TECHNISCHE UNIVERSITÄT WIEN

Diplomarbeit

Die Kleinstquadratmethode zur Faktorextraktion in der Faktorenanalyse

---

Ausgeführt am Institut für

Stochastik und Wirtschaftsmathematik

---

der Technischen Universität Wien

unter der Anleitung von Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Karl Grill

durch

Friedrich Winkelbauer, BSc

Wien, am

Friedrich Winkelbauer, BSc

Zusammenfassung

Die hier vorliegende Arbeit befasst sich mit der Faktorenanalyse. Es werden die mathematischen Grundlagen sowie das statistische Modell dazu erklärt.
Der Hauptteil der Arbeit besteht im Vergleich einer neuen Methode zur Kleinstquadrat-Faktorextraktion, einer Modifikation der iterierten Hauptachsenmethode zur Vermeidung von Heywoodfällen, mit bekannten Methoden. Dafür werden die jeweiligen Algorithmen implementiert und sowohl Laufzeit- als auch Genauigkeitstests durchgeführt, wobei auch die Resistenz gegen Heywoodfälle beobachtet wird.
Außerdem wird das Verhalten der iterierten Hauptachsentransformation auf theoretischer Ebene im einfachsten Modell mit drei Variablen und einem Faktor untersucht.

Abstract

This thesis treats the subject of factor analysis. It introduces the mathematical basis and the statistical model.
The principal part consists in a comparison between a new method for least squares factor extraction, a modification of the iterated principal axis transformation method, and other known methods. The algorithms are implemented and tested for runtime and accuracy with the added viewpoint of resistance to Heywood cases.
Additionally, the behaviour of the iterated principal axis transformation method is examined on a theoretical level for the easiest model with three variables and one factor.

Inhaltsverzeichnis

1 Einleitung

Das Ziel dieser Arbeit ist es, eine neue Methode zur Kleinstquadrat-Faktorextraktion, einer Modifikation der iterierten Hauptachsentransformation zur Vermeidung von Heywoodfällen vorzustellen und mit anderen Methoden zu vergleichen. Außerdem soll das Verhalten der iterierten Hauptachsentransformation theoretisch für den einfachsten Fall mit drei Variablen und einem Faktor betrachtet werden.

In Kapitel 2 erfolgt eine kurze Einführung zur Faktorenanalyse. Danach werden die mathematischen Grundlagen dargestellt und die Geschichte der Faktorenanalyse erklärt.

Das 3. Kapitel stellt das faktoranalytische Modell vor. Dabei wird auch das Rotationsproblem angesprochen.

Kapitel 4 widmet sich der Vorstellung der Methoden der Faktorenanalyse. Vorgestellt werden die Maximum-Likelihood Methode, die Methoden, die aus der Hauptkomponentenanalyse resultieren sowie die Minres Methode.

Im 5. Kapitel werden Heywoodfälle eingeführt und die Konvergenz der iterierten Hauptachsentransformation in einem ausgewählten Fall betrachtet.

Danach werden in Kapitel 6 die Algorithmen vorgestellt und anschließend im Hinblick auf Konvergenz, ihre Laufzeiten als auch auf die Genauigkeit bei Heywoodfällen getestet.

Die Tests werden in R durchgeführt, die Programmcodes für Tests und Analysen sind im Anhang zu finden.

2 Einführung in die Faktorenanalyse

2.1 Einführung in die Faktorenanalyse

Bevor wir uns mit dem mathematischen Hintergrund, dem Modell und den Methoden der Faktorenanalyse beschäftigen, wollen wir kurz darstellen, was die Faktorenanalyse ist und welche Funktion sie hat.
Die Idee hinter der Faktorenanalyse ist, dass sich Größen (im Folgenden Faktoren genannt) durch messbare miteinander korrelierte Variable ausdrücken. Der Faktor tritt dabei nicht selbst auf und ist nicht direkt messbar. Er wirkt nur durch die Korrelation in den anderen Variablen. Es lassen sich die Zusammenhänge in den Daten also durch den Faktor erklären, sofern man ihn kennt. Die Faktorenanalyse setzt hier an und will die den Daten zu Grunde liegenden Faktoren extrahieren. Überla schreibt hierzu:
„Das Hauptziel der Faktorenanalyse ist die Ableitung hypothetischer Größen oder Faktoren aus einer Menge beobachteter Variablen. Die Faktoren sollen möglichst einfach sein und die Beobachtungen hinreichend genau beschreiben und erklären.“ ([3] S.3)

2.2 Mathematische Grundlagen

In diesem Kapitel werden die mathematischen Grundlagen dargestellt, im Speziellen die algebraischen Matrixoperationen, die für die Formalisierung der Faktorenanalyse eine große Rolle spielen. Vergleiche hierfür [1], [2], [7] und [8].

Da in der Faktorenanalyse oftmals Systeme linearer Gleichungen gelöst werden müssen, wird zu Beginn nochmals die Berechnung von Determinanten gezeigt. Wir bezeichnen mit

$$\det A \text{ oder } |A| \text{ beziehungsweise } \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}$$

die Determinante einer Matrix.

Für 2×2-Matrizen und 3×3-Matrizen lassen sich noch kompakte und relativ einfache Formeln angeben:

Lemma 2.2.1 *Sei* $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ *, dann lässt sich die Determinante* $\det A = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$ *berechnen als*

$$a_{11}a_{22} - a_{21}a_{12}.$$

Es erfolgt also die Multiplikation der Hauptdiagonalelemente und anschließend die Subtraktion der multiplizierten Nebendiagonalelemente.

Lemma 2.2.2 *(Regel von Sarrus) Für* 3×3*-Matrizen gilt:*

$$\det A = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33}+a_{12}a_{23}a_{31}+a_{13}a_{32}a_{21}-a_{13}a_{22}a_{31}-a_{23}a_{32}a_{11}-a_{33}a_{21}a_{12}$$

Natürlich könnte man auch für höherdimensionale Matrizen noch Regeln angeben, allerdings sind diese bereits zu kompliziert, um noch von praktischem Nutzen zu sein. Aus diesem Grund wird für die Bestimmung der Determinanten größerer Matrizen auf den Entwicklungssatz von Laplace zurückgegriffen. Aus diesem lassen sich auch die Regeln für 2×2 und 3×3 Matrizen herleiten.

Definition 2.2.3 *Ist* $A = (a_{ij})$ *eine* $n \times n$*-Matrix, so bezeichnen wir mit* A^-_{kl} *die* $(n-1) \times (n-1)$*-Matrix, die entsteht, wenn aus der Matrix A die k-te Zeile und l-te Spalte weggestrichen werden. Hierbei muss darauf geachtet werden, die Indizes von* A^-_{kl} *entsprechend anzupassen, sodass keine Sprünge in der Indexmenge entstehen, das heißt* $i > k$ *werden zu* $i-1$ *und* $j > l$ *zu* $j-1$*. Der Kofaktor* $a^\sharp_{lk}$ *berechnet sich dann als* $a^\sharp_{lk} = (-1)^{k+l}|A^-_{kl}|$

Lemma 2.2.4 *(Entwicklungssatz von Laplace) Ist* $A = (a_{ij}) \in K^{n \times n}$ *und* $A^\sharp = (a^\sharp_{kl}) \in K^{n \times n}$ *die Matrix der Kofaktoren von A. Dann gilt für beliebiges* $i, j \in \{1, 2, \ldots, n\}$ *:*

$$\begin{aligned} \det A &= a_{i1}a^\sharp_{1i} + a_{i2}a^\sharp_{2i} + \cdots + a_{in}a^\sharp_{ni} \\ \det A &= a_{1j}a^\sharp_{j1} + a_{2j}a^\sharp_{j2} + \cdots + a_{nj}a^\sharp_{jn} \end{aligned}$$

Das heißt man kann die Determinante einer Matrix mittels wiederholter Entwicklung nach der i-ten Zeile beziehungsweise der j-ten Spalte berechnen.

Anschließend werden einige wichtige Grundlagen, Definitionen und Zusammenhänge von Vektoren, Matrizen und Matrizengleichungen wiederholt:

Definition 2.2.5 *Es wird im Folgenden mit A' die transponierte Matrix von A bezeichnet, das heißt für $A = (a_{ij})$ ist $A' = (a_{ji})$ die Matrix, die entsteht, wenn die Zeilen und Spalten von A vertauscht werden.*

Definition 2.2.6 *Eine Matrix, die gleich viele Spalten wie Zeilen hat, heißt quadratisch.*

Definition 2.2.7 *Man nennt eine quadratische Matrix $D = (d_{ij}) \in K^{n\times n}$, die $d_{ij} = 0 \; \forall \, i \neq j$ erfüllt, Diagonalmatrix und schreibt auch $D = diag(d_{11}, d_{22}, \ldots, d_{nn})$. Wenn eine Matrix A bekannt ist, schreibt man auch $diag(A)$, wenn man die Matrix meint, die nur aus der Hauptdiagonale von A besteht und deren andere Einträge alle 0 sind.*

Definition 2.2.8 *Eine spezielle Form der Diagonalmatrix ist die Skalarmatrix, bei der die Diagonalelemente alle gleich sind, also $a_{11} = a_{22} = \cdots = a_{nn}$. Für Skalarmatrizen $K = diag(k, k, \ldots, k)$ gilt mit gleich großen Matrizen A, dass $KA = AK = k \cdot A$.*

Definition 2.2.9 *Den Spezialfall einer Skalarmatrix stellt die Einheitsmatrix $I_n = diag(1, 1, \ldots, 1) \in K^{n\times n}$ dar, es gilt für alle Matrizen $IA = AI = A$, solange die Dimensionen eine Multiplikation zulassen. In der Literatur ist die Bezeichnung I statt I_n sehr weit verbreitet, das heißt, meistens entfällt die Dimensionsangabe im Index. Wir haben uns in dieser Arbeit, mit Ausnahme des Grundlagen-Abschnitts, der vorherrschenden Notation angepasst.*

Definition 2.2.10 *Man sagt, dass eine quadratische Matrix A den Rang n hat, und schreibt $rg\, A = n$, wenn A n viele unabhängige Spalten- oder Zeilenvektoren besitzt.*

Definition 2.2.11 *Eine quadratische Matrix heißt singulär, wenn ihre Determinante 0 ist. Andernfalls heißt sie regulär.*

Definition 2.2.12 *Eine Matrix $A = (a_{ij})$ ist genau dann gleich einer Matrix $B = (b_{ij})$, wenn jedes Element von A mit dem entsprechenden Element aus B übereinstimmt. Es bedeutet also $A = B$, dass $a_{ij} = b_{ij} \; \forall j, k$. Das bedeutet, dass eine Matrixgleichung für jedes Element der beiden Matrizen eine Gleichung festlegt.*

Definition 2.2.13 *Eine Matrix ist genau dann symmetrisch, wenn sie gleich ihrer transponierten Matrix ist, also $A = (a_{ij}) = (a_{ji}) = A'$ gilt.*

Definition 2.2.14 *(Addieren und Subtrahieren von Matrizen) Werden zwei Matrizen A und B, die von gleicher Dimension sind, addiert oder subtrahiert, entsteht eine neue Matrix gleicher Größe, deren Einträge den addierten oder subtrahierten Elementen der Ausgangsmatrizen entsprechen.*

Definition 2.2.15 *(Matrixmultiplikation) Für zwei Matrizen $A = (a_{ij}) \in K^{m\times n}$ und $B = (b_{jk}) \in K^{n\times k}$ ist das Matrizenprodukt eine Matrix $C = (c_{ik}) \in K^{m\times k}$, deren Einträge wie folgt berechnet werden können*

$$c_{ik} = \sum_{j=1}^{n} a_{ij} \cdot b_{jk} \, .$$

Lemma 2.2.16 *Dabei gilt, dass die Matrizenmultiplikation im Allgemeinen nicht kommutativ ist, das heißt $AB \neq BA$.*

Definition 2.2.17 *Ist eine quadratische Matrix $A \in K^{n\times n}$ regulär, dann gibt es eine reguläre Matrix $A^{-1} \in K^{n\times n}$, für die gilt, dass $AA^{-1} = A^{-1}A = I_n$. Man nennt A^{-1} die Inverse von A.*

Definition 2.2.18 *Eine quadratische Matrix A ist genau dann orthogonal, wenn $A^{-1} = A'$ gilt. Es folgt, dass für eine orthogonale Matrix A auch A' und A^{-1} orthogonale Matrizen sind.*

Definition 2.2.19 *Die Spur einer $(n \times n)$-Matrix $A = (a_{ij})$ ist die Summe ihrer Hauptdiagonalelemente, also $tr(A) = \sum_{i=1}^{n} a_{ii} = a_{11} + a_{22} + \cdots + a_{nn}$.*

Lemma 2.2.20 *Es gilt $(ABC)' = C'B'A'$ oder allgemeiner formuliert, die Transponierte eines Matrixprodukts ist das Produkt der Transponierten der Faktoren in umgekehrter Reihenfolge.*
Ähnliches gilt auch für die Inverse: $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$ oder wieder verallgemeinert, die Inverse des Produkts entspricht dem Produkt der Inversen der Faktoren in umgekehrter Reihenfolge.

Definition 2.2.21 *Ein Skalar λ ist ein Eigenwert der $(n \times n)$-Matrix A, wenn für mindestens einen Vektor a, der nicht der Nullvektor ist, gilt, dass $A \cdot a = \lambda \cdot a$. Existiert ein solcher Vektor a, nennt man ihn den zum Eigenwert λ gehörigen Eigenvektor der Matrix A. Formt man die obige Gleichung um, erhält man $(A - \lambda I_n)a = 0$.*
Wir nennen $|A-\lambda I_n|$ das charakteristische Polynom $P_n(\lambda)$. Die Nullstellen von $P_n(\lambda)$ sind genau die Eigenwerte λ_i von A. Als Polynom n-ten Grades hat $P(\lambda)$ sicher n Nullstellen, die aber nicht unbedingt voneinander verschieden sein müssen. Die zu den Eigenwerten λ_i gehörigen Eigenvektoren a_i erhält man dann als Lösung der Gleichungssysteme

$$(A - \lambda_i I_n)a_i = 0 \ .$$

Es werden kurz die statistischen Grundlagen gezeigt, die im Folgenden benötigt werden. Es wird dabei X_j als die j-te Variable bezeichnet, schreibt man X_{ji} ist der Wert der j-ten Variable für das i-te Messobjekt gemeint, wobei $j \in \{1, 2, \ldots, n\}$ und $i \in \{1, 2, \ldots, N\}$. Wir werden uns jedoch der in [1] benutzten Fachsprache anpassen und im Folgenden Messobjekt durch Individuum ersetzen.

Definition 2.2.22 *Gibt es N Beobachtungen, definieren wir das Stichprobenmittel einer Variablen als*

$$\overline{X}_j = \sum_{i=1}^{N} \frac{X_{ji}}{N} \ .$$

Definition 2.2.23 *Wird der Ursprung auf das Stichprobenmittel gelegt, nennt man*

$$x_{ji} = X_{ji} - \overline{X}_j$$

Abweichung und

$$s_j^2 = \sum_{i=1}^{N} \frac{x_{ji}^2}{N}$$

die unkorrigierte Stichprobenvarianz. Multipliziert man mit N und dividiert durch $N+1$ erhält man die korrigierte Stichprobenvarianz:

$$s_j^2 = \sum_{i=1}^{N} \frac{x_{ji}^2}{N+1} \ .$$

Die korrigierte Stichprobenvarianz ist erwartungstreu.

Definition 2.2.24 *Mit der unkorrigierten Stichprobenvarianz definiert man die standardisierten Werte:*

$$z_{ji} = \frac{x_{ji}}{s_j}.$$

Definition 2.2.25 *Die Menge aller Werte einer Variablen z_j, also die Werte z_{ji} für $i = 1, 2, \ldots, N$ heißt eine Variable z_j in Standardform. Die Varianz von z_j ist 1 und man definiert die kleinen standardisierten Werte als*

$$z_{ji}^* = \frac{z_{ji}}{\sqrt{N}} \ .$$

Definition 2.2.26 *Für zwei Variablen z_j und z_k ist die Stichprobenkovarianz definiert*

als

$$s_{jk} = \sum_{i=1}^{N} \frac{x_{ji}x_{ki}}{N} \ .$$

Definition 2.2.27 *Der Korrelationskoeffizient innerhalb einer Stichprobe ist gegeben als*

$$r_{jk} = \frac{s_{jk}}{s_j s_k} = \frac{\sum_{i=1}^{N} x_{ji}x_{ki}}{\sqrt{\sum_{i=1}^{N} x_{ji}^2 \sum_{i=1}^{N} x_{ki}^2}} = \sum_{i=1}^{N} \frac{z_{ji}z_{ki}}{N} = \sum_{i=1}^{N} z_{ji}^* z_{ki}^*$$

Definition 2.2.28 *Man definiert den Erwartungswert einer Zufallsvariable X mit Dichte f als $E(X) = \int_{-\infty}^{\infty} xf(x)dx$.*

Definition 2.2.29 *Sei $x = (x_1, x_2, \ldots, x_p)'$ ein Zufallsvektor, dann bezeichnet $E(x) = \mu = (\mu_1, \mu_2, \ldots, \mu_p)' = (E(x_1), E(x_2), \ldots, E(x_p))'$ den Erwartungswert(-vektor) von x. Der Erwartungswert wird also elementeweise genommen, daher wäre $\mu_i = \int_{-\infty}^{\infty} x_i f(x)dx$, wobei $f(x)$, die zu x gehörige Dichte darstellt.*

Definition 2.2.30 *Ist X eine Zufallsvariable und ist $E(|X|) < \infty$, dann kann die Varianz allgemein definiert werden als $var(X) = E((X - E(X))^2)$. Die Dichte kann auch mit Hilfe eines Verschiebungssatzes berechnet werden als $var(X) = E(X^2) - E(X)^2$.*

Definition 2.2.31 *Allgemein definiert man die Kovarianz für zwei Zufallsvariable X, Y als $cov(X, Y) = E((X - E(X))(Y - E(Y)))$, wobei natürlich die Erwartungswerte von X und Y existieren müssen. Für $X = Y$ erhält man $cov(X, X) = E((X - E(X))(X - E(X))) = E((x-E(X))^2) = var(X)$, das heißt die Kovarianz kann als Verallgemeinerung der Varianz angesehen werden. Die Berechnung der Kovarianz kann durch folgenden Verschiebungssatz oft vereinfacht werden: $cov(X, Y) = E(XY) - E(X)E(Y)$.*

Definition 2.2.32 *Sei $x = (x_1, x_2, \ldots, x_p)'$ ein Zufallsvektor und $\sigma_{ij} = cov(x_i, x_j)$ für $i \neq j$ die Kovarianz und $\sigma_i^2 = \sigma_{ii} = var(x_i)$ die Varianz von x_i. Dann heißt*

$$cov(x) = E((x - \mu)(x - \mu)') = \begin{pmatrix} \sigma_{11} & \cdots & \sigma_{1p} \\ \vdots & & \vdots \\ \sigma_{p1} & \cdots & \sigma_{pp} \end{pmatrix} = \Sigma$$

Kovarianzmatrix von x. Für ein x, das mit μ und Σ verteilt ist, schreibt man auch kurz $x \sim (\mu, \Sigma)$.

