Analyse dünnbesetzter Hauptachsen für Frequenzdaten

Tobias Bork

Geboren am 21. November 1997 in Reutlingen

2. Februar 2020

Bachelorarbeit Mathematik

Betreuer: Prof. Dr. Jochen Garcke

Zweitgutachter: Prof. Dr. X Y

Mathematisches Institut für Numerische Simulation

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Danksagung

The acknowledgments and the people to thank go here, don't forget to include your project advisor…

Inhaltsverzeichnis

Danksagung **1**

1 Einführung **1**
1.1 Motivation . . . 1
1.2 Dimensionsreduktionsverfahren . . . 1
1.3 Sparse Approximations / Representations . . . 2
1.4 Interpretierbarkeit . . . 2
1.5 Compressed Sensing Beispiel . . . 2

2 Mathematische Grundlagen **3**
2.1 Lineare Algebra . . . 3
2.1.1 Orthogonalität . . . 3
2.1.2 Matrixzerlegungen . . . 5
2.1.3 Matrix Approximation . . . 6
2.2 Analysis . . . 7
2.2.1 Norm . . . 8
2.3 Generalisierte lineare Modelle . . . 8
2.3.1 Grundlagen aus der Statistik . . . 9
2.3.2 Lineare Regression . . . 10
2.3.3 Ridge Regression . . . 12
2.3.4 LASSO . . . 12
2.3.5 Elastic Net . . . 15
2.3.6 Vergleich der Regressionsmethoden . . . 16
2.4 Signaltheorie . . . 18
2.4.1 Fouriertransformation . . . 18
2.4.2 Nyquist-Shannon Abtasttheorem . . . 18

3 Hauptkomponentenanalyse **19**
3.1 Konstruktion . . . 20
3.1.1 Problemformulierung als Varianzmaximierung . . . 22
3.1.2 Formulierung als Singulärwertzerlegung . . . 22
3.1.3 Formulierung als beste Rang k Rekonstruktion . . . 23
3.1.4 Formulierung als Regressionsproblem . . . 23
3.2 Selektion der Hauptkomponenten . . . 24
3.3 Grenzen der Anwendbarkeit . . . 25
3.4 Erweiterungen der Hauptkomponentenanalyse . . . 26
3.5 Implementierung . . . 26
3.6 Theoretische Aussagen . . . 26

4 Dünnbesetzte Hauptkomponentenanalyse **29**
4.1 Problemformulierung . . . 29
4.2 Relaxation . . . 30
4.3 Konstruktion Sparse PCA . . . 32

4.4 Anpassung der Varianzen . . . 33
4.5 Theoretische Aussagen . . . 33

5 Implementierung **35**
5.1 Numerische Lösung . . . 35
5.2 Algorithmus . . . 36
5.3 Numerische Lösung im Fall p » n . . . 37
5.4 Implementierung in scikit-learn in python . . . 37
5.5 Laufzeitvergleich . . . 37

6 Anwendung **39**
6.1 Anwendung auf Simulationsdaten . . . 39
6.2 Der Datensatz . . . 39
6.3 Anwedung auf Frequenzdaten . . . 39
6.4 Auswertung der Ergebnisse . . . 39
6.5 Vergleich mit PCA Resultaten . . . 39
6.6 Hyperparameter . . . 39
6.6.1 Zeit . . . 39
6.6.2 Effekt auf Resultate . . . 39

7 Ausblick / Zusammenfassung **41**
7.1 Einsetzbarkeit . . . 41
7.2 Übertragbarkeit . . . 41
7.3 Ongoing Research / Weitere Techniken . . . 41

Literatur **43**

Kapitel 1

Einführung

We are drowning in information and starving for knowledge. - John Naisbitt

Machine Learning, unsupervised, supervised Methoden

Zahlen und Fakten zu Big Data

Ab wann sprechen wir eigentlich von Big Data?

Ït should be noted, however, that even when one has an apparently massive data set, the effective number of data points for certain cases of interest might be quite small. In fact, data across a variety of domains exhibits a property known as the long tail, shich means that a few things are very common, but most things are quite rare."(Murphy, Machine Learning)

1.1 Motivation

So ist man meist besonders an der Bildung sog. Cluster, also Gruppierungen, interessiert. Datenpunkte, die im entstehendem Bild nach Anwendung der Hauptkomponentenanalyse nah beieinander liegen, sind in gewisser Weise ähnlich zueinander während Datenpunkte, die weit von einander entfernt liegen, wenig Ähnlichkeit aufweisen. Abbildung CITE zeigt die Entstehung solcher Cluster auf dem Datensatz. Mit diesem Verfahren lässt sich daher eine Art Struktur in den Daten erkennen, die für weitere Analysezwecke ausgenutzt werden kann.

The goals of PCA are to

(1)

extract the most important information from the data table; (2)

compress the size of the data set by keeping only this important information; (3)

simplify the description of the data set; and (4)

analyze the structure of the observations and the variables.

1.2 Dimensionsreduktionsverfahren

CURSE OF DIMENSIONALITY (Bellman 1961)

High dimensionality means that the dataset has a large number of features. The primary problem associated with high-dimensionality in the machine learning field is model overfitting, which reduces the ability to generalize beyond the examples in

the training set. Richard Bellman described this phenomenon in 1961 as the Curse of Dimensionality where "Many algorithms that work fine in low dimensions become intractable when the input is high-dimensional. "

Let's say that you want to predict what the gross domestic product (GDP) of the United States will be for 2017. You have lots of information available: the U.S. GDP for the first quarter of 2017, the U.S. GDP for the entirety of 2016, 2015, and so on. You have any publicly-available economic indicator, like the unemployment rate, inflation rate, and so on. You have U.S. Census data from 2010 estimating how many Americans work in each industry and American Community Survey data updating those estimates in between each census. You know how many members of the House and Senate belong to each political party. You could gather stock price data, the number of IPOs occurring in a year, and how many CEOs seem to be mounting a bid for public office. Despite being an overwhelming number of variables to consider, this just scratches the surface. TL;DR — you have a lot of variables to consider. If you've worked with a lot of variables before, you know this can present problems. Do you understand the relationships between each variable? Do you have so many variables that you are in danger of overfitting your model to your data or that you might be violating assumptions of whichever modeling tactic you're using? You might ask the question, "How do I take all of the variables I've collected and focus on only a few of them?" In technical terms, you want to "reduce the dimension of your feature space." By reducing the dimension of your feature space, you have fewer relationships between variables to consider and you are less likely to overfit your model. (Note: This doesn't immediately mean that overfitting, etc. are no longer concerns — but we're moving in the right direction!) Somewhat unsurprisingly, reducing the dimension of the feature space is called "dimensionality reduction." There are many ways to achieve dimensionality reduction, but most of these techniques fall into one of two classes: Feature Elimination Feature Extraction

2. Why is Dimensionality Reduction required? Here are some of the benefits of applying dimensionality reduction to a dataset:

Space required to store the data is reduced as the number of dimensions comes down Less dimensions lead to less computation/training time Some algorithms do not perform well when we have a large dimensions. So reducing these dimensions needs to happen for the algorithm to be useful It takes care of multicollinearity by removing redundant features. For example, you have two variables – 'time spent on treadmill in minutes' and 'calories burnt'. These variables are highly correlated as the more time you spend running on a treadmill, the more calories you will burn. Hence, there is no point in storing both as just one of them does what you require It helps in visualizing data. As discussed earlier, it is very difficult to visualize data in higher dimensions so reducing our space to 2D or 3D may allow us to plot and observe patterns more clearly

1.3 Sparse Approximations / Representations

1.4 Interpretierbarkeit

In Zeiten der Datenvielfalt erfährt die Interpretation dieser enorme Wichtigkeit.

1.5 Compressed Sensing Beispiel

Kapitel 2

Mathematische Grundlagen

In diesem Kapitel werden wir die wichtigsten mathematischen Grundlagen, die wir für die Formulierung der dünnbesetzten Hauptkomponentenanalyse benötigen, einführen. Dazu beschäftigen wir uns zunächst mit Grundbegriffen aus des linearen Algebra, Matrixzerlegungen und ausgewählten Approximationsproblemen in Abschnitt 2.1. Ein Großteil dieses Kapitel ist linearen Regressionsmodellen gewidmet, welche wir mit verschiedenen Straftermen versehen werden, um gewisse Effekte zu erzielen. Anhand eines Beispiel-Datensatzes werden wir in der Lage sein, diese Effekte visuell nachzuvollziehen. Für den weiteren Verlauf dieser Arbeit ist das Verständnis der Strafterme von entscheidender Bedeutung. Zu Schluss werden wir in Abschnitt 2.4 Grundlagen der Signalverarbeitung ausarbeiten, da wir in Kapitel 6 die dünnbesetzte Hauptkomponentenanalyse auf Frequenzdaten anwenden.

2.1 Lineare Algebra

Ein Großteil der Mathematik der Hauptkomponentenanalyse beruht auf Methoden der linearen Algebra. Aufgrund des Anwendungsfalls werden wir uns auf die Einführung der Grundbegriffe in reellen Vektorräumen beschränken.

2.1.1 Orthogonalität

Definition 2.1 (Skalarprodukt [Jän07])**.** Sei V ein reeller Vektorraum. Ein *Skalarprodukt* in V ist eine Abbildung $\langle \cdot, \cdot \rangle : V \times V \longrightarrow \mathbb{R}$ mit den folgenden Eigenschaften:

(i) Für jedes $x \in V$ sind die Abbildungen

$$\begin{aligned} \langle \cdot, x \rangle : V &\longrightarrow \mathbb{R} \\ v &\longmapsto \langle v, x \rangle \end{aligned} \qquad \begin{aligned} \langle x, \cdot \rangle : V &\longrightarrow \mathbb{R} \\ v &\longmapsto \langle x, v \rangle \end{aligned}$$

linear. (Bilinearität)

(ii) $\langle x, y \rangle = \langle y, x \rangle$ für alle $x, y \in V$ (Symmetrie)

(iii) $\langle x, x \rangle > 0$ für alle $x \neq 0$ (Positive Definitheit)

Allgemein versteht man unter einem *euklidischem Vektorraum* ein Paar $(V, \langle \cdot, \cdot \rangle)$, welches aus einem reellem Vektorraum V und einem Skalarprodukt $\langle \cdot, \cdot \rangle$ auf V besteht. Durch das Skalarprodukt wird eine Norm auf V induziert:

$$\|v\| := \sqrt{\langle v, v \rangle}$$

In den folgenden Kapiteln werden wir uns vor allem mit dem *Standardskalarprodukt* im $\mathbb{R}^n$ beschäftigen. Dies ist gegeben durch

$$\langle x, y \rangle = x_1 y_1 + \cdots + x_n y_n.$$

Theorem 2.2 (Verallgemeinerter Satz des Pythagoras [Ant98]). *Für orthogonale Vektoren u, v in einem euklidischem Vektorraum V gilt*

$$\|u + v\|^2 = \|u\|^2 + \|v\|^2 .$$

Definition 2.3 (Orthogonalität [Jän07]). Zwei Elemente v, w eines euklidischen Vektorraumes V heißen *orthogonal* (geschrieben $v \perp w$) wenn ihr Skalarprodukt null ist, d.h.

$$v \perp w \iff \langle v, w \rangle = 0.$$

Eine Familie $(v_1, \ldots, v_n)$ in V heißt *orthogonal* oder *Orthogonalsystem*, wenn

$$v_i \perp v_j \quad \text{für alle} \quad i \neq j.$$

Gilt zusätzlich $\langle v_i, v_i \rangle = 1$ für alle $1 \leq i \leq n$, so spricht man von einem *Orthonormalsystem*.

Theorem 2.4 (Existenz einer Orthonormalbasis [Fis13]). *Jeder endlichdimensionale euklidische Vektorraum besitzt eine Orthonormalbasis.*

Der Begriff der Orthogonalität lässt sich auf Matrizen übertragen.

Definition 2.5 (Orthogonale Matrix [Ant98]). Eine Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ heißt *orthogonal*, falls deren Zeilen- und Spaltenvektoren paarweise orthonormal bezüglich des Standardskalarprodukts sind, d.h.

$$\mathbf{A}^\top \mathbf{A} = \mathbb{1}_n$$

Definition 2.6 (Orthogonalprojektion [Ant98]). Eine *Orthogonalprojektion* auf einen Untervektorraum U eines Vektorraumes V ist eine lineare Abbildung $P_U \colon V \to V$, die für alle Vektoren $v \in V$ die beiden Eigenschaften

(i) $P_U(v) \in U$ (Projektion)

(ii) $\langle P_U(v) - v, u \rangle = 0$ für alle $u \in U$ (Orthogonalität)

erfüllt.

Mithilfe einer Orthogonalbasis für U lässt sich aus dieser Definition eine Lösung für die Orthogonalprojektion $P_U(v)$ herleiten.

Theorem 2.7 ([Ant98]). *Ist $(u_1, \ldots, u_n)$ eine Orthogonalbasis von U, so gilt für alle $v \in V$*

$$P_U(v) = \sum_{i=1}^{n} \frac{\langle v, u_i \rangle}{\langle u_i, u_i \rangle} u_i$$

Später werden wir die Orthogonalprojektion in einer anderen Form nutzen. Wir können die Projektion auch als Matrix-Vektor-Produkt auffassen. Verwenden wir das

Standardskalarprodukt gilt mit einer Orthogonalbasis $(u_1, \ldots, u_n)$ von U:

$$P_U(v) = \sum_{i=1}^{n} \frac{v^\top u_i}{u_i^\top u_i} u_i = \sum_{i=1}^{n} \frac{u_i u_i^\top}{u_i^\top u_i} v = \mathbf{A}\mathbf{A}^\top v \tag{2.1}$$

wobei $\mathbf{A} = \begin{bmatrix} \frac{u_1}{\|u_1\|} & \cdots & \frac{u_n}{\|u_n\|} \end{bmatrix}$. Die Orthonormalitätsbedingung in Theorem 2.7 kann auch weggelassen werden. Ist $(u_1, \ldots, u_n)$ eine beliebige Basis von U, so gilt:

$$P_U(v) = \mathbf{A}(\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top v \tag{2.2}$$

Wir nennen $\mathbf{H} = \mathbf{A}(\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top$ die *orthogonale Projektionsmatrix*. Mithilfe von Theorem 2.2 lässt sich zeigen, dass der orthogonal auf den Unterraum projizierte Vektor den Abstand zwischen dem Ausgangsvektor und dem Unterraum minimiert.

Theorem 2.8 ([Ant98]). *Sei U ein Unterraum eines euklidischen Vektorraumes V. Dann ist $P_U(v)$ die beste Näherung von u in U, d.h.*

$$\|P_U(v) - v\|^2 \leq \|u - v\|^2 \quad \textit{für alle } u \in U$$

2.1.2 Matrixzerlegungen

In diesem Abschnitt führen wir zwei wichtige Matrixzerlegungen ein, die auch in vielen Bereichen der Numerik Anwendung finden.

Definition 2.9 (Eigenwert, Eigenvektor [Ant98]). Sei $\mathbf{A} \in \mathbb{R}^{n \times n}$. Ein von Null verschiedener Vektor $x \in \mathbb{R}^n$ heißt *Eigenvektor* von $\mathbf{A}$, falls

$$\mathbf{A}x = \lambda x$$

für einen Skalar $\lambda \in \mathbb{R}$. Die Zahl λ heißt *Eigenwert* von $\mathbf{A}$.

