Bonner Zentrum für Lehrerbildung (BZL)

Singulärwertzerlegung. Theorie und Anwendung

Bachelorarbeit im Fach: Mathematik

vorgelegt von Noah Pferdekamp
Matrikelnummer: 123456

Erstbetreuer: Prof. Dr. Philipp Hieronymi
Zweitgutachter: Dr. Thoralf Räsch
Mathematisches Institut

Wintersemester 2024/2025
Bonn, den 31. März 2025

Selbstständigkeitserklärung

Ich versichere hiermit, dass die Bachelorarbeit mit dem Titel „Singulärwertzerlegung. Theorie und Anwendung“ von mir selbst und ohne jede unerlaubte Hilfe selbstständig angefertigt wurde, dass sie noch an keiner anderen Hochschule zur Prüfung vorgelegen hat und dass sie weder ganz noch in Auszügen veröffentlicht worden ist. Die Stellen der Arbeit — einschließlich Tabellen, Karten, Abbildungen usw. —, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen sind, habe ich in jedem einzelnen Fall kenntlich gemacht.

Bonn, den 31. März 2025

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Notation

$\mathbf{0}$	Nullmatrix
$\overline{z}$	komplexe Konjugation
$\mathfrak{Re}$	Realteil
$\mathfrak{Im}$	Imaginärteil
$\mathcal{L}$	Lineare Hülle
$\|v\|$	euklidische Norm
$\langle v, w\rangle$	Skalarprodukt
$\mathrm{diag}(\lambda_1, \dots, \lambda_n)$	Diagonalmatrix
$(A)_{ij}$	(i,j)-tes Element von A
$\mathbf{e}_1$	kanonischer Einheitsvektor
$\boldsymbol{I}_n$	Einheitsmatrix
$x \ll y$	viel kleiner
$\mathrm{rg}(X)$	Rang
$\mathrm{df}(X)$	Defekt
$\mathrm{Im}(X)$	Bild
$\ker(X)$	Kern
$\mathrm{proj}_u(x)$	orthogonale Projektion von x auf u

Abkürzungen

SVD	Singular value decomposition (Singulärwertzerlegung)
PCA	Principal component analysis (Hauptkomponentenanalyse)
u.d.B.	unter den Bedingungen

1 Einleitung

> The Singular Value Decomposition is a highlight of linear algebra.[1]
>
> *(Gilbert Strang)*

In den Jahren 2006 bis 2009 zog ein Wettbewerb, ausgerufen von der Streaming-Plattform Netflix, große mediale Aufmerksamkeit auf sich. Es wurde ein Preisgeld von einer Million US-Dollar ausgerufen für das Team, welches den vorhanden Empfehlungsalgorithmus von Netflix zur Vorhersage von Nutzerbewertungen signifikant verbessern konnte. Die vielversprechendsten Ansätze basierten allesamt auf dem Prinzip der *Matrixfaktorisierung*, also der Zerlegung einer Matrix in mehrere andere Matrizen [KBV09]. Eine besondere Art der Matrixfaktorisierung bildet die Singulärwertzerlegung (engl. *Singular Value Decomposition*, SVD), da sie im Gegensatz zu vielen anderen Matrixzerlegungen auf jede beliebige Matrix angewendet werden kann und dabei ebenso spezielle wie nützliche Eigenschaften mit sich bringt. Dank dieser Eigenschaften bildete sie eine zentrale Grundlage der leistungsfähigsten Modelle innerhalb des Netflix-Wettbewerbs.

Der Verwendungszweck der Singulärwertzerlegung beschränkt sich allerdings nicht nur auf Empfehlungssysteme. Ein weiteres populäres Anwendungsgebiet findet sich in der *Data Science*, einem rapide wachsenden Bereich, in dem große Datenmengen sinnvoll verarbeitet werden müssen. Andere Einsatzmöglichkeiten sind die Bildverarbeitung, Molekularbiologie, Robotik oder sogar hochspezielle Themen wie die Kristallisierungsraten von Gesteinen [MP12].

Das Ziel dieser Arbeit ist herauszuarbeiten, was die Singulärwertzerlegung so besonders macht und wie dieses zunächst sehr theoretisch wirkende Konzept Anwendung in der Praxis findet. Dafür werden zunächst die wesentlichen mathematischen Grundlagen gelegt und die oben erwähnten speziellen Eigenschaften

[1] Zitiert nach [Str09, S. 363].

der SVD hergeleitet. Anschließend betrachten wir zwei verbreitete Anwendungen der Singulärwertzerlegung ausführlicher: die *Hauptkomponentenanalyse*, ein statistisches Verfahren der Data Science, welches komplexe Daten auf ihr Wesentliches reduziert sowie die bereits angesprochenen Empfehlungssysteme, bei denen unter anderem untersucht wird, wie eine Matrixzerlegung überhaupt mit Nutzer-Film-Bewertungen zusammenhängt. Den Abschluss des Hauptteils bildet die Implementierung eines eigenen Empfehlungssystems in `Python`, basierend auf realen Nutzerbewertungen aus einer Online-Datenbank.

Es sei darauf hingewiesen, dass grundlegende Kenntnisse in `Python` zum Verständnis der Implementierung vorausgesetzt werden. Innerhalb des Textes werden zentrale Passagen des Codes erläutert, während der vollständige Programmcode — mit Kommentaren an den nicht besprochenen Stellen — jeweils mit Verweis im Anhang zu finden ist.

2 Mathematische Theorie

In diesem Kapitel wird zunächst unter Verwendung vorher eingeführter Sätze und Definitionen die Existenz der Singulärwertzerlegung bewiesen. Anschließend wird an einem konkreten Beispiel die Berechnung durchgeführt und die SVD visualisiert. Um das Kapitel abzuschließen, erfolgt eine Beweisführung von zwei wichtigen Eigenschaften der SVD, die auch im Anwendungsteil erneut aufgegriffen werden.

2.1. Existenzbeweis der SVD

Es wird davon ausgegangen, dass der Leser[1] mit den Grundlagen der linearen Algebra vertraut ist, insbesondere mit Matrizen und ihren Eigenschaften. Bekannte Definitionen werden nicht erneut aufgeführt, die einzige Ausnahme bildet Definition 2.1, da diese für jegliche Beweisführung und für das Verständnis in diesem Kapitel unerlässlich ist und deswegen eine Auffrischung sinnvoll erscheint.

Definition 2.1.
Sei $n \in \mathbb{N}$ und $A \in \mathbb{R}^{n \times n}$. Für

$$Av = \lambda v$$

heißen die Lösungen $\mathbb{R}^n \ni v \neq \mathbf{0}$ *Eigenvektoren* und die zugehörigen λ *Eigenwerte*.

Vorausgesetzte Sätze werden ohne weiteren Beweis verwendet, wobei die entsprechenden Beweise aus [Räs21] zu entnehmen sind. Vor der Verwendung werden die Sätze kurz rekapituliert, wie an Wiederholung 2.2 verdeutlicht wird.

[1] Aus Gründen der besseren Lesbarkeit wird das generische Maskulinum verwendet, wobei alle Geschlechter mit eingeschlossen sind.

WIEDERHOLUNG 2.2 (Basisergänzungssatz).
Sei V ein beliebiger Vektorraum, $L \subseteq V$ linear unabhängig und $E \subseteq V$ ein Erzeugendensystem von V. Dann kann L durch Elemente aus E zu einer Basis von V ergänzt werden.

Um die Existenz der Singulärwertzerlegung für beliebige Matrizen zu beweisen, bedarf es der Hilfe eines anderen Satzes, des sogenannten Spektralsatzes. Dieser hat keine eindeutige Formulierung, sondern beschreibt vielmehr mehrere verwandte Aussagen der Mathematik, wobei sich in dieser Arbeit auf seine Folgerungen für symmetrische Matrizen beschränkt wird. Es ist ebenfalls wichtig zu betonen, dass der Spektralsatz zwar hier „nur" für den Beweis der Singulärwertzerlegung verwendet wird, eine Bezeichnung als Hilfssatz jedoch irreführend wäre, da der Satz für sich genommen bereits eine bedeutende Aussage der linearen Algebra und Funktionalanalysis darstellt. Um den Spektralsatz für symmetrische Matrizen einführen und anschließend beweisen zu können, benötigen wir zunächst Lemmata 2.3 und 2.4 sowie Wiederholung 2.5.

LEMMA 2.3.
Sei $z \in \mathbb{C}$ und $a, b \in \mathbb{R}$ mit $z = a + bi$. Es gilt $z = \overline{z}$ genau dann, wenn $z \in \mathbb{R}$.

Beweis. „⇒" Durch $z = \overline{z}$ gilt

$$\begin{aligned} & a + bi = a - bi \\ \Leftrightarrow \quad & 2bi = 0. \end{aligned}$$

Da $i \neq 0$ muss $b = 0$, womit $\mathfrak{Im}(z) = 0$. Also ist $z = \mathfrak{Re}(z) \in \mathbb{R}$.
„⇐" Folgt direkt aus der Definition der komplexen Zahlen. □

LEMMA 2.4.
Sei $n \in \mathbb{N}$, $A \in \mathbb{R}^{n \times n}$ und $\lambda \in \mathbb{R}$ ein reeller Eigenwert von A.
Dann gibt es einen zugehörigen Eigenvektor $\mathbb{R}^n \ni v \neq 0$, der ebenfalls reell ist.

Beweis. Angenommen $\mathbb{C}^n \ni v \neq 0$ mit $v = x + iy$ für $x, y \in \mathbb{R}^n$. Dann gilt nach Definition 2.1

$$\begin{aligned} & Av = \lambda v \\ \Leftrightarrow\ & A(x + yi) = \lambda(x + yi) \end{aligned}$$

$$\Leftrightarrow Ax + Ayi = \lambda x + \lambda yi.$$

Durch einen Koeffizientenvergleich erhalten wir damit

$$Ax = \lambda x \quad \wedge \quad Ay = \lambda y.$$

Da $v \neq 0$ ist, muss entweder $x \neq 0$ oder $y \neq 0$ sein, womit nach Definition 2.1 mindestens ein reeller Eigenvektor zu λ existiert. □

Wiederholung 2.5 (Gram-Schmidtsches Orthonormalisierungsverfahren).
Sei V ein euklidischer Vektorraum und $\{u_1, \dots, u_n\}$ eine Menge von linear unabhängigen Vektoren in V. Dann kann eine Menge $\{v_1, \dots, v_n\}$ aus Vektoren in V konstruiert werden, sodass $\{v_1, \dots, v_n\}$ orthonormal ist und

$$\mathcal{L}\{v_1, \dots, v_n\} = \mathcal{L}\{u_1, \dots, u_n\}.$$

Satz 2.6 (Spektralsatz).
Sei $n \in \mathbb{N}$ und $A \in \mathbb{R}^{n \times n}$ quadratisch und symmetrisch. Dann gilt:

(i) A hat nur reelle Eigenwerte.

(ii) Es existiert eine orthogonale Matrix $R \in \mathbb{R}^{n \times n}$, sodass $R^{-1}AR = R^TAR = \Lambda \in \mathbb{R}^{n \times n}$ diagonal ist.

Beweis. Die Behauptungen werden nacheinander bewiesen [Cra22].

Zu (i). Sei $\lambda \in \mathbb{C}$ ein Eigenwert von A mit zugehörigem Eigenvektor $\mathbb{C}^n \ni v \neq 0$. Dann ist mit Definition 2.1

$$\begin{aligned} & Av = \lambda v \\ \Leftrightarrow \quad & A\overline{v} = \overline{\lambda}\overline{v}, \end{aligned} \tag{2.1}$$

da A reell ist und somit $A = \overline{A}$ nach Lemma 2.3. Nun gilt zum einen

$$(Av)^T\overline{v} = \left(\lambda v\right)^T\overline{v} = \lambda v^T\overline{v} \tag{2.2}$$

und zum anderen

$$(Av)^T\overline{v} = v^TA^T\overline{v} \overset{A \text{ sym.}}{=} v^TA\overline{v} \overset{(2.1)}{=} v^T\overline{\lambda}\overline{v} = \overline{\lambda}v^T\overline{v}. \tag{2.3}$$

Mit (2.2)=(2.3) ergibt sich

$$\lambda v^T \overline{v} = \overline{\lambda} v^T \overline{v}.$$

Da $v \neq 0$ ist, erhalten wir

$$\lambda = \overline{\lambda}.$$

Nach Lemma 2.3 ist dann $\lambda \in \mathbb{R}$. □

Zu (ii). Induktion über $n \in \mathbb{N}$:
Induktionsanfang. Für $n = 1$ sind A und R Skalare. Setze $R = 1$. Damit ist R orthogonal, da $R^{-1} = R^T$ und $\mathbb{R} \ni A = R^{-1}AR$ trivialerweise diagonal.
Induktionshypothese. Die Behauptung (ii) gelte für festes, beliebiges $n - 1 \in \mathbb{N}$. Es soll gezeigt werden, dass sie dann auch für n gilt.
Induktionsschritt. Sei λ_1 ein beliebiger Eigenwert von A mit zugehörigem normiertem Eigenvektor v_1, also $\|v_1\| = 1$. Nach (i) gilt $\lambda_1 \in \mathbb{R}$ und damit nach Lemma 2.4 auch $v_1 \in \mathbb{R}^n$. Mit dem Basisergänzungssatz (Wiederholung 2.2) kann v_1 durch Vektoren $u_2, \dots, u_n$ zu einer Basis von $\mathbb{R}^n$ ergänzt werden. Nun kann das Gram-Schmidt'sche Orthonormalisierungsverfahren (Wiederholung 2.5) angewendet werden, wodurch eine orthonormale Basis $\{v_1, \dots, v_n\}$ von $\mathbb{R}^n$ konstruiert wird. Der Leser wird daran erinnert, dass orthonormale Vektoren normiert und orthogonal sind. Sei

$$P = \begin{bmatrix} | & | & | & | \\ v_1 & v_2 & \cdots & v_n \\ | & | & | & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

mit $v_1, \dots, v_n$ als Spaltenvektoren und setze $B = P^{-1}AP = P^TAP \in \mathbb{R}^{n \times n}$.

