Studienabschlussarbeiten
Fakultät für Mathematik, Informatik
und Statistik

Seiler, Moritz:

Statistical Shape Analysis unter Verwendung der Independent Component Analysis: Ein methodischer Vergleich

Bachelorarbeit, Sommersemester 2017

Fakultät für Mathematik, Informatik und Statistik

Ludwig-Maximilians-Universität München

https://doi.org/10.5282/ubm/epub.40349

Statistical Shape Analysis unter Verwendung der Independent Component Analysis: Ein methodischer Vergleich

Moritz Seiler

04/05/2017

Statistical Shape Analysis unter Verwendung der Independent Component Analysis: Ein methodischer Vergleich

Moritz Seiler

Bachelorthesis

Institut für Statistik
Ludwig–Maximilians–Universität
München

vorgelegt von
Moritz Seiler

München, den 04/05/2017

Inhaltsverzeichnis

Zusammenfassung **xi**

1 Einleitung **1**

2 Statistical Shape Analysis **3**

2.1 Shape . . . 3
2.2 Landmark . . . 5
2.3 Shape Space . . . 6
2.4 Procrustes-Analyse . . . 7

3 Independet Component Analysis **13**

3.1 Definition . . . 13
3.2 Vergleich zwischen ICA und PCA . . . 17
3.3 Vorverarbeitung . . . 19
3.3.1 Zentrierung . . . 19
3.3.2 Whitening . . . 20
3.4 Unabhängigkeit . . . 21
3.5 Nicht-Normalität . . . 22
3.5.1 Erfordernis der Nicht-Normalität . . . 22
3.5.2 Maße der Nicht-Normalität . . . 23
3.5.2.1 Kurtosis . . . 23
3.5.2.2 Negentropie . . . 24
3.6 Methoden . . . 26
3.6.1 FastICA . . . 26
3.6.2 InfoMax . . . 30
3.6.3 JADE . . . 32
3.7 Limitationen . . . 36

4 Methodischer Vergleich **37**
4.1 Datenbeschreibung 37
4.2 Datenaufbereitung 39
4.3 Point Distribution Model unter Verwendung der ICA 42
4.4 Ergebnisse 44
4.4.1 Computationaler Vergleich 45
4.4.2 Vergleich der Variabilität der Shapes 48

5 Fazit **57**

A Computationaler Vergleich - Übersicht **59**

B Variabilitätsstrukturen **61**

C Komponentenvektoren der Variationen **67**

Abbildungsverzeichnis

2.1 Landmarks auf der Kontur des Corpus Callosums 5
2.2 Vergleich der Landmark-Positionen des Corpus Callosums zweier Gruppen 6
2.3 Zwei Shapes vor und nach der Procrustes-Analyse 7

3.1 Cocktail-Party Problem 16
3.2 Streudiagramm der Realisierungen zweier gleichverteilter Zufallsvariablen . 17
3.3 Streudiagramm der beobachtbaren Mischungen 18
3.4 Dekomposition der Mischungen anhand PCA und ICA 19
3.5 Feedforward Netz des `InfoMax` Algorithmus 30

4.1 Shape aus dem Datensatz `digit3.dat` sowie `mice` 38
4.2 Shapes vor und nach der Zentrierung (Datensatz: `digit3.dat`) 39
4.3 Angeglichene Shapes und mittlere Shape (Datensatz: `digit3.dat`) 41
4.4 Punktwolken der Landmarks im Datensätze `digit3.dat` und `mice` mit mittlerer Shape 42
4.5 Anzahl der Iterationen der Algorithmen (Datensatz: `digit3.dat`) 45
4.6 Anzahl der Iterationen der Algorithmen (Datensatz: `mice`) 46
4.7 Rechenzeit der Algorithmen (Datensatz: `digit3.dat`) 47
4.8 Rechenzeit des `FastICA` sowie des `InfoMax` (Datensatz: `digit3.dat`) . . . 47
4.9 Rechenzeit der Algorithmen (Datensatz: `mice`) 48
4.10 Variabilitätsstruktur der generierten Shapes mit $c = \pm 1$ bei 2 ICs (Datensatz: `digit3.dat`) 50
4.11 Variabilitätsstruktur der generierten Shapes mit $c = \pm 1$ bei 2 ICs (Datensatz: `mice`) 51
4.12 Variabilitätsstruktur der generierten Shapes mit $c = \pm 1$ bei 22 ICs (Datensatz: `digit3.dat`) 52

4.13 Variabilitätsstruktur der generierten Shapes mit $c = \pm 1$ bei 8 ICs (Datensatz: `mice`) 53

C.1 Unabhängige Komponentenvektoren bei 2 ICs (Datensatz: `digit3.dat`) . . 67

C.2 Unabhängige Komponentenvektoren bei 2 ICs (Datensatz: `mice`) 68

Tabellenverzeichnis

4.1 Mathematische Landmarks (Datensatz: `digit3.dat`) 38
4.2 Mathematische Landmarks (Datensatz: `mice`) 38
4.3 Globale Variabilität sowie Größe der Shapes mit $c = 1$ (Datensatz: `digit3.dat`) 54
4.4 Globale Variabilität sowie Größe der Shapes mit $c = 1$ (Datensatz: `mice`) . 55

A.1 Computionaler Vergleich der Algorithmen (Datensatz: `digit3.dat`) 59
A.2 Computionaler Vergleich der Algorithmen (Datensatz: `mice`) 60

Zusammenfassung

In der vorliegenden Arbeit wird die Anwendung der *Independent Component Analysis* im Rahmen der *Statistical Shape Analysis* betrachtet, dabei werden die Algorithmen `FastICA`, `InfoMax` und `JADE` der *Independent Component Analysis* verglichen. Ein Vergleich der Konvergenzgeschwindigkeit sowie der benötigten Rechenzeit der Algorithmen untersucht indes die computationalen Eigenschaften der Optimierungsalgorithmen jener ICA Methoden. Dabei ist für den `JADE` Algorithmus die höchste Konvergenzgeschwindigkeit der betrachteten Algorithmen zu beobachten, jedoch benötigt dieser eine deutlich längere Rechenzeit in höher-dimensionalen Räumen. Der `FastICA` und der `InfoMax` stellen hingegen einen Kompromiss zwischen diesen betrachteten Untersuchungsmerkmalen dar.
Neben eines computationalen Vergleichs wird ferner die Variabilitätsstruktur von *Shapes* im Rahmen eines modifizierten *Point Distribution Model* unter Verwendung der drei ICA Algorithmen verglichen. Dabei ist eine Abhängigkeit der Variabilitätsstruktur der *Shapes* von der Anzahl zu schätzender unabhängiger Komponenten zu beobachten; so führt eine geringe Anzahl an Komponenten zu einer globalen Variationsstruktur der *Shapes* durch die einzelnen Komponenten, wohingegen diese mit zunehmender Anzahl an Komponenten eine stärkere lokalen Differenzierung der Variation zu verzeichnen ist. Ein Vergleich der betrachteten Variationsstrukturen zwischen den Algorithmen `FastICA`, `InfoMax` sowie `JADE` zeigt hingegen eine hohe Ähnlichkeit der Ergebnisse, lediglich marginale lokale Unterschiede waren dabei zu beobachten.

Kapitel 1

Einleitung

Die *Statistical Shape Analysis* spielt gegenwärtig durch eine fortschreitende Digitalisierung eine entscheidende Rolle in einer Vielzahl von Bereichen wie beispielsweise der *Computer Vision*, dem *Medical Imaging* oder der Biometrie. Dabei werden im Rahmen der *Statistical Shape Analysis* die geometrischen Eigenschaften der *Shapes* untersucht und verglichen. Als klassisches Instrument im Rahmen dieser Analyse dient die *Principal Component Analysis*, auf Basis derer eine geeignete Repräsentation der *Shapes* bestimmt wird (Zhang & Golland 2016, S. 156). Als eine Erweiterung jener *Principal Component Analysis* ist die *Independent Component Analysis* anzusehen, die in den vergangenen Jahren in vielen Bereichen Anwendung gefunden hat (Hyvärinen, Karhunen *u. a.* 2001, S. XVIIf.). Im Rahmen der vorliegenden Arbeit wird dabei die Verwendung der *Independent Component Analysis* im Kontext der *Statistical Shape Analysis* betrachtet.

Das Ziel der Arbeit ist dabei ein Vergleich der Algorithmen `FastICA`, `InfoMax` sowie `JADE` der *Independent Component Analysis* im Kontext der *Statistical Shape Analysis* in Form eines computationalen Vergleichs sowie eines strukturellen Vergleichs hinsichtlich der Modellierung der Variabilität von *Shapes*.

Der Aufbau der vorliegenden Arbeit gliedert sich dabei in zwei Teile, so werden in Kapitel 2 und Kapitel 3 die theoretischen Grundlagen sowohl der *Statistical Shape Analysis* als auch der *Independent Component Analysis* betrachtet, die anschließend in Kapitel 4 im Rahmen eines praktischen methodischen Vergleichs zusammengefügt werden.

In Kapitel 2 werden die Grundlagen der *Statistical Shape Analysis* betrachtet. Dazu werden einführend die Definitionen einer *Shape* sowie eines *Landmarks* gegeben, um darauf aufbauend einen *Shape Space* zu definieren. Im Anschluss an diese definitorische Einführung wird die im Rahmen der *Statistical Shape Analysis* in dieser Arbeit im Fokus stehende

Procrustes-Analyse vorgestellt. In Kapitel 3 werden die theoretischen Grundlagen der *Independent Component Analysis* dargestellt. Im Rahmen dieser Betrachtung wird eingangs in Abschnitt 3.1 eine Definition der *Independent Component Analysis* gegeben, in Rahmen derer insbesondere das generative Modell der *Independent Component Analysis* dargestellt wird. In Abschnitt 3.2 folgt darauf eine Gegenüberstellung der *Independent Component Analysis* sowie der *Principal Component Analysis* zu einer Abgrenzung jener Konzepte. Anschließend werden die Vorverarbeitungsschritte zur Vereinfachung der anschließenden *Independent Component Analysis* beschrieben. Ferner werden in den Abschnitten 3.4 und 3.5 die zentralen statistischen Eigenschaften in Form der stochastischen Unabhängigkeit und der Nicht-Normalität im Rahmen einer *Independent Component Analysis* dargestellt. Darauf aufbauend werden in Abschnitt 3.6 schließlich die Algorithmen `FastICA`, `InfoMax` sowie `JADE` der *Independent Component Analysis* betrachtet. Abschließend folgt eine Darstellung der Limitationen dieser statistischen Methodik.
In Kapitel 4 werden diese theoretischen Ausführungen zusammengeführt und im Rahmen eines methodischen Vergleichs dargestellt. Dabei wird neben einer Beschreibung der Datensätze sowie deren Aufbereitung in Abschnitt 4.3 eine Modifikation des *Point Distribution Model* auf Basis der *Independent Component Analysis* eingeführt. Anschließend werden in Abschnitt 4.4 die Ergebnisse des computationalen Vergleichs sowie des Vergleichs der Struktur der Variabilität hinsichtlich des `FastICA`, `InfoMax` sowie `JADE` Algorithmus dargestellt.

Kapitel 2

Statistical Shape Analysis

Im Rahmen dieses Kapitels wird das Feld der *Statistical Shape Analysis* betrachtet, wobei der Fokus dieses Kapitels im Rahmen der vorliegenden Arbeit insbesondere auf der Procrustes-Analyse liegt. Dazu werden zuerst die Begriffe der *Shape*, des *Landmarks* sowie des *Shape Space* definiert, bezüglich derer die Methoden der Procrustes-Analyse eingeführt werden. Im Rahmen der Procrustes-Analyse wird dabei insbesondere das *Point Distribution Model* zur Betrachtung der Variabilität von *Shapes* vorgestellt.

2.1 Shape

Der Begriff der *Shape* oder Form ist im Rahmen der Alltagssprache sehr gebräuchlich und bezieht sich für gewöhnlich auf die Erscheinung eines Objektes. Um im Weiteren detailiert die Thematik der *Statistical Shape Analysis* zu betrachten, ist jedoch eine Definition einer *Shape* im statistischen Sinne erforderlich. Diese ist dabei wiefolgt definiert:

> Als *Shape* wird die Gesamtheit der geometrischen Informationen bezeichnet, die nach Entfernung von Skalierungs-, Translations- und Rotationseffekten eines Objektes verbleiben (Dryden & Mardia 2016, S. 1).

Somit ist die *Shape* eines Objektes invariant gegenüber den euklidischen Transformationen in Form von Translation, Skalierung und Rotation; so besitzen zwei Objekte die selbe *Shape*, wenn diese durch Anwendung eben jener Transformationen übereinstimmen (ebd., S. 1f.). Sei im Folgenden $\mathbf{X} \, \epsilon \, \mathbb{R}^{k \times m}$ eine *Shape*.
Eine Translation einer *Shape* $\mathbf{X}$ ergibt sich durch die Addition einer Matrix $\mathbf{1}_k \gamma$ mit $\gamma \, \epsilon \, \mathbb{R}^m$

konstant (Dryden & Mardia 2016, S. 63), sodass die Translation der *Shape* gegeben ist durch

$$\mathbf{X}_{translated} = \mathbf{X} + \mathbf{1}_k \gamma^T. \tag{2.1}$$

Eine Rotation einer *Shape* $\mathbf{X}$ ist gegeben durch die Multiplikation mit einer $m \times m$ Rotationsmatrix Γ, wobei diese die Bedingungen $\Gamma^T\Gamma = \Gamma\Gamma^T = I_m$ und $\det(\Gamma) = 1$ erfüllt (. ebd., S. 61). Die Rotation einer *Shape* ist somit gegeben durch

$$\mathbf{X}_{rotated} = \Gamma\mathbf{X} \tag{2.2}$$

mit

$$\begin{aligned} \Gamma^T\Gamma = \Gamma\Gamma^T = I_m, \\ \det(\Gamma) = 1, \end{aligned}$$

wobei $\mathbf{X} \, \epsilon \, \mathbb{R}^{k\times m}$ und $\mathbf{\Gamma} \, \epsilon \, \mathbb{R}^{m\times m}$. Für $m = 2$ kann die Rotationsmatrix $\mathbf{\Gamma}$ durch einen Winkel θ mit $-\pi \leq \theta \leq \pi$ parametrisiert werden, so gilt für die Rotation einer *Shape* $\mathbf{x} \, \epsilon \, \mathbb{R}^{k\times m}$ im Uhrzeigersinn bezüglich des Ursprungs

$$\mathbf{X}_{rotated} = \mathbf{X}\Gamma = \mathbf{X} \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}.$$

Eine Skalierung einer *Shape* $\mathbf{X}$ ergibt sich durch die Multiplikation mit einem Skalar β

$$\mathbf{X}_{scaled} = \beta\mathbf{X} \tag{2.3}$$

mit $\beta \, \epsilon \, \mathbb{R}^+$ und $\mathbf{X} \, \epsilon \, \mathbb{R}^{k\times m}$ (ebd., S. 62).

Diese Transformationen lassen sich als euklidische Transformationen zusammenfassen, die hinsichtlich einer *Shape* $\mathbf{X} \, \epsilon \, \mathbb{R}^{k\times m}$ wiefolgt definiert sind

$$\left\{\beta\mathbf{\Gamma}\mathbf{x} + \gamma : \beta \, \epsilon \, \mathbb{R}^+, \mathbf{\Gamma} \, \epsilon \, SO(m), \gamma \, \epsilon \, \mathbb{R}^m\right\}, \tag{2.4}$$

wobei $SO(m)$ die Menge aller $m \times m$ Rotationsmatrizen bezeichnet (ebd., S. 61f.). Eine Möglichkeit zur Beschreibung von *Shapes* erfolgt durch die Platzierung von Referenzunkten auf den Konturen jener *Shapes*, was im Folgenden zum Konzept der *Landmarks* führt.

2.2 Landmark

Im Rahmen der vorliegenden Arbeit wird eine *Shape* durch eine endliche Anzahl an Punkten beschrieben, die als *Landmarks* bezeichent werden.

> Ein *Landmark* ist ein Korrespondenzpunkt auf jedem Objekt, der zwischen Populationen und innerhalb von Populationen übereinstimmt (ebd., S. 3).

Dabei werden drei verschieden Arten von *Landmarks* unterschieden:

1. *wissenschaftlicher Landmark*
   durch Experten gesetzter, wissenschaftlich fundierter Landmark,

2. *mathematischer Landmark*
   aufgrund mathematischer oder geometrischer Eigenschaft,

3. *Pseudo-Landmark:*
   auf der Kontur einer *Shape* oder zwischen *wissenschaftlichen Landmarks* und /oder *mathematischen Landmarks* liegend.

Durch diese Art der Betrachtung auf Basis von *Landmarks* ist lediglich die Charakterisierung der Grenzen einer Form und keine Beschreibung jener inneren Struktur möglich. In Abbildung 2.1 sind 9 *Landmarks* zur geometrischen Beschreibung des Corpus Callosums dargestellt.

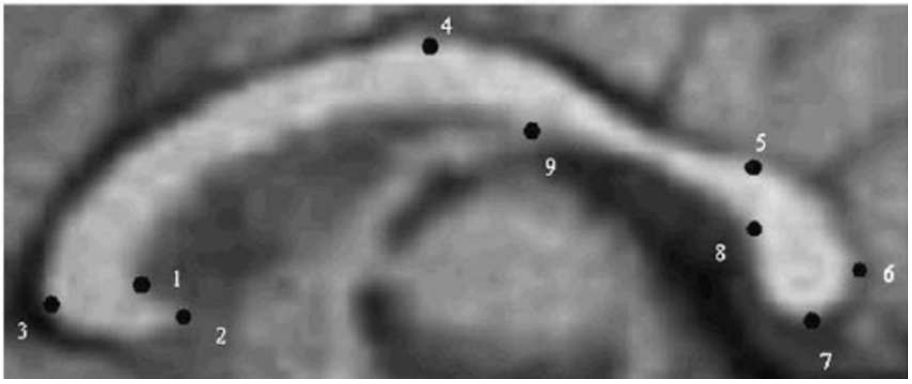


Abbildung 2.1: *Landmarks* auf der Kontur des Corpus Callosums

Quelle: He *u. a.* 2010

In Abbildung 2.2 sind die *Landmarks* zweier Individuen vor der *mittleren Shape* des Corpus Callosums einer Population dargestellt.

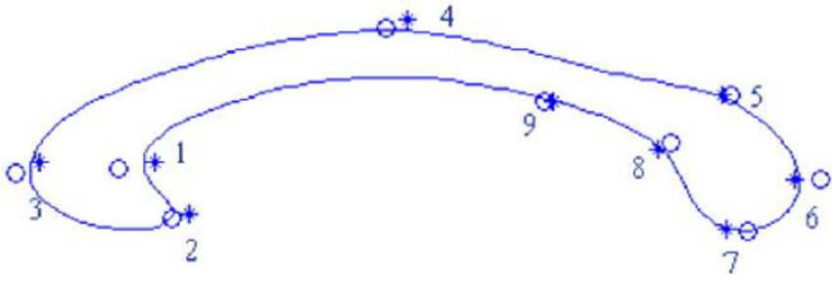


Abbildung 2.2: Vergleich der *Landmark*-Positionen des Corpus Callosums zweier Gruppen

Quelle: He *u. a.* 2010

Dabei wird deutlich, dass *Landmarks* Korrespondenzpunkte zur Zusammenfassung geometrischer Informationen innerhalb einer Population sind anhand derer ein Vergleich der ein Vergleich von *Shapes* möglich ist. Eine mathematische Repräsentation einer *Shape* mit k *Landmarks* in m Dimensionen kann in einem $k \times m$ Konfigurationsmatrix dargestellt werden (T. F. Cootes *u. a.* 1992, S. 267). Eine Betrachtung der Koordinaten der *Landmarks* führt zu dem sog. *Point Distribution Model* (T. Cootes *u. a.* 1995, S. 38), das in Abschnitt 2.4 eingeführt wird.

2.3 Shape Space

Bevor jedoch das *Point Distribution Model* eingeführt werden kann, ist zuvor eine Definition des *Shape Space* erfroderlich. Dieser ist wiefolgt definiert:

> Als *Shape Space* wird der Raum bezeichnet, der die Gesamtheit aller *Shapes* enthält. Der *Shape Space* $\sum_m^k$ ist dabei jener Raum der nicht-zufälligen k Punktmengen im $\mathbb{R}^m$ unter Verwendung der euklidischen Transformationen (Translation, Rotation, Skalierung) (Dryden & Mardia 2016, S. 65).

Die Verwendung dieser euklidischen Transformationen, die anschließend im Rahmen der Procrustes-Analyse betrachtet wird, bringt die Menge der *Shapes* somit in den definierten *Shape Space*. Die Dimension dieses Raumes ist dabei gegeben durch

$$q = kn - n - 1 - \frac{n(n-1)}{2}. \tag{2.5}$$

Diese ergibt sich durch die kn Dimensionen der ursprünglichen k Punktmengen im $\mathbb{R}^m$ abzüglich der m Dimensionen betreffend die Translation, einer Dimension für die Skalierung

sowie $\frac{1}{2}n(n-1)$ Dimensionen für die Rotation der Shapes. Die Anpassung reduziert somit die Dimension, sodass der *Shape Space* einen Unterraum des ursprünglichen Konfigurationsraumes[1] mit Dimension km darstellt (ebd., S. 65).Zur Definition eines *Shape Space* ist ferner die Definition eines Distanzmaßes erforderlich. Kann eine mögliche Beziehung zwischen der Distanz im *Shape Space* und der Euklidischen Distanz im ursprünglichen Raum hergestellt werden, so bildet die Menge der *Shapes* eine Riemann Manigfaltigkeit, die die zu betrachtende Objektklasse enthält (Bookstein 1996, S. 227).[2] Häufig verwendete Metriken umfassen dabei die Haussforff-Metrik sowie die Procrustes-Distanz, wobei letztere in den weiteren Ausführungen verwendet wird.

