrative Daten) auf Ebene der Schüler (Ebene 1) und auf Ebene der Schulen (Ebene 2). Variablen beider Ebenen können dabei fehlende Werte besitzen. Im Datensatz `data08J` sind fehlende Werte des Datensatzes `data08H` durch eine Ersetzung von Werten bereits aufgefüllt. Außerdem liegen Item Responses der Schüler für den Bereich Hörverstehen (Listening, L) im Datensatz `data08K` vor. Folgende R-Pakete werden in diesem Kapitel verwendet: `mice`, `miceadds`, `TAM`, `pls`.

```
library(miceadds)
library(mice)
library(TAM)
library(pls)
```

Abschnitt 8.1.1: Konsequenzen fehlender Daten und messfehlerbehafteter Variablen:

Listing 1: Deskriptive Statistiken des Datensatzes: Mit folgendem R-Code werden deskriptive Statistiken des Datensatzes `data08I` ermittelt, an denen die Bedeutung der geeigneten Behandlung fehlender Werte und von Messfehlern herausgearbeitet werden soll.

```
data(datenKapitel08)
dat <- datenKapitel08$data08I[,-1]
#*** Missinganteile
round( colMeans( is.na(dat), na.rm=TRUE) , 2 )
#*** Mittelwerte
round( apply( dat , 2 , mean , na.rm=TRUE ) , 2 )
#*** Zusammenhang von Missingindikator und Variablen
round( miceadds::mi_dstat( dat[,c("WLE","X")] ) , 2 )
#*** Varianzen
round( apply( dat , 2 , var , na.rm=TRUE ) , 2 )
#*** Korrelationsmatrix
round( cor( dat , use = "pairwise.complete.obs") , 2 )
```

Abschnitt 8.2.5: Durchführung der multiplen Imputation in R:

Listing 2: Variablenauswahl und leere Imputation: In diesem Abschnitt wird die multiple Imputation basierend auf dem MICE-Ansatz im Paket `mice` in R umgesetzt. Als Datensatz soll `data08H` verwendet werden. Zur Vereinfachung der Darstellung wählen wir auf der Ebene der Schüler die Variablen Sozialstatus (`HISEI`), Anzahl der Bücher zu Hause (`buch`) und den WLE der Hörverstehenskompetenz (`E8LWLE`) sowie einen auf der Schulebene erfassten Sozialstatus (`SES_Schule`) aus.

```
set.seed(56)
dat <- datenKapitel08$data08H
# wähle Variablen aus
dat1 <- dat[ , c("idschool", "HISEI", "buch", "E8LWLE",
+ "SES_Schule") ]
colMeans(is.na(dat1))
# führe leere Imputation durch
imp0 <- mice::mice(dat1, m=0, maxit=0)
```

Listing 3: Spezifikation der Imputationsmethoden: Die nachfolgende Syntax zeigt die Spezifikation der Imputationsmethoden im Vektor `impMethod` in `mice` für unser Datenbeispiel.

```
impMethod <- imp0$method
impMethod["HISEI"] <- "2l.continuous"
```

```
# [...] weitere Spezifikationen
impMethod["SES_Schule"] <- "2lonly.norm"
impMethod
```

Listing 4: Definition der Prädiktormatrix für die Imputation in mice: Nachfolgender R-Code zeigt die Definition der Prädiktormatrix (Matrix `predMatrix`) für die Imputation in `mice`.

```
predMatrix <- imp0$predictorMatrix
predMatrix[-1,"idschool"] <- -2
# [...]
predMatrix
```

Listing 5: Datenimputation: Die Imputation kann nun unter dem Aufruf der Funktion `mice` unter Übergabe der Imputationsmethoden und der Prädiktormatrix erfolgen. Für das PMM werden 5 Donoren gewählt. Insgesamt sollen 10 imputierte Datensätze generiert werden, wobei der Algorithmus 7 Iterationen durchlaufen soll.

```
imp1 <- mice::mice( dat1, imputationMethod=impMethod,
+ predictorMatrix=predMatrix, donors=5,
+ m=10, maxit=7 )
```

Abschnitt 8.3.2: Dimensionsreduzierende Verfahren für Kovariaten im latenten Regressionsmodell:

Listing 6: Kovariatenauswahl, Interaktionsbildung und Bestimmung PLS-Faktoren: Die Methode des Partial Least Squares soll für den Datensatz `data08J` illustriert werden. Es wird zum Auffüllen der Kovariaten mit fehlenden Werten nur ein imputierter Datensatz erstellt. Danach wird eine PLS-Regression des WLE der Hörverstehenskompetenz `E8LWLE` auf Kovariaten und deren Interaktionen bestimmt. Für die Bestimmung der PLS-Faktoren wird das R-Paket `pls` verwendet. Die nachfolgende R-Syntax zeigt die Kovariatenauswahl, die Bildung der Interaktionen und die Bestimmung der PLS-Faktoren. Insgesamt entstehen durch Aufnahme der Haupteffekte und Interaktionen 55 Kovariaten.

```
dat <- datenKapitel08$data08J
#*** Kovariatenauswahl
kovariaten <- scan(what="character", nlines=2)
1: female migrant HISEI eltausb buch
6: SK LF NSchueler NKlassen SES_Schule
Read 10 items

X <- scale( dat[, kovariaten ] )
V <- ncol(X)
# bilde alle Zweifachinteraktionen
c2 <- combinat::combn(V,2)
X2 <- apply( c2 , 2 , FUN = function(cc){
+ X[,cc[1]] * X[,cc[2]] } )
X0 <- cbind( X , X2 )
mod1 <- pls::plsr( dat$E8LWLE ~ X0 , ncomp=55 )
```

```
summary(mod1)
```

Abschnitt 8.3.3: Ziehung von Plausible Values in R: In diesem Abschnitt soll die Ziehung von Plausible Values mit dem R-Paket TAM illustriert werden. Dabei beschränken wir uns auf den Kompetenzbereich des Hörverstehens.

Listing 7: PLS-Faktoren auswählen: In Abschnitt 8.3.2 wurden dabei die Kovariaten auf PLS-Faktoren reduziert. Für die Ziehung von Plausible Values werden nachfolgend 10 PLS-Faktoren verwendet.

```
facs <- mod1$scores[,1:10]
```

Listing 8: Rasch-Skalierung: Für die Hörverstehenskompetenz im Datensatz data08K wird nachfolgend das Messmodell als Rasch-Modell geschätzt. Dabei werden Stichprobengewichte für die Bestimmung der Itemparameter verwendet.

```
dat2 <- datenKapitel08$data08K
items <- grep("E8L", colnames(dat2), value=TRUE)
# Schätzung des Rasch-Modells in TAM
mod11 <- TAM::tam.mml( resp= dat2[,items ] ,
+ pid = dat2$idstud,
+ pweights = dat2$wgtstud )
```

Listing 9: Individuelle Likelihood, latente Regressionsmodell und PV-Ziehung: Bei einer Fixierung von Itemparametern ist die bedingte Verteilung $P(\mathbf{X}|\boldsymbol{\theta})$ des Messmodells konstant. Die Schätzung von Item-Response-Modellen erfolgt in TAM unter Verwendung eines diskreten Gitters von $\boldsymbol{\theta}$-Werten. Während der Anpassung des Rasch-Modells in mod11 liegt daher die auf diesem Gitter ausgewertete sog. individuelle Likelihood $P(\mathbf{X}|\boldsymbol{\theta})$ vor, die mit der Funktion IRT.likelihood aus dem Objekt mod11 extrahiert werden kann. Das latente Regressionsmodell kann unter Rückgriff auf die individuelle Likelihood mit der Funktion tam.latreg angepasst werden. Die Ziehung der Plausible Values erfolgt anschließend mit der Funktion tam.pv.

```
#*** extrahiere individuelle Likelihood
lmod11 <- IRT.likelihood(mod11)
#*** schätze latentes Regressionsmodell
mod12 <- TAM::tam.latreg( like = lmod11 , Y = facs )
#*** ziehe Plausible Values
pv12 <- TAM::tam.pv(mod12, normal.approx=TRUE,
+ samp.regr=TRUE , ntheta=400)
```

Listing 10: Plausible Values extrahieren: Die imputierten Plausible Values lassen sich im Element $pv des Ergebnisobjekts aus tam.pv extrahieren.

```
#*** Plausible Values für drei verschiedene Schüler
round( pv12$pv[c(2,5,9),] , 3 )
```

Author(s)

Alexander Robitzsch, Giang Pham, Takuya Yanagida

References

Robitzsch, A., Pham, G. & Yanagida, T. (2016). Fehlende Daten und Plausible Values. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 259–293). Wien: facultas.

See Also

Zu datenKapitel08, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 7, Statistische Analysen produktiver Kompetenzen .
Zu Kapitel 9, Fairer Vergleich in der Rückmeldung.
Zur Übersicht.