Das Ziel ist, die Induktionshypothese auf eine symmetrische Untermatrix $C \in \mathbb{R}^{(n-1)\times(n-1)}$ von B anzuwenden. Dafür wird zunächst die Symmetrie von B gezeigt:

$$B^T = \left(P^TAP\right)^T = (AP)^TP = P^TA^TP \overset{A \text{ sym.}}{=} P^TAP = B.$$

Betrachte jetzt die erste Spalte von B. Die erste Spalte einer beliebigen Matrix erhält man durch Multiplikation mit dem kanonischen Einheitsvektor $\mathbf{e}_1$:

$$\begin{aligned} B\mathbf{e}_1 &= P^TAP\mathbf{e}_1 \\ &= P^TAv_1 && (v_1 \text{ ist die erste Spalte von } P) \\ &= P^T\lambda_1 v_1 && (\lambda_1 \text{ ist der Eigenwert zu } v_1) \\ &= P^Tv_1\lambda_1 \end{aligned}$$

$$
\begin{aligned}
&= \begin{bmatrix} — \, v_1 \, — \\ — \, v_2 \, — \\ \vdots \\ — \, v_n \, — \end{bmatrix} v_1 \lambda_1 \\
&= \begin{bmatrix} \langle v_1, v_1 \rangle \\ \langle v_2, v_1 \rangle \\ \dots \\ \langle v_n, v_1 \rangle \end{bmatrix} \lambda_1 \\
&= \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \lambda_1. \qquad \big(\|v_1\| = 1 \text{ und bel. } v_i, v_j \in \{v_1, \dots, v_n\} \text{ orthogonal} \big)
\end{aligned}
$$

Mit der Darstellung als Blockmatrix und durch Symmetrie von B folgt somit $B = \begin{bmatrix} \lambda_1 & \mathbf{0} \\ \mathbf{0} & C \end{bmatrix}$ mit $C \in \mathbb{R}^{(n-1)\times(n-1)}$ symmetrisch.

Nach Induktionshypothese gibt es damit ein orthogonales $Q \in \mathbb{R}^{(n-1)\times(n-1)}$ mit $Q^T C Q = D$ diagonal. Dann gilt

$$
\begin{aligned}
P^T A P &= B \\
&= \begin{bmatrix} \lambda_1 & \mathbf{0} \\ \mathbf{0} & C \end{bmatrix} \\
&= \begin{bmatrix} \lambda_1 & \mathbf{0} \\ \mathbf{0} & Q D Q^T \end{bmatrix} \\
&= \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q \end{bmatrix} \begin{bmatrix} \lambda_1 & \mathbf{0} \\ \mathbf{0} & D \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q^T \end{bmatrix}.
\end{aligned}
$$

Also ist

$$
\begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q^T \end{bmatrix} P^T A P \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q \end{bmatrix} = \begin{bmatrix} \lambda_1 & \mathbf{0} \\ \mathbf{0} & D \end{bmatrix}.
$$

Definiere

$$
R = P \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q \end{bmatrix}.
$$

Es gilt

$$
R^T = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q^T \end{bmatrix} P^T = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & Q^{-1} \end{bmatrix} P^{-1} = R^{-1}.
$$

Dementsprechend ist R orthogonal und $R^{-1}AR = R^TAR = \begin{bmatrix} \lambda_1 & \mathbf{0} \\ \mathbf{0} & D \end{bmatrix} = \Lambda$ diagonal, da D diagonal ist. □

KOROLLAR 2.7.
Seien die Voraussetzungen aus Satz 2.6 gegeben.
Dann gilt, dass die Diagonalwerte von $\Lambda \in \mathbb{R}^{n\times n}$ die Eigenwerte der Matrix A und die Spalten von R die zugehörigen normierten Eigenvektoren von A sind.

Beweis. Nach dem Spektralsatz gibt es ein orthogonales $R = [r_1 \dots r_n]$ mit $r_1, \dots, r_n \in \mathbb{R}^n$ und $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$, sodass

$$R^{-1}AR = \Lambda.$$

Dann ist

$$AR = R\Lambda$$

oder spaltenweise

$$Ar_i = \lambda_i r_i, \quad \text{für } i = 1, \dots, n.$$

Da R orthogonal ist, sind die Spaltenvektoren $r_1, \dots, r_n$ von R der Definition nach orthonormal. Für beliebiges r_i gilt somit $\|r_i\| = 1$, wodurch $r_i \neq \mathbf{0}$ sein muss. Nach Definition 2.1 sind also $r_1, \dots, r_n$ die (normierten) Eigenvektoren von A mit zugehörigen Eigenwerten $\lambda_1, \dots, \lambda_n$.

Hinweis. Mithilfe von Zeilen- und Spaltenvertauschungen innerhalb von R und Λ kann Λ so geordnet werden, dass $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$. Diese Sortierung wird für den Rest der Arbeit angenommen. □

BEMERKUNG 2.8.
Durch Korollar 2.7 lässt sich direkt die nützliche Aussage treffen, dass bei einer symmetrischen Matrix Eigenvektoren zu verschiedenen Eigenwerten orthogonal zueinander sind.

Für den Existenzbeweis der Singulärwertzerlegung werden zusätzlich zum Spektralsatz noch Wiederholung 2.9, Lemma 2.10 und Wiederholung 2.11 benötigt. Der Beweis von Lemma 2.10 erfolgt nach [Che13].

Wiederholung 2.9 (Dimensionssatz).
Seien $m, n \in \mathbb{N}$ und $A \in \mathbb{R}^{m \times n}$. Dann gilt:

$$\mathrm{df}(A) + \mathrm{rg}(A) = n.$$

Lemma 2.10.
Seien $m, n \in \mathbb{N}$ und $X \in \mathbb{R}^{m \times n}$. Dann ist $\mathrm{rg}(X) = \mathrm{rg}(X^T X)$.

Beweis. Sei $v \in \ker(X)$ beliebig. Dann ist

$$\begin{aligned} & Xv = \mathbf{0} \\ \Rightarrow \quad & X^T X v = \mathbf{0}, \end{aligned}$$

womit $v \in \ker(X^T X)$. Gleichzeitig gilt für $v \in \ker(X^T X)$

$$\begin{aligned} & X^T X v = \mathbf{0} \\ \Rightarrow \quad & v^T X^T X v = \mathbf{0} \\ \Leftrightarrow \quad & \|Xv\|^2 = \mathbf{0} \\ \Leftrightarrow \quad & Xv = \mathbf{0}, \end{aligned}$$

also $v \in \ker(X)$. Damit ist $\ker(X) = \ker(X^T X)$ und es folgt $\mathrm{rg}(X) = \mathrm{rg}(X^T X)$ nach dem Dimensionssatz (Wiederholung 2.9). □

Wiederholung 2.11.
Der Rang einer reell symmetrischen Matrix ist die Anzahl der von null verschiedenen Eigenwerten.

Nun kann mithilfe der vorangegangenen Aussagen die Existenz der Singulärwertzerlegung für beliebige reelle Matrizen bewiesen werden. Der Beweis orientiert sich dabei an [Che20].

Satz 2.12 (Singulärwertzerlegung).
Seien $m, n \in \mathbb{N}$ und $X \in \mathbb{R}^{m \times n}$.
Dann existieren orthogonale Matrizen $U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}$ und eine Matrix $\Sigma \in \mathbb{R}^{m \times n}$, sodass

$$X = U \Sigma V^T.$$

Beweis. Sei $C = X^T X \in \mathbb{R}^{n\times n}$ und $r = \operatorname{rg}(X) \leq \min(m,n)$. Dann ist C symmetrisch und positiv semidefinit, da zum einen

$$C^T = \left(X^T X\right)^T = X^T X = C$$

und zum anderen für beliebiges $\mathbb{R}^n \ni w \neq 0$ gilt:

$$w^T C w = w^T (X^T X) w = \left(Xw\right)^T \left(Xw\right) = \langle Xw, Xw\rangle = \|Xw\|^2 \geq 0.$$

Damit sind alle Eigenwerte von C positiv oder gleich null. Nach Korollar 2.7 gibt es durch die Symmetrie ein orthogonales

$$\mathbf{V} = \left[v_1 \dots v_n\right] \in \mathbb{R}^{n\times n}$$

und diagonales $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_n) \in \mathbb{R}^{n\times n}$ mit $\lambda_1 \geq \dots \geq \lambda_r > 0 \overset{2}{=} \lambda_{r+1} = \dots = \lambda_n$, sodass $C = V\Lambda V^T$. Sei $\sigma_i = \sqrt{\lambda_i}$ für $i \in \{1, \dots, r\}$ und

$$\mathbf{\Sigma} = \begin{bmatrix} \operatorname{diag}(\sigma_1, \dots, \sigma_r) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{m\times n}$$

Sei außerdem

$$u_i = \frac{1}{\sigma_i} X v_i \in \mathbb{R}^m.$$

Dann sind $u_1, \dots, u_r$ orthornomal für $j \in \{1, \dots, r\}$:

$$\begin{aligned}
u_i^T u_j &= \left(\frac{1}{\sigma_i} X v_i\right)^T \left(\frac{1}{\sigma_j} X v_j\right) \\
&= \frac{1}{\sigma_i \sigma_j} v_i^T \underbrace{X^T X}_{=C} v_j \\
&= \frac{1}{\sigma_i \sigma_j} v_i^T (\lambda_j v_j) && \left(\lambda_j \text{ ist der Eigenwert zu } v_j\right) \\
&= \frac{\sigma_j}{\sigma_i} v_i^T v_j && \left(\lambda_j = \sigma_j^2\right) \\
&= \begin{cases} 1, & i = j. \\ 0, & i \neq j. \end{cases} && \left(v_i, v_j \text{ sind orthonormal}\right)
\end{aligned}$$

Wie bereits im Beweis des Spektralsatzes können $u_1, \dots, u_r$ mithilfe des Basisergänzungssatzes (Wiederholung 2.2) und des Gram-Schmidt-Verfahrens (Wiederholung 2.5) durch Vektoren $u_{r+1} \dots, u_m \in \mathbb{R}^m$ zu einer orthonormalen

[2] Nach Lemma 2.10 und Wiederholung 2.11 sowie der positiven Semidefinitheit von $X^T X$.

Basis von $\mathbb{R}^m$ ergänzt werden. Damit ist

$$\mathbf{U} = \begin{bmatrix} u_1 \dots u_r u_{r+1} \dots u_m \end{bmatrix} \in \mathbb{R}^{m \times m}$$

orthogonal. Es bleibt zu zeigen, dass $XV = U\Sigma$ ist, also:

$$X\begin{bmatrix} v_1 \dots v_r v_{r+1} \dots v_n \end{bmatrix} = \begin{bmatrix} u_1 \dots u_r u_{r+1} \dots u_m \end{bmatrix} \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_r \\ \hline & \mathbf{0} & & \mathbf{0} \end{array}\right].$$

Für $1 \leq i \leq r$ gilt $Xv_i = u_i\sigma_i$ nach Konstruktion.
Für $i > r$ soll gezeigt werden, dass $Xv_i = 0u_i = \mathbf{0}$ ist. Betrachte dafür

$$X^T X v_i = C v_i \overset{(*)}{=} 0 v_i = \mathbf{0}$$

$(*)$ *Der zugehörige Eigenwert zum Eigenvektor v_i ist 0 für $i > r$.*

Damit muss wie erwünscht $Xv_i = \mathbf{0}$ gelten, oder $X^T = \mathbf{0}$, wodurch ebenfalls $Xv_i = \mathbf{0}$ folgt. Dementsprechend ist $X = U\Sigma V^T$ und die Aussage ist bewiesen. □

Bemerkung 2.13.

- Die Diagonalwerte von Σ heißen **Singulärwerte** von X und werden in der Regel absteigend sortiert.
- Die Spalten von U heißen **linke Singulärvektoren** von X.
- Die Spalten von V heißen **rechte Singulärvektoren** von X.

Der zentrale Beweis dieses Kapitels ist somit abgeschlossen und die Berechnung der Singulärwertzerlegung kann an einem Beispiel veranschaulicht und visualisiert werden.

2.2. Beispiel und Visualisierung

Beispiel 2.14. Sei

$$A = \begin{bmatrix} 1 & -1 & 3 \\ 3 & 1 & 1 \end{bmatrix} \in \mathbb{R}^{2 \times 3}.$$

Um die SVD dieser Matrix zu finden, muss der Beweis von Satz 2.12 mithilfe unserer konkreten Werte schrittweise nachvollzogen werden. Zuerst wird also

$$A^T A = \begin{bmatrix} 10 & 2 & 6 \\ 2 & 2 & -2 \\ 6 & -2 & 10 \end{bmatrix} \in \mathbb{R}^{3\times 3}$$

bestimmt. Davon sollen die Eigenwerte mit zugehörigen normierten Eigenvektoren berechnet werden. Für die Eigenwerte muss zunächst

$$\det(A^T A - \lambda \boldsymbol{I}_3) = 0$$

gesetzt und die Lösungen λ_i für $i = 1, 2, 3$ gefunden werden. Auf die genaue Berechnung wird an dieser Stelle verzichtet, das Ergebnis lautet:

$$\lambda_1 = 16, \quad \lambda_2 = 6, \quad \lambda_3 = 0.$$

Mit $\sigma_j = \sqrt{\lambda_j}$ für $j = 1, 2$ (da $\operatorname{rg}(A) = 2$) erhalten wir

$$\Sigma = \begin{bmatrix} 4 & 0 & 0 \\ 0 & \sqrt{6} & 0 \end{bmatrix} \in \mathbb{R}^{2\times 3}.$$

Durch die Lösungen $\mathbb{R}^3 \ni v_i \neq 0$ von

$$(A^T A - \lambda \boldsymbol{I}_3)v = \boldsymbol{0}$$

ergeben sich die normierten Eigenvektoren

$$v_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix}, \quad v_2 = \begin{bmatrix} -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{bmatrix}, \quad v_3 = \begin{bmatrix} -\frac{1}{\sqrt{6}} \\ \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{bmatrix}$$

und damit

$$V^T = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \end{bmatrix} \in \mathbb{R}^{3\times 3}.$$

Es muss nur noch $U \in \mathbb{R}^{2\times 2}$ bestimmt werden, welches spaltenweise durch

$$u_j = \frac{1}{\sigma_j} X v_j$$

ausgedrückt wird. Wir erhalten also

$$u_1 = \frac{1}{4}\begin{bmatrix} 1 & -1 & 3 \\ 3 & 1 & 1 \end{bmatrix}\begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

und

$$u_2 = \frac{1}{\sqrt{6}}\begin{bmatrix} 1 & -1 & 3 \\ 3 & 1 & 1 \end{bmatrix}\begin{bmatrix} -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}.$$

Dadurch ist

$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

und die Berechnung ist abgeschlossen mit

$$A = \begin{bmatrix} 1 & -1 & 3 \\ 3 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}\begin{bmatrix} 4 & 0 & 0 \\ 0 & \sqrt{6} & 0 \end{bmatrix}\begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \end{bmatrix} = U\Sigma V^T.$$

Damit genauer verstanden wird, was genau durch die Singulärwertzerlegung geschieht, betrachten wir die Wirkung der Matrix A aus Beispiel 2.14 auf einen Vektor $v \in \mathbb{R}^3$, indem wir v und Av grafisch darstellen. Da dies an einem einzelnen Vektor schwer visualisiert werden kann, multiplizieren wir A mit allen Punkten auf der Einheitssphäre, also allen

$$v \in \left\{ \begin{bmatrix} \cos(x)\sin(y) \\ \sin(x)\sin(y) \\ \cos(y) \end{bmatrix} \in \mathbb{R}^3 \mid x \in [0, 2\pi],\ y \in [0, \pi] \right\}.$$

Das Ergebnis ist in Abbildung A.1, zu finden in Anhang A, dargestellt.

