universität wien

Diplomarbeit

Titel der Diplomarbeit

Simulation des Risikos 1. Art und der Teststärke von vier verschiedenen Modelltests für das Rasch-Modell

Verfasserin

Karin Futschek

Angestrebter akademischer Grad

Magistra der Naturwissenschaften (Mag. rer. nat.)

Wien, 2014

Studienkennzahl:	298
Studienrichtung:	Diplomstudium Psychologie
Betreuer:	Univ.-Prof. i.R. Mag. Dr. Klaus D. Kubinger

Danksagung

Ich danke Univ.-Prof. Mag. Dr. Klaus Kubinger für die Betreuung der Diplomarbeit. Insbesondere möchte ich diese Gelegenheit nutzen, mich allgemein bei ihm dafür zu bedanken, dass die besonders gute Lehre, die er im Fachbereich Psychologische Diagnostik an der Universität Wien aufgebaut hat, in mir das brennende Interesse daran und in weiterer Folge am Thema dieser Arbeit geweckt hat.

Mein Dank gilt auch ganz besonders Mag. Jan Steinfeld für die schnelle und unkomplizierte Hilfe bei allen (programmier-)technischen Fragen und Problemen. Insbesondere danke ich ihm auch für alle Maßnahmen, die es erlaubt haben, die Rechenzeit der Simulation zu verkürzen.

Sehr herzlich möchte ich mich bei Mag. Dr. Ingrid Koller für ihre wertvollen Tipps und Literaturempfehlungen bedanken.

Großer Dank gilt meiner Kollegin Ronja Gaderer für ihre Korrekturvorschläge und das gewissenhafte Lesen dieser Arbeit.

Abstract – deutsche Version

Nur wenn ein psychologisch-diagnostisches Verfahren Rasch-Modell-konform ist, ist die Anzahl gelöster Aufgaben ein suffizienter Schätzer für die Fähigkeit von Personen. Um die verschiedenen Annahmen des Rasch-Modells zu prüfen, stehen eine Reihe von Modelltests zur Verfügung. In der vorliegenden Arbeit wurden in einer Simulationsstudie vier dieser Modelltests, der Likelihood-Ratio-Test nach Andersen, der z-Test nach Fischer und Scheiblechner mit der Schätzfunktion nach Wald, der Martin-Löf-Test, sowie die 2009 von Kubinger, Rasch und Yanagida vorgeschlagene Möglichkeit, eine dreifache teilhierarchische Varianzanalyse zu verwenden, in Bezug auf ihren Fehler erster Art sowie ihre Teststärke miteinander verglichen.

Für 20 Items und zwischen 100 und 300 Personen wurde folgendes simuliert: Keine Modellverletzung, Modellverletzung mit einem DIF-Paar, wobei die Itemparameterdifferenz (Effektstärke) zwischen einer halben und zwei Standardabweichungen der Personenparameter variierte, und eine Modellverletzung unter Multidimensionalität mit einer latenten Korrelation von 0,5.

Es zeigte sich, dass der Likelihood-Ratio-Test nach Andersen, die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida sowie der z-Test nach Fischer und Scheiblechner das Risiko 1. Art von 5% größtenteils einhielten, während beim Martin-Löf-Test das Risiko erster Art bei 0% lag.

Die Teststärke der dreifachen Varianzanalyse und die des Likelihood-Ratio-Tests erwies sich als annähernd gleich groß, wobei die der dreifachen Varianzanalyse etwas größer war. Dahingegen lag die Teststärke des z-Tests nach Fischer und Scheiblechner deutlich darunter. Wie zu erwarten hing die Teststärke von der Personenanzahl und der Effektstärke ab. Ab einer Effektstärke von einer Standardabweichung lagen die Teststärke der dreifachen Varianzanalyse und die des Likelihood-Ratio-Tests nach Andersen für alle simulierten Personenzahlen über 0,8.

Bei der Modellverletzung durch Multidimensionalität war die Teststärke des Likelihood-Ratio-Tests auf dem nominellen Niveau des Risikos 1. Art, die des Martin-Löf-Tests lag bei 100 Personen bei 3% und erreichte bei 300 Personen 70% (α=0,05).

Die Arbeit zeigt, dass für die hier simulierten Bedingungen die dreifache Varianzanalyse etwas besser abschneidet, als der Likelihood-Ratio-Test nach Andersen und diese somit eine attraktive Alternative zu ihm darstellt.

Schlüsselwörter: Rasch-Modell, Likelihood-Ratio-Test nach Andersen, dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida, z-Test nach Fischer und Scheiblechner, Martin-Löf-Test, Fehler 1. Art, Fehler 2. Art, Teststärke

Abstract – English version

Only if the Rasch model holds for a psychological diagnostic assessment the sum of right answers is a sufficient estimator of the person's ability. To check the assumptions of the Rasch model a set of different model tests is available. In this paper a simulation study was done to compare four model tests regarding their type I risk and power: The Andersen's Likelihood-Ratio test, the z-Test of Fischer and Scheiblechner with estimation of Wald, the Martin-Löf test as well as the new approach of Kubinger, Rasch and Yangida (2009) who proposed to use a three-way nested analysis of variance design.

For 20 items and 100 to 300 persons the following scenarios were simulated: No violation of the model, violation by one pair of DIF, which varied between a half and two standard deviations of the person's parameter, as well as violation of the model by multidimensionality with a latent correlation of the two dimensions of 0.5.

It was shown that the Andersen's Likelihood-Ratio test, the analysis of variance and the Wald test mostly hold the type I risk of 5%, while the simulated type I risk of the Martin-Löf test was 0%.

The power of the analysis of variance was of the same magnitude, but slightly higher than that of Andersen's Likelihood-Ratio test. In contrast the power of the Wald test was considerably lower. As expected the power depended on the number of persons and the effect size. Starting from an effect size of one standard deviation the power of the analysis of variance and of Andersen's Likelihood-Ratio test was over 0.8 for all simulated numbers of persons.

In case of multidimensionality the power of Andersen's Likelihood-Ratio test was at the same level as its type I risk. The power of the Martin-Löf test was 3% in case of 100 persons and reached 70% for 300 persons (α=0.05).

For the simulated conditions it was shown, that the analysis of variance design is an attractive alternative to Andersen's Likelihood-Ratio test.

Keywords: Rasch model, Andersen's Likelihood-Ratio test, analysis of variance, mixed model, z-Test of Fischer and Scheiblechner, Martin-Löf test, type I risk, type II risk, power

Inhaltsverzeichnis

I. Einleitung

In der Angewandten Psychologie ist es häufig von Interesse, Ausprägungen von Eigenschaften oder Fähigkeiten messbar zu machen. Sei es um ein differenziertes Bild über die Fähigkeiten einer Person zu bekommen, um eine qualifizierte Bildungsberatung anbieten zu können, geeignete Bewerber/innen für Studienplätze auszuwählen oder um Vergleiche zwischen den schulischen Leistungsniveaus verschiedener Länder anzustellen, wie das bei der PISA-Studie der Fall ist.

Als Messinstrumente dienen hier unterschiedliche psychologisch-diagnostische Verfahren, bei deren Konstruktion eine oder mehrere Annahmen getroffen werden, beispielsweise darüber, wie der tatsächliche Ausprägungsgrad (latente Variable) mit dem gemessenen Ausprägungsgrad (manifeste Variable) zusammenhängt. Man bedient sich eines Modells, um diesen Zusammenhang zu beschreiben.

Ein statistisches Modell sollte überprüfbare Modellannahmen besitzen. Aus dem Modell und dessen Annahmen sind Modelleigenschaften abzuleiten, mit Hilfe derer relevante Aussagen über die Daten getroffen werden können, auf die das Modell angewandt wird.

Eines dieser Modelle, das den Zusammenhang zwischen latenter und manifester Variable beschreibt, ist das Rasch-Modell. Das dichotome logistische Modell von Rasch kann dann für die Konstruktion von psychologisch-diagnostischen Verfahren herangezogen werden, wenn diese aus mehreren Aufgaben oder Fragen bestehen, die unabhängig voneinander beantwortet werden können und die dieselbe zugrundeliegende Dimension messen. Für die Auswertung jeder einzelnen Frage dürfen nur zwei Kategorien wie richtig/falsch oder ja/nein verwendet werden. Daher eignet sich das Rasch-Modell insbesondere für die Konstruktion von Leistungstests, deren Fragen entweder richtig oder falsch beantwortet werden.

Der Vorteil dieses Testmodells gegenüber anderen Testmodellen ist, dass die getroffenen Modellannahmen überprüfbar sind. Es stehen eine Reihe von unterschiedlichen Modelltests zur Verfügung, die streng genommen prüfen, ob das Modell nicht gilt, da bei diesen Modelltests als Nullhypothese davon ausgegangen wird, dass das Modell gilt.

Insbesondere hervorzuheben ist, dass die Gültigkeit des Rasch-Modells die Voraussetzung für faires Testen darstellt, wenn die ungewichtete Summe gelöster Aufgaben als Schätzer für die Personenfähigkeit dienen soll. Fischer (1974, 1995) lieferte dazu den Beweis.

Faires Testen ist nicht nur in Bezug auf die ethischen Standards innerhalb der Psychologie wichtig, sondern es ist auch ein zentrales Qualitätsmerkmal psychologisch-diagnostischer Verfahren. Die Gültigkeit des Rasch-Modells für ein psychologisch-diagnostischer Verfahren ermöglicht außerdem die Erfüllung des Gütekriteriums Skalierung, welches verlangt, dass empirische Verhaltensrelationen durch die Verrechnungsvorschrift der Testwerte adäquat abgebildet werden (Kubinger, 2009).

Bei der praktischen Anwendung der Modelltests zum Rasch-Modell in der Testkonstruktion wird im Allgemeinen häufig davon ausgegangen, dass das Risiko 1. Art bei 5% liegt, über das Risiko 2. Art und somit die Teststärke liegen jedoch keine Informationen vor. Es ist nicht üblich die Stichprobengröße anhand einer gewünschten Effektstärke, Teststärke und dem Risiko 1. Art zu bestimmen. Da keine Formel zur Berechnung dieser Größen vorhanden ist, bleibt nur die Möglichkeit sie über Simulation zu ermitteln.

In der vorliegenden Arbeit sollen nun vier verschiedene Modelltests des Rasch-Modells via Simulationsstudie miteinander verglichen werden und ihr tatsächliches Risiko 1. Art und ihre Teststärke bei gegebener Stichprobengröße (Personenanzahl) und Effektstärke ermittelt werden.

Im theoretischen Teil der Arbeit wird das Rasch-Modell mit seinen Eigenschaften und Methoden der Parameterschätzung beschrieben, sowie vier Modelltests, der Likelihood-Ratio-Test nach Andersen, die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida, der z-Test nach Fischer und Scheiblechner und der Martin-Löf-Test. Allgemein wird kurz auf den Fehler 1. und 2. Art im Rasch-Modell eingegangen.

Der zweite Teil der Arbeit beschreibt die durchgeführte Simulationsstudie, im Anhang ist der vollständige Programmcode zur Simulation angeführt.

Da das Rasch-Modell meistens im Zusammenhang mit Leistungstests verwendet wird, wird in dieser Arbeit überwiegend nur mehr der Begriff Test statt psychologisch-diagnostisches Verfahren verwendet, wenn die Anwendung des Rasch-Modells beispielhaft beschrieben wird. Um zu charakterisieren, was die beiden Parameter des Rasch-Modells schätzen, werden folglich die Begriffe Personenfähigkeit und Itemschwierigkeit verwendet.

II. Theoretischer Teil

In diesem Teil der Arbeit werden die wichtigsten Grundlagen zusammengefasst, die für das Verständnis der durchgeführten Simulationsstudie relevant sind. Es wird das Rasch-Modell mit seinen, für die Testkonstruktion wichtigen, Eigenschaften dargestellt und insbesondere die Verwobenheit der Modelleigenschaften herausgearbeitet. Danach werden drei Möglichkeiten erläutert, wie die Parameter im Modell geschätzt werden können. Schließlich werden noch die vier Modelltests beschrieben, die in der Simulationsstudie zur Anwendung kamen, sowie der Fehler 1. Art und die Teststärke im Rasch-Modell.

1. Das Rasch-Modell

Das Rasch-Modell des dänischen Statistikers Georg Rasch ist ein dichotomes, logistisches Testmodell. Dichotom, da es auf Tests angewandt werden kann, die genau zwei Antwortkategorien zulassen und logistisch, da dem Modell eine logistische Funktion zu Grunde liegt. Es ist das einfachste Modell der probabilistischen Testtheorie, die auch Item-Response-Theorie (IRT) genannt wird.
Das besondere an der probabilistischen Testtheorie ist, dass zwischen manifesten und latenten Variablen unterschieden wird. Latente Variablen sind Konstrukte, wie Intelligenz oder bestimmte Persönlichkeitseigenschaften, die nicht direkt beobachtbar sind. Manifeste Variablen hingegen, wie das Antwortverhalten einer Person in psychologisch-diagnostischen Verfahren, sind direkt beobachtbar, und dienen als Indikatoren für die latenten Variablen. Es soll somit von einer manifesten Variable auf eine latente Variable geschlossen werden. Dafür müssen Annahmen über den Zusammenhang dieser beiden Variablen getroffen werden.
Im Rasch-Modell wird die Wahrscheinlichkeit dafür, wie eine Person v mit bestimmter Fähigkeit θ_v, ein Item i mit bestimmter Schwierigkeit β_i beantwortet, als logistische Funktion modelliert, die von der Differenz der Fähigkeit der Person und der Schwierigkeit des Items abhängt:

$$P(X_{vi} = x_{vi}|\theta_v, \beta_i) = \frac{e^{x_{vi}(\theta_v - \beta_i)}}{1 + e^{\theta_v - \beta_i}}$$

X_{vi} ist die Variable, die das Antwortverhalten einer Person v beim Item i beschreibt, ihre Ausprägungen x_{vi} können nur die Werte 0 und 1 annehmen, da es sich um ein

dichotomes Modell handelt. x_{vi} ist 0 wenn ein Item nicht gelöst wurde und 1 wenn es gelöst wurde. Die Wahrscheinlichkeit, ein Item zu lösen bzw. nicht zu lösen, lässt sich somit wie folgt anschreiben:

$$P(1|\theta_v, \beta_i) = \frac{e^{(\theta_v - \beta_i)}}{1 + e^{\theta_v - \beta_i}} \qquad P(0|\theta_v, \beta_i) = \frac{1}{1 + e^{\theta_v - \beta_i}}$$

θ wird als Personen- oder Fähigkeitsparameter bezeichnet, β wird als Item- oder Schwierigkeitsparameter bezeichnet.

Die Funktion, die die Lösungswahrscheinlichkeit für ein Item mit festgesetzter Schwierigkeit in Abhängigkeit von der Personenfähigkeit beschreibt, wird Itemcharakteristikkurve (ICC) genannt. Zur Veranschaulichung seien in der folgenden Grafik die ICCs für sieben verschiedene Items mit einer Schwierigkeit zwischen -3 und 3 dargestellt. Am Wendepunkt der Funktion beziehungsweise bei einer Lösungswahrscheinlichkeit 0,5 kann auf der x-Achse der Schwierigkeitsparameter abgelesen werden, da die Lösungswahrscheinlichkeit für ein Item, dessen Schwierigkeitsparameter gleich dem Personenparameter ist, 0,5 beträgt.

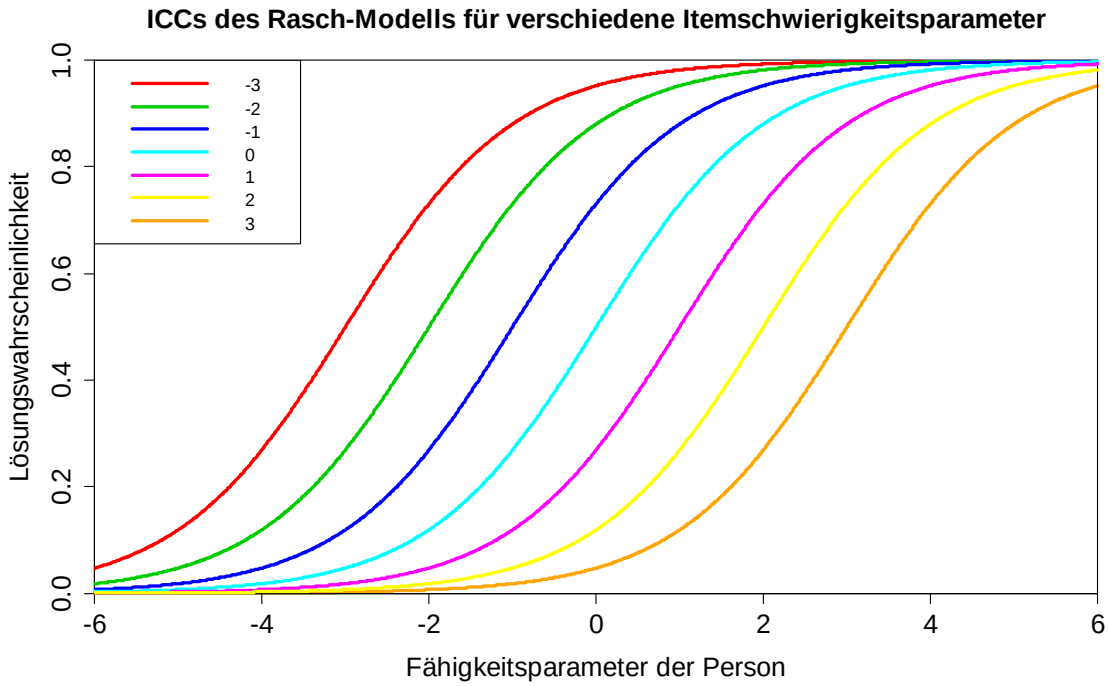

Abbildung 1: ICCs des Rasch-Modells

Wie man der Grafik (Abb. 1) entnehmen kann, sind die einzelnen ICCs nur um die Itemschwierigkeit verschoben. Der Anstieg ist daher für alle Kurven bis auf die

Verschiebung gleich, was bedeutet, dass im Rasch-Modell alle Items die gleiche Trennschärfe haben.
Damit das Rasch-Modell gilt und die Formel bzw. die ICCs den Zusammenhang zwischen Personenfähigkeit, Itemschwierigkeit und Lösungswahrscheinlichkeit beschreiben, müssen verschiedene Voraussetzungen erfüllt sein, die an das Rasch-Modell gestellt werden. Diese Voraussetzungen können auch als Eigenschaften des Rasch-Modells bezeichnet werden.