2.4 Procrustes-Analyse

Nachdem eingangs eine *Shape* als Gesamtheit der geometrischen Informationen ohne jegliche Rotations-, Translations- und Skalierungseffekte definiert wurde, werden im Rahmen der Procrustes-Analyse eben jene euklidischen Transformationen zur Schaffung einheitlicher Repräsentationen betrachtet.Dabei st die Ausgangsproblematik der Procrustes-Analyse graphisch in Abbildung 2.3 dargestellt, die zwei *Shapes* sowohl vor als auch nach der Anwendung der Procrustes-Methoden zur Angleichung der *Shapes* zeigt. Die verwende-

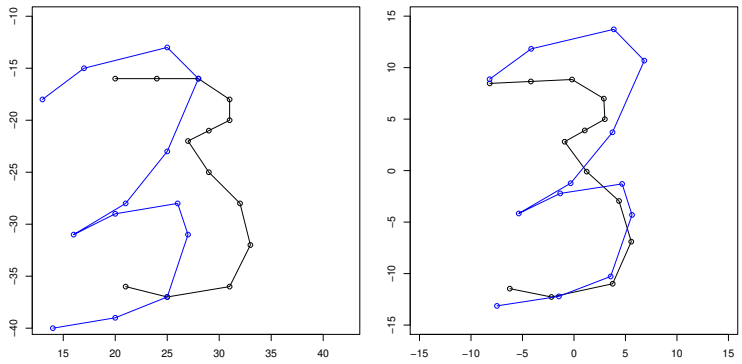


Abbildung 2.3: Zwei *Shapes* vor *(links)* und nach *(rechts)* der Procrustes-Analyse

ten Methoden erweisen sich dabei als hilfreiches Mittel insbesondere zur Bestimmung einer *mittleren Shape* sowie der Variabilität von *Shapes* einer gewissen Menge (Dryden & Mardia 2016, S. 125). Die Procustes-Analyse wird hinsichtlich der der Anzahl der zu betrachtenden *Shapes* unterschieden in die gewöhnliche Procrustes-Analyse (OPA) zur Angleichung

[1]Der Konfigurationsraum ist der Raum aller *Landmark* Koordinaten (Dryden & Mardia 2016, S. 33).
[2]Dieser Raum wird auch als *Kendall Space* bezeichnet.

zweier Konfigurationen sowie der generalisierten Procrustes-Analyse (GPA) insbesondere zur Bestimmung einer *mittleren Shape* (Dryden & Mardia 2016, S. 125).[3] Die Basis einer solchen Procrustes-Analyse bildet die (volle) Procrustes-Distanz, die zwischen zwei Konfigurationen $\mathbf{X}_1$ und $\mathbf{X}_2$ ist wiefolgt definiert ist (ebd., S. 71)

$$d_F(\mathbf{X}_1, \mathbf{X}_2) = \inf_{\beta\mathbf{\Gamma}} \|\mathbf{Z}_2 - \beta\mathbf{Z}_1\mathbf{\Gamma}\|, \tag{2.6}$$

wobei $\mathbf{Z}_r = \frac{\mathbf{HX}_r}{\|\mathbf{HX}_r\|}$ mit $r = 1, 2$ und $\mathbf{H}$ Helmert Submatrix[4] sowie $\|\cdot\|$ die euklidische Norm ist (ebd., S. 71).[5] Die (volle) Procrustes-Distanz ist somit als minimale Euklidische Norm zwischen zwei Konfigurationen unter Verwendung von Skalierung, Translation sowie Rotation definiert.

Zur Angleichung der *Shapes* seien dazu $\mathbf{X_1}, ..., \mathbf{X_n}$ Konfigurationsmatrizen mit $n \geq 2$ aus einer Population $[\mu]$ gegeben. Dann umfasst die generalisierte Procrustes-Analyse Translation, Rotation sowie Skalierung der Konfigurationsmatrizen relativ zueinander zur Minimierung der Quadratsumme der Procrustes-Distanzen zur *mittleren Shape* im Rahmen eines iterativen Verfahrens

$$G(\mathbf{X_1}, ..., \mathbf{X_n}) = \sum_{i=1}^{n} \|\beta_i \mathbf{X}_i \mathbf{\Gamma}_i + \mathbf{1}_k \gamma_i^T) - \mu\|^2 \tag{2.7}$$

bezüglich $\beta_i, \mathbf{\Gamma}_i, \gamma_i$ mit $i = 1, ..., n$ und μ unter einer allgemeinen Beschränkung durch bspw.

$$\sum_{i=1}^{n} S^2(\beta_i \mathbf{X}_i \mathbf{\Gamma}_i + \mathbf{1}_k \gamma_i^T) = \sum_{i=1}^{n} S^2(\mathbf{X}_i) \tag{2.8}$$

mit

$$S(\mathbf{X}_i) = \sqrt{\sum_{j=1}^{k} \|(\mathbf{X}_i)_j - \bar{\mathbf{X}}_i\|^2} \tag{2.9}$$

der Distanz zum Schwerpunkt der *Shape (Centroid Size)*, wobei $\|\cdot\|$ die Euklidische Norm ist (ebd., S. 134).

[3] Im Folgenden wird dabei ausschließlich die generalisierte Procrustes-Analyse betrachtet.

[4] Die Helmert Submatrix wird zum Entfernen der Translations- sowie der Skalierungseffekte benötigt. Ebenso die Verwendung einer Zentrierungsmatrix $\mathbf{C}$ anstelle der Helmert Submatrix möglich.

[5] $\mathbf{Z}$ wird aufgrund der Entfernung von Translations- und Skalierungseffekten auch als *Pre-Shape* bezeichnet.

Dadurch ergeben sich für die Konfigurationsmatrizen die folgenden Procrustes-Koordinaten

$$\mathbf{X}_i^P = \hat{\beta}_i \mathbf{X}_i \hat{\mathbf{\Gamma}}_i + \mathbf{1}_i \hat{\gamma}_i^T, \tag{2.10}$$

wobei $\hat{\mathbf{\Gamma}}_i \, \epsilon \, SO(m)$ Rotationsmatrix, $\hat{\beta}_i > 0$ Skalierungsparameter, $\hat{\gamma}_i^T$ Lokationsparameter mit $i = 1, ..., n$ (ebd., S. 135).

Als Schätzer für μ ergibt sich im Rahmen der Iterationsschritte (ebd., S. 135)

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{X}_i^P. \tag{2.11}$$

Somit ist die Schätzung der Transformationsparameter im Rahmen der generalisierten Procrustes-Analyse äquivalent zur Minimierung von

$$\begin{aligned} G(\mathbf{X}_1, ..., \mathbf{X}_n) &= \inf_{\beta_i \mathbf{\Gamma}_i, \gamma_i} \sum_{i=1}^{n} \left\| (\beta_i \mathbf{X}_i \mathbf{\Gamma}_i + \mathbf{1}_k \gamma_i^T) - \frac{1}{n} \sum_{j=1}^{n} (\hat{\beta}_j \mathbf{X}_j \hat{\mathbf{\Gamma}}_j + \mathbf{1}_k \hat{\gamma}_j^T) \right\|^2 \\ &= \inf_{\beta_i \mathbf{\Gamma}_i, \gamma_i} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left\| (\beta_i \mathbf{X}_i \mathbf{\Gamma}_i + \mathbf{1}_k \gamma_i^T) - (\hat{\beta}_j \mathbf{X}_j \hat{\mathbf{\Gamma}}_j + \mathbf{1}_k \hat{\gamma}_j^T) \right\|^2. \end{aligned} \tag{2.12}$$

Die *mittlere Shape* ergibt sich nach der iterativen Bestimmung der Procrustes-Koordinaten durch (ebd., S. 135f.)

$$\bar{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{X}_i^P. \tag{2.13}$$

Neben der *mittleren Shape* ist im Rahmen der Procrustes-Analyse die Struktur der Variationen der *Shapes* von Interesse, die im Rahmen eines *Point Distribution Model* betrachtet werden kann, das wiefolgt definiert ist:

> Ein *Point Distribution Model* ist eine Repräsentation einer Klasse von *Shapes* unter Verwendung eines flexiblen Modells zur Erfassung der Koordinaten der *Landmarks* (T. F. Cootes *u. a.* 1992, S. 10-14).

Ein *Point Distribution Model* generiert somit ein statistisches Modell einer *Shape* und Variationen dieser *Shape* auf Basis einer Trainingsmenge. Sei dazu die Konfigurationsmatrix

$\mathbf{X}_i \epsilon \mathbb{R}^{k\times 2}$ als Vektor formuliert, sodass

$$\mathbf{x}_i^P = \text{vec}(\mathbf{X}_i^P) = (x_{i1}^P, ..., x_{ik}^P, y_{i1}^P, ..., y_{ik}^P)^T \epsilon \mathbb{R}^{2k},$$

dann folgt für die *mittlere Shape* mit $i = 1, ..., n$ aus Formel (2.13)

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^P.$$

Des Weiteren ist zur Anpassung eines *Point Distribution Model* eine Singulärwertzerlegung der Kovarianzmatrix der Procrustes-Residuen[6] im Rahmen einer *Principal Component Analysis* erforderlich; diese ist gegeben durch

$$\Sigma = Cov(\mathbf{x}^P) = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i^P - \bar{\mathbf{x}})(\mathbf{x}_i^P - \bar{\mathbf{x}})^T. \tag{2.14}$$

Sei nun $\mathbf{D}$ eine $2k \times n$ Matrix mit

$$\mathbf{D} = \left(\mathbf{x}_1^P, ..., \mathbf{x}_n^P\right), \tag{2.15}$$

dann gilt für $\mathbf{\Sigma}$

$$\Sigma = \frac{1}{n}\mathbf{D}\mathbf{D}^T \tag{2.16}$$

Sei zusätzlich $\mathbf{T}$ eine $n \times n$ Matrix, die wiefolgt definiert ist

$$\mathbf{T} = \frac{1}{n}\mathbf{D}^T\mathbf{D}, \tag{2.17}$$

und sei $\mathbf{e}_i$ ein Eigenvektor von $\mathbf{T}$ zum Eigenwert γ_i, dann gilt

$$\mathbf{T}\mathbf{e}_i = \gamma_i\mathbf{e}_i. \tag{2.18}$$

Daraus folgt

$$\frac{1}{n}\mathbf{D}^T\mathbf{D}\mathbf{e}_i = \gamma_i\mathbf{e}_i, \tag{2.19}$$

[6]Procrustes-Residuen bezeichnen die Residuen zwischen einer angepassten *Shape* und der *mittleren Shape* $d\mathbf{X} = \mathbf{X}^P - \bar{\mathbf{X}}$.

multipliziert mit $\mathbf{D}$

$$\frac{1}{n}\mathbf{D}\mathbf{D}^T\mathbf{D}\mathbf{e}_i = \gamma_i\mathbf{D}\mathbf{e}_i \tag{2.20}$$

$$\mathbf{\Sigma}\mathbf{D}\mathbf{e}_i = \gamma_i\mathbf{D}\mathbf{e}_i. \tag{2.21}$$

Dies zeigt, dass $\mathbf{D}\mathbf{e}_i$ Eigenvektor von $\mathbf{\Sigma}\mathbf{D}$ zum Eigenwert γ_i ist. Für den normierten Eigenvektor $\mathbf{p}_i$ von $\mathbf{\Sigma}$ ergibt sich

$$\mathbf{p}_i = \frac{1}{\gamma_i n}\mathbf{D}\mathbf{e}_i \tag{2.22}$$

mit $\mathbf{p}_i\mathbf{p}_i^T = 1$ zum Eigenwert $\lambda_i = \gamma_i$ (T. Cootes *u. a.* 1995, S. 58).

Nach *Cootes/Taylor* (1992) ist eine *Shape* im Rahmen eines *Point Distribution Model* definiert als

$$\mathbf{x}^P = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b} \tag{2.23}$$

mit

$$\begin{aligned}\mathbf{x}^P &= (x_1^P, ..., x_n^P, y_1^P, ..., y_n^P)^T,\\ \mathbf{P} &= (\mathbf{p}_1, ..., \mathbf{p}_t),\\ \mathbf{b} &= (b_1, ..., b_n)^T,\end{aligned}$$

wobei (x_i^P, y_i^P) die Procrustes-Koordinaten der *Landmarks*, $\bar{\mathbf{x}}$ die *mittlere Shape* analog zu Formel (2.13), $\mathbf{P}$ die Matrix der ersten t Eigenvektoren $\mathbf{p}_i$ und $\mathbf{b}$ der Vektor der Gewichte b_i ist. Dabei sind die Spalten der Matrix $\mathbf{P}$ orthogonal, sodass

$$\mathbf{P}^T\mathbf{P} = \mathbf{I},$$

und

$$\mathbf{b} = \mathbf{P}^T(\mathbf{x} - \bar{\mathbf{x}}).$$

Auf Basis von Formel (2.23) anschließend eine Generierung neuer *Shapes* durch Variation des Parameters b_i in gewissen Grenzen möglich (T. F. Cootes *u. a.* 1992, S. 268). Typische

Grenzen für b_i sind dabei

$$-3\sqrt{\lambda_i} \leq b_i \leq 3\sqrt{\lambda_i}.$$

Nachdem in diesem KapiteL theoretische Konzepte der *Statistical Shape Analysis* betrachtet wurden, werden im folgenden Kapitel die theoretischen Grundlagen der *Independent Component Analysis* behandelt.

Kapitel 3

Independet Component Analysis

Die *Independent Component Analysis (ICA)* ist als eine Erweiterung der *Principal Component Analysis (PCA)* anzusehen, in Rahmen derer stochastisch unabhängige und nicht-normalverteilte Komponenten zur Repräsentation multivariater Daten gesucht werden (Meyer-Baese & Schmid 2014, S. 264). In diesem Abschnitt werden neben einer eingehenden Definition der *Independent Component Analysis* und die in deren Zusammenhang wichtigen Konzepte der stochastischen Unabhängigkeit sowie der Nicht-Normalität insbesondere die wichtigsten Algorithmen der ICA – `FastICA`,`InfoMax` und `JADE` – dargestellt.

3.1 Definition

geeignete Repräsentaion multivariater Daten Sei x_i eine Zufallsvariable mit $i = 1, ..., n$, die als Linearkombination von n stochastisch unabhängigen Zufallsvariablen s_j dargestellt werden kann

$$x_i = a_{i1}s_1 + a_{i2}s_2 + ... + a_{in}s_n = \sum_{j=1}^{n} a_{ij}s_j, \tag{3.1}$$

wobei a_{ij} ein Mischungskoeffizient ist. In Matrixform ergibt sich

$$\mathbf{x} = \mathbf{A}\mathbf{s}, \tag{3.2}$$

mit $\mathbf{x} = (x_1, ..., x_n)^T$, $\mathbf{A} = (a_{ij})$ und $\mathbf{s} = (s_1, ..., s_n)^T$ für alle $i, j = 1, ..., n$. Dieses generative Modell stellt damit das Basismodell der *Independent Component Analysis* dar. Die Ausgangsproblematik im Kontext einer *Independent Component Analysis* lautet dabei

wiefolgt: Gegeben der direkt beobachtbaren Realisierungen von x_i müssen sowohl die nicht-beobachtbaren unabhängigen Komponenten s_j als auch die unbekannten Mischungskoeffizienten a_{ij} geschätzt werden (Comon 1994, S. 288). Die Identifizierbarkeit dieses Modells erfordert zu diesem Zweck folgende Annahmen (ebd., S. 294):

1. Die Mischungsmatrix **A** ist konstant und regulär.

2. Die Komponenten s_j sind für alle j stochastisch unabhängig.

3. Die Komponenten s_j sind nicht-normalverteilt.[1]

Die Annahmen bzgl. der Mischungsmatrix **A** sind dabei nicht zwingend erforderlich, führen jedoch zu theoretischen Vereinfachungen. Die fundamentale Veränderung im Vergleich zu konventionellen multivariaten Methoden ist hier die Annahme nicht-normalverteilter Komponenten, die im Weiteren noch detailiert betrachtet wird.
Die theoretischen Grundlagen der *Independent Component Analysis* zur Bestimmung der unbekannten Mischungsmatrix **A** und der unbeobachtbaren unabhängigenen Komponenten s_j basieren auf dem Zentralen Grenzwertsatz, demzufolge tendiert die Verteilung der Summe unabhängiger und identisch verteilter Zufallsvariablen zunehmend zu einer Normalverteilung

$$\sum_{i=1}^{n} x_i \overset{a}{\sim} N(n\mu, n\sigma^2) \tag{3.3}$$

mit

$$E(x_i) = \mu,$$
$$Var(x_i) = \sigma^2.$$

Somit ist die Verteilung der Summe unabhängiger Zufallsvariablen $\sum_i x_i$ Gauß-ähnlicher als die Verteilungen der einzelnen Zufallsvariablen x_i (Hyvarinen & Oja 2000, S. 416). Zur Illustration wird dabei vereinfachend angenommen, dass die unabhängigen Komponenten s_j allesamt identisch verteilt sind und der Vektor **x** dem Datenmodell in Formel (3.1) entstammt. Um eine unabhängige Komponente zu schätzen, wird eine Linearkombination

[1] Höchstens eine der Komponenten s_j darf normalverteilt sein.

der x_i betrachtet, sodass

$$\mathbf{y} = \mathbf{w}^T\mathbf{x} = \sum_i w_i x_i, \tag{3.4}$$

wobei der Vektor $\mathbf{w}$ zu bestimmen ist. Entspricht der Vektor $\mathbf{w}$ nun einer der Reihen von $\mathbf{A}^{-1}$, so ist diese Linearkombination gleich einer der unabhängigen Komponenten s_j. Im Rahmen der Bestimmung des Vektors $\mathbf{w}$ kann nun der *Zentrale Grenzwertsatz* genutzt werden, sodass $\mathbf{w}$ einer Zeile der Inversen $\mathbf{A}^{-1}$ entspricht. Aufgrund der fehlenden Kenntnis über die Matrix $\mathbf{A}$ kann der Vektor $\mathbf{w}$ nicht direkt determiniert werden, sondern muss durch eine Schätzung bestimmt werden (ebd., S. 416). Um dies zu verdeutlichen gelte im Folgenden $\mathbf{z} = \mathbf{A}\mathbf{w}$, sodass

$$\mathbf{y} = \mathbf{w}^T\mathbf{x} = \mathbf{w}^T\mathbf{A}\mathbf{s} = \mathbf{z}^T\mathbf{s}. \tag{3.5}$$

Somit ist y_i eine Linearkombination der unabhängigen Komponenten s_j und Gewichtungen z_j. Da gemäß des Zentralen Grenzwertsatzes die Verteilung der Summe unabhängiger und identisch verteilter Zufallsvariablen ${}_sum_j s_j$ Gauß-ähnlicher ist als die Verteilungen der s_j, besitzt diese im Falle der Identität zu einer Verteilung einer Komponente s_j die geringste Gauß-Ähnlichkeit, somit gilt lediglich für ein z_j des Vektors $\mathbf{z}$, dass $z_j \neq 0$. Demzufolge wird der Vektor $\mathbf{w}$ gesucht, der die Nicht-Gauß-Ähnlichkeit von $\mathbf{w}^T\mathbf{x}$ maximiert und somit jenem Vektor $\mathbf{z}$ entspricht, sodass

$$\mathbf{w}^T\mathbf{x} = \mathbf{z}^T\mathbf{x} \tag{3.6}$$

und damit gleich einer der unabhängigen Komponenten s_j ist (ebd., S. 416f.). Die Maximierung der Nicht-Gauß-Ähnlichkeit von $\mathbf{w}^T\mathbf{x}$ führt somit zur Bestimmung einer unabhängigen Komponente, wohingegen die Optimierung im n-dimensionalen Raum der Vektoren $\mathbf{w}$ zu $2n$ lokalen Maxima, s_j und $-s_j$ für jede unabhängige Komponente, führt.[2]

Das Ziel der noch zu betrachtenden Algorithmen im Rahmen der *Independent Component Analysis* ist die Ermittlung eben jener Gewichtungsmatrix $\mathbf{W}$ mit

$$\mathbf{W} = \mathbf{A}^{-1}, \tag{3.7}$$

sodass die Komponenten $\mathbf{s} = \mathbf{W}\mathbf{x}$ möglichst unabhängig sind (Hyvärinen, Karhunen *u. a.*

[2]Das Vorzeichen der unabhängigen Komponente verändert das Modell nicht (siehe Abschnitt 3.7).

2001, S. 7). Jene Gewichtungsmatrix **W** kann dabei jedoch nicht in geschlossener Form dargestellt werden, sodass Kostenfunktionen, die im späteren als *Zielfunktionen* bzw. *Kontrastfunktionen* bezeichnet werden, benötigt werden, deren Minima bzw. Maxima die Lösungen **W** im Rahmen der *Independent Component Analysis* liefern.
Die Verwendung der *Independent Component Analysis* eignet sich insbesondere im Rahmen der *Blind Source Separation (BSS)* sowie der *Feature-Extraktion* (Hyvarinen & Oja 2000, S. 411). Ein klassisches Beispiel der *Blind Source Separation* stellt das *Cocktail-Party Problem* dar, das in der folgenden Abbildung graphisch dargestellt ist.

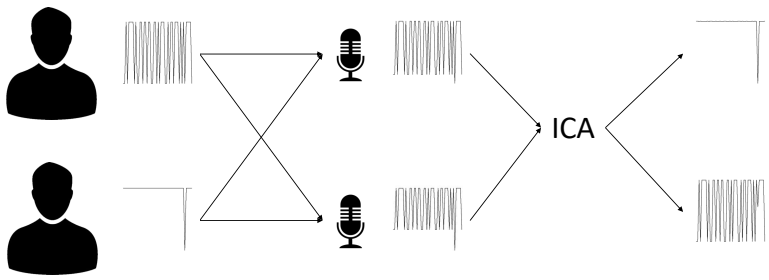


Abbildung 3.1: Cocktail-Party Problem

Während einer Cocktail-Party führen zwei im gelichen Raum befindliche Personen eine Konversation. Zusätzlich sind zwei Mikrophone an unterschiedlichen Orten in diesem Raum platziert, welche die Konversation der beiden Personen aufzeichnen. Die Mikrophone können dabei die einzelnen Signalquellen nicht separiert sondern lediglich die Mischungen x_1, x_2 jener Quellen aufnehmen, sodass sowohl die zwei Sprechsignale s_1, s_2 als auch die Mischungskoeffizienten a_{ij} mit $i, j = 1, 2$ unbekannt sind. Auf Basis des *Independent Component Analysis* können die Mischungskoeffizienten a_{ij} unter Verwendung der dargestellten Annahmen geschätzt werden und somit die Signale s_1, s_2 von deren Mischungen x_1, x_2 separiert werden (Hyvärinen, Karhunen *u. a.* 2001, S. 147-150).
Im Rahmen der Feature-Extraktion wird auf Basis der *Independent Component Analysis* eine geeignete Repräsentation mit möglichst unabhängigen Features beispielsweise von Bildern oder Audiodateien mit dem Ziel der Kompression oder des Entrauschens jener Dateien gesucht (Hyvarinen & Oja 2000, S. 412).