Definition 2.10 (Diagonalisierbar [Ant98]). Eine quadratische Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ heißt *diagonalisierbar*, wenn eine invertierbare Matrix $\mathbf{P}$ existiert, so dass $\mathbf{D} = \mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ Diagonalgestalt hat.

Es gibt verschiedene Kriterien für die Diagonalisierbarkeit von Matrizen. Für unsere spätere Anwendung interessieren wir uns vor allem für die Frage, ob es zu einer gegebenen Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ eine orthogonale Matrix $\mathbf{P}$ gibt, die $\mathbf{A}$ diagonalisiert. Eine derartige Diagonalisierung wird auch als *Hauptachsentransformation* bezeichnet. Dieser Name stammt ursprünglich aus der Theorie der Kegelschnitte. Hierbei ist eine Hauptachsentransformation eine orthogonale Abbildung, welche die Koordinatenachsen in die Richtungen der beiden *Hauptachsen* überführt. Wir wollen uns aber vorerst nicht mit dieser geometrischen Interpretation beschäftigen, sondern mit einem mathematisch äquivalenten, in den Anwendungen aber wichtigeren Problem.

Theorem 2.11 (Hauptachsentransformation [Jän07]). *Sei $\mathbf{A} \in \mathbb{R}^{n \times n}$ eine symmetrische Matrix. Dann gibt es eine orthogonale Transformation $\mathbf{P}$, welche $\mathbf{A}$ in eine Diagonalmatrix*

$\mathbf{D} := \mathbf{P}^{-1}\mathbf{AP}$ *der Gestalt*

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & & & & & & \\ & \ddots & & & & & \\ & & \lambda_1 & & & & \\ & & & \ddots & & & \\ & & & & \lambda_r & & \\ & & & & & \ddots & \\ & & & & & & \lambda_r \end{bmatrix}$$

überführt. Hierbei sind $\lambda_1, \ldots, \lambda_r$ *die verschiedenen Eigenwerte von* $\mathbf{A}$.

Zusammenfassend besitzt eine symmetrische Matrix also eine Zerlegung $\mathbf{A} = \mathbf{PDP}^\top$. Man kann $\mathbf{P}$ konstruieren, so dass die Spalten genau den Eigenvektoren von $\mathbf{A}$ entsprechen. Wir werden diese Umformung in späteren Kapiteln unter dem Begriff *Eigenwertzerlegung* (Englisch: *Eigenvalue Decomposition*) verwenden.

Eine eng verwandte, aber vielseitigere Faktorisierung von Matrizen ist die *Singulärwertzerlegung*. Sie ermöglicht eine allgemeine Zerlegung von nicht quadratischen oder nicht symmetrischen Matrizen.

Theorem 2.12 (Singulärwertzerlegung [SW04])**.** *Jede Matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *besitzt eine Singulärwertzerlegung*

$$\mathbf{A} = \mathbf{UDV}^\top$$

mit orthogonalen Matrizen $\mathbf{U} \in \mathbb{R}^{m\times m}$ *und* $\mathbf{V} \in \mathbb{R}^{n\times n}$*, sowie der Diagonalmatrix* $\mathbf{D} = (\sigma_j \delta_{ij}) \in \mathbb{R}^{m\times n}$.

Definition 2.13 (Singulärwert)**.** Die positiven Diagonaleinträge $\sigma_i > 0$ von $\mathbf{D}$ werden *Singulärwerte* genannt.

Singulärwerte einer Matrix $\mathbf{A}$ sind eindeutig bestimmt und stehen durch $\sigma_i = \sqrt{\lambda_i}$ in einer engen Beziehung mit den Eigenwerten λ_i. Konventionell werden die Singulärwerte von $\mathbf{D}$ absteigend sortiert, d.h. $\sigma_1 \geq \cdots \geq \sigma_r$. Geometrisch bedeutet diese Zerlegung, dass sich die Matrix $\mathbf{A}$ in zwei Drehungen $\mathbf{U}, \mathbf{V}$ und eine Streckung unterteilen lässt. Dabei korrespondieren die Streckungsfaktoren mit den Einträgen der Diagonalmatrix $\mathbf{D}$.

2.1.3 Matrix Approximation

In diesem Abschnitt werden wir zwei Approximationsprobleme für Matrizen formulieren, die eine explizite Lösung besitzen. Zunächst führen wir dafür eine Matrixnorm ein, von welcher wir auch später sehr häufig Gebrauch machen werden.

Definition 2.14 (Frobeniusnorm [SW04])**.** Für eine Matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ ist die *Frobeniusnorm* definiert durch

$$\|\mathbf{A}\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{\frac{1}{2}}.$$

Man zeigt leicht, dass $\|\mathbf{A}\|_F^2 = \operatorname{tr}\left(\mathbf{A}^\top \mathbf{A}\right)$ gilt. Eine weitere wichtige Eigenschaft von Matrizen ist der *Rang*.

Definition 2.15 (Rang [Ant98])**.** Die Dimension des Zeilen- und des Spaltenraumes einer Matrix $\mathbf{A}$ heißt *Rang* von $\mathbf{A}$ und wird mit $\operatorname{rank}(\mathbf{A})$ bezeichnet.

Wir möchten nun eine Matrix $\mathbf{A}$ durch eine andere, simplere Matrix $\widehat{\mathbf{A}}$ mit niedrigerem Rang approximieren. Dieses Problem fällt unter die Kategorie *low rank approximation*, welche eine enge Verbindung zur Hauptkomponentenanalyse aufweist. In Anwendung korrespondiert die Rang-Bedingung mit der Komplexität eines Modells. Mithilfe der Singulärwertzerlegung können wir eine explizite Lösung angeben.

Theorem 2.16 (Eckart-Young-Mirsky-Theorem [EY36]). *Sei* $\mathbf{A} \in \mathbb{R}^{m \times n}$ *mit* $m \leq n$ *und*

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$$

eine Singulärwertzerlegung von $\mathbf{A}$. *Wir partitionieren* $\mathbf{U}, \mathbf{D}$ *und* $\mathbf{V}$ *wie folgt:*

$$\mathbf{U} =: \begin{bmatrix}\mathbf{U}_1 & \mathbf{U}_2\end{bmatrix}, \quad \mathbf{D} =: \begin{bmatrix}\mathbf{D}_1 & 0 \\ 0 & \mathbf{D}_2\end{bmatrix}, \quad \mathbf{V} =: \begin{bmatrix}\mathbf{V}_1 & \mathbf{V}_2\end{bmatrix},$$

wobei $\mathbf{U}_1 \in \mathbb{R}^{m \times r}$, $\mathbf{D}_1 \in \mathbb{R}^{r \times r}$ *und* $\mathbf{V}_1 \in \mathbb{R}^{n \times r}$. *Dann löst die abgeschnittene Singulärwertzerlegung (Englisch: truncated singular value decomposition)*

$$\widehat{\mathbf{A}}^* = \mathbf{U}_1\mathbf{D}_1\mathbf{V}_1^\top,$$

das Approximationsproblem

$$\min_{\operatorname{rank}(\widehat{\mathbf{A}}) \leq r} \|\mathbf{A} - \widehat{\mathbf{A}}\|_F = \|\mathbf{A} - \widehat{\mathbf{A}}^*\|_F = \sqrt{\sigma_{r+1}^2 + \cdots + \sigma_m^2}, \tag{2.3}$$

wobei σ_i *die Singulärwerte von* $\mathbf{A}$ *sind. Der Minimierer* $\widehat{\mathbf{A}}^*$ *ist genau dann eindeutig, wenn* $\sigma_{r+1} \neq \sigma_r$.

Das Eckart-Young-Mirsky-Theorem wird es uns in Abschnitt 3.6 ermöglichen, eine wertvolle Eigenschaft der Hauptkomponentenanalyse zu zeigen.

Ein anderes Approximationsproblem für Matrizen ist das *orthogonale Procrustes Rotationsproblem*. Hierbei sind uns zwei Matrizen $\mathbf{M}$ und $\mathbf{N}$ gegeben, welche durch eine orthogonale Transformation ineinander überführt werden sollen. Wieder hilft uns die Singulärwertzerlegung bei der Findung einer Lösung.

Theorem 2.17 (Procrustes Rotationsproblem [GD+04]). *Seien* $\mathbf{M} \in \mathbb{R}^{n \times p}$, $\mathbf{N} \in \mathbb{R}^{n \times k}$ *und* $\mathbf{M}^\top\mathbf{N} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ *eine Singulärwertzerlegung. Dann löst*

$$\widehat{\mathbf{A}} = \mathbf{U}\mathbf{V}^\top$$

das Approximationsproblem

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \left\|\mathbf{M} - \mathbf{N}\mathbf{A}^\top\right\|_F^2 \tag{2.4}$$

unter der Nebenbedingung, dass $\mathbf{A}^\top\mathbf{A} = \mathbf{I}_{k \times k}$.

In Abschnitt 5.1 wird sich (2.4) als Subproblem der dünnbesetzten Hauptkomponentenanalyse herausstellen.

2.2 Analysis

In diesem Abschnitt möchten wir die

2.2.1 Norm

Definition 2.18 ([Hie19]). Eine Abbildung $\|\cdot\| : \mathbb{R}^n \longrightarrow \mathbb{R}_0^+$ heißt *Norm*, falls für alle Vektoren $x, y \in \mathbb{R}^n$ und alle Skalare $\alpha \in \mathbb{R}$ die folgenden drei Axiome gelten:

(i) $\|x\| = 0 \Leftrightarrow x = 0$ (Definitheit)

(ii) $\|\alpha \cdot x\| = |\alpha| \cdot \|x\|$ (Homogenität)

(iii) $\|x + y\| \leq \|x\| + \|y\|$ (Subadditivität)

Definition 2.19 (ℓ_p-Norm [SW04]). Auf dem $\mathbb{R}^n$ sind die l_p-Normen für $1 \leq p < \infty$ definiert als

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} \quad x \in \mathbb{R}^n$$

und für $p = \infty$ als

$$\|x\|_\infty := \max_{1 \leq i \leq n} |x_i| \quad x \in \mathbb{R}^n.$$

Im Fall $p = \infty$ spricht man auch von der *Maximumsnorm* und im Fall $p = 2$ von der *euklidischen Norm*.

Um Verwirrung auszuschließen werden wir im Folgenden von der ℓ_q-Norm sprechen, da wir mit p die Anzahl an Variablen in einem Modell bezeichnen. Eine weitere wichtige Norm, die wir im Zuge dieser Arbeit verwenden werden ist die ℓ_0-"Norm". Diese zählt die von null verschiedenen Einträge eines Vektors und misst somit, ob ein Vektor dünnbesetzt ist.

Definition 2.20 (ℓ_0-"Norm" [FR13]). Die sogenannte ℓ_0-"Norm" ist definiert durch

$$\|x\|_0 := |\{i : \ x_i \neq 0, \quad 1 \leq i \leq n\}| .$$

Die übliche Schreibweise $\|x\|_0$ - die Notation $\|x\|_0^0$ wäre angemessener - entspringt der Beobachtung, dass

$$\|x\|_q^q = \sum_{i=1}^{n} |x_i|^q \quad \underset{q \to 0}{\longrightarrow} \quad \sum_{i=1}^{n} \mathbb{1}_{\{x_i = 0\}} = |\{i : \ x_i \neq 0 , \quad 1 \leq i \leq n\}|$$

Wir werden diese Schreibweise analog für Matrizen anstatt Vektoren verwenden. Dabei wollen wir betonen, dass die ℓ_0-"Norm" keine wirkliche Norm ist, da die Abbildung nicht homogen ist. Trotzdem ist diese "Norm" in der Theorie der komprimierten Erfassung (Englisch: *compressive sensing*) sehr nützlich. Des Weiteren werden wir die Notation $\|\cdot\|_q$ gemäß der Definition in 2.19 auch für Werte $0 < q < 1$ verwenden, obwohl durch diese Abbildung ebenfalls keine Norm gegeben ist.

2.3 Generalisierte lineare Modelle

Es existiert eine sehr enge Verbindung zwischen der Hauptkomponentenanalyse, die wir in Kapitel 3 näher kennenlernen werden, und der Regressionsanalyse. Viele der Ideen und Ansätze im folgendem Abschnitt werden wir später gebrauchen und spielen eine maßgebliche Rolle bei der Formulierung der dünnbesetzten Hauptkomponentenanalyse.

2.3.1 Grundlagen aus der Statistik

Seien $(x_1, \ldots, x_n)$ Datenpunkte. Es bezeichne $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ das Stichprobenmittel. Stichprobenvarianz

$$s_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x}_n)^2$$

Empirische Kovarianzmatrix!

Sei $g\colon \Theta \to \mathbb{R}$ eine zu schätzende reelle Parameterfunktion in einem statistischem Modell $(X, \mathcal{A}, \mathcal{P})$ wobei $\mathcal{P} = \{P_\vartheta\colon \vartheta \in \Theta\}$. Wir werden nun einige wichtige Grundbegriffe für einen Schätzer $d\colon (X, \mathcal{A}) \to (\mathbb{R}, \mathcal{B}(\mathbb{R}))$ in $\mathcal{L}^1(\mathcal{P})$ einführen.

Definition 2.21 (Verzerrung [Rüs14])**.** Die *Verzerrung* (Englisch: *Bias*) des Schätzers d bei ϑ ist definiert durch

$$\mathrm{Bias}_\vartheta(d) := \mathrm{E}_\vartheta(d) - g(\vartheta).$$

Um verschiedene Schätzer miteinander zu vergleichen bedienen wir uns häufig des *mittleren quadratischen Fehlers*. Dieser gibt an, welche Abweichung zwischen dem Schätzer und dem wahrem Parameter zu erwarten ist. Damit bietet sich uns eine Möglichkeit den erwarteten Fehler eines Lernalgorithmus analysieren.

Definition 2.22 (Mittlerer quadratischer Fehler [Koh05] (Def 15.7))**.** Der *mittlere quadratische Fehler* (Englisch: *Mean Squared Error (MSE))* ist definiert durch

$$\mathrm{MSE}(d, \vartheta) := \mathrm{E}_\vartheta\left(\left(d - g(\vartheta)\right)^2\right).$$

Theorem 2.23 (Verschiebungssatz [Koh05] ())**.** *Der mittlere quadratische Fehler zerfällt in Varianz und Bias, d.h.*

$$\mathrm{MSE}(d, \vartheta) = \mathrm{Var}_\vartheta(d) + \left(\mathrm{Bias}_\vartheta(d)\right)^2$$

Für die Bewertung eines Schätzers ist also sowohl Verzerrung als auch Varianz zu berücksichtigen. Leider ist in der Praxis selten möglich, beide Fehlerquellen zeitgleich zu minimieren. Im Bereich des überwachten maschinellen Lernens ist das Problem unter dem *Verzerrung-Varianz-Dilemma* (Englisch: *bias-variance tradeoff*) bekannt. Idealerweise versucht man ein Modell zu wählen, welches sowohl die Gesetzmäßigkeiten in den Trainingsdaten genau erfasst, als sich auch auf ungesehene Testdaten generalisieren lässt. Aufgrund von falschen Annahmen kann es bei einem Lernalgorithmus zu einer hohen Verzerrung kommen. Beziehungen zwischen Eingabe und Ausgabe können nicht geeignet modelliert werden und es kommt zu einem Fehler zwischen System und Modell. Man spricht in diesem Fall von einer *Unteranpassung* (Englisch: *underfitting*). Demgegenüber sind Modelle mit hoher Varianz meist komplexer und ermöglichen eine präzise Darstellung der Trainingsdaten. Dadurch läuft man aber Gefahr, sich dem Rauschen der Daten anzupassen und nicht die Gesetzmäßigkeiten der Trainingsdaten zu erkennen. Wir bezeichnen dieses Phänomen als Überanpassung (Englisch: *overfitting*), was in ungenauen Vorhersagen auf Testdaten münden kann.

