TECHNISCHE UNIVERSITÄT DRESDEN

FAKULTÄT INFORMATIK
INSTITUT FÜR SOFTWARE- UND MULTIMEDIATECHNIK
PROFESSUR FÜR COMPUTERGRAPHIK UND VISUALISIERUNG
PROF. DR. STEFAN GUMHOLD

Großer Beleg

Dimensionsreduktion in der computergraphischen Fluidsimulation

Ludwig Schmutzler
(Mat.-Nr.: 3027133)

Betreuer: Dipl.-Phys. Niels v. Festenberg

Dresden, 23. August 2010

Aufgabenstellung

Thema:

Dimensionsreduktionsverfahren in der computergraphischen Fluidsimulation

Zielstellung:

Ziel dieser Arbeit ist es, verschiedene Verfahren zur Dimensionsreduktion auf die interaktive Animation von Fluidgrenzflächen anzuwenden und zu bewerten. Die Arbeit von A. Treuille et al. *Model Reduction for Real-time Fluids* (2006) bildet dabei das Vorbild. Zur Auswertung stehen Zeitreihen des Lehrstuhlfluidsimulators zur Verfügung. Im Einzelnen sind folgende Teilaufgaben zu bearbeiten:

- Literaturrecherche über Dimensionsreduktionsverfahren in der Computergraphik mit Ausblicken in die mathematischen Grundlagen
- Anwendung der linearen Hauptkomponentenanalyse auf verschiedene Fluidsimulationszeitreihen in Form von Level-set-Rohdaten
- Anwendung wenigstens einer nichtlinearen Dimensionsreduktionsmethode (z.B. Kern-PCA oder Diffusionsabbildung)
- Erprobung und prototypische Implementierung der interaktiven Simulation in reduzierten Systemen per Galerkin-Projektion analog Treuille et al.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tag dem Prüfungsausschuss der Fakultät Informatik eingereichte Belegarbeit zum Thema:

Dimensionsreduktion in der computergraphischen Fluidsimulation

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Dresden, den 23. August 2010

Ludwig Schmutzler

Inhaltsverzeichnis

1 Einleitung

Motivation und Zielstellung

In der Computergraphik werden nicht selten komplexe hochdimensionale Beschreibungen von Naturphänomenen hervorgebracht, welche einerseits von anspruchsvollen mathematisch-physikalisch motivierten Ansätzen und Modellen herrühren, andererseits durch das Abtasten und Diskretisieren der realen Natur bedingt sind. Durch die Flut an Daten fallen hierbei zum Teil riesige Informationsmengen an. Oft jedoch liegen der gefundenen hochdimensionalen Datenmenge viel einfachere Strukturen zugrunde, mit welcher sich dieselben Vorgänge beschreiben lassen.

Die Methoden der Dimensionsreduktionsverfahren finden ebenjene Strukturen, woraufhin eine Vereinfachung der Daten vorgenommen werden kann. Ein bekanntes mathematisches Verfahren stellt hierbei die Hauptkomponentenanalyse dar, im englischen Sprachraum als *Principal Components Analysis*, kurz **PCA**, bezeichnet, welche Nebenabhängigkeiten in Datenstrukturen findet. Dieses Verfahren wurde um 1930 entwickelt, das innewohnende Potenzial konnte allerdings erst mit der Verfügbarkeit von geeigneten Großrechnern in vollem Umfang ausgeschöpft werden.

In der Computergrafik findet die Hauptkomponentenanalyse unter anderem in der Fluidsimulation Anwendung. So ist es mittlerweile möglich, rechenintensive Fluidsimulationen nicht nur echtzeitfähig im Sinne der graphischen Visualisierung, sondern auch durch Echtzeit-Interaktion für den Nutzer erforschbar zu machen. Dies wird erreicht, indem die notwendigen Berechnungen der Navier-Stokes-Gleichungen sowie der Nutzereingaben vollständig in einem reduzierten Raum stattfinden (vgl. [TLP06]).

Die vorliegende Arbeit knüpft an diesem Punkt an und untersucht die PCA im Hinblick auf ihre Eigenschaften als Dimensionsreduktionsverfahren bei der Anwendung auf Fluidgrenzflächen. Dabei kann sie einerseits direkt zur Kompression der Daten verwendet werden, andererseits können mit ihr Strukturen in den Daten gefunden werden, welche erlauben, ein Simulationsmodell zu definieren, das nutzerabhängige Animationen ermöglicht. Im Gegensatz zu [TLP06] besteht hierbei die Schwierigkeit, dass anstelle eines Vektorfeldes lediglich ein skalares Feld in Form von *level sets* als Lösung einer Offline-Fluidsimulation zur Verfügung steht, um zugrundeliegende Strukturen zu finden; d.h. insbesondere, dass die Vorgänge, welche zur Ausprägung der entsprechenden Fluidoberfläche führten, unbekannt sind.

Neben der Anwendung der PCA als lineares Verfahren wird des Weiteren der Fragestellung nachgegangen, inwieweit nichtlineare Verfahren zur Daten-Analyse geeignet sind und ob durch sie womöglich noch bessere Ergebnisse erreicht werden können. Zu diesem Zweck wurde die sogenannte *Kern-PCA* untersucht, die das PCA-Verfahren mit den Mitteln der nichtlinearen Kernmethoden kombiniert.

Gliederung der Arbeit

Zunächst werden in Kapitel 2 verwandte Arbeiten thematisiert, welche sich mit Dimensionsreduktionsverfahren in der computergraphischen Fluidsimulation beschäftigen. Kapitel 3 widmet sich den methodischen Grundlagen der linearen und der nichtlineare Hauptkomponentenanalyse. Dabei wird detailliert auf die Algorithmen sowie die mathematischen Aspekte eingegangen, ebenso werden Probleme und deren Lösungsansätze diskutiert. Den Abschluss dieses Kapitels bildet die Beschreibung des entwickelten Simulationsmodelles.

Begleitend zur vorliegenden Arbeit wurde eine Analysesoftware entwickelt, deren Funktionalitäten und Hilfsstrukturen in Kapitel 4 beschrieben werden. In den darauffolgenden Kapiteln 5 und 6 werden die Ergebnisse der Untersuchungen an den Testzeitreihen vorgestellt und diskutiert. Kapitel 7 schließt die Arbeit mit der Zusammenfassung.

2 Verwandte Arbeiten

Die Hauptkomponentenanalyse ist als Dimensionsreduktionsverfahren in der Fluidsimulation nicht unbekannt und findet seit den siebziger Jahren des zwanzigsten Jahrhunderts Anwendung, z.B. durch J. Lumley [Lum70], L. Sirovich [Sir87] oder P. Holmes [HLB96]. Sie ist auch unter der Bezeichnung *Proper Orthogonal Decomposition*, kurz *POD* (vgl. [Row05] und [TLP06]), bekannt. Bei diesen Verfahren wird u.a. die sogenannte Galerkin-Projektion angewandt, um neben den Datenvektoren auch die Zeitentwicklungsgleichungen des Fluidmodells in einen reduzierten Raum zu überführen und folglich mit weniger Unbekannten zu rechnen. Ein Vorteil der POD-Methode ist u.a. der Wegfall der Behandlung von Randbedingungen, da sie durch die Vollsimulation a priori vorliegen und damit implizit in das reduzierte Modell einfließen. Weiterhin sind alle Berechnungen unabhängig von der Größe des Simulationsraumes, sondern nur von der Größe der Basis, welche durch die Hauptkomponenten gebildet wird.

Zur Reduzierung der Rechenkomplexität bei der Berechnung der Hauptkomponenten, im Duktus der POD-Methode auch als *POD modes* bezeichnet (vgl. [Row05]), führte Sirovich die sogenannte *Snap Shot*-Methode ein, um ein reduziertes Modell nicht mehr aus der gesamten Fluidsimulation, sondern aus einer Reihe von Momentanzuständen zu errechnen.

Diesen Ansatz verfolgte A. Treuille [TLP06] in seiner Arbeit und präsentiert ein um eine interaktive Komponente erweitertes Modell: Es können anwendergesteuerte Hindernisse in Echtzeit im Fluidstrom simuliert werden. Treuille koppelt das reduzierte Modell des gesamten Fluidstromes mit einem reduzierten Modell, welches aus den Fluidbewegegungen in der unmittelbaren Nähe des interaktiven Objektes hervorgeht und wiederum mithilfe der PCA und der *Snap Shot*-Methode gewonnen wurde, so, dass die *No-Slip*-Bedingung am Objektrand effizient berechnet werden kann. Dazu führt er Operatoren ein, welche Rotation und Translation der Bezugssysteme ineinander überführen sowie einen *Speed*-Operator zur Berechnung der Normalengeschwindigkeiten an allen abgetasteten Oberflächenpunkten. In einem ersten Schritt, dem *feed foward*, werden die Bereiche des globalen Fluid-Vektorfeldes in der lokalen Nähe des interaktiven Objektes in dessen Koordinatensystem umgerechnet und die *No-Slip*-Bedingung erzwungen. Im zweiten Schritt, dem *feed backward* erfolgt die Berechnung der Kräfte, die durch die Nutzer-Interaktion mit dem Objekt auf das Fluid einwirken, welche anschließend wieder vom lokalen Objekt-Bezugssytem in das globale Koordinatensystem umgerechnet werden.

Zur Effizienzsteigerung seines Interaktionsmodells führt Treuille eine Vorberechnung der *feed foward*- und *feed backward*-Matrizen für diskretisierte Orientierungen, d.h. Rotation und Translation, durch, wobei der Speicheraufwand für diese Matrizen quadratisch von der Diskretisierungsschrittweite abhängt. Durch die Offline-Berechnung kann nunmehr sogar die Kopplung zwischen Fluid und interaktivem Objekt vollständig im reduzierten Raum berechnet werden.

Ansätze, welche die PCA/POD-Methode mit Algorithmen verbinden, die über lange Simulationszeiten eine höhere Stabilität und akkuratere Ergebnisse als die *Snap Shot*-Methode liefern, sind u.a. durch C.W. Rowley [Row05] beschrieben. Er kombiniert die Methode der *balanced truncation* mit der POD-Methode zur *balanced POD*. Eine approximierende Variante der PCA sowie der nichtlinearen Kern-PCA (vgl. Kapitel 3.3) wird von Shawn Martin [Mar06] vorgestellt und auf Taylor-Couette-Flüsse angewandt.

3 Methodik

3.1 Oberflächen in der Fluidsimulation: Level Sets

Die Repräsention einer Oberfläche als sogenanntes *level set* ist eine weitverbreitete Darstellungsform und eng verwandt mit der mathematischen Definition für *implizite Flächen*[1].

Im Folgenden werden diese beiden Darstellungsformen kurz erklärt. Zunächst sei die Darstellung einer impliziten Fläche beschrieben:

Es wird eine Funktion $\Phi(\underline{x}) : \mathbb{R}^N \rightarrow \mathbb{R}$ eingeführt, welche an allen Punkten $\underline{x}$ ($\underline{x} \in \mathbb{R}^n$) des betrachteten Raumes definiert sei. Sie wird weiterhin so gewählt, dass durch sie, abhängig von ihrem Funktionswert, drei Regionen Ω^+, Ω^- und $\delta\Omega$ erklärt werden können (vgl. [OF02]). Für alle Punkte $\underline{x}_+$, die außerhalb der impliziten Fläche liegen, soll der Funktionswert von $\Phi(\underline{x}_+)$ größer Null sein. Die Menge dieser Punkte ist $\Omega^+ \equiv \{\underline{x}|\Phi(\underline{x}) > 0\}$. Die Menge Ω^- der Punkte, welche innerhalb der impliziten Fläche liegen, ist analog mit $\Omega^- \equiv \{\underline{x}|\Phi(\underline{x}) < 0\}$ beschrieben. Die implizite Fläche $\delta\Omega$ ist genau dort definiert, wo der Funktionswert von Φ gleich Null wird, d.h. $\delta\Omega \equiv \{\underline{x}|\Phi(\underline{x}) = 0\}$.[2]

Ein *level set* ist nun die Menge von Punkten, für welche die Funktion $\Phi(\underline{x})$ ein und denselben konstanten Funktionswert Δ liefert. Sei diese Menge mit Γ_Δ bezeichnet, so gilt (analog $\delta\Omega$):

$$\Gamma_\Delta \equiv \{\underline{x}|\Phi(\underline{x}) = \Delta\} \tag{3.1}$$

Man sieht, dass sich das *level set* Γ_Δ ebenso als implizite Fläche darstellen lässt. Dazu wird die implizite Funktion $\Phi_\Delta(\underline{x}) = \Phi(\underline{x}) - \Delta$ gewählt. Sie hat als Nullstellenmenge $\delta\Omega_\Delta$ genau die Menge der Punkte, welche durch das level set Γ_Δ beschrieben sind.

Weitere Bezeichnungen für *level set* sind u.a. *Isokonturlinien* im zweidimensionalen bzw. *Isoflächen* im dreidimensionalen Fall.

[1] Anm.: Beide Darstellungsformen unterscheiden sich semantisch, lassen sich jedoch gegenseitig ineinander überführen.

[2] Während $\Phi(\underline{x})$ im n-dimensionalen Raum definiert ist, so hat $\delta\Omega$ die Dimension $(n-1)$ (vgl. [OF02]).