3.2 Vergleich zwischen ICA und PCA

Zu einer Abgrenzung der Ansätze seien an dieser Stelle die Mechanismen der *Independent Component Analysis* und der *Principal Component Analysis* gegenübergestellt.
Obwohl das Ziel einer geeigneten Repräsentation multivariater Daten beider Methoden gleich und somit die Form des Datenmodells im Rahmen der *Principal Component Analysis* analog zu jener der *Independent Component Analysis* (siehe Formel (3.1)) ist, unterscheiden sich die Methoden hinsichtlich der Ermittlung der Komponenten s_j bzw. deren Eigenschaften(Meyer-Baese & Schmid 2014, S. 266). So verwendet die *Principal Component Analysis* als Maß der Redundanz die Korrelation der Komponenten s_j, wohingegen die *Independent Component Analysis* das stärkere Kriterium der stochastischen Unabhängigkeit nutzt (Hyvärinen, Karhunen *u. a.* 2001, S. 125).
Im Rahmen eines Beispiels zur Illustration werden im Folgenden zwei stochastisch unabhängige Komponenten s_1, s_2 betrachtet, die einer Gleichverteilung mit folgender Dichtefunktion folgen

$$f(s_i) = \begin{cases} \frac{1}{10} & \text{falls } |s_i| \leq 5 \quad \text{mit } i = 1, 2, \\ 0 & \text{sonst.} \end{cases} \tag{3.8}$$

Die bivariate Verteilung der Zufallsvariablen s_1, s_2 ist dabei im folgenden Streudiagramm dargestellt.

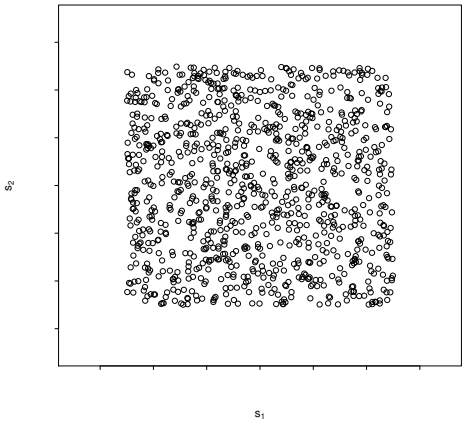


Abbildung 3.2: Streudiagramm der Realisierungen zweier gleichverteilter Zufallsvariablen

Der Vektor der unabhängigen Komponenten $\mathbf{s}$ wird anschließend mit einer Mischungsmatrix

A multipliziert

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{3.9}$$

mit

$$\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}, \tag{3.10}$$

sodass **x** der Vektor der Mischungen x_1, x_2 ist. Die gemeinsame Verteilung der Mischungen x_1, x_2 ist im folgenden Streudiagramm dargestellt.

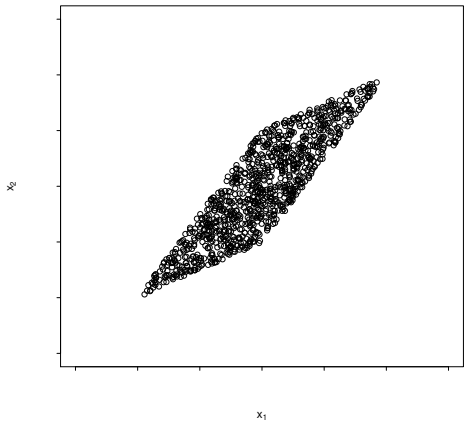


Abbildung 3.3: Streudiagramm der beobachtbaren Mischungen

Die resultierende gemeinsame Verteilung der Mischungen x_1, x_2 ist wiederum eine Gleichverteilung, jedoch sind die Mischungen x_1, x_2 nicht mehr stochastisch unabhängig.
Mithilfe der *Principal Component Analysis* sowie der *Independent Component Analysis* soll nun eine geeignete Repräsentation der Daten gefunden werden. Dabei wird im Rahmen der *Principal Component Analysis* der Vektor **x** linear transformiert, sodass die Redundanzen auf Basis der Korrelation durch eine Rotation des Koordinatensystems nicht weiter existent sind. Im Rahmen der *Independent Component Analysis* werden hingegen die Mischungen derart linear transformiert,sodass die Komponenten möglichst stochastisch unabhängig sind. Die Dekomposition der Mischungen sind dabei in der folgenden Abbildung dargestellt. Die Abbildung zeigt, dass die *Principal Component Analysis* die Komponenten durch Beschränkung der Matrix **W** auf eine orthogonale Matrix dekorreliert, diese jedoch nicht stochastisch unabhängig sind; wohingegen die *Independent Component Analysis* unter die Ver-

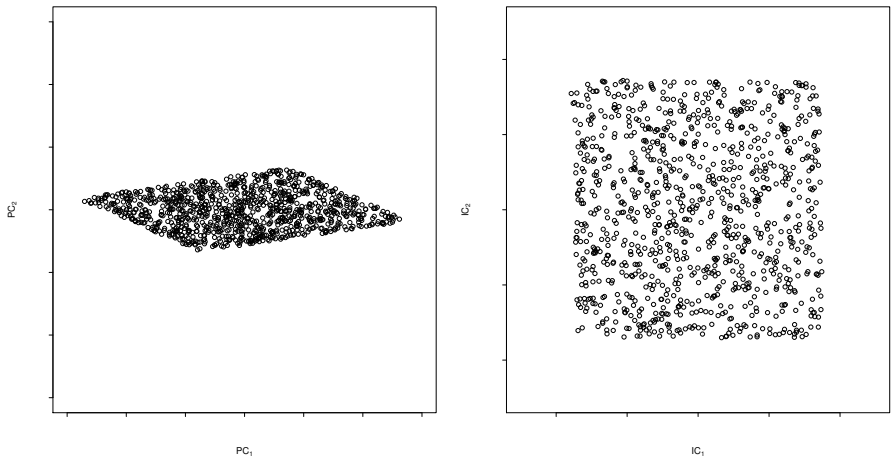


Abbildung 3.4: Dekomposition der Mischungen anhand PCA (*links*) und ICA (*rechts*)

wendung Statistiken höherer Ordnung bzw. informationstheoretischer Kriterien die Komponenten separiert, wobei die Matrix $\mathbf{W}$ zumeist nicht orthogonal ist (Meyer-Baese & Schmid 2014, S. 266).

3.3 Vorverarbeitung

Zur Vereinfachung der Schätzung im Rahmen der *Independent Component Analysis* ist eine Aufbereitung der Daten durchaus nützlich (ebd., S. 268.). So unterteilt eine Vielzahl von ICA Algorithmen die Schätzung des Modells in zwei Schritte: Vorverarbeitung und Modellschätzung. Bei jenen Vorverarbeitungsschritten, die im Folgenden dargestellt werden, handelt es sich um die Zentrierung und das *Whitening* der Daten.

3.3.1 Zentrierung

Die grundlegendste und notwendigste Form der Aufbereitung ist die Zentrierung der Mischung $\mathbf{x}$, sodass $E(\mathbf{x}^*) = \mathbf{0}$ mit $\mathbf{x}^* = \mathbf{x} - E(\mathbf{X})$, dann gilt

$$E(\mathbf{x}^*) = E(\mathbf{A}\mathbf{s}^*) = \mathbf{A}E(\mathbf{s}^*) = 0 \quad \Rightarrow \quad E(\mathbf{s}^*) = \mathbf{0} \text{ für } \mathbf{A} \neq \mathbf{0}. \tag{3.11}$$

Die Zentrierung der Daten wird dabei lediglich zur Vereinfachung des *Independent Com-*

ponent Analysis Algorithmus vorgenommen (Hyvarinen & Oja 2000, S. 421).[3]

3.3.2 Whitening

Im Anschluss an jene Zentrierung wird im Rahmen der *Independent Component Analysis* oftmals ein *Whitening* der beobachteten Daten durchgeführt. Dabei wird der Vektor $\mathbf{x}$ linear in einen Vektor $\mathbf{z}$ transformiert (Meyer-Baese & Schmid 2014, S. 268)

$$\mathbf{z} = \mathbf{V}\mathbf{x}, \tag{3.12}$$

sodass für die Kovarianzmatrix gilt

$$E(\mathbf{z}\mathbf{z}^T) = \mathbf{I}. \tag{3.13}$$

Da es sich im Rahmen des *Whitening* somit um eine Dekorrelation in Verbindung mit einer Skalierung der Daten handelt, kann dazu die Methodik der *Principal Component Analysis* verwendet werden (Hyvärinen, Karhunen *u. a.* 2001, S. 140), sodass die lineare Transformation (siehe Formel (3.12) durch eine Diagonalisierung der Kovarianzmatrix erreicht werden kann. Für die Kovarianzmatrix gilt somit

$$E(\mathbf{x}\mathbf{x}^T) = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T, \tag{3.14}$$

wobei $\mathbf{Q}$ die orthonormal Matrix der Eigenvektoren von $E(\mathbf{x}\mathbf{x}^T)$ und $\mathbf{\Lambda} = diag(\lambda_i)$ die Diagonalmatrix mit deren Eigenwerten λ_i ist (Meyer-Baese & Schmid 2014, S. 268). Für die Transformation ergibt sich dann

$$\mathbf{z} = \mathbf{Q}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{Q}^T\mathbf{x} \tag{3.15}$$

und daraus folgt für die orthogonale Mischungsmatrix $\tilde{\mathbf{A}}$

$$\mathbf{z} = \mathbf{Q}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{Q}^T\mathbf{A}\mathbf{s} = \tilde{\mathbf{A}}\mathbf{s} \tag{3.16}$$

mit

$$E(\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T) = \tilde{\mathbf{A}}\tilde{\mathbf{A}}^T = \mathbf{I}. \tag{3.17}$$

[3]Zur notationellen Vereinfachung wird im Folgenden $\mathbf{x}^*$ mit $\mathbf{x}$ bezeichnet.

Durch ein *Whitening* der Daten wird somit eine Reduzierung der Mischung auf eine Rotationsmatrix $\tilde{\mathbf{A}}$ erreicht, so müssen anstelle von n^2 Parametern lediglich $\frac{n(n-1)}{2}$ Parameter geschätzt werden (ebd., S. 268).[4] Ferner kann durch eine Vernachlässigung der kleinen Eigenwerte im Rahmen der *Principal Component Analysis* eine Dimensionsreduktion erreicht werden, die ein *Overlearnig* im Zuge der *Independent Component Analysis* verhindert (Hyvärinen, Särelä *u. a.* 1999b, S. 426).

3.4 Unabhängigkeit

Ein zentrales Konzept, das die Basis der *Independent Component Analysis* bildet, ist die stochastische Unabhängigkeit. So ist die Annahme unabhängiger Komponenten erforderlich zur Identifizierbarkeit des Modells (siehe Formel (3.1)) im Rahmen der *Independent Component Analysis* (Hyvärinen, Karhunen *u. a.* 2001, S. 27). Dabei werden zwei Zufallsvariablen s_1, s_2 als stochastisch unabhängig bezeichnet, wenn gilt

$$f_{s_1 s_2}(s_1, s_2) = f_{s_1}(s_1) f_{s_2}(s_2), \tag{3.18}$$

wobei $f_{s_1 s_2}(s_1, s_2)$ die gemeinsame Dichtefunktion von s_1 und s_2, und $f_{s_1}(s_1)$ bzw. $f_{s_2}(s_2)$ die jeweiligen Randdichten von s_1 bzw. s_2 darstellt (Meyer-Baese & Schmid 2014, S. 267). Die gemeinsame Dichtefunktion $f_{s_1 s_2}(s_1, s_2)$ unabhängiger Zufallsvariablen s_1, s_2 ist somit faktorisierbar und kann als Produkt der einzelnen Randdichten $f_{s_1}(s_1), f_{s_2}(s_2)$ dargestellt werden. Daraus ergibt sich für die Kovarianz zweier unabhängiger Zufallsvariablen s_1, s_2

$$E(g(s_1)h(s_2)) = E(g(s_1))E(h(s_2)) \tag{3.19}$$

mit $g(\cdot)$, $h(\cdot)$ beliebige absolut integrierbare Funktionen (Hyvärinen, Karhunen *u. a.* 2001, S. 27). Für $g(s_1) = s_1$ und $h(s_2) = s_2$ lineare Funktionen und somit

$$Cov(s_1, s_2) = E(s_1 s_2) = E(s_1)E(s_2) = 0, \tag{3.20}$$

heißen die Zufallsvariablen s_1, s_2 unkorreliert. Demzufolge impliziert die stochastische Unabhängigkeit zweier Zufallsvariablen s_1, s_2 auch deren Unkorreliertheit, jedoch ist die Rückrichtung i.A. nicht gültig. Eine Äquivalenz zwischen stochastischer Unabhängigkeit und

[4] Im Folgenden werden aufgrund einer vereinfachten Notation die Daten nach jener Vorverarbeitung mit $\mathbf{x}$ und die transformierte Mischungsmatrix mit $\mathbf{A}$ bezeichnet.

Unkorreliertheit liegt lediglich im Falle normalverteilter Zufallsvariablen vor (Hyvärinen, Karhunen *u. a.* 2001, S. 33). Eine Verallgemeinerung der stochastischen Unabhängigkeit analog dem Fall zweier Zufallsvariablen ist ebenso für mehrere Zufallsvariablen möglich.

3.5 Nicht-Normalität

Das zweite zentrale Konzept, das im Rahmen der *Independent Component Analysis* zur Identifizierbarkeit des Modells erforderlich ist, ist das Konzept der Nicht-Normalität der Komponenten s_j. Im Rahmen dieses Abschnitts werden daher neben der Erfordernis der Nicht-Normalität der Komponenten Maße der Nicht-Normalität im Kontext der *Independent Component Analysis* betrachtet.

3.5.1 Erfordernis der Nicht-Normalität

Wie bereits in Abschnitt 3.1 erwähnt, bedarf die *Independent Component Analysis* neben der zuvor betrachteten stochastischen Unabhängigkeit die Nicht-Normalität der Komponenten s_j; so darf höchstens eine jener Komponenten normalverteilt sein. Diese Restriktion ist begründet durch die Unmöglichkeit einer Bestimmung der Mischungsmatrix $\mathbf{A}$ im Falle normalverteilter Komponenten s_j, was anhand des folgenden Beispiels verdeutlicht werden soll.
Seien dazu s_1, s_2 unabhängig und normalverteilt, sodass die gemeinsame Dichtefunktion gegeben ist durch

$$f_{s_1 s_2}(s_1, s_2) = \frac{1}{2\pi} \exp\left(-\frac{s_1^2 + s_2^2}{2}\right) = \frac{1}{2\pi} \exp\left(-\frac{\|\mathbf{s}\|^2}{2}\right). \tag{3.21}$$

Ferner sei die Mischungsmatrix $\mathbf{A}$ eine orthogonale Matrix.[5] Aufgrund der Orthogonalität gilt $\mathbf{A}^{-1} = \mathbf{A}^T$ und somit folgt für die gemeinsame Verteilung der Mischungen x_1, x_2

$$f_{x_1 x_2}(x_1, x_2) = \frac{1}{2\pi} \exp\left(-\frac{\|\mathbf{A}^T\mathbf{x}\|^2}{2}\right) \left|\det(\mathbf{A}^T)\right|. \tag{3.22}$$

Aus der Orthogonalität von $\mathbf{A}$ folgt $\|\mathbf{A}^T\mathbf{x}\|^2 = \|\mathbf{x}\|^2$ sowie $|\det(\mathbf{A})| = 1$, und somit gilt

$$f_{x_1 x_2}(x_1, x_2) = \frac{1}{2\pi} \exp\left(-\frac{\|\mathbf{x}\|^2}{2}\right). \tag{3.23}$$

[5] Diese Annahme ist zulässig, da von einem zuvor erfolgten *Whitening* ausgegangen wird.

Die gemeinsame Dichtefunktion der Mischungen x_1, x_2 ist somit unabhängig von der orthogonalen Mischungsmatrix $\mathbf{A}$, sodass die gemeinsamen Verteilungen der Komponenten s_1, s_2 sowie der Mischungen x_1, x_2 somit identisch sind. Aufgrund dieser Identität kann die Mischungsmatrix $\mathbf{A}$ nicht auf Basis der Mischungen x_1, x_2 geschätzt werden, sodass die Nicht-Normalität im Rahmen der *Independent Component Analysis* erforderlich ist (Dryden & Mardia 2016, S. 161f.).

3.5.2 Maße der Nicht-Normalität

Zur Bestimmung des Vorliegens nicht-normalverteilter Komponenten werden im Folgenden zwei Maße in Form der Kurtosis sowie der Negentropie betrachtet, die im Rahmen der Kontrastfunktionen der Algorithmen einer *Independent Component Analysis* Verwendung finden.

3.5.2.1 Kurtosis

Ein klassisches Maß der Nicht-Normalität von Zufallsgrößen ist die Kurtosis, die ein Spezialfall der Kumulante vierter Ordnung ist und somit eine Statistik höherer Ordnung darstellt. Die Kurtosis ist dabei wiefolgt definiert

$$kurt(s) = E(s^4) - 3\left(E(s^2)\right)^2. \tag{3.24}$$

Gilt die Bedingung in Formel (3.13), d.h. ein *Whitening* der Daten ist erfolgt, dann vereinfacht sich der Ausdruck zu

$$kurt(s) = E(s^4) - 3,, \tag{3.25}$$

was einer verschobenen Form des vierten Moments entspricht (Hyvärinen, Karhunen *u. a.* 2001, S. 38). Normalverteilte Zufallsvariablen besitzen eine Kurtosis von null, wohingegen die Kurtosis der meisten nicht-normalverteilten Zufallsvariablen ungleich null ist.[6] Dabei werden Verteilungen mit einer negativen Kurtosis als *platykurtisch* bezeichnet und haben eine 'flache' Wahrscheinlichkeitsdichte mit eher um null konstanten und sonst sehr kleinen Werten. Verteilungen mit einer positiven Kurtosis hingegen werden als *leptokurtisch* bezeichnet und besitzen eine 'spitze' Wahrscheinlichkeitsdichte mit schweren Rändern (*heavy*

[6] Es existieren ebenfalls nicht-normalverteilte Zufallsvariablen mit einer Kurtosis von null, diese können allerdings als selten angenommen werden.

tails) (Hyvärinen, Karhunen *u. a.* 2001, S. 38f.). Die Kurtosis oder vielmehr deren absoluter Wert ist aufgrund der sowohl computationalen als auch theoretischen Einfachheit ein häufig verwendetes Maß der Nicht-Normalität von Zufallsvariablen (Hyvarinen & Oja 2000, S. 417).[7] Die computationale Einfachheit folgt dabei aus der lediglichen Verwendung der Schätzung des vierten Momentes (siehe Formel (3.25)) aus den Daten, wohingegen die Erfüllung des Superpositionsprinzips im Rahmen der Kurtosis als lineare Operation zu jener theoretischen Einfachheit führt. Somit gilt für zwei unabhängige Zufallsvariablen s_1, s_2 und ein Skalar α (Meyer-Baese & Schmid 2014, S. 270)

$$kurt(s_1 + s_2) = kurt(s_1) + kurt(s_2), \tag{3.26}$$

$$kurt(\alpha s) = \alpha^4 kurt(s). \tag{3.27}$$

Die Verwendung der Kurtosis im Rahmen der *Independent Component Analysis* erfordert, ausgehend von einer Gewichtungsmatrix $\mathbf{W}$, die Berechnung der Richtungen mit der am stärksten wachsenden (positive Kurtosis) bzw. abnehmenden (negative Kurtosis) Kurtosis zur Maximierung/Minimierung der Nicht-Normalität (ebd., S. 270). Entgegen der Einfachheit der Kurtosis als Maß der Nicht-Normalität von Zufallsvariablen, besitzt diese gleichsam Nachteile, die es zu berücksichtigen gilt. So ist die Kurtosis sensitiv für Ausreißer und somit kein robustes Maß der Nicht-Normalität (Huber 1985, S. 446). Ferner ist sie für einen Vergleich zwischen *platykurtischen* und *leptokurtischen* Zufallsvariablen nicht geeignet. So ist der Wertebereich der Kurtosis für *leptokurtische* Zufallsvariablen nach oben unbeschränkt, wohingegen dieser für *platykurtische* Zufallsvariablen nach unten beschränkt ist (Hyvärinen, Karhunen *u. a.* 2001, S. 39f.).[8] Die Kurtosis dient somit als einfaches Maß der Nicht-Normalität, falls die Gesamtheit der zu betrachtenden Zufallsvariablen entweder allesamt *platykurtisch* oder *leptokurtisch* sind.

3.5.2.2 Negentropie

Als weiteres Maß der Nicht-Normalität dient neben der Kurtosis das informationstheoretische Maß der Entropie $H(\cdot)$ einer Zufallsvariable, die den Grad der Information bei Beobachtung der Variable angibt. So kann die Entropie einer Zufallsvariable als Maß für die Unsicherheit eines Ereignisses angesehen werden, die somit für eine Verteilung mit höchstmöglicher Zufälligkeit maximal ist (Meyer-Baese & Schmid 2014, S. 270). Die differentielle

[7] Eine Verwendung der quadrierten Kurtosis ist ebenfalls denkbar.

[8] Das Minimum der Kurtosis einer normierten *platykurtischen* Zufallsvariable liegt bei -2.

Entropie[9] $H(\cdot)$ eines Zufallsvektors $\mathbf{y}$ ist definiert als

$$H(\mathbf{y}) = -\int f(\mathbf{y}) \log(f(\mathbf{y}))d\mathbf{y} \tag{3.28}$$

mit gegebener Dichtefunktion $f(\mathbf{y})$ (Cover & Thomas 2001, S. 224).

Es kann gezeigt werden, dass die Normalverteilung im Vergleich zu anderen Wahrscheinlichkeitsverteilungen bei gegebener Varianz die höchste Entropie aufweist (ebd., S. 233f.). Somit kann die (differentielle) Entropie als Maß der Nicht-Normalität von Komponenten genutzt werden. Um ein Maß zu erhalten, das nicht-negativ und lediglich für normalverteilte Variablen null ist, wird häufig eine modifizierte Form der differentiellen Entropie in Form der Negentropie[10] $J(\cdot)$ verwendet. Diese ist definiert als

$$J(\mathbf{y}) = H(\mathbf{y_{gauss}}) - H(\mathbf{y}), \tag{3.29}$$

wobei $\mathbf{y_{gauss}}$ normalverteilt ist mit Mittelwert und Varianz identisch zu $\mathbf{y}$ (Hyvarinen & Oja 2000, S. 8). Die differentielle Entropie von $\mathbf{y_{gauss}}$ ist gegeben durch

$$H(\mathbf{y_{gauss}}) = \frac{1}{2} \log |\det \Sigma| + \frac{n}{2} (1 + \log 2\pi) . \tag{3.30}$$

Ferner ist die Negentropie sowohl invariant gegenüber einer inversen linearen Transformationen als auch skaleninvariant (Comon 1994, S. 305).