Examples

```
## Not run:
library(TAM)
library(mice)
library(miceadds)
library(pls)
library(combinat)
library(mitml)

data(datenKapitel08)
data08H <- datenKapitel08$data08H
data08I <- datenKapitel08$data08I
data08J <- datenKapitel08$data08J
data08K <- datenKapitel08$data08K

## ----------------------------------------------------------------
## Abschnitt 8.1.1: Konsequenzen fehlender Daten und
##                  messfehlerbehafteter Variablen
## ----------------------------------------------------------------

# ----------------------------------------------------------------
# Abschnitt 8.1.1, Listing 1: Deskriptive Statistiken des
#                             Illustrationsdatensatzes
#

data(datenKapitel08)
dat <- datenKapitel08$data08I[,-1]
#*** Missinganteile
round( colMeans( is.na(dat), na.rm=TRUE) , 2 )
#*** Mittelwerte
round( apply( dat , 2 , mean , na.rm=TRUE ) , 2 )
#*** Zusammenhang von Missingindikator und Variablen
round( miceadds::mi_dstat( dat[,c("WLE","X")] ) , 2 )
#*** Varianzen
```

```
round( apply( dat , 2 , var , na.rm=TRUE ) , 2 )
#*** Korrelationsmatrix
round( cor( dat , use = "pairwise.complete.obs") , 2 )


## ------------------------------------------------------------
## Abschnitt 8.2: Multiple Imputation
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 1: Variablenauswahl und leere
#                             Imputation
#

set.seed(56)
data(datenKapitel08)
dat <- datenKapitel08$data08H
# wähle Variablen aus
dat1 <- dat[ , c("idschool", "HISEI", "buch", "E8LWLE",
                 "SES_Schule") ]
colMeans(is.na(dat1))
# führe leere Imputation durch
imp0 <- mice::mice(dat1, m=0, maxit=0)

# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 2: Spezifikation der Imputations-
#                             methoden
#

impMethod <- imp0$method
impMethod["HISEI"] <- "2l.continuous"
# [...]  weitere Spezifikationen
impMethod["SES_Schule"] <- "2lonly.norm"
impMethod

# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 2b: Ergänzung zum Buch
#

# [...]  weitere Spezifikationen
impMethod["buch"]  <- "2l.pmm"
impMethod

# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 3: Definition der Prädiktormatrix
#                             für die Imputation in mice
#

predMatrix <- imp0$predictorMatrix
predMatrix[-1,"idschool"] <- -2
# [...]
predMatrix
```

```
# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 3b: Ergänzung zum Buch
#

# [...]
predMatrix[2:4,2:4] <- 3*predMatrix[2:4,2:4]
predMatrix

# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 4: Führe Imputation durch
#

imp1 <- mice::mice( dat1, imputationMethod=impMethod,
  predictorMatrix=predMatrix, donors=5, m=10, maxit=7)

# ------------------------------------------------------------
# Abschnitt 8.2.5, Listing 4b: Ergänzung zum Buch
#

#-- Mittelwert HISEI
wmod1 <- with( imp1 , lm(HISEI ~ 1))
summary( mice::pool( wmod1 ) )

#-- lineare Regression HISEI auf Büchervariable
wmod2 <- with( imp1 , lm(E8LWLE ~ HISEI) )
summary( mice::pool( wmod2 ))

#-- Inferenz Mehrebenenmodelle mit Paket mitml
imp1b <- mitml::mids2mitml.list(imp1)
wmod3 <- with(imp1b, lme4::lmer( HISEI ~ (1|idschool)) )
mitml::testEstimates(wmod3, var.comp=TRUE)

## ------------------------------------------------------------
## Abschnitt 8.3.2: Dimensionsreduzierende Verfahren für
## Kovariaten im latenten Regressionsmodell
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 8.3.2, Listing 1: Kovariatenauswahl, Interaktions-
#                          bildung und Bestimmung PLS-Faktoren
#

set.seed(56)
data(datenKapitel08)
dat <- datenKapitel08$data08J

#*** Kovariatenauswahl
kovariaten <- scan(what="character", nlines=2)
  female migrant HISEI  eltausb buch
  SK LF NSchueler NKlassen SES_Schule

X <- scale( dat[, kovariaten ] )
V <- ncol(X)
```

```
# bilde alle Zweifachinteraktionen
c2 <- combinat::combn(V,2)
X2 <- apply( c2 , 2 , FUN = function(cc){
            X[,cc[1]] * X[,cc[2]] } )
X0 <- cbind( X , X2 )
# Partial Least Squares Regression
mod1 <- pls::plsr( dat$E8LWLE ~ X0 , ncomp=55  )
summary(mod1)

# ------------------------------------------------------------
# Abschnitt 8.3.2, Listing 1b: Ergänzung zum Buch
# Abbildung: Aufgeklärter Varianzanteil
#

# Principal Component Regression (Extraktion der Hauptkomponenten)
mod2 <- pls::pcr(  dat$E8LWLE ~ X0 , ncomp=55 )
summary(mod2)

#*** extrahierte Varianzen mit PLS-Faktoren und PCA-Faktoren
res <- mod1
R2 <- base::cumsum(res$Xvar) / res$Xtotvar
ncomp <- 55
Y <- dat$E8LWLE
R21 <- base::sapply( 1:ncomp , FUN = function(cc){
  1 - stats::var( Y -  res$fitted.values[,1,cc] ) / stats::var( Y )
} )
dfr <- data.frame("comp" = 1:ncomp , "PLS" = R21 )

res <- mod2
R2 <- base::cumsum(res$Xvar) / res$Xtotvar
ncomp <- 55
Y <- dat$E8LWLE
R21 <- base::sapply( 1:ncomp , FUN = function(cc){
           1 - stats::var( Y -  res$fitted.values[,1,cc] ) / stats::var( Y )
} )
dfr$PCA <- R21


plot( dfr$comp , dfr$PLS , type="l" , xlab="Anzahl Faktoren" ,
      ylab="Aufgeklärter Varianzanteil" ,
      ylim=c(0,.3) )
points( dfr$comp , dfr$PLS , pch=16 )
points( dfr$comp , dfr$PCA , pch=17 )
lines( dfr$comp , dfr$PCA , lty=2 )
legend( 45 , .15 , c("PLS" , "PCA") , pch=c(16,17) , lty=c(1,2))

## ------------------------------------------------------------
## Abschnitt 8.3.3: Ziehung von Plausible Values in R
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 8.3.3, Listing 1: PLS-Faktoren auswählen
#
```

```
facs <- mod1$scores[,1:10]

# ------------------------------------------------------------
# Abschnitt 8.3.3, Listing 1b: Ergänzung zum Buch
#
set.seed(98766)

# ------------------------------------------------------------
# Abschnitt 8.3.3, Listing 2: Anpassung kognitive Daten
#

data(datenKapitel08)
dat2 <- datenKapitel08$data08K
items <- grep("E8L", colnames(dat2), value=TRUE)
# Schätzung des Rasch-Modells in TAM
mod11 <- TAM::tam.mml( resp= dat2[,items ] ,
       pid = dat2$idstud, pweights = dat2$wgtstud )

# ------------------------------------------------------------
# Abschnitt 8.3.3, Listing 3: Individuelle Likelihood, latentes
#                             Regressionsmodell und PV-Ziehung
#

#*** extrahiere individuelle Likelihood
lmod11 <- IRT.likelihood(mod11)
#*** schätze latentes Regressionsmodell
mod12 <- TAM::tam.latreg( like = lmod11 , Y = facs )
#*** ziehe Plausible Values
pv12 <- TAM::tam.pv(mod12, normal.approx=TRUE,
               samp.regr=TRUE , ntheta=400)

# ------------------------------------------------------------
# Abschnitt 8.3.3, Listing 4: Plausible Values extrahieren
#

#*** Plausible Values für drei verschiedene Schüler
round( pv12$pv[c(2,5,9),] , 3 )


# ------------------------------------------------------------
# Abschnitt 8.3.3, Listing 4b: Ergänzung zum Buch
#

hist( pv12$pv$PV1.Dim1 )

# Korrelation mit Kovariaten
round( cor( pv12$pv$PV1.Dim1 , dat[,kovariaten] ,
            use="pairwise.complete.obs") , 3 )
round( cor( dat$E8LWLE , dat[,kovariaten] ,
            use="pairwise.complete.obs" ) , 3 )

## End(Not run)
```

Kapitel 9 *Kapitel 9: Fairer Vergleich in der Rueckmeldung*

Description

Das ist die Nutzerseite zum Kapitel 9, *Fairer Vergleich in der Rückmeldung*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Details

Vorbereitungen: Der zur Illustration verwendete Datensatz `dat` beinhaltet (imputierte) aggregierte Leistungs- und Hintergrunddaten von 244 Schulen, bestehend aus 74 ländlichen allgemeinbildenden Pflichtschulen (APS, Stratum 1), 69 städtischen APS (Stratum 2), 52 ländlichen allgemeinbildenden höheren Schulen (AHS, Stratum 3) und 49 städtischen AHS (Stratum 4). Im Kapitel wird zur Bildung von Interaktionseffekten und quadratischen Termen der Kovariaten eine neue Funktion `covainteraction` verwendet.

```
> data(datenKapitel09)
> dat <- datenKapitel09
>
> covainteraction <- function(dat,covas,nchar){
+ for(ii in 1:(length(covas))){
+ vv1 <- covas[ii]
+ # Interaktion von vv1 mit sich selbst
+ subname1 <- substr(vv1,1,nchar)
+ intvar <- paste0(subname1, subname1)
+ if(vv1 == covas[1]){
+ dat.int <- dat[,vv1]*dat[,vv1];
+ newvars <- intvar } else {
+ dat.int <- cbind(dat.int,dat[,vv1]*dat[,vv1]);
+ newvars <- c(newvars,intvar)
+ }
+ # Interaktion von vv1 mit restlichen Variablen
+ if(ii < length(covas)){
+ for(jj in ((ii+1):length(covas))){
+ vv2 <- covas[jj]
+ subname2 <- substr(vv2,1,nchar)
+ intvar <- paste0(subname1, subname2)
+ newvars <- c(newvars, intvar)
+ dat.int <- cbind(dat.int,dat[,vv1]*dat[,vv2])
+ }
+ }
+
+ }
```

```
+ dat.int <- data.frame(dat.int)
+ names(dat.int) <- newvars
+ return(dat.int)
+ }
```

Abschnitt 9.2.5.1: Kovariaten und Interaktionsterme:

Listing 1: Kovariatenauswahl und z-Standardisierung: Als Variablen zur Beschreibung von Kontext und Schülerzusammensetzung in den Schulen werden in diesem Beispiel die logarithmierte Schulgröße `groesse`, der Anteil an Mädchen `female`, der Anteil an Schülerinnen und Schülern mit Migrationshintergrund `mig` und der mittlere sozioökonomische Status (SES) `sozstat` eingeführt. Die abhängige Variable ist die aggregierte Schülerleistung der Schule `TWLE`. Alle Kovariaten (`vars`) werden zunächst z-standardisiert (`zvars`).

```
> vars <- c("groesse","female","mig","sozstat")
> zvars <- paste0("z",vars)
> dat[,zvars] <- scale(dat[,vars],scale = TRUE)
```

Listing 2: Interaktionen bilden, z-standardisieren: Zur Optimierung der Modellspezifikation werden Interaktionseffekte und quadratische Terme der Kovariaten gebildet, dann z-standardisiert und in den Gesamtdatensatz hinzugefügt. Die neuen Variablennamen sind in der Liste `intvars` aufgelistet.

```
> dat1 <- LSAmitR::covainteraction(dat = dat,covas = zvars,nchar = 4)
> intvars <- names(dat1) # Interaktionsvariablen
> dat1[,intvars] <- scale(dat1[,intvars],scale = TRUE)
> dat <- cbind(dat,dat1)
```

Listing 3: Haupt- und Interaktionseffekte:

```
> maineff <- zvars # Haupteffekte
> alleff <- c(zvars,intvars) # Haupt- und Interaktionseffekte
```

Abschnitt 9.2.5.2: OLS-Regression:

Listing 4: OLS-Regression mit Haupteffekten: Das OLS-Regressionsmodell mit den Haupteffekten als Modellprädiktoren (`ols.mod1`) für Schulen im Stratum (`st`) 4

```
> fm.ols1 <- paste0("TWLE ~ ",paste(maineff,collapse=" + "))
> fm.ols1 <- as.formula(fm.ols1) # Modellgleichung
> st <- 4
> pos <- which(dat$stratum == st) # Schulen im Stratum st
> ols.mod1 <- lm(formula = fm.ols1,data = dat[pos,]) # Regression
```

Abschnitt 9.2.5.3: Lasso-Regression:

Listing 5: Datenaufbereitung: Für die Durchführung der Lasso-Regression wird das R-Paket glmnet (Friedman et al., 2010) eingesetzt. Die Kovariatenmatrix (`Z`) sowie der Vektor der abhängigen Leistungswerte (`Y`) müssen definiert werden.

```
> library(glmnet)
> Z <- as.matrix(dat[pos,alleff]) # Kovariatenmatrix
> Y <- dat$TWLE[pos] # Abhängige Variable
```

Listing 6: Bestimmung Teilmengen für Kreuzvalidierung, Lasso-Regression: Das Lasso-Verfahren wird mit der Funktion `cv.glmnet()` durchgeführt. Zur Auswahl eines optimalen shrinkage λ wird das Verfahren der K-fachen Kreuzvalidierung verwendet. Die Schulstichprobe wird durch ID-Zuweisung (`foldid`) verschiedenen Teilmengen zugewiesen.

```
> nid <- floor(length(pos)/3) # Teilmengen definieren
> foldid <- rep(c(1:nid),3,length.out=length(pos)) # Zuweisung
> lasso.mod2 <- glmnet::cv.glmnet(x=Z,y=Y,alpha = 1, foldid = foldid)
```