3.2 Die Hauptkomponentenanalyse (PCA)

3.2.1 Die historische Entwicklung der Hauptkomponentenanalyse

Die Methode der Hauptkomponentenanalyse ist ein wohlbekanntes Verfahren der multivariaten Statistik, um komplexe Daten zu vereinfachen und zu restrukturieren (vgl. [Jol02]). Ihre Ursprünge reichen bis zum Ende des 19. Jahrhunderts zurück, als u.a. durch E. Beltrami und M.E.C. Jordan Methoden zur Singulärwertzerlegung in einer Form entwickelt wurden, wie sie auch in der heutigen PCA Anwendung finden. Die ersten Beschreibungen des Verfahrens, welches heute als Hauptkomponentenanalyse bezeichnet wird, geht auf die Arbeiten von Karl Pearson (1901) und Harold Hotelling (1933) zurück, wobei beide unterschiedliche Ansätze verwendeten – Pearson versuchte Mengen von Punkten in p-dimensionalen Räume durch Geraden und Ebenen anzunähern, während Hotelling ausgehend von der Faktorenanalyse zur PCA kam. Pearson erklärte, dass seine Methoden einfach auf entsprechende numerische Probleme hoher Dimension angewandt werden können, auch wenn die Berechnungen per Hand für vier oder mehr Unbekannte mühsam würden – dies war lange vor der allgemeinen Verfügbarkeit von elektronischen Rechenanlagen. Der Begriff *Hauptkomponente* geht auf Hotelling zurück: Seine Methode orientierte sich an der Faktorenanalyse, welche von dem Psychologen Charles Spearman im beginnenden 20. Jahrhundert entwickelt und durch den der Begriff des *Faktors* geprägt wurde. Um seine Methode abzugrenzen und um weiterer Konfusion mit dem mathematischen Begriff *Faktor* aus dem Weg zu gehen, führte Hotelling den Begriff *Komponente* ein. Er wählte sie so, dass sie je einen akkumulativen Anteil an der Summe der gesamten Varianz aller ursprünglichen Variablen leisteten und nannte die so erhaltenen Vektoren *Hauptkomponenten*. Das Verfahren betitelte er mit *„method of principal components“* – Methode der Hauptkomponenten. Hotelling beschrieb weiterhin, wie die Hauptkomponenten mithilfe der Potenzmethode gefunden werden können sowie deren geometrische Interpretation. In den unmittelbaren Jahren nach Hotellings Publikation wurden nur wenige verschiedene Anwendungen und Erweiterungen der PCA-Methode vorgenommen. Mit dem Vormarsch der elektronischen Rechner sollte sich dies ändern: Etwa 25 Jahre nach Hotellings Paper wurden explosionsartig Weiterentwicklungen und differenzierte Anwendungen der PCA vorangetrieben. Nun wurde das Potenzial deutlich, welches sich aus der Anwendung der PCA auf hochdimensionale Daten ergibt[3]. Als weitere wichtige Publikationen in den, bezüglich der PCA, aufblühenden 60-er Jahren des 20. Jahrhunderts seien die Arbeiten von T.W. Anderson [And63], C.R. Rao [Rao64], J.C. Gower [Gow66] und J.N.R. Jeffers [Jef67] genannt (vgl. [Jol02]).

Die Hauptkomponentenanalyse entwickelte sich zu einem etablierten Verfahren, welches heute in nahezu allen wissenschaftlichen Disziplinen ihre Anwendung findet. In der Computergraphik sind die Auf-

[3] Anm.: „Hochdimensional“ nicht mehr im Sinne Pearsons, sondern mit weit mehr als vier Unbekannten.

gabengebiete der PCA weit gestreut: Sie findet Anwendung in einfachen Berechnungsverfahren, z.B. beim Berechnen optimaler Boundingboxen (vgl. [DKKR07]), wie auch in komplexen Beleuchtungsrechnungen, z.B. beim Rendern komplexer Oberflächen mithilfe bidirektionaler Texturfunktionen (vgl. [MMK04]) oder zur Manipulation sowie Inter- und Extrapolation aufgezeichneter Motion Capture Daten (vgl. [GBT04]).

Synonym zum Terminus *Hauptkomponentenanalyse* werden auch die Begriffe *Karhunen-Loève-Transformation*, *Hotelling-Transformation* bzw. *Proper Orthogonal Decomposition* (vgl. Kap. 2) verwendet.

3.2.2 Grundlagen zur PCA

Im Folgenden werden die Ansätze und wesentlichen Schritte des PCA-Algorithmus beschrieben.

Die grundsätzliche Zielstellung der PCA ist die Transformation der Daten in ein geeignetes Koordinatensystem, in welchem sie im Idealfall redundanzfrei beschrieben werden können. Eine solche Redundanz ist insbesondere dann gegeben, wenn zwei Variablen eines Datensatzes einen Zusammenhang aufweisen, d.h. wenn sich der Anteil einer Variablen aus der jeweils anderen ableiten lässt. Die PCA transformiert die Datensätze so, dass genau diese Zusammenhänge minimiert und die Daten mit weniger Variablen dargestellt werden können.

Zu einer Menge von T mittelwertbereinigten Datensätzen $\vec{x}_t \in \mathbb{R}^N$ wird eine neue Basis $\mathbf{B} = \{\vec{b}_i | i = 0..N-1, N \in \mathbb{N}\}$ gefunden, deren Basisvektoren $\vec{b}_i \in \mathbb{R}^N$ orthonormal sind – sie werden die *Hauptkomponenten* genannt. Ein Datensatz lässt sich sodann als Linearkombination dieser I Basisvektoren darstellen:

$$\vec{x}_t = \sum_{i=0}^{N-1} a_i \cdot \vec{b}_i \tag{3.2}$$

Eine Reduktion der Dimension erfolgt durch die Vernachlässigung von Hauptkomponenten. Der Datenvektor $\vec{x}_t$ wird durch den Vektor $\tilde{\vec{x}}_t$ approximiert, wobei $\tilde{\vec{x}}_t$ aus einer Linearkombination der ersten M Basisvektoren hervorgeht:

$$\tilde{\vec{x}}_t = \sum_{i=0}^{M-1} a_i \cdot \vec{b}_i \qquad (M < I) \tag{3.3}$$

Das Ziel besteht darin, den mittleren quadratischen Fehler respektive den euklidischen Abstand zwischen dem Ursprungsdatensatz $\vec{x}_t$ und dem approximierenden Vektor $\tilde{\vec{x}}_t$ zu minimieren, d.h. es erfolgt eine Optimierung im Sinne der *Methode der kleinsten Quadrate*.

$$\min ||\tilde{\vec{x}}_t - \vec{x}_t||^2 \tag{3.4}$$

Bei der Hauptkomponentenanalyse wird dies erreicht, indem man die Basis $\mathbf{B}$ so wählt, dass die Richtungen der Hauptkomponenten gleich den Richtungen der größten Streuungen in den Datensätzen entsprechen. Als Streuungsmaß wird hierbei die Varianz verwendet. Das Minimierungsziel ist erfüllt, wenn die Hauptkomponenten entsprechend des zugehörigen Streuungsanteils sortiert werden. Genau dann wird $||\tilde{\vec{x}}_t - \vec{x}_t||^2$ minimal.

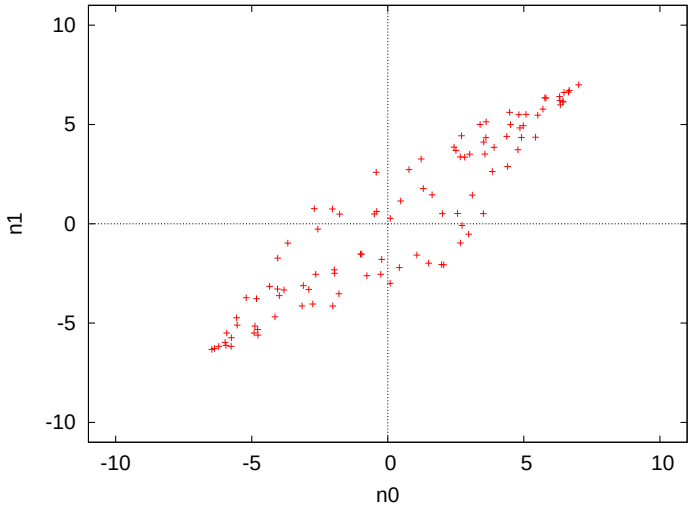

Abbildung 3.1: Plot von 100 Datensätzen $\vec{x}_t = (n_0, n_1)^T$ (vgl. [Jol02]).

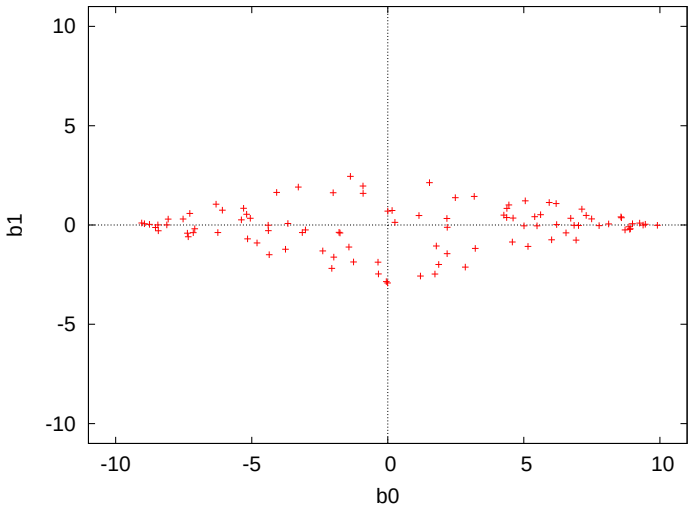

Abbildung 3.2: Plot der Daten aus Abb. 3.1 bezüglich ihrer Hauptkomponenten b_0 und b_1.

Die Abbildung 3.1 zeigt beispielhaft eine Menge von Datensätzen bestehend aus zwei stark korrelierten Variablen n_0 und n_1; beide besitzen ungefähr die gleiche Streuung. Die größte Streuung in den

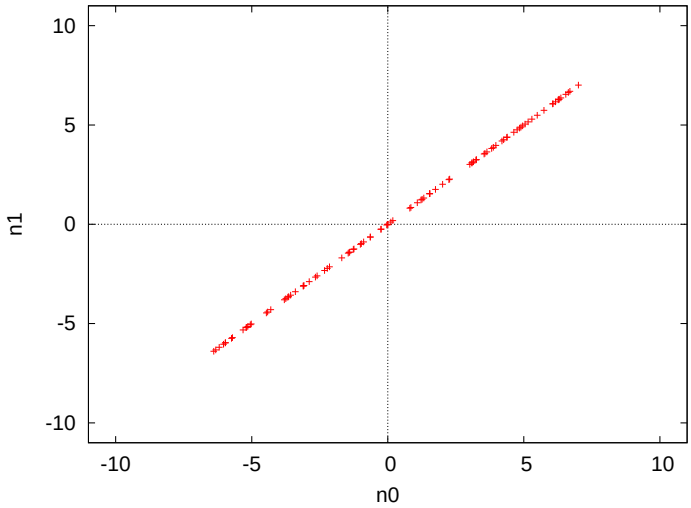

Abbildung 3.3: Plot der rekonstruierten Daten aus Abb. 3.1 mithilfe der ersten Hauptkomponente

Daten lässt sich jedoch in einer Richtung finden, welche schräg zu den Richtungen von n_0 und n_1 verläuft. Sie ist genau die Richtung der ersten Hauptkomponente b_0, orthogonal dazu ist die Richtung der zweiten Hauptkomponente b_1, welche einen geringeren Streuungsanteil hat. Abbildung 3.2 zeigt die Datensätze bezüglich ihrer Hauptkomponenten. Die Dimensionreduktion besteht nun darin, von einer Datenbeschreibung in Abhängigkeit zweier Variablen n_0 und n_1 auf eine Darstellung abhängig von einer Variablen b_0 überzugehen und die Richtung b_1 mit dem kleinsten Streuungsanteil zu verwerfen. Das Ergebnis nach der Rücktransformation in den ursprünglichen Raum der Eingabedaten ist eine lineare Korrelation zwischen n_0 und n_1 (vgl. Abb. 3.3).

3.2.3 Varianz und Kovarianz in der PCA

Die Varianz σ^2 ist ein Maß für die Streuung einer Variablen und Grundlage für das Optimierungsziel der PCA, da sie die Richtungen der Hauptkomponenten festlegt. Sie ist für eindimensionale Datensätze x_t ($t = 0..T-1$) definiert als:

$$\sigma^2 \equiv \frac{1}{T-1} \sum_{t=0}^{T-1} (x_t - \bar{x})^2 \tag{3.5}$$

Die Varianz lässt sich für mehrdimensionale Datenvektoren $\vec{x}_t \in \mathbb{R}^N$ analog definieren als:

$$\vec{\sigma}^2 \equiv \begin{pmatrix} \sigma_0^2 \\ \vdots \\ \sigma_n^2 \\ \vdots \\ \sigma_{N-1}^2 \end{pmatrix} \equiv \frac{1}{T-1} \sum_{t=0}^{T-1} \begin{pmatrix} (\vec{x}_{t,0} - \bar{x}_0)^2 \\ \vdots \\ (\vec{x}_{t,n} - \bar{x}_n)^2 \\ \vdots \\ (\vec{x}_{t,N-1} - \bar{x}_{N-1})^2 \end{pmatrix} \tag{3.6}$$

Hierbei bezeichnet $x_{t,n}$ die n-te Komponente des t-ten Datenvektors und $\bar{x}_n$ den Mittelwert der n-ten Komponente der Datenvektoren.

Im Folgenden sei $\vec{v}_n$ der Vektor aller Einträge der jeweils n-ten Komponente aus den T Datensätzen, d.h.

$$\vec{v}_n \equiv (x_{0,n}, x_{1,n}, \ldots, x_{t,n}, \ldots, x_{T-1,n}) \tag{3.7}$$

Sofern die Datensätze mittelwertbereinigt sind, lässt sie sich dann die Varianz σ_n^2 der n-ten Komponente übersichtlich als Skalarprodukt darstellen:

$$\sigma_n^2 \equiv \frac{1}{T-1} \langle \vec{v}_n, \vec{v}_n \rangle \tag{3.8}$$

Die Kovarianz ist eine Verallgemeinerung des Varianzbegriffes und beschreibt den linearen Zusammenhang zwischen zwei Komponenten $n1$ und $n2$. Sie wird analog zur Varianz definiert:

$$\sigma_{n1,n2}^2 \equiv \frac{1}{T-1} \langle \vec{v}_{n1}, \vec{v}_{n2} \rangle \tag{3.9}$$

Der Betrag von $\sigma_{n1,n2}^2$ ist ein Maß für die Stärke des linearen Zusammenhangs zwischen den Komponenten $n1$ und $n2$. Je höher dieser Betrag ist, desto größer ist der Zusammenhang. Das Vorzeichen von $\sigma_{n1,n2}^2$ beschreibt die Richtung des Zusammenhangs. Ein positives Vorzeichen bedeutet einen gleichsinnigen Zusammenhang: Hat eine Komponente $n1$ einen hohen Betrag, so hat die Komponente $n2$ ebenfalls einen hohen Betrag (vgl. Abb. 3.1). Ein negatives Vorzeichen bedeutet einen gegensätzlichen Zusammenhang: Hat eine Komponente $n1$ einen hohen Betrag, so hat die Komponente $n2$ einen niedrigen Betrag. Ist $\sigma_{n1,n2}^2 = 0$ erfüllt, so besteht kein linearer Zusammenhang zwischen den Komponenten $n1$ und $n2$.