Um nachzuvollziehen, wie dieses Ergebnis zustande gekommen ist, verwenden wir die Singulärwertzerlegung und veranschaulichen die Zwischenschritte von $Av = U\Sigma V^T v$ anhand von einzelner Plots (siehe Abbildung 2.1).

Die Grundlage für diese Visualisierung bildet das Wissen, dass im euklidischen Raum orthogonale Matrizen Drehungen und Diagonalmatrizen Skalierungen (entlang der Hauptachsen) entsprechen. An dieser Stelle sei angemerkt, dass

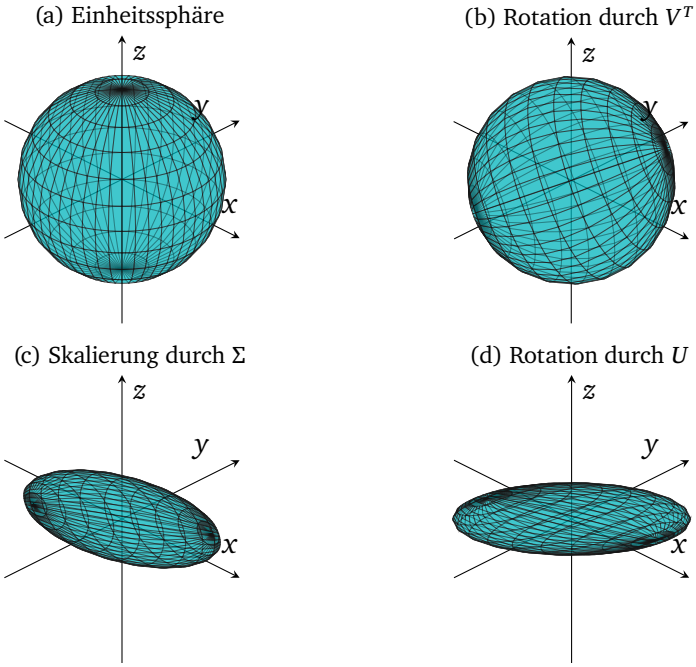


Abb. 2.1. Visualisierung der Singulärwertzerlegung

in Abbildung 2.1c und Abbildung 2.1d der zu beobachtende Wertebereich vergrößert wurde, also eine größere Streckung durch Σ erfolgt ist, als auf dem Plot zu sehen ist. Außerdem ist ab Abbildung 2.1c die Darstellung der z-Achse überflüssig und eigentlich nicht zu empfehlen, da sich dort durch die Dimensionsreduktion mittels Σ im zweidimensionalen Raum bewegt wird. Zur besseren Vergleichbarkeit wird die Darstellung dennoch beibehalten.

Zusammenfassend zerlegt also die Singulärwertzerlegung eine Matrix in grundlegende geometrische Transformationen: Drehung, Skalierung und gegebenenfalls Dimensionsreduktion oder -erhöhung. Mit dieser Erkenntnis wird sich dem nächsten Abschnitt zugewandt, in dem die wichtigsten beiden Arten der SVD definiert werden.

2.3. Arten der SVD

Die Singulärwertzerlegung, die im vorherigen Teil der Arbeit beschrieben wurde, ist die klassische und vollständige Zerlegung. In den tatsächlichen Anwendungsgebieten, welche im nächsten Kapitel ausgeführt werden, finden häufig

Variationen Verwendung.

Definition 2.15.
Seien $m, n \in \mathbb{N}$, $A \in \mathbb{R}^{m\times n}$, $\mathrm{rg}(A) = r$ und $A = U\Sigma V^T$ die vollständige SVD von A mit $U \in R^{m\times m}$, $\Sigma \in \mathbb{R}^{m\times n}$ und $V \in \mathbb{R}^{n\times n}$.
Definiere die *reduzierte SVD* von A:

$$A = U_r \Sigma_r V_r^T$$

mit

$$\begin{aligned} U_r &= [u_1 \dots u_r] \in \mathbb{R}^{m\times r}, \\ \Sigma_r &= \mathrm{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{r\times r}, \\ V_r &= [v_1 \dots v_r] \in \mathbb{R}^{n\times r}. \end{aligned}$$

Die in Definition 2.15 beschriebene Art der Singulärwertzerlegung besitzt den Vorteil, dass eine exakte Berechnung mit deutlich weniger Speicherbedarf möglich ist, insbesondere bei großen Matrizen $X \in \mathbb{R}^{m\times n}$ mit $\mathrm{rg}(X) \ll \min(m, n)$.

Bevor zur nächsten Variation übergegangen werden kann, betrachten wir eine weitere Darstellungsform von Definition 2.15.

Bemerkung 2.16.
Sei

$$A = [u_1 \dots u_r] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_r^T \end{bmatrix}$$

wie in Definition 2.15.
Dann ist $(A)_{11} = u_{1,1}\sigma_1 v_{1,1}^T + u_{2,1}\sigma_2 v_{2,1}^T + \cdots + u_{r,1}\sigma_r v_{r,1}^T$.
Für die andere Komponenten von A kann die Summe analog gebildet werden. Für $x \in \{1, \dots, m\}$ und $y \in \{1, \dots, n\}$ erhalten wir also

$$\begin{aligned} (A)_{xy} &= \sum_{k=1}^{r} \sigma_k u_{k,x} v_{k,y}^T \\ \Rightarrow \quad A \quad &= \sum_{i=1}^{r} \sigma_i u_i v_i^T. \end{aligned}$$

Damit kann A als *Summe von Rang-1-Matrizen* dargestellt werden, da für $i \in \{1, \dots, r\}$ die Zeilen von $(\sigma_i u_i v_i^T) \in \mathbb{R}^{m\times n}$ ein Vielfaches von v_i^T und

die Spalten ein Vielfaches von u_i sind.
Beachte, dass $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ und $\|u_i\| = 1 = \|v_i\|$, also gilt komponentenweise: $\sigma_1 u_1 v_1^T \geq \sigma_2 u_2 v_2^T \geq \cdots \geq \sigma_r u_r v_r^T$.

Nun wird die womöglich interessanteste Art der Singulärwertzerlegung für diverse Anwendungsfälle definiert.

DEFINITION 2.17.
Seien $m, n \in \mathbb{N}$, $A \in \mathbb{R}^{m \times n}$, $\operatorname{rg}(A) = r$ und $A = U_r \Sigma_r V_r^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$ die reduzierte SVD von A.
Für $k \leq r$ ist die *trunkierte SVD* definiert durch:

$$A_k := \sum_{i=1}^{k} \sigma_i u_i v_i^T \approx A.$$

Zur Definition 2.17 sei angemerkt, dass $\operatorname{rg}(A_k) = k \leq r$ ist, aufgrund dessen spricht man auch von einer *Niedrigrang-Approximation*. Außerdem sollte betont werden, dass sich A_k für größere Werte von k zunehmend A annähert. Da jedoch komponentenweise $\sigma_1 u_1 v_1^T \geq \cdots \geq \sigma_r u_r v_r^T$ gilt (siehe Bemerkung 2.16), kann bereits mit $k \ll r$ eine gute Approximation erzielt werden. Eine Präzisierung dieser Aussage findet im nachfolgenden und letzten Abschnitt des theoretischen Teils statt.

2.4. EIGENSCHAFTEN DER SVD

Die trunkierte Singulärwertzerlegung A_k ist nicht nur eine gute Approximation von A, sondern die beste Rang-k-Approximation für A bezüglich der Spektralnorm und der Frobeniusnorm. Diese Behauptung ist auch als *Eckart-Young-Satz* bekannt und wird im Folgenden nach [Zha22, S. 7] für die Spektralnorm bewiesen. Für den Beweis werden allerdings zunächst Lemmata 2.18 und 2.19 benötigt.

LEMMA 2.18.
Seien $m, n \in \mathbb{N}$ und $V \in \mathbb{R}^{m \times n}$ orthogonal.
Dann gilt für $x \in \mathbb{R}^n$

$$\|Vx\| = \|x\|.$$

Beweis.

$$\|Vx\| = \sqrt{\langle Vx, Vx\rangle} = \sqrt{x^T V^T V x} = \sqrt{x^T x} = \|x\|.$$

□

Lemma 2.19.
Seien $m, n \in \mathbb{N}$ und $A \in \mathbb{R}^{m\times n}$.
Dann gilt für die Spektralnorm

$$\|A\|_2 := \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\| = \sigma_1,$$

wobei σ_1 den größten Singulärwert von A beschreibt.

Beweis. Der Beweis orientiert sich an [Fra11, S. 7].
Sei $A = U_r \Sigma_r V_r^T$ die reduzierte SVD von A für $\operatorname{rg}(A) = r$. Dann gilt für $x \in \mathbb{R}^n$ mit $\|x\| = 1$

$$\begin{aligned}
\|Ax\| &= \|U_r \Sigma_r V_r^T x\| \\
&= \|\Sigma_r V_r^T x\| && \text{(Lemma 2.18)} \\
&= \|\Sigma_r y\| && \text{(für } y := V_r^T x\text{)} \\
&= \sqrt{\sigma_1^2 y_1^2 + \cdots + \sigma_r^2 y_r^2}.
\end{aligned}$$

Nach Lemma 2.18 ist $\|y\| = \|x\| = 1$ und da $\sigma_1 \geq \cdots \geq \sigma_r$ wird der Ausdruck maximiert für $y = \mathbf{e}_1$. Damit ist $\|A\|_2 = \sigma_1$. □

Satz 2.20 (Eckart-Young).
Seien $m, n \in \mathbb{N}$ und $A, B \in \mathbb{R}^{m\times n}$ mit $\operatorname{rg}(B) = k \leq r = \operatorname{rg}(A)$. Sei außerdem $A_k \in \mathbb{R}^{m\times n}$ die trunkierte SVD von A nach Definition 2.17.
Dann gilt

$$\|A - B\|_2 \geq \|A - A_k\|_2.$$

Beweis. Für $k = r$ ist $A_k = A$ und

$$\|A - B\|_2 \geq 0$$

gilt nach Definition der Norm.

Sei also $k < r$. Wir zeigen zunächst, dass $\|A - A_k\|_2 = \sigma_{k+1}$. Mit der redu-

zierten SVD von $A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$ gilt:

$$A - A_k = \sum_{i=1}^{r} \sigma_i u_i v_i^T - \sum_{i=1}^{k} \sigma_i u_i v_i^T = \sum_{i=k+1}^{r} \sigma_i u_i v_i^T = \sum_{i=1}^{r-k} \sigma_{i+k} u_{i+k} v_{i+k}^T.$$

Die letzte Summe bildet dann die reduzierte SVD von $A - A_k$, wodurch nach Lemma 2.19:

$$\|A - A_k\|_2 = \sigma_{k+1}.$$

Wir wollen nun einen Widerspruchsbeweis führen. Sei also angenommen, dass ein $B \in \mathbb{R}^{m \times n}$ existiert mit $\operatorname{rg}(B) = k$ und

$$\|A - B\|_2 < \|A - A_k\|_2 = \sigma_{k+1}.$$

Für beliebiges $w \in \mathbb{R}^n$ gilt

$$\|(A - B)w\| \leq \|A - B\| \|w\| < \sigma_{k+1} \|w\|.$$

Sei zusätzlich $w \in \ker(B)$. Dann ist

$$\|Aw\| = \|Aw - Bw\| = \|(A - B)w\| < \sigma_{k+1} \|w\|. \tag{2.4}$$

Wir definieren $V_{k+1} := [v_1 \dots v_{k+1}]$ für die ersten $k+1$ rechten Singulärvektoren von A.