Die Frage nach einem günstigen Modell liegt also in der Modellkomplexität und es gilt eine Balance zwischen den beiden beschriebenen Extrema zu finden. Diese Idee haben wir in 2.1 veranschaulicht.

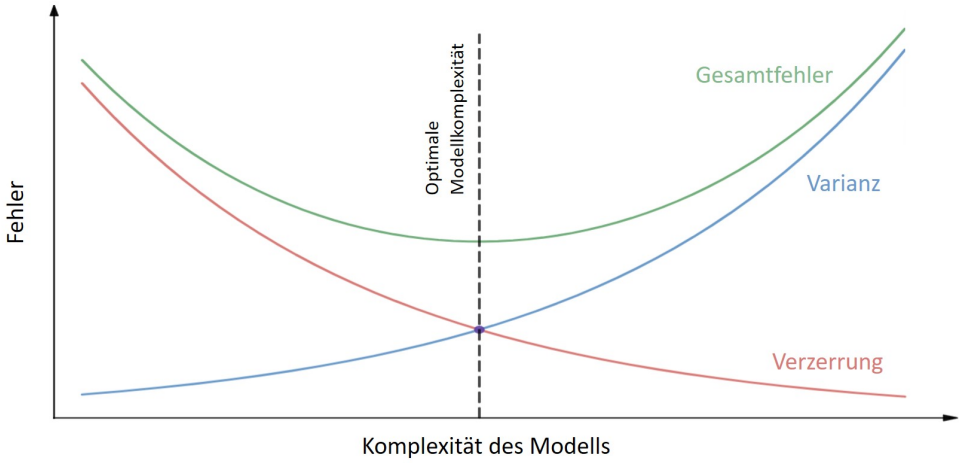


ABBILDUNG 2.1: Verzerrung-Varianz-Dilemma

2.3.2 Lineare Regression

[HTF09]

Bei der Regressionsanalyse werden Zusammenhänge zwischen mehreren Merkmalen untersucht. Man versucht eine unabhängige Variable Y durch eine oder mehrere abhängige Variablen $X_1, \ldots, X_p$ zu erklären. Ein lineares Regressionsmodell hat also die Form

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j \tag{2.5}$$

wobei β_j die Regressionskoeffizienten sind. Bei der Verwendung dieses Modells nehmen wir an, dass die Regressionsfunktion $\mathrm{E}(Y|X)$ linear ist bzw. ein lineares Modell eine geeignete Approximation ist.

Typischerweise verfügen wir über eine Menge von Trainingsdaten $(x_1, y_1), \ldots, (x_n, y_n)$. Jedes $x_i = (x_{i1}, \ldots, x_{ip})^\top$ stellt eine Beobachtung dar, die wir für die Schätzung der Parameter β_j benutzen. Die bekannteste Methode für diesen Zweck ist sicherlich die *Methode der kleinsten Quadraten* (Englisch: *(Ordinary) Least Squares*), in welcher wir $\beta = (\beta_0, \ldots, \beta_p)^\top$ wählen, so dass die Summe der Residuenquadrate (Englisch: *Residual Sum of Squares*, kurz RSS) minimiert wird. Wir definieren

$$\begin{aligned}
\mathrm{RSS}(\beta) &= \sum_{i=1}^{n} (y_i - f(x_i))^2 && (2.6)\\
&= \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 && (2.7)\\
&= (y - \mathbf{X}\beta)^\top (y - \mathbf{X}\beta) && (2.8)\\
&= \|y - \mathbf{X}\beta\|_2^2 && (2.9)
\end{aligned}$$

und das dazugehörige Minimierungsproblem

$$\hat{\beta}^{\mathrm{OLS}} = \arg\min_{\beta} \mathrm{RSS}(\beta) \tag{2.10}$$

wobei $\mathbf{X} \in \mathbb{R}^{n\times(p+1)}$ die Matrix der x_i mit einer 1 an erster Stelle ist und $y = (y_1, \ldots, y_n)^\top$. Eine Visualisierung der Methode befindet sich in Abbildung REF. Vielleicht Bild zur Regression in 2 bzw. 3D?

An dieser Stelle möchten wir erwähnen, dass bei Verwendung dieser Methode keine Aussage über die Gültigkeit des Modells getroffen, sondern lediglich die beste lineare Approximation gefunden wird.

Falls $\mathbf{X}$ vollen Rang hat, ist es ist leicht zu zeigen, dass (2.10) die eindeutige Lösung

$$\hat{\beta}^{\text{OLS}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top y \tag{2.11}$$

besitzt. Die Zielgröße ergibt sich dann durch

$$\hat{y} = \mathbf{X}\hat{\beta}^{\text{OLS}} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top y \tag{2.12}$$

Die Matrix $\mathbf{H} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$ haben wir bereits in REF kennengelernt. Sie projiziert y orthogonal auf den durch die Spalten von $\mathbf{X}$ aufgespannten Unterraum. Dies ermöglicht eine geometrische Interpretation der linearen Regression.

Wenn $\mathbf{X}$ keinen vollen Rang hat, ist die Lösung von (2.10) nicht mehr eindeutig. Dieser Art Probleme ereignen sich häufig in der Bild- und Signalanalyse, da es mehr Variablen als Beobachtungen gibt ($p > n$). Um ein gewünschtes Verhalten der Regression zu gewährleisten, bestehen verschiedene Möglichkeiten der Filterung oder Regularisierung. In letzterem Fall versehen wir den Regressionsterm mit sog. *Straftermen*, welche eine bedeutende Rolle in den folgenden Kapitel spielen werden. Wir werden uns auf die Einführung dieser in linearer Regression beschränken. Das Konzept kann aber allgemeiner auch in logistischer Regression oder anderen verallgemeinerten linearen Modellen verwendet werden. Mit leicht abgewandelten Argumentationen erhalten wir dann bei Einbettung der Strafterme in die jeweilige Zielfunktion dieselben Effekte. Mehr dazu in CITE.

Gelegentlich sind wir mit den Ergebnissen der Methode der kleinsten Quadrate nicht zufrieden. Dies kann zum Beispiel an der Multikollinearität des Datensatzes liegen. Mit zunehmender Multikollinearität ist die Methode instabil und bei der Schätzung der Regressionskoeffizienten ungenau. Es ergeben sich die folgenden zwei Probleme:

- Vorhersagegenauigkeit: Meist hat das erzeugte Modell einen niedrigen Bias, aber eine hohe Varianz. Somit sind neue Vorhersagen, die wir auf zuvor ungesehenen Daten treffen, oft sehr ungenau. In manchen Fällen können wir die Ergebnisse durch eine Erhöhung des Bias verbessern, indem wir die Regressionskoeffizienten verkleinern oder sogar auf 0 setzen.
- Interpretation: Eine hohe Anzahl an Variablen, die in das Modell einfließen erschwert unzweifelhaft eine Interpretation. Daher kann es von Vorteil sein nur den Teil der Variablen für das Modell auszuwählen, die den größten Effekt für die Vorhersage erzielen.

Ein naheliegender Ansatz zur Lösung dieser Probleme wäre es zu versuchen, die beste Teilmenge an Variablen zu finden, die eine minimale Summe der Residuenquadrate aufweist. In [HTF09] werden verschiedene Methoden zur exakten und approximativen Berechnung dieser Teilmenge beschrieben. (Darunter fallen die leaps and bounds Methode (Furnival and Wilson, 1974), Forward Stepwise und Backward Stepwise.) Nicht immer wird die Genauigkeit der Vorhersagen durch Verwendung

dieses Ansatzes besser. Dies liegt daran, dass es sich um einen diskreten Prozess handelt und somit Variablen für das Modell entweder ausgewählt oder verworfen werden. Daher beschäftigen wir uns nun mit Methoden, die eine kontinuierliche Schrumpfung der Regressionskoeffizieten erlauben.

2.3.3 Ridge Regression

Die *Tikhonov Regularisierung*, die auch unter dem Namen *Ridge Regression* bekannt ist, kann genutzt werden, um das Problem der Multikollinearität zu lösen und im Fall $p > n$ eine eindeutige Lösung zu erhalten. Wie wir bereits in REF gesehen haben, kann es sinnvoll sein den Bias zu erhöhen, um genauere Vorhersagen treffen zu können. Daher führen wir nun eine *ridge penalty*, einen Strafterm, ein und formulieren das Ridge Regression Problem in der Lagrange Form.

$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{2.13}$$

wobei $\lambda \geq 0$ ein Parameter ist, der die Stärke der Schrumpfung der Regressionskoeffizienten kontrolliert. Je größer λ, desto stärker ist die Schrumpfung der β_j. Man kann zeigen, dass die Lösung von (2.13) in zwei Teile aufgeteilt werden kann. Da β_0 nicht im Strafterm vorkommt, schätzen wir $\beta_0 = \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ und zentrieren die Eingaben $x_{ij} = x_{ij} - \bar{x}_j$. Die eindeutige Lösung der zentrierten Version von (2.13) ist dann gegeben durch

$$\hat{\beta}^{\text{ridge}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top y \tag{2.14}$$

wobei $\beta = (\beta_1, \ldots, \beta_p)$ und $\mathbf{X} \in \mathbb{R}^{n \times p}$ die Matrix der x_i. (Vor der Anwendung von Ridge Regression werden die Daten meist standardisiert, da $\hat{\beta}^{\text{ridge}}$ nicht äquivariant unter Skalierung ist.)

2.3.4 LASSO

Die durch die Ridge Regression erzeugten Koeffizienten sind also um den Faktor $\frac{1}{1+\lambda}$ gegenüber denen von Least Squares skaliert. Die Regressionskoeffizieten werden also erst für $\lambda \to 0$ auf Null geschrumpft. Um eine bessere Interpretation des Modells zu ermöglichen kreiert das *Lasso* eine dünnbesetzte Lösung, bei welcher viele Koeffizienten gleich Null sind. Das Lasso wurde erstmals von Tibshirani in [Tib96] eingeführt und ist in der Signalanalyse unter dem Namen *Basis Pursuit* [CDS98] bekannt. Mathematisch erreichen wir eine Dünnbesetzung durch Einbettung eines ℓ_1-Strafterms.

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{2.15}$$

Es wird also im Vergleich zu (2.13) lediglich die ℓ_2-Norm durch eine ℓ_1-Norm ausgetauscht. Bevor wir uns mit der Lösung dieses Problems beschäftigen, möchten wir erklären, warum die ℓ_1-Norm eine Dünnbesetzung hervorruft. Die lässt sich auf zwei Arten erklären. Zunächst geben wir eine geometrische Erklärung, welche in Abbildung 2.2 zu sehen ist. Dort sind die ℓ_1- und ℓ_2-Beschränkungen sowie die Höhenlinien der RSS Funktion in zwei Dimensionen aufgezeichnet. Die optimalen

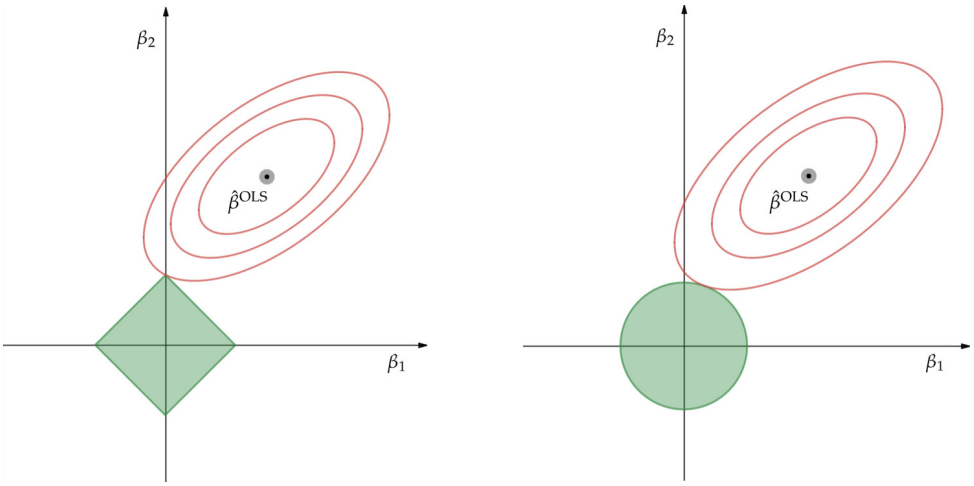


ABBILDUNG 2.2: Die Abbildung zeigt die Beschränkungen der ℓ_1-Norm (links) und der ℓ_2-Norm (rechts) zusammen mit den Höhenlinien der RSS-Funktion, welche $\hat{\beta}^{\text{OLS}}$ als Minimierer besitzt. Verdeutlicht wird hier die geometrische Findung von $\hat{\beta}^{\text{lasso}}$ (links) und $\hat{\beta}^{\text{ridge}}$ (rechts)
Basiert auf [HTF09]

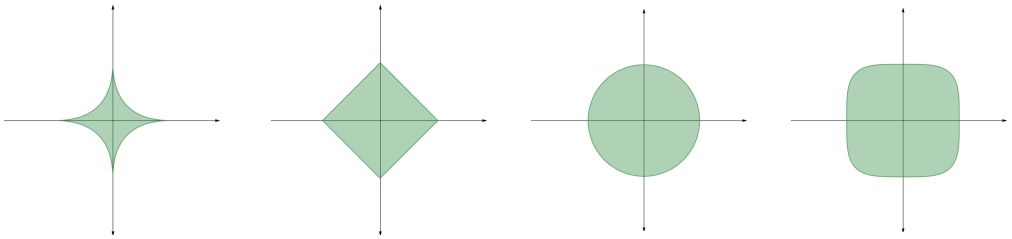
Koeffizienten von Ridge Regression und Lasso ergeben sich nun aus dem Schnittpunkt der Höhenlinie und der Norm-Begrenzung. Im Falle der ℓ_1-Norm ist dieser Schnittpunkt mit einer hohen Wahrscheinlichkeit an eine der Ecken und einer der beiden Koeffizienten wird auf Null gesetzt. Im Gegensatz gibt es bei der Wahl einer ℓ_2-Begrenzung keine Ecken. Somit kommt jeder Randpunkt der Begrenzung als Schnittpunkt in Frage und es wird keine Dünnbesetzung, sondern lediglich eine kontinuierliche Schrumpfung der Koeffizienten hervorgerufen. Dieser Effekt verstärkt sich in höheren Dimensionen.