Eine weitere mögliche Notationsform für die Kovarianz ist die Matrixschreibweise. Fasst man die Vektoren $\vec{v}_{n1}$ und $\vec{v}_{n2}$ als $(1 \times T)$-Matrizen auf, so lässt sich die Kovarianz als Matrixprodukt aufschreiben:

$$\sigma_{n1,n2}^2 \equiv \frac{1}{T-1} \mathbf{v_{n1}} \cdot \mathbf{v_{n2}}^T \tag{3.10}$$

Mithilfe dieser Darstellungsform kann nun von der Kovarianz zweier Komponenten auf die Kovarianzen ganzer Mengen von Komponenten verallgemeinert werden.

Zunächst werden die Datensätze in Matrizenschreibweise notiert. Die Menge der T Datensätze sei durch die Matrix $\mathbf{X}$ repräsentiert. Die Datenvektoren werden hierbei als Spaltenvektoren aufgefasst:

$$\begin{aligned}\mathbf{X} &\equiv (\vec{x}_0\ \vec{x}_1\ \ldots\ \vec{x}_t\ \ldots\ \vec{x}_{T-1}) \\ &= \begin{pmatrix} x_{0,0} & x_{1,0} & \cdots & x_{T-1,0} \\ x_{0,1} & x_{1,1} & \cdots & x_{T-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{0,N-1} & x_{1,N-1} & \cdots & x_{T-1,N-1} \end{pmatrix}\end{aligned} \tag{3.11}$$

Die Menge aller Kovarianzen können ebenfalls kompakt in einer Matrix zusammengefasst werden – der sog. *Kovarianzmatrix*. Diese sei mit $\mathbf{C}$ bezeichnet. Sie wird wie die Kovarianz $\sigma^2_{n1,n2}$ als Matrixprodukt beschrieben:

$$\mathbf{C} \equiv \frac{1}{T-1}\mathbf{X} \cdot \mathbf{X}^T \tag{3.12}$$

Eine weitere nützliche Darstellung der Kovarianzmatrix ist die Notation als Summe der Dyaden der Datenvektoren:

$$\mathbf{C} \equiv \frac{1}{T-1}\sum_{t=0}^{T-1} \vec{x}_t \cdot \vec{x}_t^T \tag{3.13}$$

Diese Form entsteht durch die Zerlegung von $\mathbf{X}$ in Blockmatrizen, und zwar so, dass ein Block gerade einem Datenvektor entspricht. Mit der Definition von $\mathbf{X}$ aus Gleichung 3.11 gilt (vgl. [Ser02]):

$$\mathbf{X} = (\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_t, \ldots, \vec{x}_{T-1}) = (\mathbf{X_0}\ \mathbf{X_1}\ \ldots\ \mathbf{X_t}\ \ldots\ \mathbf{X_{T-1}}) \tag{3.14}$$

sowie

$$\begin{aligned}\mathbf{X}\mathbf{X}^T &= (\mathbf{X_0} \ldots \mathbf{X_{T-1}}) \cdot \left(\mathbf{X_0}^T \ldots \mathbf{X_{T-1}}^T\right) \\ &= \mathbf{X_0} \cdot \mathbf{X_0}^T + \ldots + \mathbf{X_{T-1}} \cdot \mathbf{X_{T-1}}^T \\ &= \sum_{t=0}^{T-1} \mathbf{X_t} \cdot \mathbf{X_t}^T = \sum_{t=0}^{T-1} \vec{x}_t \cdot \vec{x}_t^T\end{aligned} \tag{3.15}$$

Die Einträge auf der Hauptdiagonalen von $\mathbf{C}$ sind genau die Varianzen der Datenreihe, während auf den restlichen Einträgen die paarweisen Kovarianzen zu finden sind.

$$\mathbf{C} = \frac{1}{T-1}\begin{pmatrix} \sigma^2_{0,0} & \sigma^2_{0,1} & \cdots & \sigma^2_{0,N-1} \\ \sigma^2_{1,0} & \sigma^2_{1,1} & \cdots & \sigma^2_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma^2_{N-1,0} & \sigma^2_{N-1,1} & \cdots & \sigma^2_{N-1,N-1} \end{pmatrix} \tag{3.16}$$

Die Kovarianzmatrix ist stets symmetrisch, da das Skalarprodukt zweier Vektoren kommutativ ist:

$$\begin{aligned}\sigma^2_{n1,n2} &= \frac{1}{T-1}\langle \vec{v}_{n1}, \vec{v}_{n2}\rangle \\ &= \frac{1}{T-1}\langle \vec{v}_{n2}, \vec{v}_{n1}\rangle \\ &= \sigma^2_{n2,n1}\end{aligned}$$

Eine symmetrische Matrix hat die Eigenschaft, in jedem Fall orthogonal diagonalisierbar zu sein; eine Eigenwertzerlegung von C liefert orthogonale Eigenvektoren sowie stets reelle Eigenwerte (vgl. [Ser02]).

3.2.4 Der PCA-Algorithmus: Von der Kovarianzmatrix zum Eigenwertproblem

Das Ziel der PCA ist das Finden einer orthonormalen Basis $\mathbf{B}$, welche das Koordinatensystem so transformiert, dass die paarweisen Kovarianzen der Komponenten der Datenreihe $\mathbf{X}$ nach dem Basiswechsel den Wert Null und die Varianzen maximale Werte annehmen, was bedeutet, dass die einzelnen Komponentenvektoren nunmehr keinen linearen Zusammenhang mehr aufweisen. Die zugehörige Kovarianzmatrix hat dann die Gestalt einer Diagonalmatrix.

Die Datenmatrix nach der Transformation sei mit $\mathbf{Y}$ bezeichnet und es sei $\mathbf{Y} \equiv \mathbf{B} \cdot \mathbf{X}$. Ist die Kovarianzmatrix $\mathbf{C_Y} = \frac{1}{T-1}\mathbf{Y} \cdot \mathbf{Y}^T$ eine Diagonalmatrix, so sind die Zeilenvektoren der Basis $\mathbf{B}$ die gesuchten Hauptkomponenten.

Man kann zeigen, dass die Eigenwertzerlegung der Kovarianzmatrix $\mathbf{C_X} = \frac{1}{T-1}\mathbf{X} \cdot \mathbf{X}^T$ die gewünschten Basisvektoren $\vec{b_i}$ liefert. Dazu bringt man $\mathbf{C_Y}$ in eine neue Form:

$$\mathbf{C_Y} = \frac{1}{T-1}\mathbf{Y} \cdot \mathbf{Y}^T \tag{3.17}$$

$$= \frac{1}{T-1}(\mathbf{BX}) \cdot (\mathbf{BX})^T \tag{3.18}$$

$$= \frac{1}{T-1}\mathbf{BXX^TB^T} \tag{3.19}$$

$$= \frac{1}{T-1}\mathbf{B}(\mathbf{XX^T})\mathbf{B^T} \tag{3.20}$$

Die Matrix $\mathbf{XX}^T$ ist aufgrund ihrer Definition immer symmetrisch (vgl. Kap. 3.2.3). Daraus folgt, dass sie stets diagonalisierbar ist. $\mathbf{XX}^T$ lässt sich demnach formulieren als $\mathbf{XX}^T = \mathbf{EDE}^T$. Dabei enthalten die Spalten der Matrix $\mathbf{E}$ die Eigenvektoren von $\mathbf{XX}^T$, $\mathbf{D}$ ist die Diagonalmatrix der Eigenwerte von $\mathbf{XX}^T$. Wählt man nun $\mathbf{B}$ so, dass $\mathbf{B} \equiv \mathbf{E}^T$ gilt, dann enthält $\mathbf{B}$ nun die Eigenvektoren von $\mathbf{XX}^T$ in Form von Zeilenvektoren. $\mathbf{XX}^T$ lässt sich nun als $\mathbf{XX}^T = \mathbf{B}^T\mathbf{DB}$ formulieren. Da die Orthonormalität eine Bedingung an die Basis $\mathbf{B}$ ist, folgt $\mathbf{B}^T = \mathbf{B}^{-1}$ und es gilt:

$$\mathbf{XX}^T = \mathbf{B}^{-1}\mathbf{DB} \tag{3.21}$$

Setzt man 3.21 in 3.20 ein, so erhält man:

$$\mathbf{C_Y} = \frac{1}{T-1}\mathbf{B}(\mathbf{X}\mathbf{X}^T)\mathbf{B}^T \tag{3.22}$$

$$= \frac{1}{T-1}\mathbf{B}(\mathbf{B}^{-1}\mathbf{D}\mathbf{B})\mathbf{B}^T \tag{3.23}$$

$$= \frac{1}{T-1}(\mathbf{B}\mathbf{B}^{-1})\mathbf{D}(\mathbf{B}\mathbf{B}^{-1}) \tag{3.24}$$

$$= \frac{1}{T-1}\mathbf{D} \tag{3.25}$$

Wie man sieht, ist die Kovarianzmatrix $\mathbf{C_Y}$ diagonalisiert und $\mathbf{B}$ erfüllt die ursprüngliche Zielsetzung: $\mathbf{B}$ transformiert einen Datensatz $\mathbf{X}$ so, dass die Kovarianzen zwischen den Komponenten verschwinden. Um also die Hauptkomponenten zu finden, müssen die Eigenvektoren von $\mathbf{X}\mathbf{X}^T$ berechnet werden. Dies sind dann genau die Zeilenvektoren von $\mathbf{B}$.

In der Praxis kann dies effektiv z.B. über die *SVD*[4]-Methode erfolgen:

Zu jeder Matrix $\mathbf{Y}$ gibt es eine Zerlegung $\mathbf{Y} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, wobei $\mathbf{U}$ und $\mathbf{V}$ orthonormal sind. Es gilt weiterhin:

$$\begin{aligned}\mathbf{Y}^T \cdot \mathbf{Y} &= (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T \cdot \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \\ &= \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T \cdot \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \\ &= \mathbf{V}\mathbf{\Sigma}^T\mathbf{\Sigma}\mathbf{V}^T\end{aligned} \tag{3.26}$$

Somit ist $\mathbf{V}$ die Matrix der Eigenvektoren und $\mathbf{\Sigma}^T\mathbf{\Sigma}$ die Matrix der Eigenwerte von $\mathbf{Y}^T\mathbf{Y}$. Wählt man

$$\mathbf{Y} \equiv \frac{1}{\sqrt{T-1}}\mathbf{X}^T \tag{3.27}$$

so erhält man für

$$\mathbf{Y}^T\mathbf{Y} = \frac{1}{T-1}\mathbf{X}\mathbf{X}^T \tag{3.28}$$

die Kovarianzmatrix von $\mathbf{X}$. Die gesuchte Basis $\mathbf{B}$ ergibt sich nun wie folgt: Die *SVD*-Zerlegung von $\mathbf{Y}$ liefert die orthonormale Basis $\mathbf{V}$, welche die gesuchten Eigenvektoren von $\mathbf{B}$ in Form von Spaltenvektoren enthält, d.h. $\mathbf{V}^T = \mathbf{B}$.

3.2.5 Die Bedeutung der Eigenwerte

In der Hauptkomponentenanalyse kann der Eigenwert λ_i der Kovarianzmatrix als ein Maß für die „Relevanz“ des zugehörigen Eigenvektors $\vec{b_i}$, d.h. der zugehörigen Hauptkomponente, betrachtet werden. Die Eigenwerte sind gleich den Einträgen auf den Diagonalen der Kovarianzmatrix $\mathbf{C_Y}$ des Datenvektors $\mathbf{Y}$

[4] Anm.: Die Abkürzung SVD steht für **S**ingular **V**alue **D**ecomposition.

nach Ausführung der PCA. Sie sind also gleich den Varianzen des transformierten Datensatzes. Werden sie so normiert, dass sie in der Summe Eins ergeben, dann lässt sich direkt der von ihnen erfasste Anteil an der Gesamtvarianz angeben. Die normierten Eigenwerte $\tilde{\lambda_i}$ bestimmen sich wie folgt:

$$\tilde{\lambda_i} = \frac{\lambda_i}{\sum_i \lambda_i} \tag{3.29}$$

Liegen die Paare von Eigenwerten und Eigenvektoren in einer geordneten Reihenfolge vor, so kann die Dimensionsreduktion durch das Vernachlässigen aller Eigenvektoren bis auf die ersten M erfolgen, deren Eigenwerte einen bestimmten Anteil an der Gesamtvarianz erklären. Dieser Anteil sei mit ϵ_M ($\epsilon_M = 0 \ldots 1$, $\epsilon_M \in \mathbf{R}$) bezeichnet. Er lässt sich wie folgt berechnen:

$$\epsilon_M = \sum_{i=0}^{M-1} \tilde{\lambda_i} \tag{3.30}$$

Der Wert ϵ_M kann als der prozentuale Anteil am Informationsgehalt des Datensatzes betrachtet werden, welcher in den ersten M Hauptkomponenten verbleibt. Um in einem konkreten Anwendungsfall die Anzahl der verwendeten Hauptkomponenten festzulegen, wird ein Schwellwert ϵ^* eingeführt, welcher den prozentualen Mindestanteil an der Gesamtinformation angibt. Gesucht ist dann das *kleinste* M, für das $\epsilon_M > \epsilon^*$ erfüllt ist.