Eindimensionalität

Zentrale Eigenschaft des Rasch-Modells ist, dass der Test eindimensional messen muss. Das zeigt sich in zweierlei Hinsicht: Zum einen müssen alle Items dieselbe Dimension (Fähigkeit) messen, zum anderen darf die Beantwortung eines Items für alle Personen nur von derselben zugrunde liegenden latenten Dimension (Fähigkeit) abhängen.
Außerdem soll der Test über verschiedene Personengruppen hinweg dieselbe Dimension messen. Für verschiedene inhaltlich relevante Personengruppen, wie Personen unterschiedlicher sozialer Herkunft oder unterschiedlichen Geschlechts, sollen die Items gleich schwierig sein.
Da das Messen von lediglich einer Fähigkeit ein wesentliches Qualitätskriterium eines Tests ist und die Voraussetzung für faires Testen darstellt, ist diese Eigenschaft der Eindimensionalität von besonderer praktischer Relevanz in der Testkonstruktion.

Lokale stochastische Unabhängigkeit

Im Rasch-Modell wird von lokaler stochastischer Unabhängigkeit ausgegangen. Auf Itemebene bedeutet das, dass die Antwort auf ein Item nicht davon abhängt wie ein anderes Item beantwortet wurde, sondern, dass die Wahrscheinlichkeit ein Item zu lösen nur von der Fähigkeit der Person, der Schwierigkeit des Items und dem Zufall abhängt (Fischer, 1974).
Wenn die lokale stochastische Unabhängigkeit auf Itemebene gegeben ist, weil Items unabhängig voneinander beantwortet werden, und die Personenfähigkeit konstant gehalten wird, verschwindet die Interkorrelation zwischen den Items. Items sind also „lokal“, das heißt bei konstanter Personenfähigkeit, voneinander unabhängig.
Die lokale stochastische Unabhängigkeit auf Itemebene kann verletzt sein, wenn zwei Items inhaltlich sehr ähnlich oder gleich sind, wenn Lerneffekte zwischen den Items auftreten oder wenn Items aus inhaltlichen Gründen aufeinander aufbauen.

Verletzungen der lokalen stochastischen Unabhängigkeit auf Itemebene können mit den später beschriebenen Modelltests ausfindig gemacht werden. Kann man die Ursache der Modellverletzung nicht beheben, beispielsweise durch eine Änderung der Reihenfolge der Items oder durch inhaltliche Änderungen am Item, werden die Items in der Regel aus dem Test herausgenommen. Dies ist nicht nur sinnvoll, damit das Rasch-Modell gilt und alle Eigenschaften, die daraus abgeleitet werden können, sondern auch aus Gründen der Effizienz, da diese Items keine zusätzliche Information über den Personenparameter liefern.

Die Verletzungsmöglichkeit der lokalen stochastischen Unabhängigkeit auf Personenebene, wenn zwei Personen voneinander abschreiben, wird in der Regel nicht gezielt mit Modelltests überprüft, da sich dieses Problem bei der Testvorgabe verhindern lässt.

Die lokale stochastische Unabhängigkeit der Items ist auch deshalb unabdingbar, da sie eine Voraussetzung für die Berechnung der weiter unten beschriebenen suffizienten Statistik ist.

Spezifische Objektivität

Die Eigenschaft der spezifischen Objektivität im Rasch-Modell bezieht sich auf die Vergleichbarkeit zweier Items bezüglich ihrer Schwierigkeit und die Vergleichbarkeit zweier Personen bezüglich ihrer Fähigkeit. Die spezifische Objektivität erlaubt Personenparameter zu vergleichen unabhängig davon, welche endlich vielen Items aus einem hypothetischen Itemuniversum bearbeitet wurden, um so eine Aussage über die Fähigkeit zu treffen, wie gut ein bestimmter Itemtyp bearbeitet werden kann. Sofern nicht alle Items zu schwierig für die Personen sind, ist es somit egal, welche und wie viele Items man heranzieht und welche anderen Personen getestet wurden, um die Fähigkeiten zweier Personen einzuschätzen. Ebenso ist es egal, welche Personen man testet, um die Schwierigkeiten zweier Items einzuschätzen.

Im Gegensatz zur Differenz zweier Item- oder Personenparameter ist die relative Lösungswahrscheinlichkeit keine spezifisch objektive Messung, da sie nicht gleich bleibt, wenn ein Item durch ein anderes mit gänzlich anderer Schwierigkeit ausgetauscht wird (Fischer, 1974).

Wenn das Rasch-Modell für einen Test gilt, dann muss die Auswahl der Personen und Items aus der Population nicht mehr zufällig sein, um ihre Fähigkeit/Schwierigkeit zu vergleichen. Diese Eigenschaft wird als Stichprobenunabhängigkeit bezeichnet (Fischer, 1974).

Ein zentraler Aspekt der spezifischen Objektivität ist, dass für verschiedene Subgruppen die Itemschwierigkeistparameterrelationen gleich sind. Dies wird mit Modelltests anhand von internen (Teilung nach der Testleistung in zwei Gruppen) sowie verschiedenen externen Teilungskriterien (Geschlecht, Bildungsstatus, Muttersprache, …) überprüft. Dabei werden für zwei Gruppen getrennt die Itemparameter geschätzt und mit Modelltests geprüft, ob sich diese unterscheiden.

Items können auf verschiedene Art und Weise in bestimmten Subgruppen unterschiedlich „funktionieren“. Ist Differential Item Functioning (DIF) im Test vorhanden, so misst dieser unfair, weil nicht alle Items für alle Personen gleich schwierig sind. Die Summe der gelösten Items als Rohscore zu verwenden würde eine Gruppe benachteiligen. Die ICCs wären nicht mehr parallel. In der folgenden Grafik (Abb. 2) ist dargestellt, wie die ICCs zweier Teilgruppen (grüne und rote Kurve) im Vergleich zur Gesamtgruppe (blaue Kurve) aussehen. Gäbe es ein weiteres Item (orangene Kurve), das gleich schwierig für beide Teilgruppen wäre und parallel zu ihnen verlaufen würde, so wären die ICCs nicht parallel und das Rasch-Modell verletzt.

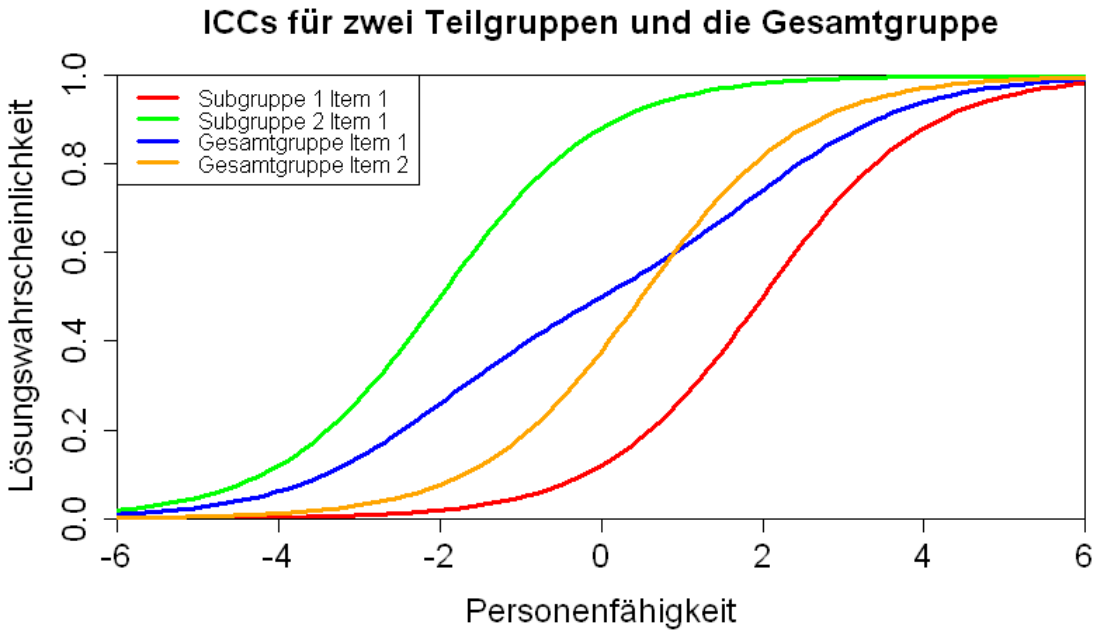

Abbildung 2: ICCs für zwei Teilgruppen und die Gesamtgruppe

Suffiziente Statistik

Messbare Funktionen, die aus dem Stichprobenraum in einen beliebigen Messraum abbilden, werden als suffizient bezeichnet, wenn dabei ein mehrdimensionaler Datenvektor in eine einfachere Form transformiert wird, ohne dass dabei Information über die Wahrscheinlichkeitsverteilung des mehrdimensionalen Datenvektors verloren

geht. Eine Statistik ist also dann suffizient, wenn sie alle Informationen enthält, die in den Daten enthalten sind.

Gilt das Rasch-Modell, weil alle genannten Eigenschaften zutreffen, sind die Randsummen der Datenmatrix eine suffiziente Statistik für die Schätzer der Item- und Personenparameter. Die Summe über alle gelösten Aufgaben ist eine suffiziente Statistik für den Personenparameter, und die Summe über alle Personen, die ein Item gelöst haben, ist eine suffiziente Statistik für den Itemparameter. Es reicht folglich aus für jedes Item zu wissen wie viele Personen es gelöst haben, und für jede Person zu wissen, wie viele Items sie gelöst hat, um die Item- und Personenparameter zu schätzen. Welche Person, welches Item gelöst hat, ist irrelevant, da die gesamte wichtige Information in den Randsummen steckt. Diese Eigenschaft ist unter anderem auch für die Parameterschätzung wichtig, die im nächsten Kapitel beschrieben wird.

Dass die Randsummen im Rasch-Modell eine suffiziente Statistik für die Parameter sind, kann aufgrund der Annahme der lokalen stochastischen Unabhängigkeit unter Zuhilfenahme des Faktorisierungstheorems von Neyman gezeigt werden.

Das Faktorisierungstheorem (nachzulesen bei Hogg, McKean und Craig, 2005) besagt, dass eine Statistik suffizient ist, wenn man die gemeinsame Dichtefunktion, abhängig von den Daten x_1 bis x_n, und den Parametern, in zwei Faktoren aufteilen kann, wobei der erste Faktor von der Statistik (hier: Randsumme) und dem Parameter abhängt und der zweite Faktor nicht vom Parameter abhängt.

Da im Rasch-Modell von lokaler stochastischer Unabhängigkeit ausgegangen wird, lässt sich die Dichte folgendermaßen schreiben:

$$P(X = x|\theta, \beta) = \prod_{v=1}^{N} \prod_{i=1}^{k} \frac{e^{x_{vi}(\theta_v - \beta_i)}}{1 + e^{\theta_v - \beta_i}} = \frac{e^{\sum_{v=1}^{N} \sum_{i=1}^{k} x_{vi}\theta_v}}{e^{\sum_{v=1}^{N} \sum_{i=1}^{k} x_{vi}\beta_i}} \Big/ \prod_{v=1}^{N} \prod_{i=1}^{k} (1 + e^{\theta_v - \beta_i})$$

Die Faktoren dieses Bruchs kann man für den Parameter β und für den Parameter θ so auswählen, dass je ein Faktor nicht vom Parameter abhängt. Die Randsummen sind somit eine suffiziente Statistik für die Parameter.

Im Besonderen gibt Fischer (1974) den Beweis (nachzulesen auf den Seiten 193-203), dass die suffiziente Statistik, gemeinsam mit einigen technischen Voraussetzungen, eine notwendige und hinreichende Bedingung für das Rasch-Modell ist. Als weitere Voraussetzungen zusätzlich zur suffizienten Statistik nennt er dichotome Items, die dieselbe Fähigkeit erfassen (eindimensional sind), monoton steigende ICCs und lokale stochastische Unabhängigkeit.

Da die Randsummen eine suffiziente Statistik darstellen, ist eine Analyse der Antwortmuster auf individueller Basis nicht notwendig und nur dann sinnvoll, wenn das Rasch-Modell nicht gilt. Auch eine Gewichtung der Items nach der Schwierigkeit bei der Summenbildung ist nicht notwendig, wenn das Rasch-Modell gilt. Dieser Sachverhalt unterstreicht die Vorzüge der Verwendung des Rasch-Modells bei der Testkonstruktion.

2. Parameterschätzung

Hat man einen neuen Test erstellt und die Aufgaben einer Gruppe von Personen vorgegeben, so sind neben der tatsächlichen Qualität der Items zwei Dinge von Interesse: Wie fähig sind die getesteten Personen? Und wie schwierig sind die konstruierten Items? Beim Kalibrieren eines Tests nach dem Rasch-Modell gibt es verschiedene Möglichkeiten die Item- und Personenparameter zu schätzen. Die Schätzmethoden sind verschiedene Varianten der Maximum-Likelihood Methode. Anhand der beobachteten Daten soll die Likelihood für bestimmte Personen- und Itemparameter maximiert werden:

$$L(\theta_1, \dots, \theta_N, \beta_1, \dots, \beta_k | x_{11}, \dots, x_{Nk}) = \prod_{v=1}^{N} \prod_{i=1}^{k} \frac{e^{x_{vi}(\hat{\theta}_v - \hat{\beta}_i)}}{1 + e^{\hat{\theta}_v - \hat{\beta}_i}} \rightarrow max$$

Die Formel beschreibt die Likelihoodfunktion für die Personenparameter θ_v und die Itemparameter β_i, gegeben der beobachteten Datenmatrix X, die aus N Zeilen und k Spalten besteht, wobei N die Anzahl der Personen ist und k die Anzahl der Items, v der Laufindex für die Personen und i der Laufindex für die Items. Die Personen- und Itemparameter sollen bei der Schätzung nun so gewählt werden, dass die aufmultiplizierten Wahrscheinlichkeiten, genau die beobachteten Daten zu erhalten, maximal wird. Eine anschauliche Beschreibung des Prinzips anhand von Daten geben Koller, Alexandrowicz und Hatzinger (2012, S. 30-35). Es gibt verschiedene Likelihoodfunktionen, die herangezogen werden können, um die Parameter zu schätzen. Die zur Veranschaulichung oben beschriebene Formel wird bei der Joint Maximum-Likelihood-Schätzung verwendet. Es werden beide Parameter simultan geschätzt. Das Problem, wenn beide Parameter gemeinsam geschätzt werden, ist jedoch, dass die Genauigkeit der Schätzung der Itemparameter von der Anzahl der zu schätzenden Personenparameter abhängt und umgekehrt. Da meistens deutlich mehr Personen getestet werden als Items vorliegen, ist bei der simultanen Schätzung der Parameter die Schätzung der Personenparameter ungenau, während die Itemparameter präziser geschätzt werden können.

Eine elegantere Schätzung der Itemparameter im Rasch-Modell ist die conditional Maximum-Likelihood-Methode. Bei dieser Methode wird die besondere Eigenschaft des Rasch-Modells genutzt, dass die Summe der gelösten Items eine suffiziente Statistik für den Personenparameter darstellt. Durch Umformen der oben beschriebenen Likelihoodfunktion und unter Verwendung der Eigenschaft der suffizienten Statistik, kann die conditional Likelihoodfunktion hergeleitet werden, die den Parameter θ nicht mehr enthält (Fischer, 1974). Die Möglichkeit, Parameter getrennt voneinander zu schätzen, wird als Separierbarkeit der Parameter bezeichnet. Die Formel für die conditional Likelihoodfunktion lautet dann wie folgt:

$$cL = \prod_{v=1}^{N} \frac{e^{-\sum_i^k x_{vi}\widehat{\beta}_i}}{\sum_{\underline{x}|r} \prod_{i=1}^{k} e^{-x_i\widehat{\beta}_i}}$$

wobei r die Anzahl unterschiedlicher Itemparameter ist und $\underline{x}$ für ein bestimmtes Antwortmuster steht.

