

# Explorative Faktorenanalyse

Bißantz, Jalynskij, Kupffer & Prestele

BF3 Testtheorie

- 1 Die “Large Data Set Challenge”
- 2 Das Common Factor Model (CFM)
- 3 Faktorenanalyse: “A hurdle race”
- 4 Extraktionsproblem
- 5 Rotationsproblem
- 6 Problem der Anzahl zu extrahierender Faktoren

# Section 1

## Einstieg

## Example

Hinter untenstehendem Link verbirgt sich ein Fragebogen der Plattform [openpsychometrics](https://openpsychometrics.org) (eine gute Quelle um kostenlos Datensätze für Analysen zu ergattern). Schauen Sie sich den Big-Five-Fragebogen einmal an. Wenn Sie möchten, beantworten Sie ihn kurz. Im Anschluss daran stellen wir uns auf dieser Grundlage der Large Dataset Challenge!

<https://openpsychometrics.org/tests/IPIP-BFFM/><sup>1</sup>

- Zeit: 2-3 Minuten

---

<sup>1</sup>Für das Projekt, siehe: <https://openpsychometrics.org/>

## Section 2

### Die “Large Data Set Challenge”

# Die "Large Data Set Challenge"

## Example

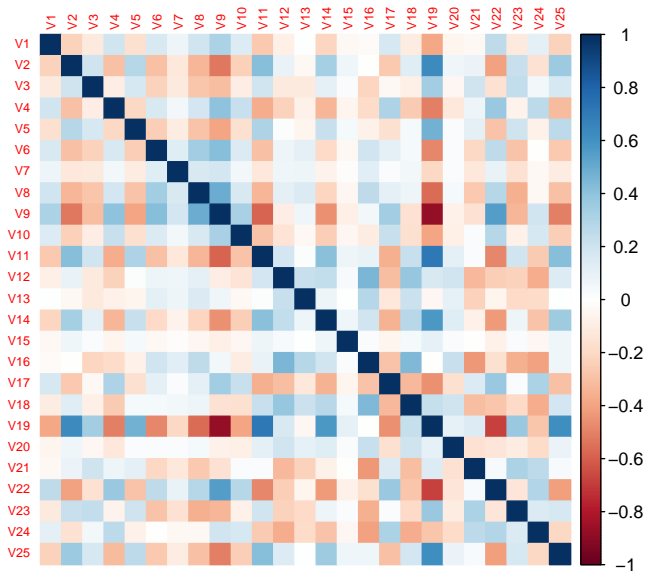
Stellen Sie sich vor, die von Ihnen soeben gesehenen/beantworteten Fragen ergäben die Korrelationsmatrix  $R$  auf der nächsten Folie. Die "Large Data Set Challenge" (LDC) lautet: Erkennen Sie eine Struktur in den Daten? Besprechen Sie sich mit ihrem Nachbarn.

Welche Items könnte man ihrer Meinung nach zu Itemgruppen zusammenfassen?

- Zeit: 2-3 Minuten

Anmerkung: Nein, das sind nicht ihre Antworten; V1 – V30 sind Zufallsvariablen!

# Übungsaufgabe 1: Struktur erkennen & Itemgruppen finden



# Die "Large Data Set Challenge" (..in theory)

## Large Data Set Challenge

Mit zunehmender Itemzahl nimmt die Anzahl der Korrelationen, die für eine Analyse zu berücksichtigen sind schnell zu. Die "Challenge" ist eine mögliche Struktur zu entdecken!

In a Nutshell:

- Problem: Anzahl der Korrelationen
- z.B.: 25 Items  $\hat{=}$   $2^{25} = 625$  Korrelationen
- Krux: Struktur erkennen
- $\Leftrightarrow$  finde: hoch korrelierende Itemgruppen



# Explorative Faktorenanalyse & LDC

- (ein) Hilfsmittel für die LDC: (explorative) *Faktorenanalyse*

## Faktorenanalyse

"The basic idea is to find latent variables (factors) based on the correlation structure of the manifest input variables (indicators)." (Mair 2018, S. 23)

- andere Helferlein zur *Datenreduktion* (eine Auswahl):
  - ▶ Hauptkomponentenanalyse
  - ▶ Clusteranalyse
  - ▶ Explorative Likertskalierung
  - ▶ (Non-) Metric Data Scaling (Distanzmatrizen!)