An dieser Stelle kann man auf den Gedanken kommen, andere Strafterme zu verwenden, welche bei geometrischer Betrachtung die Wahrscheinlichkeit erhöhen eine Dünnbesetzung der Koeffizienten hervorzurufen. So kann man zum Beispiel die ℓ_q-Normen als Strafterm für Werte $q < 1$ in Betracht ziehen.

$$\hat{\beta}^{\text{sparse}} = \arg\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q \tag{2.16}$$

In Abbildung 2.3 sind die Begrenzungen der ℓ_q-Normen für verschiedene Werte eingezeichnet. NOTATION (p Anzahl Variablen) Für $q \to 0$ entstehen sternförmige Höhenlinien, welche immer weiter zum Ursprung gedrückt werden. Somit wird es immer wahrscheinlicher, dass die Höhenlinien der RSS-Funktion eine Ecke treffen. Das Problem liegt allerdings nicht im Effekt der verschiedenen Strafterme, sondern in der Berechnung. Für $q < 1$ ist (2.16) ein nicht-konvexes Optimierungsproblem, da $\|\cdot\|_q$ dann keine Norm gemäß Definition 2.18 ist. Im Extremfall der ℓ_0-"Norm" wird (2.16) sogar NP-schwer. CITE. Somit besteht in beiden Fällen keine effiziente Methode zur Berechnung von $\hat{\beta}^{\text{sparse}}$ zur Verfügung. Der Wert $q = 1$ ist eine Art Kompromisslösung, die einerseits effizient zu berechnen ist und andererseits noch immer eine dünnbesetzte Lösung liefert.

Um eine mathematisch gründliche Erklärung für die Dünnbesetzung zu liefern, wenden wir uns der Lösung von (2.15) zu. Diese kann nur dann explizit angegeben



$q = 0.5$ $q = 1$ $q = 2$ $q = 4$

ABBILDUNG 2.3: Die Abbildung zeigt die Begrenzungen der ℓ_q-Norm im $\mathbb{R}^2$ für verschiedene Werte von q, also die Mengen $\{x \in \mathbb{R}^2 \colon \|x\|_q \leq c\}$.

werden, wenn $\mathbf{X}$ orthonormale Spalten hat. Es hilft uns aber trotzdem diese Lösung zu verstehen??

$$\hat{\beta}_j^{\text{lasso}} = \text{sign}(\hat{\beta}_j^{\text{OLS}}) \left(\left| \hat{\beta}_j^{\text{OLS}} \right| - \frac{\lambda}{2} \right)_+ \tag{2.17}$$

wobei $(\cdot)_+ = max(\cdot, 0)$ ist. Der Beweis kann in [Mur12] nachgelesen werden. Die Lösung ist also durch den der sog. *soft thresholding operator* gegeben, welcher durch

$$\text{soft}_\delta(x) = \text{sign}(x)(|x| - \delta)_+ \tag{2.18}$$

definiert wird.

Der Operator ist in Abbildung 2.4 noch einmal graphisch dargestellt. Nun sind wir auch in der Lage zu verstehen, warum Tibshirani [Tib96] den Begriff Lasso eingeführt hat. Dieser steht für *Least absolute selection and shrinkage operator*, was bedeutet, dass ein Teil der Koeffizienten $\hat{\beta}_j^{\text{OLS}}$ ausgewählt und im Betrag geschrumpft wird. Die neuen Lasso Koeffizienten ergeben sich also dadurch, dass zunächst alle Koeffizienten in $\hat{\beta}^{\text{OLS}}$, die kleiner als $\frac{\lambda}{2}$ sind, auf 0 gesetzt werden und anschließend die Verbliebenden im Betrag um $\frac{\lambda}{2}$ geschrumpft werden. Somit ist klar, dass für $\lambda \geq \lambda_{\max}$ alle Koeffizienten $\hat{\beta}_j^{\text{lasso}} = 0$ sind, wobei $\lambda_{\max} = \left\| \mathbf{X}^\top y \right\|_\infty$. (Der Wert $\lambda_{\max}$ beruht auf der Beobachtung, dass 0 der optimale Wert der Koeffizienten ist falls $(\mathbf{X}^\top y)_j \in [-\lambda, \lambda]$ für alle j.)

Wir wenden uns nun einer mathematischen Lösung des Lasso zu. Im Gegensatz zu Ridge Regression kann man im Allgemeinen keine explizite Lösung angeben. Nur für den Fall, dass. Da $\|\beta\|_1$ nicht differenzierbar ist wenn $\beta_j = 0$ ist, sind wir bei (2.15) mit einem nicht glattem Optimierungsproblem konfrontiert. Seit der Problemformulierung in 1996 wurde eine Vielzahl an Algorithmen entwickelt bzw. adaptiert, die eine numerische Lösung liefern. Dazu gehören Least-angle Regression (LARS) [Tib+04], Koordinaten-Abstiegsverfahren [FHT10], Subdifferential Methoden und Näherungs-Gradientenverfahren [Yan+13; Van19]. Letztere sind eine natürliche Erweiterung von Gradientenverfahren wenn die Zielfunktion nicht differenzierbar ist. Wir werden später auf das Koordinaten-Abstiegsverfahren näher eingehen, welches wir bei der Implementierung der dünnbesetzten Hauptkomponentenanalyse nutzen.

Es stellt sich heraus, dass das Lasso zwei wesentliche Nachteile besitzt. Falls es im Datensatz Gruppen stark korrelierter Variablen gibt, so tendiert die Methode dazu

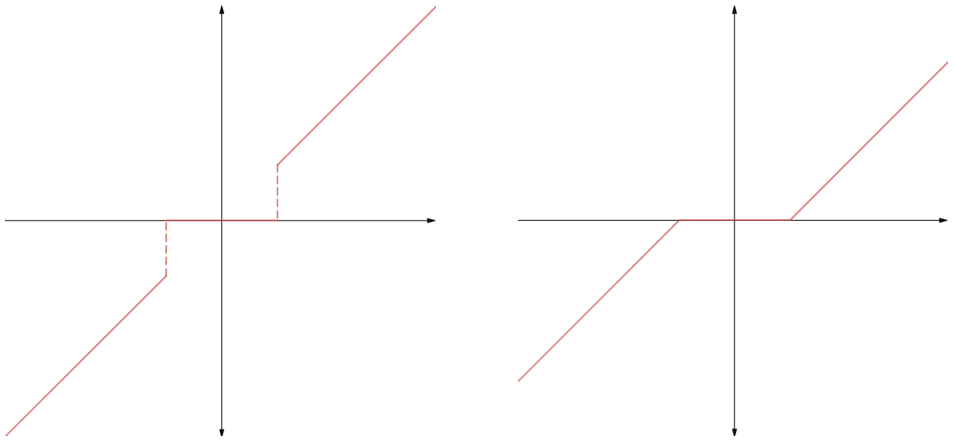


ABBILDUNG 2.4: Die Abbildung zeigt die beiden Operationen soft und hard thresholding

nur eine Variable aus einer Gruppe statt die Gruppe als Ganzes auszuwählen. In vielen Anwendungen ist man aber gerade daran interessiert. Zum Beispiel bei der Suche nach Genen, welche mit einer bestimmten Krankheit verbunden sind, möchte man alle assoziierten Koeffizienten finden anstatt nur einem Gen aus einer Gruppe. Darüber hinaus führt dies oft dazu, dass der Vorhersagefehler vergrößert wird und somit die Methode nicht ganz so robust ist. Um diesem Problem zu entgegnen kann man das sog. *Group Lasso* verwenden [YL06], bei welcher man zuvor Gruppen im Datensatz festlegen kann. Der im Zuge dieser Arbeit aber wichtigere Aspekt ist, dass das Lasso im Fall $p > n$ maximal n Variablen selektieren kann. Dies ist für moderne Datensätze, für welche $p \gg n$ gilt oft nicht ausreichend. Der Grund dafür wird in ... klar.

2.3.5 Elastic Net

Damit im Fall $p > n$ mehr als n Variablen selektiert werden, kann man die beiden vorgestellten Methoden kombinieren. Durch die Einbettung einer ℓ_1 und ℓ_2-Norm erhalten wir das sog. *Elastic Net* [ZH05].

$$\hat{\beta}^{\text{en}} = \arg\min_{\beta} \|y - \mathbf{X}\beta\|_2^2 + \lambda_2 \|\beta\|_2^2 + \lambda_{1,j} \|\beta\|_1 \tag{2.19}$$

Wie zuvor wird durch den ℓ_1-Strafterm ein dünnbesetztes Modell generiert. Der ℓ_2-Strafterm fördert den Gruppeneffekt, stabilisiert den ℓ_1 Regularisierungspfad und lässt eine beliebige Anzahl zu selektierender Variablen zu.

Ähnlich wie bei der Lasso Regression kann nur im Fall orthogonaler Spalten von $\mathbf{X}$ eine explizite Lösung mithilfe des soft thresholding operators von (2.19) angegeben werden. Mit leichten Modifikationen der Algorithmen für das Lasso erhalten wir eine numerische Lösung. So wurde LARS-EN [ZH05] oder ein Koordinatenabstiegsverfahren vorgeschlagen [FHT10].

We propose an efficient algorithm called LARS-EN to solve the elastic net efficiently, which is based on the recently proposed algorithmLARSof Efron et al. (2004). They proved that, starting from zero, the lasso solution paths grow piecewise linearly in a predictable way. They proposed a new algorithm called LARS to solve the entire

	AGE	SEX	BMI	BP	...	Serum Measurements				...	Response
Patient	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	59	2	32.1	101	157	93.2	38	4	4.9	87	151
2	48	1	21.6	87	183	103.2	70	3	3.9	69	75
3	72	2	30.5	93	156	93.6	41	4	4.7	85	141
4	24	1	25.3	84	198	131.4	40	5	4.9	89	206
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
441	36	1	30.0	95	201	125.2	42	5	5.1	85	220
442	36	1	19.6	71	250	133.2	97	3	4.6	92	57

TABELLE 2.1: Diabetes Datensatz [Tib+04; Tib+]

lasso solution path efficiently by using the same order of computations as a single OLS fit

2.3.6 Vergleich der Regressionsmethoden

Zur Veranschaulichung der oben eingeführten Methoden werden wir diese auf ein Beispiel anwenden. Dabei greifen wir auf einen durch scikit-learn [Ped+11] bereitgestellten Datensatz, der erstmals durch [Tib+04] öffentlich gemacht worden ist, zurück. In diesem wurden für $n = 442$ Diabetes Patienten $p = 10$ verschiedene Variablen gemessen. Dazu gehören Alter (AGE), Geschlecht (SEX), Body Mass Index (BMI), Blutdruck (BP) und verschiedene Blutproben (Serum Measurements). Die Zielgröße y enthält Werte für den Krankheitsfortschritt ein Jahr nach Behandlungsbeginn. Ein Ausschnitt des Datensatzes befindet sich in Tabelle 2.1.

Schrumpfung bei Ridge Regression kontinuierlich. Die Ergebnisse der Regressionen können in Abbildung 2.5 und **??** eingesehen werden.

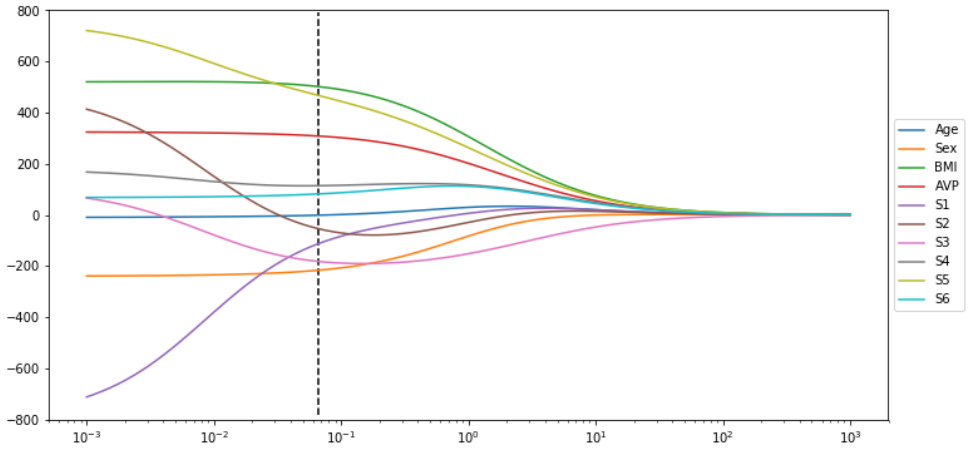
Die Implementierung des Elastic Nets in scikit-learn [Ped+11] beruht auf einer anderen, aber sehr ähnlichen mathematischen Formulierung

$$\hat{\beta}^{\text{en}} = \arg\min_{\beta} \frac{1}{2n} \|y - \mathbf{X}\beta\|_2^2 + \alpha\gamma \|\beta\|_2^2 + \frac{1}{2}\alpha(1-\gamma) \|\beta\|_1 \tag{2.20}$$

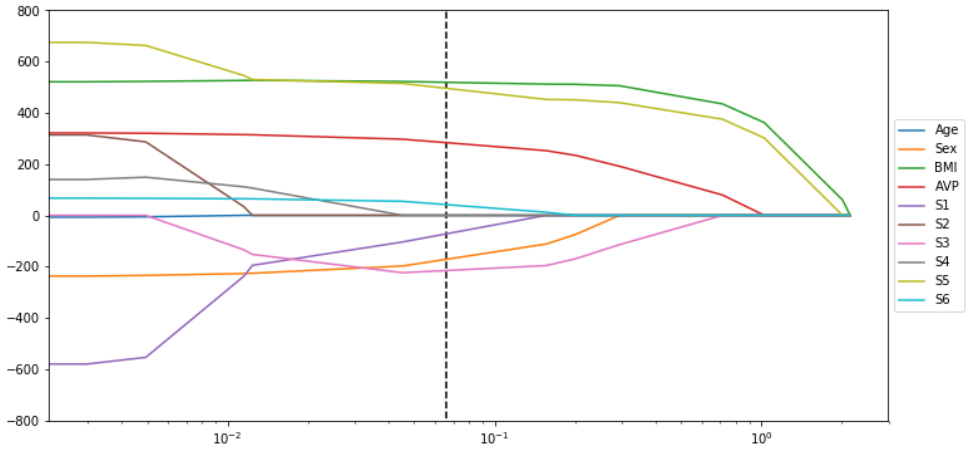
wobei n die Anzahl an Beobachtungen im Datensatz und γ das Verhältnis der ℓ_2- zur ℓ_1-Norm ist. Wählen wir $\gamma = 0$ so reduziert sich (2.20) auf Ridge Regression und für $\gamma = 1$ auf das Lasso. So können wir mit γ das Verhältnis der beiden Regressionen kontrollieren und mit α die Stärke der Bestrafung. Man beachte, dass bei dieser Implementierung nicht die Möglichkeit besteht die Koeffizienten unterschiedlich zu bestrafen. Setzt man

$$\alpha = \frac{2\lambda_2 + \lambda_1}{2n} \quad \text{und} \quad \gamma = \frac{\lambda_1}{2\lambda_2 + \lambda_1} \tag{2.21}$$

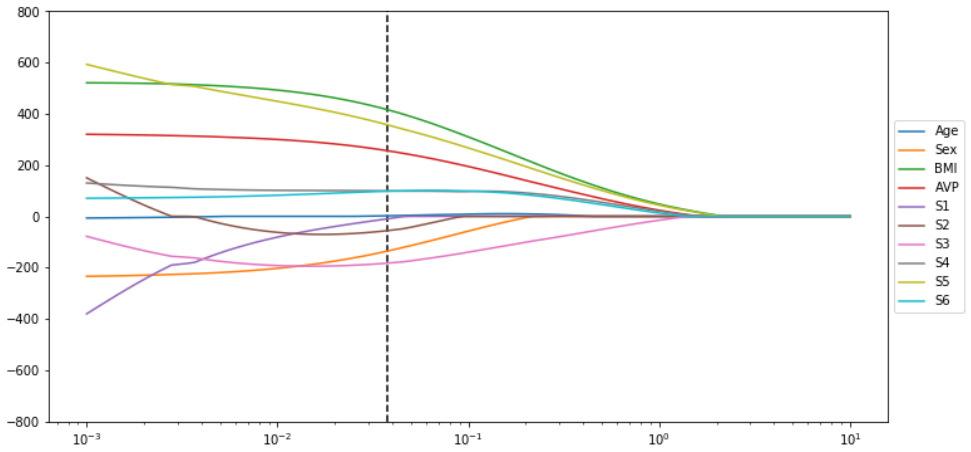
so reduziert sich (2.20) auf unser ursprünglich formuliertes Problem.