Eine weitere Möglichkeit die Itemparameter zu schätzen ist die marginal Maximum-Likelihood-Methode. Um die Itemparameter bei dieser Methode ebenfalls unabhängig von den Personenparametern zu schätzen, wird eine Annahme über die Verteilung der Personenparameter getroffen und schließlich über die Personenparameter integriert (Rost, 2004).

Werden die Itemparameter mit der conditional oder der marginal Maximum-Likelihood-Funktion geschätzt, so müssen die Personenparameter in einem zweiten Schritt gesondert ermittelt werden. Für diese Schätzung kann eine Maximum-Likelihood-Methode wie bei der joint Maximum-Likelihood-Methode angewandt werden, wobei die Itemparameter nicht mehr mitgeschätzt werden. Aufgrund der suffizienten Statistik wird allen Personen, die dieselbe Anzahl von Aufgaben gelöst haben, derselbe Personenparameter zugeordnet. Löst eine Person keines oder alle Items, so ist eine endliche Schätzung des Personenparameters nicht möglich. Es werden daher nur k-1 verschiedene Personenparameter geschätzt. Auch die Itemparameter können nicht immer geschätzt werden. Wenn alle oder keine Person ein Item lösen kann, ist es nicht möglich, die Schwierigkeit des Items zu schätzen.

Die geschätzten Personen- und Itemparameter liegen auf einer Differenzenskala. Sie sind somit fixiert bis auf eine additive Transformation. Deshalb wird ein Bezugspunkt festgelegt, damit sowohl Item- als auch Personenparameter als eindeutige Zahl angegeben werden können. Dies geschieht entweder indem einem Item eine bestimmte Schwierigkeit zugeordnet wird, oder indem die Items Summe-null normiert

werden, was bedeutet, dass die Summe der Itemparameter null ergibt. Die Differenz zweier Personenparameter kann, wenn das Rasch-Modell gilt, aufgrund der spezifischen Objektivität als Fähigkeitsunterschied zweier Personen interpretiert werden, unabhängig davon, ob die Itemparameter normiert wurden und unabhängig davon, ob der Test eher einfache oder schwierige Items enthält (Fischer, 1974).
Die Genauigkeit der Parameterschätzung hängt von der Verteilung der Item- und Personenparameter in der Stichprobe ab.
In der vorliegenden Arbeit werden die Itemparameter mit der conditional Maximum-Likelihood Methode geschätzt. Die im folgenden Kapitel beschriebenen Modelltests verwenden ebenfalls diese Schätzmethode für die Modellprüfung. Mit Hilfe der Likelihoodfunktion können nicht nur die Parameter geschätzt werden, sie wird auch herangezogen um verschiedene Modelle zu vergleichen. Je größer die Likelihood eines Modells ist, desto wahrscheinlicher ist das Modell. Die Likelihoodfunktionen verschiedener Modelle werden miteinander verglichen, um eine Aussage darüber zu treffen, welches Modell die Daten besser beschreibt. Im nächsten Kapitel werden verschiedene Modelltests beschrieben, um die Geltung des Rasch-Modells zu prüfen.

3. Modellgeltungstests für das Rasch-Modell

Mit dem Likelihood-Ratio-Test nach Andersen kann die Gültigkeit des Rasch-Modells einerseits anhand des internen Teilungskriteriums überprüft werden, andererseits kann mit externen Teilungskriterien geprüft werden ob DIF vorliegt. Es wird so global für den Test überprüft, ob sich die Itemschwierigkeistparameter in verschiedene Gruppen unterscheiden. Der z-Test nach Fischer und Scheiblechner prüft dies ebenfalls, jedoch lokal für jedes Item. Darüber hinaus gibt es noch grafische Modellgeltungskontrollen, die die Itemparameterschätzungen für verschiedene Gruppen grafisch sichtbar machen. Auch die dreifache Varianzanalyse von Kubinger, Rasch und Yanagida prüft auf globaler Ebene Unterschiede der Itemschwierigkeistparameter zwischen Gruppen in Bezug auf ein bestimmtes externes Teilungskriterium. Der Martin-Löf-Test überprüft Eindimensionalität.

Likelihood-Ratio-Test nach Andersen

Der Likelihood-Ratio-Test nach Andersen (Andersen, 1973) ist ein globaler Modelltest, der die Itementsprechung in Bezug auf ein bestimmtes Teilungskriterium für alle Items gleichzeitig prüft. Der Test vergleicht dabei die Schätzungen der Itemschwierigkeiten

für (mindestens) zwei verschiedene Gruppen mit der Schätzung der Itemschwierigkeiten über alle Personen. Die Parameterschätzung basiert auf der conditional Maximum-Likelihood Schätzung. Wird in zwei Teilstichproben geteilt, lautet die Teststatistik T_{LR}:

$$T_{LR} = -2 * \ln \frac{cL_0}{cL_1 * cL_2} \sim \chi^2_{(k-1)}$$

cL_0 ist die conditional Likelihood in der Gesamtstichprobe, cL_1 die der ersten und cL_2 die der zweiten Teilstichprobe, k ist die Anzahl der Items. Der Likelihood-Ratio-Test vergleicht so die Likelihood zweier Modelle, wobei im Zähler das Gesamtmodell steht und im Nenner das Modell mit zwei Teilstichproben. Der Quotient aus der Schätzung für die Gesamtstichprobe und dem Produkt der Teilstichproben wird dann sehr klein, wenn das Modell mit den Teilstichproben wahrscheinlicher ist. Durch die Multiplikation mit -2 werden die Größenverhältnisse umgedreht, sodass im Falle von unterschiedlichen Schätzungen der Itemparameter in den beiden Teilgruppen die Teststatistik groß und der χ^2-Test signifikant wird. Bei ähnlichen Schätzungen der Itemparameter in den beiden Gruppen ist die Teststatistik klein und der χ^2-Test wird nicht signifikant.

Als Teilungskriterien für den Andersen-Likelihood-Ratio-Test kommen der Testscore - das interne Teilungskriterium und externe Teilungskriterien (z.B. nach Geschlecht) in Frage.

Alexandrowicz (2002) führte eine umfangreiche Simulationsstudie zur Teststärke des Likelihood-Ratio-Tests nach Andersen durch und prüfte dabei das interne Teilungskriterium des Tests. Bei der Simulation der Teststärke variierte er unter anderem die Art der Modellverletzung des Rasch-Modells und die Stichprobengröße. Es zeigte sich, dass die Stichprobengröße die Teststärke am deutlichsten beeinflusste. Es wurde der übliche Zusammenhang deutlich, dass große Stichproben mit großer Teststärke einhergehen.

Stelzl (1979) zeigte in einer Simulationsstudie, dass der Likelihood-Ratio-Test nach Andersen mit dem internen Teilungskriterium nicht immer geeignet ist, Modellverletzungen unter Mehrdimensionalität zu erkennen.

Dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida

2009 stellten Kubinger, Rasch und Yanagida eine neue Möglichkeit vor, das Rasch-Modell auf Modellgültigkeit zu prüfen. Für die Überprüfung des Rasch-Modells anhand

eines externen Kriteriums, wird eine genestete Varianzanalyse der Form (A $\succ$ ***B***) x C vorgeschlagen. Es handelt sich um einen dreifaktoriellen, teilhierarchischen Versuchsplan. A ist ein fester Faktor und teilt die Daten in zwei Teilstichproben, repräsentiert somit das externe Teilungskriterium. B ist ein zufälliger Faktor und steht für die Personenfähigkeit. B ist zufällig, da in der Regel nicht alle möglichen Personen getestet werden, sondern nur eine zufällige Stichprobe gezogen wird. Außerdem ist B in A genested, da eine Person immer nur in einer der beiden Gruppen von A sein kann. C steht für die Itemschwierigkeit und ist ebenfalls ein fester Faktor.

Die Modellgleichung lautet: $y_{ijk} = \mu + a_i + \mathbf{b_{ij}} + c_k + (ac)_{ik} + \mathbf{e_{ijk}}$ *(zufälliger Faktor **fett** gedruckt)*

Das Modell beinhaltet den Gesamtmittelwert μ, die Haupteffekte A, B und C, die Wechselwirkung zwischen A und C sowie den Fehler e_{ijk}.
Der F-Test des Wechselwirkungseffekts zwischen A (Teilungskriterium) und C (Itemparameter) prüft die Itementsprechung. Ist er signifikant, so wird die unter der Nullhypothese angenommene Gültigkeit des Rasch-Modells verworfen. Es wird angenommen, dass die Itemparameter in den durch das Teilungskriterium gebildeten Gruppen verschieden sind (Kubinger, Rasch & Yanagida, 2009).
Die neue Methode, eine Varianzanalyse für die Auswertung heranzuziehen, wird insbesondere vor dem Hintergrund vorgeschlagen, in Zukunft die Möglichkeiten zu nutzen, bei der Varianzanalyse das Risiko 2. Art bei gegebener Teststärke und gegebenem Risiko 1. Art zu kontrollieren und Stichprobengrößen-Vorausberechnungen bei der Prüfung des Rasch-Modells vorzunehmen. Da bei der Anwendung auf Daten, die bei der Testkonstruktion nach dem Rasch-Modell vorliegen, die Voraussetzung normalverteilter Daten für die Varianzanalyse verletzt ist, da es sich um dichotome Daten handelt, und nur eine einzige Beobachtung pro Faktorkombination vorhanden ist, führten die Autoren 2009 und 2011 Simulationsstudien durch, um die Auswirkungen auf das Risiko 1. und 2. Art zu prüfen.
2009 zeigten Kubinger, Rasch und Yanagida, dass das simulierte Risiko 1. Art der dreifachen Varianzanalyse nahe am tatsächlichen Risiko 1. Art liegt, wenn kein Haupteffekt von A vorliegt. Liegt hingegen ein Haupteffekt vor, so entsteht ein zu hohes Risiko erster Art für die Wechselwirklung zwischen A und C. Laut der Studie hängt die Teststärke des F-Tests zwischen A und C von der Stichprobengröße ab und ist größer als die des Likelihood-Ratio-Tests nach Andersen. 2011 haben die Autoren in weiteren Simulationen die Auswirkungen der Verteilungen der Item- und Personenparameter auf das Risiko 1. und 2. Art untersucht. Es zeigt sich der Trend, dass das Risiko 1. Art bei

steigender Stichprobengröße und Aufgabenanzahl zunimmt, sowie bei großer Spannweite der Aufgabenparameter. Eingehalten wird das Risiko 1. Art, wenn die Aufgabenparameter im Intervall [-3 ,3] liegen und unimodal verteilt sind, die Personenparameter normalverteilt sind, mit einer Standardabweichung, die nicht größer als 1,5 ist, die Anzahl der Aufgaben nicht größer als 100 und die Anzahl der Personen nicht größer als 150 pro Teilgruppe ist. Die Teststärke des F-Tests wird größer, wenn die Modellverletzung Aufgaben mit mittlerer Schwierigkeit betrifft, wenn mehr Aufgaben von der Modellverletzung betroffen sind und je größer die Personenstichprobe ist. Für gehäufte Verletzungen im mittleren Fähigkeitsbereich lag die Teststärke im akzeptablen Bereich, wenn die Aufgabenzahl nicht größer als 40 und die Stichprobengröße zwischen 150 und 500 Personen pro Teilstichprobe lagen.

z-Test nach Fischer und Scheiblechner

Im Gegensatz zu den anderen beschriebenen Verfahren ist der z-Test nach Fischer und Scheiblechner (Fischer & Scheiblechner, 1970) mit der Schätzfunktion nach Wald (Wald, 1943) ein Test auf Itemebene. Wie der Likelihood-Ratio-Test nach Anderson prüft er die Itementsprechung. Der Datensatz wird hierfür im einfachsten Fall in zwei Gruppen eingeteilt und es wird für jedes einzelne Item geprüft, ob die Itemparameter in den beiden Gruppen gleich groß sind. Für zwei Gruppen lautet die Teststatistik für ein Item i:

$$T_{W_i} = \frac{(\hat{\beta}_i^{(1)} - \hat{\beta}_i^{(2)})^2}{{\sigma_i^{(1)}}^2 + {\sigma_i^{(2)}}^2}$$

wobei $\hat{\beta}_i^{(1)}$ der Schätzer für die Itemschwierigkeit des i-ten Items in Gruppe 1 ist und $\hat{\beta}_i^{(2)}$ der Schätzer für die Itemschwierigkeit des i-ten Items in Gruppe 2 ist, σ_i^2 steht für die jeweilige Varianz. Die dazugehörige Testgröße ist asymptotisch χ^2-verteilt mit einem Freiheitsgrad, die Wurzel daraus, versehen mit dem richtigen Vorzeichen, ist demnach standardnormalverteilt (Glas & Verhelst, 1995).

Martin-Löf-Test

Der Martin-Löf-Test (veröffentlicht von Martin-Löf 1973 in schwedischer Sprache, zitiert nach Koller, Alexandrowicz, & Hatzinger, 2012) überprüft die Itemhomogenität. Demnach werden für die Modellprüfung die Items in zwei Gruppen geteilt und nicht die Personen.

Wie beim Likelihood-Ratio-Test nach Andersen, werden beim Martin-Löf-Test Likelihoods miteinander verglichen. Man schätzt mit Hilfe der erschöpfenden Statistik getrennt für zwei Testhälften die Personenparameter. Nun wird geprüft, ob die Daten durch die Aufteilung in zwei Testhälften oder durch einen Gesamttest besser erklärt werden. Dies geschieht mit folgender Teststatistik T_{ML} für zwei Itemgruppen:

$$T_{ML} = -2 \ln \frac{\prod_{r=0}^{k} \left(\frac{n_r}{N}\right)^{n_r} * cL_0}{\prod_{t=0}^{k_1} \prod_{s=0}^{k_2} \left(\frac{n_{ts}}{N}\right)^{n_{ts}} * cL_1 * cL_2}$$

k ist die Gesamtzahl der Items, k_1 und k_2 sind die Anzahl der Items in den beiden Gruppen, N ist die Anzahl der Beobachtungen, n_r sind die Häufigkeiten der Personenscores im gesamten Test, n_t sind die Personenscores des ersten Subtests, n_s die des zweiten. cL_0 ist die bedingte Likelihood für alle Itemparameter gemeinsam, cL_1 und cL_2 sind die bedingten Likelihoods getrennt für die beiden Itemgruppen 1 und 2.
Diese Teststatistik ist assympotisch χ^2-verteilt mit $k_1 * k_2 - 1$ Freiheitsgraden. Für die Anwendung des Tests benötigt man eine Hypothese darüber, welche Itemgruppen unterschiedliche Eigenschaften/Fähigkeiten der Personen ansprechen. Die Nullhypothese lautet, dass dem Test eine Dimension zugrunde liegt. Wenn die beiden Testhälften die Daten jedoch besser erklären als der Gesamttest, dann wird die Alternativhypothese angenommen, dass der Test mehrere Dimensionen misst.
Das Teilungskriterium richtet sich bei diesem Modelltest nach den Items. Beim internen Teilungskriterium werden die Items in die Gruppe der einfachen und in die Gruppe der schwierigen Items geteilt. Bei externen Teilungskriterien werden die Items nach inhaltlichen Merkmalen aufgeteilt.
Da die Testgröße des Martin-Löf-Tests unter der Nullhypothese assymptotisch χ^2-verteilt ist, nähert sie sich erst ab einer großen Personenanzahl tatsächlich der χ^2-Verteilung an (Verhelst, 2001). Die Itemanzahl hat zusätzlich einen großen Einfluss darauf, ob der Martin-Löf-Test überhaupt signifikant werden kann. In einer Simulationsstudie von Verguts und De Boeck (2000) lag das Risiko 1. Art für 24 Items und 5 000 Personen bei 0%. Für weniger Items näherte sich das Risiko 1. Art den 5% an, auch schon bei kleineren Personenanzahlen.
Die Teststärke des Martin-Löf-Tests, die Zweidimensionalität eines Datensatzes zu erkennen, hängt von der Höhe der Korrelation der beiden Dimensionen ab und ist eingeschränkt, wenn diese hoch ist. Für die Praxis könnte dies ein Nachteil sein, da Dimensionen oft hoch korreliert sind (Verhelst, 2001).

4. Fehler 1. und 2. Art im Rasch-Modell

Bei der Modellprüfung im Rasch-Modell wird als Nullhypothese angenommen, dass das Modell gilt. Ist ein Modelltest signifikant, nimmt man die Alternativhypothese an und geht man davon aus, dass das Modell verletzt ist. Einen Fehler 1. Art im Rasch-Modell zu begehen, bedeutet somit, versehentlich anzunehmen, dass das Modell nicht gilt, obwohl es eigentlich gilt. In weiterer Folge würde das bedeuten, dass der Test umgestaltet, Items aus dem Test herausgenommen oder der Test nicht publiziert werden würde.