Nachteilig ist im Zuge dieses Maßes jedoch die damit verbundene hohe Rechenintensität, da gemäß der Definition eine Schätzung der Dichtefunktion $f(\mathbf{y})$ erforderlich ist, sodass eine Approximation der Negentropie notwendig ist, die im Folgenden dargestellt ist

$$J(\gamma) \propto [E(G(\gamma)) - E(G(\nu))]^2 , \tag{3.31}$$

wobei ν standardnormalverteilt, γ Zufallsvariable einer Verteilung mit Mittelwert null und Varianz eins sowie $G(\cdot)$ eine beliebige nicht-quadratische Funktion ist (Hyvarinen & Oja 2000, S. 419). Dabei ist eine geeignete Wahl der Funktion $G(\cdot)$ entscheidend, so führen langsam wachsende $G(\cdot)$ zu robusteren Schätzern. Als nützliche Wahl haben sich dabei

[9] Die differentielle Entropie bezeichnet die Entropie für stetige Zufallsvariablen.
[10] Kurzform für negative Entropie.

folgende Funktionen erwiesen

$$G_1(u) = \frac{1}{a_1} \log \cosh(a_1 u)$$
$$G_2(u) = -\exp\left(-\frac{u^2}{2}\right) \tag{3.32}$$

mit $1 \leq a_1 \leq 2$ konstant (Hyvarinen & Oja 2000, S. 419). Die somit erhaltenen Approximationen der Negentropie bilden einen Kompromiss zwischen den Eigenschaften der klassischen Maße in Form der Kurtosis und der Negentropie, da sie einfach, nicht rechenintensiv und robust sind (ebd., S. 419).

3.6 Methoden

Nachdem im vorangegangenen Abschnitt verschiedene Maße der Nicht-Normalität von Zufallsvariablen betrachtet wurden, werden diese nun im Rahmen von Kontrastfunktionen. Ein *Independent Component Analysis* Algorithmus umfasst zudem einen Optimierungsalgorithmus zur Optimierung jener Kontrastfunktionen und somit Schätzung der Mischungsmatrix $\mathbf{A}$ bzw. der inversen Matrix $\mathbf{W}$ sowie der unabhängigen Komponenten $\mathbf{s}$. So ist eine Methode oder ein Algorithmus im Rahmen der *Independent Component Analysis* charakterisiert durch eine Kontrastfunktion sowie einen Optimierungsalgorithmus

$$\textbf{ICA Algorithmus} = \textbf{Kontrastfunktion} + \textbf{Optimierungsalgorithmus}.$$

Im Folgenden werden die Methoden der `FastICA`, `InfoMax` sowie `JADE` eingehend beschrieben.

3.6.1 FastICA

Als erste jener Methoden wird der `FastICA` Algorithmus betrachtet, in Rahmen dessen eine erfolgte Zentrierung sowie ein *Whitening* der Daten analog zu Abschnitt 3.3 angenommen wird.
Vereinfachend wird zu Beginn die `FastICA` Methode mit einer Einheit betrachtet.[11] Diese Einheit besitzt einen Gewichtungsvektor $\mathbf{w}$, der im Rahmen der Iterationsschritte anpasst

[11] Eine Einheit entspricht dabei einem künstlichen Neuron. Aufgrund der Verwendung lediglich einer Einheit werden diese Algorithmen auch als *one-unit Algorithmen* bezeichnet.

wird, sodass die Projektion $\mathbf{w}^T\mathbf{x}$ die Nicht-Normalität in Form der approximierten Negentropie[12] $J(\mathbf{w}^T\mathbf{x})$ in Formel (3.31) maximiert (ebd., S. 422f.). Der `FastICA` Algorithmus verwendet dabei als Optimierungsalgorithmus eine Fixpunktiteratiton zur Bestimmung des Maximums der Nicht-Normalität von $\mathbf{w}^T\mathbf{x}$ (ebd., S. 423). Die grundlegende Form des `FastICA` Algorithmus lautet wiefolgt

1. Vorverarbeitung der Daten durch Zentrierung und *Whitening*
2. Zufällige Initialisierung des Gewichtungsvektors $\mathbf{w}$ mit $\|\mathbf{w}\| = 1$ als Startwert
3. Anpassung des Gewichtungsvektors durch
$$\mathbf{w} \leftarrow E(\mathbf{x}g(\mathbf{w}^T\mathbf{x})) - E(g'(\mathbf{w}^T\mathbf{x}))\mathbf{w}$$
4. Normierung des Gewichtungsvektors durch $\mathbf{w} \leftarrow \frac{\mathbf{w}}{\|\mathbf{w}\|}$
5. Wenn keine Konvergenz, zurück zu Schritt 3.

Die Funktion $g(\cdot)$ ist dabei die Ableitung der Funktion $G(\cdot)$ in Formel (3.31), sodass in Verbindung mit Formel (3.32) folgt

$$\begin{aligned} g_1(u) &= \tanh(\alpha_1 u) \\ g_2(u) &= u \exp\left(-\frac{u^2}{2}\right) \end{aligned} \tag{3.33}$$

mit $1 \leq \alpha_1 \leq 2$ konstant (ebd., S. 423). Dabei bedeutet Konvergenz, dass die alten und neuen Werte von $\mathbf{w}$ in dieselbe Richtung zeigen, d.h. ihr Skalarprodukt ist (fast) 1 (ebd., S. 423). Dabei ist die Konvergenz des Vektors in einem einzigen Punkt nicht notwendig, da $\mathbf{w}$ und $-\mathbf{w}$ dieselbe Richtung definieren (ebd., S. 423).
Die Maxima der Approximation der Negentropie von $\mathbf{w}^T\mathbf{x}$ werden bei bestimmten Optima von $E(G(\mathbf{w}^T\mathbf{x}))$ erhalten; so ist gemäß den Kuhn-Tucker Bedingungen das Optimum unter der Nebenbedingung $E(\mathbf{w}^T\mathbf{x}) = \|\mathbf{w}\|^2 = 1$ gegeben, wobei

$$\underbrace{E(xg(\mathbf{w}^T\mathbf{x})) - \beta\mathbf{w}}_{=:F(\mathbf{w})} = 0. \tag{3.34}$$

[12]Ebenso ist eine Verwendung der Kurtosis im Rahmen der `FastICA` Methode möglich, jedoch konnte eine höhere Genauigkeit im Rahmen der Approximation der Negentropie nachgewiesen werden (siehe beispielsweise Hyvärinen 1998).

Sei $F(\cdot)$ Bezeichnung für die linke Seite der Formel (3.34), dann folgt für die Jacobi-Matrix

$$JF(\mathbf{w}) = E(\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})) - \beta\mathbf{I}. \tag{3.35}$$

Zur Vereinfachung der Invertierung dieser Matrix wird dabei der erste Term approximiert durch

$$E(\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})) \approx E(\mathbf{x}\mathbf{x}^T)E(g'(\mathbf{w}^T\mathbf{x}) = E(g'(\mathbf{w}^T\mathbf{x}))\mathbf{I},$$

wodurch die Jacobi-Matrix zu einer Diagonalmatrix wird. Die approximative Newton-Interation ist somit gegeben durch

$$\mathbf{w} \leftarrow \mathbf{w} - \frac{E(\mathbf{x}g(\mathbf{w}^T\mathbf{x})) - \beta\mathbf{w}}{E(g'(\mathbf{w}^T\mathbf{x})) - \beta}. \tag{3.36}$$

Ferner führt eine Multiplikation der Formel (3.36) mit $\beta - E(g'(\mathbf{w}^T\mathbf{x})$ zu folgender Vereinfachung

$$\mathbf{w} \leftarrow E(\mathbf{x}g(\mathbf{w}^T\mathbf{x})) - E(g'(\mathbf{w}^T\mathbf{x}))\mathbf{w}, \tag{3.37}$$

die den `FastICA` Algorithmus liefert (Hyvarinen 1999a, S. 630).

Nachdem der `FastICA` Algorithmus vereinfachend für eine Einheit betrachtet wurde, wird im Folgenden die Verwendung im mit mehreren Einheiten betrachtet. Dabei wird der zuvor betrachtete *one-unit Algorithmus* um weitere Einheiten unter Berücksichtigung der jeweiligen Gewichtungsvektoren $\mathbf{w_i}$ erweitert (Hyvarinen & Oja 2000, S. 423). Um im Falle mehrere Einheiten eine Konvergenz verschiedener Vektoren gegen das selbe Maximum zu verhindern, müssen die Outputs $\mathbf{w_1}^T\mathbf{x}, ..., \mathbf{w_n}^T\mathbf{x}$ nach jeder Iteration dekorreliert werden, wobei dies begründet durch das *Whitening* äquivalent zur Orthogonalisierung ist (Hyvärinen, Karhunen *u. a.* 2001, S. 192-194).

Im Kontext der Orthogonalisierung existieren verschiedene Methoden, wobei in den folgenden Ausführungen lediglich die Methode der symmetrischen Dekorrelation betrachtet wird, da sie nach *Meyer/Schmid* (2014) gegenüber anderen Methoden folgende Vorteile bietet (Meyer-Baese & Schmid 2014, S. 275):

1. Die Gewichtungsvektoren $\mathbf{w_i}$ werden parallel und nicht sequentiell geschätzt.

2. Fehler eines Gewichtungsvektors werden nicht auf einen weiteren übertragen.

Die symmetrische Orthogonalisierung der Gewichtungsmatrix $\mathbf{W} = (\mathbf{w_1}, ..., \mathbf{w_n})^T$ wird dabei erreicht durch

$$\mathbf{W} = (\mathbf{W}\mathbf{W}^T)^{-\frac{1}{2}}\mathbf{W}, \tag{3.38}$$

wobei die Inverse der Quadratwurzel $(\mathbf{W}\mathbf{W}^T)^{-\frac{1}{2}}$ durch die Diagonalisierung von $\mathbf{W}\mathbf{W}^T = \mathbf{F}\mathbf{\Lambda}\mathbf{F}^T$ als $(\mathbf{W}\mathbf{W}^T)^{-\frac{1}{2}} = \mathbf{F}^{-\frac{1}{2}}\mathbf{\Lambda}\mathbf{F}^T$ erhalten wird.
Eine Beschreibung des `FastICA` Algorithmus zur Schätzung mehrerer unabhängiger Komponenten lautet wiefolgt

1. Vorverarbeitung der Daten durch Zentrierung und *Whitening*
2. Wahl der Anzahl der zu schätzenden unabhängigen Komponenten m
3. Wahl eines zufälligen Startwertes für jeden Gewichtungsvektor $\mathbf{w_i}$ mit $\|w_i\| = 1$ für $i = 1, ..., m$
4. Anpassung der einzelnen Gewichtungsvektoren $\mathbf{w_i}$ mit $i = 1, ..., m$ gemäß

   $$\mathbf{w_i} \leftarrow E(\mathbf{x}g(\mathbf{w_i}^T\mathbf{x})) - E(g'(\mathbf{w_i}^T\mathbf{x}))\mathbf{w_i}$$

   mit $g(\cdot)$ aus (3.33).
5. Symmetrische Orthogonalisierung der Matrix $\mathbf{W} = (\mathbf{w_1}, ..., \mathbf{w_m})^T$

   $$\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^T)^{-\frac{1}{2}}\mathbf{W}$$

6. Wenn keine Konvergenz, zurück zu Schritt 4

Dabei bietet die `FastICA` Methode die folgenden Eigenschaften (Hyvarinen & Oja 2000, S. 424):

- Schnelle Konvergenz, da unter Annahme des ICA Datenmodells quadratisch oder kubisch,[13]
- Einfache Verwendung, da im Vergleich zu gradientbasierten Verfahren keine Parameter für die Schrittweite zu wählen sind,

[13] Bei gradientbasierten Methoden ist die Konvergenz im Vergleich lediglich linear.

- Direkte Schätzung der unabhängigen Komponenten von (fast) jeder Verteilung abgesehen der Normalverteilung,
- Optimierung der Leistungsfähigkeit durch geeignetes nicht-lineares $g(\cdot)$,
- Unabhängige Komponenten können einzeln bestimmt werden,
- Fixpunktalgorithmen besitzen die meisten Vorteile neuronaler Algorithmen, so sind diese parallel, verteilt, nicht rechenintensiv und benötigen eine geringe Speicherkapazität.

3.6.2 InfoMax

Als weiterere Methode im Rahmen der *Independent Component Analysis* wird nun der `InfoMax` Algorithmus betrachtet. Dieser bezeichnet dabei einen Lernalgorithmus in Form eines Gradientverfahrens zur Maximierung der Output-Entropie $H(\mathbf{y})$ (siehe Formel (3.28)) eines neuronalen Netzes mit nicht-linearen Einheiten.[14]. Die Architektur eines solchen neuronalen Netzes ist in Abbildung 3.5 dargestellt.

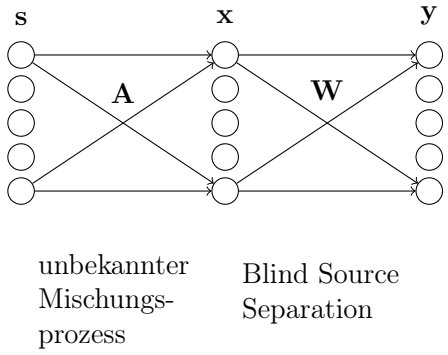


Abbildung 3.5: Feedforward Netz des `InfoMax` Algorithmus

Quelle: Eigene Erstellung angelehnt an (Bell & Sejnowski 1995, S. 1139)

Das Netzwerk besitzt dabei n Input- und n Output-Einheiten sowie eine $n \times n$ Gewichtungsmatrix $\mathbf{W}$, die die Neuronen des Input-Layers mit jenen des Output-Layers verbindet.

[14]Dies ist äquivalent zur Maximierung der gemeinsamen Information $I(\mathbf{x}, \mathbf{y}) = H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{X})$, die in einem neuronalen Netz mit nicht-linearen Einheiten transferiert wird (siehe dazu Bell & Sejnowski 1995, S. 1131).

Bei Annahmen sigmoider Aktivierungsfunktionen ergibt sich für den Output

$$\mathbf{y} = g(\mathbf{u}) \tag{3.39}$$

mit

$$\begin{aligned} \mathbf{u} &= \mathbf{W}\mathbf{x}, \\ g(u_i) &= \frac{1}{1+\exp(-u_i)}. \end{aligned}$$

Sei $g(u)$ invertierbar, dann kann die Wahrscheinlichkeitsdichte des Outputs $f_{\mathbf{y}(\mathbf{y})}$ als Funktion der Wahrscheinlichkeitsdichte des Inputs $f_{\mathbf{x}}(\mathbf{x})$ formuliert werden (Papoulis & Pillai 1991, S. 143

$$f_{\mathbf{y}(\mathbf{y})} = \frac{f_{\mathbf{x}}(\mathbf{x})}{|J|}, \tag{3.40}$$

mit $|J|$ die Determinante der Jacobimatrix

$$J = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_n}{\partial x_1} & \cdots & \frac{\partial y_n}{\partial x_n} \end{pmatrix}. \tag{3.41}$$

Für die Output-Entropie $H(\mathbf{y})$ gilt somit

$$\begin{aligned} H(\mathbf{y}) &= -E\left(\log(f_{\mathbf{y}}(\mathbf{y})\right) \\ &= E\left(|J|\right) - E\left(\log(f_{\mathbf{x}}(\mathbf{x}))\right). \end{aligned} \tag{3.42}$$

Die Idee des `InfoMax` Algorithmus ist die Bestimmung einer optimalen Gewichtungsmatrix $\mathbf{W}$ im Sinne einer maximalen Entropie $H(\mathbf{y})$. Die geschieht auf Basis eines Gradientenverfahren (*stochastic gradient ascend)* mit der folgenden Lernregel (Bell & Sejnowski 1995, S.

1154f.)

$$\begin{aligned}\Delta\mathbf{W} \propto \frac{\partial H}{\partial \mathbf{W}} &= \frac{\partial}{\partial \mathbf{W}} \log|J| \\ &= \frac{\partial}{\partial \mathbf{W}} \log|\mathbf{W}| + \frac{\partial}{\partial \mathbf{W}} \prod_i |y_i'| \\ &= \mathbf{W}^{-T} + (1 - 2\mathbf{y})\mathbf{x}^T \end{aligned} \tag{3.43}$$

Eine Beschreibung des `InfoMax` Algorithmus ist im Folgenden gegeben

1. Vorverarbeitung der Daten durch Zentrierung und *Whitening*
2. Wahl der Anzahl der unabhängigen Komponenten m
3. Initialisierung der Gewichtungsmatrix $\mathbf{W}$
4. Anpassung der Gewichtungsmatrix anhand der Lernregel

$$\Delta\mathbf{W} \propto \mathbf{W}^{-T} + (1 - 2\mathbf{y})\mathbf{x}^T$$

5. Wenn keine Konvergenz, zurück zu Schritt 4

3.6.3 JADE

Als letzte Methode im Rahmen der *Independent Component Analysis* wird der *Joint Approximate Diagonalization of Eigenmatrices* Algorithmus oder kurz `JADE` betrachtet. Diese Methode basiert auf der Kumulanten vierter Ordnung bzw. der in Formel (3.24) definierten Kurtosis

$$kurt(x) = E(x^4) - 3\left(E(x^2)\right)^2$$

zur Schätzung der Inversen der Mischungsmatrix $\mathbf{W} = \mathbf{A}^{-1}$ sowie der unabhängigen Komponenten $\mathbf{s}$ (Cardoso & Souloumiac 1993, S. 336). Im Rahmen des des `JADE` Algorithmus erfolgt in einem ersten Schritt analog zu Abschnitt 3.3 eine Vorverarbeitung in Form einer Zentrierung und eines *Withening* der Daten (ebd., S. 366), sodass

$$\mathbf{z} = \mathbf{V}\mathbf{x} = \underbrace{VA}_{=:\tilde{\mathbf{A}}}\mathbf{s}$$

mit $\mathbf{V}$ *Whitening*-Matrix und $\mathbf{z}$ geweißten Mischungsvektor.

Das Ziel des `JADE` Algorithmus ist die Bestimmung der Komponenten s_i, sodass diese möglichst unabhängig sind. Dazu wird im Rahmen dieser Methode jene Rotationsmatrix $\mathbf{R}$ unter Verwendung der Kumulanten vierter Ordnung gesucht, die zur stochastischen Unabhängigkeit der Komponenten von $\mathbf{s}$ führt (Cardoso 1999, S. 169). Die Kumulante vierter Ordnung mit $i, j, k, l = 1, ..., n$ ist dabei definiert als

$$\begin{aligned} cum(z_i, z_j, z_k, z_l) = {} & E(z_i z_j^T z_k z_l^T) - E(z_i z_j^T)E(z_k z_l^T) \\ & - E(z_i z_k^T)E(z_j z_l^T) - E(z_i z_l^T)E(z_j z_k^T), \end{aligned} \tag{3.44}$$

sodass für $i = j = k = l$ die Kurtosis aus Abschnitt 3.5.2.1 folgt

$$\kappa_4(z, z, z, z) = E(z^4) - 3\left(E(z^2)\right)^2. \tag{3.45}$$

Der Kumulantentensor für $\mathbf{z}$ aus Formel (3.44) wird fener nach *Cardoso/Souloumiac* (1993) durch $n \times n$ Kumulantenmatrizen $\mathbf{Q}^{\mathbf{z}}(\mathbf{M})$ der Form

$$[\mathbf{Q}^{\mathbf{z}}(\mathbf{M})]_{ij} = \sum_{k,l=1}^{n} cum(z_i, z_j, z_k, z_l)\mathbf{M}_{kl} \tag{3.46}$$

mit $i, j = 1, ..., n$ und $\mathbf{M}$ einer beliebigen $n \times n$ Matrix dargestellt (Cardoso & Souloumiac 1993, S. 364f.). Aufgrund der angenommenen Unabhängigkeit der Komponenten s_i mit $i = 1, ..., n$ des Zufallsvektors $\mathbf{s}$ folgt für die Kumulantenmatrizen durch Formel (3.44)

$$\mathbf{Q}^{\mathbf{z}}(\mathbf{M}) = \tilde{\mathbf{A}}\boldsymbol{\Delta}(\mathbf{M})\tilde{\mathbf{A}}^T \tag{3.47}$$

mit

$$\boldsymbol{\Delta}(\mathbf{M}) = \begin{pmatrix} \kappa_4(s_1)\tilde{\mathbf{a}}_1^T\mathbf{M}\tilde{\mathbf{a}}_1 & 0 & \dots & 0 \\ 0 & \kappa_4(s_2)\tilde{\mathbf{a}}_2^T\mathbf{M}\tilde{\mathbf{a}}_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \kappa_4(s_n)\tilde{\mathbf{a}}_n^T\mathbf{M}\tilde{\mathbf{a}}_n \end{pmatrix}$$

und

$$\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T = \mathbf{I}$$

(siehe Abschnitt 3.3.2), wobei $\tilde{a}_i$ die i-te Spalte von $\tilde{\mathbf{A}}$ ist (Cardoso 1999, S. 169). Sei nun $\mathcal{M} = \{\mathbf{M}_1, ..., \mathbf{M}_P\}$ eine Menge von $n \times n$ Matrizen $\mathbf{M_i}$ und somit $\mathbf{Q^z}_i := \mathbf{Q^z}(\mathbf{M}_i)$ mit $i = 1, ..., P$. Die größtmögliche Menge der Kumulantenmatrizen $\{\mathbf{Q^z}_i\}$ ergibt sich per Definition, wenn $\mathcal{M}$ eine orthonormale Basis des linearen Raums der $n \times n$ Matrizen $\mathbf{M}_i$ ist, sodass $|\{\mathbf{Q^z}_i\}| = P = n^2$ (ebd., S.171). *Cardoso/Souloumiac* (1999) geben jedoch eine alternative zur Reduzierung der Anzahl benötigter $n \times n$ Kumulantenmatrizen $\mathbf{M}_i$, vorausgesetzt das generative Modell in Formel (3.1) ist korrekt spezifiziert. Dann gilt $\text{rang}(\mathbf{Q}_i^{\mathbf{z}}) = n$ aufgrund der n Diagonalelemente von $\mathbf{\Delta}$.[15] Durch die Symmetrie der Kumulanten folgt die Existenz von n Eigenmatrizen $\mathbf{E}_1, ..., \mathbf{E}_n$, die orthonormal sind und $\mathbf{Q}_i^{\mathbf{z}}(\mathbf{E}_i) = \mu_i \mathbf{E}_i$ erfüllen, wobei μ_i der korrespondierende Eigenvektor ist (ebd., S. 176f.). Somit lässt sich die Anzahl der Matrizen $\mathbf{M}_i$ auf n reduzieren.