Sei außerdem $w \in \operatorname{Im}(V_{k+1})$. Dann existieren $c_1, \dots, c_{k+1} \in \mathbb{R}$ mit $w = \sum_{i=1}^{k+1} c_i v_i$ und es gilt

$$\|w\|^2 = \|\sum_{i=1}^{k+1} c_i v_i\|^2 = \sum_{i=1}^{k+1} c_i v_i^T \sum_{j=1}^{k+1} c_j v_j \overset{(*)}{=} \sum_{i=1}^{k+1} c_i^2 \|v_i\| = \sum_{i=1}^{k+1} c_i^2. \tag{2.5}$$

(∗) *Da v_i, v_j orthogonal sind für $i \neq j$.*

Damit ist

$$\begin{aligned} \|Aw\|^2 = \|\sum_{i=1}^{k+1} c_i A v_i\|^2 = \|\sum_{i=1}^{k+1} c_i \sigma_i u_i\|^2 &= \sum_{i=1}^{k+1} c_i \sigma_i u_i^T \sum_{j=1}^{k+1} c_j \sigma_j u_j \\ &= \sum_{i=1}^{k+1} c_i^2 \sigma_i^2 \\ &\overset{(2.5)}{\geq} \sigma_{k+1}^2 \|w\|^2. \end{aligned} \tag{2.6}$$

Da (2.4) und (2.6) nicht gleichzeitig gelten können, bleibt es zu zeigen, dass

ein $w \in \ker(B) \cap \mathrm{Im}(V_{k+1})$ existiert. Wir betrachten dafür:

$$\begin{aligned}\dim\left[\ker(B) \cap \mathrm{Im}(V_{k+1})\right] &= \mathrm{df}(B) + \mathrm{rg}(V_{k+1}) - \dim\left[\ker(B) + \mathrm{Im}(V_{k+1})\right] \\ &\overset{(*)}{=} (n-k) + (k+1) - \dim\left[\ker(B) + \mathrm{Im}(V_{k+1})\right] \\ &\geq (n-k) + (k+1) - n \\ &= 1,\end{aligned}$$

(∗) *Siehe Wiederholung 2.9.*

da $\ker(B), \mathrm{Im}(V_{k+1}) \subseteq \mathbb{R}^n$, womit auch $\dim\left[\ker(B) + \mathrm{Im}(V_{k+1})\right] \leq n$ sein muss. Folglich existiert ein $w \neq \mathbf{0}$ mit den gewünschten Eigenschaften und der Widerspruch wurde gezeigt. □

Eine weitere wichtige Eigenschaft der Singulärwertzerlegung besteht in ihrer Verbindung zu den *vier fundamentalen Unterräumen*. Diese sind in ihren Definitionen zwar bereits bekannt, werden jedoch nach dem Vorbild von [Str09, S. 187] in der folgenden Form notiert:

Definition 2.21.
Seien $m, n \in \mathbb{N}$ und $A \in \mathbb{R}^{m \times n}$. Definiere folgende Unterräume zu A:

- *Spaltenraum:* $\mathrm{Im}(A) = \{b \in \mathbb{R}^m \mid \exists x \in \mathbb{R}^n, Ax = b\}$.
- *Zeilenraum:* $\mathrm{Im}(A^T) = \{z \in \mathbb{R}^n \mid \exists y \in \mathbb{R}^m, A^T y = z\}$.
- *Kern/Nullraum:* $\ker(A) = \{x \in \mathbb{R}^n \mid Ax = \mathbf{0}\}$.
- *Linkskern:* $\ker(A^T) = \{y \in \mathbb{R}^m \mid A^T y = \mathbf{0}\}$.

Die Unterräume geben umfangreichen Aufschluss über die Wirkung einer Matrix auf verschiedene Vektoren und stehen dabei in Verbindung mit zahlreichen Themen der linearen Algebra.

Die Art der Beziehung zwischen der Singulärwertzerlegung und den vier fundamentalen Unterräumen wird in Korollar 2.22 zusammengefasst, wobei der Beweis nach [Joh21, S. 214 f.] geführt wird.

Korollar 2.22.
Seien $m, n \in \mathbb{N}$, $X \in \mathbb{R}^{m \times n}$ und $r = rg(X)$.
Sei außerdem $X = U\Sigma V^T$ die vollständige Singulärwertzerlegung von X.
Dann gilt:

- Die ersten r Spalten von U sind eine Basis des Spaltenraums von X.

- Die letzten $m - r$ Spalten von U sind eine Basis des Linkskerns von X.
- Die ersten r Spalten von V sind eine Basis des Zeilenraums von X.
- Die letzten $n - r$ Spalten von V sind eine Basis des Kerns von X.

Beweis. Wir haben

$$U = \begin{bmatrix} u_1 \dots u_r u_{r+1} \dots u_m \end{bmatrix} \in \mathbb{R}^{m \times m},$$
$$V = \begin{bmatrix} v_1 \dots v_r v_{r+1} \dots v_n \end{bmatrix} \in \mathbb{R}^{n \times n}$$

und

$$\Sigma = \begin{bmatrix} \operatorname{diag}(\sigma_1, \dots, \sigma_r) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{m \times n},$$

sodass $X = U\Sigma V^T$. Sei $i \in \{1, \dots, n\}$ und betrachte

$$Xv_i = U\Sigma V^T v_i \overset{(*)}{=} U\Sigma \mathbf{e}_i = U\sigma_i \mathbf{e}_i = \sigma_i U \mathbf{e}_i = \sigma_i u_i.$$

(∗) *Für $j \in \{1, \dots, n\}$ sind v_i, v_j orthonormal.*

Fall 1: $1 \leq i \leq r$.
Damit ist $\sigma_i > 0$ und

$$X\frac{v_i}{\sigma_i} = u_i.$$

Nach Definition 2.21 sind dann $u_1, \dots, u_r \in \operatorname{Im}(X)$. Nun ist $\operatorname{rg}(X) = r$ und da $\mathcal{B}_S := \{u_1, \dots, u_r\}$ genau r orthonormale Vektoren enthält, bildet $\mathcal{B}_S$ eine Basis vom $\operatorname{Im}(X)$, also vom Spaltenraum.

Fall 2: $i \geq r + 1$.
Damit ist $\sigma_i = 0$ und

$$Xv_i = \mathbf{0}.$$

Nach Definition 2.21 sind dann $v_{r+1}, \dots, v_n \in \ker(X)$. Mithilfe von Wiederholung 2.9 wissen wir, dass $\operatorname{df}(X) = n - r$. Durch $\mathcal{B}_K := \{v_{r+1}, \dots, v_n\}$ haben wir $n - r$ orthonormale Vektoren gegeben, also bildet $\mathcal{B}_K$ eine Basis vom $\ker(X)$.

Die Beweise für die Basen des Linkskerns und des Zeilenraums werden analog gezeigt, indem

$$X^T u_i = V\Sigma U^T u_i$$

betrachtet wird. □

Es sind damit alle wesentlichen Grundlagen für die Vertiefung verschiedener Anwendungsmöglichkeiten der SVD gelegt. Der theoretische Teil der Arbeit wird hiermit abgeschlossen und es folgt nun die erste praktische Anwendung.

3 Hauptkomponentenanalyse

Die Hauptkomponentenanalyse (engl. *Principal Component Analysis*, PCA) ist ein Verfahren zur Dimensionsreduktion von Daten. Genauer: Es handelt sich um eine Methode, um komplexe Daten auf ihr Wesentliches zu reduzieren, was eine Weiterverarbeitung und Visualisierung erleichtert.

In diesem Kapitel wird zunächst die grundlegende Idee der PCA mithilfe eines intuitiven Ansatzes erläutert. Anschließend leiten wir die vorher informell eingeführten Konzepte der Hauptkomponentenanalyse mathematisch her. Zum Abschluss wird die Verbindung zwischen der PCA und SVD aufgezeigt und die potenziellen Vorteile der Berechnung mit der SVD anhand eines Beispiels in `Python` veranschaulicht. Da die PCA eng mit der Anwendung des nächsten Kapitels verknüpft ist und dort erneut aufgegriffen wird, verzichten wir hier auf ausführlichere Anwendungsmöglichkeiten der Hauptkomponentenanalyse.

3.1. Intuition der PCA

Angenommen, beim Familienessen käme die Frage auf, welche der mitgebrachten Weine sich am ähnlichsten sind. Um diese Frage zu beantworten, überlegt sich die Familie verschiedene Merkmale und ordnet jedem Wein für jedes Merkmal eine Zahl zwischen -3 und 3 zu. Dadurch können die Weine als Punkte im Raum bezüglich der verschiedenen Werte dargestellt und anschließend analysiert werden, welche Weine sich gruppieren, sich also ähneln.

In Abbildung A.3, zu finden in Anhang A, wird dies für verschiedene $n :=$ *Anzahl der Merkmale* verdeutlicht.

Das Problem wird schnell ersichtlich: Eine visuelle Interpretation ist zwar möglich, allerdings nur im niedrig-dimensionalen Raum, für eine größere Anzahl an Merkmalen (Dimensionen) besteht die Notwendigkeit, die Anzahl zu reduzieren. Diese Dimensionsreduktion stellt in vielen Fällen auch unabhängig von der visuellen Interpretation eine sinnvolle Maßnahme dar. In Bezug auf unser Beispiel bestehe die Möglichkeit, dass „Alkoholgehalt“ und „Schwere“ stark

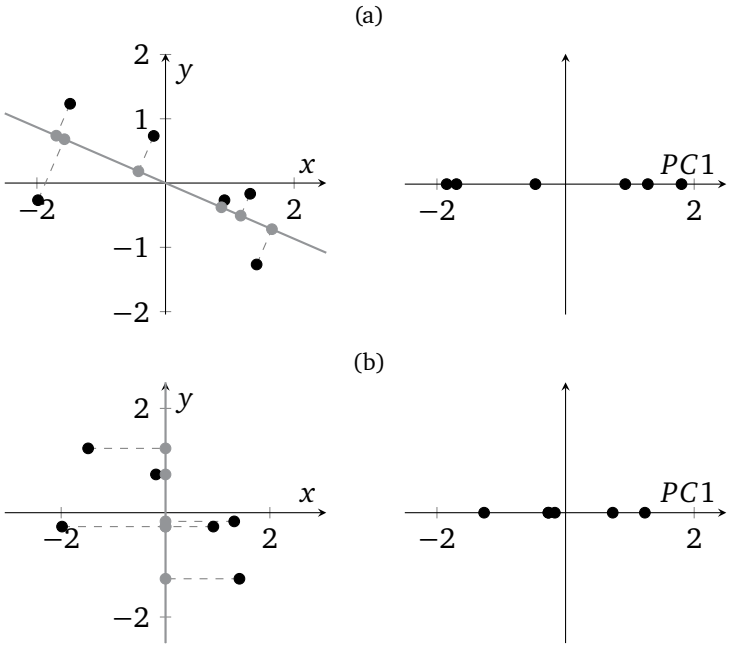


Abb. 3.1. Projektionen im zweidimensionalen Raum

korrelieren und somit redundant für eine Rekonstruktion der Weine sind. Eine andere Möglichkeit zur Reduzierung der Merkmale ist dadurch gegeben, dass gewisse Merkmale wenig Informationen über die Charakteristiken der Weine enthalten, wie beispielsweise die Flaschenform oder die Fließgeschwindigkeit.

Die Hauptkomponentenanalyse konstruiert neue, unkorrelierte Richtungen, die sich als Linearkombinationen aus den bestehenden Merkmalen zusammensetzen. Dadurch könnten beispielsweise „Alkoholgehalt“ und „Schwere“ zu einer neuen *Hauptrichtung* zusammengefasst werden. Anschließend werden die Weine auf den durch diese Richtung definierten Unterraum projiziert. Die projizierten Koordinaten entlang dieser neuen Achse ergeben die erste *Hauptkomponente* (PC1). Der Unterschied zwischen Hauptrichtung und Hauptkomponente wird am Ende der Intuition präzisiert.

Die Projektion wird in Abbildung 3.1 veranschaulicht. Der Unterschied zwischen Abbildung 3.1a und Abbildung 3.1b zeigt folgendes Problem auf: Wie kann die neue Richtung optimal gewählt werden, sodass unsere Daten trotz der Dimensionsreduktion so originalgetreu wie möglich rekonstruiert werden können? Dafür gibt es zwei alternative, aber äquivalente, Formulierungen:

1. Die Richtung wird so konstruiert, dass ein Minimum an Informationen verloren wird.

2. Die Richtung wird so konstruiert, dass eine maximale Varianz erhalten bleibt.

Eine Äquivalenz dieser beiden Aussagen kann durch den Satz des Pythagoras hergeleitet werden, indem das rechtwinklige Dreieck zwischen einem Punkt, dessen Projektion und dem Ursprung betrachtet wird. Wird der Informationsverlust (Distanz zwischen der Projektion und dem Original) minimiert, maximiert sich die Varianz (Abstand zum Ursprung). Es sei angemerkt, dass dafür von einer Zentrierung der Daten ausgegangen wird, also von einem Mittelwert gleich null.

Mit diesem Hintergrund wird intuitiv ersichtlich, dass Abbildung 3.1a im Vergleich zu Abbildung 3.1b vorzuziehen ist, was sich im Ergebnis widerspiegelt, in dem die räumliche Verteilung im Wesentlichen erhalten bleibt.

In vielen mathematischen Texten erfolgt keine klare Differenzierung zwischen den Begriffen Hauptrichtung und Hauptkomponente. Stattdessen wird der Begriff Hauptkomponente häufig synonym für beide verwendet. In dieser Arbeit definieren wir die erste Hauptrichtung als den Einheitsvektor, der den eindimensionalen Unterraum aufspannt, auf den die Daten mit maximaler Varianz projiziert werden. Die erste Hauptkomponente bezeichnet die projizierten Koordinaten der Datenpunkte entlang dieser Hauptrichtung. In höheren Dimensionen wird die zweite Hauptrichtung so gewählt, dass sie orthogonal zur ersten Hauptrichtung liegt und die verbleibende Varianz maximiert. Dies kann beliebig fortgesetzt werden, die PCA besitzt allerdings die nützliche Eigenschaft, dass die Hauptrichtungen nach „Wichtigkeit“ sortiert sind, also die ersten Richtungen bereits den Großteil der Varianz erklären. Eine genauere Erläuterung dieser Aussage wird dabei in der folgenden mathematischen Herleitung gegeben.

3.2. Mathematische Herleitung der PCA

Bevor die vorangegangenen Überlegungen formalisiert werden können, rekapitulieren wir durch Wiederholung 3.1 die Formel der orthogonalen Projektion.

> **Wiederholung 3.1.**
> Sei $n \in \mathbb{N}$ und $u, x \in \mathbb{R}^n$ mit $\|u\| = 1$.