Wird ein Fehler 2. Art begangen, wird eine vorhandene Modellverletzung nicht erkannt. In der Praxis bedeutet dass, dass ein Test publiziert und als Rasch-Modell-konform bezeichnet werden würde, der Modellverletzungen enthält. Man würde somit davon ausgehen, dass der Test nach dem Rasch-Modell skaliert ist und fair misst, obwohl dies nicht der Fall ist.

Bei der Prüfung des Rasch-Modells könnten theoretisch beliebig viele Modelltets durchgeführt werden. So könnten als externe Teilungskriterien Personen nach allen existenten Eigenschaften aufgeteilt werden, in denen sich zumindest eine Person von den anderen Personen unterscheidet. Es könnten auf diese Weise unter Umständen so lange Tests durchgeführt werden, bis eine Nullhypothese verworfen wird und das Modell als nicht gültig eingestuft wird. Es ist daher wichtig, sinnvolle Modellprüfungen auszuwählen und das Modell als gültig im Bezug auf die überprüften Merkmale zu bezeichnen.

III. Simulationsstudie

5. Ziel der Simulationsstudie

Ziel der Studie ist es, das Risiko 1. Art und die Teststärke des Likelihood-Ratio-Tests nach Andersen, des Martin-Löf-Tests, des z-Tests nach Fischer und Scheiblechner und der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida für unterschiedliche Anzahlen von Personen und für verschiedene Effektstärken zu ermitteln und direkt zu vergleichen. So kann für die Praxis abgeschätzt werden unter welchen Bedingungen welche Modellgeltungstests geeignet sind.
Die ersten drei genannten Testgrößen sind unter der Nullhypothese nur asymptotisch χ^2-verteilt und es existiert keine Formel zur Berechnung des Risikos 2. Art. Es ist daher notwendig, mit Simulationsstudien zu prüfen, wie hoch die Risiken 1. und 2. Art tatsächlich sind.

6. Methoden

Die Simulation wurde mit der Library eRm (Mair, Hatzinger, & Maier, 2011) des Statistikprogramms R (R Development Core Team, 2012) durchgeführt. Um die Berechnungen zu beschleunigen, wurde zusätzlich die Library snow (Tierney, Rossini, Li, & Sevcikova, 2012) verwendet.

Simulationsaufbau

Es gab 15 verschiedene Simulationsszenarien, die auf die vier Modelltests, den Likelihood-Ratio-Test nach Andersen, den Martin-Löf-Test, den z-Test nach Fischer und Scheiblechner und die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida, angewandt wurden. Die Anzahl der Personen, das Ausmaß der Effektstärke und die Art der Modellverletzung bei den Simulationen zur Teststärke wurden dabei variiert, während die Anzahl von 20 Items und das Signifikanzniveau von 5% gleich blieben.
In der folgenden Abbildung (Abbildung 3) ist dargestellt, welche Szenarien simuliert wurden und welche Modelltests angewandt wurden:

<table>
<tr><th rowspan="2">Simulation</th><th colspan="3">Personenanzahl</th></tr>
<tr><td>100</td><td>200</td><td>300</td></tr>
<tr><td>Risiko 1. Art</td><td colspan="3">Likelihood-Ratio-Test nach Andersen
Dreifache Varianzanalyse nach Kubinger Rasch und Yanagida
z-Test nach Fischer und Scheiblechner
Martin-Löf-Test</td></tr>
<tr><td>Effektstärke ½ sd bei 1 Paar DIF</td><td colspan="3" rowspan="3">Likelihood-Ratio-Test nach Andersen
Dreifache Varianzanalyse nach Kubinger Rasch und Yanagida
z-Test nach Fischer und Scheiblechner</td></tr>
<tr><td>Effektstärke 1 sd bei 1 Paar DIF</td></tr>
<tr><td>Effektstärke 2 sd bei 1 Paar DIF</td></tr>
<tr><td>Multidimensionalität latente Korrelation 0,5</td><td colspan="3">Martin-Löf-Test
Likelihood-Ratio-Test nach Andersen</td></tr>
</table>

Abbildung 3: Simulationsszenarien

Jede Simulation wurde mit 10 000 Wiederholungen durchgeführt, mit Ausnahme der Simulationen unter Multidimensionalität, hier wurden 1 000 Wiederholungen durchgeführt.

Die Simulationsparameter wurden so gewählt, dass mögliche Beispiele, der in der Testkonstruktion praktisch vorkommenden Rahmenbedingungen für einen Test, widergespiegelt werden, damit die Ergebnisse bei zukünftigen Testkonstruktionen angewendet werden können. Es wurden für die Simulation 20 Items herangezogen. Diese Itemanzahl entspricht einer typischen Testlänge eines Leistungstests, der eine bestimmte Fähigkeit misst. Die Schwierigkeitsparameter dieser fiktiven Tests liegen zwischen -3 und +3, und stehen somit so im Verhältnis zu den Personenparametern, die normalverteilt mit Mittelwert null und Standardabweichung 1,5. sind. Auf diese Weise können ungefähr 2% aller fiktiven Personen keine einzige Aufgabe und 2% aller fiktiven Personen alle Aufgaben lösen. Zur besseren Differenzierungsfähigkeit im mittleren Fähigkeitsbereich gibt es mehr mittelschwierige als sehr einfache und sehr schwierige Aufgaben.

Alle verwendeten Befehle, sowie der gesamte Simulationscode sind im Anhang nachzulesen. In den folgenden Abschnitten werden die Details der verschiedenen Simulationen beschrieben.

Simulation zum Risiko 1. Art

Für die Simulation zum Risiko 1. Art wurden 10 000 Rasch-Modell-konforme Daten simuliert. Die Itemparameter waren auf den Bereich zwischen -3 und 3 verteilt, wobei es etwas mehr Items im mittleren Fähigkeitsbereich gab. Folgende Itemparameter wurden verwendet: -3, -2,5, -2, -1,5, -1,2, -0,9, -0,75, -0,5, -0,3, -0,1, 0,1, 0,3, 0,5, 0,75, 0,9, 1,2, 1,5, 2, 2,5, 3.

Die folgende Grafik (Abbildung 4) zeigt, wie die Itemparameter verteilt sind:

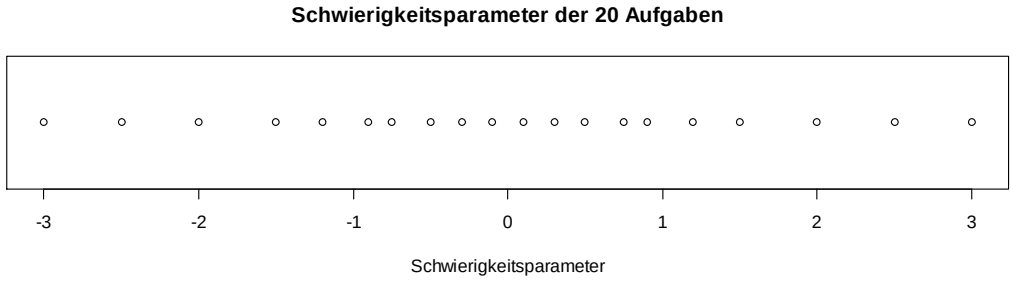

Abbildung 4: Schwierigkeitsparameter der 20 Aufgaben

Die Personenparameter wurden zufällig aus einer Normalverteilung mit Mittelwert null und Standardabweichung 1,5 gezogen. Die Simulation wurde für eine Personenanzahl von 100, 200 und 300 Personen durchgeführt. Das Risiko 1. Art wurde auf 5% festgesetzt. Die 10 000 Rasch-Modell-konformen Datensätze wurden anschließend mit dem Likelihood-Ratio-Test nach Andersen mit dem internen Teilungskriterium, mit dem Martin-Löf-Test mit einem zufälligen Teilungskriterium, mit dem z-Test nach Fischer und Scheiblechner und mit der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida mit zufälligen Gruppen ausgewertet. Ist ein Modelltest bei dieser Simulation signifikant, wird bei der Entscheidung für eine Hypothese ein Fehler 1. Art begangen.

Simulation zur Teststärke mit der Effektstärke einer halben Standardabweichung bei einem DIF-Paar

Auch bei der Simulation zur Teststärke wurde die Simulation für eine Personenanzahl von 100, 200 und 300 Personen durchgeführt, das Risiko 1. Art wurde auf 5% festgesetzt und 10 000 Wiederholungen durchgeführt. Jedoch mussten für die Simulation zur Teststärke Daten simuliert werden, die den Annahmen des Rasch-Modells widersprechen. Die Personenparameter wurden wiederum aus einer Normalverteilung mit Mittelwert null und Standardabweichung 1,5 gezogen. Die fiktiven Personen wurden jedoch anschließend je zur Hälfte zwei Gruppen zugeteilt. Für die eine Gruppe hatten die Items 1-20 der Reihenfolge nach folgende Schwierigkeiten:

3, -2,5, -2, -1,5, -1,2, -0,9, -0,75, -0,5, **-0,375**, -0,1, 0,1, **0,375**, 0,5, 0,75, 0,9, 1,2, 1,5, 2, 2,5, 3.

In der anderen Gruppe hatten die Items 1-20 der Reihenfolge nach jedoch folgende Schwierigkeiten:

3, -2,5, -2, -1,5, -1,2, -0,9, -0,75, -0,5, **0,375**, -0,1, 0,1, **-0,375**, 0,5, 0,75, 0,9, 1,2, 1,5, 2, 2,5, 3.

Wie man am **fett** gedruckten erkennen kann, sind Item 9 und Item 12 in den beiden Gruppen unterschiedlich schwierig und die Differenz der Itemschwierigkeiten liegt bei 0,75 was einer halben Standardabweichung der Personenparameter entspricht.

Für die beiden Gruppen wurde zunächst getrennt je ein Set von Rasch-Modell-konformen Daten simuliert. Das heißt im Fall von 100 Personen wurde für die ersten 50 Personen ein Modell mit der ersten Gruppe von Itemparametern geschätzt, ein getrenntes Modell wurde hingegen mit den zweiten 50 Personen und der zweiten Gruppe von Itemparametern geschätzt. Diese beiden, in sich Rasch-Modell-konformen Datensätze, wurden dann zusammengefügt und ergaben folglich einen Datensatz, der dem Rasch-Modell widerspricht. Stellt man sich vor, dass die ersten 50 Personen aus Großbritannien stammen und die zweiten 50 Personen US-amerikanischer Herkunft sind, so wäre es möglich, dass die Aufgaben 9 und 12 kulturspezifisch unterschiedlich schwierig sind. Item 12 wäre somit schwieriger für Personen aus Großbritannien, als für Personen aus den USA zu lösen, während Item 9 für US-Amerikaner/innen schwieriger zu lösen wäre als für Personen aus Großbritannien.

Die 10 000 auf diese Weise simulierten, nicht Rasch-Modell-konformen Datensätze wurden anschließend mit dem Likelihood-Ratio-Test nach Andersen mit den beiden Personengruppen als Teilungskriterium, mit dem z-Test nach Fischer und Scheiblechner und mit der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida, mit den Personengruppen als Gruppenfaktor, ausgewertet. Ein signifikantes Ergebnis bedeutet hier, dass die Modellverletzung erkannt wird.

Simulation zur Teststärke mit der Effektstärke einer Standardabweichung bei einem DIF-Paar

Die Simulation zur Teststärke mit der Effektstärke einer Standardabweichung wurde analog zu der mit einer halben Standardabweichung durchgeführt, mit dem Unterschied, dass diesmal die Items 7 und 14 von der Modellverletzung betroffen waren. Die DIF-Differenz von 1,5, die einer Standardabweichung der Personenparameter entspricht, ist wieder **fett** gedruckt. Für die erste Gruppe lauten die Itemparameter der Items 1-20 in der Reihenfolge somit:

-3, -2,5, -2, -1,5, -1,2, -0,9, **-0,75**, -0,5, -0,3, -0,1, 0,1, 0,3, 0,5, **0,75**, 0,9, 1,2, 1,5, 2, 2,5, 3.

In der zweiten Gruppe lauten sie hingehen wie folgt:

-3, -2,5, -2, -1,5, -1,2, -0,9, **0,75**, -0,5, -0,3, -0,1, 0,1, 0,3, 0,5, **-0,75**, 0,9, 1,2, 1,5, 2, 2,5, 3.

Simulation zur Teststärke mit der Effektstärke von zwei Standardabweichungen bei einem DIF-Paar

Die Simulation zur Teststärke mit der Effektstärke von zwei Standardabweichungen wurde analog zu der mit einer halben und der mit einer Standardabweichung durchgeführt, wobei hier jedoch die Items 4 und 17 von der Modellverletzung betroffen waren. Die DIF-Differenz von 3, die zwei Standardabweichungen der Personenparameter entspricht, ist wieder **fett** gedruckt. Für die erste Gruppe lauten die Itemparameter der Items 1-20 in der Reihenfolge somit:

-3, -2,5, -2, **-1,5**, -1,2, -0,9, -0,75, -0,5, -0,3, -0,1, 0,1, 0,3, 0,5, 0,75, 0,9, 1,2, **1,5**, 2, 2,5, 3.

In der zweiten Gruppe lauten sie hingehen wie folgt:

-3, -2,5, -2, **1,5**, -1,2, -0,9, -0,75, -0,5, -0,3, -0,1, 0,1, 0,3, 0,5, 0,75, 0,9, 1,2, **-1,5**, 2, 2,5, 3.

Die DIF-Differenzen wurden aus inhaltlichen Gründen so festgelegt. Aus zwei Gründen ist dies naheliegend: Einerseits ermöglichen die besonderen Eigenschaften des Rasch-Modells, dass Item- und Personenparameter auf derselben Skala gemessen werden, andererseits werden Fähigkeitsbereiche in psychologisch-diagnostischen Verfahren häufig über Einheiten von Standardabweichungen definiert. Die Effektstärke in Standardabweichungen der Personenparameter zu messen, erleichtert auf diese Weise ihre Interpretierbarkeit. So kann (je nach Lage) eine zwei Paar DIF-Differenz von zwei Standardabweichungen bedeuten, dass die beiden Items in einer Teilgruppe nicht mehr dem überdurchschnittlich schwierigen Bereich sondern dem unterdurchschnittlich schwierigem Bereich angehören und umgekehrt. Von einem guten Modelltest würde man erwarten, dass mindestens solche Modellverletzungen erkannt werden.

Simulation zur Teststärke unter Mehrdimensionalität

Um die Teststärke des Martin-Löf-Tests zu simulieren, ist die Modellverletzung über DIF nicht geeignet, da er im Gegensatz zu den anderen drei Modelltests nicht darauf ausgelegt ist, diese Modellverletzung zu erkennen. Daher wurden mehrdimensionale

Daten simuliert. Den Items wurden zwei verschiedene latente Dimensionen zugrundegelegt, deren Korrelationskoeffizient 0,5 betrug. Die Itemparameter wurden so gewählt, dass die beiden Itemgruppen, denen je eine latente Dimension zugrunde liegt, die gleichen Schwierigkeitsparameter aufwiesen. Die Parameter der Items in beiden Gruppen waren -3, -2, -1, -0,4, -0,1, 0,1, 0,4, 1, 2 und 3. In der folgenden Grafik (Abbildung 5) sind die Itemparameter dargestellt. Man sieht, dass etwas mehr Items im mittleren Fähigkeitsbereich simuliert wurden.

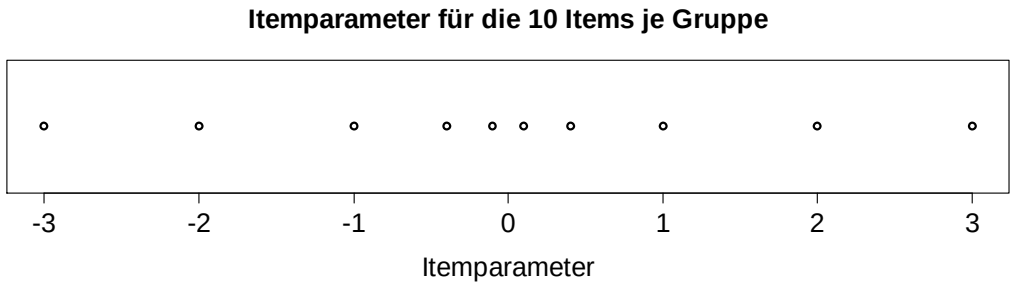

Abbildung 5: Verteilung der Itemparameter der 10 Items je Gruppe

Die Personenparameter wurden, wie bei den anderen Simulationen, aus einer Normalverteilung mit Mittelwert null und Standardabweichung 1,5 gezogen und es sollten für 100, 200 und 300 Personen 1 000 Simulationen durchgeführt werden. Das Risiko 1. Art wurde auf 5% festgesetzt.

Zur Veranschaulichung könnte man sich bei dieser Simulation vorstellen, dass 10 Items des fiktiven Tests, physikalisches Allgemeinwissen prüfen und 10 Items biologisches Allgemeinwissen. Beide Aufgabentypen decken den gesamten Schwierigkeitsbereich ab. Der latente Korrelationskoeffizient zwischen den beiden beträgt 0,5, da biologisches und physikalisches Wissen mittelmäßig stark zusammenhängt. Die Modelltests sollen nun erkennen, dass der Test nicht eindimensional misst.