Listing 7: Erwartungswerte der Schulen: Entsprechend `lambda.min` werden die Regressionskoeffizienten und die entsprechenden Erwartungswerte der Schulen bestimmt.

```
> lasso.pred2 <- predict(lasso.mod2,newx = Z,s="lambda.min")
> dat$expTWLE.Lasso2[pos] <- as.vector(lasso.pred2)
```

Listing 8: Bestimmung R^2: Bestimmung des aufgeklärten Varianzanteils der Schulleistung `R^2`.

```
> varY <- var(dat$TWLE[pos])
> varY.lasso.mod2 <- var(dat$expTWLE.Lasso2[pos])
> R2.lasso.mod2 <- varY.lasso.mod2/varY
```

Abschnitt 9.2.5.4: Nichtparametrische Regression:

Listing 9: Distanzberechnung: Der erste Schritt zur Durchführung einer nichtparametrischen Regression ist die Erstellung der Distanzmatrix zwischen Schulen. In diesem Beispiel wird die euklidische Distanz als Distanzmaß berechnet, alle standardisierten Haupteffekte sind eingeschlossen. Außerdem setzen wir die Gewichte von allen Kovariaten (`gi`) auf 1. `dfr.i` in diesem Beispiel ist die Distanzmatrix für erste Schule im Stratum.

```
> N <- length(pos) # Anzahl Schulen im Stratum
> schools <- dat$idschool[pos] # Schulen-ID
> i <- 1
> # Teildatensatz von Schule i
> dat.i <- dat[pos[i],c("idschool","TWLE",maineff)]
> names(dat.i) <- paste0(names(dat.i),".i")
> # Daten der Vergleichsschulen
> dat.vgl <- dat[pos[-i],c("idschool","TWLE",maineff)]
> index.vgl <- match(dat.vgl$idschool,schools)
```

```
> # Daten zusammenfügen
> dfr.i <- data.frame("index.i"=i,dat.i,"index.vgl"=index.vgl,
+ dat.vgl, row.names=NULL)
> # Distanz zur Schule i
> dfr.i$dist <- 0
> gi <- c(1,1,1,1)
> for(ii in 1:length(maineff)){
+ vv <- maineff[ii]
+ pair.vv <- grep(vv, names(dfr.i), value=T)
+ dist.vv <- gi[ii]*((dfr.i[,pair.vv[1]]-dfr.i[,pair.vv[2]])^2)
+ dfr.i$dist <- dfr.i$dist + dist.vv }
```

Listing 10: H initiieren:
$p(x) = \frac{\lambda^x e^{-\lambda}}{x!}$.
Die Gewichte w_{ik} für jedes Paar (i, k) von Schulen werden mithilfe der Distanz, der Gauß'schen Kernfunktion (dnorm) als Transformationsfunktion sowie einer schulspezifischen Bandweite h_i berechnet. Die Auswahl des optimalen Werts $\hat{h_i}$ für jede Schule i erfolgt nach Vieu (1991). Zunächst wird ein Vektor H so gewählt, dass der optimale Wert $\hat{h_i}$ größer als der kleinste und kleiner als der größte Wert in H ausfällt. Je kleiner das Intervall zwischen den Werten in H ist, desto wahrscheinlicher ist, dass ein Listenelement den optimalen Wert erlangt. Auf der anderen Seite korrespondiert die Rechenzeit mit der Länge von H. Gemäß der Größe der Vergleichsgruppe wählen wir eine Länge von 30 für H, zusätzlich wird ein sehr großer Wert (100000) für die Fälle hinzugefügt, bei denen alle Gewichte beinahe gleich sind.

```
> d.dist <- max(dfr.i$dist)-min(dfr.i$dist)
> H <- c(seq(d.dist/100,d.dist,length=30),100000)
> V1 <- length(H)
> # Anzahl Vergleichsschulen
> n <- nrow(dfr.i)
```

Listing 11: Leave-One-Out-Schätzer der jeweiligen Vergleichsschule k nach h in H: Auf Basis aller Werte in H und dem jeweils entsprechenden Gewicht w_{ik} (wgt.h) werden die Leave-One-Out-Schätzer der jeweiligen Vergleichsschule (pred.k) berechnet.

```
> sumw <- 0*H # Vektor w_{ik} initiieren, h in H
> av <- "TWLE"
> dfr0.i <- dfr.i[,c("idschool",av)]
> # Schleife über alle h-Werte
> for (ll in 1:V1 ){
+ h <- H[ll]
+ # Gewicht w_{ik} bei h
+ dfr.i$wgt.h <- dnorm(sqrt(dfr.i$dist), mean=0, sd=sqrt(h))
+ # Summe von w_{ik} bei h
+ sumw[which(H==h)] <- sum(dfr.i$wgt.h)
+ # Leave-one-out-Schätzer von Y_k
+ for (k in 1:n){
+ # Regressionsformel
+ fm <- paste0(av,"~",paste0(maineff,collapse="+"))
```

```
+ fm <- as.formula(fm)
+ # Regressionsanalyse ohne Beitrag von Schule k
+ dfr.i0 <- dfr.i[-k,]
+ mod.k <- lm(formula=fm,data=dfr.i0,weights=dfr.i0$wgt.h)
+ # Erwartungswert anhand Kovariaten der Schule k berechnen
+ pred.k <- predict(mod.k, dfr.i)[k]
+ dfr0.i[k,paste0( "h_",h) ] <- pred.k
+ }}
> # Erwartungswerte auf Basis verschiedener h-Werte
> dfr1 <- data.frame("idschool.i"=dfr.i$idschool.i[1],"h"=H )
```

Listing 12: Kreuzvalidierungskriterium nach h in H: Zur Berechnung der Kreuzvalidierungskriterien (CVh, je kleiner, desto reliabler sind die Schätzwerte) für alle Werte h in H verwenden wir in diesem Beispiel die Plug-in-Bandweite nach Altman und Leger (1995) (hAL), die mit der Funktion ALbw() des R-Pakets kerdiest aufrufbar ist.

```
> library(kerdiest)
> hAL <- kerdiest::ALbw("n",dfr.i$dist) # Plug-in Bandweite
> dfr.i$cross.h <- hAL
> dfr.i$crosswgt <- dnorm( sqrt(dfr.i$dist), mean=0, sd = sqrt(hAL) )
> # Kreuzvalidierungskriterium CVh
> vh <- grep("h_",colnames(dfr0.i),value=TRUE)
> for (ll in 1:V1){
+ dfr1[ll,"CVh"] <- sum( (dfr0.i[,av] - dfr0.i[,vh[ll]])^2 *
+ dfr.i$crosswgt) / n}
```

Listing 13: Bestimmung des optimalen Wertes von h: Der optimale Wert von h in H (h.min) entspricht dem mit dem kleinsten resultierenden CVh.

```
> dfr1$min.h.index <- 0
> ind <- which.min( dfr1$CVh )
> dfr1$min.h.index[ind] <- 1
> dfr1$h.min <- dfr1$h[ind]
```

Listing 14: Kleinste Quadratsumme der Schätzfehler: Kleinste Quadratsumme der Schätzfehler der nichtparametrischen Regression mit h=h.min.

```
> dfr1$CVhmin <- dfr1[ ind , "CVh" ]
```

Listing 15: Effizienzsteigerung: Die Effizienz (Steigerung der Schätzungsreliabilität) der nichtparametrischen Regression gegenüber der linearen Regression (äquivalent zu CVh bei h=100000).

```
> dfr1$eff_gain <- 100 * ( dfr1[V1,"CVh"] / dfr1$CVhmin[1] - 1 )
```

Listing 16: Durchführung der nichtparametrischen Regression:

```
> h <- dfr1$h.min[1] # h.min
```

```
> dfr.i$wgt.h <- dnorm(sqrt(dfr.i$dist),sd=sqrt(h))/
+ dnorm(0,sd= sqrt(h)) # w_{ik} bei h.min
> dfr.i0 <- dfr.i
> # Lokale Regression
> mod.ii <- lm(formula=fm,data=dfr.i0,weights=dfr.i0$wgt.h)
> # Kovariaten Schule i
> predM <- data.frame(dfr.i[1,paste0(maineff,".i")])
> names(predM) <- maineff
> pred.ii <- predict(mod.ii, predM) # Schätzwert Schule i
> dat$expTWLE.np[match(dfr1$idschool.i[1],dat$idschool)] <- pred.ii
```

Abschnitt 9.2.7, Berücksichtigung der Schätzfehler: Der Erwartungsbereich wird nach der im Buch beschriebenen Vorgehensweise bestimmt.

Listing 17: Bestimmung des Erwartungsbereichs: Bestimmung der Breite des Erwartungsbereichs aller Schulen auf Basis der Ergebnisse der OLS-Regression mit Haupteffekten.

```
> vv <- "expTWLE.OLS1" # Variablenname
> mm <- "OLS1" # Kurzname des Modells
> dfr <- NULL # Ergebnistabelle
> # Schleife über mögliche Breite von 10 bis 60
> for(w in 10:60){
+ # Variablen für Ergebnisse pro w
+ var <- paste0(mm,".pos.eb",w) # Position der Schule
+ var.low <- paste0(mm,".eblow",w) # Untere Grenze des EBs
+ var.upp <- paste0(mm,".ebupp",w) # Obere Grenze des EBs
+ # Berechnen
+ dat[,var.low] <- dat[,vv]-w/2 # Untere Grenze des EBs
+ dat[,var.upp] <- dat[,vv]+w/2 # Obere Grenze des EBs
+ # Position: -1=unterhalb, 0=innerhalb, 1=oberhalb des EBs
+ dat[,var] <- -1*(dat$TWLE < dat[,var.low]) + 1*(dat$TWLE > dat[,var.upp])
+ # Verteilung der Schulpositionen
+ tmp <- data.frame(t(matrix(prop.table(table(dat[,var])))))
+ names(tmp) <- c("unterhalb","innerhalb","oberhalb")
+ tmp <- data.frame("ModellxBereich"=var,tmp); dfr <- rbind(dfr,tmp) }
>
> # Abweichung zur Wunschverteilung 25-50-25
> dfr1 <- dfr
> dfr1[,c(2,4)] <- (dfr1[,c(2,4)] - .25)^2
> dfr1[,3] <- (dfr1[,3] - .5)^2
> dfr1$sumquare <- rowSums(dfr1[,-1])
> # Auswahl markieren
> dfr$Auswahl <- 1*(dfr1$sumquare == min(dfr1$sumquare) )
```

Author(s)

Giang Pham, Alexander Robitzsch, Ann Cathrice George, Roman Freunberger

References

Pham, G., Robitzsch, A., George, A. C. & Freunberger, R. (2016). Fairer Vergleich in der Rückmeldung. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 295–332). Wien: facultas.

See Also

Zu datenKapitel09, den im Kapitel verwendeten Daten.
Zurück zu Kapitel 8, Fehlende Daten und Plausible Values.
Zu Kapitel 10, Reporting und Analysen.
Zur Übersicht.