Wichtig: "meaningful" (Faktoren) vs. "full compression" (Komponenten)

## Section 3

### Strategie & Vorgehen: Simulation & Evaluation

# Strategie & Vorgehen: Simulation & Evaluation

- ① Man erschaffe  $\geq 1$  eine latente Variable (LV)
- ② ...lasse die LV Antwortmuster produzieren
- ③ ...wandel sie in eine Korrelationsmatrix (R) um
- ④ ... und versucht die Struktur mit der Faktorenanalyse aufzufinden

## Vom generativen Prozess zur Korrelationsmatrix

Der generative Prozess, d.h. wie genau ein Konstrukt die Antworten auf den Items erzeugt, bleibt meist verborgen. Wir untersuchen meistens lediglich Verhaltensspuren des Konstruktes, die sich in den Items ausdrückt, d.h. in der Struktur der Korrelationsmatrix niederschlägt. Strukturen zu simulieren ist hilfreich, weil wir dort "die Wahrheit" kennen und das Verfahren damit besser beurteilen können ( $\sim$  fake data analysis)

## Let's do it! (..in R)

“Playing Creator”: zwei latente Variable erschaffen<sup>2</sup>

```
# Faktorladungen; 8 items
load_F1 <- c(0.6, -0.3, 0.5, 0.7, 0.1, 0.2, 0.2, 0.3)
load_F2 <- c(-0.1, 0.1, 0.1, 0.1, -0.7, 0.5, -0.6, 0.7)
fx <- cbind(load_F1, load_F2)
# Zwischenfaktorkorrelation
phi <- diag(rep(1, 2)) ; phi[1, 2] <- phi[2, 1] <- 0.6
# Struktur
S <- psych::sim.structure(fx, phi, n=1000)
# Korrelations- und Datenmatrix
R <- S$model ; X <- S$observed # R <- cor(X)
```

---

<sup>2</sup>Das psych package bietet derzeit nicht die Möglichkeit eine Grafik des Prozesses in Rmarkdown einzubinden. Im R Skript zur folgenden Übung, können Sie die Grafik allerdings selbst erzeugen.

## Übungsaufgabe 2: Selbstexperiment

### Example

Versuchen Sie es selbst! Nutzen Sie den Code zur Übungsaufgabe 2 06-EFA.R und simulieren Sie ihr eigenes Beispiel. Verändern sie systematisch  $F1$ ;  $F2$  und  $\phi$ . Wie verändert sich die Korrelationsmatrix, in Abhängigkeiten Ihrer Veränderungen? Können Sie eine eindeutige Struktur konstruieren? Wenn ja, mit welchen Werten von  $F1$ ;  $F2$  und  $\phi$  haben Sie ihr Ziel erreicht?

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!

Zusatz: Haben Sie ein Strukturmodell gefunden, dass ihnen gefällt? Ja, dann überlegen Sie sich jetzt für welche Konstrukte diese Struktur Sinn macht (z.B.: *extraversion*  $\sim$  *openness to experience*)

# Logik latenter Variablen (..reversed)

## Von der Korrelationsmatrix zur latenten Variable

Die Faktoranalyse ist ein strukturentdeckendes Verfahren. D.h. wir versuchen den generativen Prozess zu rekonstruieren; d.h. wie ein Konstrukt die Antworten auf den Items verursacht hat anhand der Struktur die sich in der vorgegebenen Korrelationsmatrix zu *modellieren*.

- Modell: *Common Factor Model* (CFM)

## Eingangsgleichung des CFM

$$x = \Lambda\xi + \epsilon \quad (1)$$

*“In other words EFA tries to find  $p$  latent variables on the basis of the correlation structure of the  $m$  manifest variables.” (Ebd.)*

## Section 4

### Das Common Factor Model (CFM)

# Das Common Factor Model (CFM)

Anmerkung: Für die Reformulierung von Gleichung (1) zu (2): siehe McCallum (2009)

## Fundamentaltheorem

$$P = \Lambda\Phi\Lambda^t + \Psi \quad (2)$$

- $P$ : Modell-implizierte Korrelationsmatrix
- $\Lambda$ : Ladungsmatrix
- $\Phi$ : Matrix der Zwischenfaktorkorrelationen
- $\Psi$ : Uniqueness

## Zusammenhang: Modell & Struktur

Die von ihnen konstruierte Struktur versuchen wir nun mit der Faktorenanalyse unter Einsatz des CFM zu rekonstruieren. Das CFM ist also Ihr Tool im bevorstehenden Rekonstruktionsprozess!



## Section 5

### Fakotrenanalyse: “A hurdle race”

# Faktorenanalyse: “A hurdle race”

- ① Hürde: Extraktionsproblem
- ② Hürde: Rotationsproblem
- ③ Hürde: Problem der Anzahl zu extrahierender Faktoren

*“Unfortunately, factor analysis is frequently misunderstood and often misused. Some researchers appear to use factor analysis as a kind of divining rod, hoping to find gold hidden underneath tons of dirt. But there is nothing magical about the technique. [...] Factor analysis will yield meaningful results only when the research was meaningful to begin with.” Gregory (2014, S. 165)*

Konkl.: Versuchen Sie ihr Modell zu verstehen! (siehe: Selbststudium)

## Section 6

### Extraktionsproblem

## Extraktionsproblem

Wie extrahieren wir die zur Reproduktion nötigen Faktoren/Komponenten?