(A) Ridge Regression



(B) Lasso Regression



(C) Elastic Net, $\gamma = 0.98$

ABBILDUNG 2.5: Schrumpfung der Koeffizienten für verschiedene Regressionsmethoden bei Erhöhung des jeweiligen Regularisierungsparameters. Die vertikalen Linien stellen den Wert des jeweiligen Parameters dar, der durch ein 10-faches Kreuzvalidierungsverfahren bestimmt worden ist. (Erstellt mit scikit-learn)

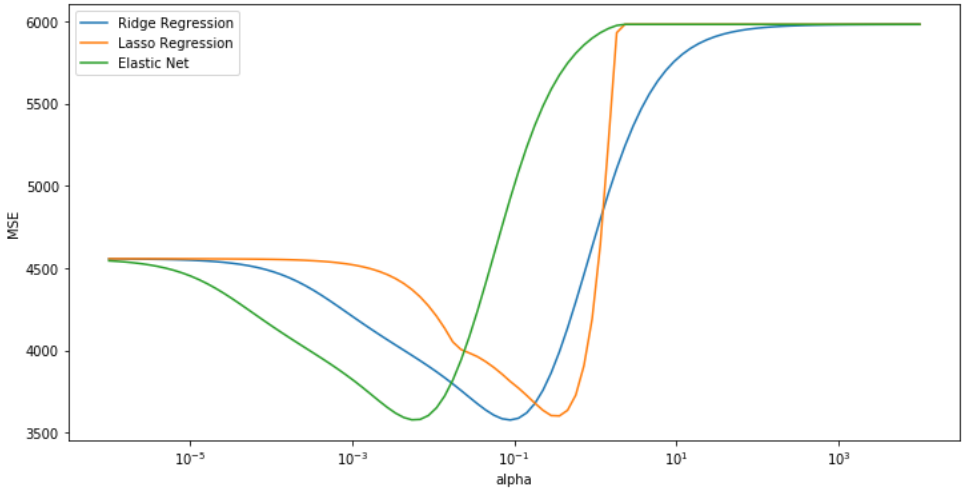


ABBILDUNG 2.6: Selektion des Modells gemäß der mittleren quadratischen Abweichung für Ridge, Lasso und Elastic Net Regression

2.4 Signaltheorie

2.4.1 Fouriertransformation

2.4.2 Nyquist-Shannon Abtasttheorem

Kapitel 3

Hauptkomponentenanalyse

To Do: Kovarianzmatrix / Stichprobenkovarianzmatrix einheitlich! Begriffe wie samples, PCA, oder features erklären, EIGENVALUE = VARIANCE

Die Hauptkomponentenanalyse ist ein weitverbreitetes multivariates statistisches Verfahren zur Dimensionsreduktion. Multivariate Verfahren zielen darauf ab, die in einem Datensatz enthaltene Zahl der Variablen zu verringern, ohne die darin enthaltene Information (zu verlieren) / (wesentlich zu reduzieren). Dadurch können umfangreiche Datensätze strukturiert, veranschaulicht und vereinfacht werden. Somit ist das Verfahren Teil der explorativen Statistik, welche Datensätze hinsichtlich ihrer Zusammenhänge analysiert. Die sich ergebende Struktur kann für weitere Analysezwecke ausgenutzt werden.

Aus diesem Grund hat die Hauptkomponentenanalyse in vielen Bereichen erfolgreich Anwendung gefunden. Darunter fällt die Erkennung handgeschriebener Zahlen, welche zum Beispiel zur automatischen Sortierung von Briefen nach Postleitzahl genutzt wird [HTF09]. An diesem Beispiel lässt es sich besonders gut verdeutlichen, was es heißt Zusammenhänge zu analysieren und Strukturen auf den Daten zu finden. Man erhofft, dass nach Anwendung einer Dimensionsreduktion wie PCA auf den Datensatz 10 verschiedene Gruppierungen zu erkennen sind, die für die Ziffern 0 bis 9 stehen (siehe dazu Bild?). Idealerweise gehören alle Datenpunkte im demselben Cluster zur selben Ziffer. Außerdem korrespondieren nahe beieinanderliegende Cluster mit Ziffern, die ähnlich aussehen. Weitere Anwendung findet das Verfahren in der Bildverarbeitung. Hier kann es zum Beispiel zur Rauschunterdrückung [BSP12] oder zur Gesichtserkennung [Tai+17] genutzt werden. Um Bilder für solch ein Verfahren nutzbar zu machen, werden einzelne Pixel oder patches, also lokale Gruppierungen von Pixeln, eines Bildes als Variable interpretiert.

Das mathematische Problem der Hauptkomponentenanalyse kann auf verschiedene Weisen beschrieben werden. Zunächst wollen wir es so konstruieren, dass die Idee des minimalen Informationsverlust im Vordergrund steht. Anschließend werden wir das Problem auf eine Singulärwertzerlegung zurückführen, die auch zur effizienten Implementierung genutzt wird. Des Weiteren werden wir die Hauptkomponentenanalyse als Regressionsproblem betrachten und die geometrische Interpretation weiter verdeutlichen. Zu Schluss werden wir einige theoretische Aussagen angeben, die für die folgenden Kapitel relevant sind.

3.1 Konstruktion

Gegeben sei ein Datensatz mit n samples und p Variablen. Die zentrale Idee der Hauptkomponentenanalyse besteht darin, die p bestehenden Variablen in k neue, unkorrelierte Variablen zu überführen. Um eine Reduktion der Dimension, also $k < p$ zu erreichen, müssen die bestehenden Variablen *zusammengefasst* werden. Idealerweise sollte bei diesem Prozess möglichst wenig Information verloren gehen. Als Maß für den Informationsgehalt der Daten wird hierbei die Varianz verwendet. Das heißt, je größer die Varianz einer Variable, desto mehr Information birgt sie und desto *wichtiger* ist sie. Denn eine Variable, die für alle Beobachtungen ähnliche Werte aufweist, ist nicht von Nutzen bei der Unterscheidung verschiedener samples. Um die Dimension zu reduzieren könnte man einfach nach den Eigenschaften größter Varianz suchen und alle Variablen unterhalb eines festgelegten Grenzwertes verwerfen. Dieses Vorgehen fällt allgemein unter die Methodik der *feature selection*. Die Hauptkomponentenanalyse verwendet allerdings ein anderes Prinzip, welches der Methodik der *feature extraction* zuzuordnen ist: Anstatt Eigenschaften mit hoher Varianz auszuwählen, konstruiert man neue Variablen, die die Bestehenden zusammenfassen. Variablen mit hoher Varianz werden in der Konstruktion einen größeren Beitrag spielen.

Konkret suchen wir also sukzessive nach einer Linearkombination der bestehenden Variablen. Finden wir nun zunächst die Richtung größter Varianz in unserem Datensatz, die erste *Hauptachse*. Der zugehörige Vektor spiegelt dabei den Beitrag bzw. den Informationsgehalt jeder einzelnen Variable wider. Anschließend finden wir weitere Hauptachsen, indem wir unter den Richtungen, die orthogonal zu allen vorherigen Hauptachsen sind, die mit der größten Varianz wählen. (Man iteriert diesen Prozess solange ...) Die Orthogonalität garantiert, dass die entstehenden Variablen unkorreliert sind. (Was hat das für einen Vorteil?) Nach der Identifizierung der Hauptachsen wollen wir unsere Beobachtungen bezüglich dieser darstellen. Wir erhalten die *Hauptkomponenten* unseres Datensatzes, indem wir die einzelnen samples auf die Hauptachsen transformieren. Aufgrund der schrittweisen Konstruktion verfügen Diese über eine sehr wichtige Sortierung. So beinhaltet die erste Hauptkomponente die meiste Information. Mit jeder weiteren Hauptkomponente erhält man mehr Information, aber der Informationsgewinn wird mit jeder Hauptkomponente geringer. Abbildung SCREE PLOT verdeutlicht diesen Verlauf.

Die eigentliche Dimensionsreduktion findet dann durch Selektion statt. Je nach Komplexität des Modells, welches man erreichen möchte, können so mehr oder weniger Hauptkomponenten ausgewählt werden. Je mehr Hauptkomponenten man auswählt, desto mehr Information erhält man über den Datensatz. Allerdings wird das Modell mit steigender Anzahl an Variablen immer komplizierter. Es gilt einen Punkt der Balance zu finden, der ein gutes Mittel aus Information und Komplexität liefert. Dieser kann vom Anwendungsfall abhängen. Wir werden uns mit diesem Thema weiter in 3.2 beschäftigen. Zusammenfassend haben wir somit unseren ursprünglichen Datensatz in neuen Hauptkomponenten konzentriert, die aber trotzdem einen Großteil an Information beinhalten.

Um dieses Prinzip zu veranschaulichen, wenden wir uns nun einem simplem Beispiel zu. Gegeben seien die Größe [cm] und das Gewicht [kg] zu 1000 Personen (Daten sind simuliert, keine real-world-data) (siehe dazu Abbildung). In diesem Fall ist also $n = 1000$ und $p = 2$. Bei Betrachtung der Abbildung fällt schnell auf, dass

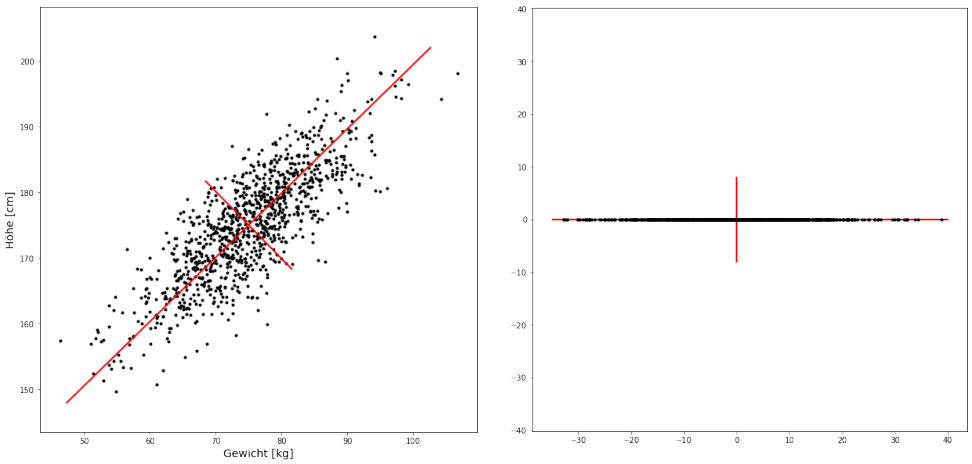


ABBILDUNG 3.1: Die Abbildung zeigt die Richtung größter Varianz
...

die beiden Variablen positiv korreliert sind, d.h. prinzipiell erkennt man folgende Tendenz: Je größer eine Person, desto schwerer ist sie.

Standardisierung

Bevor wir die Hauptkomponentenanalyse auf den Datensatz anwenden, gibt es aber noch einen wichtigen Bearbeitungsschritt zu beachten. Wenn eine Variable weniger variiert als eine Andere aufgrund der verwendeten Einheit oder Skala (Meter oder Kilo) kann dies zu ungewollten Ergebnissen führen. Ohne eine Vorbehandlung der Daten hat so im obigen Beispiel eine Änderung von 1m die gleiche Bedeutung wie eine Änderung von 1kg. (Satz schöner formulieren) Allerdings sind zwei Menschen, deren Größe 1m variiert, sehr verschieden, während zwei Menschen, die eine Differenz von 1kg haben, sehr ähnlich sind. Daher werden die Daten häufig einem sogenanntem preprocessing unterzogen. Ein zu diesem Zweck oft verwendetes Verfahren ist die Standardisierung oder auch z-Transformation genannt. In diesem Schritt werden die Variablen so transformiert, dass sie *vergleichbarer* werden. Seien dazu Y_i die Zufallsvariablen mit Erwartungswert $E[Y_i] = \mu$ und Varianz $Var[Y_i] = \sigma^2$. So erhält man die zugehörigen standardisierten Zufallsvariablen X_i durch Zentrierung und anschließender Division durch die Standardabweichung $X_i = \frac{Y_i - \mu}{\sigma}$. Somit gilt dann:

- $E[X_i] = 0$ für alle $1 \leq i \leq p$
- $Var[X_i] = 1$ für alle $1 \leq i \leq p$

Mathematisch gesehen wendet man das Verfahren also nicht auf die Kovarianzmatrix, sondern auf die Korrelationsmatrix an.

3.1.1 Problemformulierung als Varianzmaximierung

Wir wollen nun die Intuition des minimalen Informationsverlust mathematisch beschreiben. Gegeben sei dazu eine Matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, wobei n die Anzahl der Samples bzw. Beobachtungen und p die Anzahl der Variablen ist. Wir nehmen im Folgenden ohne Beschränkung der Allgemeinheit an, dass die Variablen zuvor zentriert wurden. Aufgabe der Hauptkomponentenanalyse ist es nun sukzessive Richtungen größter Varianz zu finden. Die erste Hauptkomponente ist definiert durch $Z_1 = \sum_{j=1}^{p} v_{1j} X_j = \mathbf{X}v$ (Ist v*X korrekt?) wobei die Hauptachse $v_1 = (v_{11}, \ldots, v_{1p})^T$ so gewählt wird, dass die Varianz von Z_1 maximiert wird, d.h.

$$v_1 = \underset{\|v\|_2=1}{\arg\max} \operatorname{Var}[\mathbf{X}v] = \underset{\|v\|_2=1}{\arg\max}\, v^T \mathbf{K}_{xx} v$$

mit $\mathbf{K_{xx}} = \frac{\mathbf{X}^T\mathbf{X}}{n-1}$ als Stichprobenkovarianzmatrix. Die restlichen Hauptachsen können nun sukzessive definiert werden.

$$\begin{aligned} v_{k+1} &= \underset{\|v\|=1}{\arg\max}\, v^T \mathbf{K}_{xx} v \\ v_{k+1}^T v_l &= 0 \quad \forall 1 \leq l \leq k \end{aligned} \tag{3.1}$$

Man sucht also unter den Richtungen, die orthogonal zu allen bisherigen Hauptachsen sind, diejenige, die die Varianz maximiert. Wie oben beschrieben erhält man dann die Hauptkomponenten, also die Darstellung der Daten bezüglich der neu gefundenen Hauptachsen, durch Projektion der Daten $Z_i = \mathbf{X}v_i$. [ZX18] CITE JOLLIFE

Wie wir bereits in THEOREM gesehen haben, entsprechen die Eigenvektoren der Kovarianzmatrix genau den Richtungen maximaler Varianz. Daher können wir anstatt sukzessiver Berechnung einzelner Hauptachsen die Kovarianzmatrix $\mathbf{K}_{xx}$ direkt diagonalisieren. Dies ist möglich, da $\mathbf{K}_{xx}$ symmetrisch ist. Die Diagonalisierung ergibt

$$\mathbf{K}_{xx} = \mathbf{V}\mathbf{L}\mathbf{V}^T$$

wobei $\mathbf{L}$ eine Diagonalmatrix mit Eigenwerten λ_i und $\mathbf{V}$ die Matrix der Eigenvektoren ist, d.h. jede Spalte entspricht einem Eigenvektor von $\mathbf{K}_{xx}$. Somit können die Hauptachsen direkt aus $\mathbf{V}$ abgelesen werden. Die Projektion der Daten auf die Hauptachsen wird dann wie zuvor durch Multiplikation der Beobachtungen mit den Eigenvektoren erreicht.

$$\mathbf{Z} = \mathbf{X}\mathbf{V}$$

Die i-te Spalte in $\mathbf{Z}$ entspricht also der i-ten Hauptkomponente und die einzelnen Beobachtungen bezüglich der neuen Darstellung sind die Zeilen von $\mathbf{Z}$.