Nachdem im vorangegangenen Schritt die größtmögliche Menge der Kumulantenmatrizen $\{\mathbf{Q}_i^{\mathbf{z}}\}$ bestimmt wurde, ist nun jene Rotationsmatrix $\mathbf{R}$ mit

$$\mathbf{y} = \mathbf{Rz} \tag{3.48}$$

zu ermitteln, sodass die Kumulantenmatrizen $\mathbf{Q}_i^{\mathbf{z}}$ simultan diagonalisiert werden. Dazu definieren *Cardoso/Souloumiac* (1999) als Maß der Diagonalität einer Matrix $\mathbf{F}$ die Summe der quadrierten Nicht-Diagonalelemente (ebd., S. 171)

$$\text{Off}(\mathbf{F}) = \sum_{i \neq j} (f_{ij})^2. \tag{3.49}$$

Es gilt $\text{Off}(\tilde{\mathbf{A}}^T \mathbf{Q}_i \tilde{\mathbf{A}}) = \text{Off}(\mathbf{\Delta}_i) = 0$, da $\mathbf{Q}_i = \tilde{\mathbf{A}} \mathbf{\Delta}_i \tilde{\mathbf{A}}$ und $\tilde{\mathbf{A}} \tilde{\mathbf{A}}^T = \mathbf{I}$. Daher ergibt sich $\forall\, \mathbf{M}_i \in \mathcal{M}$ mit $i = 1, ..., n$ als nicht-negatives simultanes Diagonalisierungskriterium für den `JADE` Algorithmus

$$\mathcal{D}_{\mathcal{M}}(\mathbf{R}) = \sum_{\mathbf{M}_i \in \mathcal{M}} \text{Off}(\mathbf{R}^T \mathbf{Q^z}(\mathbf{M}_i) \mathbf{R})^2, \tag{3.50}$$

mit

$$\mathbf{R}\mathbf{R}^T = \mathbf{I}.$$

Nach der Definition des Maßes der Diagonalität der Kumulantenmatrizen wird wird die

[15] Korrekterweise entspricht $\text{rang}(\mathbf{Q}_i^{\mathbf{z}})$ der Anzahl der Komponenten mit einer Kurtosis ungleich 0.

Rotationsmatrix $\mathbf{R}$ mit

$$\mathbf{R} = \arg\min_{\mathbf{R}} \sum_{\mathbf{M}_i \,\epsilon\, \mathcal{M}} \text{Off}(\mathbf{R}^T \mathbf{Q}^{\mathbf{z}}(\mathbf{M}_i)\mathbf{R})^2, \tag{3.51}$$

auf Basis einer erweiterten Jacobi-Methode bestimmt (Cardoso & Souloumiac 1993, S. 366).[16] Abschließend wird die Entmischungsmatrix $\mathbf{W} = \mathbf{A}^{-1}$ und somit der Vektor $\mathbf{s}$ mit den unabhängigen Komponenten s_i bestimmt (Cardoso 1999, S. 175).

$$\mathbf{s} = \mathbf{W}\mathbf{x} = \mathbf{R}^T\mathbf{z}. \tag{3.52}$$

Eine Beschreibung des `JADE` Algorithmus lautet wiefolgt:

1. Vorverarbeitung der Daten durch Zentrierung und *Whitening*
2. Bestimmung der größtmöglichen Menge der Kumulantenmatrizen vierter Ordnung

   $$\{\mathbf{Q}_i^{\kappa_4}\}$$

3. Bestimmung der Rotationsmatrix $\mathbf{R}$ zur simultanen Diagonalisierung der Kumulantenmatrizen $\mathbf{Q}_i^{\mathbf{z}}$

   $$\mathbf{R} = \arg\min_{\mathbf{R}} \sum_i \text{Off}(\mathbf{R}^T \mathbf{Q}_i^{\mathbf{z}} \mathbf{V})$$

4. Schätzung der Mischungsmatrix $\mathbf{A}$

Sowohl Vorteile als auch Nachteile des `JADE` Algorithmus liegen in der lediglichen Verwendung von Matrizenberechnungen zur Bestimmung der Entmischungsmatrix $\mathbf{W} = \mathbf{A}^{-1}$, so ist entgegen gradientbasierten Verfahren kein Lernparameter zu wählen, jedoch ist eine Verwendung des Algorithmus aufgrund der Rechenintensität in hoch-dimensonalen Räumen nicht möglich (Hyvärinen, Karhunen *u. a.* 2001, S. 234).

[16] Dabei ist die Minimierung der Nicht-Diagonalelementen äquivalent zur Maximierung der Diagonalelemente und somit der Maximierung der Kurtosis.

3.7 Limitationen

Nach einer Betrachtung der Algorithmen werden in diesem Abschnitt die Limitationen einer *Independent Component Analysis* angeführt. Aus Formel (3.2) ist ersichtlich, dass im Rahmen einer *Independent Component Analysis* folgende zwei Uneindeutigkeiten vorliegen. Die erste dieser Uneindeutigkeiten ist die Unmöglichkeit der Bestimmung der Varianzen der unabhängigen Komponenten s_i aufgrund der Unkenntnis sowohl über die Mischungsmatrix $\mathbf{A}$ als auch über den Vektor der unabhängigen Komponenten $\mathbf{s}$. Sei dazu

$$\mathbf{x} = \sum_i (\frac{1}{\alpha_i}\mathbf{a}_i)(s_i\alpha_i) \tag{3.53}$$

mit $\mathbf{a_i}$ Spaltenvektor der Matrix $\mathbf{A}$ und einem Skalar α_i, dann kann die Multiplikation einer unabhängigen Komponente s_i mit α_i jederzeit mit der Division des Mischungsvektors $\mathbf{a}_i$ durch α_i aufgehoben werden. Dem kann zum Teil durch die Restriktion $E(s_i^2) = 1$ entgegnet werden, wodurch diese Uneindeutigkeit auf das Vorzeichen der unabhängigen Komponenten reduziert wird, was in den meisten Fällen unproblematisch ist (Hyvärinen, Karhunen *u. a.* 2001, S. 414).
Ferner ist eine Ordnung der unabhängigen Komponenten s_i ebenso aufgrund der Unkenntnis der Mischungsmatrix $\mathbf{A}$ und des Vektors der unabhängigen Komponenten $\mathbf{s}$ nicht möglich. Sei dazu

$$\mathbf{x} = \left[\mathbf{A}\mathbf{P}^{-\mathbf{1}}\mathbf{\Lambda}^{-\mathbf{1}}\right]\left[\mathbf{\Lambda}\mathbf{P}\mathbf{s}\right] \tag{3.54}$$

mit $\mathbf{P}$ beliebige Permutationsmatrix, $\mathbf{\Lambda}$ beliebige Diagonalmatrix. Dann können beliebige Ordnungen der unabhängigen Komponenten $\mathbf{Ps}$ entstehen deren Mischungsmatrix $\mathbf{A}\mathbf{P}^{-1}$ mit einer der zuvor betrachteten Methoden zu bestimmen ist (ebd., S. 154).

Kapitel 4

Methodischer Vergleich

Im Rahmen dieses Kapitels werden nun die theoretischen Betrachtungen der *Statistical Shape Analysis* und der *Independent Component Analysis* der vorangegangenen Kapitel zusammengeführt. Dabei werden die in Abschnitt 3.6 vorgestellten Algorithmen `FastICA`, `InfoMax` und `JADE` im Rahmen der *Statistical Shape Analysis* verglichen. Zu Beginn des Kapitels werden zuerst die Datensätze sowie deren Aufbereitung beschrieben, woraufhin im Anschluss eine Modifikation des *Point Distribution Model* unter Verwendung der *Independent Component Analysis* gegeben wird. Anschließend werden die Ergebnisse der Analyse dargestellt, wobei neben der computationalen Eigenschaften die Variabilität der *Shapes* durch ein *Point Distribution Model* auf Basis der verschiedenen Algorithmen betrachtet wird.

4.1 Datenbeschreibung

Als Datengrundlage dieser Betrachtung dienen die Datensätze `digit3.dat` sowie `mice` aus dem R-Package `shapes` (Dryden 2017). Exemplarisch ist jeweils eine *Shape* der Datensätze mitsamt der *Landmarks* in Abbildung 4.1 dargestellt.
Der Datensatz `digit3.dat` umfasst 30 *Shapes* der handschriftlich verfassten Ziffer 3 aus einer Stichprobe britischer Postleitzahlen. Die *Shapes* der Ziffer werden dabei durch 13 *Landmarks* definiert, die in 5 *mathematische Landmarks* sowie 8 *Pseudo-Landmarks* in approximativ äquidistanten Intervallen zwischen den *mathematischen Landmarks* unterteilt werden können (Dryden & Mardia 2016, S. 12). Die geometrischen Eigenschaften der *mathematischen Landmarks* sind in Tabelle 4.1 aufgeführt.

Nummer	Charakteristik
1	äußerster Wert unten-links
4	maximale Krümmung des unteren Bogens
7	äußerster Wert des mittigen Vorsprungs
10	maximale Krümmung des oberen Bogens
13	äußerster Wert oben-links

Tabelle 4.1: *Mathematische Landmarks* (Datensatz: `digit3.dat`)

Der Datensatz `mice` hingegen umfasst 76 *Shapes* des zweiten Brustwirbels (T2) von Mäusen aus einer Studie zur Untersuchung der Größe sowie *Shape* des Wirbels bei Mäusen unterschiedlicher Größe (siehe beispielsweise Mardia & Dryden 1989). So sind die Mäuse in den Gruppen *"Large"* ($n = 23$), *"Small"* ($n = 23$) und *"Control"* ($n = 30$) zugeordnet. Die *Shapes* des Brutwirbels werden dabei durch 6 *Landmarks* definiert, die allesamt *mathematische Landmarks* sind und dabei folgende Charakteristika aufweisen.

Nummer	Charakteristik
1	äußerster Wert linker Bogen
2	äußerster Wert rechter Bogen
3	Extremwert der negativen Krümmung am Rande des Dornfortsatzes rechts
4	äußerster Wert des Dornfortsatzes
5	Extremwert der negativen Krümmung am Rande des Dornfortsatzes links
6	äußerster Wert gegenüberliegend *Landmark* 4

Tabelle 4.2: *Mathematische Landmarks* (Datensatz: `mice`)

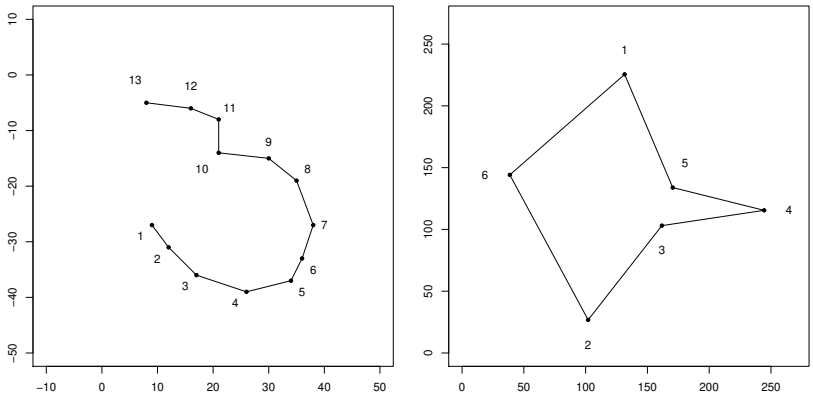


Abbildung 4.1: *Shape* aus dem Datensatz `digit3.dat` *(links)* sowie `mice` *(rechts)*

4.2 Datenaufbereitung

Bevor ein Vergleich der Algorithmen `FastICA`, `InfoMax` und `JADE` angestellt werden kann, sind die Daten gemäß der in Abschnitt 2.4 betrachteten generalisierten Procrustes-Analyse sowie ein anschließendes *Whitening* aufzubereiten. Das Ziel der Procrustes-Analyse ist, wie in Abschnitt 2.4 bereits beschrieben, die Anpassung der *Shapes* resultierend in einer einheitlichen Repräsentation. Dazu wird im Folgenden der Algorithmus der generalisierten Procrustes-Analyse nach *Gower* (1975) in Verbindung mit der Modifikation nach *Ten Berge* (1977) verwendet. Zu Beginn des Algorithmus werden die *Shapes* zur Entfernung der Translation zentriert (Gower 1975, S. 40)

$$\mathbf{X}_i^P = \mathbf{C}\mathbf{X}_i, \quad i = 1, ..., n, \tag{4.1}$$

wobei $\mathbf{C}$ Zentrierungsmatrix ist mit

$$\mathbf{C} = \mathbf{I}_k - \frac{1}{k}\mathbf{1}_k\mathbf{1}_k^T.$$

So zeigt Abbildung 4.2 exemplarisch die Gesamtheit der *Shapes* des Datensatzes `digit3.dat` sowohl vor als auch nach der Zentrierung der *Shapes*.

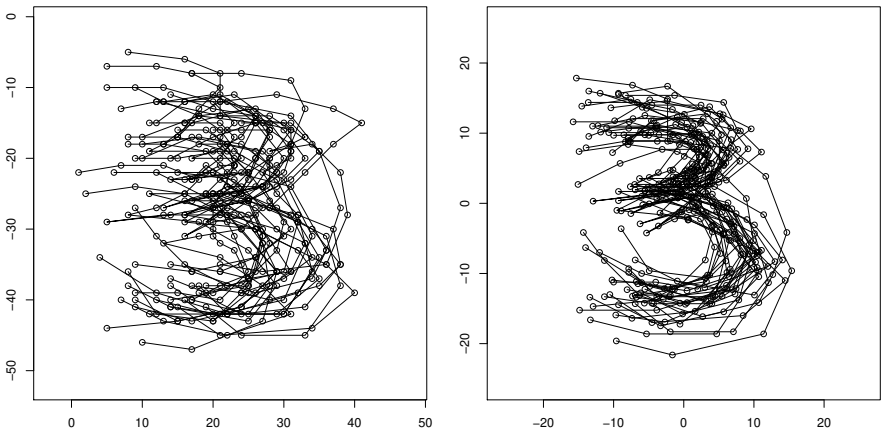


Abbildung 4.2: *Shapes* vor *(links)* und nach *(rechts)* der Zentrierung (Datensatz: `digit3.dat`)

Im Anschluss an jene Zentrierung der *Shapes* zur Entfernung der Translation erfolgt die Bereinigung der Rotations- sowie Skalierungseffekte in einem iterativen Verfahren. Dabei

wird zuerst die Rotation der *Shapes* betrachtet. Für die i-te Konfiguration sei dazu

$$\bar{\mathbf{X}}_{(i)} = \frac{1}{n-1} \sum_{j \neq i} \mathbf{X}_j^P, \tag{4.2}$$

dann ist die neue i-te Konfiguration $\mathbf{X}_i^P$ durch eine gewöhnliche Procrustes-Analyse lediglich unter Berücksichtigung der Rotation mit der alten i-ten Konfiguration $\mathbf{X}_i^P$ und $\bar{\mathbf{X}}_{(i)}$ gegeben.[1] Dieser Rotationsschritt wird wiederholt bis die folgende Quadratsumme

$$\sum_{i=1}^{n} \|\mathbf{X}_i \hat{\mathbf{\Gamma}}_i - \bar{\mathbf{X}}\|^2 = \left(\frac{n-1}{n}\right)^2 \left\| \mathbf{X}_i \hat{\mathbf{\Gamma}}_i - \frac{1}{n-1} \sum_{i \neq j} \mathbf{X}_j \hat{\mathbf{\Gamma}}_j \right\|^2 \tag{4.3}$$

geringer ist als eine zuvor definierte Toleranzschwelle (Gower 1975, S. 41f.). Diese Toleranzschwelle liegt im Rahmen dieser Analyse bei $b_1 = 10^{-5}$. Für den Datensatz `digit3.dat` ergaben sich drei Iterationen im Kontext der Rotation, wohingegen für den Datensatz `mice` vier Iterationen erforderlich waren.
Im Anschluss an die Rotation erfolgt die Skalierung der *Shapes*. Sei dazu $\mathbf{\Theta}$ die $n \times n$ Korrelationsmatrix des *Shape*-Vektors $\mathbf{x}_i^P = \text{vec}(\mathbf{X}_i^P) = (x_1, ..., x_k, y_1, ..., y_k)^T$

$$\mathbf{\Theta} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T \tag{4.4}$$

mit

$$\mathbf{P}\mathbf{P}^T = \mathbf{I},$$

wobei $\mathbf{P} = (\mathbf{p}_1, ..., \mathbf{p}_n)^T$ die Matrix der Eigenvektoren $\mathbf{p}_i$ und $\mathbf{\Lambda}$ die Diagonalmatrix mit korrespondierenden Eigenwerten λ_i ist.Dann folgt nach *Ten Berge (1977)* für den Skalierungsparameter

$$\hat{\beta}_i = \left(\frac{\sum_{j=1}^{n} \|\mathbf{X}_j^P\|^2}{\|\mathbf{X}_i^P\|^2} \right)^{\frac{1}{2}} \mathbf{p}_{i1}, \tag{4.5}$$

wobei $\mathbf{p}_{i1}$ der Eigenvektor zum jeweilig höchsten Eigenwert λ_{i1} für $i = 1, ..., n$ ist (Ten Berge, J. M. F. 1977, S. 275). Dieser Rotations- sowie Skalierungsschritt sind zu wiederholen, bis die Quadratsumme in Formel (2.7) geringer ist als eine weitere zuvor definierte

[1] Für eine ausführliche Betrachtung der gewöhnlichen Procrustes-Analyse siehe bspw. Dryden & Mardia 2016, S.126-134.

Toleranzschwelle, die hier ebenfalls bei $b_2 = 10^{-5}$ liegt. So waren für diesen Rotations- und und Skalierungsschritt für beide Datensätze 2 Interationen erforderlich.
Anschließend ist die *mittlere Shape* gemäß Formel (2.13) auf Basis der Procrustes-Koordinaten zu bestimmen

$$\bar{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{X}_i^P.$$

In der folgenden Abbildung 4.3 sind exemplarisch die im Rahmen der generalisierten Procrustes-Analyse angeglichenen *Shapes* sowie die *mittlere Shape* des Datensatzes `digit3.dat` dargestellt.

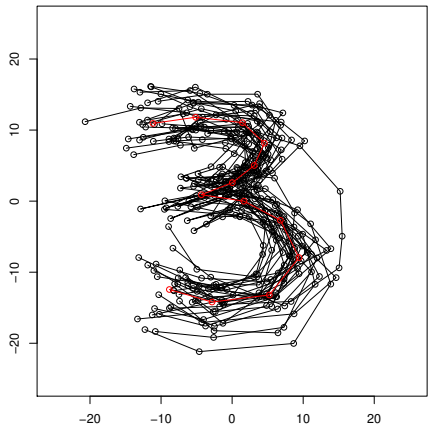


Abbildung 4.3: Angeglichene *Shapes* und *mittlere Shape (rot)* (Datensatz: `digit3.dat`)

Durch diese generalisierte Procrustes-Analyse sind die Mengen der *Shapes* des Datensatzes `digit3.dat` in einen *Shape Space* Σ_2^{13} bzw. des Datensatzes `mice` in einen *Shape Space* Σ_2^6 gebracht worden.
Neben der Procrustes-Analyse ist ein in Abschnitt 3.3.2 beschriebenes *Whitening* der Daten als Vorverarbeitungsschritt im Rahmen der *Independent Component Analysis* erfolgt, dabei wurden allerdings nicht die Procrustes-Koordinaten sondern vielmehr die Prokrustes-Residuen $d\mathbf{x} = \mathbf{x}^P - \bar{\mathbf{x}}$ zwischen den *Shapes* und der *mittleren Shape* betrachtet. Die lineare Transformation im Zuge des *Whitening* ist dann analog zu Formel (3.15) durch eine Diagonalisierung der Kovarianzmatrix von $d\mathbf{x}$ gegeben

$$\mathbf{z} = \mathbf{Q}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{Q}^T d\mathbf{x}$$

mit

$$E(\mathbf{z}\mathbf{z}^T) = \mathbf{I}.$$

Nach eben jener Datenaufbereitung werden die in Abschnitt 3.6 eingeführten Algorithmen der *Independent Component Analysis* – `FastICA`, `InfoMax`, `JADE` – zur Bestimmung der unabhängigen Komponenten verwendet. Um die Struktur der Variabilität der *Shapes* analysieren zu können, wird im folgenden Abschnitt zuvor jedoch eine Modifikation des *Point Distribution Model* eingeführt.

4.3 Point Distribution Model unter Verwendung der ICA

In Abschnitt 2.4 wurde das *Point Distribution Model* bereits als statistisches Modell einer *Shape* sowie deren Variationen in einem Datensatz eingeführt. Als klassische Methode zur Betrachtung der Variationen wird dabei im Rahmen des generativen Modells in Formel (2.23) die *Pricipal Component Analysis* verwendet. In diesem Abschnitt soll nun hingegen eine Modifikation des *Point Distribution Model* unter Verwendung der *Independent Component Analysis* zur Betrachtung eben jener Variationen eingeführt werden. Dazu seien zuvor die Variationen der *Shapes* durch Punktwolken der einzelnen *Landmarks* für die zu betrachtenden Datensätze in der folgenden Abbildung dargestellt.