- ① Lösung: Bestimmung der “Principal Components” (*PCA*)
  - Modellgleichung:  $R = CC^t$
  - Note: Eigenwert-, Singulärwertzerlegung (closed form solution)
- ② Lösung: Iterative Bestimmung der “Principal Components” (*PAF*)
  - Modellgleichung:  $R^* = FF^t$
  - Note(s): Reduzierte Matrix, iterativer Prozess (convergence issues)
- ③ Lösung: Finde die plausibelsten (“most likely”) Werte (*MLF*)
  - Modellgleichung:  $P = \Lambda\Phi\Lambda^t + \Psi$
  - Note(s): Reduzierte Matrix, iterativer Prozess (convergence issues)

## Let's do it! (..in R): PCA

```
# Old School!
# (dino_pca <- princomp(X, cor=TRUE))
# New School
(pca_fit <- psych::principal(R, nfactors = 2,
                             rotate = "none"))

## Komponentenladungen
pca_fit$loadings
## Kommunalitäten
pca_fit$communality
## Einzigartigkeit
pca_fit$uniquenesses
## Eigenwerte
pca_fit$values
# Quadrierte multiple Korrelation
pca_fit$R2
```

## Übungsaufgabe 3: Selbstexperiment

### Example

Versuchen Sie es nun selbst! Nutzen Sie den Code zur Übungsaufgabe 3 06-EFA.R und führen eine PCA durch. Interpretieren Sie die entsprechenden Kennwerte für ihr Model und präsentieren Sie diese ihrem Nachbarn.

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!

Anmerkung: Normalerweise kennen Sie die Anzahl der zu extrahierenden Komponenten/Faktoren nicht. Wie man dieses Problem angeht, dazu gleich mehr!

## Let's do it! (..in R): Hauptachsenanalyse (PAF)

```
(fit_paf <- psych::fa(R, nfactors=2,  
                      rotate="none",  
                      fm="pa"))  
  
# Kommunalitäten  
fit_paf$communality  
  
# Eigenwerte  
fit_paf$e.values  
  
# Einzigartigkeit  
fit_paf$uniquenesses  
  
# Quadrierte multiple Korrelation  
fit_paf$R2
```

## Übungsaufgabe 4: Selbstexperiment

### Example

Versuchen Sie es nun selbst! Nutzen Sie den Code zur Übungsaufgabe 4 06-EFA.R und führen Sie eine Hauptachsenanalyse (PAF) durch. Vergleichen Sie die Ergebnisse der PAF mit denen der PCA. Bestehen Unterschiede zwischen den Ergebnissen? Zusatz: Haben Sie eine Vermutung wie mögliche Unterschiede zustande kommen?

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!



## Übungsaufgabe 5: Kritische Reflexion & Diskussion

### Example

Eine Folge der Extraktion mittels der Hauptkomponentenmethode ist, dass die extrahierten Faktoren unabhängig voneinander sind. Glauben Sie dem Modell uneingeschränkt, nähmen sie damit implizit an, dass auch die zugrundeliegende Konstrukte unabhängig voneinander sein müssten. Denken Sie an ihr Beispiel. Für wie plausibel halten Sie diese Annahme? Diskutieren Sie!

# Primer: Maximum Likelihood Faktorenanalyse (MLF)

## Kommunalitätenproblem

Das CFM ( $\Lambda\Phi\Lambda' + \Psi$ ) ist unbestimmt. D.h. es hat (deutlich) mehr unbekannte als bekannte Bestandteile. Die Folge: Das Modell kann deshalb nicht einfach "gelöst" werden. Wir müssen es schätzen.

- MLFA bietet nun als Lösung die plausibelsten (eng. "most likely") Werte zur Reproduktion der Informationen in der Korrelationsmatrix an. *"Assuming that the residual variance reflects normally distributed random error, the most elegant statistical solution is that of maximum likelihood"* (Revelle, in prep. S. 156)

## Let's do it (..in R): Maximum Likelihood Faktorenanalyse (MLF)

```
(fit_mlf <- psych::fa(R, nfactors=2,  
                      rotate="none",  
                      fm="ml"))  
  
# Kommunalitäten  
fit_mlf$communality  
# Eigenwerte  
fit_mlf$e.values  
# Einzigartigkeit  
fit_mlf$uniquenesses  
# Quadrierte multiple Korrelation  
fit_mlf$R2
```

# Übungsaufgabe 6: Selbstexperiment

## Example

Versuchen Sie es nun selbst! Nutzen Sie den Code zur Übungsaufgabe 6 in 06-EFA.R und führen Sie eine Maximum Likelihood Faktorenanalyse (MLF) durch. Vergleichen Sie die Ergebnisse mit den anderen Methoden. Prüfen die Validität der Ergebnisse in Bezug auf Ihr simuliertes Beispiel. Hat das Verfahren ihre simulierte Struktur entdeckt?

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!

Zusatz: Haben Sie eine Vermutung wie mögliche Unterschiede zwischen den Verfahren zustande kommen?