Die auf diese Weise simulierten Daten wurden mit dem Martin-Löf-Test und mit dem Likelihood-Ratio-Test nach Andersen ausgewertet. Das Teilungskriterium für den Martin-Löf-Test waren die beiden Itemgruppen, die je eine latente Dimension repräsentierten. Der Likelihood-Ratio-Test wurde anhand des internen Teilungskriteriums durchgeführt.

Auswertungsmethoden der Simulation

Bei der Auswertung des **Likelihood-Ratio-Tests nach Andersen** und des **Martin-Löf-Tests** wurde bei der Simulation zum Risiko 1. Art gezählt, wie oft der Test signifikant

war. Das aktualisierte Risiko 1. Art ergibt sich aus dem Quotienten der gezählten signifikanten Ergebnisse und der Anzahl an Simulationsdurchgängen, für die eine Schätzung möglich war. In einigen wenigen Fällen konnten die Parameter nicht geschätzt werden, weshalb sich die Anzahl an Simulationsdurchgängen geringfügig verringert. Bei der Simulation zur Teststärke wurde genauso vorgegangen, wobei hier der Quotient aus signifikanten Tests und der Anzahl der Simulationsdurchgänge, für die eine Schätzung möglich war, die Teststärke angibt.

Beim **z-Test nach Fischer und Scheiblechner** wird jedes einzelne Item auf dem Niveau von 5% auf Signifikanz geprüft. Bei 20 Items und 10 000 Wiederholungen einer bestimmten Simulationsbedingung werden somit pro Bedingung 200 000 Signifikanztests durchgeführt. Wiederum muss berücksichtigt werden, dass nicht für alle Items eine Schätzung und somit ein Prüfen der Signifikanz möglich ist. Bei der Simulation zum Risiko 1. Art ist das aktualisierte Risiko 1. Art jener Anteil an Items, die signifikant sind, an den Items, für die eine Signifikanzprüfung möglich ist. Bei der Simulation zur Teststärke muss für den z-Test nach Fischer und Scheiblechner berücksichtigt werden, dass nur bestimmte Items von der Modellverletzung betroffen sind. Daher werden für die Berechnung der simulierten Teststärke pro Simulationswiederholung nur zwei Signifikanztests berücksichtigt, nämlich die jener beiden Items, deren Schwierigkeit in den beiden Gruppen vertauscht wurde. Bei 10 000 Wiederholungen werden so pro Bedingung 20 000 Signifikanztests durchgeführt. Die aktualisierte Teststärke ist der Anteil an signifikanten Items, die das Modell verletzen, an den Items, die das Modell verletzen und für die eine Schätzung möglich ist.

Bei der Auswertung der 10 000 Datensätze mit der **dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida** der Form (A $\succ$ ***B***) x C wurde ausgezählt, wie häufig der Effekt der Gruppenzuordnung (Effekt A), der Effekt der Personen, die in die Gruppen genestet sind (Effekt B(A)), der Effekt der Items (Effekt C) und die Wechselwirkung zwischen der Gruppenzuordnung und den Items (Effekt AC) signifikant waren. Wichtig ist jedoch nur, ob der Wechselwirkungseffekt AC signifikant ist und dass der Haupteffekt A nicht signifikant ist. Simulationsdurchgänge mit signifikantem Haupteffekt A werden von der Auswertung ausgeschlossen, da sich in der Simulationsstudie von Kubinger, Rasch und Yanagida (2009) zeigte, dass ein signifikanter Haupteffekt A zu einem überhöhten Risiko 1. Art bei dem F-Test für die Wechselwirkung AC führt. Der Anwendungsbereich der dreifachen Varianzanalyse von Kubinger, Rasch und Yanagida wird somit auf jene Fälle begrenzt, für die der Haupteffekt A nicht signifikant ist.

Der Quotient aus der Anzahl der signifikanten Wechselwirkungen zwischen A und C und der Anzahl an Simulationsdurchgängen, für die der Haupteffekt A nicht signifikant war, liefert das aktualisierte Risiko 1. Art bei der Simulation zum Risiko 1. Art. Bei der Simulation zur Teststärke liefert dieser Quotient die Teststärke.

7. Ergebnisse

Im Folgenden werden die Ergebnisse in Tabellen dargestellt. Es wird für die verschiedenen Personenanzahlen und für jeden Modelltest aufgelistet, wie viele Signifikanztests vollständig berechnet werden konnten (Anzahl gültiger Durchgänge/Items), wie viele davon signifikant waren und wie hoch demnach das aktualisierte Risiko 1. Art beziehungsweise die Teststärke ist. Die Anzahl gültiger Durchgänge beim Likelihood-Ratio-Test nach Andersen (LR-Test) weicht dann von 10 000 ab, wenn das Rasch-Modell oder der Test mit den simulierten Datensätzen nicht berechnet werden konnten. Beim z-Test nach Fischer und Scheiblechner weicht die Zahl von 200 000 beziehungsweise von 20 000 ab, wenn das Rasch-Modell oder der z-Test nach Fischer und Scheiblechner mit den simulierten Datensätzen nicht berechnet werden konnte. Bei der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida (3VA), kann aus der Differenz von 10 000 und der Anzahl gültiger Durchgänge abgelesen werden, wie oft der Haupteffekt A signifikant war.

Simulation zum Risiko 1. Art

In Tabelle 1 sind die Ergebnisse der Simulation zum Risiko 1. Art zusammengefasst. Für den Fall von 100 Personen konnte für 9 998 Simulationen das Rasch-Modell berechnet und der Likelihood-Ratio-Test nach Andersen durchgeführt werden. 360 dieser Tests waren signifikant. Das entspricht einem Risiko erster Art von 3.6%. Alle weiteren Ergebnisse sind der Tabelle zu entnehmen.

Tabelle 1: Ergebnisse der Simulation zum Risiko 1. Art

Personen-anzahl		LR-Test	3VA	z-Test	Martin-Löf-Test
100	Anzahl gültiger Durchgänge/Items	9998	9999	185141	9999
	Anzahl signifikanter Ergebnisse	360	454	7409	0
	Risiko 1. Art	0,0360	0,0454	0,0400	0
200	Anzahl gültiger Durchgänge/Items	10000	10000	194549	10000
	Anzahl signifikanter Ergebnisse	394	563	8567	0
	Risiko 1. Art	0,0394	0,0563	0,0440	0
300	Anzahl gültiger Durchgänge/Items	10000	10000	197543	10000
	Anzahl signifikanter Ergebnisse	457	363	9148	0
	Risiko 1. Art	0,0457	0,0363	0,0463	0

Wie man der Tabelle entnehmen kann, wird das Risiko 1. Art vom Likelihood-Ratio-Test nach Andersen, von der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida sowie vom z-Test nach Fischer und Scheiblechner ungefähr eingehalten und liegt für diese drei Tests zwischen 3.6 und 5.6%

Der Martin-Löf-Test wurde unter diesen Simulationsbedingungen kein einziges Mal signifikant. Bei genauerer Betrachtung der p-Werte des Martin-Löf-Test (Tabelle 2: Deskriptive Statistik der p-Werte des Martin-Löf-Tests) sieht man, dass diese weit davon entfernt sind, das Signifikanzniveau von 0,05 zu unterschreiten. Je größer jedoch die Stichprobe wird, desto eher kommen auch niedrigere p-Werte vor.

Tabelle 2: Deskriptive Statistik der p-Werte des Martin-Löf-Tests: Minimum, 1. Quantil, Median, 2. Quantil, Maximum, Mittelwert und Standardabweichung der p-Werte

	min	1.Quantil	Median	2.Quantil	Max	$\bar{p}$	sd
100 Personen	0,7603	1	1	1	1	0,9992	0,0072
200 Personen	0,5697	0,9995	1	1	1	0,9972	0,0146
300 Personen	0,4071	0,9988	0,9999	1	1	0,9949	0,0229

Simulation zur Teststärke mit der Effektstärke eines DIF-Paars von 0,75

In Tabelle 3 sind die Ergebnisse der Simulation zur Teststärke mit einer Effektstärke eines DIF-Paars von 0,75 zusammengefasst.

Die Teststärke des z-Tests nach Fischer und Scheiblechner ist sehr niedrig und steigt von ca. 0,05 bei 100 Personen auf 0,06 bei 300 Personen. Sie liegt somit nur wenig über dem Risiko 1. Art. Die Teststärke des Likelihood-Ratio-Tests und der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida sind ähnlich groß, wobei die Power der dreifachen Varianzanalyse immer etwas größer ist. Mit der getesteten Personenanzahl steigt die Power dieser beiden Modeltests von über 0,2 auf um die 0,6 an.

Tabelle 3: Ergebnisse der Simulation zur Teststärke mit einem DIF-Paar von 0,75

Personen-anzahl		LR-Test	3VA	z-Test
100	Anzahl gültiger Durchgänge/Items	9980	9996	20000
	Anzahl signifikanter Ergebnisse	2238	2523	942
	Teststärke	0,2242	0,2524	0,0471

200	Anzahl gültiger Durchgänge/Items	9995	10000	20000
	Anzahl signifikanter Ergebnisse	4176	4828	1190
	Teststärke	0,4178	0,4828	0,0595
300	Anzahl gültiger Durchgänge/Items	5990 10000	9998	20000
	Anzahl signifikanter Ergebnisse	5990 10000	6264	1218
	Teststärke	0,599	0,6265	0,0609

Simulation zur Teststärke mit der Effektstärke eines DIF-Paars von 1,5

In Tabelle 4 sind die Ergebnisse der Simulation zur Teststärke mit einer Effektstärke eines DIF-Paars von 1,5 zusammengefasst.

Es zeigt sich, dass ab einer Effektstärke von einer Standardabweichung, die Power des Likelihood-Ratio-Tests und der dreifachen Varianzanalyse von Kubinger, Rasch und Yanagida, die häufig willkürlich festgelegte Größe von 0,8 überschreitet und somit in einem gewünschten Bereich liegt, sogar für kleine Stichproben von 100 Personen. Für 200 und 300 Personen liegt die Power schon sehr nahe bei 1, wobei die des Likelihood-Ratio-Tests etwas niedriger liegt.

Die Power des z-Tests nach Fischer und Scheiblechner hingegen ist niedrig: Sie liegt zwischen 0,06 bei 100 Personen und knapp 0,14 bei 300 Personen.

Tabelle 4: Ergebnisse der Simulation zur Teststärke mit einem DIF-Paar von 1,5

Personen-anzahl		LR-Test	3VA	z-Test
100	Anzahl gültiger Durchgänge/Items	9984	9998	19996
	Anzahl signifikanter Ergebnisse	8290	8768	1252
	Teststärke	0,8303	0,877	0,0626
200	Anzahl gültiger Durchgänge/Items	9997	10000	20000
	Anzahl signifikanter Ergebnisse	9886	9941	2029
	Teststärke	0,9889	0,9941	0,1015
300	Anzahl gültiger Durchgänge/Items	10000	9999	20000
	Anzahl signifikanter Ergebnisse	9998	9998	2748
	Teststärke	0,9998	0,9999	0,1374

Simulation zur Teststärke mit der Effektstärke eines DIF-Paars von 3

In Tabelle 5 sind die Ergebnisse der Simulation zur Teststärke mit einer Effektstärke eines DIF-Paars von 3 zusammengefasst.
In jedem Simulationsdurchgang wurden die Modellverletzung vom Likelihood-Ratio-Test nach Andersen und von der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida erkannt. Die Power dieser beiden Tests liegt somit für alle Personenanzahlen bei 1. Der z-Test nach Fischer und Scheiblechner erkennt diese sehr große Modellverletzung von zwei Standardabweichungen vergleichsweise wieder schlechter. Seine Power liegt zwischen 0,28 für 100 Personen und 0,77 bei 300 Personen.

Tabelle 5: Ergebnisse der Simulation zur Teststärke mit einem DIF-Paar von 3

Personen-anzahl		LR-Test	3VA	z-Test
100	Anzahl gültiger Durchgänge/Items	9987	9996	19996
	Anzahl signifikanter Ergebnisse	9987	9996	5657
	Teststärke	1	1	0,2829
200	Anzahl gültiger Durchgänge/Items	9998	10000	20000
	Anzahl signifikanter Ergebnisse	9998	10000	11023
	Teststärke	1	1	0,5512
300	Anzahl gültiger Durchgänge/Items	10000	9998	20000
	Anzahl signifikanter Ergebnisse	10000	9998	15304
	Teststärke	1	1	0,7652

Simulation zur Multidimensionalität

Die Teststärke des Likelihood-Ratio-Tests nach Andersen entspricht in etwa dem aktualisierten Risiko 1. Art des Tests, die des Martin-Löf-Tests liegt daüber und wächst mit steigender Personenanzahl deutlich an. Für 100 Personen ist die Teststärke sehr niedrig mit 0,03, für 200 Personen liegt sie bei 0,32 und für 300 Personen bei 0,7. In Tabelle 6 sind die Ergebnisse der Simulation zur Multidimensionalität zusammengefasst.

Tabelle 6: Ergebnisse der Simulation zur Multidimensionalität

Personen-anzahl		Martin-Löf-Test	LR-Test
100	Anzahl gültiger Durchgänge/Items	994	1000
	Anzahl signifikanter Ergebnisse	32	28
	Teststärke	0,0322	0,028
200	Anzahl gültiger Durchgänge/Items	1000	1000
	Anzahl signifikanter Ergebnisse	315	39
	Teststärke	0,315	0,039
300	Anzahl gültiger Durchgänge/Items	1000	1000
	Anzahl signifikanter Ergebnisse	699	45
	Teststärke	0,699	0,045

8. Diskussion

Wie zu erwarten, haben die vier Modelltests unterschiedliche Stärken und Schwächen und sind, je nach Simulationsbedingung, unterschiedlich gut geeignet, Modellverletzungen aufzudecken. Wie üblich führen größere Effektstärken und größere Personenanzahl allgemein zu einer größeren Teststärke.

Für die hier simulierten Bedingungen halten der Likelihood-Ratio-Test nach Andersen, die dreifach Varianzanalyse nach Kubinger, Rasch und Yanagida und der z-Test nach Fischer und Scheiblechner das Risiko 1. Art in etwa ein. Das Risiko 1. Art des Martin-Löf-Tests liegt hingegen bei 0%. Es zeigt sich bei diesem Test allgemein die Tendenz, dass er erst bei größeren Stichproben signifikant wird. So ist beispielsweise seine Teststärke unter Multidimensionalität sehr stark von der Stichprobegröße abhängig und liegt bei einer latenten Korrelation von 0,5 erst im Fall von 300 Personen bei 70%.

Der Likelihood-Ratio-Test nach Andersen ist mit dem internen Teilungskriterium hingegen nicht in der Lage, Multidimensionalität zu erkennen. Dieses Ergebnis stimmt mit den Simulationen von Stelzl (1979) und Alexandrowicz (2002) überein und ist zu erwarten, da das interne Teilungskriterium Zweidimensionalität nicht abbilden kann.

Das Ergebnis des fast nicht vorhandenen Fehlers 1. Art beim Martin-Löf-Test passt zu den Ergebnissen der Simulationsstudie von Verguts und De Boeck (2000), dort liegt das Risiko 1. Art für 24 Items und 5 000 Personen ebenfalls bei 0% und sie berichten, dass die Itemanzahl beim Martin-Löf-Test stark beeinflusst, wie oft er signifikant wird, wobei längere Tests zu weniger Signifikanz führen.

Für die in dieser Arbeit simulierten Bedingungen haben der Likelihood-Ratio Test nach Andersen und die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida bei entsprechend großer Effektstärke (ab einer Standardabweichung von 1,5 bei einem DIF-Paar) auch schon für 100 Personen eine Teststärke von über 80%. Die Teststärke der dreifachen Varianzanalyse liegt dabei etwas über der des Likelihood-Ratio-Tests nach Andersen. Im Falle der Teilung nach einem externen Kriterium sollte daher eher die dreifache Varianzanalyse angewandt werden.

Die Teststärke des z-Tests nach Fischer und Scheiblechner wird erst bei sehr großer Effektstärke und Personenanzahl größer und liegt für zwei Standardabweichungen der Personenparameter bei einem DIF-Paar und 300 Personen immer noch unter 80%.