Examples

```
## Not run:
library(miceadds)
library(glmnet)
library(kerdiest)

covainteraction <- function(dat,covas,nchar){
  for(ii in 1:(length(covas))){
    vv1 <- covas[ii]
    # Interaktion von vv1 mit sich selbst
    subname1 <- substr(vv1,1,nchar)
    intvar <- paste0(subname1, subname1)
    if(vv1 == covas[1]){
      dat.int <- dat[,vv1]*dat[,vv1];
      newvars <- intvar } else {
        dat.int <- cbind(dat.int,dat[,vv1]*dat[,vv1]);
        newvars <- c(newvars,intvar)
      }
    # Interaktion von vv1 mit restlichen Variablen
    if(ii < length(covas)){
      for(jj in ((ii+1):length(covas))){
        vv2 <- covas[jj]
        subname2 <- substr(vv2,1,nchar)
        intvar <- paste0(subname1, subname2)
        newvars <- c(newvars, intvar)
        dat.int <- cbind(dat.int,dat[,vv1]*dat[,vv2])
      }
    }

  }
  dat.int <- data.frame(dat.int)
  names(dat.int) <- newvars
  return(dat.int)
}

data(datenKapitel09)
dat <- datenKapitel09
```

```
# Platzhalter für Leistungsschätzwerte aller Modelle
dat$expTWLE.OLS1 <- NA
dat$expTWLE.OLS2 <- NA
dat$expTWLE.Lasso1 <- NA
dat$expTWLE.Lasso2 <- NA
dat$expTWLE.np <- NA

## ------------------------------------------------------------
## Abschnitt 9.2.5, Umsetzung in R
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 9.2.5.1, Listing 1: Kovariatenauswahl und
#                               z-Standardisierung
#

vars <- c("groesse","female","mig","sozstat")
zvars <- paste0("z",vars)
dat[,zvars] <- scale(dat[,vars],scale = TRUE)

# ------------------------------------------------------------
# Abschnitt 9.2.5.1, Listing 2:
#

# Interaktionen bilden, z-standardisieren
dat1 <- LSAmitR::covainteraction(dat = dat,covas = zvars,nchar = 4)
intvars <- names(dat1) # Interaktionsvariablen
dat1[,intvars] <- scale(dat1[,intvars],scale = TRUE)
dat <- cbind(dat,dat1)

# ------------------------------------------------------------
# Abschnitt 9.2.5.1, Listing 3: Modellprädiktoren: Haupt- und
#                               Interaktionseffekte
#

maineff <- zvars # Haupteffekte
alleff <- c(zvars,intvars) # Haupt- und Interaktionseffekte

# ------------------------------------------------------------
# Abschnitt 9.2.5.2, Listing 4: OLS-Regression mit Haupteffekten
#

fm.ols1 <- paste0("TWLE ~ ",paste(maineff,collapse=" + "))
fm.ols1 <- as.formula(fm.ols1) # Modellgleichung
st <- 4
pos <- which(dat$stratum == st) # Schulen im Stratum st
ols.mod1 <- lm(formula = fm.ols1,data = dat[pos,]) # Regression

# ------------------------------------------------------------
# Abschnitt 9.2.5.3, Listing 5: Lasso-Regression
# Datenaufbereitung
#
```

```
library(glmnet)
Z <- as.matrix(dat[pos,alleff]) # Kovariatenmatrix
Y <- dat$TWLE[pos] # Abhängige Variable

# ------------------------------------------------------------
# Abschnitt 9.2.5.3, Listing 6: Lasso-Regression
# Bestimmung Teilmengen für Kreuzvalidierung, Lasso-Regression
#

nid <- floor(length(pos)/3) # Teilmengen definieren
foldid <- rep(c(1:nid),3,length.out=length(pos)) # Zuweisung
lasso.mod2 <- glmnet::cv.glmnet(x=Z,y=Y,alpha = 1, foldid = foldid)

# ------------------------------------------------------------
# Abschnitt 9.2.5.3, Listing 7: Lasso-Regression
# Erwartungswerte der Schulen
#

lasso.pred2 <- predict(lasso.mod2,newx = Z,s="lambda.min")
dat$expTWLE.Lasso2[pos] <- as.vector(lasso.pred2)

# ------------------------------------------------------------
# Abschnitt 9.2.5.3, Listing 8: Lasso-Regression
# Bestimmung R^2
#

varY <- var(dat$TWLE[pos])
varY.lasso.mod2 <- var(dat$expTWLE.Lasso2[pos])
R2.lasso.mod2 <- varY.lasso.mod2/varY

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 9: Nichtparametrische Regression
# Distanzberechnung zur Schule i (Stratum st)
#

N <- length(pos) # Anzahl Schulen im Stratum
schools <- dat$idschool[pos] # Schulen-ID
i <- 1
# Teildatensatz von Schule i
dat.i <- dat[pos[i],c("idschool","TWLE",maineff)]
names(dat.i) <- paste0(names(dat.i),".i")
# Daten der Vergleichsschulen
dat.vgl <- dat[pos[-i],c("idschool","TWLE",maineff)]
index.vgl <- match(dat.vgl$idschool,schools)
# Daten zusammenfügen
dfr.i <- data.frame("index.i"=i,dat.i,"index.vgl"=index.vgl,
                    dat.vgl, row.names=NULL)
# Distanz zur Schule i
dfr.i$dist <- 0
gi <- c(1,1,1,1)
for(ii in 1:length(maineff)){
  vv <- maineff[ii]
  pair.vv <- grep(vv, names(dfr.i), value=T)
```

```
  dist.vv <- gi[ii]*((dfr.i[,pair.vv[1]]-dfr.i[,pair.vv[2]])^2)
  dfr.i$dist <- dfr.i$dist + dist.vv }

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 10: Nichtparametrische Regression
#

# H initiieren
d.dist <- max(dfr.i$dist)-min(dfr.i$dist)
H <- c(seq(d.dist/100,d.dist,length=30),100000)
V1 <- length(H)
# Anzahl Vergleichsschulen
n <- nrow(dfr.i)

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 11: Nichtparametrische Regression
# Berechnung der Leave-One-Out-Schätzer der jeweiligen
# Vergleichsschule k nach h in H
#

sumw <- 0*H # Vektor w_{ik} initiieren, h in H
av <- "TWLE"
dfr0.i <- dfr.i[,c("idschool",av)]
# Schleife über alle h-Werte
for (ll in 1:V1 ){
  h <- H[ll]
  # Gewicht w_{ik} bei h
  dfr.i$wgt.h <- dnorm(sqrt(dfr.i$dist), mean=0, sd=sqrt(h))
  # Summe von w_{ik} bei h
  sumw[which(H==h)] <- sum(dfr.i$wgt.h)
  # Leave-one-out-Schätzer von Y_k
  for (k in 1:n){
    # Regressionsformel
    fm <- paste0(av,"~",paste0(maineff,collapse="+"))
    fm <- as.formula(fm)
    # Regressionsanalyse ohne Beitrag von Schule k
    dfr.i0 <- dfr.i[-k,]
    mod.k <- lm(formula=fm,data=dfr.i0,weights=dfr.i0$wgt.h)
    # Erwartungswert anhand Kovariaten der Schule k berechnen
    pred.k <- predict(mod.k, dfr.i)[k]
    dfr0.i[k,paste0( "h_",h) ] <- pred.k
}}
# Erwartungswerte auf Basis verschiedener h-Werte
dfr1 <- data.frame("idschool.i"=dfr.i$idschool.i[1],"h"=H )

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 12: Nichtparametrische Regression
# Berechnung des Kreuzvalidierungskriteriums nach h in H
#

library(kerdiest)
hAL <- kerdiest::ALbw("n",dfr.i$dist) # Plug-in Bandweite
dfr.i$cross.h <- hAL
```

```
dfr.i$crosswgt <- dnorm( sqrt(dfr.i$dist), mean=0, sd = sqrt(hAL) )
# Kreuzvalidierungskriterium CVh
vh <- grep("h_",colnames(dfr0.i),value=TRUE)
for (ll in 1:V1){
  dfr1[ll,"CVh"] <- sum( (dfr0.i[,av] - dfr0.i[,vh[ll]])^2 *
                           dfr.i$crosswgt) / n}

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 13: Nichtparametrische Regression
# Bestimmung optimales Wertes von h (h.min)
#

dfr1$min.h.index <- 0
ind <-  which.min( dfr1$CVh )
dfr1$min.h.index[ind ] <- 1
dfr1$h.min <- dfr1$h[ind]

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 14: Nichtparametrische Regression
# Kleinste Quadratsumme der Schätzfehler
#

dfr1$CVhmin <- dfr1[ ind , "CVh" ]

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 15: Nichtparametrische Regression
# Effizienzsteigerung berechnen
#

dfr1$eff_gain <-  100 * ( dfr1[V1,"CVh"] / dfr1$CVhmin[1] - 1 )

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Listing 16: Nichtparametrische Regression
# Durchführung der nichtparametrischen Regression bei h=h.min
#

h <- dfr1$h.min[1]  # h.min
dfr.i$wgt.h <- dnorm(sqrt(dfr.i$dist),sd=sqrt(h))/
  dnorm(0,sd= sqrt(h)) # w_{ik} bei h.min
dfr.i0 <- dfr.i
# Lokale Regression
mod.ii <- lm(formula=fm,data=dfr.i0,weights=dfr.i0$wgt.h)
# Kovariaten Schule i
predM <- data.frame(dfr.i[1,paste0(maineff,".i")])
names(predM) <- maineff
pred.ii <- predict(mod.ii, predM) # Schätzwert Schule i
dat[match(dfr1$idschool.i[1],dat$idschool), "expTWLE.np"] <- pred.ii

## ------------------------------------------------------------
## Abschnitt 9.2.5, Umsetzung in R, Ergänzung zum Buch
## ------------------------------------------------------------

# Korrelationen zwischen Haupteffekten
```

```
cor(dat[,maineff]) # gesamt
# Pro Stratum
for(s in 1:4) print(cor(dat[which(dat$stratum == s),maineff]))

# ------------------------------------------------------------
# Abschnitt 9.2.5.2, Ergänzung zum Buch
# OLS-Regression
#

# Modellgleichung nur mit Haupteffekten
fm.ols1 <- paste0("TWLE ~ ",paste(maineff,collapse=" + "))
fm.ols1 <- as.formula(fm.ols1)

# Modellgleichung mit Haupteffekten ohne zgroesse
fm.ols1a <- paste0("TWLE ~ ",paste(setdiff(maineff,c("zgroesse")),
                                   collapse=" + "))
fm.ols1a <- as.formula(fm.ols1a)

# Modellgleichung mit Haupt- und Interaktionseffekten
fm.ols2 <- paste0("TWLE ~ ",paste(alleff,collapse=" + "))
fm.ols2 <- as.formula(fm.ols2)

# Ergebnistabelle über 4 Strata hinweg vorbereiten
tab1 <- data.frame("Variable"=c("(Intercept)",maineff))
tab2 <- data.frame("Variable"=c("(Intercept)",alleff))

# Durchführung: Schleife über vier Strata
for(st in 1:4){
  # st <- 4
  # Position Schulen des Stratums st im Datensatz
  pos <- which(dat$stratum == st)

  #---------------------------------
  # OLS-Modell 1

  # Durchführung
  ols.mod1 <- lm(formula = fm.ols1,data = dat[pos,])
  ols.mod1a <- lm(formula = fm.ols1a,data = dat[pos,])

  # Modellergebnisse anzeigen
  summary(ols.mod1)
  summary(ols.mod1a)

  # Erwartungswerte der Schulen
  dat$expTWLE.OLS1[pos] <- fitted(ols.mod1)

  # Ergebnisse in Tabelle speichern
  par <- summary(ols.mod1)
  tab.s <- data.frame(par$coef,R2=par$r.squared,R2.adj=par$adj.r.squared)