## Section 7

### Rotationsproblem

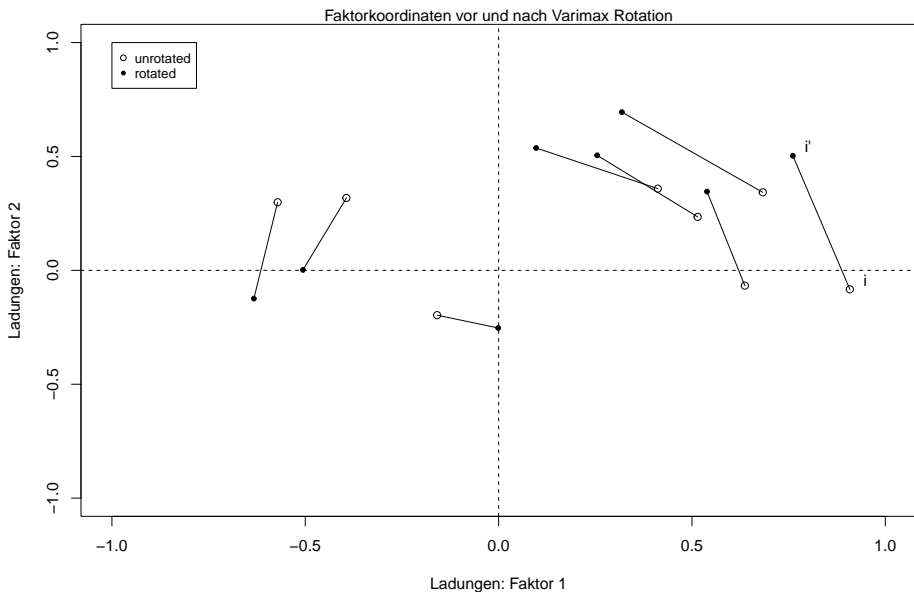
# Rotationsproblem

## Rotationsproblem

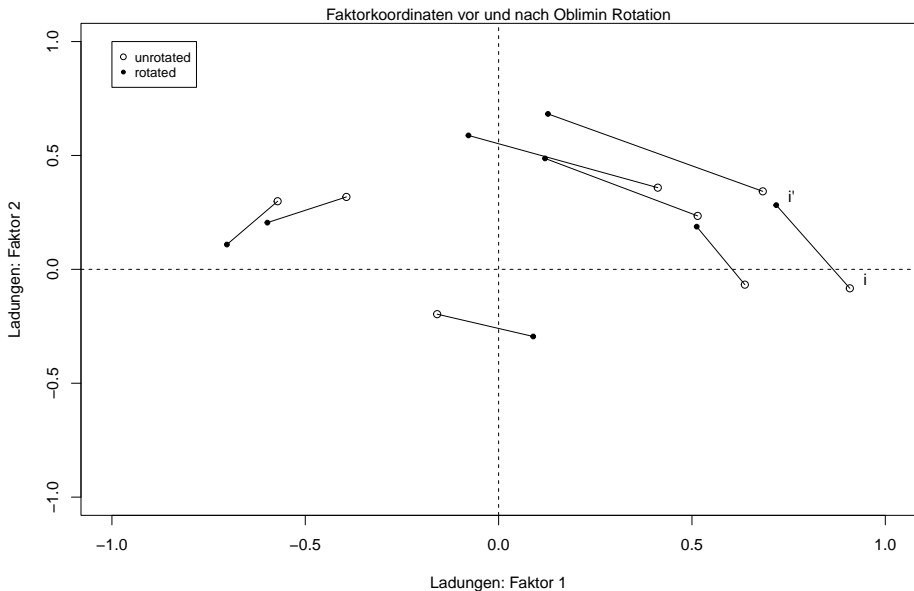
Wie rotiert die erhaltene Ladungsmatrix ( $\Lambda$ ) derart, dass eine möglich einfach interpretierbare Lösung daraus hervorgeht?

- ① Lösung: orthogonale Rotation
- ② Lösung: oblique Rotation

# Graphik: Orthogonale Rotation mit Varimax



# Graphik: Oblique Rotation mit Oblimin





## Let's do it (..in R): orthogonale Rotation (Varimax)

```
(fit_vmax <- psych::fa(R, nfactors=2,  
                        rotate="varimax",  
                        fm="ml"))  
  
# Kommunalitäten  
fit_vmax$communality  
  
# Eigenwerte  
fit_vmax$e.values  
  
# Einzigartigkeit  
fit_vmax$uniquenesses  
  
# Quadrierte multiple Korrelation  
fit_vmax$R2
```

## Let's do it (..in R): oblique Rotation (Oblimin)

```
(fit_obl <- psych::fa(R, nfactors=2,  
                      rotate="oblimin",  
                      fm="ml"))  
  
# Kommunalitäten  
fit_obl$communality  
# Eigenwerte  
fit_obl$e.values  
# Einzigartigkeit  
fit_obl$uniquenesses  
# Quadrierte multiple Korrelation  
fit_obl$R2  
# Wichtig: Phi  
fit_obl$Phi
```

# Übungsaufgabe 7: Selbstexperiment

## Example

Versuchen Sie es nun selbst! Nutzen Sie den Code zur Übungsaufgabe 7 in 06-EFA.R und führen Sie eine MLF durch. Rotieren Sie anschließend die Ergebnisse mit der Varimax und Oblimin Rotation. Wie verändert die Rotation ihre Interpretation der Ergebnisse? Erleichtert sie sie? Vergleichen Sie die Lösung dazu (1) mit der unrotierten Lösung und vergleichen Sie (2) die Ergebnisse zwischend den Rotationsmethoden.