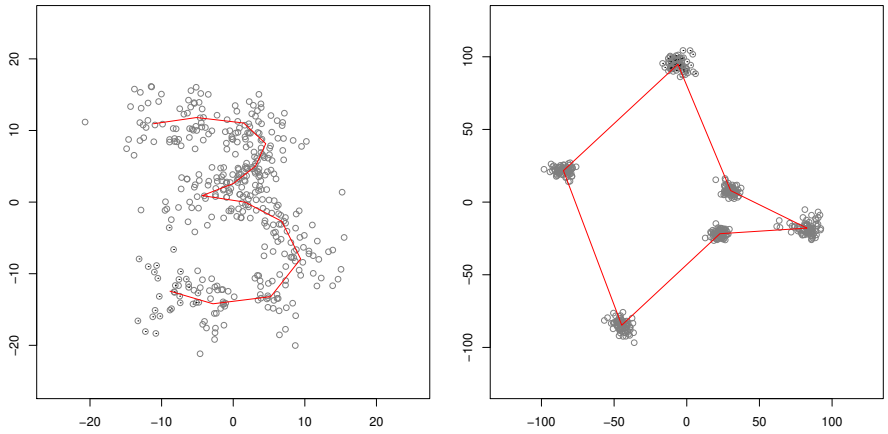


Abbildung 4.4: Punktwolken der *Landmarks* im Datensätze `digit3.dat` *(links)* und `mice` *(rechts)* mit *mittlerer Shape (rot)*

Sei $\mathbf{x} \in \mathbb{R}^{2k}$ ein *Shape*-Vektor bestehend aus k Paaren von (x_i^P, y_i^P) Koordinaten der einzel-

nen *Landmarks*

$$\mathbf{x}_i^P = \left(x_{i1}^P, ..., x_{ik}^P, y_{i1}^P, ..., y_{ik}^P\right)^T$$

mit $i = 1, ..., n$. Dann ist die *mittlere Shape* nach vorausgegangener Procustes-Analyse gegeben durch

$$\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i^P.$$

Ferner lassen sich die Procrustes-Residuen $d\mathbf{x}_i$ als Differenz zwischen den *Shapes* $\mathbf{x}_i^P$ und der *mittleren Shape* $\bar{\mathbf{x}}$ ermitteln

$$d\mathbf{x}_i = \mathbf{x}_i - \bar{\mathbf{x}}.$$

Für die Kovarianzmatrix der Procrustes-Residuen ergibt sich somit

$$\begin{aligned} \mathbf{\Sigma} &= \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i^P - \bar{\mathbf{x}})(\mathbf{x}_i^P - \bar{\mathbf{x}})^T \\ &= \frac{1}{n} \sum_{i=1}^{n} d\mathbf{x}_i d\mathbf{x}_i^T \end{aligned}$$

bzw. für die geweißten Procrustes-Residuen

$$\mathbf{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{z}_i \mathbf{z}_i^T. \tag{4.6}$$

Somit kann analog zu Formel (2.23) eine *Shape* $\mathbf{x}$ durch eine Linearkombination der *mittleren Shape* und den unabhängigen Komponenten $\mathbf{\Phi} \, \epsilon \, \mathbb{R}^{2k \times p}$ der *Shape*-Variationen dargestellt werden. Das generative Modell lautet dabei wiefolgt

$$\mathbf{x} = \bar{\mathbf{x}} + c \, \mathbf{\Phi} \mathbf{b} \tag{4.7}$$

mit $\bar{\mathbf{x}}$ *mittlere Shape*, $\mathbf{b} \, \epsilon \, \mathbb{R}^p$ Gewichtungsvektor sowie $c \, \epsilon \, \mathbb{R}$ Skalierungsparameter (Suinesiaputra *u. a.* 2004, S. 77). Da eine natürliche Ordnung der Komponenten analog zur *Principal Component Analysis* wie in Abschnitt 3.7 bereits gezeigt im Rahmen der *Independent Component Analysis* nicht existiert, hängen im Zuge derer die Wahl sowie die

Ordnung der unabhängigen Komponenten vielmehr von der Zielsetzung der *Statistical Shape Analysis* ab (Suinesiaputra *u. a.* 2004, S. 741).[2] Somit werden in dieser Betrachtung die Modelle auf Basis der einzelnen Komponenten miteinander verglichen, sodass der Gewichtungsvektor $\mathbf{b}$ jener Vektor mit $b_i = 1$ und 0 sonst für $i = 1, ..., p$ zur Auswahl des i-ten Komponentenvektors ist.
Zur Spezifikation eines *Point Distribution Model* auf Basis der *Independent Component Analysis* ist zudem eine Spezifikation des ICA Algorithmus erforderlich. Im Rahmen der Analyse gilt bezüglich der Ausführungen in Abschnitt 3.6

- `FastICA`
  - Approximation der Negentropie durch $J(\gamma) \propto [E(G(\gamma)) - E(G(\nu))]^2$ mit $G_1(u) = \log\cosh(u)$
  - parallele Schätzung
- `InfoMax`
  - Aktivierungsfunktion $g(u_i) = \frac{1}{1+\exp(-u_i)}$
  - Lernrate $\eta = 1$

Eine Spezifikation des `JADE` Algorithmus ist aufgrund der lediglichen Verwendung von Matrixoperationen nicht erforderlich. Bezüglich der Konvergenz der Algorithmen wurde ein Schwellenwert von $b = 10^{-6}$ im Rahmen der Veränderungen der Iterationen festgelegt sowie die maximale Anzahl der Iterationen auf 400 beschränkt.

4.4 Ergebnisse

Nach der Spezifikation des *Point Distribution Model* werden nun die Ergebnisse der Analyse vorgestellt. Dazu wurde zuerst ein computationaler Vergleich der Algorithmen `FastICA`, `InfoMax` sowie `JADE` insbesondere zur Betrachtung der computaitonalen Eigenschaften der Optimierungsalgorithmen angestellt. Neben diesem computationalen Vergleich wurden ferner die Variabilitätsstrukturen der *Shapes* durch ein *Point Distribution Model* unter Verwendung der verschiedenen Algorithmen miteinander verglichen.

[2] Es existieren bisweilen unterschiedliche Ansätze zur Ordnung der Komponenten, ein einheitliches Kriterium hat sich dabei in der Literatur jedoch nicht durchgesetzt.

4.4.1 Computationaler Vergleich

Im Rahmen dieses Vergleichs wurden insbesondere die computationalen Eigenschaften der unterschiedlichen Optimierungsalgorithmen der Methoden `FastICA`, `InfoMax` sowie `JADE` untersucht. Die hier zu betrachteten computationalen Eigenschaften umfassen dabei die Konvergenzgeschwindigkeit der Algorithmen sowie die Komplexität deren Rechenoperationen in Form der benötigten Rechenzeit zur Schätzung der unabhängigen Komponenten der Variationen auf Basis der Datensätze `digit3.dat` sowie `mice`.
Die folgende Abbildung 4.5 zeigt die Anzahl der benötigten Iterationen der Algorithmen für den Datensatz `digit3.dat`.

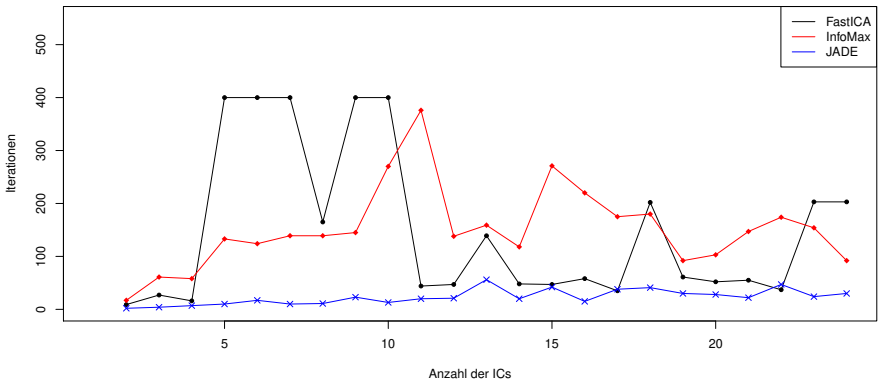


Abbildung 4.5: Anzahl der Iterationen der Algorithmen (Datensatz: `digit3.dat`)

So verdeutlicht die Abbildung die Vorteile des `JADE` und dessen Optimierungsalgorithmus hinsichtlich der Konvergenzgeschwindigkeit gegenüber den anderen beiden Algorithmen. Neben der hohen Konvergenzgeschwindigkeit weist dieser eine hohe Robustheit hinsichtlich der Konvergenz auf. Diese Superiorität ist möglicherweise durch die nicht-zufällige Initialisierung der Gewichtungsmatrix bzw. der Nicht-Verwendung eines Lernparameters im Zuge des Optimierungsalgorithmus begründet. Der `FastICA` Algorithmus zeigt hingegen gerade für geringe Anzahlen zu schätzender Komponenten Schwächen bezüglich der Konvergenz, so konvergiert dieser in 5 der betrachteten Fälle nicht, was möglicherweise der zufälligen Initialisierung der Gewichtungsmatrix geschuldet ist. Bei hohen Anzahlen zu bestimmender Komponenten weist dieser jedoch ähnlich schnelle Konvergenzgeschwindigkeiten wie der `JADE` Algorithmus auf. Der `InfoMax` Algorithmus besitzt abgesehen der Fälle, in denen der `FastICA` nicht konvergiert, zumeist geringere Konvergenzgeschwindigkeiten im Vergleich zu den anderen Algorithmen. Entgegen dem Fixpunktalgorithmus des

`FastICA` ist das Gradientenverfahren im Zuge des `InfoMax` bezüglich der Konvergenz jedoch robuster.
Im Rahmen des Datensatzes `mice` zeichnet sich ein ähnliches Bild für die benötigten Konvergenzgeschwindigkeit der Algorithmen ab, die in der Abbildung 4.6 dargestellt sind.

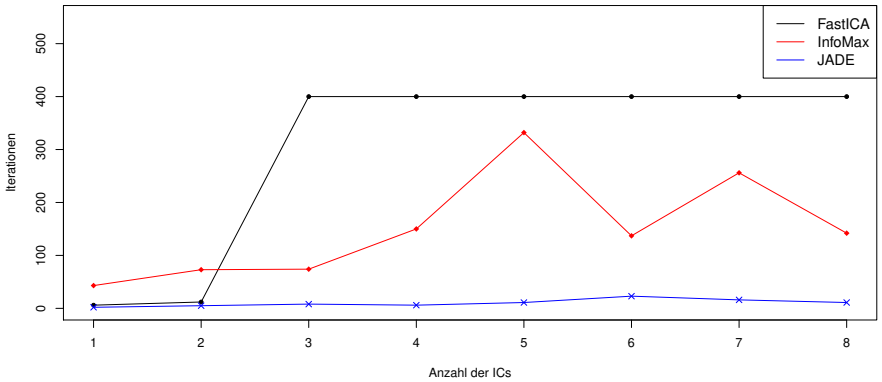


Abbildung 4.6: Anzahl der Iterationen der Algorithmen (Datensatz: `mice`)

Analog zur vorangegangenen Betrachtung belegt auch diese eine hohe Konvergenzgeschwindigkeit des `JADE` Algorithmus sowie Robustheit bezüglich der Konvergenz im Vergleich jenen anderen. Der `FastICA` zeigt wiederum auch hier Schwächen bezüglich der Konvergenz, so konvergiert der Algorithmus lediglich in zwei der betrachteten Fälle; wohingegen der `InfoMax` ebenfalls hier robust hinsichtlich der Konvergenz ist, aber gleichsam eine geringere Konvergenzgeschwindigkeiten aufweist als der `JADE`.
Um hingegen neben der Konvergenzgeschwindigkeit die Komplexität der Rechenoperationen zu betrachten, wurde zudem die benötigte Rechenzeit der Algorithmen zur Bestimmung der unabhängigen Komponenten der *Shape*-Variationen gemessen. Als Maß der Komplexität wurde dabei das arithmetische Mittel der Rechenzeit aus 100 Wiederholungen der jeweiligen Algorithmen verwendet. In Abbildung 4.7 sind die durchschnittlichen Rechenzeiten der Algorithmen hinsichtlich des Datensatzes `digit3.dat` dargestellt. Dies veranschaulicht die in Abschnitt 3.6.3 beschriebenen Nachteile der simultanen Diagonalisierung der Kumulantenmatrizen in hoch-dimensionalen Räumen im Rahmen des `JADE` Algorithmus, so benötigt dieser ab eine Anzahl von 11 zu schätzenden Komponenten trotz der hohen Konvergenzgeschwindigkeit eine deutlich höhere Rechenzeit verglichen mit den anderen Algorithmen.
Aufgrund der deutlichen Diskrepanz hinsichtlich der Rechenzeit werden in Abbildung 4.8

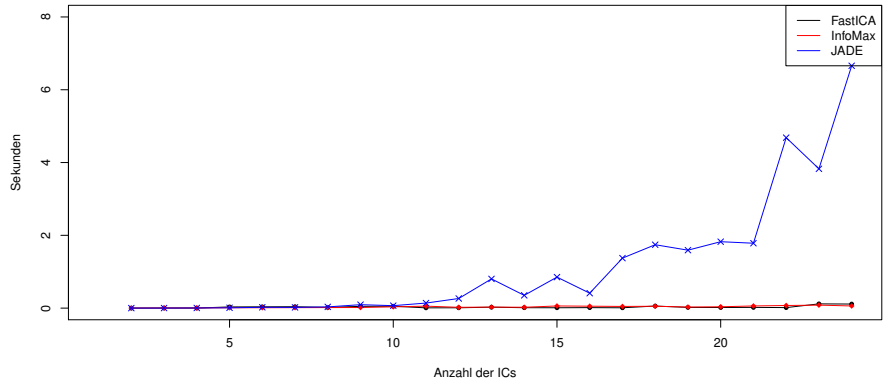


Abbildung 4.7: Rechenzeit der Algorithmen (Datensatz: `digit3.dat`)

lediglich die durchschnittlichen Rechenzeiten des `FastICA` sowie des `InfoMax` zu einer besseren Differenzierung betrachtet. Die Abbildung zeigt dabei die in Abschnitt 3.6.1 bereits

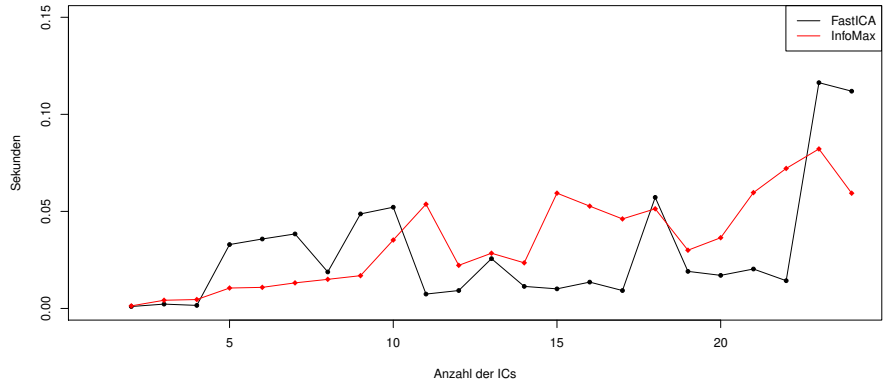


Abbildung 4.8: Rechenzeit des `FastICA` sowie des `InfoMax` (Datensatz: `digit3.dat`)

beschrieben Vorteile des `FastICA` durch die Verwendung eines Fixpunktalgorithmus zur Optimierung der Kontrastfunktion. So weist der `FastICA` Algorithmus abgesehen der Betrachtungen, in denen der Algorithmus nicht konvergiert, aufgrund der parallelen Architektur zumeist geringere Rechenzeit auf als der `InfoMax` Algorithmus. Zudem scheint die Rechenzeit des `FastICA` trotz wachsender Anzahl zu schätzender Komponenten annähernd konstant zu bleiben, wohingegen für den `InfoMax` ein Anstieg der Rechenzeit zu beobachten ist.

Die Betrachtung der benötigten durchschnittlichen Rechenzeit auf Basis des Datensatzes

`mice` zeigt hingegen aufgrund der geringen Anzahl zu schätzender Komponenten der Variation der *Shapes* ein weniger differenziertes Bild. So weisen die Algorithmen hier allesamt

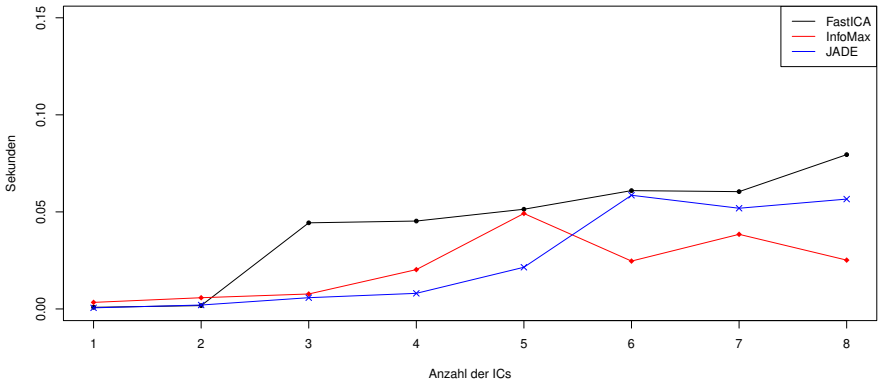


Abbildung 4.9: Rechenzeit der Algorithmen (Datensatz: `mice`)

vergleichbare Rechenzeiten auf, wobei die höheren Rechenzeiten des `FastICA` Algorithmus durch die fehlenden Konvergenzen begründet sind. Eine deutliche Erhöhung der Rechenzeit des `JADE` ist im Rahmen dieser Betrachtung nicht zu beobachten, sodass die Dimensionen der Räume für eine deutliche Zunahme der Rechenzeit nicht ausreichend hoch zu sein scheint.

Dieser computationale Vergleich zeigt zugleich die Vorteile der simultanen Diagonalisierung der Kumulantenmatrizen im Rahmen des `JADE` durch schnelle Konvergenz des Algorithmus als auch deren Nachteile bei Zunahme der Dimension des zu betrachtenden Raumes durch eine deutlich höhere Rechenzeit im Vergleich zu den anderen Algorithmen. Der `FastICA` sowie der `InfoMax` stellen hingegen einen Kompromiss zwischen Konvergenzgeschwindigkeit und Komplexität der Rechenoperationen dar. So scheint der `FastICA` gerade in Räumen mit zunehmender Dimension sowohl hinsichtlich der Konvergenzgeschwindigkeit als auch der benötigten Rechenzeit Vorteile gegenüber dem `InfoMax` zu besitzen, ist aber gerade in Räumen mit geringer Dimension weniger robust hinsichtlich der Konvergenz als dieser.[3]

4.4.2 Vergleich der Variabilität der Shapes

Neben jenem computationalen Vergleich wurde die Variation der *Shapes* durch die in Abschnitt 4.3 eingeführte Modifikation des *Point Distribution Model* unter Verwendung der

[3]Für eine tabellarische Zusammenfassung der Ergebnisse siehe Anhang A.

Algorithmen `FastICA`, `InfoMax` und `JADE` betrachtet. Dabei wurde die Variabilitätsstruktur der durch das jeweilige *Point Distribution Model* generierten *Shape* zwischen jenen Algorithmen verglichen, um diese somit auf strukturelle Unterschiede zu untersuchen.
Neben diesem strukturellen Vergleich der Variabilität wurde nach *Dryden/Mardia* (2016) das quadratische Mittel der Procrustes-Distanz (siehe Formel (2.6)) $RMS(d_F(\cdot,\cdot))$ zur *mittleren Shape* $\bar{\mathbf{x}}$ als Maß für die globale Variabilität der *Shapes* verwendet (Dryden & Mardia 2016, S. 160), um neben möglichen lokalen ebenso globale Unterschiede zu erfassen. Das quadratische Mittel der Procrustes-Distanz $RMS(d_F(\cdot,\cdot))$ ist dabei gegeben durch

$$RMS(d_F(\mathbf{x},\bar{\mathbf{x}})) = \sqrt{\frac{1}{2k}\sum_{i=1}^{2k}(d_F(\mathbf{x},\bar{\mathbf{x}}))^2}. \tag{4.8}$$

Da dieses globale Variabilitätsmaß invariant gegenüber der Größe der *Shapes* ist, wird zu einer Untersuchung der Größe die Distanz der generierten *Shapes* zum Schwerpunkt *(Centroid Size)* betrachtet, die bereits in Formel (2.9) wiefolgt definiert wurde

$$S(\mathbf{x}) = \sqrt{\sum_{j=1}^{k}(x_j-\bar{x})^2+(y_j-\bar{y})^2}.$$

Aufgrund des Fehlens einer natürlichen Ordnung der Komponenten im Rahmen der *Independent Component Analysis* werden vielmehr Komponententriple zwischen den Algorithmen betrachtet, die auf Basis des Spearman-Korrelationskoeffizientens ermittelt wurden. Damit konnten die Variabilitätsstrukturen der generierten *Shapes* sowohl zwischen den einzelnen Komponenten als auch zwischen den zu betrachtenden Algorithmen verglichen werden. Um die unterschiedlichen Ziele der *Statistical Shape Analysis* in Verbindung mit der *Independent Component Analysis* darzustellen, werden im Folgenden exemplarisch die Ergebnisse zweier Betrachtungen dargestellt, die sich hinsichtlich der Anzahl zu bestimmender Komponenten unterscheiden.
Die erste Betrachtung umfasst ein *Point Distribution Model* in Rahmen dessen 2 unabhängige Komponenten anhand der ICA Algorithmen geschätzt wurden. Die Variabilitätsstruktur der *Shapes* auf Basis des Datensatzes `digit3.dat` ist für die jeweiligen Komponenten in Abbildung 4.10 abgetragen. Dabei sind die generierten *Shapes* mit $c=-1$ $(-\star-)$ bzw. mit $c=+1$ $(-+-)$ dargestellt, die *mittlere Shape* $(-\bullet-)$ ist zudem als Referenz eingezeichnet. Die Betrachtung zeigt dabei im Rahmen aller Modelle eine globale Variabilitätsstruktur

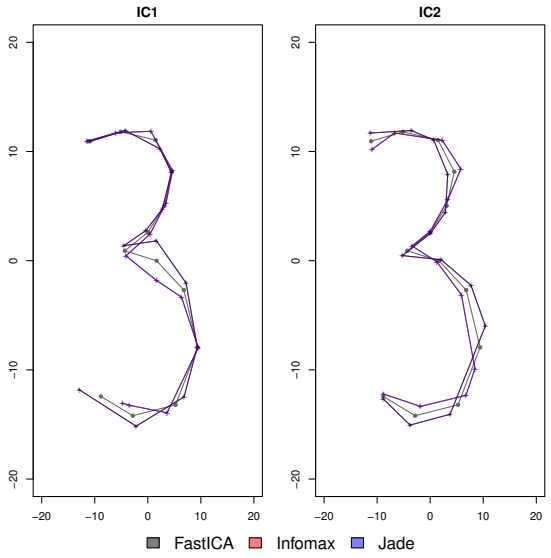


Abbildung 4.10: Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 2 ICs (Datensatz: `digit3.dat`)

der *Shapes*, die nicht auf einzelne *Landmarks* konzentriert ist. Dabei beschreibt die erste Komponente insbesondere die horizontale Variationen des *Landmark* 1 sowie die vertikale Variationen der *Landmarks* 2 und 9, wohingegen die zweite Komponente hauptsächlich die vertikale Variationen der Landmark-Positionen $2, 3, 4$ erklärt. Bei einem Vergleich zwischen den Algorithmen scheinen die generierten *Shapes* nahezu identisch hinsichtlich der Variabilitätsstruktur zu sein, jedoch wird bei Betrachtung der Struktur eine in Abschnitt 3.7 bereits angesprochenen Limitationen der *Independent Component Analysis* deutlich. So besitzen die Komponenten auf Basis des `FastICA` und jene basiernde auf dem `InfoMax` sowie `JADE` entgegengesetzte Richtungen.[4]

Analog dazu sind die generierten *Shapes* mit $c = \pm 1$ für beide Komponenten auf Basis des Datensatzes `mice` in Abbildung 4.11 dargestellt. Dabei ist ebenfalls eine globale Struktur der Variabilität der *Shapes* zu beobachten, obgleich diese im Rahmen dieses Datensatzes eher gering ist. Durch die erste Komponente wird indes die Variabilität bezüglich der *Landmark*-Positionen $1, 2$ sowie des Dornfortsatzes (*Landmark* 4) erklärt, wohingegen durch die zweite Komponente die Variationen der *Landmarks* $1, 2, 4, 6$ beschrieben wird. Der Vergleich zwischen den Algorithmen zeigt hier ebenfalls eine nahezu identische Struktur der *Shapes*, wobei hier die Komponentenvektoren in dieselbe Richtung zeigen.

[4]Die Richtungen der Komponentenvektoren sind in Abhang C dargestellt.