  names(tab.s) <- paste0("stratum",st,
                         c("_coef","_SE","_t","_p","_R2","_R2.adj"))
  tab1 <- cbind(tab1, tab.s)
```

```
  # Durchführung OLS-Modell 2
  ols.mod2 <- lm(formula = fm.ols2,data = dat[pos,])

  # Modellergebnisse anzeigen
  summary(ols.mod2)

  # Erwartungswerte der Schulen
  dat$expTWLE.OLS2[pos] <- fitted(ols.mod2)

  # Ergebnisse in Tabelle speichern
  par <- summary(ols.mod2)
  tab.s <- data.frame(par$coef,R2=par$r.squared,R2.adj=par$adj.r.squared)
  names(tab.s) <- paste0("stratum",st,
                         c("_coef","_SE","_t","_p","_R2","_R2.adj"))
  tab2 <- cbind(tab2, tab.s)

}

# Daten Schule 1196 ansehen
dat[which(dat$idschool == 1196),]

# Schätzwerte nach ols.mod1 und ols.mod2 vergleichen
summary(abs(dat$expTWLE.OLS1 - dat$expTWLE.OLS2))
cor.test(dat$expTWLE.OLS1,dat$expTWLE.OLS2)

# Grafische Darstellung des Vergleich (Schule 1196 rot markiert)
plot(dat$expTWLE.OLS1,dat$expTWLE.OLS2,xlim=c(380,650),ylim=c(380,650),
     col=1*(dat$idschool == 1196)+1,pch=15*(dat$idschool == 1196)+1)
abline(a=0,b=1)

# ------------------------------------------------------------
# Abschnitt 9.2.5.3, Ergänzung zum Buch
# Lasso-Regression
#

library(glmnet)

# Variablen für Erwartungswerte
dat$expTWLE.Lasso2 <- dat$expTWLE.Lasso1 <- NA

# Tabelle für Modellergebnisse
tab3 <- data.frame("Variable"=c("(Intercept)",maineff))
tab4 <- data.frame("Variable"=c("(Intercept)",alleff))

for(st in 1:4){
  # st <- 4

  # Position Schulen des Stratums st im Datensatz
  pos <- which(dat$stratum == st)

  #-------------------------------------------------------------#
  # Lasso-Regression mit den Haupteffekten
```

```
# Kovariatenmatrix
Z <- as.matrix(dat[pos,maineff])
# Abhängige Variable
Y <- dat$TWLE[pos]

# Kreuzvalidierung: Teilmengen definieren
nid <- floor(length(pos)/3)
# Schulen zu Teilmengen zuordnen
foldid <- rep(c(1:nid),3,length.out=length(pos))

# Regression
lasso.mod1 <- cv.glmnet(x=Z,y=Y,alpha = 1, foldid = foldid)

# Ergebnisse ansehen
print(lasso.mod1)

# Lasso-Koeffizienten bei lambda.min
print(lasso.beta <- coef(lasso.mod1,s="lambda.min"))

# Erwartungswerte der Schulen
lasso.pred1 <- predict(lasso.mod1,newx = Z,s="lambda.min")
dat$expTWLE.Lasso1[pos] <- as.vector(lasso.pred1)

# R2 bestimmen
varY <- var(dat$TWLE[pos])
varY.lasso.mod1 <- var(dat$expTWLE.Lasso1[pos])
print(R2.lasso.mod1 <- varY.lasso.mod1/varY)

# Ergebnistabelle
vv <- paste0("coef.stratum",st); tab3[,vv] <- NA
tab3[lasso.beta@i+1,vv] <- lasso.beta@x
vv <- paste0("lambda.stratum",st); tab3[,vv] <- lasso.mod1$lambda.min
vv <- paste0("R2.stratum",st); tab3[,vv] <- R2.lasso.mod1

#-----------------------------------------------------------#
# Lasso-Regression mit Haupt- und Interaktionseffekten

# Kovariatenmatrix
Z <- as.matrix(dat[pos,alleff])

# Regression
lasso.mod2 <- cv.glmnet(x=Z,y=Y,alpha = 1, foldid = foldid)

# Ergebnisausdruck
print(lasso.mod2)

# Lasso-Koeffizienten bei lambda.min
print(lasso.beta <- coef(lasso.mod2,s="lambda.min"))

# Erwartungswerte der Schulen
lasso.pred2 <- predict(lasso.mod2,newx = Z,s="lambda.min")
dat$expTWLE.Lasso2[pos] <- as.vector(lasso.pred2)
```

```
  # R2 bestimmen
  varY.lasso.mod2 <- var(dat$expTWLE.Lasso2[pos])
  R2.lasso.mod2 <- varY.lasso.mod2/varY
  R2.lasso.mod2

  # Ergebnistabelle
  vv <- paste0("coef.stratum",st); tab4[,vv] <- NA
  tab4[lasso.beta@i+1,vv] <- lasso.beta@x
  vv <- paste0("lambda.stratum",st); tab4[,vv] <- lasso.mod2$lambda.min
  vv <- paste0("R2.stratum",st); tab4[,vv] <- R2.lasso.mod2


}

# Regressionresiduen = Schätzung von SChul- und Unterrichtseffekt
dat$resTWLE.Lasso1 <- dat$TWLE - dat$expTWLE.Lasso1
dat$resTWLE.Lasso2 <- dat$TWLE - dat$expTWLE.Lasso2

# ------------------------------------------------------------
# Abschnitt 9.2.5.4, Ergänzung zum Buch
# Nichtparametrische Regression
#

#
# Achtung: Der nachfolgende Algorithmus benötigt viel Zeit!
#

av <- "TWLE" # Abhängige Variable
dfr3 <- NULL # Ergebnistabelle

# Variable für Leistungsschätzwerte

# Schleife über 4 Strata
for(st in 1:4){
  # st <- 1
  pos <- which(dat$stratum == st)
  N <- length(pos)
  schools <- dat$idschool[pos]

  ###
  # Distanzmatrix dfr für alle Schulen im Stratum erstellen
  dfr <- NULL

  for (i in 1:N){
    # i <- 1
    # Teildatensatz von Schule i
    dat.i <- dat[pos[i],c("idschool","TWLE",maineff)]
    # Daten der Vergleichsgruppe
    dat.vgl <- dat[pos[-i],c("idschool","TWLE",maineff)]
    # Variablennamen von dat.vgl umbenennen
    # names(dat.vgl) <- paste0("vgl.",names(dat.vgl))
    # Variablennamen von dat.i umbenennen
    names(dat.i) <- paste0(names(dat.i),".i")
```

```
  # Daten zusammenfügen
  index.vgl <- match(dat.vgl$idschool,schools)
  dfr.i <- data.frame("index.i"=i,dat.i,
                      "index.vgl"=index.vgl,dat.vgl,
                      row.names=NULL)

  # Distanz zur i
  dfr.i$dist <- 0
  gi <- c(1,1,1,1)
  for(ii in 1:length(maineff)){
    vv <- maineff[ii]
    pair.vv <- grep(vv, names(dfr.i), value=T)
    dist.vv <- gi[ii]*((dfr.i[,pair.vv[1]]-dfr.i[,pair.vv[2]])^2)
    dfr.i$dist <- dfr.i$dist + dist.vv
  }

  print(i) ; flush.console()
  dfr <- rbind( dfr , dfr.i )
}

dfr1 <- index.dataframe( dfr , systime=TRUE )

###
# h-Auswahl und Nichtparametrische Regression pro Schule i
dfr1.list <- list()
for (i in 1:N){
  # i <- 1
  dfr.i <- dfr[ dfr$index.i == i , ]
  n <- nrow(dfr.i)

  # Startwertliste für h initiieren
  d.dist <- max(dfr.i$dist)-min(dfr.i$dist)
  H <- c(seq(d.dist/100,d.dist,length=30),100000)
  V1 <- length(H) # Anzahl der Startwerte in H

  # Startwerte: Summe von w_ik
  sumw <- 0*H
  dfr0.i <- dfr.i[,c("idschool",av)]
  # Schleife über alle h-Werte
  for (ll in 1:V1 ){
    h <- H[ll]
    # Gewicht w_ik bei h
    dfr.i$wgt.h <- dnorm(sqrt(dfr.i$dist), mean=0, sd=sqrt(h))
    # Summe von w_ik bei h
    sumw[which(H==h)] <- sum(dfr.i$wgt.h)
    # Leave-one-out-Schätzer von Y_k
    for (k in 1:n){
      # Regressionsformel
      fm <- paste0(av,"~",paste0(maineff,collapse="+"))
      fm <- as.formula(fm)
      # Regressionsanalyse ohne Beitrag von Schule k
      dfr.i0 <- dfr.i[-k,]
```

```
    mod.k <- lm(formula=fm,data=dfr.i0,weights=dfr.i0$wgt.h)
    # Erwartungswert anhand Kovariaten der Schule k berechnen
    pred.k <- predict(mod.k, dfr.i)[k]
    dfr0.i[k,paste0( "h_",h) ] <- pred.k
  }
  print(paste0("i=",i,", h_",ll))
}
# Erwartungswerte auf Basis verschiedener h-Werte
dfr1 <- data.frame("idschool.i"=dfr.i$idschool.i[1],"h"=H )

# Berechnung des Kreuzvalidierungskriteriums
library(kerdiest)
hAL <- kerdiest::ALbw("n",dfr.i$dist) # Plug-in Bandbreite nach Altman und
                                      # Leger
name <- paste0( "bandwidth_choice_school" , dfr.i$idschool.i[1] ,
                "_cross.h_" , round2(hAL,1) )
# Regressionsgewichte auf Basis cross.h
dfr.i$cross.h <- hAL
dfr.i$crosswgt <- dnorm( sqrt(dfr.i$dist), mean=0, sd = sqrt(hAL) )

dfr.i <- index.dataframe( dfr.i , systime=TRUE )

# Kreuzvalidierungskriterium CVh
vh <- grep("h_",colnames(dfr0.i),value=TRUE)
for (ll in 1:V1){
  # ll <- 5
  dfr1[ll,"CVh"] <- sum( (dfr0.i[,av] - dfr0.i[,vh[ll]])^2 *
                           dfr.i$crosswgt) / n
  print(ll)
}

# Bestimmung h.min
dfr1$min.h.index <- 0
ind <-  which.min( dfr1$CVh )
dfr1$min.h.index[ind ] <- 1
dfr1$h.min <- dfr1$h[ind]
# Kleinste Quadratsumme der Schätzfehler
dfr1$CVhmin <- dfr1[ ind , "CVh" ]

# Effizienzsteigerung berechnen
dfr1$eff_gain <-  100 * ( dfr1[V1,"CVh"] / dfr1$CVhmin[1] - 1 )

# h auswählen
h <- dfr1$h.min[1]

# Gewichte anhand h berechnen
dfr.i$wgt.h <- dnorm( sqrt( dfr.i$dist ) , sd = sqrt( h) ) /
               dnorm( 0 , sd = sqrt( h) )
dfr.i0 <- dfr.i
mod.ii <- lm(formula = fm,data = dfr.i0,weights = dfr.i0$wgt.h)

# Leistungsschätzwerte berechnen
predM <- data.frame(dfr.i[1,paste0(maineff,".i")])
```

```
    names(predM) <- maineff

    pred.ii <- predict( mod.ii ,  predM )
    dfr1$fitted_np <- pred.ii
    dfr1$h.min_sumwgt <- sum( dfr.i0$wgt.h )
    dfr1$h_sumwgt  <- sumw

    # Leistungsschätzwerte zum Datensatz hinzufügen
    dat$expTWLE.np[match(dfr1$idschool.i[1],dat$idschool)] <- pred.ii
    dfr1.list[[i]] <- dfr1
  }

  ###
  # Ergebnisse im Stratum st zusammenfassen
  dfr2 <- NULL

  for(i in 1:length(dfr1.list)){
    dat.ff <- dfr1.list[[i]]
    dfr2.ff <- dat.ff[1,c("idschool.i","h.min","fitted_np","h.min_sumwgt",
                          "CVhmin","eff_gain")]
    dfr2.ff$CVlinreg <- dat.ff[V1,"CVh"]
    names(dfr2.ff) <- c("idschool","h.min","fitted_np","h.min_sumwgt",
                        "CVhmin","eff_gain","CVlinreg")
    dfr2 <- rbind(dfr2, dfr2.ff)
    print(i)
  }

  #-----------------------------------------------------##
  # R2 berechnen
  varY <- var(dat$TWLE[pos])
  varY.np <- var(dat$expTWLE.np[pos])
  dfr2$R2.np <- varY.np/varY

  #-----------------------------------------------------##
  # Zur Gesamtergebnistabelle
  dfr3 <- rbind(dfr3,cbind("Stratum"=st,dfr2))

}

# Effizienz der NP-Regression gegenüber OLS-Regression
summary(dfr3$eff_gain)
table(dfr3$eff_gain > 5)
table(dfr3$eff_gain > 10)
table(dfr3$eff_gain > 20)

# Regressionsresiduen
dat$resTWLE.np <- dat$TWLE - dat$expTWLE.np