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!

Zusatz: Sie kennen das “richtige” Ergebnis. Sie haben die Antworten immerhin simuliert. Welche Lösung kommt ihren Ergebnissen am nächsten? Haben Sie eine Idee warum? (Tipp: Denken Sie an  $\Phi$ )

## Section 8

### Problem der Anzahl zu extrahierender Faktoren

# Problem der Anzahl zu extrahierender Faktoren

## Rotationsproblem

Wie ermittle ich die Anzahl der zu extrahierenden Faktoren ohne sie zu kennen?

- ① Lösung: Kaiser Kriterium
- ② Lösung: Scree-Test
- ③ Lösung: Horns Parallel Analyse

Anmerkung: Es gibt mittlerweile zahlreiche Weiterentwicklungen. Interessieren Sie sich dafür? (siehe: Lorenzo-Seva et al., 2011 & Timmerman et al., 2018)

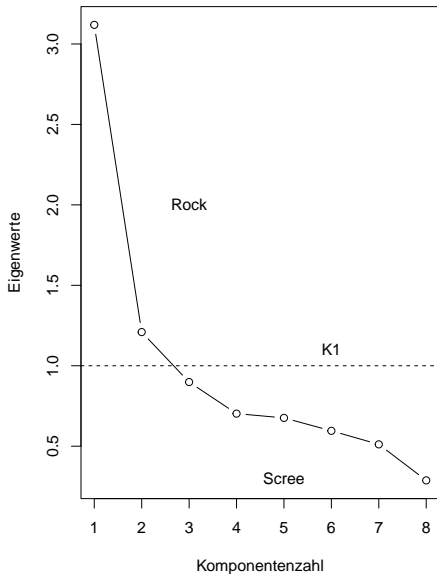
# EigenwertepLOTS (Kaiser Kriterium & Scree Test)

## Kaiserkriterium & Scree Test

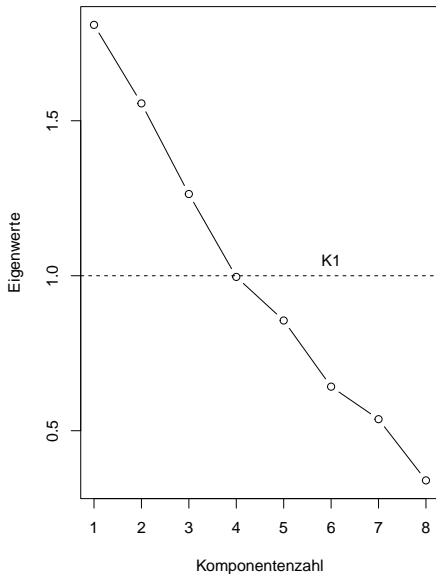
Zerlege die Korrelationsmatrix in ihre Eigenwerte (SVD/EVD) und plotte diese gegen einen Index. (1) Kaiserkriterium: Ist das Varianzaufklärungspotential eines Faktors kleiner als eines Items beende deine Suche ( $\cong EW > 1$ ). (2) Scree-Test: Finde den "ellbow" der den "rock" vom "scree" unterscheidet.

# Grafik: Eigenwertplot

Eigenwertplot



Eigenwertplot



## Let's do it (..in R): Eigenwertplots

```
# Barfuß  
ev <- eigen(R)$values  
plot(ev, main="Eigenwertplot", xlab="Eigenwerte",  
      ylab = "Komponentenzahl", type="b")  
abline(h=1, lty=2)  
  
# psych package  
# psych::scree(R, factors = FALSE)  
# psych::scree(R, factors = TRUE)
```

Anmerkung: Der Code ist auskommentiert, dass er in RMD nicht durchläuft



# Parallel Analyse

## Parallel Analyse

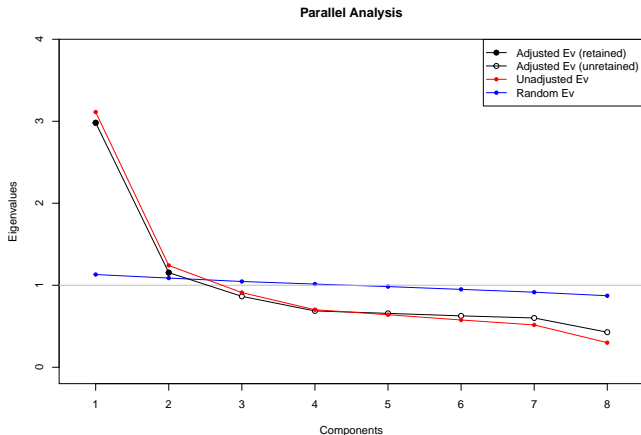
In parallel analysis, the criterion to be used in order to determine the number of factors is the following A factor is considered as "significant" if its eigenvalue is larger than the 95random or resampled data. (Mair 2018, S.31)

Anmerkung: Im Original wurde die PA für Komponenten konstruiert. Mittlerweile gibt es sie auch für das CFM.