Definition 2.2.33 *Weiters definiert man allgemein die Korrelation zwischen x_i und x_j als*

$$\varrho(x_i, x_j) = \varrho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} = \frac{cov(x_i, x_j)}{\sqrt{var(x_i)var(x_j)}}$$

und nennt

$$R = \begin{pmatrix} 1 & \varrho_{12} & \cdots & \varrho_{1p} \\ \varrho_{21} & 1 & \cdots & \varrho_{2p} \\ \vdots & & & \vdots \\ \varrho_{p1} & \varrho_{p2} & \cdots & 1 \end{pmatrix}$$

die Korrelationsmatrix von x.

Da wir uns im Folgenden vermehrt mit den Kovarianzmatrizen beschäftigen, die die Faktorenanalyse mit sich bringt, setzen wir uns im Vorfeld noch mit symmetrischen Matrizen auseinander und betrachten einige besondere Eigenschaften. Siehe hierfür im Besonderen [7] Anhang A.11.

In den folgenden Erkenntnissen ist A immer eine symmetrische $(n \times n)$-Matrix.

Lemma 2.2.34 *Eigenschaften symmetrischer Matrizen*

- *Alle Eigenwerte von A sind reell.*
- *Die Eigenvektoren verschiedener Eigenwerte sind paarweise orthogonal.*
- *Es gibt eine orthogonale Matrix P, so dass $P'AP = \Lambda$ beziehungsweise $A = P\Lambda P'$ gilt. Die Matrix Λ ist eine Diagonalmatrix, deren Diagonale genau aus den Eigenwerten von A besteht. Die Spaltenvektoren von P andererseits entsprechen den paarweise orthonormalen Eigenvektoren von A. Praktisch bei dieser Art von Diagonalisierung ist, dass die charakteristischen Daten von A in Λ erhalten bleiben, da Λ dasselbe charakteristische Polynom, dieselben Eigenwerte, dieselbe Spur und den gleichen Rang wie A hat.*
- *Ist A regulär, gilt sogar, dass $A^n = \Gamma\Lambda\Gamma'$ mit ganzzahligem n ist und die Diagonalelemente von Λ^n entsprechen den n-fachen Potenzen der Eigenwerte von A.*

Lemma 2.2.35 *Die Eigenwerte λ_i einer $(n \times n)$-Matrix A erfüllen außerdem folgende praktische Eigenschaften:*

- $tr(A) = \sum_{i=1}^{n} \lambda_i$
- $det(A) = \prod_{i=1}^{n} \lambda_i$

2.3 Geschichte und Anwendungen der Faktorenanalyse

Dieser Abschnitt befasst sich mit den Wurzeln der Faktorenanalyse. Vergleiche hierfür [1] und [3].

Die Faktorenanalyse wurde in der Psychologie begründet und lange Zeit auch ausschließlich in der psychologischen Theorie benutzt. Das führte dazu, dass die Faktorenanalyse oftmals zur Psychologie gezählt wurde, obwohl sie eigentlich in den Fachbereich der Statistik fällt.

Es ist allgemein anerkannt, dass C. Spearman die Faktorenanalyse 1904 in seinem Artikel "General Intelligence, Objectively Determined and Measured"[1] einführte.
Aus Korrelationen zwischen diversen psychologischen Intelligenztests löste Spearman einen allgemeinen Intelligenzfaktor, den g-Faktor, und einen spezifischen Faktor, der sich von Test zu Test unterschied, heraus. Diese sogenannte Zwei-Faktoren-Theorie basiert auf der Idee, dass diese zwei Faktoren ausreichen, um die Korrelationen zwischen den Tests zu erklären.

Es handelte sich zu Beginn nur um eine Ein-Faktor-Analyse, die jedoch den Grundstein für die Entwicklung und Bereicherung der Theorie in den nächsten Jahrzehnten legte. Spearman selbst widmete der Intelligenztheorie und der Faktorenanalyse weitere vierzig Jahre seines Lebens und wird als Vater der Faktorenanalyse angesehen.

Als sich jedoch in den 1930er-Jahren immer deutlicher zeigte, dass die Zwei-Faktoren-Theorie unzureichend zur Erklärung aller empirischen Daten war, kamen neue Ideen zur Faktorenanalyse auf.

Ein wichtiger Schritt hin zur heutigen Faktorenanalyse war die Bi-Faktor-Methode von K. Holzinger. Er weitete die restriktive Anforderung Spearmans aus und ließ Gruppenfaktoren zu. Diese Gruppenfaktoren entstanden, indem man (oft etwas willkürlich) Variable zu Gruppen zusammenfasste, die dann durch einen einzelnen Faktor repräsentiert

[1] Dieser Artikel kann hier eingesehen werden: `http://psychclassics.yorku.ca/Spearman/`

wurden. Holzinger behielt jedoch weiterhin den allgemeinen g-Faktor von Spearman bei. Das Verfahren wurde damit komplexer als Spearmans, konnte jedoch auch negative Korrelationen erklären, die noch Probleme bei Spearmans Zwei-Faktoren-Theorie darstellten.

Schließlich setzte sich die gleichzeitige Extraktion mehrerer Faktoren durch. Erstmals tauchte das Konzept bereits 1919 bei J.C. Maxwell Garnett auf.
Breitere Bekanntheit erlangte es jedoch erst durch die Arbeit von L.L. Thurstone, der auch den Namen multiple-factor analysis (multiple Faktorenanalyse) prägte. Thurstones Beitrag zur Erweiterung der Theorie war vielfältig, seine wichtigste Erkenntnis war aber der Zusammenhang zwischen dem Rang der Korrelationsmatrix und der Anzahl von Faktoren:
Die mindestens benötigte Anzahl von Faktoren ist gleich dem Rang der Korrelationsmatrix.

In den darauf folgenden Jahren gab es noch weitere große Einflussfaktoren: die mathematische Statistik und das Aufkommen von elektronischen Rechenmaschinen beziehungsweise Computern.
Die mathematische Statistik brachte mathematisch korrekte Lösungen und Methoden, die zwar Fortschritte für die Modelle bedeuteten, aber in der Praxis oft keinen echten Nutzen brachten. Großen Nutzen brachten hingegen die elektronischen Rechenmaschinen, die die oft rein theoretischen Methoden zum ersten Mal berechenbar machten. Der Computer ermöglichte auch erstmals die Faktorenanalyse für große Korrelationsmatrizen und die Kontrolle durch Simulationen.

Hauptanwendungsbereich der Faktorenanalyse war und ist noch heute die Psychologie, im Besonderen die Intelligenztheorie, wo sie, wie bereits beschrieben, auch ihren Ursprung nahm. Im Laufe der Jahre wurde sie aber auch in anderen Forschungsfeldern mit Erfolg eingesetzt. Sowohl Harman [1] als auch Überla [3] beschreiben solche Einsatzgebiete, Überla schreibt jedoch:
„Die Faktorenanalyse wurde in den meisten Fachgebieten von Außenseitern verwendet und ist lediglich in bestimmten Bereichen der Psychologie die tragende Methodik.“([3] S.7)
Dennoch wurden auch Erfolge außerhalb der Psychologie erzielt. Einige Beispiele:

- Medizin: Schätzung des nicht direkt messbaren Thyroxinspiegels ohne Schilddrüsenfunktionstest
- Wirtschaft: Investment-Entscheidungen bei Unsicherheiten

- Meteorologie: Vorhersagen
- Biologie: Klassifikation von Hefen

In der Psychologie werden ebenfalls auch andere Fortschritte erzielt. So kann zum Beispiel dank der Faktorenanalyse mit einer Reihe von Testbatterien eine Trennung zwischen Neurotikern und Normalpersonen vorgenommen werden, die den Diagnosen von Fachärzten sehr ähnelt.
Eine ausführlichere Liste von Studien, die die Faktorenanalyse beinhalten, findet sich bei [1].

3 Das faktoranalytische Modell

Im diesem Abschnitt werden das faktoranalytische Modell und die allgemeinen Annahmen der Faktorenanalyse betrachtet sowie die Existenz sowie Eindeutigkeit von Lösungen. Vergleiche [7] für die Notation und die ersten drei Unterkapitel sowie [1] für den Rest des Kapitels.

3.1 Grundmodell

Bei der Faktorenanalyse wird allgemein von p Zufallsvariablen y_i ausgegangen. Diese werden in einem Zufallsvektor $y = (y_1, y_2, \ldots, y_p)'$ zusammengefasst und es werden der Erwartungswert $\mu = (\mu_1, \mu_2, \ldots, \mu_p)'$ sowie die Kovarianzmatrix Σ bestimmt. Wie bereits erwähnt, basiert das Modell auf der Annahme, dass es zwischen diesen Variablen und anderen, nicht direkt messbaren Zufallsvariablen einen linearen Zusammenhang gibt. Weiters nehmen wir an, dass die Anzahl dieser unbeobachtbaren Faktoren kleiner ist als die Anzahl der y_i und nennen $f_1, f_2, \ldots, f_k$ gemeinsame Faktoren, wobei $k < p$ ist. Wir erhalten damit ein lineares Gleichungssystem:

$$\begin{aligned} y_1 &= \mu_1 + l_{11}f_1 + l_{12}f_2 + \cdots + l_{1k}f_k + e_1 \\ y_2 &= \mu_2 + l_{21}f_1 + l_{22}f_2 + \cdots + l_{2k}f_k + e_2 \\ &\vdots \\ y_p &= \mu_p + l_{p1}f_1 + l_{p2}f_2 + \cdots + l_{pk}f_k + e_p \end{aligned} \tag{1}$$

Die Koeffizienten l_{ij} mit $i = 1, \ldots, p$ und $j = 1, \ldots, k$ heißen Ladung und stellen den Einfluss dar, den ein bestimmter gemeinsamer Faktor f_j auf die Variable y_i hat. Die Variablen e_i stellen alle Einwirkungen dar, die nur die Variable y_i betreffen, sie fassen spezifische oder Einzelrestfaktoren sowie Messfehler zusammen.

Fasst man die Faktoren, Fehler und die Ladungen als Vektoren beziehungsweise als Matrix zusammen, also $f = (f_1, f_2, \ldots, f_k)'$, $e = (e_1, e_2, \ldots, e_p)$ und die $(p \times k)$-Ladungsmatrix $L = (l_{ij})$, so lässt sich (2) einfach ausdrücken als:

$$y - \mu = Lf + e \tag{2}$$

Da sowohl die Faktoren als auch die Einzelrestfaktoren nicht beobachtbar sind, kann man ihren Nullpunkt frei wählen und setzt daher ihre Erwartungswerte ohne Beschränkung der Allgemeinheit auf 0, das heißt $E(f) = 0$ und $E(e) = 0$.
Weiters nehmen wir an, dass die Einzelrestfaktoren e unkorreliert sind, aber beliebige

Varianz haben können und nennen die $(p \times p)$-Matrix V die Einzelrestvarianzmatrix, wobei V folgendermaßen aussieht:

$$cov(e, e) = E(ee') = V = diag(v_1^2, v_2^2, \ldots, v_p^2) \ .$$

Zusätzlich gehen wir davon aus, dass die Faktoren f standardisiert sind, also zusätzlich zu Erwartungswert 0 auch Varianz 1 haben, und verlangen noch die Unkorreliertheit der Faktoren f und den Einzelrestfaktoren e, also $cov(f, e) = E(fe') = 0$.

Die bisher genannten Annahmen gelten sowohl für das orthogonale als auch für das oblique Faktorenmodell. Der Unterschied zwischen diesen beiden Modellen besteht lediglich darin, dass im orthogonalen Faktorenmodell die zusätzliche Forderung nach untereinander unkorrelierten Faktoren f besteht, also soll $cov(f, f) = E(ff') = I$ gelten. Wir werden uns in weiterer Folge nur mit dem orthogonalen Faktorenmodell beschäftigen.

3.2 Fundamentaltheorem der Faktorenanalyse

Aus den allgemeinen Annahmen und der Forderung nach Unkorreliertheit der Faktoren, lässt sich das Fundamentaltheorem der Faktorenanalyse herleiten. Ist Σ die Kovarianzmatrix von y, so gilt:

$$\begin{aligned}
\Sigma &= E((y-\mu)(y-\mu)') = E((Lf+e)(Lf+e)') \\
&= E(Lff'L') + E(Lfe') + E(ef'L') + E(ee') \\
&= LL' + 0 + 0 + V \\
&= LL' + V
\end{aligned} \tag{3}$$

Die Gleichung (3) heißt Grundgleichung oder auch Fundamentaltheorem der Faktorenanalyse. Aus ihr lässt sich auch für die p Varianzen s_i^2 von y_i folgern:

$$s_i^2 = l_{i1}^2 + \cdots + l_{ik}^2 + v_i^2 \ .$$

Man nennt den Anteil der gemeinsamen Faktoren an der Varianz s_i^2, also

$$h_i^2 = l_{i1}^2 + \cdots + l_{ik}^2 \ ,$$

die Kommunalität von y_i. Weiters folgt, dass wegen

$$\begin{aligned} cov(y, f) &= E((y - \mu)f') = E((Lf + e)f') \\ &= E(Lff' + ef') = L \cdot E(ff') + E(ef') \\ &= L \end{aligned} \tag{4}$$

die Kovarianz von y mit f gleich L ist, da laut unserer Annahmen die Terme bei (4) $E(ff') = I$ und $E(ef') = E(fe') = 0$ sind.

Die Aufgabe der Faktorenanalyse ist es nun, ausgehend von den Modellannahmen sowie der Grundgleichung (3), die unbekannten Größen k, L und V zu schätzen. Dafür wird die empirische Kovarianzmatrix der Daten herangezogen und man kann (bei entsprechend großer Stichprobe) den Vektor μ durch sein Stichprobenmittel, also $\overline{y} = \frac{1}{N}\sum_{n=1}^{N} y_n$ schätzen. Damit wird die Kovarianzmatrix Σ und ihre Zerlegung zum Hauptziel der Bemühungen.

Bevor wir uns jedoch den Methoden zur Berechnung der Matrix Σ und ihrer Zerlegung zuwenden, werden wir noch die Existenz und Eindeutigkeit eben dieser Zerlegung betrachten.

3.3 Existenz und Eindeutigkeit

Damit das Modell gültig ist, muss es eine Matrix Σ geben, die wie in (3) in L und V zerlegt werden kann. Punkt 3 des Lemmas (2.2.34) garantiert die Diagonalisierbarkeit der Kovarianzmatrix.
Im Falle, dass $p = k$, ist Σ nach (3) darstellbar mit $V = 0$.
Auch im Fall $k \leq p$ und vorgegebenem V ist eine passende Zerlegung möglich, wenn $\Sigma - V$ positiv semidefinit ist mit $rg(\Sigma - V) = k$. Das bedeutet aber auch, dass man Kovarianzmatrizen nicht generell nach (3) zerlegen kann.
Oft ist eine Zerlegung mathematisch zwar möglich, aber dann nicht mehr sinnvoll, vor allem wenn einzelne Elemente von L größer als 1 oder die Einzelrestvarianzen negativ würden. Diese Art von Problem wird später noch ausführlicher betrachtet, wenn wir die Methoden und die Heywood-Fälle vorstellen.

Wir haben uns nun bereits mit der Existenz von passenden L und V beschäftigt, die logische Folgefrage setzt sich mit der Eindeutigkeit auseinander. Oder anders gesagt: Unter welchen Bedingungen können wir L eindeutig festlegen, wenn wir k und V kennen?

Im Fall $k = 1$ ist unsere Ladungsmatrix L nichts anderes als ein Spaltenvektor der Dimension p. Weil wir von standardisierten Faktoren f ausgegangen sind, ist dieser Vektor bis auf sein Vorzeichen eindeutig bestimmt. Das Vorzeichen stellt jedoch nur die Richtung der Faktorvariablen dar und ist nicht weiter wichtig, weshalb es vernachlässigt werden kann und unser Modell für den Fall $k = 1$ eindeutig bestimmt ist.

Größere Probleme treten jedoch bei $k > 1$ auf. Hier ist L nicht mehr eindeutig, was leicht ersichtlich ist, wenn wir statt L die Matrix LM verwenden und f durch $M'f$ ersetzen, wobei M eine orthogonale $(k \times k)$-Matrix ist. Nennt man $L^* = LM$ und $f^* = M'f$ und setzt in die Grundgleichung (2) ein, folgt:

$$y - \mu = Lf + e = LMM'f + e = L^*f^* + e$$

Zuerst verändert sich an der Gleichung nicht viel und auch die Eigenschaften bleiben erhalten:

$$\begin{aligned} E(f^*) &= M'E(f) = 0 \\ cov(f^*) &= M'cov(f)M = M'IM = M'M = I \end{aligned}$$

Die neue Ladungsmatrix LM erfüllt sogar wie L die Zerlegung von Σ:

$$\Sigma = LL' + V = LMM'L' + V = L^*L^{*\prime} + V$$

Die Ladungsmatrix L ist daher nicht eindeutig, sondern nur bis auf orthogonale Transformationen eindeutig bestimmt. Diese Transformationen stellen eine Rotation des Koordinatensystems dar und helfen in der Praxis oft bei Interpretationsschwierigkeiten. Für jetzt sind sie jedoch hinderlich, weshalb wir eine Nebenbedingung finden wollen, die eine eindeutige Lösung sichert.

Wir gehen dabei von einer skalierten Kovarianzmatrix aus:

$$V^{-\frac{1}{2}}\Sigma V^{-\frac{1}{2}\prime} = \Sigma^*$$

Man sieht daraus, dass gilt:

$$V^{-\frac{1}{2}}(\Sigma - V)V^{-\frac{1}{2}\prime} = \Sigma^* - I$$

Diese Differenz ist positiv semidefinit und von Rang k, daher genauso wie $\Sigma - V$ darstellbar

als

$$\Sigma^* - I = \Omega\Delta\Omega' \ ,$$

wie in (2.2.34) angegeben. Das heißt

$$L = V^{\frac{1}{2}}\Omega\Delta^{\frac{1}{2}\prime} \tag{5}$$

erfüllt die Grundgleichung (3), weil gilt, dass

$$LL' = V^{\frac{1}{2}}\Omega\Delta\Omega' V^{\frac{1}{2}\prime} = V^{\frac{1}{2}}(\Sigma^* - I)V^{\frac{1}{2}\prime} = \Sigma - V \ .$$

Damit (5) die eindeutige Lösung wird, ist folgende Nebenbedingung nötig:

$$L'V^{-1}L = \Delta^{\frac{1}{2}}\Omega' V^{\frac{1}{2}} V^{-1} V^{\frac{1}{2}}\Omega\Delta^{\frac{1}{2}} = \Delta \ \text{ diagonal}$$

Diese Bedingung legt L bis auf Vorzeichen und Umordnung der Spalten fest, was sich aber ebenfalls beheben lässt, indem man Δ eine gewissen Reihenfolge in den Diagonalelementen vorschreibt. Man hat damit ein Kriterium, das es ermöglicht, aus der Menge aller Lösungen, die man durch orthogonale Transformation erhalten würde, genau eine Lösung auszuwählen.