Dann ist der orthogonal projizierte Vektor $\text{proj}_u(x)$ von x auf u gegeben durch

$$\text{proj}_u(x) = \langle x, u\rangle u = (x^T u)u.$$

Das Ziel der Herleitung in diesem Abschnitt wird durch Anwendung 3.2 zusammengefasst.

ANWENDUNG 3.2 (Hauptkomponentenanalyse).
Seien $n, d \in \mathbb{N}$,

$$X = [x_1 \dots x_d] \in \mathbb{R}^{n\times d}$$

eine standardisierte[a] Datenmatrix, die d Merkmale über n Objekte hinweg speichert und

$$\mathbb{R}^d \ni x^{(i)} := X_{i,:} \quad \text{für } i \in \{1, \dots, n\},$$

die i-te transponierte Zeile von X, also ein Datenpunkt.
Bei der Projektion der Daten in $\mathbb{R}^k$ für $\mathbb{N} \ni k < d$ mit dem Ziel, dass die maximale Varianz der $x^{(i)}$ erhalten bleiben soll, wird die Basis von $\mathbb{R}^k$ durch die ersten k Hauptrichtungen $u_1, \dots, u_k \in \mathbb{R}^d$ gegeben.
Diese entsprechen den ersten k Eigenvektoren der Matrix $\frac{1}{n}X^T X \in \mathbb{R}^{d\times d}$, sortiert in absteigender Reihenfolge der zugehörigen Eigenwerte.

[a] Siehe Schritt 1 im Beweis

Beweis. Der Beweis orientiert sich zum Großteil an [NM23, S.166-169] und [Hsu16, S.32 f.].

Schritt 1. Vorbereitung der Daten:
Wir standardisieren zunächst X, indem

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j} \quad \text{für alle } j \in \{1, \dots, d\},$$

wobei

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_j^{(i)}, \quad s_j^2 = \frac{1}{n}\sum_{i=1}^{n} (x_j^{(i)} - \mu_j)^2$$

jeweils die Mittelwerte, bzw. die Varianzen der einzelnen Merkmale, also der Spalten sind. Durch die Subtraktion des Mittelwerts werden die Daten um den Ursprung zentriert. Die Division durch die Standardabweichung verhindert

Ungenauigkeiten, die aufgrund unterschiedlicher Skalen der Merkmale entstehen könnten. Falls Merkmal A beispielsweise das Bruttoinlandsprodukt und Merkmal B die Geburtenrate verschiedener Länder darstellt, wird dadurch eine Vergleichbarkeit gewährleistet. Dies ist allerdings für die weitere Rechnung nicht zwingend erforderlich.

Schritt 2. Herleitung für $k = 1$:
Es sei daran erinnert, dass die erste Hauptrichtung der Richtung entspricht, die die Varianz der Projektion maximiert. Dafür wird zunächst mithilfe von Wiederholung 3.1 der Mittelwert μ_{proj} der projizierten Vektoren auf die Hauptrichtung u berechnet:

$$\mu_{\text{proj}} = \frac{1}{n}\sum_{i=1}^{n}\left(x^{(i)^T}u\right)u = \left(\left(\frac{1}{n}\sum_{i=1}^{n}x^{(i)}\right)^T u\right)u = \mathbf{0},$$

da durch die Standardisierung der Spaltenmittelwert für jede Spalte von X null beträgt, wodurch

$$\sum_{i=1}^{n} x^{(i)} = \mathbf{0}.$$

Folglich entspricht die Standardabweichung der Projektion eines beliebigen Vektors $x^{(i)}$ dem Abstand vom Ursprung, also

$$\|\text{proj}_u\left(x^{(i)}\right)\| = \|\left(x^{(i)^T}u\right)u\| = |\left(x^{(i)^T}u\right)|\|u\| = |x^{(i)^T}u|.$$

Damit ist die Varianz der projizierten Punkte gegeben durch

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}\left(x^{(i)^T}u\right)^2 &= \frac{1}{n}\sum_{i=1}^{n}x^{(i)^T}ux^{(i)^T}u\\ &= \frac{1}{n}\sum_{i=1}^{n}u^T x^{(i)}x^{(i)^T}u \qquad \text{(Skalarprodukt kommutativ)}\\ &= u^T\left(\frac{1}{n}\sum_{i=1}^{n}x^{(i)}x^{(i)^T}\right)u\end{aligned}$$

Definiere

$$\Sigma := \frac{1}{n}\sum_{i=1}^{n}x^{(i)}x^{(i)^T} = \frac{1}{n}X^T X \in \mathbb{R}^{d\times d}.$$

Bei zentrierten Daten ist diese Matrix als *Kovarianzmatrix* bekannt, in unserem Fall ist sie die Kovarianzmatrix der verschiedenen Merkmale. Damit haben wir

das Ziel der Herleitung auf folgendes Optimierungsproblem reduziert:

$$\begin{aligned} \max \quad & u^T \Sigma u, \\ \text{u.d.B.} \quad & \|u\| = 1. \end{aligned}$$

Dieses Optimierungsproblem wird in der Literatur meist mithilfe von Lagrange-Multiplikatoren gelöst. In dieser Arbeit werden wir einen anderen Ansatz verfolgen und mit dem, im vorherigen Kapitel bewiesenen, Spektralsatz (Satz 2.6) vorgehen. Beachte dafür zunächst, dass Σ symmetrisch ist:

$$\Sigma^T = \left(\frac{1}{n} \sum_{i=1}^{n} x^{(i)} x^{(i)^T} \right)^T = \frac{1}{n} \sum_{i=1}^{n} \left(x^{(i)} x^{(i)^T} \right)^T = \Sigma.$$

Damit sind die Voraussetzungen für den Spektralsatz erfüllt, womit

$$\Sigma = R \Lambda R^T$$

für orthogonales $R = [r_1 \dots r_d] \in \mathbb{R}^{d \times d}$ und diagonales $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_d)$ mit $\lambda_1 \geq \dots \geq \lambda_d$. Für $w := R^T u$ gilt dann

$$u^T \Sigma u = u^T R \Lambda R^T u = w^T \Lambda w = w^T \begin{bmatrix} \lambda_1 w_1 \\ \vdots \\ \lambda_d w_d \end{bmatrix} = \sum_{i=1}^{d} \lambda_i w_i^2.$$

Nach Bedingung ist $\|u\| = 1$ womit nach Lemma 2.18 auch $\|w\| = 1$. Da

$$\sum_{i=1}^{d} \lambda_i w_i^2 = \lambda_1 w_1^2 + \lambda_2 w_2^2 + \dots + \lambda_d w_d^2$$

und $\lambda_1 \geq \dots \geq \lambda_d$ wird der Ausdruck maximiert für $w = \mathbf{e}_1$. Es folgt

$$u = Rw = R\mathbf{e}_1 = r_1,$$

womit u nach Korollar 2.7 gleich dem zugehörigen Eigenvektor zum größten Eigenwert von Σ ist.

Schritt 3. Herleitung für bel. $k \in \mathbb{N}$ mit vollständiger Induktion über k:
Induktionsanfang. Gegeben durch Schritt 2.
Induktionshypothese. Angenommen die ersten $k-1$ Eigenvektoren $r_1, \dots, r_{k-1}$ von Σ maximieren die Varianz der Projektion.

Induktionsschritt. Wir wollen zeigen, dass

$$\frac{1}{n}\sum_{j=1}^{k}\sum_{i=1}^{n}\left(x^{(i)^T}u_j\right)^2$$

maximiert wird für $u_k = r_k$. Betrachte dafür

$$\frac{1}{n}\sum_{j=1}^{k}\sum_{i=1}^{n}\left(x^{(i)^T}u_j\right)^2 = \underbrace{\frac{1}{n}\sum_{j=1}^{k-1}\sum_{i=1}^{n}\left(x^{(i)^T}u_j\right)^2}_{A} + \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(x^{(i)^T}u_k\right)^2}_{B}.$$

Nach Induktionshypothese ist A maximal für $u_1 = r_1, u_2 = r_2, \dots, u_{k-1} = r_{k-1}$. Um B zu maximieren, wird analog zu Schritt 2 vorgegangen. Es wird zunächst das Optimierungsproblem

$$\begin{aligned} \max \quad & u_k^T \Sigma u_k, \\ \text{u.d.B.} \quad & \|u_k\| = 1, \\ & \langle u_k, u_i \rangle = 0 \quad \text{für alle } i \in \{1, \dots, k-1\}, \end{aligned}$$

aufgestellt mit zusätzlicher Orthogonalitätsbedingung. Anschließend kann, mithilfe der Eigenschaft, dass bei symmetrischen Matrizen Eigenvektoren zu verschiedenen Eigenwerten orthogonal sind (Bemerkung 2.8), das Optimierungsproblem durch $u_k = r_k$ gelöst werden, womit der Beweis abgeschlossen ist.

Hinweis. Die i-te Hauptkomponente ist dann durch Xu_i gegeben, wodurch die Projektionen der Datenpunkte auf u_i im durch u_i aufgespannten Unterraum zusammengefasst werden. □

Korollar 3.3.
Seien die Voraussetzungen aus Anwendung 3.2 gegeben.
Die Varianz der Datenpunkte entlang einer beliebigen Hauptrichtung $u_j \in \mathbb{R}^d$ für $j \in \{1, \dots, k\}$ entspricht genau dem zugehörigen Eigenwert λ_j zu u_j.

Beweis. Wie bereits im vorherigen Beweis gezeigt, ist die Varianz der Datenpunkte entlang von u_j gegeben durch

$$u_j^T \Sigma u_j.$$

Mit dem Wissen, dass u_j ein Eigenvektor von Σ mit zugehörigem Eigenwert λ_j ist, kann dies umgeformt werden zu

$$u_j^T(\lambda_j u_j) = \lambda_j u_j^T u_j = \lambda_j,$$

da $\|u_j\| = 1$. □

BEMERKUNG 3.4.
Daraus ergibt sich die in der Intuition erwähnte Eigenschaft der Hauptkomponentenanalyse: Ein Großteil der Varianz kann bereits durch die ersten Hauptkomponenten erklärt werden, da die Eigenwerte der Kovarianzmatrix Σ in absteigender Reihenfolge die jeweilige Varianz angeben.

3.3. VERBINDUNG ZUR SVD

Das Ziel dieses Abschnitts ist, die Hauptrichtungen, Hauptkomponenten und die Varianzen mithilfe der Singulärwertzerlegung der Datenmatrix auszudrücken, statt mit der Spektralzerlegung der Kovarianzmatrix. Diese Art der Berechnung wird in der Praxis häufig verwendet, da sie einen entscheidenden Vorteil besitzt, auf den am Ende des Abschnitts genauer eingegangen wird.

Sei also erneut $X \in \mathbb{R}^{n\times d}$ eine standardisierte Datenmatrix mit zugehöriger Kovarianzmatrix

$$\Sigma = \frac{1}{n}X^T X = V\Lambda V^T \in \mathbb{R}^{d\times d} \tag{3.1}$$

nach dem Spektralsatz für orthogonales $V \in \mathbb{R}^{d\times d}$, welches die Eigenvektoren von Σ als Spaltenvektoren enthält und diagonales $\Lambda \in \mathbb{R}^{d\times d}$ mit den zugehörigen Eigenwerten auf der Diagonalen.

Sei außerdem

$$X = USV^T$$

die Singulärwertzerlegung von X mit „diagonalem“ $S \in \mathbb{R}^{n\times d}$ und orthogonalem $U \in \mathbb{R}^{n\times n}$. Da die Eigenvektoren von $\Sigma = \frac{1}{n}X^T X$ und $X^T X$ identisch sind (folgt direkt aus Definition 2.1; nur die Eigenwerte unterscheiden sich um einen Faktor n), ist V die gleiche Matrix wie in (3.1). Folglich sind die Hauptrichtungen durch die Spalten von V gegeben.

Um die Varianzen zu ermitteln, betrachten wir

$$V\Lambda V^T = \frac{1}{n}X^T X = \frac{1}{n}VS^T U^T USV^T = V\frac{S^T S}{n}V^T.$$

Damit ist $\Lambda = \frac{S^T S}{n}$ und es kann ein beliebiger Eigenwert λ_i von Σ, also die Varianz zur i-ten Hauptrichtung, durch den Singulärwert σ_i von X ausgedrückt werden mit

$$\lambda_i = \frac{\sigma_i^2}{n},$$

für $i \in \{1, \dots, \mathrm{rg}(X)\}$, wobei $\lambda_i = 0$ für $i > \mathrm{rg}(X)$.

Die Hauptkomponente zu einer beliebigen Hauptrichtung v wird, wie bereits in der Herleitung angemerkt, durch Xv beschrieben. Es lassen sich also alle Hauptkomponenten zusammenfassen durch

$$XV = USV^T V = US.$$

Damit lässt dich die Hauptkomponentenanalyse vollständig durch die Singulärwertzerlegung ausdrücken:

- Die Hauptrichtungen durch die Spaltenvektoren von V.
- Die Hauptkomponenten durch US.
- Die Varianzen durch $\frac{\sigma_i^2}{n}$ mit $n :=$ Anzahl der Objekte.

Falls nur eine Darstellung durch die ersten k Hauptrichtungen erwünscht ist, kann dies durch die trunkierte Singulärwertzerlegung X_k erreicht werden mit $\mathrm{rg}(X_k) = k$.

Der Vorteil, die PCA mit der SVD von X zu berechnen, statt mit der Spektralzerlegung von Σ, zeichnet sich durch die numerische Stabilität der Berechnung aus. Um dies zu zeigen, wird ein Exkurs in die algorithmische Mathematik vorgenommen, wobei die für uns wichtigen Konzepte in Wiederholung 3.5 rekapituliert werden.

Wiederholung 3.5.

- Die *Kondition* eines Problems gibt an, wie sich kleine Eingabefehler auf den Ausgabefehler auswirken. Bei einem gut konditionierten Problem führen kleine Eingabefehler zu kleinen Ausgabefehlern.
- Ein Algorithmus ist dann *numerisch stabil*, falls der nicht vermeidbare Fehler, bedingt durch die Kondition, nicht weiter verstärkt wird.
- Für eine Matrix $A \in \mathbb{R}^{m \times n}$ mit $\mathrm{rg}(A) = r$ ist die *Konditionszahl* κ

gegeben durch

$$\kappa(A) = \frac{\sigma_1}{\sigma_r},$$

wobei σ_1 den größten und σ_r den kleinsten Singulärwert von A darstellt.