# Grafik: Parallel Analyse (PCA)

##

## Using eigendecomposition of correlation matrix.

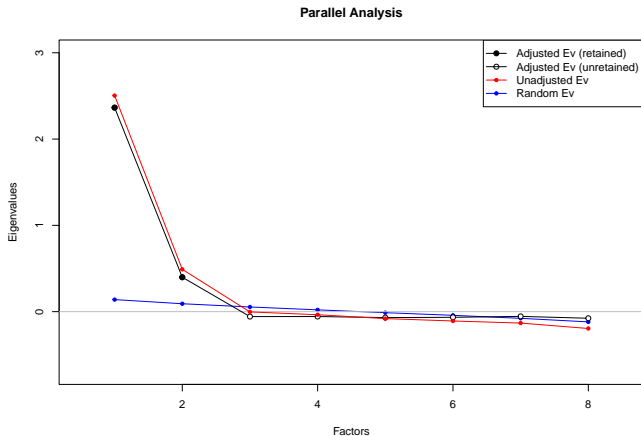


Anmerkung: Die Grafik wurde mit paran nicht psych erstellt.

# Grafik: Parallel Analyse (FA)

##

## Using eigendecomposition of correlation matrix.



Anmerkung: Die Grafik wurde mit paran nicht psych erstellt.

## Let's do it (..in R): Parallel Analyse

```
# Lange Laufzeit  
psych::fa.parallel(X, fa = "pc", fm = "ml")  
  
# Alternative  
#  
# Für Komponenten  
paran::paran(X)  
# Für Faktoren  
paran::paran(X, cfa=TRUE)
```

# Übungsaufgabe 8: Selbstexperiment

## Example

Versuchen Sie es nun selbst! Nutzen Sie den Code zur Übungsaufgabe 8 in 06-EFA.R und finden Sie die Anzahl zu extrahierender Faktoren in ihrem Beispiel mit den drei genannten Möglichkeiten. Welches der drei Verfahren "errät" ihre Anzahl zu extrahierender Faktoren richtig? Haben Sie eine Idee warum? (Tipp:  $\Lambda$ ,  $\Phi$ )

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!

Anmerkung: Sie haben den Luxus, das richtige Ergebnis zu kennen. Normalerweise ist das nicht der Fall! Darum sind Simulationen wichtig. So können wir das Verhalten einzelner Kriterien unter verschiedenen Umständen testen.

# (Minimal-)Voraussetzungen: Faktorenanalyse

## Minimalvoraussetzungen

Sind meine Daten, bzw. ist meine Korrelationsmatrix zur Analyse überhaupt geeignet?

- ① Lösung: Bartlett-Test
- ② Lösung: Kaiser-Meyer-Olkin Kriterium (KMO)

# (Minimal-)Voraussetzungen: Bartlett-Test

## Bartlett-Test

Ist die zu faktorisierende Korrelationsmatrix signifikant von einer Identitätsmatrix verschieden?

$$H_0 : \text{verwerfen} \Leftrightarrow R = I \quad (3)$$

## Output: Identitätsmatrix

```
(I <- diag(8))
```

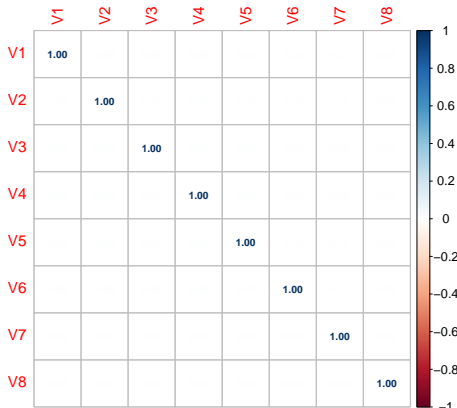
##		[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]
##	[1,]	1	0	0	0	0	0	0	0
##	[2,]	0	1	0	0	0	0	0	0
##	[3,]	0	0	1	0	0	0	0	0
##	[4,]	0	0	0	1	0	0	0	0
##	[5,]	0	0	0	0	1	0	0	0
##	[6,]	0	0	0	0	0	1	0	0
##	[7,]	0	0	0	0	0	0	1	0
##	[8,]	0	0	0	0	0	0	0	1



# Grafik: I vs. R

I

R



## Let's do it (..in R): Bartlett-Test

```
# Datenmatrix (X)  
# psych::cortest.bartlett(X)  
# Korrelationsmatrix (R)  
psych::cortest.bartlett(R, n=nrow(X))
```

```
## $chisq  
## [1] 1950.464  
##  
## $p.value  
## [1] 0  
##  
## $df  
## [1] 28
```

## (Minimal-) Voraussetzungen: KMO

### KMO

Welchen Anteil an der Gesamtvarianz der Items ist auf gemeinsame Varianz rückföhrbar?