3.4 Rotationsproblem

Es ist zu beachten, dass die im letzten Abschnitt getroffene Auswahl, obwohl sie willkürlich erfolgt ist, aus mathematischer Sicht natürlich sinnvoll und praktisch (zum Beispiel in der ML-Methode), jedoch der Interpretation des Ergebnisses nicht immer dienlich ist. Wie bereits erwähnt, ist es oft nützlich oder sogar notwendig, die gefundenen Faktoren zu transformieren, um sie sinnvoll zu interpretieren.
Will man transformieren, stellt sich jedoch zusätzlich die Frage, welche die besten Transformationen sind, damit das Ergebnis möglichst gut interpretiert werden kann. Die Thematik und die Suche nach der besten Transformation für die Interpretierbarkeit ist als Rotationsproblem bekannt. Dafür gibt es einige Methoden, hier sollen aber nur das Varimax- und das Quartimax-Kriterium vorgestellt werden.

3.4.1 Varimax-Kriterium

Das Ziel dieses Kriteriums ist es, Faktoren zu finden, deren Ladungsquadrate entweder sehr groß oder sehr klein sind. Die Varianz der Ladungsquadrate soll maximiert werden, was dem Kriterium auch den Namen gibt. Mathematisch angeschrieben, heißt das:
Ist $\tilde{L} = (\tilde{l}_{ij})$ die Ladungsmatrix der transformierten Faktoren, $\tilde{L} = LM$, dann soll gelten:

$$e_1 = \sum_{r=1}^{k} \sum_{i=1}^{p} \left(\tilde{l}_{ir}^2 - \frac{d_r}{p} \right)^2 \to max \ ,$$

wobei

$$d_r = \sum_{i=1}^{p} \tilde{l}_{ir}^2 \ .$$

Es misst e_1 also die quadratische Abweichung der Ladungsquadrate vom jeweiligen Mittelwert. Bei der Methode wird e_1 über alle orthogonalen Matrizen $M \in \mathbb{R}^{k,k}$ maximiert.

3.4.2 Quartimax-Kriterium

Das Quartimax-Verfahren ist sehr ähnlich, versucht aber statt der Spalten die Zeilen zu verändern:

$$e_2 = \sum_{i=1}^{p} \sum_{r=1}^{k} \left(\tilde{l}_{ir}^2 - \frac{d_i}{p} \right)^2 \to max \ ,$$

wobei

$$d_i = \sum_{r=1}^{k} \tilde{l}_{ir}^2 \ .$$

3.5 Varianz und Korrelation

Dieses Unterkapitel beschäftigt sich mit den Varianzen und Korrelationen im Faktorenmodell.

Da wir wissen, dass die gemeinsamen Faktoren von den spezifischen Faktoren unkorreliert sind und die gemeinsamen Faktoren untereinander auch unkorreliert sind, gilt mit der Definition (2.2.23) der unkorrigierten Stichprobenvarianz angewandt auf unser Modell:

$$s_j^2 = 1 = l_{j1}^2 + l_{j2}^2 + \cdots + l_{jk} + v_j^2 \ .$$

Hier drücken die Terme l_{ji} den Anteil des i-ten Faktors an der Varianz der j-ten Variable aus. Der gesamte Beitrag eines Faktors f_j zur Varianz lässt sich daher ausdrücken als

$$W_n = \sum_{j=1}^{p} l_{jn} \qquad (n = 1, 2, \ldots, k)$$

und der Beitrag aller Faktoren zur Varianz ist daher als

$$W = \sum_{j=1}^{k} W_j$$

darstellbar. Es kann $\frac{W}{p}$ auch als Messparameter für die Vollständigkeit der Faktorenanalyse eingesetzt werden.

Neben den Kommunalitäten h_j^2 wollen wir uns jetzt auch mit den Einzelrestvarianzen v_j^2 (englisch: uniquenesses) beschäftigen. Wie bereits erwähnt, fassen die Einzelrestfaktoren und daher auch die Einzelrestvarianzen viele verschiedene Fehler zusammen. Sie werden jetzt aber wieder aufgespalten, um diverse Zusammenhänge deutlicher erkennen zu können. Man spaltet v_j^2 daher in einen spezifischen Teil (auch Exaktheit) und einen Fehlerteil auf. Der spezifische Teil repräsentiert den Fehler, der entsteht, weil genau diese Variablen ausgewählt wurden. Hätte man andere Variable oder mehr Variable genommen, wären eventuell andere Korrelationen zustande gekommen, die in der jetzigen Auswahl unterrepräsentiert oder gar nicht repräsentiert sind. Der Fehlerteil hingegen steht für die Messunzuverlässigkeit. Setzt man jetzt $z_j = y_j - \mu_j$ kann man das lineare Gleichungssystem (1) umschreiben zu

$$z_j = l_{j1}f_1 + l_{j_2}f_2 + \cdots + l_{jk}f_k + b_jS_j + c_jE_j \qquad (j = 1, 2, \ldots, p),$$

wobei S_j den spezifischen Teil und E_j den Fehlerteil darstellt und b_j und c_j ihre jeweiligen Koeffizienten. Dann gilt für die Einzelrestvarianzen, dass

$$v_j^2 = b_j^2 + c_j^2$$

und dass die Gesamtvarianz auch geschrieben werden kann als

$$s_j^2 = 1 = h_j^2 + v_j^2 = h_j^2 + b_j^2 + c_j^2 \ .$$

Kennt man die Zuverlässigkeit r_{jJ} von z_j, kann man den Fehleranteil c_j^2 an der Varianz

berechnen als

$$c_j^2 = 1 - r_{jJ} \ .$$

Daraus und aus

$$h_j^2 + b_j^2 + c_j^2 = 1$$

kann rasch geschlossen werden, dass

$$r_{jJ} = h_j^2 + b_j^2$$

sein muss und man kann auch gleich noch die Kommunalitäten abschätzen:

$$h_j^2 = r_{jJ} - b_j^2 \leq r_{jJ} \ .$$

Angelehnt an die Tabelle aus [1] S.20 mit etwas abgeänderter Notation gilt:

$$\begin{array}{lllllllllll}
\text{Gesamtvarianz} & 1 & = & h_j^2 & + & b_j^2 & + & c_j^2 & = h_j^2 & + & v_j^2 \\
\text{Zuverlässigkeit} & r_{jJ} & = & h_j^2 & + & & b_j^2 & & = 1 & - & c_j^2 \\
\text{Kommunalität} & h_j^2 & = & h_j^2 & & & & & = 1 & - & v_j^2 \\
\text{Einzelrestvarianz} & v_j^2 & = & & b_j^2 & & + & c_j^2 & = 1 & - & h_j^2 \\
\text{Exaktheit} & b_j^2 & = & & b_j^2 & & & & = v_j^2 & - & c_j^2 \\
\text{Fehler} & c_j^2 & = & & & & & c_j^2 & = 1 & - & r_{jJ}
\end{array} \tag{6}$$

Als nächstes wollen wir uns den Korrelationen widmen, die wir unter anderem für die Minres Methode benötigen. Dafür gehen wir wieder vom Grundmodell aus, wobei wir wie vorhin $z_j = y_j - \mu_j$ setzen:

$$\begin{aligned}
z_1 &= l_{11}f_1 + l_{12}f_2 + \cdots + l_{1k}f_k + e_1 \\
z_2 &= l_{21}f_1 + l_{22}f_2 + \cdots + l_{2k}f_k + e_2 \\
&\vdots \\
z_p &= l_{p1}f_1 + l_{p2}f_2 + \cdots + l_{pk}f_k + e_p \ .
\end{aligned} \tag{7}$$

Um auf eine sogenannte Faktorstruktur zu kommen, die die Korrelationen zwischen den Variablen und den Faktoren zeigt, muss jede Gleichung von (7) mit dem entsprechenden Faktor multipliziert werden, über die Anzahl der Beobachtungen N summiert und

anschließend durch N dividiert werden. Man erhält:

$$\begin{array}{lcllllllll} r_{z_j f_1} & = & l_{j1} & + & l_{j2} r_{f_1 f_2} & + & l_{j3} r_{f_1 f_3} & + & \ldots & + & l_{jk} r_{f_1 f_k} \\ r_{z_j f_2} & = & l_{j1} r_{f_2 f_1} & + & l_{j2} & + & l_{j3} r_{f_2 f_3} & + & \ldots & + & l_{jk} r_{f_2 f_k} \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ r_{z_j f_k} & = & l_{j1} r_{f_k f_1} & + & l_{j2} r_{f_k f_2} & + & l_{j3} r_{f_k f_3} & + & \ldots & + & l_{jk} \; . \end{array} \tag{8}$$

Für die Anwendung in der Praxis ist die Faktorstruktur ein wichtiger Bestandteil der Faktorenanalyse und ein praktisches Werkzeug, um die Korrelation zwischen Daten und Faktoren zu sehen und die Faktoren zu identifizieren und zu benennen. Sie wurden der Vollständigkeit halber angeführt, wir werden uns aber an dieser Stelle nicht mit der praktischen Anwendung aufhalten, sondern mit der Korrelation fortfahren.

Ein wichtiges Mittel dafür ist die reproduzierte Korrelation $\hat{r}_{jk}$, jene Korrelation, die wir aus unseren Gleichungen (7) erhalten. Man bestimmt sie, indem man zwei der Gleichungen aus (7) miteinander multipliziert, über die Summe der N Beobachtungen summiert und danach wieder durch N dividiert. Durch unsere Annahmen zur Unkorreliertheit der Faktoren, vereinfacht sich dieser Ausdruck zu

$$\hat{r}_{ji} = l_{j1} l_{i1} + l_{j2} l_{i2} + \cdots + l_{jk} l_{ik} \qquad (j \neq i; \quad j, i = 1, 2, \ldots, p)$$

und wie bei Korrelationen gewohnt, gilt für $i = j$

$$\hat{r}_{jj} = 1 \qquad (j = 1, 2, \ldots, p) \; .$$

Werden die reproduzierten Korrelationen zu einer Matrix zusammengefasst, erhält man die reproduzierte Korrelationsmatrix, die in der Minres Methode zur Anwendung kommt.

Ziel ist es jetzt, zu zeigen, dass die Grundgleichung (3) auch für Korrelationsmatrizen hält. Wir definieren dafür die Matrix Z, die aus den kleinen standardisierten Werten z^*_{ji} besteht, wie sie in den Grundlagen definiert sind, also

$$Z = \begin{pmatrix} z^*_{11} & z^*_{12} & \cdots & z^*_{1N} \\ z^*_{21} & z^*_{22} & \cdots & z^*_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ z^*_{p1} & z^*_{p2} & \cdots & z^*_{pN} \end{pmatrix} \; .$$

Das erspart uns im Folgenden die Division durch N. Zusätzlich schreiben wir noch

$$F = \begin{pmatrix} F_{11} & F_{12} & \dots & F_{1N} \\ F_{21} & F_{22} & \dots & F_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ F_{k1} & F_{k2} & \dots & F_{kN} \end{pmatrix} ,$$

wobei F_{ij} den Anteil des i-ten gemeinsamen Faktors am j-ten Individuum beziehungsweise der j-ten Beobachtung bezeichnet.

Das umgeschriebene Grundmodell (7) kann dann geschrieben werden als

$$Z = LF$$

und wenn man auf beiden Seiten mit F' multiplizieren, folgt

$$ZF' = LFF' .$$

Weiters erhält man per Definition die beobachtete Korrelationsmatrix als

$$R = ZZ' .$$

Setzt man jetzt statt der beobachteten Korrelationsmatrix die reproduzierte Korrelationsmatrix ein, in deren Diagonale die Kommunalitäten stehen, erhält man

$$\hat{R} = ZZ'$$

und folglich [2]

$$\hat{R} = LFF'L' .$$

Da unsere Faktoren unkorreliert sind, vereinfacht sich das zu

$$\hat{R} = LL' \tag{9}$$

und wir sehen, dass die Grundgleichung auch für Korrelationsmatrizen gilt. Das bedeutet nicht, dass die Matrix L in beiden Fällen dieselbe ist, nur dass man auch für Korrelationsmatrizen eine derartige Zerlegung finden kann.

[2] Genau wie in [1] nehmen wir hier an, dass die Residuen verschwinden, wenn wir die beobachtete durch die theoretische (also aus dem Modell berechnete) Korrelation ersetzen.

4 Methoden der Faktorenanalyse

In diesem Kapitel zeigen wir ein paar Methoden der Faktorenanalyse auf, also die verschiedenen Möglichkeiten L und V zu berechnen oder zu schätzen. Die Beschreibung der Methoden ist stark angelehnt an die Beschreibungen in [7] Kap. 11.2 und 11.3 und behält auch die Notation bei.

4.1 Maximum-Likelihood Methode

Ausgangspunkt der Maximum-Likelihood-Methode ist das Maximum Likelihood Prinzip, das heißt man berechnet ML-Schätzer für L und V und kann die passende Anzahl von Faktoren durch Likelihood-Quotienten-Tests annähern. Einzige Voraussetzung der Methode ist eine Normalverteilung der Daten, was für gewöhnlich kein allzu großes Problem darstellt, da die Daten standardisiert werden können. Wir beginnen also mit den Daten $y \sim N_p(\mu, \Sigma)$ und $rg(\Sigma) = p$.
Ohne diese Voraussetzung versagt die Methode jedoch nicht völlig, sondern es können trotzdem noch Schätzer berechnet werden, die zumindest asymptotisch dieselben wünschenswerten Eigenschaften haben wie die ML-Schätzer. Für diese nicht normalverteilte Variante muss nur die Existenz der ersten beiden Momente μ und Σ sowie die Semi-Definitheit von Σ vorausgesetzt werden.

Schätzung

Wir gehen hier davon aus, dass N unabhängige Beobachtungen von $y = (y_1, \ldots, y_p)' \sim N(\mu, \Sigma)$ vorliegen, wobei natürlich $\Sigma = cov(y)$ ist, und die Anzahl k der Faktoren bereits bekannt ist. Wie man die passende Anzahl finden kann, werden wir später noch erläutern. Ausgehend von

$$\Sigma = LL' + V \text{ und } V = diag(v_1^2, \ldots, v_p^2) \text{ mit } v_i^2 > 0$$

mit der Bedingung1

$$L'V^{-1}L \text{ ist diagonal} \tag{10}$$

gilt es also L und V zu schätzen.

Als erwartungstreuer Schätzer für Σ dient die empirische Kovarianzmatrix S von y, die

definiert ist als

$$S = \frac{1}{N}\sum_{n=1}^{N}(y_n - \overline{y})(y_n - \overline{y})' \ ,$$

wobei wie bereits erwähnt

$$\overline{y} = \frac{1}{N}\sum_{n=1}^{N} y_n \ .$$

Wir wissen, dass S daher einer Wishart-Verteilung folgt, es gilt $(N-1)S \sim W_p(\Sigma, N-1)$, deren Dichtefunktion gegeben ist als

$$L(\Sigma|S) = c\ |S|^{\frac{N-p-2}{2}}\ |\Sigma|^{\frac{-(N-1)}{2}}\ e^{-\frac{N-1}{2}\ tr(S\Sigma^{-1})} \ ,$$

wobei c eine Konstante ist, die im ML-Prinzip nicht weiter stört.

Die gewünschten Schätzer $\hat{L}$ und $\hat{V}$ erhält man jetzt wie gewohnt durch die Maximierung der Likelihoodfunktion $L(\Sigma|S) = L(L, V|S)$ oder der log-Likelihoodfunktion

$$ln\ L(L, V|S) = C + \tfrac{N-p-2}{2}\ ln|S| - \tfrac{(N-1)}{2}\ ln|\Sigma| - \tfrac{(N-1)}{2}\ tr(S\Sigma^{-1})$$

mit neuer Konstante C unter der Bedingung (10). Das ist klarerweise gleichbedeutend mit einer Minimierung von

$$f(l, V) = ln|\Sigma| + tr(S\Sigma^{-1}) \ .$$

Leitet man jetzt partiell nach L und V ab und setzt diese Ableitungen gleich 0, kommt man auf folgende implizite Gleichungen für $\hat{L}$ und $\hat{V}$:

$$\begin{aligned} &\hat{L} = S\hat{\Sigma}^{-1}\hat{L} && \text{mit } \hat{\Sigma} = \hat{L}\hat{L}' + \hat{V} \\ &diag(\hat{\Sigma}) = diag(S) && \text{mit } \hat{\Sigma} = \hat{L}\hat{L}' + \hat{V} \\ &\hat{L}'\hat{V}^{-1}\hat{L} \text{ diagonal} \end{aligned} \qquad (11)$$

Die durch (11) gegebenen ML-Schätzer sind mit einer Wahrscheinlichkeit von 1 konsistent und asymptotisch normalverteilt, sofern Σ eindeutig zerlegbar ist, die Annahmen unserer Ausgangssituation stimmen und $rg(\Sigma) = p$ ist.

Bestimmung von k

Falls die Faktorenzahl k vorgegeben oder anderweitig bereits bestimmt wurde, kann man testen, ob k Faktoren die Daten erklären. Hat man jedoch k noch nicht bestimmt, so kann man das iterativ tun, was allerdings mit großem Aufwand verbunden ist:

- Man wählt ein kleines k_0 als Startwert, wobei $k = 0$ der Frage entspricht, ob die Faktorenanalyse in diesem Fall überhaupt sinnvoll ist.
- Mit gewähltem Startwert führt man die ML-Schätzung durch und testet danach mittels Likelihood-Quotienten-Test, ob $k = k_0$ angenommen werden sollte.
- Wird der Startwert k_0 abgelehnt, so erhöht man k_0 um 1 und wiederholt die Schätzung und den Test für $k_1 = k_0 + 1$.
- Wird auch dieser abgelehnt, macht man mit $k_2 = k_1 + 1$ weiter, bis man zu einem k_n kommt, das den Test besteht.

Da die Hypothesen $H(k_0), H(k_1), \ldots, H(k_n)$ nicht unabhängig sind, kann man nicht sagen, welche Wahrscheinlichkeit $k = k_n$ hat, man kann aber sagen, dass

$$k \geq k_n \text{ mit Wahrscheinlichkeit } \geq 1 - \alpha$$

für ein Signifikanzniveau α in den verwendeten Tests gilt. Der große Aufwand entsteht durch die wiederholte Ausführung der ML-Schätzung, die für jede Hypothese erneut durchgeführt werden muss.

4.2 Hauptkomponentenanalyse

Die Hauptkomponentenanalyse reduziert p untereinander korrelierte Variable auf neue Variable, die möglichst viel der Gesamtvarianz enthalten. Die beobachteten p Variablen werden dafür linear transformiert zu $k < p$ Hauptkomponenten, die untereinander unkorreliert sind und nach fallender Varianz geordnet werden.
Es handelt sich um ein Verfahren, das kein Modell als Grundlage benötigt und sich grundsätzlich mit der Varianz und nicht mit der Kovarianz oder Korrelation beschäftigt. Die Hauptkomponentenanalyse bietet aber auch die Grundlage für die in der Faktorenanalyse verwendete Hauptkomponentenmethode und die Hauptfaktorenanalyse.

Mathematisch gesehen wird für die Hauptkomponentenanalyse eine Hauptachsentransformation durchgeführt. Hierbei wird eine standardisierte Matrix $Z \in \mathbb{R}^{N,p}$ in die Form

$Z = FL'$ gebracht, wobei $F = (F_1, ..., F_p) \in \mathbb{R}^{N,p}$ mit orthonormierten Spalten und $L \in \mathbb{R}^{p,p}$. Die Spalten $F_1, ..., F_p$ werden Hauptachsen genannt und es gilt in diesem Fall wieder $R = Z'Z = LL'$.