- Desto kleiner die Konditionszahl ist, desto besser konditioniert ist das Problem.

Wir zeigen nun, dass die Kovarianzmatrix Σ schlechter konditioniert ist als X und somit anfälliger für Ungenauigkeiten bei numerischen Berechnungen, wie dem Bestimmen von Eigenwerten. Nach Wiederholung 3.5 kann die Konditionszahl mithilfe der Singulärwerte von Σ bestimmt werden, die als die Wurzeln der positiven Eigenwerte von $\Sigma^T\Sigma = \Sigma^2$ gegeben sind.

Sei v ein beliebiger Eigenvektor von Σ zum Eigenwert λ, dann gilt

$$\Sigma^2 v = \Sigma\Sigma v = \Sigma\lambda v = \Sigma v\lambda = \lambda^2 v.$$

Daraus folgt, dass die Eigenvektoren von Σ und Σ^2 identisch sind, während die Eigenwerte quadriert werden. Die Singulärwerte von Σ entsprechen damit den Eigenwerten von Σ und

$$\kappa(\Sigma) = \frac{\sigma_1^2/n}{\sigma_r^2/n} = \left(\frac{\sigma_1}{\sigma_r}\right)^2 = \kappa(X)^2.$$

Die Konditionszahl von Σ ist somit das Quadrat der Konditionszahl von X, was insbesondere bei schlecht konditionierten Matrizen zu erheblichen Ungenauigkeiten führen kann.

Diese Folgerung veranschaulichen wir an einem Beispiel. Sei

$$X = \begin{bmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix},$$

die sogenannte *Läuchli-Matrix*. Die quadrierten Singulärwerte von X sind bekannt und gegeben durch

$$\sigma_1^2 = 3 + \varepsilon^2, \quad \sigma_2^2 = \varepsilon^2, \quad \sigma_3^2 = \varepsilon^2.$$

Um diese Werte zu berechnen, verwenden wir zum einen die Singulärwertzerlegung von X und zum anderen die direkte Berechnung der Eigenwerte von

X^TX. Mithilfe von Python erreichen wir dies durch:

```
# Eigenwerte von L^T L
eigvals = np.linalg.eigvalsh(L.T @ L)[::-1]

# Quadrate der Singulärwerte von L
singular_values = np.linalg.svd(L, compute_uv=False)
singular_values_squared = singular_values**2
```

Die Funktionen `np.linalg.eigvalsh` und `np.linalg.svd` berechnen dabei jeweils die Eigenwerte einer reell symmetrischen Matrix, bzw. die Singulärwertzerlegung einer Matrix, wobei in diesem Beispiel durch `compute_uv=False` ausschließlich die Singulärwerte berechnet werden. Der vollständige Code ist zu finden unter Code B.1 in Anhang B.

Für $\varepsilon = 10^{-3},\ 10^{10}$ und 10^{-20} erhalten wir damit folgenden Output:

```
Epsilon: 0.001
Eigenwerte vs. Quadrate der Singulärwerte:
3.000e+00        3.000e+00
1.000e-06        1.000e-06
1.000e-06        1.000e-06
-
Epsilon: 1e-10
Eigenwerte vs. Quadrate der Singulärwerte:
3.000e+00        3.000e+00
-1.770e-17       1.000e-20
-5.849e-16       1.000e-20
-
Epsilon: 1e-20
Eigenwerte vs. Quadrate der Singulärwerte:
3.000e+00        3.000e+00
-1.770e-17       1.000e-40
-5.849e-16       1.000e-40
-
```

Es lässt sich beobachten, dass für $\varepsilon = 10^{-3}$ beide Berechnungsmethoden übereinstimmen und exakt arbeiten. Für $\varepsilon = 10^{-10}$ und $\varepsilon = 10^{-20}$ treten jedoch Ungenauigkeiten in der direkten Eigenwertberechnung von X^TX auf, während die SVD weiterhin zu stabilen Ergebnissen führt.

Damit sind die Vorteile der Singulärwertzerlegung bei der Berechnung der Hauptkomponentenanalyse gezeigt und es wird zum nächsten Anwendungsbeispiel übergegangen.

4 Empfehlungssysteme

Empfehlungssysteme sind eine Anwendungsart der Singulärwertzerlegung, mit der bereits die meisten Menschen in Kontakt gekommen sind. Seien es Filmempfehlungen bei Netflix oder Produktempfehlungen bei Amazon, die Wahrscheinlichkeit ist groß, dass diese mithilfe einer Abwandlung der SVD generiert werden. Es sei jedoch angemerkt, dass Empfehlungssysteme keine eindeutige mathematische Realisierung besitzen. Vielmehr existieren zahlreiche Formen und Varianten, wobei eine Behandlung aller Formen genügend Material für eine alleinstehende Abschlussarbeit bieten würde. Aus diesem Grund konzentrieren wir uns in diesem Kapitel, mit Blick auf den Rahmen dieser Arbeit, auf ein Modell, das ausschließlich auf der bereits eingeführten reinen Singulärwertzerlegung basiert.

Dafür wird zunächst die grundlegende Idee von Empfehlungssystemen veranschaulicht. Anschließend wird der Ansatz des ausgesuchten Modells hergeleitet und anhand eines Beispiels näher erläutert. Um das Kapitel abzuschließen, wird mithilfe von `Python` das zuvor hergeleitete Empfehlungssystem realisiert und durch ein weiteres Konzept erweitert.

4.1. Grundlagen von Empfehlungssystemen

Wir verweilen in diesem Kapitel beim Beispiel der Filmempfehlungen. Die Ausgangslage für ein entsprechendes Empfehlungssystem wird in Tabelle 4.1 veranschaulicht.

Gegeben ist eine Nutzer-Item-Matrix, in der jede Zeile einen Nutzer und jede Spalte einen Film repräsentiert, wobei die einzelnen Einträge die abgegebenen Bewertungen von $1 - 5$ der Nutzer für den jeweiligen Film darstellen. Das Ziel des Empfehlungssystems ist, basierend auf den vorhandenen Bewertungen sinnvolle Empfehlungen zu generieren. Dafür wird die Annahme getroffen, dass die Bewertungen nicht unabhängig erfolgen, sondern einer bestimmten Struktur folgen. Es wird also angenommen, dass es zugrunde liegende Muster

Tab. 4.1. Nutzer-Item-Matrix

	Items			
Nutzer	Film A	Film B	Film C	Film D
Nutzer 1		4	2	0
Nutzer 2	1	2	3	5
Nutzer 3	1	2		
Nutzer 4		4	3	3
Nutzer 5	4	2	1	1
Nutzer 6	5			2

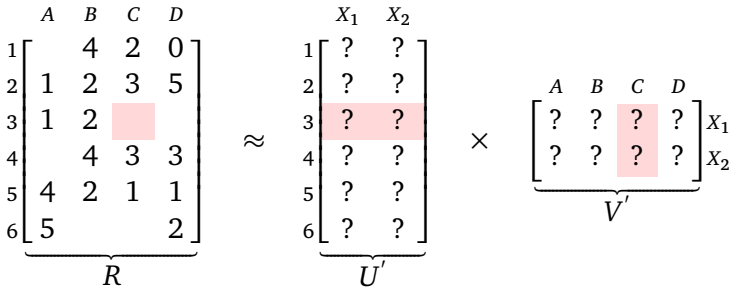


Abb. 4.1. Nutzer-Matrix und Item-Matrix

gibt, nach denen Nutzer mit ähnlichen Präferenzen auch tendenziell ähnliche Bewertungen vergeben. Ein Beispiel dafür wäre, dass Nutzer mit einer Vorliebe für Horrorfilme diese häufiger höher bewerten als andere Nutzer. Solche Muster werden als *latente Merkmale* bezeichnet [KBV09, S. 31].

Im Folgenden werden diese Merkmale genutzt, indem die Nutzer-Item-Matrix R als Produkt zweier Matrizen dargestellt wird: einer Nutzer-Matrix U', in der die Nutzer durch die latenten Merkmale beschrieben werden, und einer Item-Matrix V' mit der Beschreibung der Filme durch die Merkmale. Dieses Konzept wird in Abbildung 4.1 verdeutlicht mit den latenten Merkmalen X_1 und X_2.

Fehlende Bewertungen können als Skalarprodukt der jeweiligen Vektoren approximiert werden, wie in der Abbildung farblich hervorgehoben ist. Da Nutzer und Filme durch dieselben Merkmale repräsentiert werden, lassen sie sich in einem gemeinsamen Raum abbilden, wobei die kontextuelle Bedeutung der latenten Merkmale in der Regel nicht explizit bekannt ist. Im Rahmen des hier behandelten Beispiels könnte allerdings X_1 für „Horror“ und X_2 für „Drama“ stehen. In diesem Fall wird jeder Nutzer durch seine Vorliebe für die

beiden Genres beschrieben, während jeder Film durch seine Ausprägung dieser Genres charakterisiert wird. Wie der gemeinsame latente Raum für Nutzer und Filme dann aussehen könnte, ist in Abbildung A.2 in Anhang A veranschaulicht.

Es bleibt damit die Frage, wie die Matrizen U' und V' berechnet werden können, um sinnvolle Empfehlungen generieren zu können. Dafür fassen wir zusammen, welche Eigenschaften die Matrizen erfüllen sollen:

1. Sie sollen eine sinnvolle Approximation von R darstellen, sodass die wichtigsten Zusammenhänge der Nutzer-Item-Matrix erhalten bleiben.
2. Die Nutzer und Items sollen durch die Zeilen von U', bzw. durch die Spalten von V' mithilfe derselben latenten Merkmale ausgedrückt werden können.

4.2. Einführung in PureSVD

Es gibt zahlreiche verschiedene Ansätze, um die entsprechenden Matrizen zu berechnen. Wie bereits in der Einleitung des Kapitels angesprochen, wird sich hier auf ein Modell konzentriert, welches ausschließlich auf der Singulärwertzerlegung basiert. Dieses Modell wird als *PureSVD* [CKT10] bezeichnet. PureSVD gehört zu den *Top-N-Empfehlungssystemen*, womit das Ziel nicht darin besteht, die fehlenden Werte so präzise wie möglich vorherzusagen. Stattdessen wird versucht, basierend auf den vorhanden Bewertungen eine Auswahl an Empfehlungen zu generieren, die den Vorlieben des Nutzers entsprechen, wobei lediglich die Rangfolge relevant ist. Damit stellt beispielsweise die Approximation des Wertes in Abbildung 4.1 keine valide Vorhersage dar, sondern beschreibt vielmehr die Ähnlichkeit zwischen dem entsprechenden Nutzer und Film.

Wie auch bei der Hauptkomponentenanalyse wird das Ziel der Herleitung in Anwendung 4.1 zusammengefasst, wobei die Herleitung diesmal nicht als formaler Beweis bezeichnet werden kann.

Anwendung 4.1 (PureSVD).
Seien $m, n \in \mathbb{N}$ und $R \in \mathbb{R}^{m \times n}$ eine Nutzer-Item-Matrix, wobei fehlende Werte als null betrachtet werden.
Sei außerdem $R_k = U_k \Sigma_k V_k^T$ die trunkierte SVD von R mit $U_k \in \mathbb{R}^{m \times k}$, $\Sigma_k \in \mathbb{R}^{k \times k}$ und $V_k \in \mathbb{R}^{n \times k}$ für $\operatorname{rg}(R_k) = k$.
Dann ist die Nutzer-Matrix $U' \in \mathbb{R}^{m \times k}$ und Item-Matrix $V' \in \mathbb{R}^{k \times n}$ gegeben durch

$$U' = U_k \Sigma_k, \quad V' = V_k^T.$$

Herleitung. Es soll gezeigt werden, dass die definierten Matrizen die beiden im vorherigen Abschnitt genannten Eigenschaften erfüllen. Die erste Eigenschaft folgt dabei direkt aus dem Eckart-Young-Satz (Satz 2.20), da die trunkierte SVD die beste Rang-k-Approximation darstellt.

Ein intuitives Verständnis für die zweite Eigenschaft bietet Korollar 2.22: Da sich die Nutzer als Linearkombination der Filme darstellen lassen, befinden sie sich im Spaltenraum von R_k. Für diesen Raum bilden die Spalten von U_k eine Basis und da Σ_k als Diagonalmatrix nur einer Streckung entspricht, bleibt die Basis in U' erhalten. Dies spiegelt sich in Abbildung 4.1 wider: Jeder Nutzer kann ebenfalls als Linearkombination der latenten Merkmale ausgedrückt werden. Die gleiche Argumentation gilt für die Filme über den Zeilenraum.

Eine etwas formalere Herleitung ergibt sich durch die Hauptkomponentenanalyse. Es ist allerdings ausgesprochen wichtig zu erwähnen, dass es sich hier um eine *nicht zentrierte* PCA handelt, da die Spaltenmittelwerte nicht subtrahiert werden. Die Unterschiede zwischen dieser und der im vorherigen Kapitel eingeführten Analyse werden in [CJ09] detailliert beleuchtet, wobei hier auf eine nähere Erklärung verzichtet wird. Es genügt anzumerken, dass auch die nicht zentrierte PCA mithilfe des bekannten Weges der SVD berechnet werden kann und im Kontext von PureSVD eine Vergleichbarkeit zur klassischen PCA besteht.[1]

Wird nun die (nicht zentrierte) PCA auf R angewendet sind die ersten k Hauptkomponenten, wie im vorherigen Kapitel gezeigt, gegeben durch $U_k\Sigma_k = U'$. Damit stellen die latenten Merkmale die Hauptrichtungen dar, auf die die Nutzer projiziert werden. Da diese wiederum als Linearkombination der verschiedenen Merkmale (hier: Filme) definiert und durch die Spalten von V_k gegeben sind, ergibt sich wie gewünscht $V' = V_k^T$ und die Nutzer und Filme können durch dieselben Merkmale ausgedrückt werden.