Anmerkung: Der gemeinsame Varianzanteil ist es letztlich, was die Faktoren aufgreifen können.

$$KMO = \frac{\sum_{j \neq k} \sum r_{jk}^2}{\sum_{j \neq k} \sum r_{jk}^2 + \sum_{j \neq k} \sum p_{jk}^2} \in [0, 1] \quad (4)$$

$r_{jk}$  : Korrelation zwischen zwei Items  $p_{jk}$  : Partielle Korrelation

## Let's do it! (..in R)

```
(kmo <- psych::KMO(X))
```

```
## Kaiser-Meyer-Olkin factor adequacy  
## Call: psych::KMO(r = X)  
## Overall MSA = 0.81  
## MSA for each item =  
##      V1      V2      V3      V4      V5      V6      V7      V8  
## 0.84 0.84 0.88 0.80 0.80 0.89 0.74 0.76
```

```
# kmo$MSA
```

## Übungsaufgabe 9: Selbstexperiment

### Example

Versuchen Sie es nun selbst! Nutzen Sie den Code zur Übungsaufgabe 9 in 06-EFA.R und führen Sie einen Bartlett-Test durch. Verwenden Sie auch das KMO-Kriterium auf ihre Korrelations- oder Datenmatrix an. Wie beurteilen Sie die Ergebnisse? Sind ihre Matritzen zur Analyse geeignet?

- Zeit: 5 Minuten
- Replikation: `set.seed(123)`
- Anmerkung: Konzepte & Output verstehen  $\gg$  Codes verstehen!

# Übungsaufgabe 10: Evaluation

## Example

Welches Verfahren hat ihre anfängliche Struktur am besten aufgefunden?  
Haben Sie eine Idee warum? (Tipp: Denken Sie an  $\Phi$ )

Anmerkung: Wiederholen Sie die Übung im Selbststudium mit anderen Werten. Verändern Sie diese systematisch und prognostizieren Sie, was geschehen wird und warum. So lernen Sie jedes Verfahren, jeden Algorithmus und jedes Kriterium verstehen.

## Section 9

### Epilogue: The Power of Simulation!

# Epilogue: The Power of Simulation!

... Wenn Sie wissen wollen, wie sich ein bestimmtes Kriterium unter spezifischen Voraussetzungen verhält, dann simulieren Sie!

## The Power of Simulation

Sie besitzen nun die Power der Simulation! Simulieren Sie sich bestimmte Strukturen (wie in diesem Kurs getan) und testen Sie die Kriterien, Kennwerte, usw. auf Herz und Nieren! Nur so lernen Sie ihre Grenzen kennen.

Anmerkung: Gute Klausurübung! Schnappen Sie sich ihre Mitstudierenden simulieren Sie mögliche Welten und üben, üben, üben Sie!



# Section 10

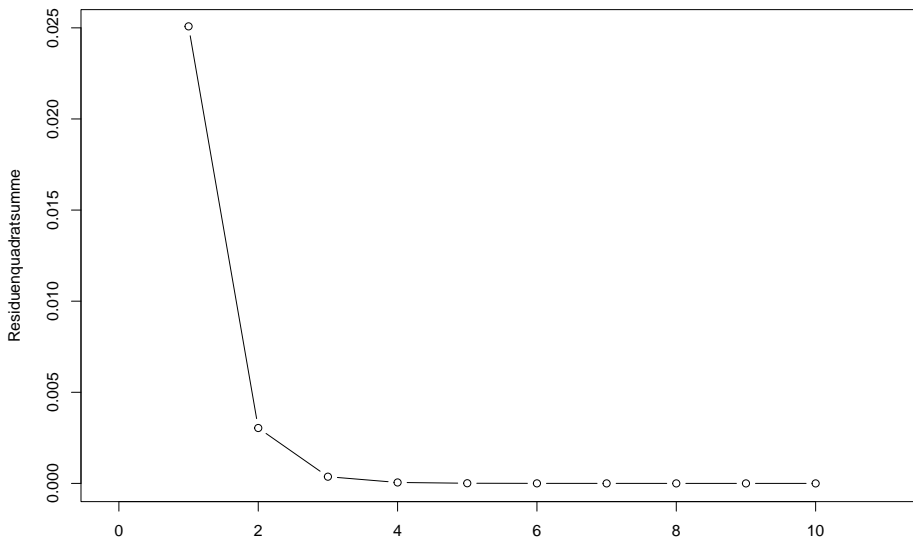
## Selbststudium

# Selbststudium 1: die Relevanz von Iterationen

*It is a useful exercise to run fa using the principal axis factor method (fm= "pa") and specifying the number of iterations (e.g., max.iter=2). Then examine the size of the residuals as the number of iterations increases. When this is done, the solution gets progressively better, in that the size of the residuals in the off diagonal matrix become progressively smaller. (Revelle, in prep., S. 152)*