Die Hauptkomponentenmethode ergibt sich, wenn erst mithilfe der Hauptachsentransformation die ersten $k < p$ Hauptachsen bestimmt und diese dann als Faktoren verwendet werden. Die Zerlegung hat dann die Form

$$Z = F^{(k)}L^{(k)'} + E \ ,$$

wobei $Z, E \in \mathbb{R}^{N,p}$, $F^{(k)} \in \mathbb{R}^{N,k}$ orthonormiert und $L^{(k)} \in \mathbb{R}^{p,k}$.

Nimmt man stattdessen die Gleichung

$$R = LL' + V \ ,$$

wobei wie bisher $V \in \mathbb{R}^{p,p}$ ist, als Grundlage und bestimmt zuerst V durch Schätzung, kann man die Hauptachsentransformation auf

$$R_h = R - V = LL'$$

anwenden. Diese Methode heißt Hauptfaktorenanalyse. Wir werden die Hauptachsentransformation einer näheren Betrachtung unterziehen, bevor wir uns wieder der Hauptkomponentenmethode und der Hauptfaktorenanalyse zuwenden.

4.2.1 Hauptachsentransformation

Allgemein ist das Ziel der Hauptachsentransformation eine lineare Transformation. Dabei werden die Vektoren $Z_1, ..., Z_p \in \mathbb{R}^N$ in orthogonale Hauptachsen $H_1, ..., H_p \in \mathbb{R}^N$ mit $H_i = (Z_1, , Z_p)t_i$ gewandelt, wobei $t_i \in \mathbb{R}^p$, $||t_i|| = 1$ normierte Gewichtsvektoren sind. Außerdem soll H_1 die maximale Varianz haben, von den zu H_1 orthogonalen Vektoren soll H_2 die größte Varianz haben usw. Die Hauptachsentransformation ist besonders gut geeignet, um mit einer geringen Anzahl von Faktoren möglichst große Teile der Varianz zu erklären.

Auskunft über die Gestalt von $H = (H_1, ..., H_p)$ liefert Satz 3.1 aus [7] S.663:

Theorem 4.2.1.1 *Sei $Z \in \mathbb{R}^{N,p}$ vom Rang p, dann lautet die Matrix $H = (H_1, ..., H_p)$*

der Hauptachsen von Z

$$H = ZT \ ,$$

wobei in der orthogonalen Matrix T *die normierten Eigenvektoren zu den geordneten Eigenwerten* $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p > 0$ *von* $R = Z'Z$ *stehen. Für* H *gilt*

$$H'H = diag(\lambda_1, ..., \lambda_p) = \Lambda \ ,$$

die Hauptachsen H_i *sind zueinander orthogonal und besitzen der Reihe nach die Quadratsummen (empirische Varianzen)* $\lambda_1, ..., \lambda_p$.
Die Hauptachsen sind so wie die Spalten von T *bis auf das Vorzeichen eindeutig, wenn sich die Eigenwerte von* R *paarweise unterscheiden. Für die Datenmatrix* Z *gilt*

$$Z = HT'$$

$$Z'Z = R = T\Lambda T' \ .$$

Eine Umskalierung liefert die Darstellung in orthogonalen und normierten Faktoren F_i, $F = (F_1, ..., F_p)$

$$Z = FL' \ mit \ F = H\Lambda^{-\frac{1}{2}}, \ L = T\Lambda^{\frac{1}{2}}, \ F'F = I \ ,$$

$$Z'Z = R = LL' \ .$$

Die normierten Hauptachsen $F_1, ..., F_p$ *heißen Hauptkomponenten. Die Zerlegung*

$$Z = F\Lambda^{\frac{1}{2}}T'$$

heißt Grundstruktur (basic structure, Horst 1965) oder Singulärwertzerlegung (Stoer/Burlisch 1978, S.19) von Z. *Auflösung nach* F *ergibt*

$$F = ZT\Lambda^{-\frac{1}{2}} = ZT\Lambda^{-1}T'T\Lambda^{\frac{1}{2}} = ZR^{-1}L \ .$$

Der zugehörige Beweis kann ebenfalls in [7] S.663f gefunden werden.

4.2.2 Hauptkomponentenmethode

Die Hauptkomponentenmethode liefert die Kleinstquadratlösung des k-Faktorenmodells, wenn die Anzahl der Faktoren k vorgegeben ist, nach

$$tr((Z - F^{(k)}L^{(k)\prime})'(Z - F^{(k)}L^{(k)\prime})) = tr(R - L^{(k)}L^{(k)\prime})$$
$$= \lambda_{k+1} + ... + \lambda_p = \min_{\substack{F \in \mathbb{R}^{N,k} \\ L \in \mathbb{R}^{p,k}}} tr((Z - FL')'(Z - FL')) \ ,$$

wobei F orthonormiert ist. Außerdem wird die Quadratsumme der Einträge der Residuenkovarianz

$$U^{(k)} = R - L^{(k)}L^{(k)\prime}$$

minimiert, d.h. es gilt

$$\sum_{i,j}(u_{ij}^{(k)})^2 = tr(U^{(k)\prime}U^{(k)}) \to min \ .$$

4.2.3 Hauptfaktorenanalyse

Bei der Hauptfaktorenanalyse wird eine Zerlegung der Form $R = LL' + V$ für die empirische Korrelationsmatrix R gesucht. Die Matrix der Einzelrestvarianzen $V = diag(v_1^2, ..., v_p^2)$ entspricht dabei der im Faktormodell und es gilt für die Kommunalitäten $h_i^2 = 1 - v_i^2$ beziehungsweise $K = I - V$.

Im ersten Schritt werden die Einzelrestvarianzen v_i^2 bestimmt, indem die Kommunalitäten geschätzt werden.

Danach wird die Hauptachsentransformation auf

$$R_h = R - V$$

durchgeführt. Dabei werden die Eigenwerte $\lambda_1 \geq ... \geq \lambda_p$ und die normierten Eigenvektoren $t_1, ..., t_p$ ermittelt und man setzt

$$L = T\Lambda^{\frac{1}{2}} \ ,$$

wobei $\Lambda = diag(\lambda_1, ..., \lambda_p)$ und $T = (t_1, ..., t_p)$ ist.

Es folgt, dass

$$LL' = T\Lambda T' = R_h$$

und

$$R = LL' + V \ .$$

Danach können die Faktoren aus

$$\begin{aligned} ZT &= F\Lambda^{\frac{1}{2}} \\ F = ZT\Lambda^{-\frac{1}{2}} &= ZR_h^{-1}L \end{aligned}$$

berechnet werden, wobei $F = (F_1, ..., F_p)$. Die Faktoren $F_1, ..., F_p$ heißen Hauptfaktoren.

Probleme ergeben sich bei negativen Eigenwerten von R_h oder wenn ein Eigenwert Null ist. Diese Eigenwerte müssen ausgesondert werden, da der normierte Eigenvektor nicht erzeugt werden kann. Das führt dazu, dass nur die ersten k Hauptfaktoren benutzt werden und die Residuenkovarianzmatrix als Summe von V und der übrigen Kovarianz ermittelt wird:

$$\begin{aligned} Z &= F^{(k)}L^{(k)\prime} + E^{(k)} \\ R = L^{(k)}L^{(k)\prime} &+ (l_{k+1}l'_{k+1} + ... + l_p l'_p) + V \\ U^{(k)} &= (l_{k+1}l'_{k+1} + ... + l_p l'_p) + V \ . \end{aligned}$$

Man kann die Schritte der Hauptfaktorenanalyse auch iterieren und jeweils $diag(U^{(k)})$ als V für den nächsten Iterationsschritt verwenden. Das führt zur iterierten Hauptachsentransformation, auf die später noch genauer eingegangen wird.

4.3 Minres Methode

Die „minimum residual“ Methode oder kurz Minres Methode genannt, führt die Faktorenanalyse durch, indem sie die Nichtdiagonalelemente der Korrelation- oder Kovarianzmatrix im Sinne der Kleinstquadratmethode schätzt. Die Abweichungen zwischen der Annäherung und den beobachteten Werten werden Residuen genannt (was der Methode den Namen gibt), weil wir dabei diese Fehler (bzw. ihre Quadrate) minimieren.

Der Ansatz bei dieser Methode ist

$$\sum_{\substack{i,j=1 \\ i<j}} \left(r_{ij} - \sum_{m=1}^{k} l_{im} l_{jm} \right)^2 \to \min_{L \in \mathbb{R}^{p,k}}$$

mit der Nebenbedingung

$$h_i^2 = \sum_{m=1}^{k} l_{im}^2 \leq 1 \quad \forall i \in \{1, ..., k\}$$

wobei sich das geschätzte V bei dieser Methode aus der Berechnung der Minres Ladungsmatrix L_{Min} ergibt als

$$V_{min} = diag(R - L_{Min}L'_{Min}) \ .$$

Die Ladungsmatrix L_{Min} ist bis auf orthogonale Rotation eindeutig.

Der folgende Satz entstammt [7] (Satz 6.2, S.698), wo auch der zugehörige Beweis gefunden werden kann, und zeigt den Zusammenhang zwischen Minres Methode und Hauptfaktorenanalyse.

Theorem 4.3.0.1 *Die ersten k Hauptkomponenten von $R - V_{Min}$ besitzen (bis auf orthogonale Transformation) die Ladung L_{Min}.*

Das bedeutet, dass die Minres Lösung eine spezielle Lösung der Hauptfaktorenanalyse ist.

Die Minres Methode und ihre Implementierung nach [1] wird später noch genauer betrachtet.

5 Heywoodfälle und stabile Punkte

In diesem Kapitel werden die sogenannten Heywoodfällen vorgestellt, eine Schwachstelle einiger Faktorenanalysemethoden. Nach der Definition von Heywoodfällen wird ein konkreter Fall vorgeführt, bei dem die iterierte Hauptachsentransformation im Umgang mit Heywoodfällen genauer analysiert wird. Wir behandeln dabei den einfachsten Fall, nämlich den Fall mit drei Variablen und einem Faktor. In diesem Fall kann die Analyse noch ganz ohne Computer durchgeführt werden und das Verhalten der iterierten Hauptachsentransformation lässt sich auf theoretischer Basis verfolgen.

5.1 Heywoodfälle

Bei der Kleinstquadratmethode werden die Kommunalitäten automatisch nebenbei mitgeschätzt. Dabei kann es passieren, dass die Kommunalitäten h_j^2 größer als 1 werden, was gleichzeitig bedeutet, dass die Einzelrestvarianzen v_j^2 negativ würden, da ja $h_j^2 = 1 - v_j^2$ gilt. Solche Fälle nennt man Heywoodfälle. Sie stellen ein nicht unerhebliches Problem dar, da man dann keine sinnvolle Lösung erhält. Es wird daher bei den kommenden Tests und Algorithmen auch darauf geachtet, ob sie für solche Fälle geeignet sind, oder ohne Ergebnis abgebrochen werden muss. Zuerst werden wir uns aber der Analyse des einfachsten Falls widmen, der Faktorenanalyse mit 3 Variablen und einem Faktor.

5.2 Existenz stabiler Punkte

Bei der Untersuchung des Falles achten wir darauf, welche kritischen Punkte es für die Quadratsumme gibt und wohin die iterierte Hauptachsentransformation läuft, also ob sie Grenzpunkte besitzt oder Elemente von L gegen Unendlich laufen.

Prinzipiell gilt, dass sich die Quadratsumme

$$S = \sum_{i \neq j} (r_{ij} - l_i l_j)^2$$

mit jedem Iterationsschritt verringert oder zumindest nicht vergrößert. Damit läuft das Verfahren entweder ins Unendliche oder S konvergiert und L hat zumindest eine konvergente Teilfolge. Für einen Grenzpunkt darf sich also S durch Iterationen nicht mehr verändern. Das bedeutet, dass die geschätzten Kommunalitäten l_i nach der Iteration die gleichen sein müssen wie davor. Es muss also gelten, dass L der Eigenvektor zu dem Eigenwert $\lambda = l_1^2 + l_2^2 + l_3^2$ ist. Für die erste Frage zu den kritischen Punkten, also vor allem zu lokalen Extrema, muss man S ableiten und Null setzen. Das sind die Ausgangspunkte

für die nächsten Überlegungen und wir werden zeigen, dass die Fragen enger verbunden sind, als es zunächst den Anschein hat.

Ausgeschrieben muss für einen Grenzpunkt also gelten, dass

$$\lambda L = (l_1^2 + l_2^2 + l_3^2) \begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \begin{pmatrix} l_1^2 & r_{12} & r_{13} \\ r_{12} & l_2^2 & r_{23} \\ r_{13} & r_{23} & l_3^2 \end{pmatrix} \begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = AL \ .$$

Multiplizieren wir aus, erhalten wir

$$\begin{pmatrix} l_1(l_1^2 + l_2^2 + l_3^2) \\ l_2(l_1^2 + l_2^2 + l_3^2) \\ l_3(l_1^2 + l_2^2 + l_3^2) \end{pmatrix} = \begin{pmatrix} l_1^3 + l_2 r_{12} + l_3 r_{13} \\ l_1 r_{12} + l_2^3 + l_3 r_{23} \\ l_1 r_{13} + l_2 r_{23} + l_3^3 \end{pmatrix} \ .$$

Wenn wir schließlich alles auf eine Seite bringen, erhalten wir drei Gleichungen:

$$\begin{aligned} I &: l_2(l_1 l_2 - r_{12}) + l_3(l_1 l_3 - r_{13}) = 0 \\ II &: l_1(l_1 l_2 - r_{12}) + l_3(l_2 l_3 - r_{23}) = 0 \\ III &: l_1(l_1 l_3 - r_{13}) + l_2(l_2 l_3 - r_{23}) = 0 \end{aligned}$$

Das sind dieselben Gleichungen, die wir erhalten, wenn wir S partiell ableiten und die Ableitungen gleich Null setzen. Im nächsten Schritt multiplizieren wir die $i-te$ Gleichung mit l_i:

$$\begin{aligned} &l_1 l_2(l_1 l_2 - r_{12}) + l_1 l_3(l_1 l_3 - r_{13}) = 0 \\ &l_2 l_1(l_1 l_2 - r_{12}) + l_2 l_3(l_2 l_3 - r_{23}) = 0 \\ &l_3 l_1(l_1 l_3 - r_{13}) + l_3 l_2(l_2 l_3 - r_{23}) = 0 \end{aligned} \tag{12}$$

und man sieht, dass die Gleichungen nur alle richtig sein können, wenn für je zwei verschiedene i, j gilt, dass $l_i l_j(l_i l_j - r_{ij}) = 0$. Damit das gilt, muss entweder $l_i = 0$, $l_j = 0$ oder $l_i l_j = r_{ij}$ gelten.

Wenn $l_1 l_2 = r_{12}$, $l_1 l_3 = r_{13}$, $l_2 l_3 = r_{23}$ gilt, ergibt zwar $S = 0$, ist aber nur für $P =$

$r_{12}r_{13}r_{23} > 0$ möglich, da wir uns in diesem Fall zum Beispiel l_1 ausdrücken können:

$$\begin{aligned} l_1 &= \frac{r_{12}}{l_2} \\ l_2 &= \frac{r_{23}}{l_3} \\ l_3 &= \frac{r_{13}}{l_1} \\ l_1 &= \frac{r_{12}}{\frac{r_{23}}{\frac{r_{13}}{l1}}} \\ l_1 &= \pm\sqrt{\frac{r_{12}r_{13}}{r_{23}}} \end{aligned}$$

Dies führt zu imaginären Ergebnissen für $P < 0$.

Dass $l_1 = l_2 = l_3 = 0$ gilt, ist uninteressant, da Eigenvektoren nicht 0 sein dürfen.

Wenn zwei Elemente von L Null sind und das dritte ungleich Null ist, beschreibt das den besten Fall für $P < 0$, da hier die geschätzte Matrix LL' nur aus 0 und einem Eintrag l_i^2 in der Diagonale besteht. Der Fehler würde aber durch falsche Vorzeichen, die bei der Multiplikation von L mit L' entstehen, vergrößert werden, wenn zwei oder alle $l_i \neq 0$ wären.

Der Fall, dass zwei Elemente von L gegen Null und das dritte gegen ∞ laufen, liefert für $P < 0$ das beste Ergebnis. Angenommen, r_{23} ist die Kovarianz mit dem kleinsten Betrag und $l_1 \neq 0$, sähe der Grenzfall so aus:

$$\begin{aligned} l_1 &\to \infty \\ l_2 &= \frac{r_{12}}{l_1} \\ l_3 &= \frac{r_{13}}{l_1} \end{aligned}$$

Für uns interessant bleibt der Fall, dass ein Element von L Null ist, die anderen beiden ungleich Null sind und $P < 0$ ist. Hier sind natürlich alle Vertauschungen möglich, wir werden im Folgenden aber o.B.d.A. annehmen, dass $l_1 = 0$ und $l_2, l_3 \neq 0$.