## ---------------------------------------------------------------
## Abschnitt 9.2.6, Ergänzung zum Buch
## Ergebnisse im Vergleich
## ---------------------------------------------------------------
```

```
# Output-Variablen
out <- grep("expTWLE",names(dat),value=T)
lt <- length(out)

# Korrelationsmatrix
tab <- tab1 <- as.matrix(round2(cor(dat[,out]),3))

# Varianzmatrix
tab2 <- as.matrix(round2(sqrt(var(dat[,out])),1))

tab3 <- matrix(NA,lt,lt)
# Differenzmatrix
for(ii in 1:(lt-1))
  for(jj in (ii+1):lt) tab3[ii,jj] <- round2(mean(abs(dat[,out[jj]] -
                                                      dat[,out[ii]])),1)

tab4 <- matrix(NA,lt,lt)
# Differenzmatrix
for(ii in 1:(lt-1))
  for(jj in (ii+1):lt) tab4[ii,jj] <- round2(sd(abs(dat[,out[jj]] -
                                                    dat[,out[ii]])),1)

# Ergebnistabelle
diag(tab) <- diag(tab2)
tab[upper.tri(tab)] <- tab3[upper.tri(tab3)]

# R2 Gesamt
varY <- var(dat$TWLE)
varexp.OLS1 <- var(dat$expTWLE.OLS1); R2.OLS1 <- varexp.OLS1/varY
varexp.OLS2 <- var(dat$expTWLE.OLS2); R2.OLS2 <- varexp.OLS2/varY
varexp.Lasso1 <- var(dat$expTWLE.Lasso1); R2.Lasso1 <- varexp.Lasso1/varY
varexp.Lasso2 <- var(dat$expTWLE.Lasso2); R2.Lasso2 <- varexp.Lasso2/varY
varexp.np <- var(dat$expTWLE.np); R2.np <- varexp.np/varY
R2 <- c(R2.OLS1,R2.OLS2,R2.Lasso1,R2.Lasso2,R2.np)
tab <- cbind(tab,R2)

# R2 pro Stratum
dat0 <- dat
for(st in 1:4){
  # st <- 1
  dat <- dat0[which(dat0$stratum == st),]
  varY <- var(dat$TWLE)
  varexp.OLS1 <- var(dat$expTWLE.OLS1); R2.OLS1 <- varexp.OLS1/varY
  varexp.OLS2 <- var(dat$expTWLE.OLS2); R2.OLS2 <- varexp.OLS2/varY
  varexp.Lasso1 <- var(dat$expTWLE.Lasso1); R2.Lasso1 <- varexp.Lasso1/varY
  varexp.Lasso2 <- var(dat$expTWLE.Lasso2); R2.Lasso2 <- varexp.Lasso2/varY
  varexp.np <- var(dat$expTWLE.np); R2.np <- varexp.np/varY
  R2 <- c(R2.OLS1,R2.OLS2,R2.Lasso1,R2.Lasso2,R2.np)
  tab <- cbind(tab,R2)
}

colnames(tab)[7:10] <- paste0("R2_stratum",1:4)
```

```
## ------------------------------------------------------------
## Abschnitt 9.2.7, Berücksichtigung der Schätzfehler
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 9.2.7, Listing 17: Bestimmung des Erwartungsbereichs
#

vv <- "expTWLE.OLS1" # Variablenname
mm <- "OLS1" # Kurzname des Modells
dfr <- NULL # Ergebnistabelle
# Schleife über alle möglichen Breite von 10 bis 60
for(w in 10:60){
  # Variablen für Ergebnisse pro w
  var <- paste0(mm,".pos.eb",w) # Position der Schule
  var.low <- paste0(mm,".eblow",w) # Untere Grenze des EBs
  var.upp <- paste0(mm,".ebupp",w) # Obere Grenze des EBs
  # Berechnen
  dat[,var.low] <- dat[,vv]-w/2 # Untere Grenze des EBs
  dat[,var.upp] <- dat[,vv]+w/2 # Obere Grenze des EBs
  # Position: -1=unterhalb, 0=innerhalb, 1=oberhalb des EBs
  dat[,var] <- -1*(dat$TWLE < dat[,var.low]) + 1*(dat$TWLE > dat[,var.upp])
  # Verteilung der Schulpositionen
  tmp <- data.frame(t(matrix(prop.table(table(dat[,var])))))
  names(tmp) <- c("unterhalb","innerhalb","oberhalb")
  tmp <- data.frame("ModellxBereich"=var,tmp); dfr <- rbind(dfr,tmp) }

# Abweichung zur Wunschverteilung 25-50-25
dfr1 <- dfr
dfr1[,c(2,4)] <- (dfr1[,c(2,4)] - .25)^2
dfr1[,3] <- (dfr1[,3] - .5)^2
dfr1$sumquare <- rowSums(dfr1[,-1])
# Auswahl markieren
dfr$Auswahl <- 1*(dfr1$sumquare == min(dfr1$sumquare) )

# ------------------------------------------------------------
# Abschnitt 9.2.7, Ergänzung zum Buch
# Bestimmung des Erwartungsbereichs
#

# Ergebnisse aller Schulen werden aus Ursprungsdatensatz geladen.
dat <- datenKapitel09

# Liste der Erwartungswerte-Variablen
exp.vars <- grep("expTWLE",names(dat),value=T)
# Modellnamen
m.vars <- gsub("expTWLE.","",exp.vars, fixed = TRUE)

# Liste der Ergebnistabelle
list0 <- list()

# Ergebnisse
tab.erg <- NULL
```

```
# Schleife über alle Erwartungswerte aller Modelle
for(ii in 1:length(exp.vars)){
  # ii <- 1
  vv <- exp.vars[ii]
  mm <- m.vars[ii]

  # Ergebnistabelle
  dfr <- NULL

  # Schleife über alle möglichen Breite von 10 bis 60
  for(w in 10:60){
    # eb <- 10
    var <- paste0(mm,".pos.eb",w) # Position der Schule
    var.low <- paste0(mm,".eblow",w) # Untere Grenze des EBs
    var.upp <- paste0(mm,".ebupp",w) # Obere Grenze des EBs
    # Untere Grenze des EBs = Erwartungswert - w/2
    dat[,var.low] <- dat[,vv]-w/2
    # Obere Grenze des EBs = Erwartungswert + w/2
    dat[,var.upp] <- dat[,vv]+w/2
    # Position der Schule bestimmen
    # -1 = unterhalb, 0 = innterhalb, 1 = oberhalb des EBs
    dat[,var] <- -1*(dat$TWLE < dat[,var.low]) + 1*(dat$TWLE > dat[,var.upp])
    # Verteilung der Positionen
    tmp <- data.frame(t(matrix(prop.table(table(dat[,var])))))
    names(tmp) <- c("unterhalb","innerhalb","oberhalb")
    tmp <- data.frame("ModellxBereich"=var,tmp)
    dfr <- rbind(dfr,tmp)
  }

  # Vergleich mit Wunschverteilung 25-50-25
  dfr1 <- dfr
  dfr1[,c(2,4)] <- (dfr1[,c(2,4)] - .25)^2
  dfr1[,3] <- (dfr1[,3] - .5)^2
  dfr1$sumquare <- rowSums(dfr1[,-1])
  # Auswahl markieren
  dfr$Auswahl <- 1*(dfr1$sumquare == min(dfr1$sumquare) )

  # Zum Liste hinzufügen
  list0[[ii]] <- dfr
  print(dfr[which(dfr$Auswahl == 1),])
  tab.erg <- rbind(tab.erg, dfr[which(dfr$Auswahl == 1),])

}

# Nur gewählte Ergebnisse im Datensatz beibehalten
all.vars <- grep("eb",names(dat),value=T)
# Untere und Obere Grenze mit speichern
eb.vars <- tab.erg[,1]
low.vars <- gsub("pos.eb","eblow",eb.vars)
upp.vars <- gsub("pos.eb","ebupp",eb.vars)
del.vars <- setdiff(all.vars, c(eb.vars,low.vars,upp.vars))
dat <- dat[,-match(del.vars,names(dat))]
```

```
## ------------------------------------------------------------
## Appendix: Abbildungen
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abbildung 9.4
#

# Koeffizienten bei der ersten 50 lambdas ausdrucken
# Stratum 4

lambda <- lasso.mod2$lambda[1:50]
a <- round2(lambda,2)
a1 <- a[order(a)]
L <- length(a)

dfr <- NULL

for(ll in 1:L){
  dfr.ll <- as.matrix(coef(lasso.mod2,newx = Z,s=a[ll] ))
  colnames(dfr.ll) <- paste0("a_",ll)
  dfr.ll <- data.frame("coef"=rownames(dfr.ll),dfr.ll)
  rownames(dfr.ll) <- NULL
  if(ll == 1) dfr <- dfr.ll else dfr <- merge(dfr, dfr.ll)
}

# Ohne Intercept
dfr <- dfr[-1,]
rownames(dfr) <- 1:nrow(dfr)

cl <- colors()
cl <- grep("grey",cl,value=T)

# Umgekehrte Reihenfolge
dfr1 <- dfr
for(x in 2:(L+1)) {dfr1[,x] <- dfr[,(L+3)-x];
names(dfr1)[x] <- names(dfr)[(L+3)-x]}

###
plot(x = log(a), y = rep(0,L), xlim = rev(range(log(a))), ylim=c(-20,22),
     type = "l", xaxt ="n", xlab = expression(paste(lambda)),
     ylab="Geschätzte Regressionskoeffizienten")
axis(1, at=log(a), labels=a,cex=1)

tmp <- nrow(dfr)
for(ll in 1:tmp){
  # ll <- 1
  lines(x=log(a),y=dfr[ll,2:(L+1)],type="l",pch=15-ll,col=cl[15-ll])
  points(x=log(a),y=dfr[ll,2:(L+1)],type="p",pch=15-ll)
  legend(x=2.8-0.7*(ll>tmp/2),y=25-2*(ifelse(ll>7,ll-7,ll)),
         legend =dfr$coef[ll],pch=15-ll,bty="n",cex=0.9)
```

```
}

# Kennzeichung der gewählten lambda
v <- log(lasso.mod2$lambda.min)
lab2 <- expression(paste("ausgewähltes ",lambda," = .43"))
text(x=v+0.6,y=-8,labels=lab2)

abline(v = v,lty=2,cex=1.2)

# ------------------------------------------------------------
# Abbildung 9.5
# Auswahl Lambda anhand min(cvm)
#

xlab <- expression(paste(lambda))
plot(lasso.mod2, xlim = rev(range(log(lambda))),
     ylim=c(550,1300),xlab=xlab,xaxt ="n",
     ylab = "Mittleres Fehlerquadrat der Kreuzvalidierung (cvm)",
     font.main=1,cex.main=1)
axis(1, at=log(a), labels=a,cex=1)

lab1 <- expression(paste(lambda," bei min(cvm)"))
text(x=log(lasso.mod2$lambda.min)+0.5,y=max(lasso.mod2$cvm)-50,
     labels=lab1,cex=1)

lab2 <- expression(paste("(ausgewähltes ",lambda," = .43)"))
text(x=log(lasso.mod2$lambda.min)+0.6,y=max(lasso.mod2$cvm)-100,
     labels=lab2,cex=1)

abline(v = log(lasso.mod2$lambda.min),lty=2)

text(x=log(lasso.mod2$lambda.min)-0.3,y = min(lasso.mod2$cvm)-30,
     labels="min(cvm)",cex=1 )
abline(h = min(lasso.mod2$cvm),lty=2)

text <- expression(paste("Anzahl der Nicht-null-Koeffizienten (",
                         lambda," entsprechend)"))
mtext(text=text,side=3,line=3)


# ------------------------------------------------------------
# Abbildung 9.6
# Rohwert-Schätzwert Schule 1196 & 1217 im Vergleich
#

id <- c(1196, 1217)
par(mai=c(1.2,3,1,.5))
plot(x=rep(NA,2),y=c(1:2),xlim=c(470,610),yaxt ="n",type="l",
     xlab="Erwartungswerte je nach Modell und Schulleistung",ylab="")
legend <- c("Schulleistung (TWLE)",paste0("", c("OLS1","OLS2","Lasso1",
                                                "Lasso2","NP"),
                                          "-Modell"))
axis(2, at=c(seq(1,1.4,0.08),seq(1.6,2,0.08)), las=1,cex=0.7,
```

```
     labels=rep(legend,2))
text <- paste0("Schule ",id)
mtext(text=text,side=2,at = c(1.2,1.8),line = 10)

exp.vars <- c("TWLE",
              paste0("expTWLE.", c("OLS1","OLS2","Lasso1","Lasso2","np")))

pch = c(19, 0,3,2,4,5)
ii <- 1
col = c("grey", rep("lightgrey",5))
for(vv in exp.vars){
  # vv <- "TWLE"
  x <- dat0[which(dat0$idschool %in% id),vv]
  abline(h = c(0.92+ii*0.08,1.52+ii*0.08), lty=1+1*(ii>1),col=col[ii])
  points(x=x,y=c(0.92+ii*0.08,1.52+ii*0.08),type="p",pch=pch[ii])
  ii <- ii + 1
}

## End(Not run)
```