3.1.2 Formulierung als Singulärwertzerlegung

Es gibt einen engen Zusammenhang zwischen der Diagonalisierung der Kovarianzmatrix $\mathbf{K}_{xx} = \mathbf{X}^T\mathbf{X}$ und der Singulärwertzerlegung von $\mathbf{X}$. Diese Beziehung können wir nutzen, um das Problem neu zu formulieren. Eine Singulärwertzerlegung der Matrix $\mathbf{X}$ ergibt

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

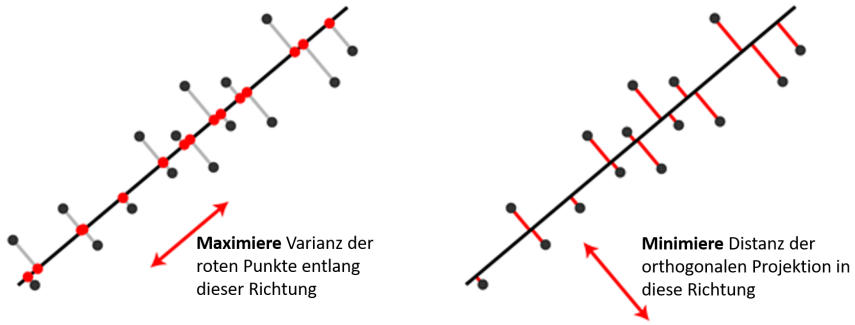


ABBILDUNG 3.2: Die Abbildung zeigt die Äquivalenz von Maximierung der Varianz und Minimierung der Distanz der orthogonalen Projektion

wobei $\mathbf{D}$ eine Diagonalmatrix mit Singulärwerten $d_1, \ldots, d_p$, $\mathbf{U}$ eine orthogonale $n \times p$ und $\mathbf{V}$ eine orthogonale $p \times p$ Matrix ist. Nun sieht man aufgrund der Orthogonalität von $\mathbf{U}$, dass

$$\mathbf{K}_{xx} = \mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{D}\mathbf{U}^T\mathbf{U}\mathbf{D}\mathbf{V}^T = \mathbf{V}\mathbf{D}^2\mathbf{V}^T$$

Wegen der Eindeutigkeit der Diagonalisierung(stimmt das?) ist $\mathbf{V}$ nun wie zuvor die Matrix der Eigenvektoren und somit der Hauptachsen. Ebenso stehen die Singulärwerte durch

$$\lambda_i = \frac{d_i^2}{n-1}$$

in Beziehung mit den Eigenwerten der Kovarianzmatrix. Die Hauptkomponenten kann man somit auch durch $\mathbf{XV} = \mathbf{UD}$ erhalten.

Computing PCA using Eigen value decomposition of the sample covariance matrix: We first have to compute the covariance matrix, which is $\mathcal{O}(p^2n)$ and then compute its eigenvalue decomposition which is $\mathcal{O}(p^3)$ giving a total cost of $\mathcal{O}(p^2n + p^3)$ (https://arxiv.org/pdf/1503.05214.pdf)

Computing PCA using SVD of the data matrix: Svd has a computational cost of $\mathcal{O}(p^2n)$

Numerical Stability? Which method is preferrable in the $n << p$ case?

3.1.3 Formulierung als beste Rang k Rekonstruktion

Further multiplying the first k PCs by the corresponding principal axes $\mathbf{V}^T k$ yields $\mathbf{X}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^T$ matrix that has the original $n \times p$ size but is of lower rank (of rank k). This matrix $\mathbf{X}_k$ provides a reconstruction of the original data from the first k PCs. It has the lowest possible reconstruction error

3.1.4 Formulierung als Regressionsproblem

Wir widmen uns nun einer letzten Formulierung der Hauptkomponentenanalyse, die eine geometrische Interpretation ermöglicht. Hierbei versucht man einen k-dimensionalen $(k < n)$ Unterraum zu finden, der die Daten bestmöglich approximiert. Wir werden diese Problemstellung nun mathematisch formulieren.

Sei dazu x_i die i-te Beobachtung, also die i-te Zeile von $\mathbf{X}$ und $\mathbf{V}_k = \begin{bmatrix} V_1 & \cdots & V_k \end{bmatrix}$ eine $p \times k$ orthonormale Matrix. Nun projizieren wir jede Beobachtung orthogonal auf den durch $V_1, \ldots V_k$ aufgespannten Unterraum. Die orthogonale Projektion wird wie in REF beschrieben durch Multiplikation mit dem Operator $\mathbf{V}_k \mathbf{V}_k^T$ erreicht. Die auf den linearen Unterraum projizierten Daten ergeben sich also durch $\mathbf{V}_k \mathbf{V}_k^T x_i$. Um die Daten bestmöglich in diesem niedrigdimensionalen Raum darzustellen minimiert man nun die Distanz zwischen jeder Beobachtung und seiner Projektion. Ein Weg, um die beste Projektion zu definieren ist l_2 Approximation erhält man folgendes Problem [ZHT06]: (Hier auch noch schreiben warum man den zweiten Term braucht, Eindeutigkeit von PCA)

$$\hat{\mathbf{V}}_k = \arg\min_{\mathbf{V}_k} \sum_{i=1}^{n} \left\| x_i - \mathbf{V}_k \mathbf{V}_k^T x_i \right\|^2$$

$$\mathbf{V}_k^T \mathbf{V}_k = I_{k \times k}$$

Man kann zeigen, dass die Lösung dieses Problems genau den ersten k Hauptachsen entspricht. Wir haben dies in einem Theorem 3.3 festgehalten. [VMS16] Zum besseren Verständnis hilft 3.2, welches die Äquivalenz von Maximierung der Varianz und Minimierung der orthogonalen Projektion verdeutlichen soll. Jeder Datenpunkt ist hier in 2 Dimensionen dargestellt. Versucht man nun die Daten bestmöglich auf einen 1-dimensionalen Unterraum, also eine Linie, orthogonal zu projizieren erhält man denselben Vektor, den man aus Sicht der Varianzmaximierung auch erhalten hätte.

Aus dieser Interpretation leitet sich auch der Name des linearen Dimensionsreduktionsverfahrens ab, denn die Daten werden auf den niedrigdimensionaleren Raum linear transformiert. Ausgehend von dieser Formulierung als Regressionsproblem werden wir im nächsten Kapitel die Variante der dünnbesetzten Hauptkomponentenanalyse beschreiben.

3.2 Selektion der Hauptkomponenten

Wie viele Hauptkomponenten sollen wir auswählen?

A simple approach is to choose the number of PCs for the variance to achieve a predetermined percentage. say 95Most existing approaches to determining the number of PC's use an index that is monotonlcally decreasing. The number of PC's is chosen when there is no significant decrease in the index after adding a PC. These approaches based on monotonic indices are subjective because (i) there may be a rather constant decrement in the index; and (ii) there can be more than one location which satisfies the criterion

Abbildung Scree Plot

Optimal singular threshold [GD14]

3.3 Grenzen der Anwendbarkeit

Obwohl die Hauptkomponentenanalyse in vielen Situationen helfen kann, Datensätze zu veranschaulichen und zu strukturieren, gibt es keine Garantie für sinnvolle Ergebnisse. Im Folgendem werden wir Szenarien beschreiben, bei denen unerwünschte Effekte bei der Verwendung dieses Verfahrens auftreten. Daher gilt es den Datensatz vorerst hinsichtlich folgender Gesichtspunkte zu untersuchen:

- Lineare Beziehung zwischen Variablen
- Korrelation der Variablen
- Vollständigkeit des Datensatzes
- Ausreißer in den Daten
- Anzahl an Beobachtungen in Relation zu Anzahl an Variablen

Wie in REF beschrieben versuchen wir Daten in einen niedrigdimensionaleren linearen oder affinen Unterraum zu transformieren. Es kann aber durchaus vorkommen, dass es keine lineare Beziehung zwischen den Variablen gibt. Nichtlineare Strukturen können von PCA nicht erfasst werden und gehen somit verloren. [VMS16] Vidal et al. zeigen diese Grenze konkret am Beispiel von Porträt-Fotos auf. Seit der Entstehung von PCA gab es aber zahlreiche nicht-lineare Erweiterungen. So nutzt zum Beispiel Kernel PCA den *Kernel Trick* aus, bei welchem man die Daten zuerst durch eine nichtlineare Transformation in ein höherdimensionalen Raum einbettet von dem man sich erhofft, dass die Daten in diesem linear verteilt. Erst anschließend wird dann die eigentliche Reduktion durchgeführt. Hierbei muss man die Daten aber nicht im höherdimensionalen Raum auswerten. CITE. Andere Erweiterungen, die allgemein unter *manifold learning* zusammengefasst werden können, basieren auf der Idee, dass die Dimension des Datensatz nur künstlich hoch ist. Man versucht die lokale Geometrie der Mannigfaltigkeit (Begriff erklären?) zu approximieren und damit direkt eine niedrigdimensionale Einbettung zu erhalten. Hierunter fallen zum Beispiel die multidimensionale Skalierung oder ISOMAP.

Damit der Datensatz für eine Dimensionsreduktion per PCA geeignet ist, müssen die verschiedenen Variablen einen gewissen Grad an Korrelation aufweisen. Im extremen Fall der Unabhängigkeit der Variablen bewirkt eine Hauptachsentransformation nichts. Reduziert man dann die Anzahl der Hauptkomponenten verliert man mit jeder Variable einen Großteil der Information.

Ein weiterer Gesichtspunkt ist die Vollständigkeit eines Datensatzes. Finden wir fehlende oder korrupte Einträge in unserem Datensatz vor, kann die klassische Hauptkomponentenanalyse ... . Für dieser Art Probleme existieren entsprechende Ergänzungen von PCA wie zum Beispiel in cite und cite. Ausreißer in den Daten können die Resultate drastisch beeinflussen. Genaue Effekte überlegen und CITE. Aus diesem Grund sollten Ausreißer vor der Anwendung von PCA entfernt werden.

Ausreißer in den Daten.

Anzahl der Variablen zu hoch.

Darüber hinaus gibt es noch eine Reihe Spezialfälle, bei denen Probleme auftreten können. So kann es zum Beispiel passieren, dass die relevanten Informationen in den Variablen mit niedriger Varianz versteckt sind. Da die Hauptkomponentenanalyse gerade diese Variablen vernachlässigt, wird sich unter Umständen nicht die

Loss Functions	regularizer	constraints
quadratic	L2 norm	Nonnegative
(real data)	(small factors)	(additive factors)
absolute	L1 norm	
(robust to outliers)	(sparse factors)	
logistic	Derivative penalties	
(binary data)	(smooth factors)	
Poisson		
(integer data)		
circular		
(angular data)		

TABELLE 3.1: Allgemeines Schema zu PCA Erweiterungen

erwünschte Struktur auf den Daten ergeben. Es bedarf anderer Methoden mit anderen Ansätzen, um eine Dimensionsreduktion zu ermöglichen. Oftmals weiß man aber im Vorhinein nicht, in welchen Variablen diese Unterscheidungsmöglichkeit versteckt ist.

Das wohl wichtigste/größte Hindernis im Zuge dieser Arbeit ist sicherlich die durch die Transformation entstehenden Interpretationsschwierigkeiten. Jede Hauptkomponente entsteht wie oben beschrieben durch eine Linearkombination der Ausgangsvariablen. Während die Ausgangsvariablen Bedeutungen wie Gewicht oder Größe hatten ist in vor allem in hochdimensionalen Fällen eine Interpretation der Hauptkomponenten nur schwer möglich (Rotation Techniques CITE). Dieser Interpretationsverlust ist Ausgangspunkt der Idee der dünnbesetzten Hauptkomponentenanalyse, genannt sparse PCA. Diesem Verfahren ist das folgende Kapitel gewidmet.

3.4 Erweiterungen der Hauptkomponentenanalyse

Wie wir bereits gesehen haben, gibt es viele verschiedene Erweiterungen von PCA. Die meisten kann man unter folgendem Schema zusammenfassen: (Welche genau?)

$$\min_{\mathbf{U},\mathbf{V}} \underbrace{\left\|\mathbf{X}-\mathbf{U}\mathbf{V}^T\right\|_F}_{\text{Loss Function}} + \underbrace{\lambda_u f_u(\mathbf{U}) + \lambda_v f_v(\mathbf{V})}_{\text{Regularisierung}}$$
$$\text{subject to} \quad \underbrace{\mathbf{U} \in \Omega_u, \mathbf{V} \in \Omega_v}_{\text{Nebenbedingungen}}$$

3.5 Implementierung

Allgemein ist dies kein konvexes Problem, aber bikonvex, also in jeder Komponente. Somit ergibt sich der einfache folgende Algorithmus

3.6 Theoretische Aussagen

non convex problem that can be solved efficiently by truncated SVD.

Algorithm 1 Alternating minimization

```
1: procedure ALTERNATE(U, V)
2:     choose initial starting Points W^(0) and C^(0)
3:     n ← 0
4:     while not converged do                ▷ Definiere Abbruchkriterium
5:         W^(n+1) ← minimize over W while holding C = C^(n) constant.
6:         C^(n+1) ← minimize over C while holding W = W^(n+1) constant.
7:         n ← n + 1
8:     end while
9: end procedure
```

Baldi Hornik 1989 all local minima are solutions to pca all non optimal critical points are saddle points or maxima

Theorem 3.1. *PCA always gives unique solution.*

Theorem 3.2 ([VMS16])**.** *Sei* $\mathbf{X} \in \mathbb{R}^n$ *und* $\mathbf{A}_{p,k} = [\alpha_1, \dots \alpha_k]$

Theorem 3.3. *PCA inconsistent for* $n \ll p$.

Kapitel 4

Dünnbesetzte Hauptkomponentenanalyse

Ein wesentlicher Nachteil der Hauptkomponentenanalyse besteht darin, dass sich die neuen Variablen aus einer Linearkombination *aller* bestehenden Variablen zusammensetzt. Dies erschwert besonders für hochdimensionale Daten eine Interpretation der Hauptachsen. Während zuvor jede Variable eine Bedeutung hatte, sind wir nach der Transformation meist nicht in der Lage den Hauptachsen eine Bedeutung im Kontext zuzuweisen. Um zu verstehen, was die Hauptachsen im Modell repräsentieren kann es besonders hilfreich sein, wenn diese *dünnbesetzt* sind, sich also nur aus wenigen Variablen zusammensetzen. Treffen wir irgendwelche Annahmen? Des Weiteren ist nicht jede Variable relevant zur Strukturerkennung. impose extra constraints, which sacrifices some variance in order to improve interpretability

Zu Anfang dieses Kapitels werden wir eine naheliegende mathematische Formulierung des Problems beschreiben. Leider wird sich diese als NP-vollständig herausstellen, weshalb wir in Abschnitt 4.2 verschiedene Wege aufzeigen, dass Problem zu relaxieren. In 4.3 möchten wir uns mit einem dieser Ansätze intensiv beschäftigen, welcher den Ausgangspunkt für den weiteren Verlauf dieser Arbeit darstellt. Der Rest dieses Kapitels ist den Details dieses Ansatzes gewidmet.