5.2.1 Fall $l_1 = 0$, $l_2, l_3 \neq 0$ und $P < 0$

Für die nächsten Überlegungen benötigen wir die Eigenschaften aus (2.2.35). Betrachten wir nämlich unsere Matrix A und bilden die Determinante, so muss sie mit dem Produkt der Eigenwerte übereinstimmen. Außerdem können wir bereits sagen, dass die Eigenwerte

λ_2 und λ_3 den gleichen Betrag und verschiedene Vorzeichen haben müssen, da

$$\begin{aligned} tr(A) &= \sum_{i=1}^{n} \lambda_i \\ l_1^2 + l_2^2 + l_3^2 &= \lambda_1 + \lambda_2 + \lambda_3 \\ l_2^2 + l_3^2 &= l_2^2 + l_3^2 + \lambda_2 + \lambda_3 \\ \lambda_3 &= -\lambda_2 \end{aligned} \tag{13}$$

gelten muss. Die Determinante von A, wobei $l_1 = 0$ gilt, ist

$$det(A) = \begin{vmatrix} 0 & r_{12} & r_{13} \\ r_{12} & l_2^2 & r_{23} \\ r_{13} & r_{23} & l_3^2 \end{vmatrix} = 2(r_{12}r_{13}r_{23}) - l_2^2 r_{13}^2 - l_3^2 r_{12}^2.$$

Wegen (13) und (12) muss also gelten, dass

$$(l_2^2 + l_3^2) \cdot \lambda_2 \cdot (-\lambda_2) = 2(r_{12}r_{13}r_{23}) - l_2^2 r_{13}^2 - l_3^2 r_{12}^2$$

5.2.2 Spezialfall $r_{12} = r_{13} = r_{23} = r$

Man betrachtet $r_{12} = r_{13} = r_{23} = r$. In diesem Fall ist klar, dass r negativ sein muss, da sonst $P = r^3$ nicht negativ sein kann. Es vereinfacht sich auch die obige Gleichung zu

$$(l_2^2 + l_3^2) \cdot \lambda_2 \cdot (-\lambda_2) = 2r^3 - l_2^2 r^2 - l_3^2 r^2$$

und ergibt nach kurzem Umformen

$$\lambda_{2,3} = \pm\sqrt{\frac{l_2^2 r^2 + l_3^2 r^2 - 2r^3}{l_2^2 + l_3^2}}.$$

Außerdem können wir aus (12) erkennen, dass $l_2 l_3 = r_{23}$ gelten muss, damit

$$l_2 l_3 (l_2 l_3 - r_{23}) = 0$$

zutrifft. Für unseren Spezialfall bedeutet das, dass $l_2 l_3 = r$ ist. Zudem soll L ein Eigenvektor zu λ_1 sein, das heißt es muss gelten

$$(A - \lambda_1 I) \cdot L = 0 \ ,$$

wobei 0 hier den Nullvektor bezeichnet. Mit A wie oben und $l_1 = 0$ ergibt das

$$\begin{pmatrix} l_2 r + l_3 r \\ l_3 r - l_2 l_3^2 \\ l_2 r - l_3 l_2^2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \ .$$

Formen wir die erste Zeile um:

$$\begin{aligned} l_2 r + l_3 r &= 0 \\ l_2 r &= -l_3 r \\ l_2 &= -l_3 \ . \end{aligned}$$

Zusammen mit $l_2 l_3 = r$ können wir daraus bereits unsere fehlenden l_i berechnen:

$$\begin{aligned} l_2 l_3 &= r \\ l_2 \cdot (-l_2) &= r \\ -l_2^2 &= r \\ l_2 &= \pm\sqrt{(-r)} \ . \end{aligned}$$

Wir wählen o.B.d.A., dass $l_2 = \sqrt{(-r)}$ und sehen, dass damit auch die zweite Gleichung erfüllt ist:

$$\begin{aligned} l_3 r - l_2 l_3^2 &= 0 \\ r(-\sqrt{-r}) - \sqrt{(-r)}(-\sqrt{(-r)})^2 &= 0 \\ (-r)^{\frac{3}{2}} - (-r)^{\frac{3}{2}} &= 0 \end{aligned}$$

und analog auch die dritte Gleichung überprüft werden kann:

$$\begin{aligned} l_2 r - l_3 l_2^2 &= 0 \\ r\sqrt{-r} - (-\sqrt{(-r)})(\sqrt{(-r)})^2 &= 0 \\ (-r)^{\frac{3}{2}} - (-r)^{\frac{3}{2}} &= 0 \ . \end{aligned}$$

Wir haben damit das L für diesen Spezialfall gefunden und können uns dem nächsten Schritt zuwenden. Wenn sowohl alle λ_i als auch L bekannt sind, muss nur noch gezeigt werden, dass $\lambda_1 = l_1^2 + l_2^2 + l_3^2$ wirklich der größte Eigenwert ist, um sicher zu gehen, dass er in der iterierten Hauptachsentransformation auch wirklich verwendet wird.

Dazu vergleichen wir λ_1 mit dem Positiven der beiden anderen λ_i (o.B.d.A λ_2), da λ_1

sicher immer größer als der Negative der beiden ist. Es soll also gelten:

$$l_2^2 + l_3^2 > \sqrt{\frac{l_2^2 r^2 + l_3^2 r^2 - 2r^3}{l_2^2 + l_3^2}} \ .$$

Setzen wir jetzt für l_2 und l_3 ein ($l_2^2 = l_3^2 = -r$) und formen um, erhalten wir:

$$\begin{aligned} l_2^2 + l_3^2 &> \sqrt{\frac{l_2^2 r^2 + l_3^2 r^2 - 2r^3}{l_2^2 + l_3^2}} \\ -2r &> \sqrt{\frac{-r^3 - r^3 - 2r^3}{-2r}} \\ -2r &> \sqrt{\frac{-4r^3}{-2r}} \\ -2r &> \sqrt{2r^2} \ . \end{aligned}$$

Das gilt sicher für alle $r \in [-1, 0)$, womit wir für den Spezialfall die Existenz eines stabilen Punkts gezeigt haben.

5.2.3 Allgemeiner Fall

Für die Darstellung des allgemeinen Falls geht man wieder von dem schon bekannten Zusammenhang zwischen den Eigenwerten aus:

$$(l_2^2 + l_3^2) \cdot \lambda_2 \cdot (-\lambda_2) = 2(r_{12} r_{13} r_{23}) - l_2^2 r_{13}^2 - l_3^2 r_{12}^2$$

Umgeformt erhalten wir diesmal:

$$\lambda_{2,3} = \pm \sqrt{\frac{l_2^2 r_{13}^2 + l_3^2 r_{12}^2 - 2(r_{12} r_{13} r_{23})}{l_2^2 + l_3^2}}$$

Betrachten wir wieder L, so muss wie beim Spezialfall gelten, dass

$$(A - \lambda_1 I) \cdot L = 0 \ ,$$

wobei 0 wieder den Nullvektor symbolisiert. Eingesetzt, ergibt das

$$\begin{pmatrix} l_2 r_{12} + l_3 r_{13} \\ l_3 r_{23} - l_2 l_3^2 \\ l_2 r_{23} - l_3 l_2^2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \ .$$

Benutzen wir zusätzlich wieder den Zusammenhang $l_2 l_3 = r_{23}$, erhalten wir

$$\begin{aligned} l_2 &= \frac{-r_{13} l_3}{r_{12}} \\ l_3 &= \pm\sqrt{\frac{r_{23} r_{12}}{-r_{13}}} \\ l_2 &= \pm\frac{-r_{13}\sqrt{\frac{r_{23} r_{12}}{-r_{13}}}}{r_{12}} \quad . \end{aligned}$$

Die so gefundenen l_2 und l_3 erfüllen natürlich wieder die anderen beiden Gleichungen und sind beide auf jeden Fall reell, da mit $P = r_{12} r_{13} r_{23} < 0$ auch $\frac{r_{23} r_{12}}{r_{13}} < 0$ ist und daher $\frac{r_{23} r_{12}}{-r_{13}} > 0$ ist und man somit immer reelle Ergebnisse für $l_{2,3}$ bekommt. Wir nehmen im Folgenden o.B.d.A. an, dass

$$l_3 = \sqrt{\frac{r_{23} r_{12}}{-r_{13}}} \; .$$

Auch diesmal muss noch gezeigt werden, dass λ_1 der größte Eigenwert sein kann, damit stabile Punkten existieren. Wieder betrachten wir nur den positiven anderen Eigenwert (o.B.d.A. $\lambda_2 > 0$), da λ_1 auf jeden Fall positiv ist. Also soll gelten:

$$\begin{aligned} \lambda_1 &> \lambda_2 \\ l_2^2 + l_3^2 &> \sqrt{\frac{l_2^2 r_{13}^2 + l_3^2 r_{12}^2 - 2(r_{12} r_{13} r_{23})}{l_2^2 + l_3^2}} \; , \end{aligned}$$

wobei

$$\begin{aligned} l_3^2 &= \frac{r_{23} r_{12}}{-r_{13}} \\ l_2^2 &= \frac{-r_{13} r_{23}}{r_{12}} \; . \end{aligned}$$

Setzen wir ein, erhalten wir

$$\frac{-r_{13} r_{23}}{r_{12}} + \frac{r_{23} r_{12}}{-r_{13}} > \sqrt{\frac{\frac{-r_{13} r_{23}}{r_{12}} r_{13}^2 + \frac{r_{23} r_{12}}{-r_{13}} r_{12}^2 - 2(r_{12} r_{13} r_{23})}{\frac{-r_{13} r_{23}}{r_{12}} + \frac{r_{23} r_{12}}{-r_{13}}}}$$

Zuerst betrachten wir den Term auf der rechten Seite:

$$\begin{aligned}
&\sqrt{\frac{\frac{-r_{13}r_{23}}{r_{12}}r_{13}^2+\frac{r_{23}r_{12}}{-r_{13}}r_{12}^2-2(r_{12}r_{13}r_{23})}{\frac{-r_{13}r_{23}}{r_{12}}+\frac{r_{23}r_{12}}{-r_{13}}}}\\
&=\sqrt{\frac{\frac{r_{13}r_{23}}{r_{12}}r_{13}^2+\frac{r_{23}r_{12}}{r_{13}}r_{12}^2+2(r_{12}r_{13}r_{23})}{\frac{r_{13}r_{23}}{r_{12}}+\frac{r_{23}r_{12}}{r_{13}}}}\\
&=\sqrt{\frac{\frac{r_{13}^2r_{23}}{r_{12}r_{13}}r_{13}^2+\frac{r_{23}r_{12}^2}{r_{13}r_{12}}r_{12}^2+\frac{2r_{12}^2r_{13}^2r_{23}}{r_{12}r_{13}}}{\frac{r_{13}^2r_{23}}{r_{12}r_{13}}+\frac{r_{23}r_{12}^2}{r_{13}r_{12}}}}\\
&=\sqrt{\frac{r_{13}^4r_{23}+r_{23}r_{12}^4+2r_{12}^2r_{13}^2r_{23}}{r_{13}^2r_{23}+r_{23}r_{12}^2}}\\
&=\sqrt{\frac{r_{23}(r_{13}^4+r_{12}^4+2r_{12}^2r_{13}^2)}{r_{23}(r_{13}^2+r_{12}^2)}}\\
&=\sqrt{\frac{r_{13}^4+r_{12}^4+2r_{12}^2r_{13}^2}{r_{13}^2+r_{12}^2}}\\
&=\sqrt{\frac{(r_{13}^2+r_{12}^2)^2}{r_{13}^2+r_{12}^2}}\\
&=\sqrt{r_{13}^2+r_{12}^2}
\end{aligned}$$

dann die linke Seite:

$$\begin{aligned}
&\frac{-r_{13}r_{23}}{r_{12}}+\frac{r_{23}r_{12}}{-r_{13}}\\
&=\frac{-r_{13}^2r_{23}}{r_{12}r_{13}}+\frac{r_{23}r_{12}^2}{-r_{13}r_{12}}\\
&=\frac{-r_{23}(r_{13}^2+r_{12}^2)}{r_{13}r_{12}}\\
&=\frac{-r_{23}}{r_{13}r_{12}}\cdot(r_{13}^2+r_{12}^2)
\end{aligned}$$

und insgesamt ergibt sich also

$$\frac{-r_{23}}{r_{13}r_{12}}\cdot(r_{13}^2+r_{12}^2)>\sqrt{r_{13}^2+r_{12}^2}\ .$$

Das heißt für alle r_{12}, r_{13}, r_{23}, für die gilt, dass

$$\frac{-r_{23}}{r_{13}r_{12}} > \frac{\sqrt{r_{13}^2 + r_{12}^2}}{r_{13}^2 + r_{12}^2} = \frac{1}{\sqrt{r_{13}^2 + r_{12}^2}} \ ,$$

haben wir wieder stabile Punkte gefunden.

5.2.4 Fazit

Wir konnten stabile Punkte identifizieren und damit zeigen, dass die Ladungen der iterierten Hauptachsentransformation (zumindest für diesen ausgewählten Fall) entweder gegen einen stabilen Punkt konvergieren oder zumindest ein Element von L gegen Unendlich läuft. Wo die Methode nicht zur Konvergenz gegen stabile Punkte führt, treten also zwangsweise Heywoodfälle auf.

Die allgemeine Frage nach der Konvergenz der iterierten Hauptachsentransformation ist jedoch nicht leicht zu beantworten. Um eine Vorstellung zu bekommen, ob das Verfahren konvergiert und vernünftige Lösungen liefert, werden wir es im nächsten Kapitel testen. Dabei werden mehrere Algorithmen verwendet, um auch gleich sehen zu können, wie sich die iterierte Hauptachsentransformation im Vergleich bezüglich Aufwand und Laufzeit darstellt.

6 Vermeidung von Heywoodfällen

In diesem Kapitel werden vier Algorithmen untersucht und diversen Tests unterzogen, um die Fragen aus dem letzten Kapitel zu beantworten. Dabei minimieren die Algorithmen die Quadratsumme der Abweichungen unter der Nebenbedingung, dass die geschätzten Kommunalitäten nicht größer als 1 sein dürfen. Nur der erste Algorithmus, die reine iterierte Hauptachsentransformation, arbeitet ohne die Nebenbedingung.

6.1 Die iterierte Hauptachsentransformation

6.1.1 Algorithmus 1

Ausgehend von einer Kovarianz oder Korrelationsmatrix (im Folgenden $\hat{\Sigma}$ genannt, nicht zu verwechseln mit der Notation aus der Maximum-Likelihood Methode), einer diagonalen Startwertmatrix D für das zu schätzende V und gewählter Faktorenanzahl k werden im ersten Iterationsschritt die k größten Eigenwerte und ihre dazugehörigen (rechten normierten) Eigenvektoren von $\hat{\Sigma} - D$ berechnet.
Aus Eigenvektoren und Eigenwerten wird dann die Ladungsmatrix L zusammengesetzt. Der neue Schätzwert von D ergibt sich dann als die Diagonale von $\hat{\Sigma} - LL'$.
Unterscheidet sich der neue Schätzwert weniger als eine vorgegebene Toleranz vom alten, haben wir Schätzwerte für L und V gefunden, sonst werden wieder Eigenwerte und Eigenvektoren von $\hat{\Sigma} - D$ berechnet, wobei D hier natürlich schon der neue Schätzwert ist.
Als Ablaufdiagramm:

1. $\hat{\Sigma}$ gegeben, wähle Startwerte für D, die Anzahl der Faktoren k und die Toleranzgrenze

2. Bestimme die k größten Eigenwerte $\lambda_1, \ldots, \lambda_k$ und die normierten rechten Eigenvektoren $x_1, \ldots, x_k$

3. Setze L zusammen, $L = (\sqrt{\lambda_1}\ x_1, \ldots, \sqrt{\lambda_k}\ x_k)$, das heißt die i-te Spalte von L entspricht $\sqrt{\lambda_i}\ x_i$

4. Neuer Wert für D ist $diag(\hat{\Sigma} - LL')$

5. Unterscheiden sich die Nicht-Diagonalelemente von L nur weniger als die Toleranzgrenze vom alten L, sind das neue L und D die Lösung. Sonst wiederholt man ab 2. und ersetzt das alte D durch das neue D.

Damit die Abbruchbedingung greift, muss die Prozedur natürlich zumindest zweimal durchlaufen werden.

Dieses Verfahren wurde in R als Funktion implementiert und wird im Folgenden als Algorithmus 1 bezeichnet. Der Code dazu findet sich im Anhang.

Es wurde außerdem ein zusätzlicher Schritt eingefügt, der nur der Benutzerfreundlichkeit dient: Vor der Erstellung von L prüft die Funktion, ob es überhaupt k positive Eigenwerte von $\hat{\Sigma} - D$ gibt.
Es ist zwar bekannt, dass alle Eigenwerte reell sein müssen, da es sich um eine symmetrische Matrix handelt, allerdings ist nicht sicher, dass sie auch alle positiv sind. Sollten zu viele Faktoren gefordert worden sein, bricht die Funktion ab und benachrichtigt den Benutzer.
Diese Vorgehensweise verhindert Endlosschleifen, da R trotzdem versucht, die Wurzeln zu ziehen und dann mit NaN (Not a Number) weiterrechnet, was natürlich nie zu einem Ergebnis führt.

Ein weiteres mögliches Problem wäre, dass R durch Rundungsfehler doch komplexe Eigenwerte berechnet, was die Funktion ebenfalls in eine Endlosschleife verwandeln würde. Dieser Fehler ist beim Testen allerdings nie aufgetreten und der R-interne Befehl *eigen*, der die Berechnung der Eigenwerte und -vektoren ausführt, scheint ausgezeichnet zu funktionieren, weshalb dieser mögliche Fehler nicht präemptiv behoben wurde.

Zum Abschluss wurde eine zusätzliche Abbruchbedingung angefügt, die die maximale Anzahl der Iterationen auf einen eingegebenen Wert fixiert. Diese Bedingung resultierte aus der Überlegung, dass mögliche Endlosschleifen verhindert werden, die manuell gestoppt werden müssten und somit die Einbindung des Algorithmus in andere Funktionen behindern würden.

6.1.2 Algorithmus 2

Das Problem mit Algorithmus 1 besteht darin, dass er keinerlei Schutz gegen Heywoodfälle bietet. Tritt ein Heywoodfall ein, bricht der Algorithmus zwar nicht ab, er liefert aber kein sinnvolles Ergebnis. Damit die Rechenzeit nicht verschwendet wird, sollte in einem solchen Fall zumindest ein näherungsweise korrektes Ergebnis erlangt werden.

Um dieses Problem in der iterierten Hauptachsentransformation zu beheben, führen wir hier einen ganz neuen Algorithmus für die iterierte Hauptachsentransformation ein. Konvergiert die Methode, so liefert sie die Lösung der Gleichungen für die Minimierung mit Nebenbedingung und man kann die Lagrangemultiplikatoren direkt ablesen. Da das ein

ganz neuer Algorithmus ist, wird uns im Folgenden ebendiese Konvergenz interessieren und ob die Ergebnisse immer sinnvoll sind. Außerdem beobachten wir, wie der Algorithmus im Vergleich mit anderen Methoden, die das Gleiche leisten, abschneidet, was Genauigkeit und Aufwand beziehungsweise Laufzeit betrifft.

Für unseren neuen Algorithmus muss man Algorithmus 1 modifizieren:
Man gibt dem Algorithmus zu Beginn einen zusätzlichen Vektor E, dessen Einträge anfangs alle gleich 0 sind.
Der Algorithmus stimmt zu Anfang noch mit der unkorrigierten iterierten Hauptachsenmethode aus Algorithmus 1 überein bis er L berechnet hat. Statt des bekannten 4. Schrittes verwenden wir aber:

4. $E_i = max(E_i + (LL' - \hat{\Sigma})_{ii}, 0)$
   $D_{ii} = (\hat{\Sigma} - LL')_{ii} + E_i$

Dadurch wird verhindert, dass D negative Komponenten bekommt und die Einträge von E_i entsprechen genau den Lagrangemultiplikatoren.

Auch dieses Verfahren wurde in R implementiert und wird im Folgenden Algorithmus 2 genannt. Der Code dazu findet sich im Anhang. Wie bei Algorithmus 1 wurde auch hier eine zusätzliche Abbruchbedingung für die maximale Iterationsanzahl eingefügt.

6.2 Das psych package und Algorithmus 3

Natürlich wurden im Laufe der Zeit bereits einige Varianten der Minres Methode implementiert und sogar in Code-Paketen zusammengefasst. Wir werden uns mit einem solchen Paket befassen und eine der darin enthaltenen Funktionen als dritten Algorithmus in unsere Testreihe aufnehmen.

6.2.1 R und das psych package

Wir werden unsere Tests in R durchführen und auch die Algorithmen 1 und 2 wurden, wie schon erwähnt, in R implementiert. R ist eine freie Software zur Erstellung statistischer Analysen und Plots. Ein besonders praktisches Feature von R ist der Aufbau in Paketform, was das Herunterladen und Benutzen von zusätzlichen, nicht im Installationsumfang der Software enthaltenen Paketen, sehr einfach macht. Diese Pakete (englisch: packages) sind oft speziell auf gewisse Forschungsfelder zugeschnitten und enthalten Funktionen, die dort häufig zur Anwendung kommen.

Für uns besonders interessant ist das psych Paket [5], das viele Funktionen beinhaltet, die in der Psychologie benötigt werden. Darunter befindet sich auch die Faktorenanalyse, die im Paket gleich mehrfach vorkommt. Das Paket hat ein eigenes Handbuch, das die enthaltenen Funktionen erklärt: [6].