Implikationen für die Testkonstruktion

Die Simulationen haben klar gezeigt, dass die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida für die hier simulierten Szenarien immer statt dem Likelihood-Ratio-Test nach Andersen eingesetzt werden könnte, wenn der Datensatz bei der Modellprüfung nach einem externen Teilungskriterium geteilt werden soll.
Für die Teilung nach dem internen Kriterium ist die Varianzanalyse nicht geeignet, da Kubinger, Rasch und Yanagida bereits 2009 in ihrer Simulationsstudie feststellten, dass für einen signifikanten Haupteffekt A, das Risiko 1. Art erhöht ist. Der Haupteffekt A wird aber insbesondere dann signifikant werden, wenn er die Teilung der Personen nach ihrer Leistung in zwei Gruppen repräsentiert. Für die Prüfung nach dem internen Teilungskriterium sollte daher der Likelihood-Ratio-Test nach Andersen herangezogen werden. Um Multidimensionalität erkennen zu können, sollte zusätzlich der Martin-Löf-Test eingesetzt werden. Die Teststärke des Martin-Löf-Tests ist allerdings für kleine Personenanzahlen noch zu gering um Multidimensionalität zu erkennen.
Der z-Test nach Fischer und Scheiblechner kann zusätzlich angewandt werden, um kritische Items zu identifizieren, es muss jedoch damit gerechnet werden, dass die Wahrscheinlichkeit diese tatsächlich zu identifizieren, für kleine Stichproben und geringe Effekte klein ist.

Einschränkungen und Ausblick auf zukünftige Forschung

Zukünftige Simulationen können zeigen, wie sich das Kombinieren von mehreren Modellverletzungen auf das Risiko 1. und 2. Art auswirkt, so könnten fiktive Tests simuliert werden, die sowohl mehrdimensional messen als auch DIF aufweisen.
In der vorliegenden Arbeit werden Simulationen für einen Test mit 20 Aufgaben durchgeführt, dessen Parameter einer bestimmten Verteilung folgen und ungefähr 2% der Personen keine und 2% alle Aufgaben lösen können. Die Ergebnisse können nicht auf andere Testlängen und Verteilungen der Item- und Personenparameter generalisiert werden.
Im Prinzip gäbe es beliebig viele zusätzliche Simulationsszenarien und eine Reihe von weiteren Modelltests, die in Bezug auf ihr Risiko 1. und 2. Art verglichen werden könnten. Um davon eine Auswahl an relevanten Szenarien treffen zu können, könnten Simulationen noch näher an der direkten Testkonstruktion und spezifisch auf diese zugeschnitten vorgenommen werden. Auf diese Weise könnte eine optimale, wenn auch nicht perfekte, Anzahl an Personen bestimmt werden, die für die Skalierung herangezogen werden soll. Verschiedene Modelltests könnten für jeden spezifischen

Fall miteinander verglichen und die Besten zur Prüfung des Modells ausgewählt werden. In Testmanualen könnte die mittels Simulation nunmehr festgestellte Teststärke der Modelltests bei gegebener Effektstärke zusätzlich berichtet werden, um die Güte der Kalibrierung näher zu quantifizieren. Diese Maßnahme würde nicht nur zusätzliche wissenschaftliche Maßstäbe setzten, sondern auch das Bewusstsein über die teilweise zu geringen Teststärken der Modelltests erhöhen.

9. Zusammenfassung

Ausgehend vom dem Wunsch, geeignete psychologisch-diagnostische Verfahren zu entwickeln, um Menschen bestmöglich beraten zu können, stellt sich die Frage nach der Güte dieser Verfahren. Eines der entscheidenden Kriterien für ein gutes Verfahren ist dessen messtheoretische Eignung. Georg Rasch entwickelte ein Testmodell, bei dessen Gültigkeit von messtheoretischer Eignung ausgegangen werden kann. Die Gültigkeit des Rasch-Modells kann mit verschiedenen Modelltests überprüft werden, die unterschiedlich gut in der Lage sind, Modellverletzungen zu erkennen.

In der vorliegenden Arbeit wurden vier Modelltests, der Likelihood-Ratio-Test nach Andersen, die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida, der z-Test nach Fischer und Scheiblechner und der Martin-Löf-Test in Bezug auf verschiedene Modellverletzungen miteinander verglichen. Es wurden Rasch-Modell-konforme Daten simuliert um das Risiko 1. Art zu aktualisieren und Daten mit Modellverletzungen zur Ermittlung der Teststärke. Die Personenparameter waren normalverteilt mit Mittelwert 0 und Standardabweichung 1,5. Die Itemparameter variierten zwischen -3 und +3, wobei es geringfügig mehr Items im mittleren Fähigkeitsbereich gab. Es wurden Daten für 20 Items und 100, 200 und 300 Personen simuliert. Für die Simulation zum aktuellen Risiko 1. Art und für die simulierten Modellverletzungen mit DIF-Paaren, die einen Abstand zwischen 0,75 und 2 aufwiesen, wurden 10 000 Datensätze generiert und ausgewertet. Für die simulierte Modellverletzung mit Multidimensionalität zweier latenter Variablen mit einer Korrelation von 0,5 waren es 1 000 Datensätze.

Es zeigte sich, dass der Likelihood-Ratio-Test nach Andersen, die dreifache Varianzanalyse nach Kubinger, Rasch und Yanagida und der z-Test nach Fischer und Scheiblechner das Risiko 1. Art einhalten. Das Risiko 1. Art des Martin-Löf-Tests lag bei 0% (ein diesbezüglich ähnliches Ergebnis, allerdings mit 5 000 Personen und 24 Items, publizierten Verguts und De Boeck, 2000).

Die Teststärke der dreifachen Varianzanalyse nach Kubinger, Rasch und Yanagida lag für die hier simulierten Bedingungen etwas höher als die des Likelihood-Ratio-Tests nach Andersen, sodass diese, unter den hier simulierten Bedingungen, als geeigneter für die Modellprüfung erscheint, wenn nach einem externen Kriterium geteilt werden soll.

Wie erwartet hing die Teststärke aller Verfahren von der Personenanzahl und der Effektstärke ab. Für die dreifache Varianzanalyse und den Likelihood-Ratio-Test nach Andersen ergab sich ab einer Effektstärke von 1,5 bei einem DIF-Paar bereits bei 100 Personen eine Teststärke von über 80%.

Die Teststärke des z-Tests nach Fischer und Scheiblechner fiel gering aus, sie lag in allen simulierten Bedingungen unter 80%.

Wie nach den Ergebnissen von Stelzl (1979) und Alexandrowicz (2002) erwartet, war der Martin-Löf-Test prinzipiell in der Lage Mehrdimensionalität zu erkennen, während der Likelihood-Ratio-Test nach Andersen diese Modellverletzung nicht erkennen konnte, was konzeptionell nachvollziehbar ist. Allerdings war die Teststärke des Martin-Löf-Tests sehr stark von der Personenanzahl abhängig. Sie lag für 100 Personen nur bei 3% und erreichte bei 300 Personen 70%.

In Zukunft könnten Simulationen auf spezifische Testkonstruktionen zugeschnitten werden, um die geeignetsten Modelltests auszuwählen. Die Teststärke der Verfahren könnte bei gegebener Effektstärke im Manual berichtet werden.

IV. Verzeichnisse

10. Literaturverzeichnis

Alexandrowicz, R. (2002). *Die Teststärke des Likelihood-Quotienten-Tests nach Andersen bei der Überprüfung der Modellgültigkeit des dichotomen logistischen Modells nach Rasch.* Wien: unveröffentlichte Dissertation Universität Wien.

Andersen, E. B. (1973). A goodness of fit test for the Rasch model. *Psychometrika*, *38*, 123-140.

Fischer, G. H. (1974). *Einführung in die Theorie psychologischer Tests Grundlagen und Anwendungen.* Bern: Hans Huber.

Fischer, G. H. (1995). Derivations of the Rasch Model. In G. H. Fischer, & I. W. Molenaar (Hrsg.), *Rasch Models Foundations, Recent Developments, and Applications* (S. 69-95). New York: Springer.

Fischer, G. H., & Scheiblechner H. H. (1970). Algorithmen und Programme für das probabilistische Testmodell von Rasch. *Psychologische Beiträge, 12,* 23-51.

Glas, C. A., & Verhelst, N. D. (1995). Testing the Rasch Model. In G. H. Fischer, & I. W. Molenaar (Hrsg.), *Rasch Models Foundations, Recent Developments, and Applications* (S. 69-95). New York: Springer.

Hogg, R. V., McKean, J. W., & Craig, A. T. (2005). *Introduction to mathematical statistics*. New York: Pearson.

Koller, I., Alexandrowicz, R., & Hatzinger, R. (2012). *Das Rasch-Modell in der Praxis Eine Einführung mit eRm.* Wien: Facultas.

Kubinger, K. D. (2009). *Psychologische Diagnostik Theorie und Praxis psychologischen Diagnostizierens.* Göttingen: Hogrefe.

Kubinger, K. D., Rasch, D., & Yanagida, T. (2009). On designing data-sampling for Rasch model calibrating an achievement test. *Psychology Science Quarterly, 51* , S. 270-384.

Kubinger, K. D., Rasch, D., & Yanagida, T. (2011). A new approach for testing the Rasch model. *Educational Research and Evaluation, 17*, 321-333.

Mair, P., Hatzinger, R., & Maier, M. (2011). *eRm: Extended Rasch Modeling.* R package version 0.14-0. http://CRAN.R-project.org/package=eRm.

R Development Core Team. (2012). *R: A language and environment for statistical computing.* Vienna: R Foundation for Statistical Computing http://www.R-project.org/.

Rost, J. (2004). *Lehrbuch Testteorie - Testkonstruktion.* Bern: Hans Huber.

Stelzl, I. (1979). Ist der Modelltest des Rasch-Modells geegnet, Homogenitätshypothesen zu prüfen? Ein Bericht über Simulationsstudien mit inhomogenen Daten. *Zeitschrift für experimentelle und angewandte Psychologie, 26*, 652-672.

Tierney, L., Rossini, A. J., Li, N., & Sevcikova, H. (2012). *snow: Simlpe Network of Workstations.* R package version 0.3-10. http://CRAN.R-project.org/package=snow.

Verguts, T., & De Boeck, P. (2000). A note on the Martin-Löf test for unidimensionality. *Methods of Psychological Research Online*, *5*, 77-82.

Verhelst, N. (2001). Testing the unidimensionality assumption of the Rasch. *Methods of Psychological Research Online*, 6, 231-271.

Wald, A. (1943). Tests of statistical hypothesis concerning several parameters when the number of observations is large. *Transactions of the American Mathematical Society*, *54*, 426–482.

11. Tabellenverzeichnis

12. Abbildungsverzeichnis

13. R-Code

Verwendete R-Pakete

```
library(eRm)
library(snow)
```

R-Code für die Simulation zum Fehler 1. Art

Funktionen: createAlpha, alphaRM, auswertenAlpha

```
# ++++++++++++++ createAlpha ++++++++++++++++++++++++++++++++

createAlpha<-function(ppar,ipar,wh) {
# Funktion erzeugt Raschdaten und Verzeichnisse
# ppar: Vektor mit Personenparametern
# ipar: Vektor mit Itemparametern, Achtung: Muss in diesem Programm
# Länge 20 haben
# wh: Anzahl der Simulationen

v <- 1:wh # Vektor der die Simulationsnummer angibt
panz<-length(ppar) # Personenanzahl

# Verzeichnisse erstellen, in die gespeichert wird
dir.create("/home/futschek/RMdat")
dir.create("/home/futschek/RMdat/alpha")
dir.create("/home/futschek/RMdat/alpha/ipar")
dir.create("/home/futschek/RMdat/alpha/p")
dir.create("/home/futschek/RMdat/alpha/Auswertung")
dir.create("/home/futschek/RMdat/alpha/s")
dir.create("/home/futschek/RMdat/alpha/A")
dir.create("/home/futschek/RMdat/alpha/M")
dir.create("/home/futschek/RMdat/alpha/V")
dir.create("/home/futschek/RMdat/alpha/Z")

#Dokumentation der vorgegebenen Parameter
write.table(ipar, file = "/home/futschek/RMdat/alpha/ipar/ipar")
write.table(ppar, file =
paste("/home/futschek/RMdat/alpha/p/p",panz,sep="")) # Speichern

# Simulation von "wh" raschkonformen Datensätzen und Speicherung -----
for (i in 1:wh) {
    simrm <- sim.rasch(ppar,ipar)

    write.table(simrm, file =
paste("/home/futschek/RMdat/alpha/s/s",v[i],sep=""))
    }
}
```

```
# ++++++++++++++++++++++++++ alphaRM ++++++++++++

alphaRM <- function(g,ppar,ipar) {
     # die Funktion führt die Modellprüfungen durch

    require(eRm)

    panz<-length(ppar) # Personenanzahl

    # Aufrufen eines Datensatzes
    # g <-
read.table(paste("/home/futschek/RMdat/alpha/s/s",v[i],sep=""))

    a <- read.table(g)

    # RM-Berechnung

    b <- try(RM(a))

    #Modelltests

    if (is(b,"dRm"))  {
    c <- try(LRtest(b, splitcr = "median"))                  # LR-Test
    d <- MLoef(b, splitcr = sample(c(rep(1:2,each=10)),20)) # Martin-
#Löf-Test
    d1 <- c(d$LR,d$df,d$p)
    e <- try(Waldtest(b)) } else  {                          # Waldtest
    c <- NA
    d <- NA
    d1<-c(NA,NA,NA)
    e <- NA
    }

    if (is(c,"LR"))  c1 <- c(c$LR,c$df,c$p) else c1<-c(NA,NA,NA)
    if (is(e,"wald")) e1 <- e$coef.table else e1<- NA

    write.table(c1, file =
paste("/home/futschek/RMdat/alpha/A/A",gsub("s","", g),sep=""))
    write.table(d1, file =
paste("/home/futschek/RMdat/alpha/M/M",gsub("s","", g),sep=""))
    write.table(e1, file =
paste("/home/futschek/RMdat/alpha/Z/Z",gsub("s","", g),sep=""))

    #Varianzanalyse
    gruppe <- as.factor(rep(1:2, each=panz*10)) # fester Faktor A
    person <- as.factor(rep(1:panz,each=20)) # zufälliger Faktor B
    item <- as.factor(rep(1:20,times=panz)) #  fester Faktor C
    ergebnis<-c(t(a)) # abhängige Variable

    res<-unlist(summary(aov(ergebnis ~ gruppe + item + item:gruppe +
Error(person + person:item)))) # Varianzanalyse

    df <- res[c(1,2,11,12,13)]
    sum <- res[c(3,4,14,15,16)]
    mea <- res[c(5,6,17,18,19)]
    fb <- res[6]/res[19]
    pb <- pf(fb,res[2],res[13], lower.tail = FALSE)
    f <- res[c(7,8,20,21,22)]
    f[2]<-fb
    p <- res[c(9,10,23,24,25)]
    p[2]<-pb
```

```
    sig <- ifelse(p<0.05,1,0)
    erg <- cbind(df,sum,mea,f,p,sig)
    rownames(erg) <- c("A","B(A)","C","AC","CB(A)")

    write.table(erg, file =
paste("/home/futschek/RMdat/alpha/V/V",gsub("s","", g),sep=""))

    gc()
}


# +++++++++++++++++++++++++++ auswertenAlpha ++++++++++++++++++++

auswertenAlpha <- function(ppar,ipar,wh) {

v <- 1:wh # Vektor, der die Simulationsnummer angibt
panz<-length(ppar) # Personenanzahl

# LR-Test nach Andersen - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
#jeder Simulation

A <- matrix(nrow=wh,ncol=3)
A[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/alpha/A/A",v[i],sep="")) #
#Einlesen
    A[i,2] <- a[3,] # p-wert extrahieren
    A[i,3] <- a[2,] # df extrahieren
    }

A <- cbind(A,ifelse(A[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
#1 sind sign Ergebnisse

write.table(A,"/home/futschek/RMdat/alpha/Auswertung/A") # Speichern

# 2. Auswertung

Aerg <- table(A[,3],A[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
Anzahl der (nicht) sign Ergebnisse
Aerg <- addmargins(Aerg)      # Hinzufügen von Randhäufigkeiten
write.table(Aerg,"/home/futschek/RMdat/alpha/Auswertung/Aerg")
#Speichern


# Martin-Löf-Test - - - - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
#jeder Simulation

M <- matrix(nrow=wh,ncol=3)
M[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/alpha/M/M",v[i],sep="")) #
#Einlesen
    M[i,2] <- a[3,] # p-wert extrahieren
```

```
    M[i,3] <- a[2,] # df extrahieren
    }

M <- cbind(M,ifelse(M[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
#1 sind sign Ergebnisse

write.table(M,"/home/futschek/RMdat/alpha/Auswertung/M") # Speichern

# Auswertung

Merg <- table(M[,3],M[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
#Anzahl der (nicht) sign Ergebnisse
Merg <- addmargins(Merg)      # Hinzufügen von Randhäufigkeiten
write.table(Merg,"/home/futschek/RMdat/alpha/Auswertung/Merg")
#Speichern


# z-Test - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten oder NAs für jedes Item

Z <- matrix(nrow=20,ncol=wh+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/alpha/Z/Z",v[i],sep="")) #
#Einlesen
    if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "", itemNr))
else itemNr <- NA # Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte extrahieren
    if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
    if (length(a)==2) c <- merge(a,b,all=T) else c <- NA # erstelltes
#Datenfile mit Vektor, der alle Itemnummern enthält, verbinden, NAs
#entstehen an den richtigen Stellen
    if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
    }
write.table(Z,"/home/futschek/RMdat/alpha/Auswertung/Z") # Speichern

# 2. Auswertung

AnzGültigItems <- sum(apply(!is.na(Z[,2:wh+1]),1,sum)) # Anzahl an
#Items, für die eine Schätzung vorliegt
ProzGültigItems <- AnzGültigItems/(wh*20) # Prozentsatz an Items, für
#die eine Schätzung vorliegt

AnzSignItems <- sum(ifelse(Z[,2:wh+1]<0.05,1,0),na.rm=T)# Anzahl
#signifikanter Items
ProzSignItems <- AnzSignItems/AnzGültigItems # Prozentsatz
#signifikanter Items an den schätzbaren Items


Zerg <-
data.frame(panz,AnzGültigItems,ProzGültigItems,AnzSignItems,ProzSignIt
ems) # Ergebnisfile
write.table(Zerg,"/home/futschek/RMdat/alpha/Auswertung/Zerg")
```

```
# Varianzanalyse - - - - - - - - - - - - - - - - - - - - - - - -

V <- matrix(nrow=wh,ncol=5)
colnames(V) <- c("Nr","A","B(A)","C","AC")
V[,1] <- 1:wh # Vektor, der Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/alpha/V/V",v[i],sep=""))
    V[v[i],2] <- a[1,6]
    V[v[i],3] <- a[2,6]
    V[v[i],4] <- a[3,6]
    V[v[i],5] <- a[4,6]
    }
write.table(V,"/home/futschek/RMdat/alpha/Auswertung/V")
Verg <- apply(V,2,sum)
write.table(Verg,"/home/futschek/RMdat/alpha/Auswertung/Verg")

}
```