4.1 Problemformulierung

Wir möchten nun Hauptachsen eines gegebenen Datensatzes identifizieren mit der Zusatzbedingung, dass diese dünnbesetzt sind. Die wohl einfachste Möglichkeit dafür ist, zuerst die gewöhnliche Hauptkomponentenanalyse durchzuführen und anschließend ein Schwellwertmethode (Englisch: simple thresholding) auf die Hauptachsen anzuwenden. Hierbei vernachlässigt man alle Koeffizienten, die kleiner als ein bestimmter Schwellenwert sind, indem man sie auf 0 setzt. Eine solche Prozedur kann aber in vielen Fällen irreführend sein, unter welcher die Qualität der Ergebnisse leidet. [CJ95] Die Wichtigkeit einer Variable in den Hauptachsen wird nicht allein durch den Koeffizient bestimmt. Zu berücksichtigen sind unter anderem sowohl die Standardabweichung als auch die Korrelationen mit anderen Variablen. Bei einer Schwellwertmethode werden diese Faktoren nicht beachtet, weshalb den Ergebnissen im Allgemeinen nicht vertraut werden darf.

Hier Regression on ordinary PCA's mit Zou et al?

Anstelle eines zweischrittigen Ansatzes kann die Dünnbesetzung direkt in die Problemformulierung mit eingebaut werden. Gegeben sei dazu wieder eine Datenmatrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, wobei n die Anzahl an Beobachtungen und p die Anzahl an Variablen ist. Des Weiteren gehen wir davon aus, dass die Matrix $\mathbf{X}$ zuvor spaltenweise zentriert wurde. Dann kann die dünnbesetzte Hauptkomponentenanalyse als sukzessives Maximierungsproblem formuliert werden:

$$\begin{aligned} v_k &= \underset{\|v\|_2=1}{\arg\max}\, v^T \mathbf{K}_{xx} v \\ v_k^T v_l &= 0 \quad \forall 1 \leq l < k \\ \text{unter der Nebenbedingung, dass } &\|v_k\|_0 \leq t \end{aligned} \tag{4.1}$$

wobei $\mathbf{K}_{xx} = \frac{\mathbf{X}^T\mathbf{X}}{n-1}$ die empirische Kovarianzmatrix ist. Der einzige Unterschied zu der gewöhnlichen Hauptkomponentenanalyse, wie wir sie in (3.1) beschrieben haben, besteht in der Einführung der ℓ_0-Norm. Somit beschränken wir uns auf die Suche von Hauptachsen, welche höchstens t von Null verschiedene Einträge haben. Wählen wir $t = p$ reduziert sich das Problem auf die gewöhnliche Hauptkomponentenanalyse. Während (4.1) eine sehr schöne und einfache mathematische Formulierung ist, wurde gezeigt, dass dieses Problem NP-vollständig ist [FR13]. Zur Berechnung dünnbesetzter Hauptachsen sind wir also angehalten eine geeignete Relaxation zu finden.

4.2 Relaxation

Es existiert eine Vielfalt an Ansätzen, um das Problem zu relaxieren. Wir wollen zunächst einen kleinen Überblick über die unterschiedlichen Ideen geben und uns anschließend mit einer genauer beschäftigen. Eine selektive Übersicht der verschiedenen Ansätze haben wir hier erstellt.

SCoTLASS

Inspiriert von der LASSO Regression [Tib96] schlugen Jolliffe et al. [JTU03] vor, die ℓ_1-Norm anstelle der ℓ_0-Norm als Strafterm zu verwenden. Wie wir bereits in Abschnitt 2.3.4 gesehen haben, kann die ℓ_1-Norm genutzt werden, um dünnbesetzte Vektoren zu erhalten. Somit liegt es nahe das Problem wie folgt zu formulieren.

$$\begin{aligned} v_k &= \underset{\|v\|_2=1}{\arg\max}\, v^T \mathbf{K}_{xx} v \\ v_k^T v_l &= 0 \quad \forall 1 \leq l < k \\ \text{unter der Nebenbedingung, dass } &\|v_k\|_1 \leq t \end{aligned} \tag{4.2}$$

Wie in (4.1) hat man mit der Wahl der Parameters t Einfluss auf die Dünnbesetzung der Hauptachsen. Aufgrund der hohen Berechnungskosten ist SCoTLASS allerdings für hochdimensionale Daten ungeeignet. Diese sind vor allem darauf zurückzuführen, dass (4.2) kein konvexes Optimierungsproblem ist. Des Weiteren ergeben sich Schwierigkeiten bei der Wahl des Hyperparameters t. Auch wenn eine passende Wahl eine gewünschte Dünnbesetzung hervorruft, gibt es kaum Orientierungshilfen. Dabei hat ScoTLASS dasselbe grundlegende Problem wie das Lasso. Die Anzahl von null verschiedener Einträge ist durch die Anzahl Beobachtungen im Datensatz limitiert, welches die Brauchbarkeit des Modells deutlich einschränkt. Zusammen

mit den hohen Berechnungskosten ist dieser Ansatz in der Praxis daher meist impraktikabel.

Semidefinite Programmierung

Konvexe Relaxation ist eine Standard-Technik, um mit schwierigen nichtkonvexen Problemen umzugehen. d'Aspremont et al. [dAs+07] entwickeln eine Ansatz, welcher sich als semidefinites Programmierungsproblem ausdrücken lässt. Zunächst werden wir (4.1) dafür mit Matrizen reformulieren.

Sei $\mathbf{V} = v_k v_k^\top$. Dann übersetzen sich die Nebenbedingungen

$$\begin{aligned} &\arg\max_{\mathbf{P}} = \operatorname{tr}(\mathbf{K}_{xx}\mathbf{P}) \\ \operatorname{tr}(\mathbf{P}) = 1, \quad &\|\mathbf{P}\|_0 \leq k^2, \quad \mathbf{P} \geq 0, \quad \operatorname{rank}(\mathbf{P}) = 1 \end{aligned} \tag{4.3}$$

Diese Formulierung ist noch immer nichtkonvex aufgrund der Rang-Bedingung und der ℓ_0-Strafterm. Per Definition ist $\mathbf{P}$ symmetrisch und $\mathbf{P}^2 = \mathbf{P}$. Somit ist

$$\|\mathbf{P}\|_F^2 = \operatorname{tr}\left(\mathbf{P}^\top \mathbf{P}\right) = \operatorname{tr}(\mathbf{P}) = 1$$

und mit der Cauchy-Schwarz-Ungleichung folgt

$$\mathbf{1}_p^\top |\mathbf{P}| \mathbf{1}_p \leq \sqrt{\|\mathbf{P}\|_0 \|\mathbf{P}\|_F^2} \leq k$$

Ersetzen wir die ℓ_0-Strafterm und lassen die Rang-Bedingung fallen erhalten wir die DSPCA-Formulierung

$$\begin{aligned} &\arg\max_{\mathbf{P}} = \operatorname{tr}(\mathbf{K}_{xx}\mathbf{P}) \\ \operatorname{tr}(\mathbf{P}) = 1, \quad &\mathbf{1}_p^\top |\mathbf{P}| \mathbf{1}_p \leq k, \quad \mathbf{P} \geq 0 \end{aligned} \tag{4.4}$$

Dies stellt ein semidefinites Programmierungsproblem dar, bei welcher die zu optimierenden Variablen symmetrische Matrizen sind unter der Nebenbedingung, dass sie positiv semidefinit sind. Für kleine Probleme kann (4.4) effizient durch *Innere-Punkte-Verfahren* (English: *interior-point methods*) gelöst werden. SDPT3 [TTT99].

In (4.4) wird allerdings $\mathbf{P}$ berechnet und nicht die eigentliche Hauptachse. Hierfür kürzen d'Aspremont et al. die Matrix $\mathbf{P}$ und behalten nur den größten Eigenvektor v_k. Anschließend erhält man weitere Hauptachsen durch Matrix Deflation, indem wir $\mathbf{K}_{xx}$ durch

$$\mathbf{K}_{xx} - (v_k^\top \mathbf{K}_{xx} v_k) v_k v_k^\top$$

ersetzen. Für größere Probleme wird eine Methode von Nesterov benutzt, um eine Laufzeit von $\mathcal{O}(\frac{\sqrt[4]{\log p}}{\epsilon})$ zu erreichen.

iterative Schwellenwert-Methoden

Basierend auf der Formulierung REF von PCA als beste Rang k Approximation an

die Datenmatrix $\mathbf{X}$ haben Shen und Huang [SH08] das folgende Optimierungsproblem formuliert

$$(u_1, v_1) = \arg\min_{u,v} \left\| \mathbf{X} - uv^\top \right\|_F^2 + \lambda \|v\|_1$$
$$\|u\|_2 = 1$$

Somit erhält man mit $\frac{v_1}{\|v_1\|}$ die erste dünnbesetzte Hauptachse. Auch hier werden die restlichen Hauptachsen sequentiell berechnet durch Ersetzen der Datenmatrix $\mathbf{X}_{(k+1)} = \mathbf{X} - \sum_{i=1}^{k} u_i v_i^\top$. Jede Iteration kann durch ein alternierendes Minimierungsverfahren gelöst werden. Fixiert man v, so ist das optimale $u = \frac{\mathbf{X}v}{\|\mathbf{X}v\|}$. Andererseits reduziert sich (**??**) bei festem u auf

$$\arg\min_{v} -2\mathrm{tr}\left(\mathbf{X}^\top uv^\top\right) + \|v\|^2 + \lambda \|v\|_1 .$$

Eine explizite Lösung ist durch den soft-thresholding Operator gegeben

$$v = \mathrm{soft}_{\frac{\lambda}{2}}(\mathbf{X}^\top U)$$

welche in Abschnitt 2.3 eingeführt worden ist.

Diese Methode ist sehr ähnlich zu der von Zou et al. [ZHT06]. Der große Unterschied besteht darin, dass die Hauptachsen dort nicht sequentiell, sondern gleichzeitig berechnet werden.

Es gibt noch eine Reihe weiterer Ideen, die in der Literatur betrachtet wurden. Dazu gehören

- eine verallgemeinerte Potenzmethode [Jou+10]
- ein alternierendes Maximierungs-Netzwerk [Ric12]
- vorwärts und rückwärts greedy Suche und exakte Methoden mittels Branch-and-Bound-Verfahren [MWA06]
- ein Bayes Formulierung [GD09]

Ein interessierter Leser

4.3 Konstruktion Sparse PCA

Wir werden uns nun mit dem von Zou, Hastie und Tibshirani (2006) in [ZHT06] eingeführten Ansatz ausführlich beschäftigen. Zou und Hastie führten zuvor in [ZH05] das sog. *elastic net* ein, welches den Grundstein für die mathematische Formulierung legt.

Zunächst werden wir einen zweischrittigen Ansatz betrachten???

Wie bereits in Kapitel 3 beschrieben kann die Hauptkomponentenanalyse auch als Regressionsproblem betrachtet werden. Das folgende Theorem erweitert die bisherige Formulierung, indem nun nicht ausschließlich orthogonale Projektionen erlaubt werden. Im Folgenden bezeichnet k die Anzahl an Hauptkomponenten, die wir extrahieren möchten und x_i die i-te Zeile von $\mathbf{X}$.

Theorem 4.1. *Sei* $\mathbf{A}_{p\times k} = [\alpha_1, \ldots, \alpha_k]$ *und* $\mathbf{B}_{p\times k} = [\beta_1, \ldots, \beta_k]$. *Für ein* $\lambda > 0$ *sei*

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \arg\min_{\mathbf{A},\mathbf{B}} \sum_{i=1}^{n} \left\| x_i - \mathbf{A}\mathbf{B}^T x_i \right\|^2 + \lambda \sum_{j=1}^{k} \|\beta_j\|^2$$

$$\textit{wobei } \mathbf{A}^T\mathbf{A} = I_{k\times k}$$

Dann ist $\hat{\beta}_j \propto V_j$ *für* $j = 1, 2, \ldots, k$.

Fordern wir $\mathbf{A} = \mathbf{B}$ so reduziert sich das Problem auf die normale Hauptkomponentenanalyse wie in (WENN A = B, dann können wir ridge penalty weglassen. Also zeigt das Theorem, dass wir immer noch exact PCA haben können, wenn wir die Bedingung B = A relaxieren und eine ridge-penalty hinzufügen.) 3.1.4 beschrieben. Theorem 4.1 FIX CROSS REFERENCES

Somit ergibt sich das folgende Kriterium, welches wir im Folgenden als das Sparse PCA Kriterium bezeichnen werden.

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \arg\min_{\mathbf{A},\mathbf{B}} \sum_{i=1}^{n} \left\| x_i - \mathbf{A}\mathbf{B}^T x_i \right\|_2^2 + \lambda \sum_{j=1}^{k} \|\beta_j\|_2^2 + \sum_{j=1}^{k} \lambda_{1,j} \|\beta_j\|_1$$

subject to $\mathbf{A}^T\mathbf{A} = I_{k\times k}$ oblique projections AB

4.4 Anpassung der Varianzen

4.5 Theoretische Aussagen

z.B. wie werden neue Varianzen berechnet

Alexandre d'Aspremont, Optimal Solutions for Sparse Principal Component Analysis We then use the same relaxation to derive sufficient conditions for global optimality of a solution, which can be tested in O(n 3) per pattern. We discuss applications in subset selection and sparse recovery and show on artificial examples and biological data that our algorithm does provide globally optimal solutions in many cases.

Kapitel 5

Implementierung

In diesem Kapitel werden wir einen Algorithmus beschreiben, der das Sparse PCA Kriterium minimiert. Dazu werden wir uns zunächst mit dem allgemeinen Fall beschäftigen bevor wir den Fall $n \ll p$ genauer betrachten, um eine effiziente Berechnung zu garantieren. Zu Schluss dieses Kapitels werden wir uns genauer mit der Laufzeit des Algorithmus auseinandersetzen.

5.1 Numerische Lösung

Zunächst möchten wir das Sparse PCA Kriterium erneut formulieren. Sei $\mathbf{A}_{p\times k} = [\alpha_1, \ldots, \alpha_k]$ und $\mathbf{B}_{p\times k} = [\beta_1, \ldots, \beta_k]$. Wie zuvor bezeichnen wir mit x_i die i-te Zeile von $\mathbf{X}$. Dann lautet das Sparse PCA Kriterium

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \arg\min_{\mathbf{A},\mathbf{B}} \sum_{i=1}^{n} \left\| x_i - \mathbf{A}\mathbf{B}^T x_i \right\|_2^2 + \lambda \sum_{j=1}^{k} \|\beta_j\|_2^2 + \sum_{j=1}^{k} \lambda_{1,j} \|\beta_j\|_1$$

$$\text{unter der Nebenbedingung, dass } \mathbf{A}^T\mathbf{A} = I_{k\times k}$$

Durch die Einführung der Matrix $\mathbf{B}$ in das Kriterium in Kapitel 4 wird über die beiden Matrizen $\mathbf{A}$ und $\mathbf{B}$ minimiert. Man kann zeigen, dass es sich bei (...) um ein nicht-konvexes Optimierungsproblem handelt. (CITE) Allerdings stellt sich heraus, dass das Problem konvex für festes $\mathbf{A}$ bzw. festes $\mathbf{B}$ ist. (Überprüfung!!!) Daher liegt es nahe einen alternierenden Ansatz zu wählen, um das Problem numerisch zu lösen. Wir betrachten im Folgenden also zwei Optimierungsprobleme.