3.2.6 Die Merkmalsvektoren

Das Ergebnis der PCA-Analyse ist die Restrukturierung des Datenvektors $\vec{x}_t$ bezüglich seiner Hauptkomponenten $\vec{b}_i$. Durch den Basiswechsel wird ein sog. *Merkmalsvektor* gewonnen, welcher den Datensatz neu beschreibt. Die Einträge des Merkmalsvektores sind genau die a_i der Linearkombination $\vec{x}_t = \sum_{i=0}^{M-1} a_i \cdot \vec{b}_i$.

Ein Merkmal a_i wird ermittelt, indem der Datenvektor auf die Hauptkomponente $\vec{b}_i$ projiziert wird:

$$a_i = \left\langle \vec{x}_t, \vec{b}_i \right\rangle \tag{3.31}$$

Analog berechnet man den Merkmalsvektor $\vec{a}_t$ eines Datensatzes in Matrixschreibweise:

$$\vec{a}_t = \begin{pmatrix} a_0 \\ \vdots \\ a_{M-1} \end{pmatrix} = \mathbf{B} \cdot \vec{x}_t \tag{3.32}$$

Die Matrix $\mathbf{A}$ aller Merkmalsvektoren $\vec{a}_t$ kann als Matrixprodukt geschrieben werden[5]:

$$\mathbf{A} = (\vec{a}_0, \vec{a}_1, \ldots, \vec{a}_t, \ldots, \vec{a}_{T-1}) = \mathbf{B} \cdot \mathbf{X} \tag{3.33}$$

[5]Anm.: Die Matrix $\mathbf{A}$ entspricht genau der Matrix $\mathbf{Y}$ aus Kap. 3.2.4, d.h. die korrespondierende Kovarianzmatrix $\mathbf{C}_A$ ist diagonal.

3.2.7 Rekonstruktion

Die Extraktion der relevanten Merkmale eines Datensatzes stellt in vielen Anwendungsgebieten eine Grundlage zur weiteren Verarbeitung dar. Durch die Dimensionsreduktion nach der Transformation der Daten in den Merkmalsraum können erhebliche Rechenkosten eingespart werden (vgl. [TLP06]), da die Länge der Merkmalsvektoren in der Praxis wesentlich kleiner ist als die der ursprünglichen Datenvektoren. Ein klassischer Anwendungsfall der Hauptkomponentenanalyse sind Klassifikationsverfahren wie die Clusteranalyse. Hier genügen oft wenige Merkmalswerte aus, um eine eindeutige Gruppenzugehörigkeit entscheiden zu können.

In dieser Arbeit wird die Hauptkomponentenanalyse hingegen zur Kompression von Daten-Zeitreihen und nicht zur Klassifikation eingesetzt. Die Rekonstruktion aus dem reduzierten Raum der Merkmale in den Raum der Eingabedaten ist daher zur Beurteilung der visuellen Qualität unumgänglich. Sie erfolgt durch die Rückprojektion der Merkmalsvektoren mithilfe der ersten M Hauptkomponenten $\vec{b}_i$ ($i = 0..M-1$). Ein rekonstruierter Datenvektor $\tilde{\vec{x}}_t$ berechnet sich aus den Merkmalen a_i wie folgt:

$$\tilde{\vec{x}}_t = \sum_{i=0}^{M-1} a_i \cdot \vec{b}_i \tag{3.34}$$

Kompakter lässt sich die Rekonstruktion in Matrixschreibweise formulieren. Sei $\tilde{\mathbf{X}}$ die Menge der rekonstruierten Datenvektoren mit $\tilde{\mathbf{X}} = (\tilde{\vec{x}}_0 \dots \tilde{\vec{x}}_{T-1})$ und $\mathbf{A}$ die Menge der Merkmalsvektoren mit $\mathbf{A} = (\vec{a}_0 \dots \vec{a}_{T-1})$ sowie die Basis $\mathbf{B} = (\vec{b}_0 \dots \vec{b}_{M-1})$ gegeben, so gilt:

$$\tilde{\mathbf{X}} = \mathbf{B}^T \cdot \mathbf{A} \tag{3.35}$$

3.2.8 Datenkompression mithilfe der PCA

Das folgende Kapitel beschäftigt sich mit den Kompressionseigenschaften der Hauptkomponentenanalyse und gibt Aufschluss über den Grad der Kompression, welcher in der Praxis zu erwarten ist. Dieser ist stark abhängig vom Inhalt der analysierten Zeitreihe; bei Datensätzen mit starkem linearem Zusammenhang zwischen den einzelnen Komponenten ist von einer hohen Kompression auszugehen, ist dieser hingegen nicht gegeben, so kann nur von einer schwachen Kompression ausgegangen werden.

Die Speicherkosten S_x eines Datenvektors $\vec{x}_t$ sind proportional zu seiner Länge N:

$$S_x \propto N \tag{3.36}$$

Der Speicherverbrauch S_X der gesamten unkomprimierten Zeitreihe $\mathbf{X}$ lässt sich demnach angeben als

Produkt der Länge N des Datenvektors multipliziert mit der Anzahl der Zeitschritte T:

$$S_X \propto N \cdot T \tag{3.37}$$

Die Speicherkosten eines Merkmalsvektors $\vec{a}_t$ der Länge M ergeben sich analog Gleichung 3.36:

$$S_a \propto M \tag{3.38}$$

Analog zu S_X können die Speicherkosten S_A der T Merkmalsvektoren der gesamten Zeitreihe bestimmt werden:

$$S_A \propto M \cdot T \tag{3.39}$$

Wird im entsprechenden Anwendungsfall allein die Matrix $\mathbf{A}$ aller Merkmalsvektoren für weitere Berechnungen benötigt, so ergibt sich der Kompressionsgrad K_A aus S_X und S_A mit den Proportionalitätsfaktoren c_1 und c_2:

$$K_A \equiv \frac{S_X}{S_A} = \frac{c_1 \cdot N \cdot T}{c_2 \cdot M \cdot T} = \frac{c_1}{c_2} \cdot \frac{N}{M} = c \cdot \frac{N}{M} \tag{3.40}$$

Geht man davon aus, dass die N Komponenten eines Datenvektors im gleichen Format wie die M Komponenten eines Merkmalvektors gespeichert werden, so gilt $c = 1$ und 3.40 vereinfacht sich zu:

$$K_A = \frac{N}{M} \tag{3.41}$$

Der Kompressionsgrad ist demnach allein durch das Verhältnis der Längen von Daten- und Merkmalsvektor beschrieben. Generell kann ein Speichervorteil nur dann erreicht werden, wenn die Länge der Merkmalsvektoren kleiner als die der Datenvekoren ist.

Da in dieser Arbeit neben der Darstellung der Datenvektoren in Form ihrer Merkmalsvektoren auch die Rekonstruktion ebenjener erforderlich ist, müssen neben den Merkmalsvektoren auch die Hauptkomponenten gespeichert werden. Dies bedeutet einen zusätzlichen Speicheraufwand. Die Länge einer Hauptkomponente $\vec{b}_i$ ist gleich der Länge eines Datenvektors und benötigt demzufolge den gleichen Speicheraufwand: $S_b = S_x$. Die Speicherkosten S_B aller M Basisvektoren sind unabhängig von der Anzahl T der Zeitschritte:

$$S_B \propto N \cdot M \tag{3.42}$$

Der kumulative Speicheraufwand S_R der komprimierten Zeitreihe ergibt sich, unter der Annahme gleicher Speicherformate für die Einträge in den Vektoren $\vec{b}_i$ und $\vec{a}_t$, d.h. gleicher Proportionalitätsfaktoren für S_B und S_A, wie folgt:

$$S_R = S_B + S_A \tag{3.43}$$
$$= c \cdot N \cdot M + c \cdot M \cdot T \tag{3.44}$$
$$= c \cdot M \cdot (N + T) \tag{3.45}$$
$$S_R \propto M \cdot (N + T) \tag{3.46}$$

Die Anzahl M der verwendeten Hauptkomponenten legt zugleich die Länge der Merkmalsvektoren fest.

Gesucht ist nun die obere Schranke M^*, bei der $S_R^* = S_X$ gilt. Es ergibt sich genau dann ein Speichervorteil, d.h. $S_R < S_X$, wenn M kleiner M^* gewählt wird. Dabei werden wiederum die Speicherformate für die Einträge in den Vektoren als identisch angenommen. M^* lässt sie wie folgt bestimmen:

$$\begin{aligned} S_R^* &= S_X \\ M^* \cdot (N + T) &= N \cdot T \\ M^* &= \frac{N \cdot T}{N + T} \end{aligned} \tag{3.47}$$

Es wird eine Hilfsvariable h eingeführt mit

$$h \equiv \frac{N}{T} \equiv \text{const.} \tag{3.48}$$

Sie beschreibt das Verhältnis von räumlicher zu zeitlicher Auflösung der Zeitreihe, mit N als Anzahl der Gitterpunkte und T als Anzahl der Zeitschritte. Im Folgenden wird angenommen, dass $N \gg T$. Setzt man Gleichung 3.48 in 3.47 ein, so erhält man für M^*:

$$\begin{aligned} M^* &= \frac{N \cdot T}{N + T} \\ &= \frac{h \cdot T^2}{h \cdot T + T} \\ &= \frac{h \cdot T^2}{(h + 1) \cdot T} \\ M^* &= \frac{h}{h + 1} \cdot T \end{aligned} \tag{3.49}$$

Unter der Bedingung $N \gg T$ gilt $\frac{h}{h+1} \approx 1$ und

$$M^* \approx T \tag{3.50}$$

Die PCA kann also nur dann zur Speicherkompression eingesetzt werden, wenn die Anzahl M der Hauptkomponenten kleiner ist als die Anzahl der Zeitschritte T. Der Kompressionsgrad K_R lässt sich hierbei

als Verhältnis von T und M darstellen:

$$K_R = \frac{N \cdot T}{(N+T) \cdot M} \tag{3.51}$$

$$= \frac{h}{h+1} \cdot \frac{T}{M} \tag{3.52}$$

$$K_R \approx \frac{T}{M} \tag{3.53}$$

3.2.9 Ein Beispiel

Den Abschluss über die Kapitel zur PCA soll eine Beispielrechnung bilden. Gegeben ist eine hypothetische Zeitreihe $\mathbf{X} = (\vec{x}_t | \vec{x}_t \in \mathbb{R}^4, t = 0 \ldots 5)$ mit $\vec{x}_t = (n_{0,t}, n_{1,t}, n_{2,t}, n_{3,t})^T$:

$$\mathbf{X} = (\vec{x}_0 \ldots \vec{x}_5) = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ -6 & -5 & -4 & -3 & -2 & -1 \\ 3/2 & 3 & 9/2 & 6 & 15/2 & 9 \\ 0 & 5 & 3 & 5 & 1 & 2 \end{pmatrix}$$

Die linearen Zusammenhänge unter den ersten drei Komponenten sind leicht erkennbar, während bei der letzten kein solcher offensichtlicher Zusammenhang mit einer jeweils anderen Komponente besteht. Abbildung 3.4 illustriert die Abhängigkeiten, dargestellt sind die Werte der einzelnen Komponenten über der Zeit. Man sieht deutlich den gleichsinnigen Zusammenhang zwischen der ersten und zweiten Komponente (vgl. Kap. 3.2.3) sowie den proportionalen Zusammenhang zwischen der ersten und dritten. Es ist zu erwarten, dass die Hauptkomponentenanalyse zwei starke Hauptkomponenten extrahiert.

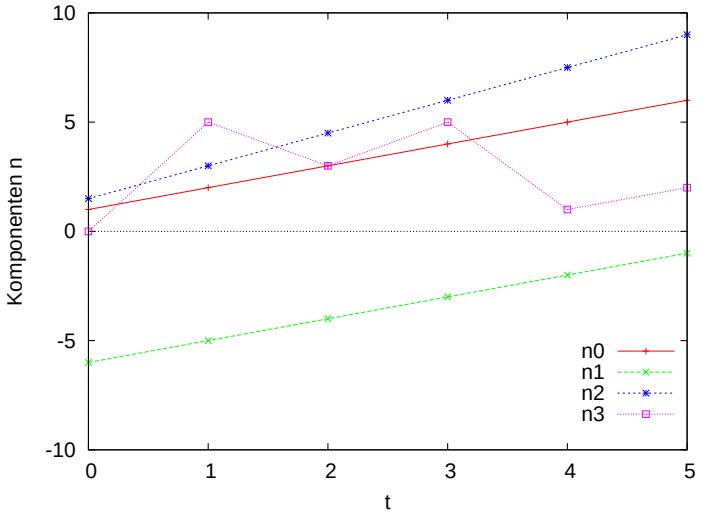

Abbildung 3.4: Plot der vier Komponenten $n_0 \ldots n_3$ der Datenvektoren $\vec{x}_t$ der Beispiel-Zeitreihe X entlang t.

Die erste beschreibt die Information, welche sich aus den drei untereinander abhängigen Komponenten n_0, n_1 und n_2 ergibt und die zweite beschreibt die Information, welche sich in der letzten Komponente n_3 verbirgt.