Befehle für die Simulation

```
# Folgende Funktionen müssen geladen sein: createAlpha, alphaRM,
#auswertenAlpha
# Daran denken, nach jedem Block die Daten zu sichern/umzubenennen,
# da diese sonst im Verzeichnis RMdat überschrieben werden

library(eRm)

# 100 Personen alphaRM

ppar100 <- rnorm(100, mean = 0, sd = 1.5)
ipar1 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -
0.1, 0.1, 0.3, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)

createAlpha(ppar100,ipar1,10000)
setwd("/home/futschek/RMdat/alpha/s")
daten <- list.files("/home/futschek/RMdat/alpha/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=alphaRM, ppar=ppar100,
ipar=ipar1)
stopCluster(kern)
auswertenAlpha(ppar100,ipar1,10000)

# 200 Personen alphaRM

ppar200 <- rnorm(200, mean = 0, sd = 1.5)
ipar1 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -
0.1, 0.1, 0.3, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)

createAlpha(ppar200,ipar1,10000)
setwd("/home/futschek/RMdat/alpha/s")
daten <- list.files("/home/futschek/RMdat/alpha/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=alphaRM, ppar=ppar200,
ipar=ipar1)
stopCluster(kern)
auswertenAlpha(ppar200,ipar1,10000)
```

```
# 300 Personen alphaRM

ppar300 <- rnorm(300, mean = 0, sd = 1.5)
ipar1 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -
0.1, 0.1, 0.3, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)

createAlpha(ppar300,ipar1,10000)
setwd("/home/futschek/RMdat/alpha/s")
daten <- list.files("/home/futschek/RMdat/alpha/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=alphaRM, ppar=ppar300,
ipar=ipar1)
stopCluster(kern)
auswertenAlpha(ppar300,ipar1,10000)
```

Nachauswertung

Mit folgendem Befehl kann geprüft werden, ob die Wechselwirkung AC der Varianzanalyse signifikant ist für alle Fälle bei denen A signifikant ist. Das Verzeichnis RMdat wurde hier in DatenALPHA100 umbenannt. Für zukünftige Simulationen wäre es gut, diese Nachauswertung in die reguläre Auswertung zu integrieren.

```
a <- read.table("/home/futschek/DatenALPHA100/alpha/Auswertung/V")
a[which(a[,2] == 1),]
```

R-Code für die Simulation zur Teststärke des LR-Tests, des z-Tests nach Fischer und Scheiblechner und der dreifachen Varianzanalse

Funktionen: createBeta, betaRM, auswertenBetaNeu, auswertenBetaNeu3, auswertenBetaNeu4

```
#+++++++++++++++++++++++++++++++ createBeta ++++++++++++++++++++++++

createBeta <- function(ppar,ipar1,ipar2,wh) {
# ppar: Vektor mit Personenparametern
# ipar1: Vektor mit Itemparametern der einen Hälfte Items, Achtung:
Muss in diesem Programm Länge 10 haben
# ipar2: Vektor mit Itemparametern der zweiten Hälfte Items, Achtung:
Muss in diesem Programm Länge 10 haben
# wh: Anzahl der Simulationen

v <- 1:wh # Vektor, der die Simulationsnummer angibt
panz<-length(ppar) # Personenanzahl

# Verzeichnisse erstellen, in die gespeichert wird
dir.create("/home/futschek/RMdat")
dir.create("/home/futschek/RMdat/beta")
dir.create("/home/futschek/RMdat/beta/ipar")
dir.create("/home/futschek/RMdat/beta/p")
dir.create("/home/futschek/RMdat/beta/Auswertung")
```

```
dir.create("/home/futschek/RMdat/beta/s")
dir.create("/home/futschek/RMdat/beta/A")
dir.create("/home/futschek/RMdat/beta/V")
dir.create("/home/futschek/RMdat/beta/Z")

#Dokumentation der vorgegebenen Parameter
write.table(ipar1, file = "/home/futschek/RMdat/beta/ipar/ipar1")
write.table(ipar2, file = "/home/futschek/RMdat/beta/ipar/ipar2")
write.table(ppar, file =
paste("/home/futschek/RMdat/beta/p/p",panz,sep="")) # Speichern


# Simulation von "wh" nicht raschkonformen Datensätzen und Speicherung

for (i in 1:wh) {
    ppar1<-ppar[1:(length(ppar)/2)]
    ppar2<-ppar[((length(ppar)/2)+1):length(ppar)]
    simrm1 <- sim.rasch(ppar1,ipar1)
    simrm2 <- sim.rasch(ppar2,ipar2)
    simrm <- rbind(simrm1, simrm2)
    write.table(simrm, file =
paste("/home/futschek/RMdat/beta/s/s",v[i],sep=""))
    }
}

# ++++++++++++++++++++++++++++ betaRM ++++++++++++++++++++++++++++++++

betaRM <- function(g,ppar,wh) {
     # die Funktion führt die Modellprüfungen durch

    require(eRm)

    panz<-length(ppar) # Personenanzahl

    # Aufrufen eines Datensatzes
    # g <-
#read.table(paste("/home/futschek/RMdat/beta/s/s",v[i],sep=""))

    a <- read.table(g)

    # RM-Berechnung

    b <- try(RM(a))

    #Modelltests

    if (is(b,"dRm"))  {
    c <- try(LRtest(b, splitcr = c(rep(1:2,each=(panz/2)))))   # LR-
#Test
    e <- try(Waldtest(b)) } else  {                            #
#Waldtest
    c <- NA
    e <- NA
    }

    if (is(c,"LR"))  c1 <- c(c$LR,c$df,c$p) else c1<-c(NA,NA,NA)
    if (is(e,"wald")) e1 <- e$coef.table else e1<- NA

    write.table(c1, file =
paste("/home/futschek/RMdat/beta/A/A",gsub("s","", g),sep=""))
```

```
    write.table(e1, file =
paste("/home/futschek/RMdat/beta/Z/Z",gsub("s","", g),sep=""))

    #Varianzanalyse
    gruppe <- as.factor(rep(1:2, each=panz*10)) # fester Faktor A
    person <- as.factor(rep(1:panz,each=20)) # zufälliger Faktor B
    item <- as.factor(rep(1:20,times=panz)) #  fester Faktor C
    ergebnis<-c(t(a)) # abhängige Variable

    res<-unlist(summary(aov(ergebnis ~ gruppe + item + item:gruppe +
Error(person + person:item)))) # Varianzanalyse

    df <- res[c(1,2,11,12,13)]
    sum <- res[c(3,4,14,15,16)]
    mea <- res[c(5,6,17,18,19)]
    fb <- res[6]/res[19]
    pb <- pf(fb,res[2],res[13], lower.tail = FALSE)
    f <- res[c(7,8,20,21,22)]
    f[2]<-fb
    p <- res[c(9,10,23,24,25)]
    p[2]<-pb
    sig <- ifelse(p<0.05,1,0)
    erg <- cbind(df,sum,mea,f,p,sig)
    rownames(erg) <- c("A","B(A)","C","AC","CB(A)")

    write.table(erg, file =
paste("/home/futschek/RMdat/beta/V/V",gsub("s","", g),sep=""))

    gc()
}

# +++++++++++++++++++++++++++++ auswertenBetaNeu ++++++++++++++++++++
# wenn Item 4 und 17 betroffen

auswertenBetaNeu <- function(ppar,wh) {

v <- 1:wh # Vektor, der die Simulationsnummer angibt
panz<-length(ppar) # Personenanzahl

# LR-Test nach Andersen 1- - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
#jeder Simulation

A <- matrix(nrow=wh,ncol=3)
A[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/A/A",v[i],sep="")) #
#Einlesen
    A[i,2] <- a[3,] # p-wert extrahieren
    A[i,3] <- a[2,] # df extrahieren
    }

A <- cbind(A,ifelse(A[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
#1 sind sign Ergebnisse

write.table(A,"/home/futschek/RMdat/beta/Auswertung/A") # Speichern
```

```
# 2. Auswertung

Aerg <- table(A[,3],A[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
#Anzahl der (nicht) sign Ergebnisse
Aerg <- addmargins(Aerg)      # Hinzufügen von Randhäufigkeiten
write.table(Aerg,"/home/futschek/RMdat/beta/Auswertung/Aerg")
#Speichern


# z-Test - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten oder NAs für jedes Item

Z <- matrix(nrow=20,ncol=wh+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
ZA <- matrix(nrow=wh,ncol=5)
ZA[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/Z/Z",v[i],sep="")) #
#Einlesen

    if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "", itemNr))
else itemNr <- NA # Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte extrahieren
    if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
    if (length(a)==2) c <- merge(a,b,all=T) else c <- NA # erstelltes
#Datenfile mit Vektor, der alle Itemnummern enthält, verbinden, NAs
#entstehen an den richtigen Stellen
    if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen

    if (length(a)==2) ZA[i,2] <- c[4,2] else ZA[i,2] <- NA    #
#Achtung hier nur richtig falls Item 4 und 17 betroffen sind!
    if (length(a)==2) ZA[i,3] <- c[17,2] else ZA[i,3] <- NA
    ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
    ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
    }
write.table(Z,"/home/futschek/RMdat/beta/Auswertung/Z") # Speichern
write.table(ZA,"/home/futschek/RMdat/beta/Auswertung/ZA")

# 2. Auswertung

AnzGültigItems <- sum(apply(!is.na(Z[,2:wh+1]),1,sum)) # Anzahl an
#Items, für die eine Schätzung vorliegt
ProzGültigItems <- AnzGültigItems/(wh*20) # Prozentsatz an Items, für
#die eine Schätzung vorliegt

AnzSignItems <- sum(ifelse(Z[,2:wh+1]<0.05,1,0),na.rm=T)# Anzahl
#signifikanter Items
ProzSignItems <- AnzSignItems/AnzGültigItems # Prozentsatz
#signifikanter Items an den schätzbaren Items

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))) # Prozentsatz der
#erkannten von den kritischen Items
```

```
Zerg <-
data.frame(panz,AnzGültigItems,ProzGültigItems,AnzSignItems,ProzSignIt
ems,teststärkeZ) # Ergebnisfile
write.table(Zerg,"/home/futschek/RMdat/beta/Auswertung/Zerg")



# Varianzanalyse - - - - - - - - - - - - - - - - - - - - - - - - - - - -

V <- matrix(nrow=wh,ncol=5)
colnames(V) <- c("Nr","A","B(A)","C","AC")
V[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/V/V",v[i],sep=""))
    V[v[i],2] <- a[1,6]
    V[v[i],3] <- a[2,6]
    V[v[i],4] <- a[3,6]
    V[v[i],5] <- a[4,6]
    }
write.table(V,"/home/futschek/RMdat/beta/Auswertung/V")
Verg <- apply(V,2,sum)
write.table(Verg,"/home/futschek/RMdat/beta/Auswertung/Verg")

}



# ++++++++++++++++++++++++++++ auswertenBetaNeu3 ++++++++++++++++++++
# wenn 9 und 12 betroffen

auswertenBetaNeu3 <- function(ppar,wh) {

v <- 1:wh # Vektor, der die Simulationsnummer angibt
panz<-length(ppar) # Personenanzahl

# LR-Test nach Andersen 1- - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
#jeder Simulation

A <- matrix(nrow=wh,ncol=3)
A[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/A/A",v[i],sep="")) #
#Einlesen
    A[i,2] <- a[3,] # p-wert extrahieren
    A[i,3] <- a[2,] # df extrahieren
    }

A <- cbind(A,ifelse(A[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
#1 sind sign Ergebnisse

write.table(A,"/home/futschek/RMdat/beta/Auswertung/A") # Speichern

# 2. Auswertung

Aerg <- table(A[,3],A[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
Anzahl der (nicht) sign Ergebnisse
```

```
Aerg <- addmargins(Aerg)      # Hinzufügen von Randhäufigkeiten
write.table(Aerg,"/home/futschek/RMdat/beta/Auswertung/Aerg")
#Speichern


# z-Test - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten oder NAs für jedes Item

Z <- matrix(nrow=20,ncol=wh+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
ZA <- matrix(nrow=wh,ncol=5)
ZA[,1] <- 1:wh # Vektor, der Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/Z/Z",v[i],sep="")) #
#Einlesen

    if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "", itemNr))
else itemNr <- NA # Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte extrahieren
    if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
    if (length(a)==2) c <- merge(a,b,all=T) else c <- NA # erstelltes
#Datenfile mit Vektor, der alle Itemnummern enthält, verbinden, NAs
#entstehen an den richtigen Stellen
    if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen

    if (length(a)==2) ZA[i,2] <- c[9,2] else ZA[i,2] <- NA    #
#Achtung, hier nur richtig, falls Item 9 und 12 betroffen sind!
    if (length(a)==2) ZA[i,3] <- c[12,2] else ZA[i,3] <- NA
    ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
    ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
    }
write.table(Z,"/home/futschek/RMdat/beta/Auswertung/Z") # Speichern
write.table(ZA,"/home/futschek/RMdat/beta/Auswertung/ZA")

# 2. Auswertung

AnzGültigItems <- sum(apply(!is.na(Z[,2:wh+1]),1,sum)) # Anzahl an
#Items, für die eine Schätzung vorliegt
ProzGültigItems <- AnzGültigItems/(wh*20) # Prozentsatz an Items, für
#die eine Schätzung vorliegt

AnzSignItems <- sum(ifelse(Z[,2:wh+1]<0.05,1,0),na.rm=T)# Anzahl
#signifikanter Items
ProzSignItems <- AnzSignItems/AnzGültigItems # Prozentsatz
#signifikanter Items an den schätzbaren Items

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))) # Prozentsatz der
#erkannten von den kritischen Items

Zerg <-
data.frame(panz,AnzGültigItems,ProzGültigItems,AnzSignItems,ProzSignIt
ems,teststärkeZ) # Ergebnisfile
write.table(Zerg,"/home/futschek/RMdat/beta/Auswertung/Zerg")
```

```
# Varianzanalyse - - - - - - - - - - - - - - - - - - - - - - - - - - - -

V <- matrix(nrow=wh,ncol=5)
colnames(V) <- c("Nr","A","B(A)","C","AC")
V[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/V/V",v[i],sep=""))
    V[v[i],2] <- a[1,6]
    V[v[i],3] <- a[2,6]
    V[v[i],4] <- a[3,6]
    V[v[i],5] <- a[4,6]
    }
write.table(V,"/home/futschek/RMdat/beta/Auswertung/V")
Verg <- apply(V,2,sum)
write.table(Verg,"/home/futschek/RMdat/beta/Auswertung/Verg")

}



# ++++++++++++++++++++++++++++ auswertenBetaNeu4 ++++++++++++++++++++
# wenn 14 und 17 betroffen

auswertenBetaNeu4 <- function(ppar,wh) {

v <- 1:wh # Vektor, der die Simulationsnummer angibt
panz<-length(ppar) # Personenanzahl

# LR-Test nach Andersen 1- - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
#jeder Simulation

A <- matrix(nrow=wh,ncol=3)
A[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/A/A",v[i],sep="")) #
#Einlesen
    A[i,2] <- a[3,] # p-wert extrahieren
    A[i,3] <- a[2,] # df extrahieren
    }

A <- cbind(A,ifelse(A[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
#1 sind sign Ergebnisse

write.table(A,"/home/futschek/RMdat/beta/Auswertung/A") # Speichern

# 2. Auswertung

Aerg <- table(A[,3],A[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
#Anzahl der (nicht) sign Ergebnisse
Aerg <- addmargins(Aerg)      # Hinzufügen von Randhäufigkeiten
write.table(Aerg,"/home/futschek/RMdat/beta/Auswertung/Aerg")
#Speichern
```

```
# z-Test - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten oder NAs für jedes Item

Z <- matrix(nrow=20,ncol=wh+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
ZA <- matrix(nrow=wh,ncol=5)
ZA[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/Z/Z",v[i],sep="")) #
#Einlesen

    if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "", itemNr))
else itemNr <- NA # Itemnummern der simulierten Daten extrahieren
    if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte extrahieren
    if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
Datenfile erstellen Itemnummern und p-Werte
    if (length(a)==2) c <- merge(a,b,all=T) else c <- NA # erstelltes
#Datenfile mit Vektor, der alle Itemnummern enthält, verbinden, NAs
#entstehen an den richtigen Stellen
    if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen

    if (length(a)==2) ZA[i,2] <- c[14,2] else ZA[i,2] <- NA    #
#Achtung, hier nur richtig, falls Item 14 und 17 betroffen sind!
    if (length(a)==2) ZA[i,3] <- c[17,2] else ZA[i,3] <- NA
    ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
    ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
    }
write.table(Z,"/home/futschek/RMdat/beta/Auswertung/Z") # Speichern
write.table(ZA,"/home/futschek/RMdat/beta/Auswertung/ZA")

# 2. Auswertung

AnzGültigItems <- sum(apply(!is.na(Z[,2:wh+1]),1,sum)) # Anzahl an
#Items, für die eine Schätzung vorliegt
ProzGültigItems <- AnzGültigItems/(wh*20) # Prozentsatz an Items, für
#die eine Schätzung vorliegt

AnzSignItems <- sum(ifelse(Z[,2:wh+1]<0.05,1,0),na.rm=T)# Anzahl
#signifikanter Items
ProzSignItems <- AnzSignItems/AnzGültigItems # Prozentsatz
#signifikanter Items an den schätzbaren Items

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))) # Prozentsatz der
#erkannten von den kritischen Items

Zerg <-
data.frame(panz,AnzGültigItems,ProzGültigItems,AnzSignItems,ProzSignIt
ems,teststärkeZ) # Ergebnisfile
write.table(Zerg,"/home/futschek/RMdat/beta/Auswertung/Zerg")
```

```
# Varianzanalyse - - - - - - - - - - - - - - - - - - - - - - - - - - - -

V <- matrix(nrow=wh,ncol=5)
colnames(V) <- c("Nr","A","B(A)","C","AC")
V[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt
for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/V/V",v[i],sep=""))
    V[v[i],2] <- a[1,6]
    V[v[i],3] <- a[2,6]
    V[v[i],4] <- a[3,6]
    V[v[i],5] <- a[4,6]
    }
write.table(V,"/home/futschek/RMdat/beta/Auswertung/V")
Verg <- apply(V,2,sum)
write.table(Verg,"/home/futschek/RMdat/beta/Auswertung/Verg")

}
```