6.2.2 Algorithmus 3

Für unsere Zwecke geeignet ist die fa-Funktion, die diverse Methoden der Faktorenanalyse durchführt, darunter auch 3 Varianten der Minres Methode. Wir werden die einfachste Minres Variante verwenden (fm="minres"). Laut Beschreibung wird dabei eine ungewichtete Kleinstquadratlösung bestimmt, die den R-Befehl optim() verwendet.

Eine detaillierte Erklärung zu diesem Befehl kann bei [9] gefunden werden. Vereinfacht ausgedrückt ist es ein Befehl zur multivariaten Optimierung, bei dem verschiedene Optimierungsmöglichkeiten ausgewählt werden können. Für die Funktion wurde das Quasi-Newton-Verfahren mit Schranken (L-BFGS-B) verwendet. Ein Link zum Code der fa-Funktion ist im Anhang beziehungsweise im Quellenverzeichnis enthalten.

Anzumerken ist für diesen Algorithmus, dass man zwar die maximale Anzahl der Iterationen und die Toleranzgrenze für Fehler eingeben kann, diese aber für die Minres Variante nicht zur Anwendung kommen.

6.3 Minres Methode nach Harman

Der letzte Algorithmus, den wir für unsere Tests heranziehen werden, entspringt [1]. Er schlägt einen anderen Weg als die iterierte Hauptachsentransformation vor und behauptet, dass die so erhaltene Variante weniger Rechenzeit benötigt und Heywoodfälle in Betracht zieht. Bevor wir uns aber dem eigentlichen Algorithmus widmen, wird noch die dazugehörige Prozedur dargestellt, um alle notwendigen Terme kennenzulernen.

6.3.1 Vorbereitung auf den Algorithmus

Ausgehend von der Idee, dass Veränderungen an einer Spalte von L nur linearen Einfluss auf die reproduzierten Korrelationen haben und die Zielfunktion als quadratische Funktion von diesen Veränderungen geschrieben werden kann, führen wir Inkremente ϵ_j ein, die wir zu unseren Ladungen addieren:

$$l_{j1} + \epsilon_1,\ l_{j2} + \epsilon_2,\ \ldots,\ l_{jk} + \epsilon_k$$

und schreiben der Einfachheit halber $w_{ji} = l_{ji} + \epsilon_i$. Das bedeutet aber nicht, dass die w_{ji} unsere neuen Faktorladungen sind, es sollen nur die nächsten Formeln verkürzt werden.

Damit schreiben wir für die reproduzierten Korrelationen jetzt

$$\hat{r}_{ji} = \sum_{m=1}^{k} l_{im} w_{jm} \qquad (j \text{ fix})$$

und (nicht zu verwechseln mit unseren Faktoren f aus dem Theorieteil)

$$f_j = \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(r_{jm} - \sum_{n=1}^{k} l_{mn} w_{jn} \right)^2 \qquad (j \text{ fix})$$

für die summierten Quadrate der Korrelationsresiduen von Variable j. Die Gleichung kann umgeschrieben werden zu

$$f_j = \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(r^*_{jm} - \sum_{n=1}^{k} l_{mn} \epsilon_n \right)^2 \ , \qquad (j \text{ fix})$$

wobei

$$r^*_{jm} = r_{jm} - \sum_{n=1}^{k} l_{mn} l_{jn} \ . \qquad (m = 1, 2, \ldots, p;\ j \neq m)$$

Jetzt betrachtet man die partiellen Ableitungen von f_j an, um die besten Werte für ϵ_j zu finden:

$$\frac{\partial f_j}{\partial \epsilon_q} = 2 \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(r^*_{jm} - \sum_{n=1}^{k} l_{mn} \epsilon_n \right) (-l_{mq}) \ . \qquad (q = 1, 2, \ldots, k)$$

Setzen wir das gleich 0 und formen etwas um, erhalten wir für die ϵ_q:

$$\sum_{n=1}^{k} \left(\sum_{\substack{m=1 \\ m \neq j}}^{p} l_{mn} l_{mq} \right) \epsilon_n = \sum_{\substack{m=1 \\ m \neq j}}^{p} r^*_{jm} l_{mq} \ , \qquad (q = 1, 2, \ldots, k)$$

was wir wieder in Matrixschreibweise bringen können als

$$\boldsymbol{\epsilon}_j L'_{)j(} L_{)j(} = r^0_j L \ ,$$

wobei $\boldsymbol{\epsilon}_j = (\epsilon_1, \epsilon_2, \ldots, \epsilon_k)$ der Vektor mit den Veränderungen an der Variable j ist, $L_{)j(}$ ist die Matrix L, bei der die Einträge der j-ten Zeile durch 0 ersetzt wurden und r_j^0 ist der Vektor der Korrelationsresiduen der Variable j und 0 an j-ter Stelle (also ist das Korrelationsresiduum von j mit j gleich 0).
Bringen wir alles außer $\boldsymbol{\epsilon}_j$ auf die andere Seite, erhalten wir die Formel für die idealen ϵ_q:

$$\boldsymbol{\epsilon}_j = r_j^0 L (L'_{)j(} L_{)j(})^{-1} \ .$$

6.3.2 Umgehen von Heywoodfällen

Bisher wurden noch keinerlei Vorkehrungen getroffen, was Heywoodfälle betrifft. Das soll sich jetzt ändern, indem eine zusätzliche Bedingung eingefügt wird:

$$\sum_{i=1}^{k} w_{ji}^2 \leq 1 \ . \tag{14}$$

Das entspricht der bereits bekannten Bedingung, dass die Kommunalitäten kleiner als 1 sein sollen. Das kann prinzipiell zu zwei Fällen führen:

1. Die Lösung wird bei $(b_{n1}, b_{n2}, \ldots, b_{nm})$ angenommen und liegt damit in dem von (14) definierten Gebiet.

2. Die Lösung liegt nicht in dem definierten Gebiet.

Im ersten Fall haben wir keine weiteren Probleme und brauchen unser Verfahren nicht weiter zu verändern. Die Lösung ist gefunden und widerspricht keiner unserer Annahmen. Im zweiten Fall jedoch müssen wir eingreifen, um eine sinnhafte Lösung zu erhalten. Dabei hilft uns der folgende Satz aus [1] S. 179 (mit abgeänderter Notation) :

Theorem 6.3.2.1 *Wird das Minimum von f_j außerhalb des von (14) definierten Gebiets angenommen, dann wird ein Minimum von f_j unter der Bedingung von (14) an einem Randpunkt des Gebiets angenommen, so dass (14) ersetzt werden kann durch*

$$\sum_{i=1}^{k} w_{ji}^2 = 1 \ . \tag{15}$$

Im Beweis aus [14] dazu wird f_j umgeformt zu

$$f_j = \sum_{i=1}^{k} (\chi_i - \xi_i)^2 + \hat{K} \tag{16}$$

mit der Bedingung

$$\sum_{i=1}^{k} \frac{\chi_i^2}{\lambda_i^2} \leq 1 \ . \tag{17}$$

Dabei sind die χ_i Funktionen in Abhängigkeit von b_{ji}, die λ_i sind Eigenwerte einer $(k \times k)$-Matrix, die ξ_i sowie K sind Konstanten. Außer den χ_i können alle anderen Werte aus bereits Bekanntem beziehungsweise durch Transformationen davon berechnet werden. Allerdings wird in [1] und auch in dem zugrunde liegenden Artikel [14] die Umformung nicht vollständig beschrieben. Da wir sie für den Algorithmus aber benötigen, werden wir die Umformung hier ausführen.

Ausgehend von

$$f_j = \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(r_{jm} - \sum_{n=1}^{k} l_{mn} w_{jn} \right)^2 \qquad (j \text{ fix}) \tag{}$$

soll zuerst die Form

$$f_j = w' G w + 2 \sum_{n=1}^{k} v_n w_{jn} + K \tag{18}$$

erreicht werden, wobei w ein Spaltenvektor mit Einträgen w_{jn} ist, G eine $(k \times k)$-Matrix und v_n sowie K Konstante. Um die neuen Variablen zu finden, betrachten wir

$$\begin{aligned} f_j &= \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(r_{jm} - \sum_{n=1}^{k} l_{mn} w_{jn} \right)^2 \\ &= \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(r_{jm}^2 - 2 \sum_{n=1}^{k} l_{mn} w_{jn} r_{jm} + \left(\sum_{n=1}^{k} l_{mn} w_{jn} \right)^2 \right) \\ &= \sum_{\substack{m=1 \\ m \neq j}}^{p} r_{jm}^2 + 2 \sum_{n=1}^{k} \sum_{\substack{m=1 \\ m \neq j}}^{p} -(l_{mn} w_{jn} r_{jm}) + \sum_{\substack{m=1 \\ m \neq j}}^{p} \left(\sum_{n=1}^{k} l_{mn} w_{jn} \right)^2 \ . \end{aligned}$$

Durch Koeffizientenvergleich erkennen wir, dass

$$K = \sum_{\substack{m=1 \\ m\neq j}}^{p} r_{jm}^2$$
$$v_n = \sum_{\substack{m=1 \\ m\neq j}}^{p} -(l_{mn} r_{jm})$$

sein müssen und wir können zusätzlich den Term

$$\sum_{\substack{m=1 \\ m\neq j}}^{p} \left(\sum_{n=1}^{k} l_{mn} w_{jn} \right)^2$$

anschreiben als

$$\begin{array}{llll}
\sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m1}^2 w_{j1}^2 & + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m1} w_{j1} l_{m2} w_{j2} & + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m1} w_{j1} l_{m3} w_{j3} & + \cdots + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m1} w_{j1} l_{mk} w_{jk} \quad + \\
\sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m2} w_{j2} l_{m1} w_{j1} & + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m2}^2 w_{j2}^2 & + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m2} w_{j2} l_{m3} w_{j3} & + \cdots + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{m2} w_{j2} l_{mk} w_{jk} \quad + \\
\sum_{\substack{m=1 \\ m\neq j}}^{p} l_{m3} w_{j3} l_{m1} w_{j1} & + \sum_{\substack{m=1 \\ m\neq j}}^{p} l_{m3} w_{j3} l_{m2} w_{j2} & + \sum_{\substack{m=1 \\ m\neq j}}^{p} l_{m3}^2 w_{j3}^2 & + \cdots + \sum_{\substack{m=1 \\ m\neq j}}^{p} l_{m3} w_{j3} l_{mk} w_{jk} \quad + \\
\vdots & & & \\
\sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{mk} w_{jk} l_{m1} w_{j1} & + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{mk} w_{jk} l_{m2} w_{j2} & + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{mk} w_{jk} l_{m3} w_{j3} & + \cdots + \sum\limits_{\substack{m=1 \\ m\neq j}}^{p} l_{mk}^2 w_{jk}^2 \quad .
\end{array}$$

Aus dieser Form können wir jetzt die Einträge unserer Matrix $G = (g_{ia})$ ablesen:

$$g_{ia} = \sum_{\substack{m=1 \\ m\neq j}}^{p} l_{mi} l_{ma}$$

und mit der Umformung fortfahren.

Da $w'Gw$ das Quadrat einer Linearkombination der w_{jn} ist und G symmetrisch ist, ist G positiv definit. Daher können wir eine orthogonale Matrix Q finden, sodass gilt:

$$Q'GQ = \Lambda \ ,$$

wobei Λ eine Diagonalmatrix ist, deren Diagonalelemente λ_n^2 den Eigenwerten von G entsprechen und P den orthonormierten Eigenvektoren von G entspricht, wie in (2.2.34) bereits beschrieben. Jetzt können wir noch w durch $Q\hat{w}$ ersetzen und damit $w'Gw$ dia-

gonalisieren:

$$\begin{aligned} w'Gw &= \\ \hat{w}'Q'GQ\hat{w} &= \\ \hat{w}'\Lambda\hat{w} &\ . \end{aligned}$$

Damit können wir auch (18) umschreiben zu

$$f_j = \sum_{n=1}^{k} \lambda_n^2 \hat{w}_{jn}^2 + \sum_{n=1}^{k} 2\hat{\upsilon}_n \hat{w}_{jn} + K \ , \tag{19}$$

wobei $\hat{\upsilon} = Q'\upsilon$, wenn wir die υ_n als Vektor auffassen. Durch die orthogonale Transformation von w müssen wir auch unsere Bedingung geringfügig abändern:

$$\sum_{i=1}^{k} \hat{w}_{ji}^2 \leq 1 \ . \tag{20}$$

Zusätzlich kann (19) noch weiter vereinfacht werden, indem auf ein ganzes Quadrat ergänzt wird:

$$f_j = \sum_{n=1}^{k} (\lambda_n \hat{w}_{jn} - \lambda_n u_n)^2 + \hat{K} \ ,$$

wobei $u_n = \dfrac{-\hat{\upsilon}_n}{\lambda_n^2}$ und zum Ausgleich $\hat{K} = K - \sum\limits_{n=1}^{k} \dfrac{\hat{\upsilon}_n^2}{\lambda_n^2}$ ist.

Setzen wir jetzt noch $\chi_n = \lambda_n \hat{w}_{jn}$ und $\xi_n = \lambda_n u_n$ erhalten wir die angestrebte Darstellung (16) :

$$f_j = \sum_{n=1}^{k} (\chi_n - \xi_n)^2 + \hat{K}$$

und die Bedingung (20) nimmt ebenfalls die bereits genannte Form an:

$$\sum_{i=1}^{k} \frac{\chi_i^2}{\lambda_i^2} \leq 1$$

Diese Form für f_j ist vorteilhaft, weil sie uns eine neue Interpretation bietet. Wir sehen, dass f_j nur der Summe über eine Konstante $\hat{K}$ sowie dem quadrierten Abstand zweier Punkte unter der Bedingung (17) entspricht, wobei einer der Punkte, $(\xi_1, \xi_2, \ldots, \xi_k)$, fest

ist. Wir suchen also nach dem Punkt $(\chi_1, \chi_2, \ldots, \chi_k)$, der den Abstand minimiert und (17) genügt. Wieder können 2 Fälle auftreten:

1. Der fixe Punkt $(\xi_1, \xi_2, \ldots, \xi_k)$ liegt selbst in dem Gebiet, also $\sum_{i=1}^{k} \frac{\xi_i}{\lambda_i} \leq 1$.
2. Der fixe Punkt $(\xi_1, \xi_2, \ldots, \xi_k)$ liegt außerhalb des Gebiets.

Im ersten Fall haben wir unsere Lösung bereits mit $\xi_i = \chi_i$ gefunden. Im zweiten Fall muss der gesuchte Punkt $(\chi_1, \chi_2, \ldots, \chi_k)$ sicher am Rand des Gebiets liegen, um die Distanz zu minimieren, das heißt die Ungleichung in (17) können wir wieder in eine Gleichung umwandeln.

Unser eigentliches Problem ist also jetzt (unter Missachtung aller Konstanten) die Minimierung von

$$\sum_{i=1}^{k} (\chi_i - \xi_i)^2 = (\chi_1 - \xi_1)^2 + (\chi_2 - \xi_2)^2 + \cdots + (\chi_k - \xi_k)^2 \tag{21}$$

unter der Bedingung

$$\sum_{i=1}^{k} \frac{\chi_i^2}{\lambda_i^2} = 1 \ . \tag{22}$$

Wir wollen jetzt mittels Lagrange Multiplikatoren das Optimierungsproblem lösen. Dafür schreiben wir zuerst die Bedingung (22) um in die Form

$$\sum_{i=1}^{k} \frac{\chi_i^2}{\lambda_i^2} - 1 = 0 \ .$$

Unsere Hilfsfunktion lässt sich dann mit μ als Lagrange Multiplikator schreiben als

$$H = (\chi_1 - \xi_1)^2 + (\chi_2 - \xi_2)^2 + \cdots + (\chi_k - \xi_k)^2 + \mu \left(\sum_{i=1}^{k} \frac{\chi_i^2}{\lambda_i^2} - 1 \right)$$

und wir können unsere Hilfsfunktion H partiell nach den Variablen $\chi_1, \chi_2, \ldots, \chi_k$ und μ

ableiten und gleich 0 setzen. Das führt uns zu

$$\begin{aligned}
\frac{\partial H}{\partial \chi_1} &= \chi_1 - \xi_1 + \mu \frac{\chi_1}{\lambda_1^2} &= 0 \\
\frac{\partial H}{\partial \chi_2} &= \chi_2 - \xi_2 + \mu \frac{\chi_2}{\lambda_2^2} &= 0 \\
&\vdots \\
\frac{\partial H}{\partial \chi_k} &= \chi_k - \xi_k + \mu \frac{\chi_k}{\lambda_k^2} &= 0 \\
\frac{\partial H}{\partial \mu} &= \frac{\chi_1^2}{\lambda_1^2} + \frac{\chi_2^2}{\lambda_2^2} + \cdots + \frac{\chi_k^2}{\lambda_k^2} - 1 &= 0 \ ,
\end{aligned}$$

wobei wir die beim Ableiten entstandenen Konstanten bereits auf die andere Seite gebracht haben. Wir können μ jetzt sofort durch jeden der anderen Parameter ausdrücken als

$$\mu = \frac{(\xi_i - \chi_i)\lambda_i^2}{\chi_i} \ . \qquad (i = 1, 2, \ldots, k)$$

Wir wählen $i = 1$ und sehen

$$\begin{aligned}
\frac{\partial H}{\partial \chi_2} &= \frac{\partial H}{\partial \chi_1} \\
\chi_2 - \xi_2 + \mu \frac{\chi_2}{\lambda_2^2} &= \chi_1 - \xi_1 + \mu \frac{\chi_1}{\lambda_1^2} \\
\chi_2 - \xi_2 + \left(\frac{(\xi_1 - \chi_1)\lambda_1^2}{\chi_1} \right) \frac{\chi_2}{\lambda_2^2} &= \chi_1 - \xi_1 + \left(\frac{(\xi_1 - \chi_1)\lambda_1^2}{\chi_1} \right) \frac{\chi_1}{\lambda_1^2} \\
\chi_2 - \xi_2 + \left(\frac{(\xi_1 - \chi_1)\lambda_1^2}{\chi_1} \right) \frac{\chi_2}{\lambda_2^2} &= 0 \\
\left(\frac{(\xi_1 - \chi_1)\lambda_1^2}{\chi_1} \right) \frac{\chi_2}{\lambda_2^2} &= \xi_2 - \chi_2 \\
\frac{(\xi_1 - \chi_1)\lambda_1^2}{\chi_1} &= \frac{(\xi_2 - \chi_2)\lambda_2^2}{\chi_2} \\
\lambda_1^2 \left(\frac{\xi_1}{\chi_1} - 1 \right) &= \lambda_2^2 \left(\frac{\xi_2}{\chi_2} - 1 \right) \ .
\end{aligned}$$

Damit können wir uns jetzt χ_2 in Abhängigkeit von χ_1 ausdrücken:

$$\chi_2 = \frac{\chi_1 \xi_2 \lambda_2^2}{(\lambda_2^2 - \lambda_1^2)\chi_1 + \lambda_1^2 \xi_1}$$

und die Prozedur natürlich für alle χ_i für $i > 2$ analog wiederholen, weshalb folgt, dass

$$\chi_i = \frac{\chi_1 \xi_i \lambda_i^2}{(\lambda_i^2 - \lambda_1^2)\chi_1 + \lambda_1^2 \xi_1} \quad . \qquad (i = 2, 3, \ldots, k) \quad (23)$$

Setzen wir diese χ_i und χ_1 jetzt in unsere Bedingung (22) ein, erhalten wir ein Polynom vom Grad $2k$, dessen Nullstelle unsere Lösung ist.