Zunächst wird die Datenreihe mittelwertbereinigt, sodass sich die Berechnung der Kovarianzen auf die Berechnung der Skalarprodukte reduziert:

$$\tilde{\mathbf{X}} = \begin{pmatrix} -5/2 & -3/2 & -1/2 & 1/2 & 3/2 & 5/2 \\ -5/2 & -3/2 & -1/2 & 1/2 & 3/2 & 5/2 \\ -15/4 & -9/4 & -3/4 & 3/4 & 9/4 & 15/4 \\ -8/3 & 7/3 & 1/3 & 7/3 & -5/3 & -2/3 \end{pmatrix}$$

Die Kovarianzmatrix $\mathbf{C}$ errechnet sich wie folgt:

$$\mathbf{C} = \frac{1}{5} \cdot \tilde{X}\tilde{X}^T = \begin{pmatrix} 7/2 & 7/2 & 21/4 & 0 \\ 7/2 & 7/2 & 21/4 & 0 \\ 21/4 & 21/4 & 63/8 & 0 \\ 0 & 0 & 0 & 64/15 \end{pmatrix}$$

Nach der Eigenwertzerlegung ergeben sich folgende, bereits normierte, Eigenwerte $\tilde{\lambda_i}$:

$$\tilde{\lambda_i} = (0.777, 0.223, 0, 0)$$

Die Vermutung wird bestätigt: Ein kumulativer Varianzanteil von 100% wird durch die ersten zwei Eigenwerte abgedeckt, die gesamte Information eines Datenvektors lässt sich demnach durch zwei anstelle von vier Variablen ausdrücken. Die den Eigenwerten zugehörigen Eigenvektoren $\vec{e}_i$ sind die Spaltenvektoren der Matrix $\mathbf{E}$:

$$\mathbf{E} = (\vec{e}_0 \dots \vec{x}_3) = \begin{pmatrix} 2/3 & 0 & -3/2 & -1 \\ 2/3 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Die beiden ersten – normierten – Spaltenvektoren $\tilde{\vec{e}}_0$ und $\tilde{\vec{e}}_1$ bilden die ersten beiden Hauptkomponenten $\vec{b}_0$ und $\vec{b}_1$:

$$\vec{b}_0 \equiv \tilde{\vec{e}}_0 = \frac{\vec{e}_0}{\|\vec{e}_0\|} = \frac{1}{\sqrt{17}} \cdot (2, 2, 3, 0)^T$$
$$\vec{b}_1 \equiv \tilde{\vec{e}}_1 = \frac{\vec{e}_1}{\|\vec{e}_1\|} = (0, 0, 0, 1)^T$$

Die Basis $\mathbf{B}$ ergibt sich aus den Hauptkomponenten als Zeilenvektoren:

$$\mathbf{B} \equiv \begin{pmatrix} \vec{b}_0 \\ \vec{b}_1 \end{pmatrix} = \frac{1}{\sqrt{17}} \cdot \begin{pmatrix} 2 & 2 & 3 & 0 \\ 0 & 0 & 0 & \sqrt{17} \end{pmatrix}$$

Die Merkmalsvektoren $\vec{a}_t$ werden durch die Projektion von $\mathbf{X}$ auf $\mathbf{B}$ gewonnen; dies entspricht ihrem Matrixprodukt:

$$\mathbf{A} \equiv (\vec{a}_0 \dots \vec{a}_5) = \mathbf{B} \cdot \mathbf{X} = \begin{pmatrix} -5\sqrt{17}/4 & -3\sqrt{17}/4 & -\sqrt{17}/4 & \sqrt{17}/4 & 3\sqrt{17}/4 & 5\sqrt{17}/4 \\ -8/3 & 7/3 & 1/3 & 7/3 & -5/3 & -2/3 \end{pmatrix}$$

Die Abbildung 3.5 veranschaulicht die Daten nach der Projektion in die Basis $\mathbf{B}$. Die Datensätze $\vec{x}_t \in \mathbb{R}^4$ sind durch die Merkmale $\vec{a}_t \in \mathbb{R}^2$ vollständig beschrieben: Die Komponenten n_0, n_1 und n_2 werden gemeinsam durch das lineare Merkmal a_0 beschrieben[6], die Komponente n_3 durch das Merkmal a_1 (man vergleiche den Verlauf des Graphen für n_3 in Abb. 3.4 mit dem für a_1 in Abb. 3.5).

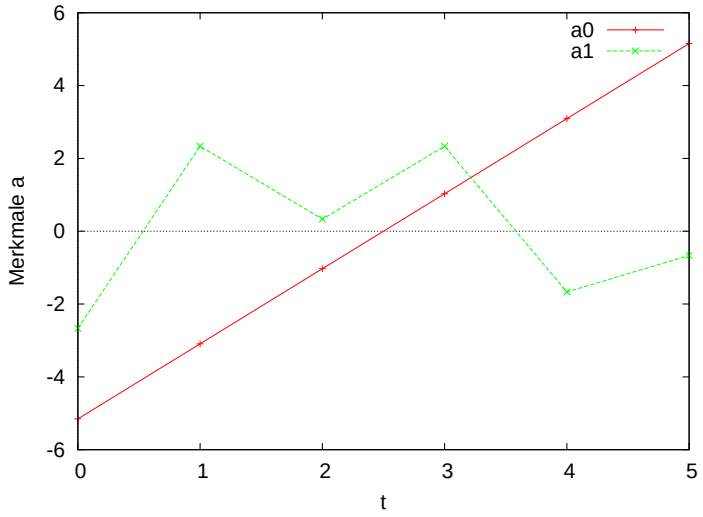

Abbildung 3.5: Plot der berechneten Merkmale a_0 und a_1 der Datenreihe $\mathbf{X}$ entlang t.

Im Folgenden werden die Kompressionsgrade K_A und K_R berechnet (vgl. Kap. 3.2.8). Zunächst wird von einer reinen Merkmalsextraktion ausgegangen, bei der sich der Kompressionsgrad K_A aus dem Verhältnis S_X/S_A ergibt:

$$K_A = \frac{S_X}{S_A} = \frac{N \cdot T}{M \cdot T} = \frac{4 \cdot 6}{2 \cdot 6} = \frac{2}{1}$$

Die Datensätze werden in diesem Fall um den Faktor 2 komprimiert. Soll später eine Rekonstruktion der ursprünglichen Zeitreihe erfolgen, so müssen zusätzlich die Hauptkomponenten gespeichert werden. Es ergibt sich folgender Kompressionsgrad K_R:

$$K_R = \frac{S_X}{S_R} = \frac{N \cdot T}{M \cdot (N+T)} = \frac{4 \cdot 6}{2 \cdot (4+6)} = \frac{6}{5}$$

Die Zeitreihe kann um den Faktor 1,2 komprimiert werden. Ein höherer Wert würde sich im Falle der Rekonstruktion dann ergeben, wenn, unter sonst gleichen Abhängigkeiten, entweder die Anzahl der Zeitschritte höher wäre oder das Verhältnis zwischen der Anzahl der Komponenten n_i und der Anzahl der

[6] Anm.: Die Linearität, durch welche sich der Graph von a_0 auszeichnet, ist jedoch allein der Wahl der Ursprungsdaten geschuldet.

Hauptkomponenten b_i größer wäre. Eine weitere Option zur Erhöhung des Kompressionsgrades bestünde in der Vernachlässigung der zweiten Hauptkomponente. Dann wären immerhin noch ca. 77% der Gesamtvarianz durch die verbleibende erste Hauptkomponente erklärt, der Kompressionsfaktor K_A würde auf 4 und K_R auf $2,4$ steigen, jedoch kann dann die vierte Komponente n_3 der Beispiel-Datenvektoren *nicht* wieder rekonstruiert werden.

3.2.10 Lineare Interpolation der Datenvektoren im Merkmalsraum

Ber der Hauptkomponentenanalyse wird ein Datenvektor durch einen Basiswechsel in einen Merkmalsvektor überführt. Es handelt sich dabei um eine *lineare* Abbildung mit der Abbildungsmatrix $\mathbf{B}$. Eine Abbildung $u : \mathbf{E} \to \mathbf{F}$, wobei $\mathbf{E}$ und $\mathbf{F}$ Vektorräume über einem Körper $\mathbb{K}$ sind, heißt linear, wenn sie folgende Bedingungen erfüllt (vgl. [Ser02]):

$$u(x+y) = u(x) + u(y)$$
$$u(tx) = tu(x)$$

mit $x, y \in \mathbf{E}$ und $t \in \mathbb{K}$.

Die Linearität der PCA lässt sich daher nutzen, um zwei Datenvektoren im Merkmalsraum linear zu interpolieren. Dazu seien $\vec{x}_1, \vec{x}_2 \in \mathbb{R}^N$ mit ihren zugehörigen Merkmalen $\vec{a}_1, \vec{a}_2$ sowie der Basis $\mathbf{B}^T$ gegeben. Es gelte:

$$\vec{x}_1 = \mathbf{B}^T \cdot \vec{a}_1$$
$$\vec{x}_2 = \mathbf{B}^T \cdot \vec{a}_2$$

Sei $\vec{x}_{int}$ das Ergebnis der linearen Interpolation von $\vec{x}_1$ und $\vec{x}_2$ mithilfe des Parameters t, so gilt:

$$\begin{aligned}
\vec{x}_{int} &= (1-t) \cdot \vec{x}_1 + t \cdot \vec{x}_2 \\
&= (1-t) \cdot \mathbf{B}^T \cdot \vec{a}_1 + t \cdot \mathbf{B}^T \cdot \vec{a}_2 \\
&= \mathbf{B}^T \cdot (1-t) \cdot \vec{a}_1 + \mathbf{B}^T \cdot t \cdot \vec{a}_2 \\
&= \mathbf{B}^T \cdot [(1-t) \cdot \vec{a}_1 + t \cdot \vec{a}_2] \\
&= \mathbf{B}^T \cdot \vec{a}_{int}
\end{aligned}$$

Man sieht, dass $\vec{x}_{int}$ durch die lineare Interpolation von $\vec{a}_1$ und $\vec{a}_2$ im Merkmalsraum mit anschließender Rückabbildung ausgedrückt werden kann.

3.3 Die nichtlineare Kern-PCA

Die Kern-PCA ist eine Verallgemeinerung der PCA und stellt die Verknüpfung von Kernmethoden mit den Werkzeugen der Hauptkomponentenanalyse dar. Grundlage dafür sind sogenannte Kernfunktionen, kurz *Kerne*, welche an späterer Stelle noch detaillierter beschrieben werden. Sie ermöglichen die Extraktion von Merkmalsvektoren, welche mit der klassischen PCA nicht gefunden werden können. Durch ihre Effizienz, die leichte Anwendbarkeit sowie der Kombinierbarkeit auf vielen Anwendungsgebieten, wie z.B. der Muster-Erkennung, Cluster-Analyse oder der Support Vektor Maschinen (vgl. [SS01], [KPK01], [YZZL08]), sind die Kernmethoden in den letzten Jahren zu einem Standardwerkzeug geworden. Dabei wird die rechentechnische Effizienz eines linearen Verfahrens, in der vorliegenden Arbeit ist dies die PCA, mit der Flexibilität eines nichtlinearen Systems kombiniert (vgl. [CST03]).

Während bei der Hauptkomponentenanalyse lineare Zusammenhänge aufgezeigt werden, können bei der Anwendung der Kern-PCA auch relevante nichtlineare Abhängigkeiten aus den Daten extrahiert werden. Im Folgenden wird die Idee hinter den Kernmethoden erläutert.

3.3.1 Die Idee der Kerne

Die Grundlage der Kernmethoden bildet die nichtlineare Abbildung $\Phi(\vec{x})$, welche einen Datenvektor $\vec{x}$ in einen hochdimensionalen Merkmalsraum $\mathcal{H}$ abbildet:

$$\Phi : \mathbb{R}^N \to \mathcal{H}$$
$$\vec{x} \mapsto \Phi(\vec{x})$$

Abbildung 3.6 zeigt die Transformation der Daten aus dem ursprünglichen Raum in den hochdimensionalen Raum $\mathcal{H}$ mithilfe von Φ. Durch die nichtlineare „Entzerrung“ werden die Richtungen der größten Varianzen in $\mathcal{H}$ linear, sodass eine Hauptkomponentenanalyse hier optimal, d.h. im Sinne des Auffindens hoher Varianzanteile in den ersten Hauptkomponenten, ansetzen kann.

Die Dimension von $\mathcal{H}$ ist dabei nicht beschränkt, was zunächst vermuten lässt, dass das Problem, welches es zu lösen gilt, rechentechnisch schwieriger wird, da durch die, gegenüber dem ursprünglichen Raum, höhere Dimension des Merkmalsraumes $\mathcal{H}$ wesentlich mehr Datensätze benötigt würden, um selbigen abzudecken. Dieses Problem ist auch bekannt unter dem Schlagwort „curse of dimensionality“, geprägt durch Richard Bellman. Es lässt sich jedoch zeigen, dass ebendies bei der Verwendung von Kernalgorithmen nicht zutreffend ist (vgl. [SS01]). So können im Merkmalsraum $\mathcal{H}$ meist einfache Klassen linearer Algorithmen verwendet werden, da die Reichhaltigkeit der Transformation allein in der Abbildung Φ steckt.

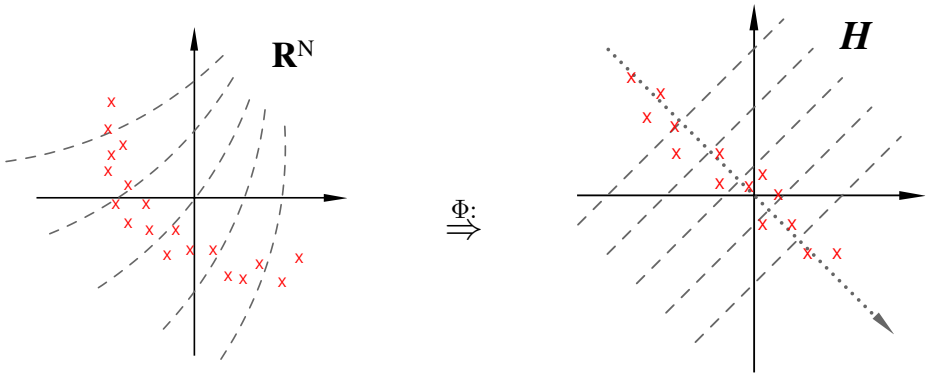

Abbildung 3.6: Die nichtlineare Transformation $\Phi : \mathbb{R}^N \to \mathcal{H}$ (vgl. [SS01] S.432).

Hier setzen die Kernfunktionen an: Mit ihrer Hilfe kann das Ergebnis des Skalarproduktes zweier Vektoren $\vec{\Phi}(\vec{x})$ und $\vec{\Phi}(\vec{y})$ im Merkmalsraum $\mathcal{H}$ effektiv berechnet werden, jedoch *ohne* die komplexe Abbildung von $\mathbb{R}^N$ auf $\mathcal{H}$ explizit ausführen zu müssen. Durch die geschickte Definition von Φ können die Skalarprodukte im ursprünglichen niederdimensionalen Raum berechnet werden.