$\mathbf{B}$ gegeben $\mathbf{A}$:

Wir wenden uns zunächst der Zielfunktion zu. Hierfür sei $\mathbf{A}_\perp \in \mathbb{R}^{p\times(p-k)}$ eine orthonormale Matrix, so dass $[\mathbf{A}; \mathbf{A}_\perp]$ $p \times p$ orthonormal ist. Dann gilt

$$\begin{aligned}
\sum_{i=1}^{n} \left\| x_i - \mathbf{A}\mathbf{B}^T x_i \right\|_2^2 &= \left\| \mathbf{X} - \mathbf{X}\mathbf{B}\mathbf{A}^T \right\|_F^2 \\
&= \|\mathbf{X}\mathbf{A}_\perp\|_F^2 + \|\mathbf{X}\mathbf{A} - \mathbf{X}\mathbf{B}\|_F^2 \\
&= \|\mathbf{X}\mathbf{A}_\perp\|_F^2 + \sum_{j=1}^{k} \|\mathbf{X}\alpha_j - \mathbf{X}\beta_j\|_2^2
\end{aligned}$$

Wir setzen $Y_j^* = \mathbf{X}\alpha_j$ für alle $1 \leq j \leq n$. Somit reduziert sich (...) für fixes $\mathbf{A}$ auf das Lösen von k elastic net Problemen

$$\hat{\beta}_j = \arg\min_{\beta_j} \left\| Y_j^* - \mathbf{X}\beta_j \right\|_2^2 + \lambda \left\| \beta_j \right\|^2 + \lambda_{1,j} \left\| \beta_j \right\|_1$$

In Kapitel 2 haben wir uns bereits mit elastic nets beschäftigt und einen effizienten Algorithmus zur Lösung dieser präsentiert.

A gegeben B:

Fixieren wir die Matrix $\mathbf{B}$, so können wir uns auf das Minimieren der Zielfunktion $\sum_{i=1}^n \left\| x_i - \mathbf{A}\mathbf{B}^T x_i \right\|_2^2 = \left\| \mathbf{X} - \mathbf{X}\mathbf{B}\mathbf{A}^T \right\|_F^2$ beschränken, da die Bedingungen an β_j nicht von Relevanz beim Optimieren über $\mathbf{A}$ sind. Somit ergibt sich

$$\hat{\mathbf{A}} = \arg\min_{\mathbf{A}} \left\| \mathbf{X} - \mathbf{X}\mathbf{B}\mathbf{A}^T \right\|_F^2$$

$$\text{unter der Nebenbedingung, dass } \mathbf{A}\mathbf{A}^T = \mathbf{I}_{k\times k}$$

Für dieses Optimierungsproblem lässt sich eine explizite Lösung angeben, da es eine Form von Procrustes Rotationsproblem ist, welches wir in Theorem REF beschrieben haben. Sei daher

$$(\mathbf{X}^T\mathbf{X})\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

eine Singulärwertzerlegung. Dann ist $\hat{\mathbf{A}} = \mathbf{U}\mathbf{V}^T$.

Es ist sinnvoll zu erwähnen, dass für die Lösung beider Teilprobleme nur die Gram-Matrix $\mathbf{X}^T\mathbf{X}$ bekannt sein muss. Dies erleichtert die Berechnung.

5.2 Algorithmus

Durch die Vorarbeit im vorangegangenem Abschnitt können wir einen effizienten Algorithmus zur Lösung des Sparse PCA Kriteriums angeben. Zuerst initialisieren wir $\mathbf{A}$ mit den ersten k Hauptachsen. Anschließend minimieren wir abwechselnd über $\mathbf{B}$ gegeben $\mathbf{A}$ und $\mathbf{A}$ gegeben $\mathbf{B}$ solange ein geeignetes Konvergenzkriterium noch nicht erfüllt ist. Normalisieren wir die Spalten von $\mathbf{B}$ erhalten wir die dünnbesetzten Hauptachsen. Eine Übersicht haben wir in Algorithmus 2 erstellt.

Es stellt sich nun die Frage nach einem passendem Abbruchkriterium. Da für uns am Schluss des Algorithmus nur die Matrix $\mathbf{B}$ relevant ist, kann $\mathbf{A}$ bei der Wahl eines Konvergenzkriterium vernachlässigt werden. Zou, Hastie, und Tibshirani wählen in ihrer Implementierung das folgende Kriterium

$$\max_{\substack{1\leq i\leq p \\ 1\leq j\leq k}} \left| \mathbf{B}_{ij}^{(l+1)} - \mathbf{B}_{ij}^{(l)} \right| < \epsilon$$

wobei $\mathbf{B}_{ij}^{(l)}$ der ij-te Eintrag der Matrix in der l-ten Iteration ist. Sobald also die Änderung in $\mathbf{B}^{(l)}$ klein genug ist, kann die while-Schleife abgebrochen werden. Um die Laufzeit des Algorithmus zu beschränken, ist es meist sinnvoll ein zusätzliches Abbruchkriterium zu definieren. So werden wir bei Anwendung des Algorithmus eine maximale Anzahl an Iterationen l_{max} festlegen, die nicht überschritten werden darf.

Algorithm 2 Sparse Principal Component Analysis

procedure SPCA($\mathbf{A}, \mathbf{B}$)
  $\mathbf{A} \leftarrow \mathbf{V}[, 1 : k]$, die ersten k Hauptachsen
  while nicht konvergiert **do**
    Gegeben festes $\mathbf{A} = [\alpha_1, \ldots, \alpha_k]$, löse das elastic net Problem

$$\hat{\beta}_j = \arg\min_{\beta_j} \left\|\mathbf{X}\alpha_j - \mathbf{X}\beta_j\right\|^2 + \lambda \left\|\beta_j\right\|^2 + \lambda_{1,j} \left\|\beta_j\right\|_1 \quad \text{für } j = 1, \ldots, k$$

    Gegeben festes $\mathbf{B} = [\beta_1, \ldots, \beta_k]$, berechne die Singulärwerzerlegung von

$$\mathbf{X}^T\mathbf{X}\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

$$\mathbf{A} \leftarrow \mathbf{U}\mathbf{V}^T$$

  end while
  $\hat{V}_j \leftarrow \frac{\hat{\beta}_j}{\|\beta_j\|}$ for $j = 1, \ldots, k$
end procedure

(In Kapitel 6 werden wir sehen, dass sich die Anzahl an Iterationen stark unterscheiden kann.)

5.3 Numerische Lösung im Fall p » n

5.4 Implementierung in scikit-learn in python

5.5 Laufzeitvergleich

Kapitel 6

Anwendung

6.1 Anwendung auf Simulationsdaten

Vergleich Tabelle PCA / Sparse PCA (Loadings)

6.2 Der Datensatz

6.3 Anwedung auf Frequenzdaten

6.4 Auswertung der Ergebnisse

6.5 Vergleich mit PCA Resultaten

6.6 Hyperparameter

Veränderung des Hyperparameters und dessen Effekte

6.6.1 Zeit

6.6.2 Effekt auf Resultate

Kapitel 7

Ausblick / Zusammenfassung

7.1 Einsetzbarkeit

Wann ist die Methode sinnvoll einzusetzen?

7.2 Übertragbarkeit

Übertragbarkeit auf andere Datensätze

7.3 Ongoing Research / Weitere Techniken

Literatur

[1] Howard Anton. *Lineare Algebra: Einführung, Grundlagen, Übungen*. Spektrum Akademischer Verlag, 1998. ISBN: 9783827403247.

[2] Y. Murali Mohan Babu, M. V. Subramanyam und M. N. Giri Prasad. „PCA based image denoising“. In: 2012. URL: https://doi.org/10.5121/sipij.2012.3218.

[3] Jorge Cadima und Ian T. Jolliffe. „Loading and correlations in the interpretation of principle compenents“. In: *Journal of Applied Statistics* 22.2 (1995), S. 203–214. URL: https://doi.org/10.1080/757584614.

[4] Scott Shaobing Chen, David L. Donoho und Michael A. Saunders. „Atomic Decomposition by Basis Pursuit“. In: *SIAM J. Sci. Comput.* 20.1 (1998), S. 33–61. URL: https://doi.org/10.1137/S1064827596304010.

[5] Alexandre d'Aspremont u. a. „A Direct Formulation for Sparse PCA Using Semidefinite Programming“. In: *SIAM Review* 49.3 (2007), S. 434–448. URL: http://www.jstor.org/stable/20453990.

[6] Carl Eckart und Gale Young. „The approximation of one matrix by another of lower rank“. In: *Psychometrika* 1.3 (1936), S. 211–218. URL: https://doi.org/10.1007/BF02288367.

[7] Gerd Fischer. *Lineare Algebra: Eine Einführung für Studienanfänger*. 18. Aufl. Vieweg+Teubner Verlag, 2013. ISBN: 9783834893659.

[8] Simon Foucart und Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. Bd. 1. Birkhäuser Basel, 2013.

[9] Jerome Friedman, Trevor Hastie und Rob Tibshirani. „Regularization paths for generalized linear models via coordinate descent“. In: *Journal of statistical software* 33.1 (2010), S. 1. URL: https://doi.org/10.18637/jss.v033.i01.

[10] Matan Gavish und David L. Donoho. „The Optimal Hard Threshold for Singular Values is $4/\sqrt{3}$“. In: *IEEE Transactions on Information Theory* 60.8 (2014), S. 5040–5053. URL: https://doi.org/10.1109/TIT.2014.2323359.

[11] John C Gower, Garmt B Dijksterhuis u. a. *Procrustes problems*. Bd. 30. Oxford University Press on Demand, 2004.

[12] Yue Guan und Jennifer G. Dy. „Sparse probabilistic principal component analysis“. In: Bd. 5. 2009, S. 185–192.

[13] Trevor Hastie, Robert Tibshirani und Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Bd. 2. Springer-Verlag New York, 2009.

[14] Matthias Hieber. „Analysis in metrischen Räumen“. In: *Analysis II*. Springer Berlin Heidelberg, 2019. ISBN: 978-3-662-57542-0.

[15] Klaus Jänich. *Lineare Algebra*. 11. Aufl. Springer-Lehrbuch. Springer Berlin Heidelberg, 2007. ISBN: 9783540755029.

[16] Ian T. Jolliffe, Nickolay T. Trendafilov und Mudassir Uddin. „A Modified Principal Component Technique Based on the LASSO“. In: *Journal of Computational and Graphical Statistics* 12.3 (2003), S. 531–547. URL: https://doi.org/10.1198/1061860032148.

[17] Michel Journée u. a. „Generalized power method for sparse principal component analysis“. In: *Journal of Machine Learning Research* 11.Feb (2010), S. 517–553. URL: https://dl.acm.org/doi/10.5555/1756006.17560210.

[18] Wolfgang Kohn. *Statistik: Datenanalyse und Wahrscheinlichkeitsrechnung*. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2005. ISBN: 978-3-540-26768-3.

[19] Baback Moghaddam, Yair Weiss und Shai Avidan. „Spectral Bounds for Sparse PCA: Exact and Greedy Algorithms“. In: *Advances in Neural Information Processing Systems 18*. Hrsg. von Y. Weiss, B. Schölkopf und J. C. Platt. MIT Press, 2006, S. 915–922. URL: http://papers.nips.cc/paper/2780-spectral-bounds-for-sparse-pca-exact-and-greedy-algorithms.pdf.

[20] Kevin P Murphy. *Machine learning: A Probabilistic Perspective*. Cambridge, MA: MIT press, 2012.

[21] F. Pedregosa u. a. „Scikit-learn: Machine Learning in Python“. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830.

[22] Richtárik, Peter and Takáč, Martin and Ahipaşaoğlu, Selin Damla. *Alternating Maximization: Unifying Framework for 8 Sparse PCA Formulations and Efficient Parallel Codes*. 2012. URL: https://arxiv.org/abs/1212.4137.

[23] Ludger Rüschendorf. *Mathematische Statistik*. Berlin: Springer Spektrum, 2014. ISBN: 978-3-642-41996-6.

[24] Robert Schaback und Holger Wendland. *Numerische Mathematik*. 5. Aufl. Mathematics and Statistics. Springer, 2004. ISBN: 9783540213949.

[25] Haipeng Shen und Jianhua Z Huang. „Sparse principal component analysis via regularized low rank matrix approximation“. In: *Journal of multivariate analysis* 99.6 (2008), S. 1015–1034. URL: https://doi.org/10.1016/j.jmva.2007.06.007.

[26] Jiang Tai-Xiang u. a. „Patch-Based Principal Component Analysis for Face Recognition“. In: *Computational Intelligence and Neuroscience* (2017). URL: https://doi.org/10.1155/2017/5317850.

[27] Robert Tibshirani. „Regression Shrinkage and Selection via the Lasso“. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), S. 267–288. URL: http://www.jstor.org/stable/2346178.

[28] Robert Tibshirani u. a. *Diabetes Data*. https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html. Zugriff: 20. Januar 2020.

[29] Robert Tibshirani u. a. „Least angle regression“. In: *The Annals of Statistics* 32.2 (2004), S. 407–499. URL: http://dx.doi.org/10.1214/009053604000000067.

[30] Kim-Chuan Toh, Michael J Todd und Reha H Tütüncü. „SDPT3 — a MATLAB software package for semidefinite programming“. In: *Optimization methods and software* 11.1-4 (1999), S. 545–581. URL: https://doi.org/10.1080/10556789908805762.

[31] L. Vandenberghe. „Optimization Methods for Large-Scale Systems“. In: *Lecture Notes ECE236C, UCLA* (2019). URL: http://www.seas.ucla.edu/~vandenbe/ee236c.html.

[32] R. Vidal, Y. Ma und S. Sastry. *Generalized Principal Component Analysis*. Bd. 1. Interdisciplinary Applied Mathematics. Springer New York, 2016. ISBN: 9780387878119. URL: https://doi.org/10.1007/978-0-387-87811-9.

[33] Allen Y. Yang u. a. „Fast ℓ_1-Minimization Algorithms for Robust Face Recognition“. In: *IEEE Transactions on Image Processing* 22.8 (Aug. 2013), S. 3234–3246. ISSN: 1941-0042. URL: http://dx.doi.org/10.1109/TIP.2013.2262292.

[34] Ming Yuan und Yi Lin. „Model selection and estimation in regression with grouped variables“. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1 (2006), S. 49–67.

[35] Hui Zou und Trevor Hastie. „Regularization and Variable Selection via the Elastic Net“. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 67.2 (2005), S. 301–320. URL: http://www.jstor.org/stable/3647580.

[36] Hui Zou, Trevor Hastie und Robert Tibshirani. „Sparse Principal Component Analysis“. In: *Journal of Computational and Graphical Statistics* 15.2 (2006), S. 265–286. URL: https://doi.org/10.1198/106186006X113430.

[37] Hui Zou und Lingzhou Xue. „A Selective Overview of Sparse Principal Component Analysis“. In: *Proceedings of the IEEE* 106.8 (2018), S. 1311–1320. URL: https://ieeexplore.ieee.org/document/8412518.