6.3.3 Algorithmus 4

Der Beschreibung in [1] (Seite 186-88) folgend, wollen wir den fertigen Algorithmus betrachten.

1. Der erste Schätzwert für L wird aus den Eigenwerten und Eigenvektoren von R bestimmt.
2. Eine Schleife berechnet jeweils die optimalen ϵ_j und erstellt die neue Version von L.
3. Es wird überprüft, ob die Kommunalität der neuen Zeile kleiner 1 ist.
4. Ist sie das nicht, beginnt hier die Prozedur zur Vermeidung von Heywoodfällen.
5. Ist die Schleife ausgelaufen wird überprüft, ob die Toleranzgrenze der Veränderungen in L bereits unterschritten wurde oder ob die Anzahl der maximalen Iterationen erreicht wurde.
6. Falls ja, wird hier abgebrochen und die Ergebnisse werden ausgegeben. Falls nein, beginnt die Schleife erneut.

Abschließend ist noch zu sagen, dass auf Grund der unzureichenden Beschreibung in [1] der letzte Teil der Heywood Prozedur neu erstellt wurde. Wie wir gesehen haben, führt die Prozedur auf ein Polynom, dessen exakte Lösung viel Aufwand erfordern kann. Harman schlägt daher als Alternative eine Annäherung vor und liefert auch einen Startwert:

$$\chi_1^0 = \frac{\xi_1}{\sqrt{\sum_{i=1}^{k} \left(\frac{\xi_i}{\lambda_i}\right)^2}}$$

Allerdings ist die nachfolgende Näherung an die exakte Lösung nur sehr kryptisch erklärt, soll aber folgendes Theorem ([1] S. 181f) einbinden:

Theorem 6.3.3.1 *Für ein χ_1 zwischen 0 und $min(\xi_1, \lambda_1)$, mit χ_i $(i = 2, 3, \ldots, k-1)$, die entsprechend (23) berechnet wurden und x_k, das anschließend aus der Bedingung (22) berechnet wurde, gilt, dass wenn*

$$\lambda_k^2 \left(1 - \frac{\xi_k}{\chi_k}\right) \gtrless \lambda_1^2 \left(1 - \frac{\xi_1}{\chi_1}\right)$$

auch

$$\chi_1 \lessgtr \chi_1^*$$

gilt, wobei χ_1^* die exakte Lösung für χ_1 bezeichnet.

Um die exakte Lösung anzunähern, wurde zuerst der oben genannte Startwert verwendet, anschließend, dem Theorem folgend, alle anderen χ_i berechnet und schließlich die Doppelungleichung aus dem Theorem zu einer Gleichung umfunktioniert, um ein neues χ_1 daraus zu berechnen:

$$\chi_1^{\text{neu}} = \frac{\xi_1}{1 - \frac{\lambda_k^2}{\lambda_1^2}\left(1 - \frac{\xi_k}{\chi_k}\right)} \,. \tag{24}$$

Vor der Berechnung eines neuen χ_1 wird dabei geprüft, ob die Differenz von

$$\lambda_k^2 \left(1 - \frac{\xi_k}{\chi_k}\right)$$

zu

$$\lambda_1^2 \left(1 - \frac{\xi_1}{\chi_1}\right)$$

bereits kleiner als 10^{-8} ist. Ist das der Fall, werden die χ_i rücktransformiert und sind unsere Lösung. Falls nicht, wird ein neues χ_1 berechnet und der Vorgang wiederholt sich.

Der Algorithmus wurde in R implementiert und der Code ist im Anhang unter Algorithmus 4 zu finden.

6.4 Testinformationen

Im nächsten Teil werden wir die Algorithmen testen, um zu sehen, welcher die beste Laufzeit hat und wie widerstandsfähig sie wirklich gegenüber Heywoodfällen sind. Die Tests wurden in R auf einem Lenovo ideabook Y700 mit Intel(R) Core(TM) i5-6300HQ CPU @ 2.30GHz und 8,00 GB RAM durchgeführt.
Der Code zu den Laufzeittests kann im Anhang gefunden werden.

6.5 Laufzeitmessungen

Für unsere Laufzeitmessungen werden wir mit einem weiteren R-Paket arbeiten, nämlich „microbenchmark“. Das Paket ist darauf ausgelegt, so genau wie möglich zu messen, wie lange die eingegebenen Codesegmente in der Ausführung benötigen.
Die Funktion microbenchmark greift dafür direkt auf die rechnerinternen Zeitabfragen zu, um auf Untermillisekundenniveau die Laufzeit zu bestimmen. Praktischerweise kann auch gleich die Anzahl der Durchläufe eingegeben werden und man erhält als Ausgabe nicht nur die Laufzeit, sondern minimale sowie maximale Laufzeit, Durchschnitt und sogar Quantile der Laufzeiten präsentiert.

Da die Sinnhaftigkeit unserer Korrelations/Kovarianzmatrizen für die reine Laufzeitmessung unerheblich ist, werden wir einfach zu erstellende Matrizen verwenden. Wir wollen für Algorithmus 1 und 2 allerdings auch Matrizen, die genug positive Eigenwerte haben. Wie wir uns erinnern, brechen beide sonst mit der Meldung, dass k kleiner gewählt werden soll, ab. Für unsere Tests wollen wir also idealerweise positiv definite symmetrische Matrizen, deren Einträge alle kleiner oder gleich 1 sind. Daher werden zufällige, positiv definite symmetrische Matrizen erstellt und alle Einträge durch das Maximum dividiert.

Das führt allerdings auch zu recht unansehnlichen Testmatrizen und beim Export aus R (mittles xtable-Befehls) werden die Zahlen außerdem auf zwei Nachkommastellen gerundet. Deshalb und aus Gründen der Überschaubarkeit werden wir nur für Fälle mit wenigen Variablen eine exemplarische Testmatrix anführen. Um Testmatrizen für 50, 100 oder mehr Variablen in leserlicher Schriftgröße darzustellen, reicht hier schlicht der Platz nicht aus und es ist auch nicht zielführend.
Der Code für die Erstellung von Testmatrizen kann im Anhang eingesehen werden.

Außerdem wird darauf hingewiesen, dass man natürlich nicht auf Grund eines einzigen Falles auf die Schnelligkeit und Qualität eines Algorithmus schließen sollte. Die hier

aufgeführten Fälle sollten daher eher als Beispiele angesehen werden und dienen der Veranschaulichung. Alle Tests wurden außerdem mehrmals durchgeführt, um so ein besseres Gefühl für das Verhalten der Algorithmen zu bekommen. Für die Tests wurde die Anzahl der maximalen Iterationen auf 1000 und die Fehlertoleranzgrenze auf 0.01 gesetzt, die Testmatrizen wurden beim Export aus R auf zwei Nachkommastellen gerundet. Die angegebenen Zeiten entsprechen den Werten, die für eine Testmatrix mit microbenchmark gemessen wurden.

Testfall 1: 5 Variable mit 2 Faktoren, keine Heywoodfälle [3]

Mit der Testmatrix

$$\begin{pmatrix} 1.00 & 0.35 & -0.06 & -0.22 & 0.04 \\ 0.35 & 1.00 & 0.07 & -0.32 & -0.15 \\ -0.06 & 0.07 & 1.00 & -0.05 & -0.08 \\ -0.22 & -0.32 & -0.05 & 1.00 & 0.02 \\ 0.04 & -0.15 & -0.08 & 0.02 & 1.00 \end{pmatrix}$$

wurden bei 100 Durchläufen mit microbenchmark folgende Zeiten gemessen [4]:

Name	Minimum	Durchschnitt	Maximum
Algorithmus 1	2.539997	2.923952	4.788883
Algorithmus 2	3.015996	3.788235	6.374214
Algorithmus 3	6.988436	9.093091	13.32487
Algorithmus 4	6.830214	8.735575	11.95954

Testfall 2: 10 Variable mit 3 Faktoren, keine Heywoodfälle

Mit der Testmatrix

[3] Während der Tests wurde schnell klar, dass manchmal Situationen eintreten, die bei einem Algorithmus zu einem Heywoodfall führen, bei anderen aber nicht. Für die Laufzeittests wurden daher nur die Fälle berücksichtigt, in denen keiner der Algorithmen einen Heywoodfall verzeichnete.

[4] Messzeiten in Millisekunden, wenn nicht anders angegeben

$$\begin{pmatrix}
1.00 & 0.05 & 0.27 & 0.10 & -0.23 & -0.13 & -0.12 & 0.11 & 0.14 & 0.11 \\
0.05 & 1.00 & 0.07 & -0.10 & 0.11 & -0.09 & -0.09 & -0.02 & -0.16 & 0.13 \\
0.27 & 0.07 & 1.00 & 0.23 & -0.03 & -0.05 & -0.15 & 0.10 & -0.31 & -0.12 \\
0.10 & -0.10 & 0.23 & 1.00 & -0.05 & -0.18 & -0.27 & 0.10 & -0.02 & 0.00 \\
-0.23 & 0.11 & -0.03 & -0.05 & 1.00 & -0.22 & -0.07 & -0.12 & -0.21 & -0.08 \\
-0.13 & -0.09 & -0.05 & -0.18 & -0.22 & 1.00 & 0.17 & 0.04 & 0.02 & 0.05 \\
-0.12 & -0.09 & -0.15 & -0.27 & -0.07 & 0.17 & 1.00 & -0.13 & 0.16 & -0.01 \\
0.11 & -0.02 & 0.10 & 0.10 & -0.12 & 0.04 & -0.13 & 1.00 & 0.08 & 0.00 \\
0.14 & -0.16 & -0.31 & -0.02 & -0.21 & 0.02 & 0.16 & 0.08 & 1.00 & 0.07 \\
0.11 & 0.13 & -0.12 & 0.00 & -0.08 & 0.05 & -0.01 & 0.00 & 0.07 & 1.00
\end{pmatrix}$$

wurden bei 100 Durchläufen mit microbenchmark folgende Zeiten gemessen:

	Minimum	Durchschnitt	Maximum
Algorithmus 1	1.656887	1.894602	3.737773
Algorithmus 2	1.963109	2.393091	5.224438
Algorithmus 3	6.715991	9.034397	27.30663
Algorithmus 4	16.18665	19.94597	56.36393

Man sieht, dass die iterierten Hauptachsenmethoden, also Algorithmus 1 und 2, im Vergleich zu Algorithmus 3 und 4 bei geringen Dimensionen und wenigen Faktoren etwas schneller arbeiten. Bei Algorithmus 3 könnte das an den Berechnungen liegen, die zusätzlich zur Berechnung der Ladungsmatrix angestellt werden. Diese zusätzliche Rechenzeit würde bei den sehr kurzen Laufzeiten stark auffallen.

In unseren beiden Beispielen wäre zu erwarten gewesen, dass Fall 1 für alle Algorithmen kürzere Laufzeiten hat als Fall 2. Für wenige Variable schwanken die Laufzeiten aber noch sehr stark, weshalb es zu Situationen wie dieser kommt. Bei größerer Variablenzahl ist das während der Tests nicht mehr aufgefallen.

Testfall 3: 25 Variable mit 7 Faktoren, keine Heywoodfälle

Testfall 3: Bei 25 Variablen und 7 Faktoren erhalten wir folgende Werte:

	Minimum	Durchschnitt	Maximum
Algorithmus 1	8.821767	11.09233	14.89909
Algorithmus 2	9.288877	12.0272	47.51594
Algorithmus 3	14.13509	16.18125	19.25686
Algorithmus 4	489.3029	553.1012	707.0622

Die Laufzeit von Algorithmus 4 stieg in diesem Test im Vergleich zu den vorherigen Fällen stark an, während die iterierten Hauptachsentransformationen und Algorithmus 3 bei keinem der Tests über 90 Millisekunden stiegen.

Testfall 4: 50 Variable mit 10 Faktoren, keine Heywoodfälle

Die Beispiel-Werte für 50 Variable und 10 Faktoren:

	Minimum	Durchschnitt	Maximum
Algorithmus 1	13.99468	16.30699	21.13113
Algorithmus 2	14.42357	16.48183	49.99916
Algorithmus 3	36.28448	39.4078	81.57252
Algorithmus 4	2.456288	2.746566	2.878144

Die Werte für Algorithmus 4 sind hier und in den folgenden Fällen in Sekunden angegeben und die Anzahl der Durchläufe für Algorithmus 4 wurde auf 10 reduziert.
Man sieht, dass die Minres Methode hier wieder deutlich länger benötigt. Eine Erklärung dafür wird nach den Laufzeittests gegeben.
Im Vergleich zum vorigen Testfall sehen wir auch, dass sich die Zeiten für die Algorithmen 1,2 und 3 weiter auseinander bewegen. In den nächsten Testfällen wird das weiter untersucht.

Testfall 5: 100 Variable mit 10 Faktoren, keine Heywoodfälle

	Minimum	Durchschnitt	Maximum
Algorithmus 1	67.87784	72.78153	116.8534
Algorithmus 2	69.33384	74.25507	115.7748
Algorithmus 3	99.13653	105.7474	140.3459
Algorithmus 4	16.78346	17.45529	18.4685

Wieder sieht man, dass die Laufzeit für Algorithmus 4 deutlich über der der anderen liegt, aber die Laufzeiten der Algorithmen 1 bis 3 sind von Testfall 4 zu Testfall 5 auch auf mehr als das Doppelte angewachsen. Dass der Maximalwert von Algorithmus 2 unter dem von Algorithmus 1 liegt, zeigt auch noch einmal, dass die Laufzeiten in 100 Durchläufen immer noch schwanken können.
Im nächsten Fall werden wir sehen, wie sich die Algorithmen bei höherer Faktorenanzahl verhalten.

Testfall 7: 100 Variable mit 20 Faktoren, keine Heywoodfälle

	Minimum	Durchschnitt	Maximum
Algorithmus 1	165.8717	172.6732	209.7789
Algorithmus 2	168.5846	175.8737	218.4446
Algorithmus 3	139.4135	156.7362	255.1104
Algorithmus 4	24.64487	26.11264	27.34449

Es scheint, als wären die iterierten Hauptachsenmethoden deutlich stärker von der Anzahl der Faktoren betroffen als die direkte Minimierung in Algorithmus 3. Die Veränderung in Algorithmus 4 ist natürlich absolut gesehen immer noch am größten. Die direkte Berechnung zeigte sich bei den Tests dieser Größe auch schneller als die iterierte Hauptachsentransformation. Die Frage ist, ob sich diese Beobachtungen im letzten Test wiederholen.

Testfall 8: 100 Variable mit 23 Faktoren, keine Heywoodfälle

	Minimum	Durchschnitt	Maximum
Algorithmus 1	182.5908	190.5112	228.6313
Algorithmus 2	188.6824	196.0464	237.0051
Algorithmus 3	135.1859	151.687	233.9469
Algorithmus 4	21.34096	22.67303	24.18869

Mit diesem letzten Testfall schließen wir unsere Studie zur Laufzeit ab. Überraschend waren in diesem Fall die Algorithmen 3 und 4, die sich in ein paar der durchgeführten Tests im Vergleich zum letzten Fall verbesserten. In den übrigen Fällen waren die Werte wie erwartet etwas höher als beim letzten Test, Algorithmus 4 war wieder am langsamsten, Algorithmus 3 wie schon im letzten Test der schnellste.

Bevor wir uns der Genauigkeit zuwenden, wollen wir noch kurz Algorithmus 4 näher betrachten. Bei den Tests wurde schnell klar, dass die Minres Methode nach Harman deutlich hinter den anderen zurückbleibt, was die Geschwindigkeit anbelangt. Das hat zum Teil mit der Implementierung zu tun, die in [1] beschrieben wird. Dabei wird nur nach einem vollständigen Durchlauf geprüft, ob die Abbruchbedingung erfüllt ist. Das heißt, dass erst alle Zeilen optimiert werden, bevor geprüft wird. Dadurch kommt es immer wieder vor, dass der Algorithmus länger läuft als er müsste und sogar genauere Ergebnisse liefert als er sollte. Wir werden Algorithmus 4 der Vollständigkeit halber auch in die folgenden Tests miteinbeziehen, müssen aber bedenken, dass er bei der Genauigkeit für den Vergleich manchmal nicht aussagekräftig ist.

6.6 Genauigkeit bei Nicht-Heywoodfällen

Nach der Beschäftigung mit den Laufzeiten, wollen wir noch evaluieren, wie genau die Algorithmen die vorgegebenen Matrizen approximieren können. Zu Beginn beschränken wir uns auf Nicht-Heywoodfälle und sehen dann im nächsten Abschnitt wie sich diese Genauigkeiten verändern.

Als Maß für die Genauigkeit wird die Quadratsumme der Differenzen der Nichtdiagonalelemente verwendet und außerdem noch der Betrag der jeweils größten Differenz betrachtet, also

$$
\begin{aligned}
m_1 &= \sum_{\substack{i,j \\ i \neq j}}^{p} \left(r_{ij} - \sum_{n=1}^{k} l_{in} l_{jn} \right)^2 \\
m_2 &= \max_{\substack{i,j \\ i \neq j}} |R - LL'|_{ij} \ .
\end{aligned}
$$

Die Werte in den nachfolgenden Tabellen sind wieder beispielhaft stellvertretend für die Ergebnisse der mehrmals durchgeführten Tests mit Matrizen der angegebenen Größe zu sehen. Wieder wurden die maximale Iterationsanzahl bei 1000 und die Toleranzgrenze bei 0.01 festgesetzt.

Da sich nicht alle Algorithmen nach den gleichen Regeln verhalten - wie wir bereits erörtert haben - ist der direkte Vergleich der Genauigkeiten nicht zielführend. Es wird daher im Folgenden weniger darum gehen, welcher Algorithmus die beste Genauigkeit bietet, sondern mehr darum, ob Algorithmus 2, also die verbesserte iterative Hauptachsentransformation, in etwa die gleichen Ergebnisse bietet wie die anderen.

	p=5, k=2	
	m_1	m_2
Algorithmus 1	0.004836267	0.02208287
Algorithmus 2	0.004836267	0.02208287
Algorithmus 3	0.003119533	0.02442068
Algorithmus 4	0.003396529	0.02306037

	p=10, k=3	
	m_1	m_2
Algorithmus 1	0.2928541	0.1558291
Algorithmus 2	0.2928541	0.1558291
Algorithmus 3	0.2917298	0.1609567
Algorithmus 4	0.2923826	0.1561601

	p=25, k=7	
	m_1	m_2
Algorithmus 1	1.287175	0.1379916
Algorithmus 2	1.287175	0.1379916
Algorithmus 3	1.286478	0.1374825
Algorithmus 4	1.287156	0.1390753

	p=50, k=10	
	m_1	m_2
Algorithmus 1	4.250577	0.1366562
Algorithmus 2	4.250577	0.1366562
Algorithmus 3	4.249637	0.1370952
Algorithmus 4	4.262946	0.1498736

	p=100, k=10	
	m_1	m_2
Algorithmus 1	16.48474	0.1801522
Algorithmus 2	16.48474	0.1801522
Algorithmus 3	16.48470	0.1803144
Algorithmus 4	16.50400	0.1778266

	p=100, k=20	
	m_1	m_2
Algorithmus 1	9.987599	0.1163665
Algorithmus 2	9.987599	0.1163665
Algorithmus 3	9.978615	0.1369878
Algorithmus 4	10.001948	0.1127608

Den Tabellen können wir entnehmen, dass die Genauigkeit bei allen Algorithmen recht ähnlich ist. Natürlich müssen Algorithmus 1 und 2 übereinstimmen, da sie ja für Nicht-Heywoodfälle das Gleiche leisten.
Aber auch Algorithmus 3 ist genauigkeitstechnisch sehr ähnlich und speziell bei größeren Matrizen sind die Unterschiede minimal.
Algorithmus 4 ist ebenfalls sehr nahe an den Werten von Algorithmus 2.
Algorithmus 3 und 4 sind bei Nicht-Heywoodfällen also nicht merklich genauer als Algorithmus 2.