Die Kernfunktion k wird wie folgt definiert:

$$k(\vec{x}, \vec{y}) \equiv \left\langle \vec{\Phi}(\vec{x}), \vec{\Phi}(\vec{y}) \right\rangle \tag{3.54}$$

Dazu ein kleines Beispiel (vgl. [SS01]):

$$\begin{aligned} \Phi : \mathbb{R}^2 &\to \mathcal{H} \\ (x_0, x_1) &\mapsto (x_0^2, x_1^2, \sqrt{2}x_0x_1) \end{aligned}$$

Für dieses Beispiel kann nun ein „passender“ Kern $k(\vec{x}, \vec{y})$ gefunden werden:

$$\begin{aligned} k(\vec{x}, \vec{y}) &= \left\langle \vec{\Phi}(\vec{x}), \vec{\Phi}(\vec{y}) \right\rangle \\ &= (x_0^2, x_1^2, \sqrt{2}x_0x_1) \cdot (y_0^2, y_1^2, \sqrt{2}y_0y_1)^T \\ &= ((x_0, x_1) \cdot (y_0, y_1)^T)^2 \\ &= \langle \vec{x}, \vec{y} \rangle^2 \end{aligned}$$

Der Kern berechnet im Eingaberaum das Skalarprodukt zweier Vektoren des „hochdimensionalen“ Merkmalsraumes rechen-extensiv, indem das euklidische Skalarprodukt gebildet und dessen Ergebnis anschließend quadriert wird. Der Beispiel-Kern $k = \langle \cdot, \cdot \rangle^2$ gehört zur Klasse der *polynomiellen* Kerne. Er kann verallgemeinert werden zu:

$$k(\vec{x}, \vec{y}) = \langle \vec{x}, \vec{y} \rangle^d \qquad (\vec{x}, \vec{y} \in \mathbb{R}^N \text{ und } N, d \in \mathbb{N})$$

Die Klasse der polynomiellen Kerne berechnen die Skalarprodukte im Raum aller Produkte von d Einträgen der Vektoren $\vec{x}$ bzw. $\vec{y}$ ([SS01]).

Neben den polynomiellen Kernen gibt es zahlreiche weitere Kernfunktionen (vgl. [SMS99], [YZZL08]) wie z.B.:

- *Sigmoid-Kerne*: $k(\vec{x}, \vec{y}) = \tanh(\kappa \langle \vec{x}, \vec{y} \rangle + \theta)$
- *Kerne für Gauß'sche Radiale-Basis-Funktionen*: $k(\vec{x}, \vec{y}) = \exp(- \|\vec{x} - \vec{y}\|^2 / (2\sigma^2))$
- *Inverse Multiquadratische Kerne*: $k(\vec{x}, \vec{y}) = \frac{1}{\sqrt{\|x-y\|^2 + c^2}}$

Es lässt sich zeigen, dass jeder Algorithmus, welcher sich allein auf Basis von Skalarprodukten formulieren lässt, durch die Anwendung der Kernmethoden implizit in $\mathcal{H}$ ausgeführt und somit nichtlinear verallgemeinert werden kann (vgl. [SMS99]). Die vorliegende Arbeit zeigt dies im Folgenden für die Verknüpfung von Kernmethoden mit der Hauptkomponentenanalyse zur sog. Kern-PCA.

3.3.2 Die Kernmatrix

Mithilfe der Kernfunktion $k(\vec{x}_i, \vec{x}_j)$ kann für eine Zeitreihe eine spezielle Form einer *Gram'schen Matrix* (vgl. [SA94]) aufgestellt werden, die sogenannte *Kernmatrix* (vgl. [SS01]). Eine Gram'sche Matrix $\mathbf{M}$ ist die Matrix aller paarweisen Skalarprodukte der T Datenvektoren $\vec{x}_t$:

$$\mathbf{M} \equiv \begin{pmatrix} < \vec{x}_0, \vec{x}_0 > & \cdots & < \vec{x}_0, \vec{x}_{T-1} > \\ \vdots & \ddots & \vdots \\ < \vec{x}_{T-1}, \vec{x}_{T-1} > & \cdots & < \vec{x}_{T-1}, \vec{x}_{T-1} > \end{pmatrix}$$

Die Kernfunktionen $k(\vec{x}_i, \vec{x}_j) = \langle \Phi(\vec{x}_i), \Phi(\vec{x}_j) \rangle$ substituieren die Skalarprodukte in $\mathbf{M}$ und die Kernmatrix $\mathbf{K}$ wird wie folgt definiert:

$$\begin{aligned} \mathbf{K} &\equiv \begin{pmatrix} \langle \Phi(\vec{x}_0), \Phi(\vec{x}_0) \rangle & \cdots & \langle \Phi(\vec{x}_0), \Phi(\vec{x}_{T-1}) \rangle \\ \vdots & \ddots & \vdots \\ \langle \Phi(\vec{x}_{T-1}), \Phi(\vec{x}_0) \rangle & \cdots & \langle \Phi(\vec{x}_{T-1}), \Phi(\vec{x}_{T-1}) \rangle \end{pmatrix} \\ &= \begin{pmatrix} k(\vec{x}_0, \vec{x}_0) & \cdots & k(\vec{x}_0, \vec{x}_{T-1}) \\ \vdots & \ddots & \vdots \\ k(\vec{x}_{T-1}, \vec{x}_0) & \cdots & k(\vec{x}_{T-1}, \vec{x}_{T-1}) \end{pmatrix} \end{aligned} \tag{3.55}$$

Die Kernmatrix ist demzufolge die Gram'sche Matrix der Zeitreihe im Merkmalsraum $\mathcal{H}$, deren Einträge im Ursprungsraum berechnet werden können.

3.3.3 Die Anwendung der Kernfunktionen auf die PCA

Gegeben sei die nichtlineare Abbildung Φ, welche die Zeitreihe $\mathbf{X} = (\vec{x}_0 \dots \vec{x}_{T-1})$ in den Merkmalsraum $\mathcal{H}$ transformiert:

$$\begin{aligned} \Phi : \mathbb{R}^N &\to \mathcal{H} \\ \mathbf{X} &\mapsto \hat{\mathbf{X}} = (\Phi(\vec{x}_0) \dots \Phi(\vec{x}_{T-1})) \end{aligned}$$

Im Folgenden wird davon ausgegangen, dass die transformierte Zeitreihe $\hat{\mathbf{X}}$ bereits zentriert, d.h. mittelwertbereinigt, wurde. Wie dies effizient ohne die explizite Berechnung von Φ bewerkstelligt werden kann, wird an späterer Stelle im Detail erläutert.

Zunächst wird die Kovarianzmatrix $\hat{\mathbf{C}}$ analog Gleichung 3.13 aus Kapitel 3.2.3 definiert (vgl. [SS01]):

$$\hat{\mathbf{C}} \equiv \frac{1}{T} \sum_{t=0}^{T-1} \Phi(\vec{x}_t) \cdot \Phi(\vec{x}_t)^T \tag{3.56}$$

Wie in der klassischen PCA gilt es nun, die Paare von Eigenwerten λ und Eigenvektoren $\vec{b}$ der Kovarianzmatrix zu finden mit $\lambda > 0$ und $\vec{b} \in \mathcal{H} \neq \vec{0}$. Dazu muss folgendes Eigenwertproblem gelöst:

$$\lambda \vec{b} = \hat{\mathbf{C}} \vec{b} \tag{3.57}$$

Alle Lösungen $\vec{b}$ mit $\lambda > 0$ liegen in der linearen Hülle von $(\Phi(\vec{x}_0) \dots \Phi(\vec{x}_{T-1}))$. Gleichung 3.57 kann somit umformuliert werden zu:

$$\forall t = 0 \dots T-1 : \lambda \left\langle \Phi(\vec{x}_t), \vec{b} \right\rangle = \left\langle \Phi(\vec{x}_t), \hat{\mathbf{C}} \vec{b} \right\rangle \tag{3.58}$$

Zudem existieren die Koeffizienten α_i ($i = 0 \dots T-1$), mit denen sich die Eigenvektoren $\vec{b}$ als Linearkombinationen der $\Phi(\vec{x}_i)$ darstellen lassen:

$$\vec{b} = \sum_{i=0}^{T-1} \alpha_i \Phi(\vec{x}_i) \tag{3.59}$$

Kombiniert man 3.58 und 3.59, so erhält man (vgl. [SS01]):

$$\forall t = 0 \dots T-1 : \lambda \sum_{i=0}^{T-1} \alpha_i \left\langle \Phi(\vec{x}_t), \Phi(\vec{x}_i) \right\rangle = \frac{1}{T} \sum_{i=0}^{T-1} \alpha_i \left\langle \Phi(\vec{x}_t), \sum_{j=0}^{T-1} \Phi(\vec{x}_j) \left\langle \Phi(\vec{x}_j), \Phi(\vec{x}_i) \right\rangle \right\rangle \tag{3.60}$$

Unter Zuhilfenahme der Definition der Kernmatrix $\mathbf{K}$ aus Gleichung 3.55 und der dyadischen Schreibweise nach Gleichung 3.13 lässt sich 3.60 formulieren als:

$$T\lambda \mathbf{K} \vec{\alpha} = \mathbf{K}^2 \vec{\alpha} \qquad (\vec{\alpha} = (\alpha_0 \dots \alpha_{T-1})^T) \tag{3.61}$$

Es kann gezeigt werden, dass genau die Lösungen des Eigenwertproblems $T\lambda\vec{\alpha} = \mathbf{K}\vec{\alpha}$ zu den Lösungen für λ $\vec{\alpha}$ aus 3.61 führen (vgl. [SS01]). Dazu seien die ersten M Eigenwerte von $\mathbf{K}$, die größer Null sind, mit $\tilde{\lambda_m}$ bezeichnet und es gilt:

$$\tilde{\lambda_m} = T\lambda_m \tag{3.62}$$

Anschließend müssen die Eigenvektoren $\vec{\alpha}_i$ normalisiert werden, und zwar dergestalt, dass die korrespondierenden Hauptkomponenten $\vec{b}_i$ wie in der Standard-PCA die Länge Eins haben, d.h. insbesondere $< \vec{b}_i, \vec{b}_i >= 1$. Es ergibt sich eine Normalisierungsvorschrift für die ersten M Eigenvektoren $\vec{\alpha}_m$:

$$1 \equiv < \vec{\alpha}_m, \mathbf{K}\vec{\alpha}_m >= \tilde{\lambda}_m < \vec{\alpha}_m, \vec{\alpha}_m > \tag{3.63}$$

Die Lösungen obiger Gleichung sind die normalisierten Eigenvektoren $\tilde{\vec{\alpha}}_m$ mit:

$$\tilde{\vec{\alpha}}_m = \frac{\vec{\alpha}_m}{\sqrt{\tilde{\lambda}_m}} \qquad \text{(mit } \tilde{\lambda}_m > 0) \tag{3.64}$$

Die Extraktion der Merkmalsvektoren, im Folgenden mit $\vec{a}_t$ bezeichnet, gestaltet sich in der Kern-PCA analog zur klassischen Hauptkomponentenanalyse (vgl. Kap. 3.2.6), d.h. es erfolgt eine Projektion der Datenvektoren auf die Hauptkomponenten $\vec{b}_i$. Im Gegensatz zur PCA entsprechen die Datenvektoren dabei den bereits transformierten Ursprungsdaten, d.h. den $\Phi(\vec{x}_t)$.

Die i-te Komponente des Merkmalsvektors $\vec{a}_t$, welcher mit dem t-ten Datenvektor verknüpft ist, sei mit $\vec{a}_{t,i}$ bezeichnet und berechnet sich wie folgt:

$$\vec{a}_{t,i} \equiv < \Phi(\vec{x}_t), \vec{b}_i >= \sum_{m=0}^{M-1} \tilde{\alpha}_{i,m} < \Phi(\vec{x}_m), \Phi(\vec{x}_t) > \tag{3.65}$$

Das Skalarprodukt $< \Phi(\vec{x}_m), \Phi(\vec{x}_t) >$ wird unter Zuhilfenahme der Kernfunktion $k(\vec{x}_i, \vec{x}_j)$ berechnet und es ergibt sich:

$$\vec{a}_{t,i} = \sum_{m=0}^{M-1} \tilde{\alpha}_{i,m} k(\vec{x}_m, \vec{x}_t) = \sum_{m=0}^{M-1} \tilde{\alpha}_{i,m} \mathbf{K}_{mt} =< \tilde{\vec{\alpha}}_i, \vec{k}_t > \tag{3.66}$$

Hierbei ist $\vec{k}_t$ der t-te Spaltenvektor der Kernmatrix $\mathbf{K}$.

Die Menge aller Merkmalsvektoren lässt sich in einer kompakten Form als Matrix $\mathbf{A}$ darstellen:

$$\mathbf{A} = (\vec{a}_0 \dots \vec{a}_{T-1}) = \begin{pmatrix} \tilde{\vec{\alpha}}_0 \\ \vdots \\ \tilde{\vec{\alpha}}_{M-1} \end{pmatrix} \cdot \mathbf{K} \tag{3.67}$$

Eine Dimensionsreduktion erfolgt bei der Kern-PCA analog zur herkömmlichen PCA durch das Vernachlässigen von Hauptkomponenten sowie den zugehörigen Einträgen in den Merkmalsvektoren. Dabei gibt

der i-te Eigenwert wiederum den durch die zugehörige i-te Hauptkomponente erklärten Varianzanteil an der Gesamtvarianz an (vgl. Kap. 3.2.5).

Durch die Mächtigkeit der gewählten nichtlinearen Abbildung Φ, respektive der Kernfunktion $k(\vec{x}, \vec{y})$, können die Datensätze jedoch, bei gleicher ursprünglicher Varianzverteilung, mit weniger Hauptkomponenten bzw. Merkmalen beschrieben werden als in der Standard-PCA, d.h. im Allgemeinen gilt: $|\vec{a}_{t,KPCA}| < |\vec{a}_{t,PCA}|$. Die Wahl der verwendeten Kernfunktion hängt dabei vom konkreten Anwendungsfall ab (vgl. [SMS99] und [SS01]).