Hinweis. Wie wir wissen, stehen die Singulärwerte von R in enger Verbindung mit der durch die Hauptkomponenten erklärten Varianz. Obwohl diese Verbindung bei der nicht zentrierten PCA nur noch eingeschränkt gilt, können wir dennoch Rückschlüsse auf die Bedeutung der Singulärwerte ziehen. In diesem Kontext gibt ihre Größe Aufschluss über die Wichtigkeit oder Stärke der latenten Merkmale. Falls beispielsweise das erste latente Merkmal das Genre „Horror“ und das zweite „Drama“ repräsentiert, kann daraus geschlossen werden, dass R mehr Informationen über Horrorfilme und deren bevorzugende Nutzer enthält als über das Genre Drama. ◇

[1] Da in der Praxis die meisten Bewertungen nicht bekannt sind, liegen meist sehr spärliche Matrizen vor, womit der Mittelwert der meisten Spalten ohnehin nahe null ist.

Da die Herleitung, wie vorher angekündigt, keinen formalen Beweis darstellt und nicht umfänglich beantwortet, warum die Multiplikation der beiden Matrizen Werte produziert, die die Ähnlichkeit zwischen Nutzern und Filmen widerspiegeln, wird dieser Zusammenhang anhand eines Beispiels nach [Nik+19, S. 62–64] verdeutlicht. Dafür definieren wir zunächst ein Maß für die Ähnlichkeit zweier Vektoren.

Definition 4.2.
Sei $n \in \mathbb{N}$ und $a, b \in \mathbb{R}^n \backslash \{\mathbf{0}\}$.
Die *Kosinus-Ähnlichkeit* zweier Vektoren a und b ist definiert durch

$$\cos(\theta) = \frac{\langle a, b \rangle}{\|a\| \|b\|} \in [-1, 1]$$

für den eingeschlossenen Winkel $\theta \in [0, \pi]$ zwischen den Vektoren.
Je größer $\cos(\theta)$, desto „ähnlicher“ sind sich a und b, wobei für $\cos(\theta) = 0$ Unabhängigkeit gilt.

Es gelten weiterhin die Voraussetzungen aus Anwendung 4.1. Sei

$$R = U \Sigma V^T$$

die vollständige Singulärwertzerlegung von R mit $U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}$ und $\Sigma \in \mathbb{R}^{m \times n}$. Dann ist

$$R V_k V_k^T = U \Sigma V^T V_k V_k^T \overset{(*)}{=} U \Sigma \underbrace{\begin{bmatrix} I_k \\ \mathbf{0} \end{bmatrix}}_{n \times k} V_k^T = U \underbrace{\begin{bmatrix} \Sigma_k \\ \mathbf{0} \end{bmatrix}}_{m \times k} V_k^T = U_k \Sigma_k V_k^T = R_k. \tag{4.1}$$

(∗) *Da V orthonormal ist.*

Die Approximationsmatrix R_k kann also nur durch die originale Matrix R und Item-Matrix V_k ausgedrückt werden.

Seien $v_i, v_j \in \mathbb{R}^k$ beliebige Zeilenvektoren von V_k für $i, j \in \{1, \dots, n\}$. Betrachten wir nun die Matrix $V_k V_k^T$ mit den latenten Merkmalen $X_1, \dots, X_k$ und Filmen $F_1, \dots, F_n$:

$$V_k V_k^T = \begin{array}{c} \\ F_1 \\ \vdots \\ F_i \\ \vdots \\ F_n \end{array} \overset{\begin{array}{c} X_1 \ \dots \ X_k \end{array}}{\begin{bmatrix} & & \\ & & \\ & & \\ - & v_i & - \\ & & \\ & & \\ & & \end{bmatrix}} \times \overset{\begin{array}{c} F_1 \ \dots \ F_j \ \dots \ F_n \end{array}}{\begin{bmatrix} & | & \\ & v_j & \\ & | & \end{bmatrix}} \begin{array}{c} X_1 \\ \vdots \\ X_k \end{array} = \overset{\begin{array}{c} F_1 \ \dots \ F_j \ \dots \ F_n \end{array}}{\begin{bmatrix} & & \\ & & \\ & & \\ & \alpha_{ij} & \\ & & \\ & & \\ & & \end{bmatrix}} \begin{array}{c} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_n \end{array}.$$

Damit gilt für ein beliebiges Element α_{ij} von $V_k V_k^T$

$$\alpha_{ij} = \langle v_i, v_j \rangle = \|v_i\| \|v_j\| \cos(\theta_{ij}) \tag{4.2}$$

nach Definition 4.2, wobei θ_{ij} der eingeschlossene Winkel zwischen v_i und v_j ist. Folglich repräsentiert α_{ij} die Kosinus-Ähnlichkeit zwischen Film F_i und F_j bezüglich der latenten Merkmale, skaliert mit einem Wert, der abhängig von der Popularität der Filme ist.[2]

Mit dieser Erkenntnis wenden wir uns der Approximationsmatrix R_k zu und betrachten, welche Empfehlungen für einen bestimmten Nutzer N_u generiert werden würden mit $u \in \{1, \dots, m\}$. Angenommen der Nutzer habe nur die Filme F_1, F_3 und F_5 bewertet mit den Bewertungen $(R)_{u1} = 1, (R)_{u3} = 2$ und $(R)_{u5} = 5$. Außerdem soll für dieses Beispiel nur entschieden werden, ob der Film F_2 oder F_4 empfohlen werden soll. Die Approximation der beiden gesuchten Werte ist gegeben durch die entsprechenden Elemente von R_k, hier notiert als $\hat{r}_{u2}$ und $\hat{r}_{u4}$. Nach (4.1) erfolgt die Berechnung der Elemente durch folgende Skalarprodukte:

$$\begin{array}{c} \\ N_1 \\ \vdots \\ N_u \\ \vdots \\ N_m \end{array}
\overbrace{\begin{bmatrix} F_1 & F_2 & F_3 & F_4 & F_5 & F_6 \dots\dots F_n \\ 1 & 0 & 2 & 0 & 5 & 0 \cdots 0 \end{bmatrix}}^{R}
\times
\overbrace{\begin{bmatrix} F_1 & F_2 & F_3 & F_4 & F_5 \dots F_n \\ & \alpha_{12} & & \alpha_{14} & \\ & \alpha_{22} & & \alpha_{24} & \\ & \alpha_{32} & & \alpha_{34} & \\ & \alpha_{42} & & \alpha_{44} & \\ & \alpha_{52} & & \alpha_{54} & \\ & \vdots & & \vdots & \\ & \alpha_{n2} & & \alpha_{n4} & \end{bmatrix}}^{V_k V_k^T}
\begin{array}{c} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ \vdots \\ F_n \end{array}.$$

Wir erhalten also

$$\hat{r}_{u2} = 1 \cdot \alpha_{12} + 2 \cdot \alpha_{32} + 5 \cdot \alpha_{52}$$
$$\hat{r}_{u4} = 1 \cdot \alpha_{14} + 2 \cdot \alpha_{34} + 5 \cdot \alpha_{54}.$$

Wie in (4.2) gezeigt wurde, repräsentiert α_{ij} die Ähnlichkeit zwischen Film F_i und F_j.

Damit berechnet sich die Approximation von PureSVD für einen Nutzer N_u und Film F_j aus der gewichteten Summe der skalierten Kosinus-Ähnlichkeiten zwischen F_j und allen bereits bewerteten Filmen von N_u bezüglich der latenten Merkmale, wobei die Gewichtungen den jeweiligen abgegebenen Bewertungen

[2] Da die latenten Merkmale dementsprechend stark oder weniger stark ausgeprägt sind.

entsprechen. Informal ausgedrückt: PureSVD empfiehlt die Filme, die den bereits bewerteten am meisten ähneln, abhängig davon wie diese bewertet wurden.

Mit diesem vertieften Verständnis der Funktionsweise von PureSVD kann zum Programmierteil übergegangen werden.

4.3. Implementierung von PureSVD

Wir verwenden das MovieLens-Datenset [HK15], genauer: Das `ml-latest-small`, welches ca. 100 000 Bewertungen von 610 Nutzern über knapp 10 000 Filme enthält.[3] Das Datenset besteht aus zwei `.csv`-Dateien, die jeweils Bewertungen anhand einer `userId` und `movieId` enthalten sowie den entsprechenden Filmnamen zur `movieId` bereitstellen. Die zentralen Teile des Programmcodes werden im Folgenden Schritt für Schritt nachvollzogen, wobei der vollständige Code in Anhang B unter Code B.2 zu finden ist.

Zunächst erstellen wir die Nutzer-Item-Matrix R, indem wir die Daten der `.csv`-Dateien mit der Bibliothek `pandas` einlesen und unbekannte Werte mithilfe von `fillna(0)` durch den Wert 0 ersetzen:

```
# Daten einlesen
ratings = pd.read_csv("programming_python/ratings.csv", header=0)
movies = pd.read_csv("programming_python/movies.csv", header=0)

# Unrelevante Daten entfernen
ratings = ratings.drop(columns=["timestamp"])
movies = movies.drop(columns="genres")

# Nutzer-Item-Matrix erstellen
user_item_table = ratings.pivot(
    index="userId", columns="movieId", values="rating"
).fillna(0)
R = user_item_table.values
```

Dabei sei angemerkt, dass die `user_item_table` einer Tabelle entspricht, bei der die Spalten durch die `movieIds` und die Zeilen durch die `userIds` gegeben sind. Die Spaltennummern entsprechen aber nicht den `movieIds`, da `ml-latest-small` nicht alle Filme enthält und die `movieIds` dementsprechend nicht fortlaufend nummeriert sind. Für die `userId` gilt dies nicht.

Wir können nun die trunkierte SVD von R mithilfe der Funktion `svds` aus der `SciPy`-Bibliothek berechnen:

[3] Zu finden unter: https://grouplens.org/datasets/movielens/latest/

```
U, S, Vt = svds(R, k=10)
Sigma = np.diag(S)

prediction_matrix = U @ Sigma @ Vt
```

Es wurden dabei nur die ersten 10 latenten Merkmale beachtet, womit die `prediction_matrix` der Matrix $R_{k=10}$ entspricht.

Als nächster Schritt wird für eine gegebene `userId` die entsprechende Zeile der `prediction_matrix` absteigend sortiert:

```
sorted_predictions = np.argsort(-prediction_matrix[user_idx])
```

Zuletzt werden die Spaltennummern der ersten `num_recom` (noch nicht geschauten) Filme aus den `sorted_predictions` als `recommended_movies_indices` gespeichert:

```
unwatched_indices = np.where(R[user_idx] == 0)[0]
recommended_movies_indices = [
    int(movie) for movie in sorted_predictions if movie in
    ↪ unwatched_indices
][:num_recom]
```

Die Ausgabe der Top 5 Empfehlungen für Nutzer 3 sähe nach einer Zuordnung der `recommended_movies_indices` zu den entsprechenden Filmnamen dann wie folgt aus:

```
Die Top 5 Empfehlungen für Nutzer 3 sind ['Aliens (1986)', 'Star Wars:
↪ Episode V - The Empire Strikes Back (1980)', 'Star Wars: Episode
↪ IV - A New Hope (1977)', 'Terminator, The (1984)', 'Star Wars:
↪ Episode VI - Return of the Jedi (1983)']
```

Es lässt sich beobachten, dass bei diesem Nutzer latente Merkmale, die mit dem Genre „Sci-Fi“ in Verbindung stehen wahrscheinlich besonders stark ausgeprägt sind.

Das Modell kann zusätzlich durch das in Definition 4.2 eingeführte Maß der Kosinus-Ähnlichkeit erweitert werden. Das Ziel ist, zu einem gewünschten Film die ähnlichsten Filme bezüglich der latenten Merkmale zu ermitteln. In der Praxis könnten dementsprechend nach dem Schauen eines Films weitere, ähnliche Filme vorgeschlagen werden. Dafür wird zunächst die Kosinus-Ähnlichkeit implementiert:

```
def cosine_similarity(v, u):
    return (v @ u) / (np.linalg.norm(v) * np.linalg.norm(u))
```

Anschließend werden die Kosinus-Ähnlichkeiten zwischen dem Spaltenvektor der gegebenen `movieId` von V_k^T und der restlichen Filme berechnet und als `similarities` gespeichert:

```
movie_vector = Vt[:, movie_idx]
similarities = np.array(
    [cosine_similarity(Vt[:, i], movie_vector) for i in
     ↪ range(Vt.shape[1])]
)
```

Die `similarities` werden nun absteigend sortiert, wobei die gegebene `movieId` selbst ausgeschlossen wird:

```
similar_movie_indices = np.argsort(-similarities)[1 : num_similar + 1]
```

Indem wie zuvor die `similar_movie_indices` den Filmnamen zugeordnet werden, erhalten wir beispielsweise folgenden Output:

```
Die Top 5 ähnlichsten Filme zu Film-ID 79132 (Inception (2010)) sind:
 ↪  ['Inglourious Basterds (2009)', 'Dark Knight, The (2008)', 'Isle
 ↪  of Dogs (2018)', 'Dark Knight Rises, The (2012)', 'Shutter Island
 ↪  (2010)']
```

Der vorgestellte Code zeigt, dass bereits mit einer simplen Implementierung sinnvolle Empfehlungen generiert werden können, die ausschließlich auf fundamentalen mathematischen Konzepten basieren. Zusammen mit der Hauptkomponentenanalyse wurden damit in dieser Arbeit zwei verschiedene, weit verbreitete Anwendungsgebiete der Singulärwertzerlegung ausführlich erläutert und es kann zum Fazit übergegangen werden.