`Kapitel 10` *Kapitel 10: Reporting und Analysen*

Description

Das ist die Nutzerseite zum Kapitel 10, *Reporting und Analysen*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenefalls erweitert.

Author(s)

Michael Bruneforth, Konrad Oberwimmer, Alexander Robitzsch

References

Bruneforth, M., Oberwimmer, K. & Robitzsch, A. (2016). Reporting und Analysen. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 333–362). Wien: facultas.

See Also

Zu `datenKapitel10`, den im Kapitel verwendeten Daten.
Zurück zu `Kapitel 9`, Fairer Vergleich in der Rückmeldung.
Zu `Kapitel 11`, Aspekte der Validierung.
Zur `Übersicht`.

Examples

```
## Not run:
library(BIFIEsurvey)
library(matrixStats)

data(datenKapitel10)
dat <- datenKapitel10$dat
dat.roh <- datenKapitel10$dat.roh
dat.schule <- datenKapitel10$dat.schule
dat.schule.roh <- datenKapitel10$dat.schule.roh

## ----------------------------------------------------------------
## Abschnitt 10.4.1: Datenbasis
## ----------------------------------------------------------------

# ----------------------------------------------------------------
# Abschnitt 10.4.1 a, Ergänzung zum Buch
# Herunterladen, entpacken und setzen des Arbeitsspeichers
#

# setwd(dir = ".../DatenKapitel10")

# ----------------------------------------------------------------
# Abschnitt 10.4.1, Listing 1: Einlesen der Schülerdaten
#

# Anlegen eines leeren Listenobjektes für Schülerdaten
dat <- list()

# Vektor mit Liste der Dateinamen für Schülerdaten
dateinamen <- paste0("e8pv__schueler_imp_",1:10,".csv")
# Schleife zum Einlesen der Daten, die in die Listenobjekte
# abgelegt werden
for (ii in 1:10) {
  schueler_dfr<-read.csv2(file = dateinamen[[ii]])
  dat[[ii]] <- schueler_dfr
}
# Überprüfen des Listenobjektes und der eingelesenen Daten
str(dat)

# Rohdaten als Datenmatrix einlesen
dat.roh <- read.csv2(file = "e8pv__schueler_raw.csv")

# ----------------------------------------------------------------
# Abschnitt 10.4.1, Listing 1a: Ergänzung zum Buch
# Einlesen der Schulendaten
#

# Anlegen eines leeren Listenobjektes für Schuldaten
dat.schule <- list()

# Vektor mit Liste der Dateinamen für Schuldaten
```

```
dateinamen <- paste0("e8pv__schule_imp_",1:10,".csv")
# Schleife zum Einlesen der Daten, die in die Listenobjekte
# abgelegt werden
for (ii in 1:10) {
  schule_dfr<-read.csv2(file = dateinamen[[ii]])
  dat.schule[[ii]] <- schule_dfr
}
# Überprüfen des Listenobjektes und der eingelesenen Daten
str(dat.schule)

#Rohdaten als Datenmatrix einlesen
dat.schule.roh <- read.csv2(file = "e8pv__schule_raw.csv")

## ------------------------------------------------------------
## Abschnitt 10.4.2: Merging verschiedener Ebenen
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.4.2, Listing 1
#

for (i in 1:10) {
   dat[[i]] <- merge(dat[[i]],dat.schule[[i]],
                     by = "idschool",all.x = TRUE)
}

# ------------------------------------------------------------
# Abschnitt 10.4.2, Listing 2

for (i in 1:10) {
   dat.agg <- aggregate(dat[[i]][,c("HISEI","E8RPV")],
                        by = list(idschool = dat[[i]]$idschool),
                        FUN = mean,na.rm = TRUE)
   dat.schule[[i]] <- merge(dat.schule[[i]],dat.agg,
                            by="idschool",all.x = TRUE)
}

## ------------------------------------------------------------
## Abschnitt 10.4.3: Erzeugen von BIFIEdata-Objekten
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.4.3, a: Ergänzung zum Buch
# Einlesen der Replikationsgewichte
#

# Zwischenspeichern des Schülerdatensatzes
dat.tmp <- dat

# Daten aus Large-Scale Assessments können mit replicate weights
#  abgespeichert werden (z.B. PISA) oder mit Informationen zu den
#  Jackknifezonen und -gewichten (z.B. PIRLS). In diesem Beispiel
#  werden beide Methoden vorgestellt, daher wird die Gewichtungs-
```

```
#  information in beiden Formen eingelesen: mit replicate weights
#  im Datensatz dat1; mit Replikationsdesign im Datensatz dat2.

# replicate weights für Schüler/innen als Datenmatrix einlesen
dat.repwgts <- read.csv2(file = "e8__schueler_repwgts.csv")
# replicate weights an Schülerdaten mergen
for (ii in 1:10) {
  dat[[ii]]<-merge(x = dat[[ii]],y = dat.repwgts,
                   by = c("idschool","idstud"))
}

# Jackknifezonen und -gewichte für Schulen als Datenmatrix einlesen
dat2 <- list()
dat.schule.jk <- read.csv2(file = "e8__schule_jkzones.csv")
# Jackknifezonen und -gewichte an schülerdaten mergen
for (ii in 1:10) {
  dat2[[ii]]<-merge(x = dat.tmp[[ii]],y = dat.schule.jk,
                   by = "idschool")
}

# ------------------------------------------------------------
# Abschnitt 10.4.3, b: Ergänzung zum Buch
# Kontrolle der Sortierung
#

# Die Observationen in den 10 Imputationen muessen gleich sortiert
# sein. Dies wir zur Sicherheit getestet.
for (i in 2:10) {
  if (sum(dat[[1]]$idstud!=dat[[i]]$idstud )>0)
      stop("Imputationsdatensätze nicht gleich sortiert!")
}

# ------------------------------------------------------------
# Abschnitt 10.4.3, c: Ergänzung zum Buch
# Verwendung des R-Datenobjekts
#

dat <- datenKapitel10$dat

# ------------------------------------------------------------
# Abschnitt 10.4.3, Listing 1: Übernahme der Gewichte aus
# Datenmatrix
#

wgtstud <- dat[[1]]$wgtstud
repwgtsvar <- grep("^w_fstr",colnames(dat[[1]]))
repwgts <- dat[[1]][,repwgtsvar]
dat <- BIFIE.data(data.list = dat,wgt = wgtstud,
                  wgtrep = repwgts,fayfac = 1,
                  cdata = TRUE)
summary(dat)

# ------------------------------------------------------------
```

```
# Abschnitt 10.4.3, Listing 2: Erzeugung der Gewichte aus
# Replikationsdesign
#

dat2 <- BIFIE.data.jack(data = dat2,wgt = "wgtstud",
                        jktype = "JK_GROUP",
                        jkzone = "jkzone",
                        jkrep = "jkrep",
                        fayfac = 1)
summary(dat2)


# ------------------------------------------------------------
# Abschnitt 10.4.3, Listing 3: Univariate Statistik Reading
#

res.univar <- BIFIE.univar(BIFIEobj = dat,
                           vars = c("E8RPV"),
                           group = "Strata")
summary(res.univar)
res2.univar <- BIFIE.univar(BIFIEobj = dat2,
                           vars = c("E8RPV"),
                           group = "Strata")
summary(res2.univar)

## ------------------------------------------------------------
## Abschnitt 10.4.4: Rekodierung und Transformation von
##                   Variablen
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.4.4, Listing 1: Neue Variable GERSER mit
# BIFIE.data.transform
#

transform.formula <- as.formula(
   "~ 0 + I(cut(E8RPV,breaks = c(0,406,575,1000),labels = FALSE))"
   )
dat <- BIFIE.data.transform(dat,transform.formula,
                            varnames.new = "GERSER")
res.freq <- BIFIE.freq(BIFIEobj = dat,vars = c("GERSER"))
summary(res.freq)

# ------------------------------------------------------------
# Abschnitt 10.4.4, Listing 2: Zwei neue Variablen PVERfit und
# PVERres mit BIFIE.data.transform
#

transform.formula <- as.formula(
   "~ 0 + I(fitted(lm(E8RPV ~ HISEI + female))) +
          I(residuals(lm(E8RPV ~ HISEI + female)))"
   )
dat <- BIFIE.data.transform(dat,transform.formula,
```

```
                            varnames.new = c("PVERfit","PVERres"))
res.univar <- BIFIE.univar(BIFIEobj = dat,
                           vars = c("PVERfit","PVERres"))
summary(res.univar)

## ------------------------------------------------------------
## Abschnitt 10.4.5: Berechnung von Kenngroessen und deren
##                   Standardfehlern
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.4.5, Listing 1: Anwenderfunktion
#
library(matrixStats)

anwenderfct.weightedMad <- function(X,w)
{
  # Die Funktion weightedMad wird auf jede Spalte der
  # übergebenen Matrix X angewendet.
  Wmad<-apply(X = X, MARGIN = 2,FUN = matrixStats::weightedMad,
              w = w, na.rm = T)
}

wgt.Mad <- BIFIE.by(BIFIEobj = dat,
                    vars =  c("HISEI", "E8RPV"),
                    userfct = anwenderfct.weightedMad,