3.3.4 Ein Beispiel

Gegeben seien die Daten $\mathbf{X} = (\vec{x}_t | \vec{x}_t \in \mathbb{R}^2)$ als Punktwolke einer gleichverteilt abgestasteten Kreisscheibe mit dem Radius Eins um den Mittelpunkt $\underline{c} = (1, 1)^T$ (vgl. Abb 3.7). Es ist leicht zu erkennen, dass keine signifikante Richtung der größten Varianz existiert. Eine herkömmliche Hauptkomponentenanalyse liefert für die normalisierten Eigenwerte folgende Ergebnisse:

$$\tilde{\lambda}_0 = 0.53$$

$$\tilde{\lambda}_1 = 0.47$$

Nach der Interpretation des prozentualen Anteils an der Gesamtinformation eines Datensatzes (vgl. Kap. 3.2.5), sieht man, dass die beiden den Eigenwerten $\tilde{\lambda}_0$ und $\tilde{\lambda}_1$ zugehörigen Hauptkomponenten annähernd den gleichen Informationsgehalt aufweisen und somit eine Dimensionsreduktion im Sinne des Vernachlässigens einer dieser beiden Vektoren nicht zielführend ist.

Im Folgenden wird geprüft, ob die Kern-PCA ebendieses leisten kann. Die Wahl des Kerns fällt auf den weiter oben bereits erwähnten polynomiellen:

$$k(x, y) = < x, y >^2$$

Die dazugehörige Abbildung ist $\Phi : (x_0, x_1) \mapsto (x_0^2, x_1^2, \sqrt{2}x_0x_1)$. Sie bildet den zweidimensionalen Datenvektor $\vec{x}_t$ auf den dreidimensionalen Bildvektor $\hat{\vec{x}}_t$ ab. Abbildung 3.8 zeigt die Daten nach ihrer Transformation.

Die Kern-PCA liefert folgende (bereits normalisierten) Eigenwerte:

$$\tilde{\hat{\lambda}}_0 = 0.67$$

$$\tilde{\hat{\lambda}}_1 = 0.31$$

$$\tilde{\hat{\lambda}}_2 = 0.02$$

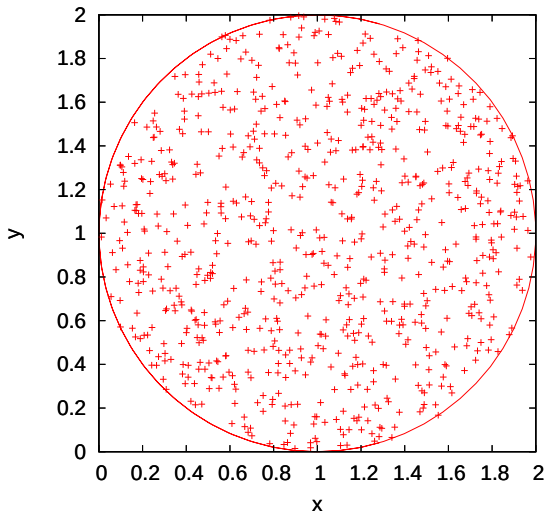

Abbildung 3.7: Die Punktwolke einer gleichverteilt abgetasteten Kreisscheibe.[7]

Hier werden aufgrund der höheren Dimensionalität des Bildraumes drei Eigenwerte ermittelt. Deren Werte weisen jedoch eine andere Verteilung als jene der klassischen PCA auf: Augenscheinlich ist der Betrag des ersten Eigenwertes bei der Kern-PCA höher als der bei der PCA, d.h. es wird ein größerer Anteil an der Gesamtinformationen durch die erste Hauptkomponente beschrieben. Dies bezieht sich jedoch auf den Informationsgehalt im *Bild*raum: Die *Bilder* der Daten können bei der Kern-PCA besser approximiert werden, als es die PCA im Raum der ursprünglichen Daten vermag.

Zur Vergleichbarkeit der Approximationsfehler e beider Verfahren wird die *relative mittlere quadratische Abweichung* als Fehlermaß verwendet[8]:

$$e \equiv \frac{\sum_t \left\| \vec{x}_t - \tilde{\vec{x}}_t \right\|^2}{\sum_t \left\| \vec{x}_t \right\|^2}$$

Es ergibt sich für die PCA ein Fehler $e_{\text{PCA}} = 0.471$ und für die Kern-PCA ein Fehler $e_{\text{KPCA}} = 0.328$.

Zu beachten ist, dass der Fehler im Falle der Kern-PCA für die Rekonstruktion im Bildraum berechnet wurde. Die Approximation nach der Rücktransformation in den ursprünglichen Raum muss deswegen nicht zwangsläufig besser sein als die der PCA.

[7] Anm.: Der eingezeichnete Umkreis der Punktwolke dient lediglich der Illustration und ist nicht Teil der Daten.

[8] Anm.: Die hier verwendete Definition der relativen mittleren quadratischen Abweichung setzt die Mittelwertfreiheit der Daten voraus, welche im Falle der PCA resp. Kern-PCA durch den Zentrierungsschritt zunächst gegeben ist. Ergo sollte die Fehlerrechnung während des Rekonstruktionsschrittes vor der Rückrechnung der Mittelwerte erfolgen.

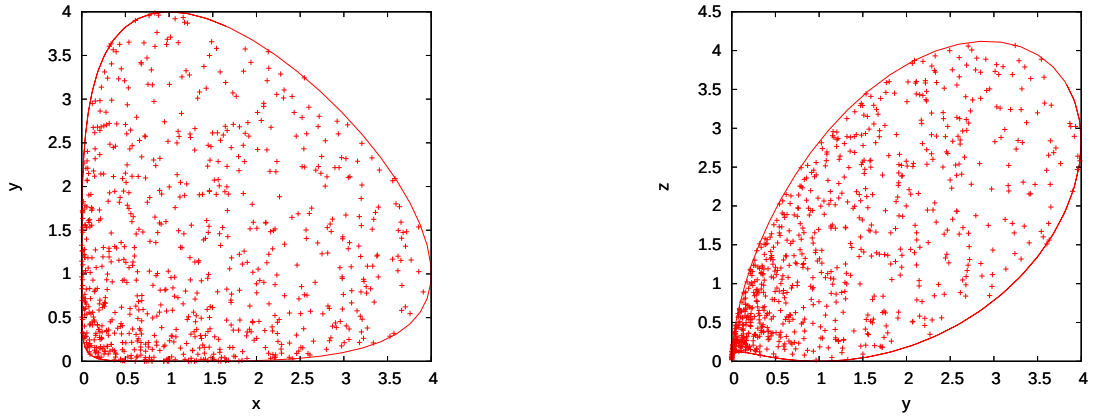

Abbildung 3.8: Die Daten aus Abb. 3.7 nach der nichtlinearen Transformation Φ in der xy-Ebene (links) und der yz-Ebene (rechts).

3.3.5 Herausforderungen im Hochdimensionalen

3.3.5.1 Indirekte Mittelwertbereinigung

Bisher erfolgte die Anwendung der Kern-PCA stets unter der Annahme, dass die Vektoren $\Phi(\vec{x}_t)$ bereits in einer mittelwertbereinigten Form vorliegen. In der Praxis kann hiervon allerdings nicht ausgegangen werden – es stellt sich demnach die Frage, wie eine solche Zentrierung erreicht werden soll.

Eine Mittelwertbereinigung der Datenreihe im Ursprungsraum, d.h. vor der Transformation in den hochdimensionalen Raum $\mathcal{H}$, ließe sich effizient bewerkstelligen, jedoch kann sie die Zentriertheit der Daten in $\mathcal{H}$ nicht erzielen, da diese im Allgemeinen durch die Abbildung Φ verloren geht. Die explizite Mittelwertbereinigung im Merkmalsraum $\mathcal{H}$ ist unter Umständen zu rechenintensiv, da die Komplexität von Φ nicht beschränkt ist (vgl. Kap. 3.3.1).[9]

Eine Mittelwertbereinigung kann jedoch indirekt über eine Modifikation der Kernmatrix $\mathbf{K}$ erfolgen. Es wird eine Matrix $\tilde{\mathbf{K}}$ wie folgt gebildet (vgl. [SS01]):

$$\tilde{\mathbf{K}}_{ij} = (\mathbf{K} - \mathbf{1}_T\mathbf{K} - \mathbf{K}\mathbf{1}_T + \mathbf{1}_T\mathbf{K}\mathbf{1}_T)_{ij} \tag{3.68}$$

Hierbei steht die Kurzschreibweise $\mathbf{1}_T$ für die komponentenweise durch T geteilte Einheitsmatrix, d.h. $(\mathbf{1}_T)_{ij} = \frac{1}{T} \cdot \mathbb{1}_{ij}$.

Wird anstelle von $\mathbf{K}$ die modifizierte Kernmatrix $\tilde{\mathbf{K}}$ verwendet und die Eigenvektoren $\tilde{\vec{\alpha}}_m$ von $\tilde{\mathbf{K}}$ berechnet, so entspricht dies einer Kern-PCA mit $\mathbf{K}$ unter Verwendung bereits zentrierter $\Phi(\vec{x}_t)$.

[9] Anm.: Genau diesem Umstand ist schließlich die Anwendung des sog. *Kerntricks* geschuldet: Der KPCA-Algorithmus umgeht die explizite Berechnung der $\Phi(\vec{x}_t)$ durch die Ausnutzung der Kernfunktionen.

3.3.5.2 Das Urbild-Problem

Der Merkmalsraum $\mathcal{H}$ ist die lineare Hülle aller Bilder der Transformation $\Phi(\vec{x}_t)$:

$$\mathcal{H} \equiv \text{span}(\Phi(\vec{x}_t)) \tag{3.69}$$

Die Menge aller Abbildungen Φ ist dabei eine Untermenge von $\mathcal{H}$ (vgl. Abb. 3.9), was bedeutet, dass nicht für alle Linearkombinationen von $\Phi(\vec{x}_t)$ ein Urbild existieren kann.

Sei Ψ ein Vektor in $\mathcal{H}$, welcher aus der Linearkombination von $\Phi(\vec{x}_t)$ entsteht:

$$\Psi = \sum_{i=0}^{T-1} \tilde{\alpha}_i \Phi(\vec{x}_i) \tag{3.70}$$

Man sieht, dass Ψ nicht notwendigerweise das Bild $\Phi(\vec{x}_i)$ eines zugehörigen Datenvektors $\vec{x}_i$ des Eingaberaums sein muss. Dies trifft insbesondere auf die Hauptkomponenten zu (vgl. Gl. 3.59), d.h. es gibt im Allgemeinen im Eingaberaum keine entsprechende Darstellung der korrespondierenden Hauptkomponente aus dem Bildraum.[10]

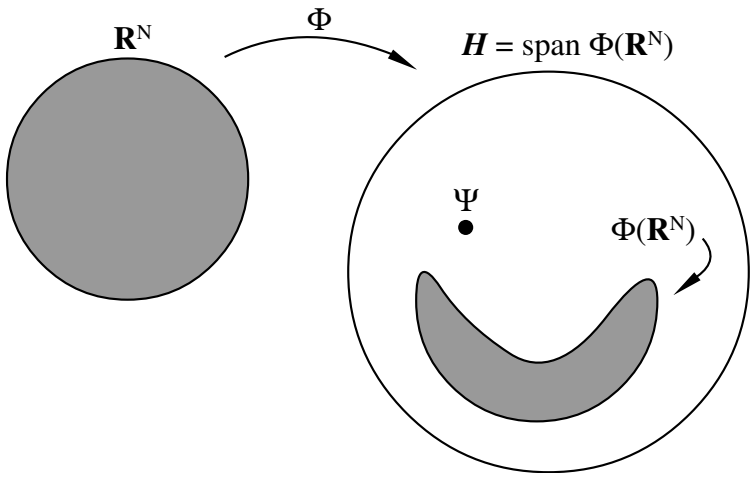

Abbildung 3.9: Das Urbildproblem: Ein Vektor $\Psi \in \mathcal{H}$, welcher aus Linearkombinationen von $\Phi(\vec{x}_i)$ gebildet wird, muss keine Entsprechung $\vec{z} = \Phi^{-1}(\Psi)$ im Urbildraum $\mathbb{R}^N$ haben (vgl. [SS01] S.545).

[10] Anm.: Außer für den trivialen Fall, bei dem für alle Hauptkomponenten gilt, dass der Merkmalswert zur zugehörigen Hauptkomponente gleich Eins und alle restlichen Merkmalswerte gleich Null sind. Dann allerdings wäre $\Psi_t = \Phi(\vec{x}_t)$.

3.3.6 Rekonstruktion

Viele Disziplinen, in denen die Kern-PCA Anwendung findet – wie z.B. die Mustererkennung – bedienen sich ihrer allein zur Berechnung der Merkmalsvektoren im Bildraum. Dass dies effizient über die Kernfunktionen erfolgen kann, wurde in Kap. 3.3.3 beschrieben. Sollen die Daten jedoch zu einem späteren Zeitpunkt rekonstruiert werden, so benötigt man neben den Merkmalen zusätzlich die Hauptkomponenten, welche jedoch aufgrund der unbeschränkten Komplexität von $\mathcal{H}$ nicht explizit berechnet werden können und für die es unter Umständen keine Entsprechung im Urbildraum gibt (vgl. Kap. 3.3.5.2). Das folgende Kapitel zeigt, wie es für ausgewählte Kernfunktionen dennoch möglich ist, eine Rekonstruktion durchzuführen.

Die Rekonstruktion der Daten im Bildraum erfolgt analog zur Standard-PCA: Die Merkmalsvektoren werden auf die Hauptkomponenten projiziert und man erhält das Bild $\Phi(\vec{x}_t)$ des rekonstruierten Datenvektors:

$$\Phi(\vec{x}_t) = \sum_{i=0}^{M-1} a_{t,i} \cdot \vec{b}_i \tag{3.71}$$

Den rekonstruierten Datenvektor im Eingaberaum erhält man schließlich durch die Anwendung der Umkehrabbildung Φ^{-1}. Durch die unbeschränkte Komplexität von Φ ist dies nicht in jedem Fall direkt möglich, durchaus jedoch für bestimmte Gruppen von Kernfunktionen.