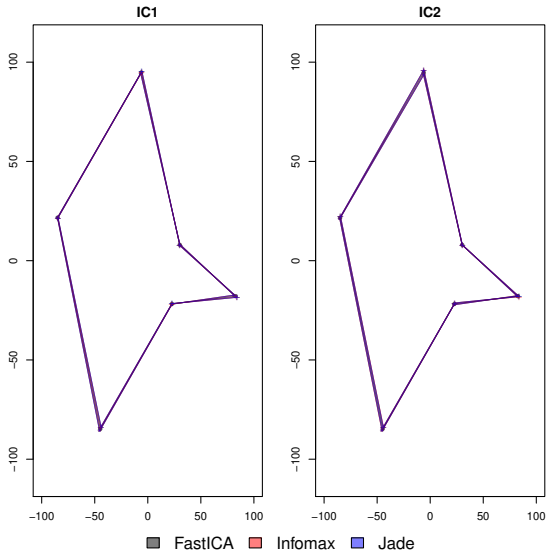


Abbildung 4.11: Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 2 ICs (Datensatz: `mice`)

Entgegen der ersten Betrachtung, in der 2 Komponenten durch die ICA Algorithmen geschätzt wurden, umfasst die zweite Betrachtung ein *Point Distribution Model*, in Rahmen dessen 22 unabhängige Komponenten der Variationen der *Shapes* des Datensatzes `digit3.dat` sowie 8 unabhängige Komponenten jener Variationen im Datensatz `mice` zu bestimmen sind. Die generierten *Shapes* des Datensatzes `digit3.dat` mit $c = \pm 1$ sind hier exemplarisch für 4 Komponenten in Abbildung 4.12 dargestellt. Die Abbildung zeigt eine nun deutlichere lokale Differenzierung der Variabilität der *Shapes* im Rahmen der einzelnen Komponenten entgegen der globalen Variation in der ersten Betrachtung; so scheint eine wachsende Anzahl an Komponenten zunehmend zur Erklärung einzelner lokaler Variationen durch die jeweiligen Komponenten zu führen. Dabei erklärt die erste Komponente beispielsweise die horizontale Variabilität des *Landmark* 1, wohingegen die dritte Komponente insbesondere die vertikale Variation des *Landmark* 9 beschreibt. Ein Vergleich zwischen den Algorithmen zeigt lediglich marginale lokale Diskrepanzen, strukturelle Unterschiede der *Shape*-Variationen sind ausgenommen erneut entgegengesetzter Richtungen der Komponentenvektoren hingegen nicht zu beobachten.

Für die generierten *Shapes* des Datensatzes `mice` mit $c = \pm 1$ sind weiterhin globale Strukturen der Variabilität zu beobachten, die in Abbildung 4.13 ebenso exemplarisch durch 4 Komponenten abgebildet sind. Diese weiterhin globale Variabilitätsstruktur der *Shapes* in

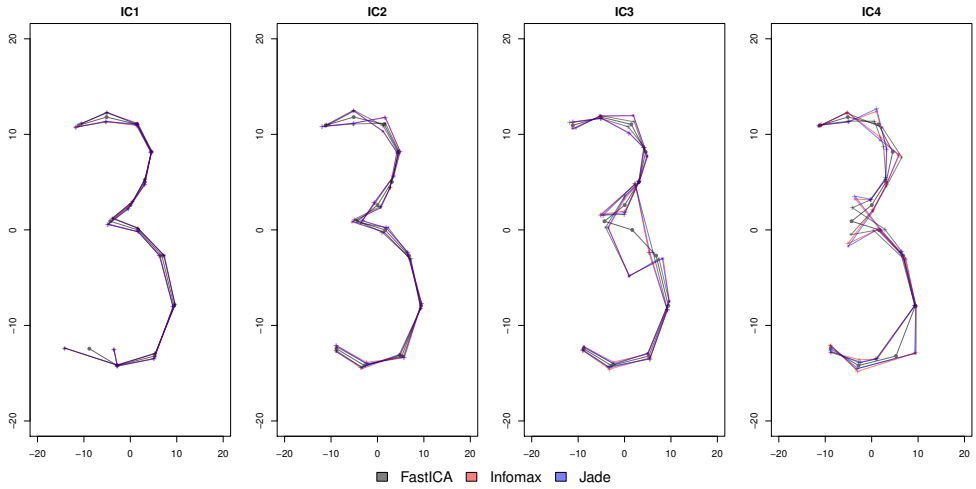


Abbildung 4.12: Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 22 ICs (Datensatz: `digit3.dat`)

diesem Datensatz ist möglicherweise durch die geringe Anzahl an *Landmarks* sowie geringe Variationen der *Shapes* zu begründen, wobei insbesondere die Komponenten 2 und 4 eine etwas stärkere lokale Ausprägung der Variabilität erklären. Die Variabilitätsstruktur der generierten *Shapes* ist auch hier zwischen den Algorithmen erneut sehr ähnlich und weist ebenso ausgenommen entgegengesetzter Richtungen der Komponentenvektoren lediglich marginale lokale Unterschiede auf.

Somit zeigen die Ergebnisse dieses strukturellen Vergleichs der *Shape*-Variationen bisweilen eine Abhängigkeit des Analyseziels der *Statistical Shape Analysis* unter Verwendung der *Independent Component Analysis* und der Anzahl der zu schätzender unabhängiger Komponenten. Bei einer geringen Anzahl an Komponenten werden globale Variationen der *Shape* ähnlich einer *Principal Component Analysis* durch die Komponenten beschrieben, während diese mit zunehmender Anzahl an zu schätzenden Komponenten stärker lokal konzentriert im Rahmen der einzelnen Komponenten sind. Die Variabilitätsstruktur der *Shapes* im Vergleich zwischen den Algorithmen weist abgesehen der Richtungen der Komponenten eine hohe Ähnlichkeit auf, bei der lediglich marginale lokale jedoch keine strukturellen Diskrepanzen zu beobachten sind.

Dieses Ergebnis wird zusätzlich durch die Betrachtung des globalen Variabilitätsmaßes in Form des quadratischen Mittels der Procrustes-Distanz $RMS(d_F(\mathbf{x}_{IC}, \bar{\mathbf{x}}))$ sowie der Größe der *Shape* auf Basis der *Centroid Size* bestätigt. Die Ergebnisse des globalen Variabilitätsmaßes sowie der *Centroid Size* sind für eine generierte *Shape* des Datensatzes `digit3.dat`

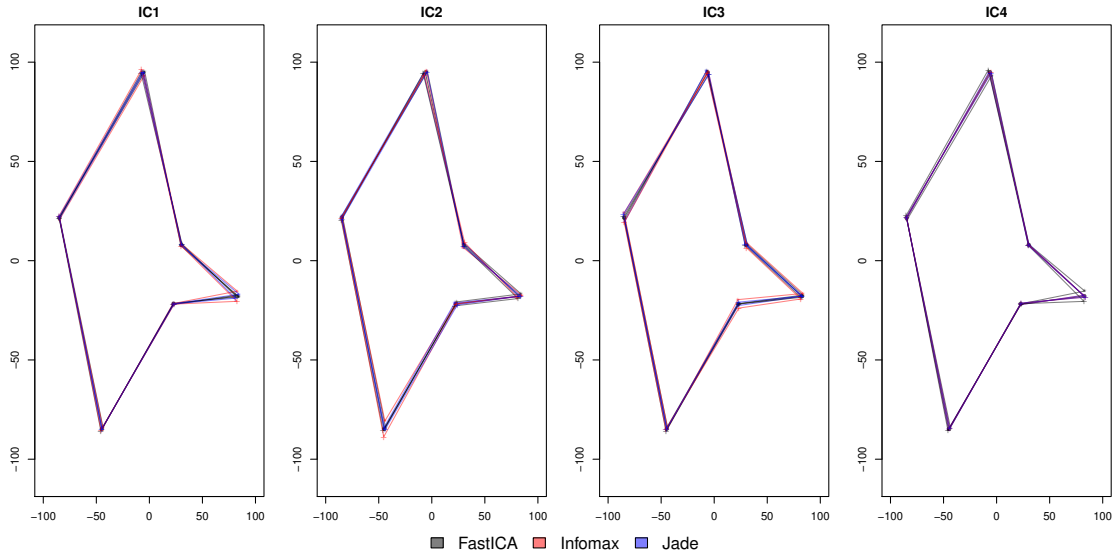


Abbildung 4.13: Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 8 ICs (Datensatz: `mice`)

mit $c = +1$ in Tabelle 4.3 dargestellt. Die *Centroid Size* der *mittleren Shape* beträgt dabei $S(\bar{\mathbf{x}}) = 39.16213$.

Insgesamt ist eine geringe globale Variabilität der *Shapes* im Rahmen beider Betrachtung zu verzeichnen, was eine hohe Ähnlichkeit der generierten *Shapes* zur *mittleren Shape* bedeutet. Hinsichtlich der Größe ist die Variation der *Centroid Size* der generierten *Shapes* ebenfalls gering und weist keine deutlichen Abweichungen von jener der *mittleren Shape* auf. Analog zur Betrachtung der Variabilitätsstruktur sind hinsichtlich der globalen Variabilität der generierten *Shapes* sowie deren Größe lediglich marginale Unterschiede zwischen den Algorithmen zu beobachten, die zumeist durch entgegengesetzte Richtungen der jeweiligen Komponentenvektoren zu erklären sind. Die Umkehrung der Richtungen der betroffenen Komponentenvektoren würde ferner zu noch geringeren Diskrepanzen zwischen den Ergebnissen der Algorithmen führen.

Die Betrachtung eben jener Maße für die generierten *Shapes* des Datensatzes `mice` mit $c = 1$ zeigen eine noch geringere Variation, wie der Tabelle 4.4 zu entnehmen ist. Die *Centroid Size* der *mittleren Shape* beträgt für diesen Datensatz $S(\bar{\mathbf{x}}) = 187.0972$. Die Ergebnisse verdeutlichen dabei eine sowohl sehr geringe globale Variabilität der *Shapes* als auch eine sehr geringe Variabilität der Größe jener*Shapes*. Zwischen den Algorithmen sind ebenso lediglich marginale lokale Diskrepanzen zu verzeichnen.

Betrachtung	IC	$RMS(d_F(\mathbf{x}_{IC}, \bar{\mathbf{x}}))$			*Centroid Size* ($S(\mathbf{x}_{IC})$)		
		FastICA	InfoMax	JADE	FastICA	InfoMax	JADE
1	1	0.1288*	0.1325	0.1325	40.0755*	38.9456	38.9478
	2	0.1034*	0.1055	0.1055	39.7640*	38.9965	38.9934
2	1	0.1200*	0.1275	0.1276	40.7191*	38.2428	38.2522
	2	0.0424*	0.0521	0.0502	39.4845*	39.2588	39.3183
	3	0.1272	0.1334	0.1337	39.8900	39.5531	39.4959
	4	0.1283	0.1237	0.1267	38.7379	39.9694	39.1868
	5	0.1285*	0.1261	0.1257	38.8252*	40.4294	40.4580
	6	0.1275*	0.1342	0.1342	40.3867*	38.6893	38.6107
	7	0.1264	0.1282	0.1168*	37.6402	37.6836	41.2270*
	8	0.0388*	0.0473	0.0457	39.3125*	38.8366	38.7976
	9	0.1058	0.1195*	0.1269	38.7228	40.7083*	38.1217
	10	0.1172*	0.1308	0.1301	41.5875*	37.0274	37.0900
	11	0.1325*	0.1307	0.1321	39.5617*	39.6167	39.5040
	12	0.1340	0.1342	0.1374*	40.0471	39.9431	39.0858*
	13	0.1311*	0.1389	0.1387	40.4873*	38.4777	38.4574
	14	0.1261	0.1338*	0.1240	40.6240	38.2108*	40.8755
	15	0.1259*	0.1344	0.1350	40.4814*	38.9260	38.9822
	16	0.1345	0.1354	0.1349	38.4673	38.6230	38.5518
	17	0.1142	0.1258*	0.1142	41.3883	37.3281*	41.5152
	18	0.1237	0.1225	0.1221	40.7281	40.7362	40.7906
	19	0.1342*	0.1379	0.1376	40.0630*	38.8714	38.9207
	20	0.1350	0.0738	0.0736*	38.1643	38.6257	40.1356*
	21	0.1132	0.1293	0.1308	40.9298	40.3131	40.0640
	22	0.0527*	0.0471	0.0463	38.9906*	39.4676	39.3998

* Entgegengesetzte Richtung der Komponentenvektoren

Tabelle 4.3: Globale Variabilität sowie Größe der *Shapes* mit $c = 1$ (Datensatz: `digit3.dat`)

Betrachtung	IC	$RMS(d_F(\mathbf{x}_{IC}, \bar{\mathbf{x}}))$			*Centroid Size* ($S(\mathbf{x}_{IC})$)		
		`FastICA`	`InfoMax`	`JADE`	`FastICA`	`InfoMax`	`JADE`
1	1	0.0090	0.0082	0.0093	187.2564	187.1889	187.2774
	2	0.0070	0.0080	0.0067	187.6027	187.6194	187.5955
2	1	0.0116	0.0065	0.0073	187.0031	187.2600	187.3122
	2	0.0104	0.0098	0.0098*	187.1643	186.5016	187.7767
	3	0.0105	0.0130*	0.0131	187.1394	186.9669*	186.8240*
	4	0.0157*	0.0148	0.0143	188.8565*	188.8117	189.0696
	5	0.0175	0.0092	0.0088	187.0725	186.5637	186.8386
	6	0.0166	0.0132	0.0130	186.8614	186.9828	187.0601
	7	0.0098*	0.0085	0.0109	188.0939*	186.0499	186.8715
	8	0.0123	0.0237*	0.0231	186.0315	187.8959*	186.2966*

* Entgegengesetzte Richtung der Komponentenvektoren

Tabelle 4.4: Globale Variabilität sowie Größe der *Shapes* mit $c = 1$ (Datensatz: `mice`)

Kapitel 5

Fazit

Im Rahmen dieser Arbeit wurde die noch relativ junge Methode der *Independent Component Analysis* im Kontext der aktuell in vielen Bereichen bedeutenden *Statistical Shape Analysis* betrachtet. Dabei wurden zu Beginn dieser Arbeit sowohl die Grundlagen der *Statistical Shape Analysis* als auch die Grundlagen der *Independent Component Analysis* betrachtet. Im Zuge des anschließenden Vergleichs der Algorithmen `FastICA`, `InfoMax` sowie `JADE` konnte die *Independent Component Analysis* ihr Potenzial in der *Statistical Shape Analysis* andeuten.

In einem computationalen Vergleich in Form der Konvergenzgeschwindigkeit sowie der benötigten Rechenzeit als Maß der Komplexität der verwendeten Rechenoperationen wurde insbesondere die Leistungsfähigkeit der Optimierungsalgorithmen vor diesem Hintergrund verglichen. Dabei waren Vorteile des `JADE` Algorithmus bezüglich der Konvergenz zu beobachten, jedoch benötigte dieser aufgrund des simultanen Diagonalisierungsverfahrens insbesondere in höher-dimensionalen Räumen deutlich mehr Rechenzeit. Die Algorithmen `FastICA` sowie `InfoMax` zeigten im Rahmen dieses Vergleichs einen Kompromiss zwischen diesen beiden Untersuchungsmerkmalen.

Zu einem Vergleich hinsichtlich der Analyse von Variationen einer *Shape* wurde in Abschnitt 4.3 eine Modifikation des klassischen *Point Distribution Model* unter Verwendung der *Independent Component Analysis* eingeführt, auf Basis dessen unabhängige Komponenten zur Erklärung der Variation der *Shape* geschätzt werden. Somit wurden abhand des *Point Distribution Model* unter Verwendung der Algorithmen `FastICA`, `InfoMax` und `JADE` *Shapes* generiert, die anschließend bezüglich ihrer Variabilitätsstruktur verglichen wurden. Dabei konnte gezeigt werden, dass eine Abhängigkeit zwischen dem Analyseziel einer *Statistical Shape Analysis* und der Anzahl zu schätzender Komponenten besteht. So erklären

die einzelnen Komponenten im Falle einer geringen Anzahl von Komponenten eine globale Variation der *Shape* ähnlich zur *Principal Component Analysis*, wohingegen eine hohe Anzahl zu schätzender Komponenten zu lokalen Variationsstrukturen der einzelnen Komponenten führt. Damit ist die *Independent Component Analysis* ein geeignetes Mittel zur Betrachtung lokaler Deformationen einer *Shape*. Der Vergleich zwischen den Algorithmen wies große Ähnlichkeiten zwischen den Variabilitätsstrukturen der jeweilig erzeugten *Shape* auf, lediglich marginale lokale Diskrepanzen waren dabei zu verzeichnen. Diese Ähnlichkeit zwischen den Algorithmen wurde auch anhand des globalen Variabilitätsmaßes sowie der *Cemtroid Size* bestätigt. Somit ist für die Wahl eines ICA Algorithmus vielmehr der Optimierungsalgorithmus betreffend die computationalen Eigenschaften des Algorithmus entscheidend. Weiterführend können nun mögliche Methoden zur Ordnung bzw. Auswahl der unabhängigen Komponenten zu einer Verbesserung der Modelle untersucht werden. Des Weiteren ist eine mögliche Erweiterung des Modells in Form eines *Active Appearance Model* in Verbindung mit Online-ICA-Algorithmen zur dynamischen Erfassung von *Shapes* denkbar.

Anhang A

Computationaler Vergleich - Übersicht

In der folgenden Tabelle sind die Ergebnisse des computationalen Vergleichs zwischen den Algorithmen `FastICA`, `InfoMax` sowie `JADE` der *Independent Component Analysis* aus Abschnitt 4.4.1 auf Basis des Datensatzes `digit3.dat` zusammengefasst.

	`FastICA`		`InfoMax`		`JADE`	
Anzahl der ICs	Iterationen	Rechenzeit (in Sek.)	Iterationen	Rechenzeit (in Sek.)	Iterationen	Rechenzeit (in Sek.)
2	6	0.0016	89	0.0058	2	0.0006
3	400	0.0345	64	0.0048	8	0.0038
4	400	0.0421	140	0.0120	11	0.0076
5	400	0.0371	118	0.0129	11	0.0108
6	164	0.0168	115	0.0111	7	0.0115
7	126	0.0156	301	0.0349	8	0.0202
8	154	0.0205	308	0.0372	8	0.0228
9	41	0.0061	73	0.0113	8	0.0332
10	46	0.0078	124	0.0172	16	0.0907
11	77	0.0148	194	0.0376	9	0.0669
12	80	0.0157	218	0.0418	18	0.1810
13	40	0.0084	126	0.0235	24	0.3236
14	36	0.0075	316	0.0664	400	7.8256
15	53	0.0124	130	0.0286	22	0.5313
16	43	0.0115	80	0.0187	27	0.8670
17	52	0.0162	99	0.0273	39	1.7762
18	73	0.0247	76	0.0245	55	2.9762
19	36	0.0160	71	0.0219	18	1.1202
20	60	0.0224	161	0.0530	18	1.4262
21	49	0.0192	315	0.1129	42	4.8770
22	67	0.0283	180	0.0781	29	3.6447

Tabelle A.1: Computionaler Vergleich der Algorithmen (Datensatz: `digit3.dat`)

In der folgenden Tabelle sind die Ergebnisse des computationalen Vergleichs zwischen den Algorithmen `FastICA`, `InfoMax` sowie `JADE` der *Independent Component Analysis* aus Abschnitt 4.4.1 auf Basis des Datensatzes `mice` zusammengefasst.

	`FastICA`		`InfoMax`		`JADE`	
Anzahl der ICs	Iterationen	Rechenzeit (in Sek.)	Iterationen	Rechenzeit (in Sek.)	Iterationen	Rechenzeit (in Sek.)
2	11	0.0020	30	0.0027	2	0.0023
3	13	0.0035	34	0.0033	8	0.0095
4	400	0.0378	35	0.0038	12	0.0079
5	400	0.0404	91	0.0084	6	0.0068
6	56	0.0064	83	0.0085	14	0.0241
7	54	0.0064	58	0.0067	8	0.0193
8	128	0.0168	157	0.0223	32	0.1046

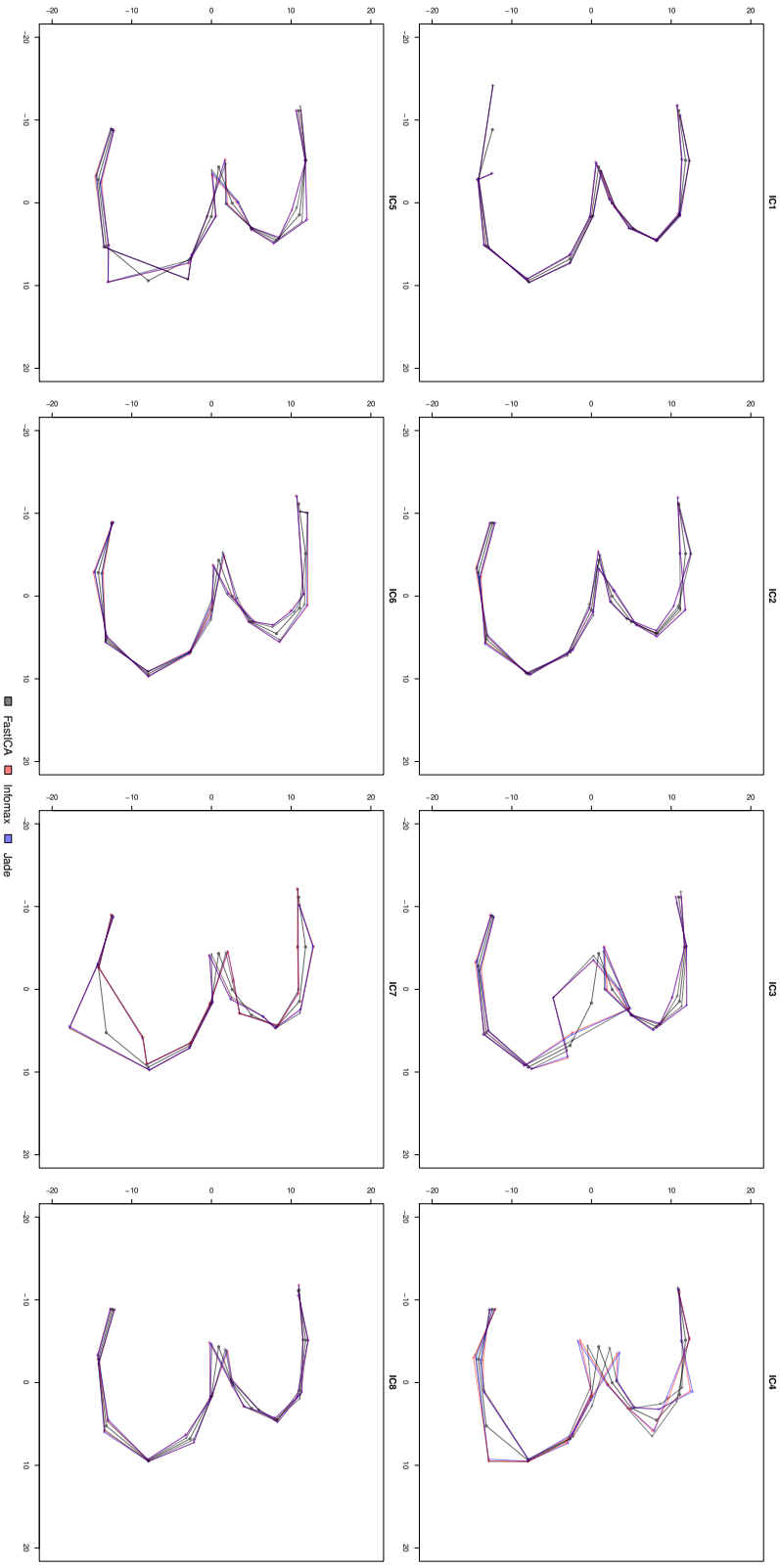
Tabelle A.2: Computionaler Vergleich der Algorithmen (Datensatz: `mice`)