# Selbststudium 1: die Relevanz von Iterationen

```
## [1] 2.508165e-02 3.040011e-03 3.681981e-04 5.153387e-05 9.111111e-06  
## [6] 1.994291e-06 5.300246e-07 1.605099e-07 1.605099e-07 1.605099e-07
```



## Selbststudium 2: händische PCA – Eigenwertzerlegung

```
# Black Box  
pca_fit <- princomp(X, cor = TRUE)  
  
# Barfuß  
R_EVD <- cor(X)  
# EVD von R ; Eigenvektoren ; Eigenwerte  
EVD_R <- eigen(R_EVD)  
# Komponentenladungen  
loadings <- EVD_R$vectors  
# Gleich?  
pca_fit$loadings ; print(loadings, digits=3)
```

## Selbststudium 3: händische EFA – Singulärwertzerlegung

```
# Black Box
pca_fit <- prcomp(X, scale. = TRUE)

# Barfuß
n <- nrow(X)
SVD_X <- svd(scale(X)/sqrt(n-1))
U <- SVD_X$u ; D <- SVD_X$d
loadings <- U %*% diag(D) * sqrt(n-1)
# Gleich?
pca_fit$x ; loadings
```

## Selbststudium 4: reduzierte Korrelationsmatrix

... Der Output zeigt die erst 8 Iterationen einer PAFA. Vergleichen Sie diese mit ihre am Anfang konstruierte Korrelationsmatrix  $R$ ). Achten Sie auf die Hauptdiagonalen!

## Selbststudium 5: PAF

*[P]rincipal axes factor analysis. This is similar to principal components, except that it is done with a reduced matrix where the diagonals are the communalities. The communalities can either be (1) specified a priori, (2) estimated by such procedures as multiple linear regression, or (3) found by iteratively doing an eigenvalue decomposition and repeatedly replacing the original 1s on the diagonal with the value of  $1 - u^2$  where  $U^2 = \text{diag}(R - FF')$ . (Revelle, in prep., S.156)*

## Selbststudium 5: händische PAF (Basialgorithmus, 1 Iteration)

```
# EVD of R
EVD_R <- eigen(R)
# Loadings
F <- EVD_R$vectors
# Residual matrix
R_ast <- R - F %*% t(F)
# Squared uniqueness matrix
U2 <- diag(R_ast)
# Replace on diagonals
(diag(R) <- 1 - U2)
R
```



## Selbststudium 5: händische PAF (1 Iteration)

```
EVD_R <- eigen(R) # EVD of R
F <- EVD_R$vectors # Loadings
k <- 5 # k largest PC
# Pre reproduction
Fk <- F[,seq(k)]
Z <- matrix(0, ncol = ncol(F) - k, nrow(F))
Fk <- cbind(Fk, Z)
# Residual matrix
R_ast <- R - Fk %*% t(Fk)
# Squared uniqueness matrix
U2 <- diag(R_ast)
# Replace on diagonals
diag(R) <- 1 - U2
#...Repeat
round(R, digits = 3)
```

## Selbststudium 5: händische PAF ( $\geq 1$ Iteration)

Anmerkung: Führen Sie das Code Snippet solange immer wieder aus, bis sich die Veränderungen in der Hauptdiagonale stabilisieren. Jede Wiederholung des Codes stellt eine Iteration in der Maschinerie der PAF dar. Nichts anderes passiert innerhalb des Algorithmuses den Sie in konventionellen Packages finden. ..nur das die Algorithmen deutlich performanter und effizienter implementiert sind.

# Literaturverzeichnis I

- Dinno, Alexis. 2018. *Paran: Horn's Test of Principal Components/Factors*.  
[http://alexisdinno.com/Software/files/PA\\_for\\_PCA\\_vs\\_FA.pdf](http://alexisdinno.com/Software/files/PA_for_PCA_vs_FA.pdf).
- Francois, Romain. 2020. *Bibtex: Bibtex Parser*.  
<https://github.com/romainfrancois/bibtex>.
- Lorenzo-Seva, Urbano, Marieke E. Timmerman, and Henk A. L. Kiers. 2011.  
“The Hull Method for Selecting the Number of Common Factors” 46 (2):  
340–64. <https://doi.org/10.1080/00273171.2011.564527>.
- R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.  
<https://www.R-project.org/>.
- Revelle, William. 2021. *Psych: Procedures for Psychological, Psychometric, and Personality Research*.  
<https://personality-project.org/r/psych/>  
<https://personality-project.org/r/psych-manual.pdf>.

- Timmerman, Marieke E., Urbano Lorenzo-Seva, and Eva Ceulemans. 2018. "The Number of Factors Problem." In, 305–24. John Wiley & Sons, Ltd. <https://doi.org/10.1002/9781118489772.ch11>.
- Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown Cookbook*. Boca Raton, Florida: Chapman; Hall/CRC. <https://bookdown.org/yihui/rmarkdown-cookbook>.