Befehle für die Simulation

```
# zwei sd
# 100 Personen beta

ipar1 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -
0.1, 0.1, 0.3, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)
ipar2 <- c(-3.0, -2.5, -2.0, 1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -0.1,
0.1, 0.3, 0.5, 0.75, 0.9, 1.2, -1.5, 2.0, 2.5, 3.0)

createBeta(ppar100,ipar1,ipar2,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar100)
stopCluster(kern)
auswertenBetaNeu(ppar100,wh=10000)


# zwei sd
# 200 Personen beta

ipar1 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -
0.1, 0.1, 0.3, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)
ipar2 <- c(-3.0, -2.5, -2.0, 1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -0.1,
0.1, 0.3, 0.5, 0.75, 0.9, 1.2, -1.5, 2.0, 2.5, 3.0)

createBeta(ppar200,ipar1,ipar2,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar200)
stopCluster(kern)
auswertenBetaNeu(ppar200,wh=10000)


# zwei sd
# 300 Personen beta

createBeta(ppar300,ipar1,ipar2,wh=10000)
```

```
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar300)
stopCluster(kern)
auswertenBetaNeu(ppar300,wh=10000)


# eine sd
# 100 Personen beta

ipar3 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.3, -
0.1, 0.1, 0.3, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)
ipar4 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, 0.75, -0.5, -0.3, -0.1,
0.1, 0.3, 0.5, -0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)

createBeta(ppar100,ipar3,ipar4,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar100)
stopCluster(kern)
auswertenBetaNeu3(ppar100,wh=10000)


# eine sd
# 200 Personen beta

createBeta(ppar200,ipar3,ipar4,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar200)
stopCluster(kern)
auswertenBetaNeu3(ppar200,wh=10000)


# eine sd
# 300 Personen beta

createBeta(ppar300,ipar3,ipar4,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar300)
stopCluster(kern)
auswertenBetaNeu3(ppar300,wh=10000)


# eine halbe sd
# 100 Personen beta

ipar5 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.375, -
0.1, 0.1, 0.375, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)
ipar6 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, 0.375, -
0.1, 0.1, -0.375, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)

createBeta(ppar100,ipar5,ipar6,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
```

```
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar100)
stopCluster(kern)
auswertenBetaNeu4(ppar100,wh=10000)


# eine halbe sd
# 200 Personen beta

ipar5 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, -0.375, -
0.1, 0.1, 0.375, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)
ipar6 <- c(-3.0, -2.5, -2.0, -1.5, -1.2, -0.9, -0.75, -0.5, 0.375, -
0.1, 0.1, -0.375, 0.5, 0.75, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0)

createBeta(ppar200,ipar5,ipar6,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar200)
stopCluster(kern)
auswertenBetaNeu4(ppar200,wh=10000)


# eine halbe sd
# 300 Personen beta

createBeta(ppar300,ipar5,ipar6,wh=10000)
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaRM, ppar=ppar300)
stopCluster(kern)
auswertenBetaNeu4(ppar300,wh=10000)
```

Nachauswertung II

Wiederum wurde im Nachhinein geprüft, ob die Wechselwirkung AC der Varianzanalyse signifikant ist für alle Fälle bei denen A signifikant ist. Für zukünftige Simulationen wäre es gut, diese Nachauswertung in die reguläre Auswertung zu integrieren. Mit folgendem Befehl und den jeweils richtigen Pfadangaben kann dies durchgeführt werden. RMdat wurde hier in DatenBETA100 umbenannt.

```
a <- read.table("/home/futschek/DatenBETA100/beta/Auswertung/V")
a[which(a[,2] == 1),]
```

Außerdem wurde für den z-Test nach Fischer und Scheiblechner nachbestimmt, für wie viele der modellverletzenden Items der z-Test nach Fischer und Scheiblechner tatsächlich durchgeführt werden konnte und wie viele davon signifikant waren. Auch hier würde sich für zukünftige Simulationen anbieten, diese Nachauswertung in die reguläre Auswertung einzubauen.

```
# Nachbestimmen Z bei beta

# 100 beta 2sd

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/DatenBETA100/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[4,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[17,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ
sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))


# 200 beta 2 sd

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/DatenBeta200/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
```

```
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[4,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[17,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ
sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))


# 300 beta 2sd

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/DatenBeta300/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[4,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[17,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ
sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))
```

```
# beta 1.5  p100

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/Daten1.5BETA100/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[7,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[14,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}


teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ

sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))



 # beta 1.5  p200

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/Daten1.5Beta200/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
```

```
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[7,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[14,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}


teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ

sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))


 # beta 1.5  p300

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/Daten1.5Beta300/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[7,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[14,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}
```

```
teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ

sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))


# beta 0.75 p100

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/Daten0.75BETA100/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[9,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[12,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}


teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ

sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))
sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)

# beta 0.75 p200

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000
```

```
for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/Daten0.75Beta200/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[9,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[12,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}


teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ

sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))
sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)


# beta 0.75 p300

ZA <- matrix(nrow=10000,ncol=5)
ZA[,1] <- 1:10000 # Vektor, der die Nummer der Simulation angibt
Z <- matrix(nrow=20,ncol=10000+1) # Gesamtmatrix
Z[,1] <- 1:20 # Vektor mit Itemnummer
b <- data.frame(itemNr = c(1:20))
v<-1:10000

for (i in 1:10000) {

    a <- read.table(file =
paste("/home/futschek/Daten0.75Beta300/beta/Z/Z",v[i],sep=""))

        if (length(a)==2) itemNr <- labels(a)[[1]] else itemNr <- NA #
#Itemnummern der simulierten Daten extrahieren
        if (length(a)==2) itemNr <- as.numeric(gsub("beta V", "",
itemNr)) else itemNr <- NA # Itemnummern der simulierten Daten
#extrahieren
        if (length(a)==2) p <- c(a$p) else p <- NA # p-Werte
#extrahieren
        if (length(a)==2) a <- data.frame(itemNr,p) else a <- NA #
#Datenfile erstellen Itemnummern und p-Werte
        if (length(a)==2) c <- merge(a,b,all=T) else c <- NA #
#erstelltes Datenfile mit Vektor, der alle Itemnummern enthält,
#verbinden, NAs entstehen an den richtigen Stellen
```

```
        if (length(a)==2) Z[,i+1] <- c$p else c <- NA # Vektor mit p-
#Werten und NAs in die Gesamtmatrix einfügen
        if (length(a)==2) ZA[i,2] <- c[9,2] else ZA[i,2] <- NA
        if (length(a)==2) ZA[i,3] <- c[12,2] else ZA[i,3] <- NA
        ZA[i,4] <- ifelse(ZA[i,2]<0.05,1,0)
        ZA[i,5] <- ifelse(ZA[i,3]<0.05,1,0)
}

teststärkeZ<-(sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)) /
(sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5])))
teststärkeZ
sum(!is.na(ZA[,4])) + sum(!is.na(ZA[,5]))
sum(ZA[,4], na.rm=T) + sum(ZA[,5], na.rm=T)
```

R-Code für die Simulation von Multidimensionalität

Funktionen: createMF, betaMF, auswertenMF

```
#++++++++++++++++++++++++ createMF ++++++++++++++++++++++++++++++++

createMF <- function(kor,panz,ipar,wh) {
# kor: Korrelationskoeffizient zwischen den Personengruppen
# panz: Personenanzahl
# ipar: Vektor mit Itemparametern, Achtung: Muss in diesem Programm
Länge 20 haben
# wh: Anzahl der Simulationen

v <- 1:wh # Vektor, der die Simulationsnummer angibt
Sigma <- matrix(c(1, kor, kor, 1), 2)
weights <- matrix(c(rep(1,10), rep(0,10),rep(0,10),rep(1,10)),ncol =
2)

# Verzeichnisse erstellen, in die gespeichert wird
dir.create("/home/futschek/RMdat")
dir.create("/home/futschek/RMdat/beta")
dir.create("/home/futschek/RMdat/beta/ipar")
dir.create("/home/futschek/RMdat/beta/p")
dir.create("/home/futschek/RMdat/beta/Auswertung")
dir.create("/home/futschek/RMdat/beta/s")
dir.create("/home/futschek/RMdat/beta/M")
dir.create("/home/futschek/RMdat/beta/A")

#Dokumentation der vorgegebenen Parameter
write.table(ipar, file = "/home/futschek/RMdat/beta/ipar/ipar")
write.table(kor, file =
paste("/home/futschek/RMdat/beta/p/kor",panz,sep="")) # Speichern
latente Korrelation

# Simulation von "wh" nicht raschkonformen Datensätzen und Speicherung

for (i in 1:wh) {

    simrm <- sim.xdim(panz, ipar, Sigma, weightmat = weights) # Sim
Daten
    row.names(simrm)<-c(1:panz)
```

```
    write.table(simrm, file =
paste("/home/futschek/RMdat/beta/s/s",v[i],sep=""))
    }
}



# ++++++++++++++++++++++ betaMF ++++++++++++++++++++++++++++++++++

betaMF <- function(g) {
     # die Funktion führt die Modellprüfungen durch

    require(eRm)

    # Aufrufen eines Datensatzes

    a <- read.table(g)

    # RM-Berechnung

    b <- try(RM(a))

    #Modelltests

    if (is(b,"dRm"))  {
    c <- try(LRtest(b, splitcr = "median"))   # LR-Test
    d <- try(MLoef(b, splitcr = c(rep(1:2,each=10))))# Martin-Löf-Test
                       } else {
    d <- NA
    c <- NA
    }

    if (is(d, "MLoef")) d1 <- c(d$LR,d$df,d$p) else  d1 <- c(NA,NA,NA)
    if (is(c,"LR"))  c1 <- c(c$LR,c$df,c$p) else c1<-c(NA,NA,NA)

    write.table(c1, file =
paste("/home/futschek/RMdat/beta/A/A",gsub("s","", g),sep=""))
    write.table(d1, file =
paste("/home/futschek/RMdat/beta/M/M",gsub("s","", g),sep=""))

    gc()
}



# ++++++++++++++++++++++++++++ auswertenMF++++++++++++++++++++++++++


auswertenMF <- function(wh) {

# Martin-Löf-Test - - - - - - - - - - - - - - - - - - - - - - - - - -
v <- 1:wh # Vektor, der die Simulationsnummer angibt

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
jeder Simulation

M <- matrix(nrow=wh,ncol=3)
M[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
```

```
    a <- read.table(paste("/home/futschek/RMdat/beta/M/M",v[i],sep="")
    # Einlesen
    M[i,2] <- a[3,] # p-wert extrahieren
    M[i,3] <- a[2,] # df extrahieren
    }

M <- cbind(M,ifelse(M[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
1 sind sign Ergebnisse

write.table(M,"/home/futschek/RMdat/beta/Auswertung/M") # Speichern

# 2. Auswertung

Merg <- table(M[,3],M[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
# Anzahl der (nicht) sign Ergebnisse
Merg <- addmargins(Merg)      # Hinzufügen von Randhäufigkeiten
write.table(Merg,"/home/futschek/RMdat/beta/Auswertung/Merg")
#Speichern

# LR-Test nach Andersen 1- - - - - - - - - - - - - - - - - - - - - -

# 1. Kreieren einer Matrix mit allen p-Werten und Freiheitsgraden
jeder Simulation

A <- matrix(nrow=wh,ncol=3)
A[,1] <- 1:wh # Vektor, der die Nummer der Simulation angibt

for (i in 1:wh) {
    a <-
read.table(paste("/home/futschek/RMdat/beta/A/A",v[i],sep="")) #
Einlesen
    A[i,2] <- a[3,] # p-wert extrahieren
    A[i,3] <- a[2,] # df extrahieren
    }

A <- cbind(A,ifelse(A[,2]>0.05,0,1)) # fügt eine Zeile mit 0/1 hinzu,
#1 sind sign Ergebnisse

write.table(A,"/home/futschek/RMdat/beta/Auswertung/A") # Speichern

# 2. Auswertung

Aerg <- table(A[,3],A[,4]) # Kreuztabelle: Anzahl der Freiheitsgrade,
#Anzahl der (nicht) sign Ergebnisse
Aerg <- addmargins(Aerg)      # Hinzufügen von Randhäufigkeiten
write.table(Aerg,"/home/futschek/RMdat/beta/Auswertung/Aerg")
#Speichern


}
```

Befehle für die Simulation

```
#für 1000 wh bei 100 Personen

iparMF1 <- c(-3.0, -2.0, -1, -0.4, -0.1, 0.1, 0.4, 1, 2, 3)

iparMF <- c(iparMF1, iparMF1) # 20 Items

createMF(0.5,100,iparMF,1000)
```

```
setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaMF)
stopCluster(kern)

auswertenMF(1000)


#für 1000 wh bei 200 Personen

iparMF1 <- c(-3.0, -2.0, -1, -0.4, -0.1, 0.1, 0.4, 1, 2, 3)

iparMF <- c(iparMF1, iparMF1) # 20 Items

createMF(0.5,200,iparMF,1000)

setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaMF)
stopCluster(kern)

auswertenMF(1000)


#für 1000 wh bei 300 Personen

iparMF1 <- c(-3.0, -2.0, -1, -0.4, -0.1, 0.1, 0.4, 1, 2, 3)

iparMF <- c(iparMF1, iparMF1) # 20 Items

createMF(0.5,300,iparMF,1000)

setwd("/home/futschek/RMdat/beta/s")
daten <- list.files("/home/futschek/RMdat/beta/s")
kern <- makeSOCKcluster(rep("localhost", 4))
nothing <- parLapply(kern, daten, fun=betaMF)
stopCluster(kern)

auswertenMF(1000)
```

14. Lebenslauf

Karin Futschek
geboren 1989 in Wien
E-Mail: karin.futschek@gmx.at

(Akademische) Ausbildung:

Seit Okt. 2009	Zweiter Studienabschnitt Diplomstudium Psychologie
Okt. 2009 – voraussichtlich März. 2014	Bakkalaureatsstudium Statistik (Abschluss: Bakk. rer. soc. oec.)
Okt. 2007 – Aug. 2009	Erster Studienabschnitt Diplomstudium Psychologie
Jun. 2007	Matura mit ausgezeichnetem Erfolg

Arbeitserfahrung:

Seit Okt. 2013	Studienassistentin am Institut für Psychologische Grundlagenforschung und Forschungsmethoden der Universität Wien
März 2011 – Juni 2011	Pflichtpraktikum (240h) im Sozialmedizinischem Zentrum Ost – Donauspital