Anhang B

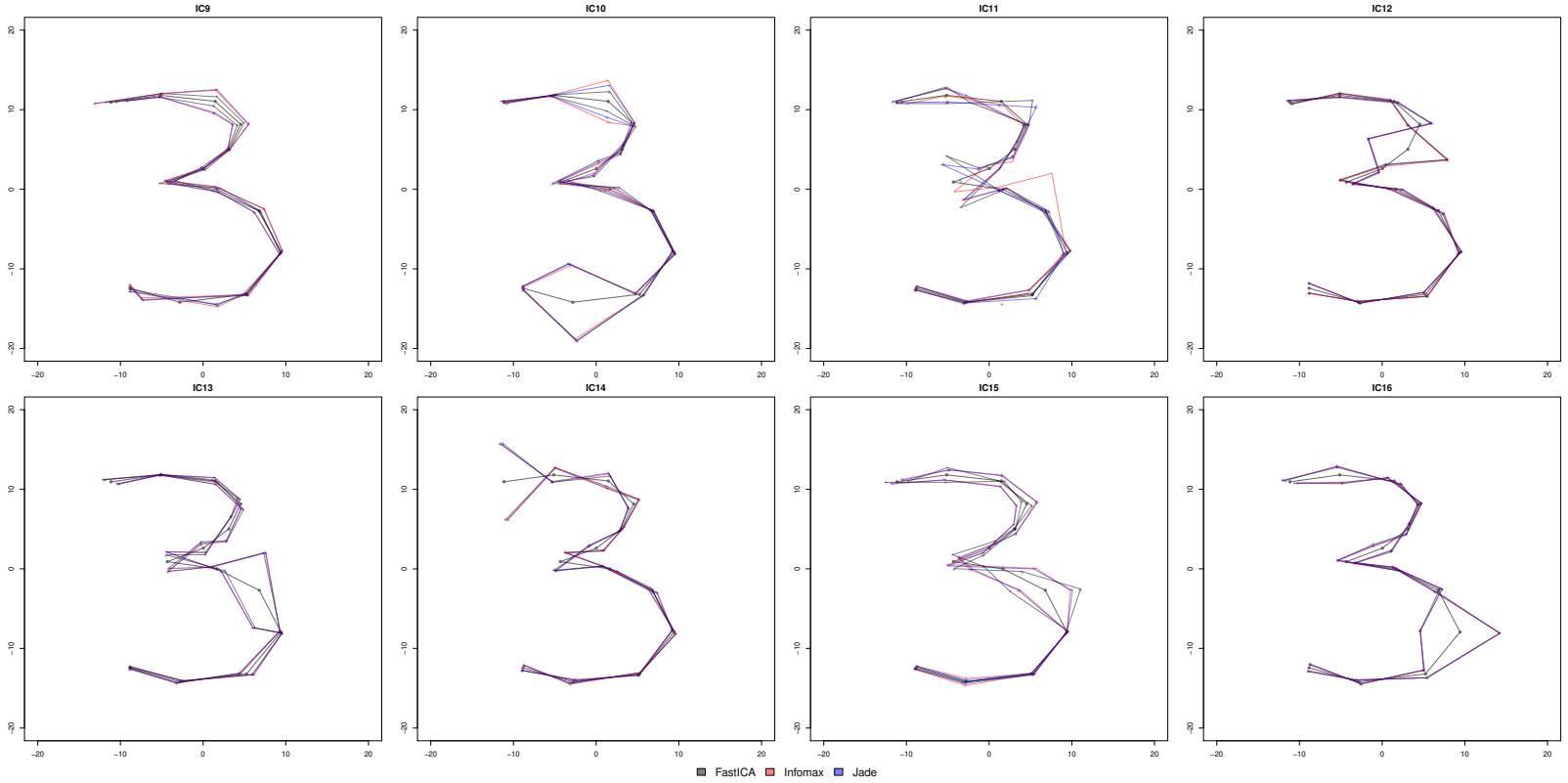
Variabilitätsstrukturen

In diesem Anhang sind die Abbildungen der Variabilitätsstruktur der generierten *Shapes* für

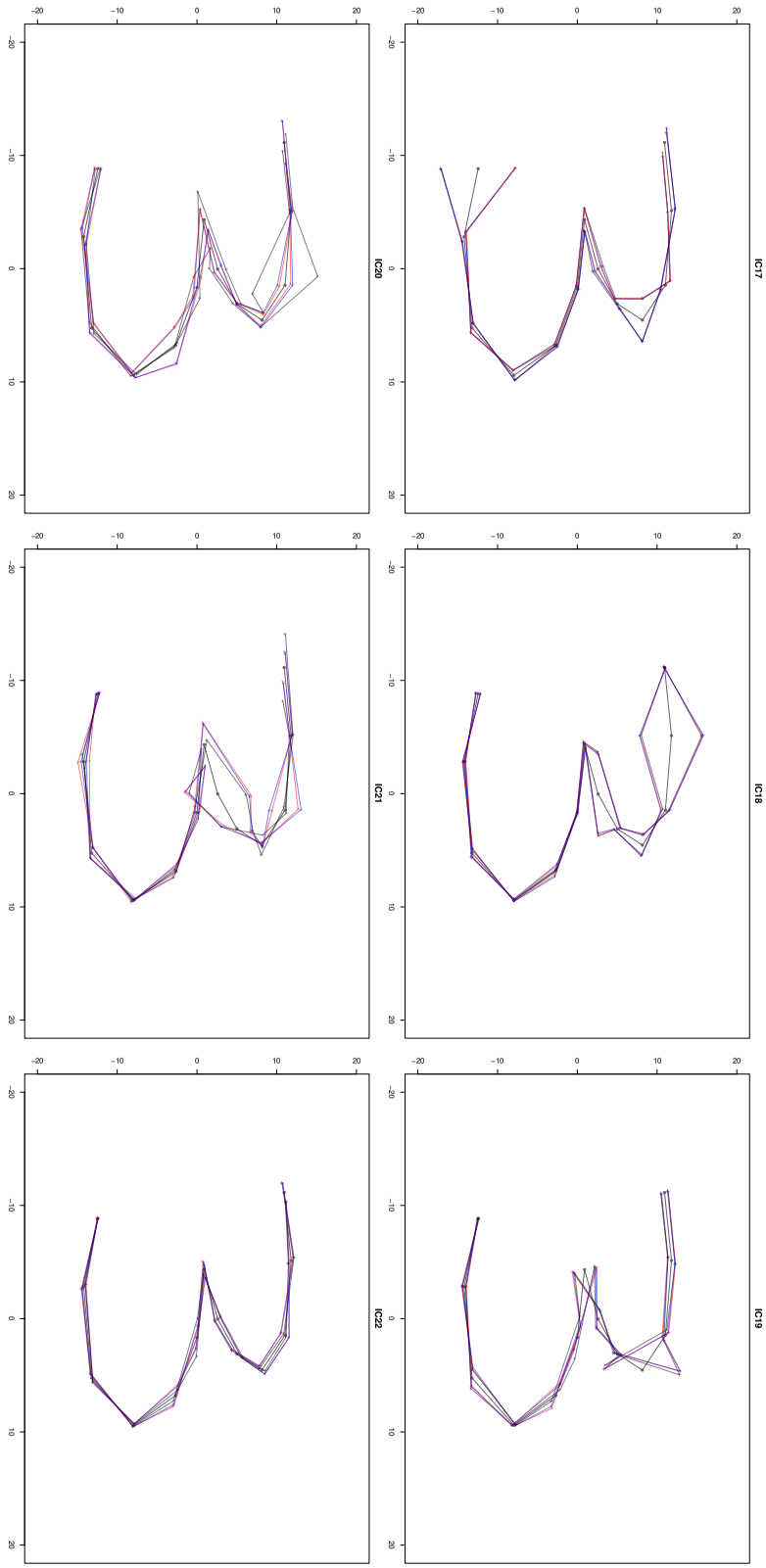
- Datensatz `digit3.dat` mit 22 unabhängigen Komponenten,
- Datensatz `mice` mit 8 unabhängigen Komponenten.



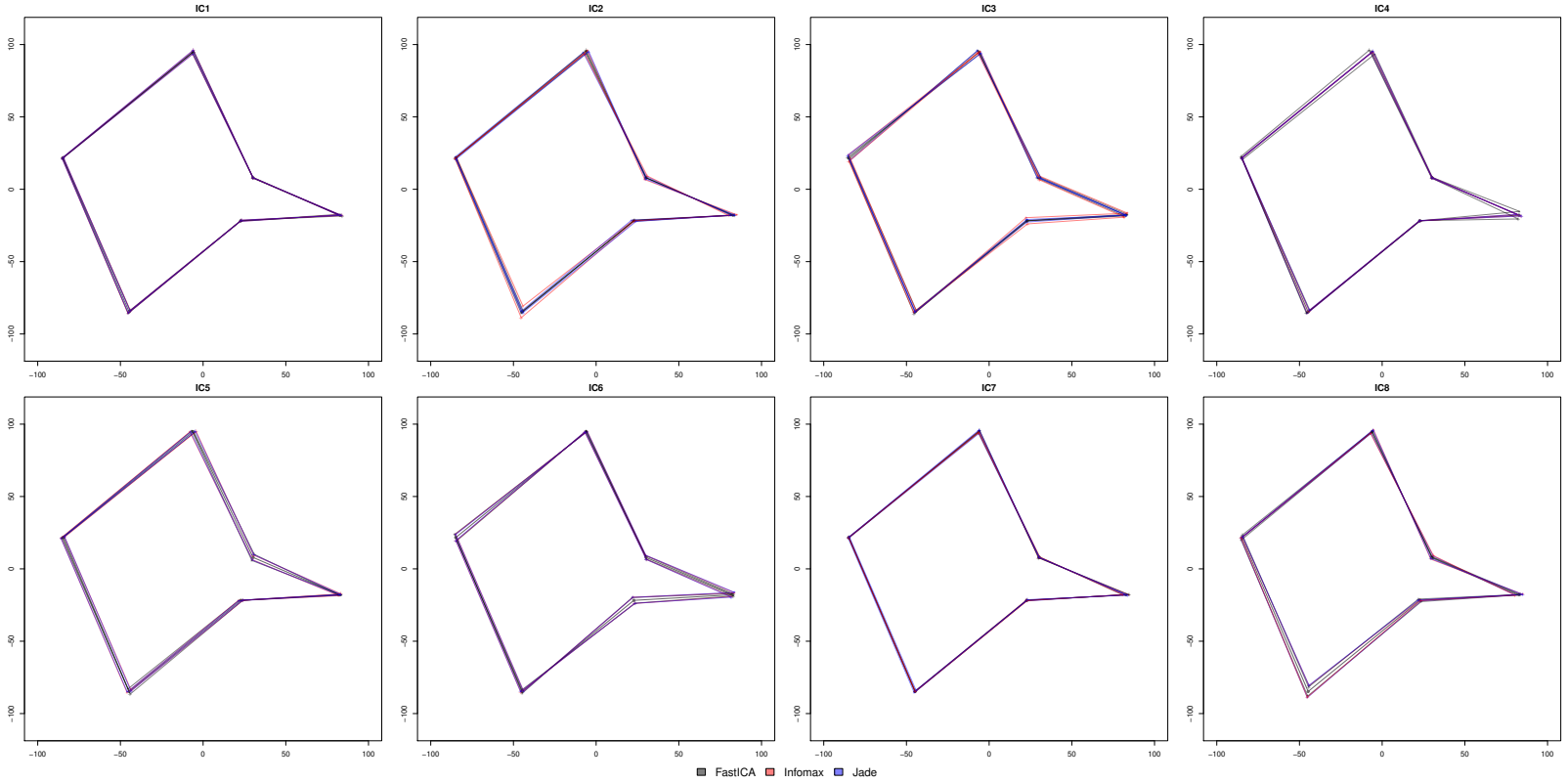
Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 22 ICs (IC 1-8) (Datensatz: `digit3.dat`))



Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 22 ICs (IC 9-16) (Datensatz: `digit3.dat`))



Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 22 ICs (IC 17-22) (Datensatz: `digit3.dat`))



Variabilitätsstruktur der generierten *Shapes* mit $c = \pm 1$ bei 8 ICs (Datensatz: `mice`))

Anhang C

Komponentenvektoren der Variationen

Im Rahmen dieses Anhangs sind die Komponentenvektore der Variationen aufgrund der teilweise entgegengesetzten Richtung der Komponentenvektoren zwischen den einzelnen Algorithmen.

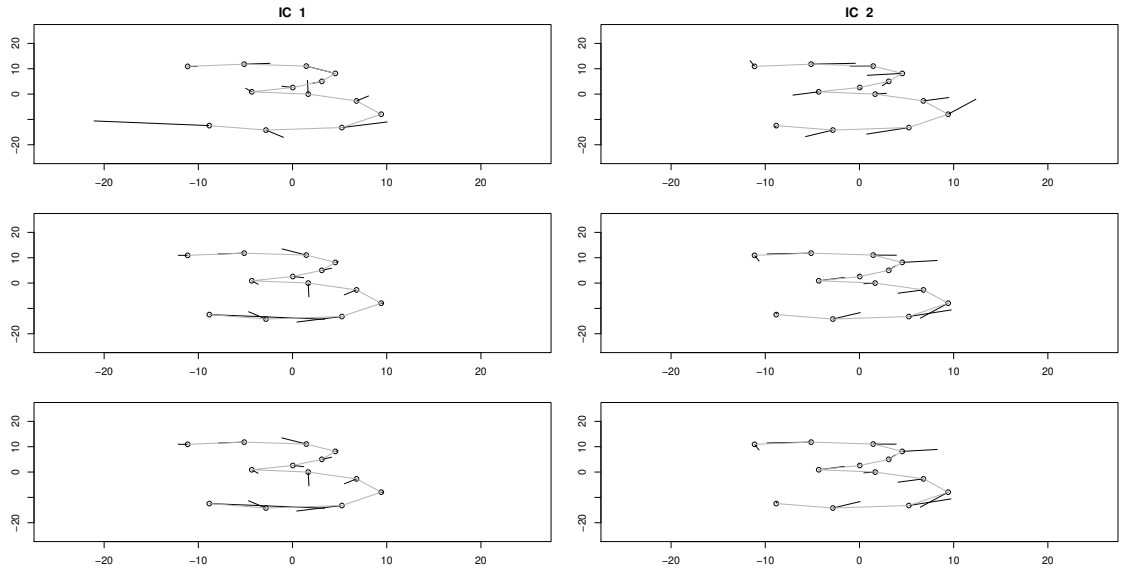


Abbildung C.1: Unabhängige Komponentenvektoren des `FastICA` *(obere Reihe)*, des `InfoMax` *(mittlere Reihe)* sowie des `JADE` Algorithmus bei 2 ICs (Datensatz: `digit3.dat`)

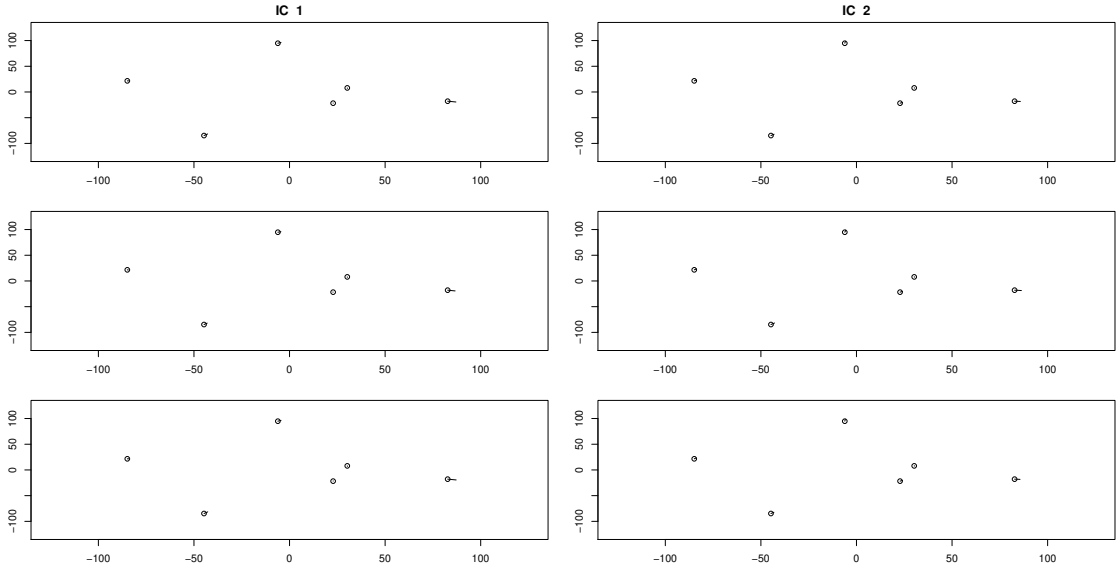


Abbildung C.2: Unabhängige Komponentenvektoren des `FastICA` *(obere Reihe)*, des `InfoMax` *(mittlere Reihe)* sowie des `JADE` Algorithmus bei 2 ICs (Datensatz: `mice`)

Literatur

1. Bell, A. J. & Sejnowski, T. J. An information-maximization approach to blind separation and blind deconvolution. *Neural computation* **7,** 1129–1159. ISSN: 0899-7667 (1995).

2. Bookstein, F. L. *Landmark methods for forms without landmarks: localizing group differences in outline shape* in *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis* (IEEE Computer Society Press, Los Alamitos, Calif., 1996), 279–289. ISBN: 0-8186-7368-0. doi:10.1109/MMBIA.1996.534080.

3. Cardoso, J. F. High-order contrasts for independent component analysis. *Neural computation* **11,** 157–192. ISSN: 0899-7667 (1999).

4. Cardoso, J. F. & Souloumiac, A. Blind beamforming for non-gaussian signals. *IEE Proceedings F Radar and Signal Processing* **140,** 362. ISSN: 0956375X (1993).

5. Comon, P. Independent component analysis, A new concept? *Signal Processing* **36,** 287–314. ISSN: 01651684 (1994).

6. Cootes, T. F., Taylor, C. J., Cooper, D. H. & Graham, J. in *BMVC92* (Hrsg. Hogg, D. & Boyle, R.) 9–18 (Springer London, London, 1992). ISBN: 978-3-540-19777-5. doi:10.1007/978-1-4471-3201-1{\textunderscore}2.

7. Cootes, T., Taylor, C., Cooper, D. & Graham, J. Active Shape Models-Their Training and Application. *Computer Vision and Image Understanding* **61,** 38–59. ISSN: 1077-3142 (1995).

8. Cover, T. M. & Thomas, J. A. *Elements of information theory* ISBN: 0-471-20061-1. <http://dx.doi.org/10.1002/0471200611> (Wiley-Interscience, Hoboken, NJ, 2001).

9. Dryden, I. L. `shapes` *package* Contributed package, Version 1.2.0. R Foundation for Statistical Computing (Vienna, Austria, 2017). <http://www.R-project.org>.

10. Dryden, I. L. & Mardia, K. V. *Statistical shape analysis: With applications in R* Second edition. ISBN: 978-0470699621 (2016).

11. Gower, J. C. Generalized procrustes analysis. *Psychometrika* **40,** 33–51. ISSN: 0033-3123 (1975).

12. He, Q., Duan, Y., Karsch, K. & Miles, J. Detecting corpus callosum abnormalities in autism based on anatomical landmarks. *Psychiatry research* **183,** 126–132. ISSN: 0165-1781 (2010).

13. Huber, P. J. Projection Pursuit. *The Annals of Statistics* **13,** 435–475. ISSN: 00905364 (1985).

14. Hyvarinen, A. Fast and robust fixed-point algorithms for independent component analysis. *IEEE transactions on neural networks* **10,** 626–634. ISSN: 1045-9227 (1999a).

15. Hyvarinen, A. & Oja, E. Independent component analysis: algorithms and applications. *Neural networks : the official journal of the International Neural Network Society* **13,** 411–430. ISSN: 0893-6080 (2000).

16. Hyvärinen, A. in *Advances in Neural Information Processing Systems 10* (Hrsg. Jordan, M. I., Kearns, M. J. & Solla, S. A.) 273–279 (MIT Press, 1998).

17. Hyvärinen, A., Karhunen, J. & Oja, E. *Independent Component Analysis* 1st ed. ISBN: 0-471-22131-7. doi:10.1002/0471221317. <http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10272411> (Wiley-Interscience, S.l., 2001).

18. Hyvärinen, A., Särelä, J. & Vigário, R. *Spikes and bumps: Artefacts generated by independent component analysis with insufficient sample size* in *Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA'99)* (1999b), 425–429.

19. Mardia, K. V. & Dryden, I. L. The Statistical Analysis of Shape Data. *Biometrika* **76,** 271. ISSN: 00063444 (1989).

20. Meyer-Baese, A. & Schmid, V. J. *Pattern Recognition and Signal Analysis in Medical Imaging* 2nd ed. ISBN: 9780124095458. <http://gbv.eblib.com/patron/FullRecord.aspx?p=1657929> (Elsevier Science, Burlington, 2014).

21. Papoulis, A. & Pillai, S. U. *Probability, random variables, and stochastic processes* 3rd ed. ISBN: 0-07-048477-5 (McGraw-Hill, Boston, 1991).

22. Suinesiaputra, A. *u. a.* in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2004* (Hrsg. Barillot, C., Haynor, D. R. & Hellier, P.) 737–744 (Springer, Berlin und Heidelberg, 2004). ISBN: 978-3-540-22976-6. doi:10.1007/978-3-540-30135-6{\textunderscore}90.

23. Ten Berge, J. M. F. Orthogonal procrustes rotation for two or more matrices. *Psychometrika* **42,** 267–276. ISSN: 0033-3123 (1977).

24. Zhang, M. & Golland, P. Statistical shape analysis: From landmarks to diffeomorphisms. *Medical Image Analysis* **33,** 155–158. ISSN: 1361-8415 (2016).