6.7 Genauigkeit bei Heywoodfällen

In diesem Abschnitt konzentrieren wir uns auf Heywoodfälle und legen unseren Fokus darauf, mit welcher Genauigkeit die Algorithmen diesmal arbeiten. Algorithmus 1 ist für diese Tests nicht nötig, da die berechnete Ladungsmatrix immer zumindest ein Element hat, das größer als 1 ist, was die Lösung unzulässig macht. Aus Referenzgründen, sind

die Werte jedoch auch in den folgenden Tabellen enthalten.

Wie bei den Nicht-Heywoodfällen wollen wir sehen, wie die Genauigkeit der verbesserten iterativen Hauptachsentransformation im Vergleich mit der direkten Minimierung in Algorithmus 3 beziehungsweise der Minres Methode nach Harman in Algorithmus 4 ausfällt.
Es wurden nur Fälle verwendet, bei denen alle Algorithmen Heywoodfälle aufwiesen, die Anzahl der Heywood Prozeduren unterschied sich jedoch. Die maximale Anzahl der Iterationen liegt wieder bei 1000, die Toleranzgrenze wieder bei 0.01.
Allerdings wurde für die Fälle mit 100 Variablen die Toleranzgrenze auf 0,001 und die maximale Iterationsanzahl auf 5000 gesetzt, da bei den Tests mit den alten Werten kein Fall eintrat, bei dem alle Algorithmen die Heywood Prozedur verwendeten.

	p=5, k=2	
	m_1	m_2
Algorithmus 1	0.2673464	0.3032267
Algorithmus 2	0.3080379	0.3096199
Algorithmus 3	0.3060207	0.3091396
Algorithmus 4	0.3571130	0.2803741

	p=10, k=3	
	m_1	m_2
Algorithmus 1	0.4683781	0.2186618
Algorithmus 2	0.4850879	0.2272166
Algorithmus 3	0.4832559	0.2250124
Algorithmus 4	0.4860837	0.2014123

	p=25, k=7	
	m_1	m_2
Algorithmus 1	2.369705	0.2712453
Algorithmus 2	2.394456	0.2732956
Algorithmus 3	2.400827	0.2430883
Algorithmus 4	2.455952	0.2763771

	p=50, k=10	
	m_1	m_2
Algorithmus 1	4.669864	0.1523507
Algorithmus 2	4.718305	0.1552141
Algorithmus 3	4.717598	0.1550468
Algorithmus 4	6.293009	0.2837277

	p=100, k=10	
	m_1	m_2
Algorithmus 1	13.10663	0.1528848
Algorithmus 2	13.12401	0.1538903
Algorithmus 3	13.12317	0.1554629
Algorithmus 4	13.20306	0.1517394

	p=100, k=20	
	m_1	m_2
Algorithmus 1	11.69043	0.1456312
Algorithmus 2	11.77385	0.1453398
Algorithmus 3	11.77352	0.1453671
Algorithmus 4	13.73762	0.2568676

Bei den Tests mit 25 Variablen erreichte Algorithmus 4 oft die Iterationsgrenze. Bei den Fällen mit noch mehr Variablen erreichte Algorithmus 4 praktisch immer die 1000 beziehungsweise 5000 Iterationen und brach deswegen ab, was die plötzliche Verschlechterung

in der Genauigkeit erklärt.

Wie bereits bei den Nicht-Heywoodfällen beobachten wir, dass die iterative Hauptachsentransformation in Algorithmus 2 durchgehend gute Werte liefert und auf dem gleichen Level wie Algorithmus 3 arbeitet.

6.8 Testfazit

Die Ergebnisse aus den durchgeführten Tests werden präsentiert und auf die Stärken und Schwächen der einzelnen Algorithmen wird eingegangen.

Algorithmus 1

Algorithmus 1 ist eine schnelle und relativ genaue Methode, um die Faktorenanalyse durchzuführen. Der große Nachteil ist die Anfälligkeit für Heywoodfälle. Hier gibt es keinerlei Vorkehrungen, um eine nicht zulässige Lösung auszuschließen oder richtigzustellen. Obwohl die Laufzeit nicht sonderlich lang ist, gibt es keinen Grund, diesen Algorithmus in der Praxis zu benutzen oder in größere Analyseverfahren einzubinden. Die Gefahr einer falschen Lösung und folglich wertlosen Analyse ist zu groß.

Algorithmus 2

Aufbauend auf dem Grundgerüst von Algorithmus 1 ist Algorithmus 2 ebenfalls ziemlich schnell und genau. Die spezielle Prozedur für Heywoodfälle ist kurz gehalten und funktioniert gut.
Dass die Methode trotz der einfachen und nicht optimierten Implementierung mit der direkten Berechnungsmethode in Algorithmus 3 mithalten konnte, ist eine Überraschung. Nur bei sehr großen Fällen (100 Variablen) fällt die Methode geschwindigkeitstechnisch hinter Algorithmus 3 zurück. Die iterative Hauptachsentransformation mit ihrer Korrektur für Heywoodfälle ist damit trotzdem auf jeden Fall als Möglichkeit in Betracht zu ziehen, wenn eine Analysemethode in ein längeres Programm eingegliedert werden soll.
Bei den durchgeführten Tests zeigte der Algorithmus keine Probleme bei der Konvergenz und die Ergebnisse waren immer plausibel. Bei keinem der betrachteten Fälle wurde eine unzulässige Lösung beobachtet. Das alleine ist noch kein Beweis, spricht aber für den Algorithmus und die Methode.

Algorithmus 3

Algorithmus 3 ist eine zuverlässige Methode, um die Faktorenanalyse in R durchzuführen. Die Funktion wird seit Jahren immer wieder verbessert und liefert anders als die anderen

Algorithmen gleich viele andere Werte mit. Damit liefert die Funktion an sich bereits eine kurze Analyse, mit den anderen Werkzeugen aus dem psych Paket ist die Funktion sehr zu empfehlen.

Algorithmus 4

Leider bleibt Algorithmus 4 hinter den Erwartungen zurück. Die lange Laufzeit und hohe Anzahl an Operationen machen den Algorithmus nicht zur ersten Wahl. Wahrscheinlich könnte eine optimierte Implementierung hier deutliche Verbesserungen erzielen, trotzdem sind andere Methoden im Moment besser geeignet.
Außerdem ist beim Testen drei Mal der Fall eingetreten, dass der Algorithmus gar kein Ergebnis, sondern eine Fehlermeldung lieferte. Das fiel nur bei den Tests zu 100 Variablen auf und ist auf Rechenungenauigkeiten zurückzuführen. Bei der Berechnung von χ_k aus (22) wird in ein paar Fällen

$$1 - \sum_{i=1}^{k-1} \frac{\chi_i^2}{\lambda_i^2}$$

so klein, dass die Rechengenauigkeit von R nicht ausreicht und den Term als 0 weiterführt. Das führt zu einem Fehler in der restlichen Berechnung.

7 Fazit

Ziel der Arbeit war es, Heywoodfälle in der iterierten Hauptachsentransformation und in anderen Methoden zu untersuchen sowie die iterative Hauptachsentransformation auf Konvergenz hin zu analysieren.

Dabei konnten für die iterierte Hauptachsentransformation für drei Variable und einen Faktor stabile Punkte gefunden werden und die Konvergenz zumindest für diese Stellen gesichert werden.

Weiters wurde die Methode implementiert und in diversen Laufzeit- und Genauigkeitstests untersucht. Die verbesserte Version der iterierten Hauptachsentransformation zeigte sich sehr resistent gegen Heywoodfälle und ließ in den Tests keine Zweifel über ihre Konvergenz aufkommen.

Außerdem wurde die Minres Methode nach Harman implementiert und die unvollständigen Stellen aus [1] und [14] wurden ergänzt.

8 Quellenverzeichnis

Literatur

[1] HARRY H. HARMAN
Modern Factor Analysis
University of Chicago Press, Chicago London, 1976

[2] HANS HAVLICEK
Lineare Algebra für Technische Mathematiker
Heldermann Verlag, Lemgo Deutschland, 2008

[3] K. ÜBERLA
Faktorenanalyse
Springer-Verlag, Berlin Heidelberg, 1971

[4] R Core Team (2016). R: A language and environment for statistical computing.
R Foundation for Statistical Computing, Vienna, Austria.
URL https://www.R-project.org/

[5] Revelle, W. (2017) psych: Procedures for Personality and Psychological Research,
Northwestern University, Evanston, Illinois, USA,
https://CRAN.R-project.org/package=psych Version = 1.7.3.
Stand 8.5.2017

[6] Revelle, W. (2017) Procedures for Psychological, Psychometric, and Personality Research
psych package Handbuch
https://cran.r-project.org/web/packages/psych/psych.pdf

Stand 8.5.2017

[7] LUDWIG FAHRMEIR, ALFRED HAMERLE, GERHARD TUTZ
Multivariate statistische Verfahren
de Gruyter, Berlin, New York, 1996, 2. Auflage

[8] W. GURKER
Angewandte Mathematische Statistik
Skriptum, TU Wien, 2013

[9] Beschreibung und Erklärung des optim-Befehls
https://stat.ethz.ch/R-manual/R-devel/library/stats/html/optim.html
Stand 8.5.2017

[10] WOLFGANG WALTER
Analysis 2
Springer-Verlag Berlin Heidelberg, 2002, 5. Auflage

[11] Olaf Mersmann (2015). microbenchmark: Accurate Timing Functions.
R package version 1.4-2.1.
https://CRAN.R-project.org/package=microbenchmark

[12] Olaf Mersmann (2015) Accurate Timing Functions
microbenchmark package Handbuch
https://cran.r-project.org/web/packages/microbenchmark/microbenchmark.pdf
Stand 22.5.2017

[13] Funktion für positiv definite Matrizen
http://r.789695.n4.nabble.com/how-to-randomly-generate-a-n-by-n-positive-definite-matrix-in-R-td846858.html
Stand 23.5.2017

[14] Harman, H. H., and Fukuda Y.
Resolution of the Heywood case in the minres solution.
1966 Psych. 31:565-68

[15] JOSEPH ADLER
R IN A NUTSHELL
O'Reilly Verlag GmbH & Co. KG 2010, Deutsche Ausgabe

[16] David B. Dahl (2016). xtable: Export Tables to LaTeX or HTML.
R package version 1.8-2.
https://CRAN.R-project.org/package=xtable

[17] M. GARTNER
Einführung in R
Skriptum, TU Wien, 2015

[18] Code für die fa-Funktion aus dem psych Paket
https://github.com/cran/psych/blob/master/R/fa.R
Stand:30.11.2017

9 Anhang

9.1 Code zu Algorithmen

Die Funktionen Algo 1-4 sowie die Funktion zur Erstellung von Testmatrizen wurden außerdem in einem R Paket zusammengefasst. Das Paket heißt lsfa (least squares factor analysis).

9.1.1 Algorithmus 1

```
algo1<-function(Sigest,Dalt,f,t,tol) {

temp<-Sigest-Dalt

eigenvalues<-eigen(temp)$values
eigenvectors<-eigen(temp)$vectors

if (all(eigenvalues[1:f]>0)) {

eigenvalues<-sqrt(eigenvalues[1:f])
Lalt<-matrix(rep(eigenvalues[1:f],dim(eigenvectors)[2]),
dim(eigenvectors)[2],f,TRUE)*eigenvectors[,1:f]
Dalt<-diag(diag(Sigest-Lalt%*%t(Lalt)))
 }
 else {return("wähle f kleiner")}



i<-2
while(TRUE){

temp<-Sigest-Dalt

eigenvalues<-eigen(temp)$values
eigenvectors<-eigen(temp)$vectors

if (all(eigenvalues[1:f]>0)) {

eigenvalues<-sqrt(eigenvalues[1:f])
```

```
Lneu<-matrix(rep(eigenvalues[1:f],dim(eigenvectors)[2]),
dim(eigenvectors)[2],f,TRUE)*eigenvectors[,1:f]

Dneu<-diag(diag(Sigest-Lneu%*%t(Lneu)))

if (all(abs((Lneu-diag(Lneu))-(Lalt-diag(Lalt)))<=tol) | i>=t) {

erg<-list(L=Lneu,D=Dneu,i=i)
return(erg) }


else {

Dalt<-Dneu
Lalt<-Lneu
i<-i+1
}
}
else {return("wähle f kleiner")}
}
}
```

9.1.2 Algorithmus 2

```
algo2<-function(Sigest,Dalt,f,t,tol) {
i<-2
Dimd<-dim(Dalt)
E<-numeric(Dimd[2])

temp<-Sigest-Dalt

eigenvalues<-eigen(temp)$values
eigenvectors<-eigen(temp)$vectors

if (all(eigenvalues[1:f]>0)) {

eigenvalues<-sqrt(eigenvalues[1:f])

Lalt<-matrix(rep(eigenvalues[1:f],dim(eigenvectors)[2]),
dim(eigenvectors)[2],f,TRUE)*eigenvectors[,1:f]

E<-pmax(E+diag(Lalt%*%t(Lalt)-Sigest),numeric(Dimd[2]))

Dalt<-diag(diag(Sigest-Lalt%*%t(Lalt))+E)
}
else {return("wähle f kleiner")}

while(TRUE){

temp<-Sigest-Dalt

eigenvalues<-eigen(temp)$values
eigenvectors<-eigen(temp)$vectors

if (all(eigenvalues[1:f]>0)) {

eigenvalues<-sqrt(eigenvalues[1:f])

Lneu<-matrix(rep(eigenvalues[1:f],dim(eigenvectors)[2]),
```

```
dim(eigenvectors)[2],f,TRUE)*eigenvectors[,1:f]

E<-pmax(E+diag(Lneu%*%t(Lneu)-Sigest),numeric(Dimd[2]))

Dneu<-diag(diag(Sigest-Lneu%*%t(Lneu))+E)

if (all(abs((Lneu-diag(Lneu))-(Lalt-diag(Lalt)))<=tol) | i>=t) {

erg<-list(L=Lneu,D=Dneu,E=E,i=i)
return(erg) }


else {

Dalt<-Dneu
Lalt<-Lneu
i<-i+1
}
}
else {return("wähle f kleiner")}
}
}
```

9.1.3 Algorithmus 3

```
algo3<-function(R,n,k,t,tol) {


fa(R,nfactors=k,n.obs = n,n.iter=0, rotate="none",
min.err = tol, max.iter = t,symmetric=TRUE, warnings=TRUE, fm="minres")

}
```

9.1.4 Algorithmus 4

```
algo4<-function(R,n,k,t,tol) {


Reigenval<-eigen(R)$values
Reigenvec<-eigen(R)$vectors

heywoodcounter<-0

Reigenval<-sqrt(Reigenval[1:k])
A<-matrix(rep(Reigenval[1:k],dim(Reigenvec)[2]),dim(Reigenvec)[2],k,TRUE)*
Reigenvec[,1:k]

A_j<-list(A)
DeltaA<-list()
i<-1

while(TRUE){
j<-1
while(j<=n) {

Aj<-A
Aj[j,]<-0
Rstern<-matrix(,ncol=n,nrow=n)


for (a in 1:n){
```

```
for (b in 1:n){
Rstern[a,b]<-R[a,b]-sum(A[a,]*A[b,])
}
}

rj<-Rstern[j,]
rj[j]<-0
epsilonj<-rj%*%A%*%solve(t(Aj)%*%Aj)
A[j,]<-A[j,]+epsilonj

A_j[[(i-1)*n+j+1]]<-A


Komm<-sum(A[j,]**2)
if(Komm<1)    {


j=j+1

}    else {

heywoodcounter<-heywoodcounter+1

Atemp<-A
Atemp[j,]<-0
vn<-numeric(k)
W<-matrix(,nrow=k,ncol=k)

for(d in 1:k){

vn[d]<- sum(-(A[,d]*R[j,]))

K<-sum(R[j,]**2)

for(e in 1:k){
W[d,e]<-sum(Atemp[,d]*Atemp[,e])
```

```
}
}
lambda<-eigen(W)$values

Q<-eigen(W)$vectors

vndach<-t(Q)%*%vn

un<-(-vndach)/(lambda)

Kdach<-K-sum(vndach**2/lambda)

xin<-un*sqrt(lambda)

xin<-abs(xin)

chin<-numeric(k)
chinstern1<-numeric(k)

chin[1]<-xin[1]/sqrt((sum((xin/lambda))))


while(TRUE){

for(d in 2:(k-1)){
chin[d]<-(lambda[d]*xin[d]*chin[1])/
( (lambda[d]-lambda[1])*chin[1]+lambda[1]*xin[1] )
}

chin[k]<-sqrt(lambda[k]*(1-sum(chin[1:k-1]**2/lambda[1:k-1])))


t1<-lambda[k]*(1-xin[k]/chin[k])
t2<-lambda[1]*(1-xin[1]/chin[1])
diff<-abs(t1-t2)

if(diff < 10^(-8)){break} else{
```

```
chin[1]<-xin[1]/(1-(lambda[k]/lambda[1])*(1-(xin[k]/chin[k])))

}


}
w<-Q%*%(chin/sqrt(lambda))
A[j,]<-w

A_j[[(i-1)*n+j+1]]<-A

j<-j+1
} #else Ende


}

if(all(abs(A_j[[i*n+1]]-A_j[[(i-1)*n+1]])<=tol) | (i*n)>=t){

erg<-list(L=A_j[[(i-1)*n+1]],t=i*n, c=heywoodcounter)


return(erg)

} else{


i=i+1

}

}

}
```

9.2 Code zu Testmatrizen

```
gettestmatrix<-function(n){       #Erstelle eine Testmatrix mit Dim n x n
a<-Posdef(n)
a<-a/max(a)
return(a)
}
```

Code aus [13]:

```
Posdef <- function (n, ev = runif(n, 0, 10))
{
Z <- matrix(ncol=n, rnorm(n^2))
decomp <- qr(Z)
Q <- qr.Q(decomp)
R <- qr.R(decomp)
d <- diag(R)
ph <- d / abs(d)
O <- Q %*% diag(ph)
Z <- t(O) %*% diag(ev) %*% O
return(Z)
}
```

9.3 fa-Funktion

Aufgrund der Länge des Codes und der Beschaffenheit im Hinblick auf Sprünge innerhalb des Codes, wurde von einer Kopie im Anhang abgesehen. Der Code kann allerdings unter `https://github.com/cran/psych/blob/master/R/fa.R` [18]
eingesehen werden.