5 Fazit

Das Ziel dieser Arbeit war zu zeigen, dass vermeintlich abstrakte und theoretisch aussehende mathematische Konzepte weitreichende Anwendung in der Praxis finden. Zu diesem Zweck wurde die Singulärwertzerlegung als eine vielseitige Methode der linearen Algebra vorgestellt. Obwohl sie in ihrer Beschreibung simpel erscheint — die Zerlegung einer beliebigen Matrix in eine Drehung, Skalierung und erneute Drehung —, besitzt sie Eigenschaften, die in diversen Anwendungsgebieten von großem Nutzen sind. Besonders hervorzuheben ist die Aussage des Eckart-Young-Satzes, wonach die trunkierte SVD die beste Rank-k-Approximation einer Matrix bietet. Zusammen mit ihrer numerischen Stabilität macht sie das zu einem wichtigen Werkzeug, insbesondere im aktuellen Kontext von künstlicher Intelligenz und Big Data, wo umfangreiche Datenmengen effizient verarbeitet und reduziert werden müssen.

Die in dieser Arbeit diskutierten Anwendungen — die Hauptkomponentenanalyse und Empfehlungssysteme — verdeutlichen bereits die praktische Relevanz der SVD. Weiterführend wäre es allerdings interessant, sie mit Methoden anderer Disziplinen zu verbinden, um komplexere Themenfelder kennenzulernen. Anbieten dafür würde sich eine Kombination mit *Deep Learning* (einem Teilbereich des maschinellen Lernens), wie beispielsweise in [Día+24] beschrieben wird.

Wir haben uns in dieser Arbeit ausschließlich mit reellen Matrizen beschäftigt. Jedoch existiert die Singulärwertzerlegung auch für Matrizen mit komplexen Elementen, wobei die zugrundeliegende Theorie weitgehend analog bleibt. Dies eröffnet zusätzliche Möglichkeiten für diverse Verwendungsgebiete, insbesondere in Bereichen wie der Quantenmechanik [MP12] oder der Kommunikationstechnik [TV05]. Ein vertiefender Blick auf die Anwendung der SVD in diesen Kontexten könnte eine spannende Richtung für zukünftige Untersuchungen darstellen.

Literaturverzeichnis

[Che13] Cheruvil Thomas, V. *Answer to "Why Does the Rank of the Design Matrix X Equal the Rank of X'X?"* Cross Validated. 2013. URL: https://stats.stackexchange.com/a/49897 (Stand: 22. 03. 2025).

[Che20] Chen, G. *Lecture 5: Singular Value Decomposition (SVD)*. San José State University, 2020. URL: https://www.sjsu.edu/faculty/guangliang.chen/Math253S20/lec5svd.pdf (Stand: 20. 01. 2025).

[CJ09] Cadima, J. & Jolliffe, I. T. „On Relationships Between Uncentred And Column-Centred Principal Component Analysis". In: *Pakistan Journal of Statistics* 25.4 (2009), S. 473–504.

[CKT10] Cremonesi, P., Koren, Y. & Turrin, R. „Performance of Recommender Algorithms on Top-N Recommendation Tasks". In: *Proceedings of the Fourth ACM Conference on Recommender Systems*. RecSys '10. New York, NY, USA: Association for Computing Machinery, 2010, S. 39–46.

[Cra22] Crawford, T. *Oxford Linear Algebra: Spectral Theorem Proof*. 2022. URL: https://tomrocksmaths.com/2022/11/18/oxford-linear-algebra-spectral-theorem-proof/ (Stand: 02. 01. 2025).

[Día+24] Díaz-Morales, P., Corrochano, A., López-Martín, M. & Le Clainche, S. „Deep Learning Combined with Singular Value Decomposition to Reconstruct Databases in Fluid Dynamics". In: *Expert Systems with Applications* 238.B (2024).

[Fra11] Frazzoli, E. *Dynamic Systems And Control. Lecture 4: Singular Values*. Massachusetts Institute of Technology, 2011. URL: https://ocw.mit.edu/courses/6-241j-dynamic-systems-and-control-spring-2011/resources/mit6_241js11_lec04/ (Stand: 09. 03. 2025).

[HK15] Harper, F. M. & Konstan, J. A. „The MovieLens Datasets: History and Context". In: *ACM Trans. Interact. Intell. Syst.* 5.4 (2015), 19:1–19:19.

[Hsu16] Hsu, D. *Machine Learning Theory. COMS 4772. Topic 5: Principal Component Analysis.* Columbia University, 2016. URL: https://www.cs.columbia.edu/~djhsu/AML/lectures/notes-pca.pdf (Stand: 25. 02. 2025).

[Joh21] Johnston, N. *Advanced Linear and Matrix Algebra*. Cham: Springer, 2021.

[KBV09] Koren, Y., Bell, R. & Volinsky, C. „Matrix Factorization Techniques for Recommender Systems“. In: *Computer* 42.8 (2009), S. 30–37.

[MP12] Martin, C. D. & Porter, M. A. „The Extraordinary SVD“. In: *The American Mathematical Monthly* 119.10 (2012), S. 838–851.

[Nik+19] Nikolakopoulos, A. N., Kalantzis, V., Gallopoulos, E. & Garofalakis, J. D. „EigenRec: Generalizing PureSVD for Effective and Efficient Top-N Recommendations“. In: *Knowledge and Information Systems* 58.1 (2019), S. 59–81.

[NM23] Ng, A. & Ma, T. *Machine Learning. CS229 Lecture Notes.* Stanford University, 2023. URL: https://cs229.stanford.edu/main_notes.pdf (Stand: 23. 02. 2025).

[Räs21] Räsch, T. *Lineare Algebra (Version 3. 3. 2)*. Universität Bonn, 2021. URL: https://www.math.uni-bonn.de/people/raesch/Papers_and_Notes/ThR_LA_Skript.pdf (Stand: 02. 01. 2025).

[Str09] Strang, G. *Introduction to Linear Algebra*. 4. Aufl. Wellesley: Wellesley - Cambridge Press, 2009.

[TV05] Tse, D. & Viswanath, P. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[Zha22] Zhang, Z. *Linear Algebra for Data Science. Lecture 4. Singular Value Decomposition (SVD)*. Carnegie Mellon University, 2022. URL: https://www.math.cmu.edu/users/zechengz/fall_2022_la/lec4_2p.pdf (Stand: 07. 03. 2025).

Abbildungen

A

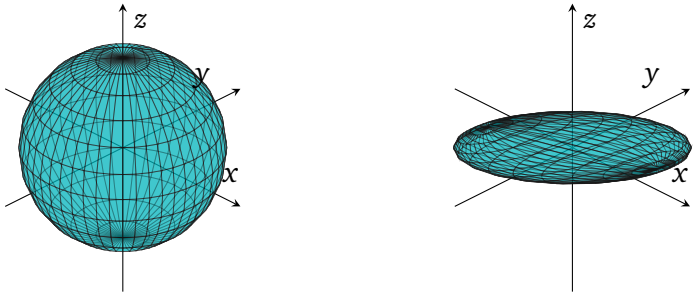


Abb. A.1. Wirkung von A auf die Einheitssphäre

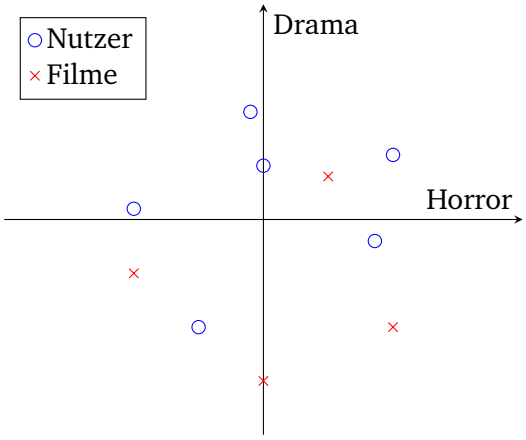
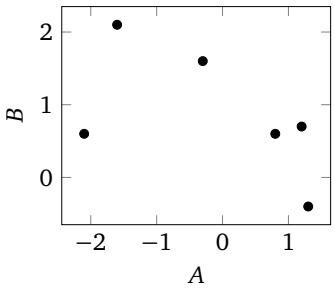


Abb. A.2. Gemeinsamer latenter Raum von Nutzern und Filmen

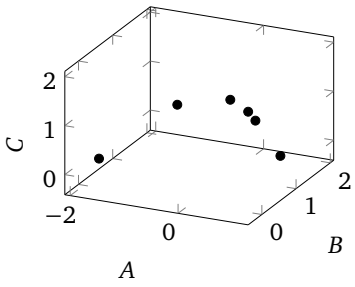
(a) $n = 2$

	Merkmale	
Weine	A	B
Wein 1	1,3	-0,4
Wein 2	-1,6	2,1
Wein 3	1,2	0,7
Wein 4	-2,1	0,6
Wein 5	0,8	0,6
Wein 6	-0,3	1,6



(b) $n = 3$

	Merkmale		
Weine	A	B	C
Wein 1	1,3	-0,4	1,9
Wein 2	-1,6	2,1	0,2
Wein 3	1,2	0,7	0,4
Wein 4	-2,1	0,6	-0,2
Wein 5	0,8	0,6	1,1
Wein 6	-0,3	1,6	0.8



(c) $n = 4$

	Merkmale			
Wein	A	B	C	D
Wein 1	1,3	-0,4	1,9	0.7
Wein 2	-1,6	2,1	0,2	0.9
Wein 3	1,2	0,7	0,4	1.3
Wein 4	-2,1	0,6	-0,2	0.5
Wein 5	0,8	0,6	1,1	-0.8
Wein 6	-0,3	1,6	0.8	-1.3

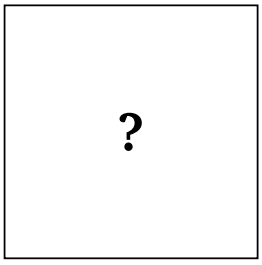


Abb. A.3. Darstellung von Daten in verschiedenen Dimensionen

Programmcode

B

```python
# lauchli.py

import numpy as np


# Funktion zur Erstellung der Läuchli-Matrix für gegebenes epsilon
def laeuchli_matrix(epsilon):
    L = np.zeros((4, 3))
    L[0, :] = 1
    np.fill_diagonal(L[1:, :], epsilon)
    return L


# Verschiedene Werte für epsilon
epsilons = [1e-3, 1e-10, 1e-20]

for epsilon in epsilons:
    L = laeuchli_matrix(epsilon)

    # Eigenwerte von L^T L
    eigvals = np.linalg.eigvalsh(L.T @ L)[::-1]

    # Quadrate der Singulärwerte von L
    singular_values = np.linalg.svd(L, compute_uv=False)
    singular_values_squared = singular_values**2

    # Output
    print(f"Epsilon: {epsilon}")
    print("Eigenwerte vs. Quadrate der Singulärwerte:")
    for ev, sv in zip(eigvals, singular_values_squared):
        print(f"{ev:.3e}          {sv:.3e}")
    print("-")
```

Code B.1. Berechnungsunterschiede Läuchli-Matrix

```python
# recommender.py

import pandas as pd
import numpy as np
from scipy.sparse.linalg import svds

# Daten: "ml-latest-small.zip" von
 ↪  https://grouplens.org/datasets/movielens/latest/

# Daten einlesen
ratings = pd.read_csv("programming_python/ratings.csv", header=0)
movies = pd.read_csv("programming_python/movies.csv", header=0)

# Unrelevante Daten entfernen
ratings = ratings.drop(columns=["timestamp"])
movies = movies.drop(columns="genres")

# Nutzer-Item-Matrix erstellen
user_item_table = ratings.pivot(
    index="userId", columns="movieId", values="rating"
).fillna(0)
R = user_item_table.values

# Trunkierte Rang-k SVD
U, S, Vt = svds(R, k=10)
Sigma = np.diag(S)
prediction_matrix = U @ Sigma @ Vt


# Generieren der Empfehlungen zu einer userId
def recommender(R, prediction_matrix, num_recom, user):
    user_idx = user - 1  # userId startet bei 1
    sorted_predictions = np.argsort(-prediction_matrix[user_idx])
    unwatched_indices = np.where(R[user_idx] == 0)[0]
    recommended_movies_indices = [
        int(movie) for movie in sorted_predictions if movie in
         ↪  unwatched_indices
    ][:num_recom]
    recommended_movies_ids = [
        int(user_item_table.columns[idx]) for idx in
         ↪  recommended_movies_indices
    ]  # Entsprechende movieIds zu den Spaltennummern
```

```
    recommended_movies = [
        movies.loc[movies["movieId"] == id, "title"].iloc[0]
        for id in recommended_movies_ids
    ]  # Entsprechende Filmnamen zu den movieIds
    return recommended_movies


# Kosinus-Ähnlichkeit
def cosine_similarity(v, u):
    return (v @ u) / (np.linalg.norm(v) * np.linalg.norm(u))


# Generieren der ähnlichsten Filme zu einer movieId
def similar_movies(movie_id_similar, num_similar):
    movie_idx = np.where(user_item_table.columns ==
    ↪  movie_id_similar)[0][
        0
    ]  # Spaltennummer zu gegebener movieId
    movie_vector = Vt[:, movie_idx]
    similarities = np.array(
        [cosine_similarity(Vt[:, i], movie_vector) for i in
        ↪  range(Vt.shape[1])]
    )
    similar_movie_indices = np.argsort(-similarities)[1 : num_similar
    ↪  + 1]
    similar_movie_ids = [
        int(user_item_table.columns[idx]) for idx in
        ↪  similar_movie_indices
    ]  # Entsprechende movieIds zu den Spaltennummern
    similar_movie_titles = [
        movies.loc[movies["movieId"] == id, "title"].iloc[0] for id in
        ↪  similar_movie_ids
    ]  # Entsprechende Filmnamen zu den movieIds
    return similar_movie_titles


num = 5  # Gewünschte Anzahl an Empfehlungen
usr = 3  # Gewünschte userId

# Output
print(
    "Die Top",
    num,
    "Empfehlungen für Nutzer",
```

```
    usr,
    "sind",
    recommender(R, prediction_matrix, num, usr),
)

num_sim = 5  # Gewünschte Anzahl an ähnlichen Filmen
mov_sim = 79132  # Gewünschte movieId

# Output
print(
    "Die Top",
    num_sim,
    "ähnlichsten Filme zu Film-ID",
    mov_sim,
    "(" + movies.loc[movies["movieId"] == mov_sim, "title"].iloc[0] +
    ↪  ")",
    "sind:",
    similar_movies(mov_sim, num_sim),
)
```

Code B.2. PureSVD in Python