                    userparnames = c("wMadHISEI", "wMadE8RPV"))
summary(wgt.Mad)

## ------------------------------------------------------------
## Abschnitt 10.6.1: Datenexploration
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.6.1, Listing 1: Ungewichtete univariate
# Statistiken
#

# Ungewichtete univariate Statistiken
# Häufigkeitstabelle zu 'eltausb' und 'migrant' (Kreuztabelle)
tab1 <- table(dat.roh[,c("eltausb","migrant")],useNA = "always")
# Ausgabe der Tabelle, ergänzt um Randsummen
addmargins(tab1, FUN = list(Total = sum), quiet = TRUE)

# Ausgabe der Tabelle als Prozentverteilungen
# (in Prozent, gerundet)
round(addmargins(prop.table(x = tab1), FUN = list(Total = sum),
  quiet = TRUE)*100,2)

# Ausgabe mit Prozentverteilungen der Spalten bzw. Zeilen
# (in Prozent, gerundet)
round(prop.table(x = tab1,margin = 2)*100,2)
round(prop.table(x = tab1,margin = 1)*100,2)
```

```
# Ausgabe nicht wiedergegeben

# ------------------------------------------------------------
# Abschnitt 10.6.1, Listing 2: Gewichtete univariate
# Statistiken an imputierten Daten


# Gewichtete univariate Statistiken an imputierten Daten
# Häufigkeitstabelle zu 'eltausb' und 'migrant'
res1 <- BIFIE.freq(BIFIEobj = dat,vars = c("eltausb","migrant"))
summary(res1)
# Häufigkeitstabelle zu 'eltausb' gruppiert nach 'migrant'
res2 <- BIFIE.freq(BIFIEobj = dat,vars = "eltausb",
                   group = "migrant")
summary(res2)
# Kreuztabelle mit zwei Variablen
res3 <- BIFIE.crosstab(BIFIEobj = dat,vars1 = "eltausb",
                       vars2 = "migrant")
summary(res3)

# ------------------------------------------------------------
# Abschnitt 10.6.1, Listing 3: Export der Tabelle
#

res_export <- res1$stat[,c("var","varval","Ncases","Nweight",
  "perc","perc_SE")]
colnames(res_export) <- c("Variable","Wert","N (ungewichtet)",
 "N gewichtet)","Prozent","Standardfehler")
write.table(x = res_export,file = "res_export.dat", sep = ";",
            dec = ",", row.names = FALSE)

## ------------------------------------------------------------
## Abschnitt 10.6.2: Analyse fehlender Werte
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.6.2, Listing 1: Fehlende Werte
#

res1 <- BIFIE.mva(dat, missvars = c("eltausb","migrant"),
                  se = TRUE)
summary(res1)

# ------------------------------------------------------------
# Abschnitt 10.6.2, Listing 2: Fehlende Werte unter Kovariaten
#

res2 <- BIFIE.mva(dat,missvars = c("eltausb","migrant"),
  covariates = c("E8RTWLE","eltausb", "migrant"), se = TRUE)
summary(res2)

## ------------------------------------------------------------
## Abschnitt 10.6.3: Mittelwerte, Perzentilbaender und Quantile
```

```
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.6.3, Listing 1: Hilfsvariable
#

# Hilfsvariable zur Gruppierung anlegen
transform.formula <- as.formula("~ 0 + I(migrant*10+female)")
dat <- BIFIE.data.transform(dat,transform.formula,
                  varnames.new="migrant_female")

# ------------------------------------------------------------
# Abschnitt 10.6.3, Listing 2: Statistiken an Hilfsvariablen
#

# Univariate Statistiken mit Mittelwerten und Standardfehlern
res1 <- BIFIE.univar(BIFIEobj = dat,vars = "E8RPV",
                    group = "migrant_female")
# summary(res1)
mittelwerte<-res1$stat[,c("groupval","M","M_SE")]

# Berechne Quantile
probs<-c(.05,.25,.75,.95)
res2 <- BIFIE.ecdf(BIFIEobj = dat,breaks = probs,
                   quanttype = 1, vars = "E8RPV",
                   group = "migrant_female")
# summary(res2)
quantile<-data.frame(t(matrix(res2$output$ecdf,nrow = 4)))
colnames(quantile)<-probs
# Führe Ergebnisse zusammen
res3<-cbind(mittelwerte,quantile)
print(res3)

# ------------------------------------------------------------
# Abschnitt 10.6.3, Listing 3: IQA
#

# Berechne Interquartilabstand (IQA)
res3$IQA<-res3$"0.75"-res3$"0.25"
# Berechne Grenzen des Vertrauensintervals
res3$VIunten<-res3$M-2*res3$M_SE
res3$VIoben<-res3$M+2*res3$M_SE
round(res3,1)

## ------------------------------------------------------------
## Abschnitt 10.6.4: Gruppenvergleiche mit Regressionen
## ------------------------------------------------------------

# ------------------------------------------------------------
# Abschnitt 10.6.4, Listing 1: Gruppenvergleich Geschlecht
#

# Gruppenvergleich Geschlecht, gesamte Population
```

```
res1 <- BIFIE.linreg(BIFIEobj = dat, formula = E8RPV ~ female)
# Alternativer Aufruf mit identischem Resultat
res1 <- BIFIE.linreg(BIFIEobj = dat,dep = "E8RPV",
                     pre = c("one","female"))

# Vollständige Ausgabe
summary(res1)

# Reduzierte Ausgabe der Ergebnisse
res1_short <- res1$stat[res1$stat$parameter == "b" &
            res1$stat$var == "female",c("est","SE")]
colnames(res1_short) <- c("Geschlechterunterschied","SE")
res1_short

# Gruppenvergleich Geschlecht getrennt nach 'migrant'
res2 <- BIFIE.linreg(BIFIEobj = dat,
                    formula = E8RPV ~ female,
                    group = "migrant")
# Vollständige Ausgabe
summary(res2)

# Reduzierte Ausgabe der Ergebnisse
res2_short <- res2$stat[res2$stat$parameter == "b" &
                        res2$stat$var == "female",
                       c("groupval","est","SE")]
colnames(res2_short) <- c("Migrant","Geschlechterunterschied",
                          "SE")
res2_short

# ------------------------------------------------------------
# Abschnitt 10.6.4, Listing 2: Wald-Test
#

res3 <- BIFIE.univar(vars = "E8RPV",BIFIEobj = dat,
                     group = c("migrant","female"))
res3_wald <- BIFIE.univar.test(BIFIE.method = res3)

# summary(res3_wald)
res3_wald$stat.dstat[,c("group","groupval1","groupval2",
                        "M1","M2","d","d_SE","d_t","d_p")]

# ------------------------------------------------------------
# Abschnitt 10.6.4, Listing 3: Kontrolle um soziale Herkunft
#

# Gruppenvergleich ohne Berücksichtigung der sozialen Herkunft
res1 <- BIFIE.linreg(BIFIEobj = dat, formula = E8RPV ~ migrant)
# summary(res1)
res1$stat[res1$stat$parameter == "b" & res1$stat$var == "migrant",
         c("groupval","est","SE")]

# Gruppenvergleich mit Berücksichtigung der sozialen Herkunft
res2 <- BIFIE.linreg(BIFIEobj = dat,
```

```
                    formula = E8RPV ~ migrant+HISEI+eltausb+buch)
# summary(res2)
res2$stat[res2$stat$parameter == "b" & res2$stat$var == "migrant",
         c("groupval","est","SE")]


## End(Not run)
```

Kapitel 11 *Kapitel 11: Aspekte der Validierung*

Description

Das ist die Nutzerseite zum Kapitel 11, *Aspekte der Validierung*, im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung. Im Abschnitt **Details** werden die im Kapitel verwendeten R-Syntaxen zur Unterstützung für Leser/innen kommentiert und dokumentiert. Im Abschnitt **Examples** werden die R-Syntaxen des Kapitels vollständig wiedergegeben und gegebenenfalls erweitert.

Details

Dieses Kapitel enthält keine Beispiele mit R.

Author(s)

Robert Fellinger, Thomas Kiefer, Alexander Robitzsch, Matthias Trendtel

References

Fellinger, R., Kiefer, T., Robitzsch, A. & Trendtel, M. (2016). Aspekte der Validierung. In S. Breit & C. Schreiner (Hrsg.), *Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung* (pp. 363–398). Wien: facultas.

See Also

Zurück zu Kapitel 10, Reporting und Analysen.
Zur Übersicht.

LSAmitR-Hilfsmethoden *Large-Scale Assessment mit R: Hilfsfunktionen aus den Kapiteln*

Description

Das ist die Nutzerseite zu den Hilfsfunktionen, die in einigen Kapiteln im Herausgeberband Large-Scale Assessment mit R: Methodische Grundlagen der österreichischen Bildungsstandardüberprüfung angewendet werden.

Usage

```
zones.within.stratum(offset, n.str)

covainteraction(dat,covas, nchar)

quintfunct(X,w)
```

Arguments

offset	siehe Kapitel 2
n.str	siehe Kapitel 2
dat	siehe Kapitel 9
covas	siehe Kapitel 9
nchar	siehe Kapitel 9
X	quintfunct ist eine Hilfsfunktion, die nicht für die weitere Verwendung gedacht ist.
w	quintfunct ist eine Hilfsfunktion, die nicht für die weitere Verwendung gedacht ist.

Index