Im Folgenden wird davon ausgegangen, dass die Rekonstruktionen der durch die Kern-PCA komprimierten Daten im Eingaberaum dargestellt werden können und es wird ein Verfahren zur exakten Rekonstruktion der Urbilder vorgestellt. Im anderen Falle können darüber hinaus approximierende Techniken angewendet werden.

3.3.6.1 Exakte Rekonstruktion

Sei Ψ eine Linearkombination von Bildern $\Phi(\vec{x}_i)$ in $\mathcal{H}$:

$$\Psi = \sum_{j=0}^{T-1} \hat{\alpha}_j \Phi(\vec{x}_j) \tag{3.72}$$

Existert zudem ein Vektor $\vec{z}$ ($\vec{z} \in \mathbb{R}^N$), so dass $\Phi(\vec{z}) = \Psi$ gilt, und eine invertierbare Funktion f_k mit

$f_k(< \vec{x}, \vec{x}' >) = k(\vec{x}, \vec{x}')$, dann gibt es eine Zerlegung von $\vec{z}$ wie folgt (vgl. [SS01]):

$$\begin{aligned} \vec{z} &= \sum_{i=0}^{N-1} < \vec{z}, \vec{e}_i > \vec{e}_i \\ &= \sum_{i=0}^{N-1} f_k^{-1}(k(\vec{z}, \vec{e}_i))\vec{e}_i \\ &= \sum_{i=0}^{N-1} f_k^{-1} \left(\sum_{j=0}^{T-1} \hat{\alpha}_j k(\vec{x}_j, \vec{e}_i) \right) \vec{e}_i \end{aligned} \tag{3.73}$$

Die $\vec{e}_i$ (mit $\vec{e}_i \in \mathbb{R}^N$ und $i = 0 \ldots N-1$) werden hierbei so gewählt, dass sie im Eingaberaum eine orthonormale Basis der Dimension N bilden.

Die Wahl von f_k ist entscheidend für das Gelingen des Verfahrens, da die Invertierbarkeit sichergestellt sein muss.

Kerne, welche eine solche Umkehrabbildung f_k^{-1} besitzen, sind unter anderem:

- Polynomielle Kerne: $k(\vec{x}, \vec{x}') = (< \vec{x}, \vec{x}' > +c)^d$ mit $c \geq 0$ und d ungerade
- Sigmoid-Kerne: $k(\vec{x}, \vec{x}') = \tanh(\kappa < \vec{x}, \vec{x}' > +\theta)$ mit $\kappa, \theta \in \mathbb{R}$

Nun müssen noch die $\hat{\alpha}_i$ gefunden, so dass $\Phi(\vec{z}) = \Psi = \sum_{j=0}^{T-1} \hat{\alpha}_i \Phi(\vec{x}_i)$ und die Zerlegung nach Gleichung 3.73 erfolgen kann. Dazu sei zunächst noch einmal die Definition eines Datenvektors im Bildraum mithilfe der Hauptkomponenten betrachtet. Dieser ist laut Gleichung 3.71:

$$\Phi(\vec{x}_t) = \sum_{i=0}^{M-1} a_{t,i} \cdot \vec{b}_i \tag{3.74}$$

Weiterhin kann die Hauptkomponente $\vec{b}_i$ selbst wiederum als Linearkombination der (normalisierten) Eigenvektoren der Kernmatrix dargestellt werden (vgl. Gl. 3.59):

$$\vec{b}_i = \sum_{j=0}^{T-1} \tilde{\alpha}_{i,j} \cdot \Phi(\vec{x}_j) \tag{3.75}$$

Setzt man nun Gl. 3.75 in Gl. 3.74 ein, so erhält man eine neue Formulierung für die $\Phi(\vec{x}_t)$:

$$\begin{aligned} \Phi(\vec{x}_t) &= \sum_{i=0}^{M-1} a_{t,i} \sum_{j=0}^{T-1} \tilde{\alpha}_{i,j} \Phi(\vec{x}_j) \\ &= \sum_{j=0}^{T-1} \underbrace{\left(\sum_{i=0}^{M-1} a_{t,i} \tilde{\alpha}_{i,j} \right)}_{\hat{\alpha}_j} \Phi(\vec{x}_j) \end{aligned} \tag{3.76}$$

Setzt man $\hat{\alpha}_j = \sum_{i=0}^{M-1} a_{t,i}\tilde{\alpha}_{i,j}$ und $\Psi = \Phi(\vec{z}) = \Phi(\vec{x}_t)$, so erhält man für Ψ die Form aus Gleichung 3.72 und es kann nun eine Rekonstruktion nach Gl. 3.73 erfolgen:

$$\begin{aligned}\vec{z} &= \sum_{i=0}^{N-1} f_k^{-1}\left(\sum_{j=0}^{T-1}\hat{\alpha}_j k(\vec{x}_j,\vec{e}_i)\right)\vec{e}_i \\ &= \sum_{i=0}^{N-1} f_k^{-1}\left(\sum_{j=0}^{T-1}\sum_{l=0}^{M-1} a_{t,l}\tilde{\alpha}_{l,j} k(\vec{x}_j,\vec{e}_i)\right)\vec{e}_i\end{aligned} \tag{3.77}$$

Die Methode der exakten Rekonstruktion hat jedoch einen entscheidenden praktischen Nachteil. Es muss davon ausgegangen werden, dass die Daten im Bildraum schon mittelwertbereinigt wurden. Dieses Problem wurde bereits in Kap. 3.3.5.1 thematisiert. Die Zentrierung kann indirekt ausgeführt werden, jedoch liegen die Mittelwerte dann nicht in einer nutzbaren expliziten Form vor, um vor der Anwendung der Umkehrfunktion f_k^{-1} die Bilder $\Phi(\vec{x}_t)$ wieder in ihre ursprüngliche Lage im Raum $\mathcal{H}$ zu bringen.

3.3.6.2 Urbild-Approximation

Da im Allgemeinen nicht davon ausgegangen werden kann, dass die Linearkombination Ψ von Datenvektoren im Bildraum auf einen Datenvektor $\vec{z}$ im Urraum zurückgeführt kann, ist die exakte Rekonstruktion hinsichtlich der praktischen Anwendbarkeit ein eher unbefriedigender Ansatz.

Eine Antwort hierauf liegt im Auffinden eines Vektors $\vec{z}$ im Eingaberaum, dessen Bild $\Phi(\vec{z})$ die Linearkombination Ψ im Sinne der kleinsten Fehlerquadrate approximiert, ergo die Minimierung des Terms $\|\Psi - \Phi(\vec{z})\|^2$. Allgemeiner kann man formulieren, dass nicht der quadratische Abstand zwischen Ψ und $\Phi(\vec{z})$, sondern $\|\Psi - \beta\Phi(\vec{z})\|^2$ minimiert werde, d.h. dass eine zusätzliche lineare Skalierung von $\Phi(\vec{z})$ zulässig ist (vgl. [SS01]). Die Lösung letzteren Minimierungszieles ist die orthogonale Projektion von Ψ auf die lineare Hülle von $\Phi(\vec{z})$ (vgl. Abb. 3.10). Der Abstand sei dabei mit ϵ bezeichnet und es gilt:

$$\begin{aligned}\epsilon &\equiv \left\| \frac{<\Psi,\Phi(\vec{z})>}{<\Phi(\vec{z}),\Phi(\vec{z})>}\cdot\Phi(\vec{z}) - \Psi \right\|^2 \\ &= \|\Psi|^2 - \frac{<\Psi,\Phi(\vec{z})>^2}{<\Phi(\vec{z}),\Phi(\vec{z})>}\end{aligned} \tag{3.78}$$

Die Minimierung von ϵ geht mit der Maximierung des Terms $\frac{<\Psi,\Phi(\vec{z})>^2}{<\Phi(\vec{z}),\Phi(\vec{z})>}$ einher, welcher jedoch mittels Kernfunktionen dargestellt werden kann, da er ausschließlich Skalarprodukte enthält.

Schölkopf [SS01] stellt iterative Algorithmen vor, welche approximierende Urbilder in diesem Sinne finden, z.B. einen Fixpunktiterationsansatz für Kerne radialer Basisfunktionen. Ein weiterer Ansatz wird durch Shawn Martin [Mar06] beschrieben. Er entwickelt eine approximierende Variante der Kern-PCA, bei welcher der Basiswechsel durch eine Umformulierung des Gram-Schmidt-Orthonormalisierungsverfahrens gewonnen wird. Eine weitere Transformation ermöglicht, dass die Vektoren des Bildraumes

durch Linearkombinationen von Ortsvektoren des Eingaberaumes ausgedrückt werden können. Durch diese Verfahrensweise kann eine Darstellung der Hauptkomponenten in $\mathcal{H}$ sowie die Rekonstruktion erreicht werden.

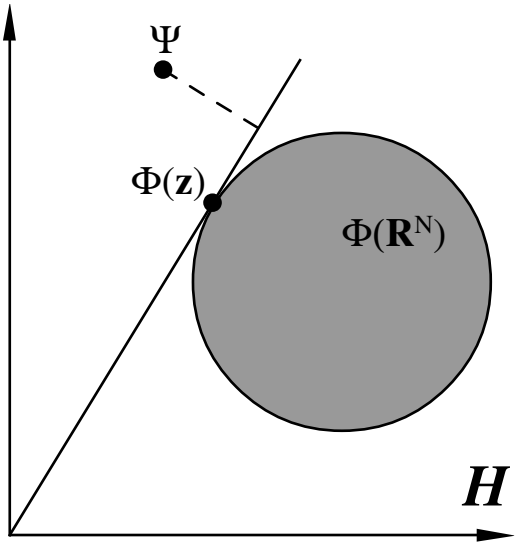

Abbildung 3.10: Auffinden von approximierenden Urbildern durch das Minimieren der Distanz zwischen Ψ und der Projektion von Ψ auf span$(\Phi(\vec{z}))$ (vgl. [SS01] S.547).

3.4 Interaktive Simulation

Im folgenden Kapitel wird ein einfaches Modell zur parametrisierten Manipulation von *level set*-Animationen entwickelt. Als Zielstellung soll hierbei, anstelle der physikalisch korrekten Simulation, die überzeugende visuelle Anmutung der Fluidoberflächenbewegung in den Vordergrund treten; die Anforderungen des Anwendungsgebietes sind Interaktion und Echtzeitfähigkeit. Die ursprüngliche Aufgabenstellung dieser Arbeit war, die Ansätze und Methoden A. Treuilles [TLP06] auf Fluidoberflächen anzuwenden, insbesondere alle notwendigen Simulationsberechnungen im reduzierten Raum effizient mit wenigen Unbekannten durchzuführen. In solchen reduzierten Fluidmodellen (vgl. Kap. 2) wird dazu häufig die sogenannte Galerkin-Projektion angewendet:

Die Galerkin-Projektion

Neben der *Dimensions*reduktion der Datenvektoren einer Zeitreihe ermöglicht die Methode der Galerkin-Projektion eine *Modell*-Reduktion, sodass die Zeitentwicklungsfunktion der Simulation im reduzierten Merkmalsraum berechnet werden kann. Es sei durch ein Dimensionsreduktionsverfahren eine Projektion $\mathbf{B} : \vec{x} \rightarrow \tilde{\vec{x}}$ in den reduzierten Raum gegeben sowie die Rückprojektion $\mathbf{B}^{-1} : \tilde{\vec{x}} \rightarrow \vec{x}$ des reduzierten

Vektors in den Raum der Datenvektoren[11]. Die Zeitentwicklung der Simulation lässt sich als gewöhnliches Differentialgleichungssystem beschreiben:

$$\dot{x} = \mathbf{F}(\vec{x}) \tag{3.79}$$

Es soll nun ein Analogon im reduzierten Raum gefunden werden:

$$\dot{\tilde{x}} = \tilde{\mathbf{F}}(\tilde{\vec{x}}) \tag{3.80}$$

$\tilde{\mathbf{F}}$ entsteht durch die Anwendung der Galerkin-Projektion von $\mathbf{F}$ in den reduzierten Raum mithilfe von $\mathbf{B}$ (vgl. [TLP06]):

$$\tilde{\mathbf{F}} = \mathbf{B} \circ \mathbf{F} \circ \mathbf{B}^{-1} \tag{3.81}$$

Somit kann die Zeitentwicklung der Simulation effizient im reduzierten Raum berechnet werden.

3.4.1 Finden eines Modells

Osher und Fedkiw [OF02] stellen Methoden vor, wie die Evolution einer Oberfläche berechnet werden kann, solange die Geschwindigkeitsvektoren des Fluidvektorfeldes in der unmittelbaren lokalen Umgebung, wenigsten jedoch die Geschwindigkeitsvektoren an der Oberfläche selbst, gegeben sind. Im Rahmen dieser Arbeit liegen die Simulationszeitreihen nur in Form von *level sets* vor; das die Fluid-Bewegung beschreibende Vektorfeld ist nicht gegeben. Die Frage ist nun, ob sich dennoch eine Zeitentwicklung mithilfe von Differentialgleichungen formulieren lässt. Bislang konnte diese jedoch nicht gefunden werden.

Ein einfaches heuristisches Modell

Zur Gewinnung eines einfachen parametrisierbaren Modells soll die Interpretation der Hauptkomponenten und Merkmalsvektoren herangezogen werden. Durch die Anwendung der Hauptkomponentenanalyse wird die Zeitreihe in ein Koordinatensystem transformiert, in welchem die Basisvektoren – die Hauptkomponenten – entlang der größten Varianzen liegen. Eine plausible visuelle Interpretation dieser Hauptkomponenten: Sie repräsentieren die Schwingungsmoden des Fluids (vgl. Abb. 3.11). Diese Interpretation wird zusätzlich durch die korrespondierenden Merkmale gestützt, deren zeitliche Verläufe als Schwingungen aufgefasst werden können (vgl. Abb. 3.13). Dabei weisen die den ersten Hauptkomponenten zugeordneten Merkmale Schwingungen mit niedriger Frequenz und hoher Amplitude auf, was

[11] Anm.: Im Falle der PCA ist $\mathbf{B}$ die Matrix, deren Spalten die Hauptkomponenten und es gilt: $\mathbf{B}^{-1} = \mathbf{B}^{T}$.

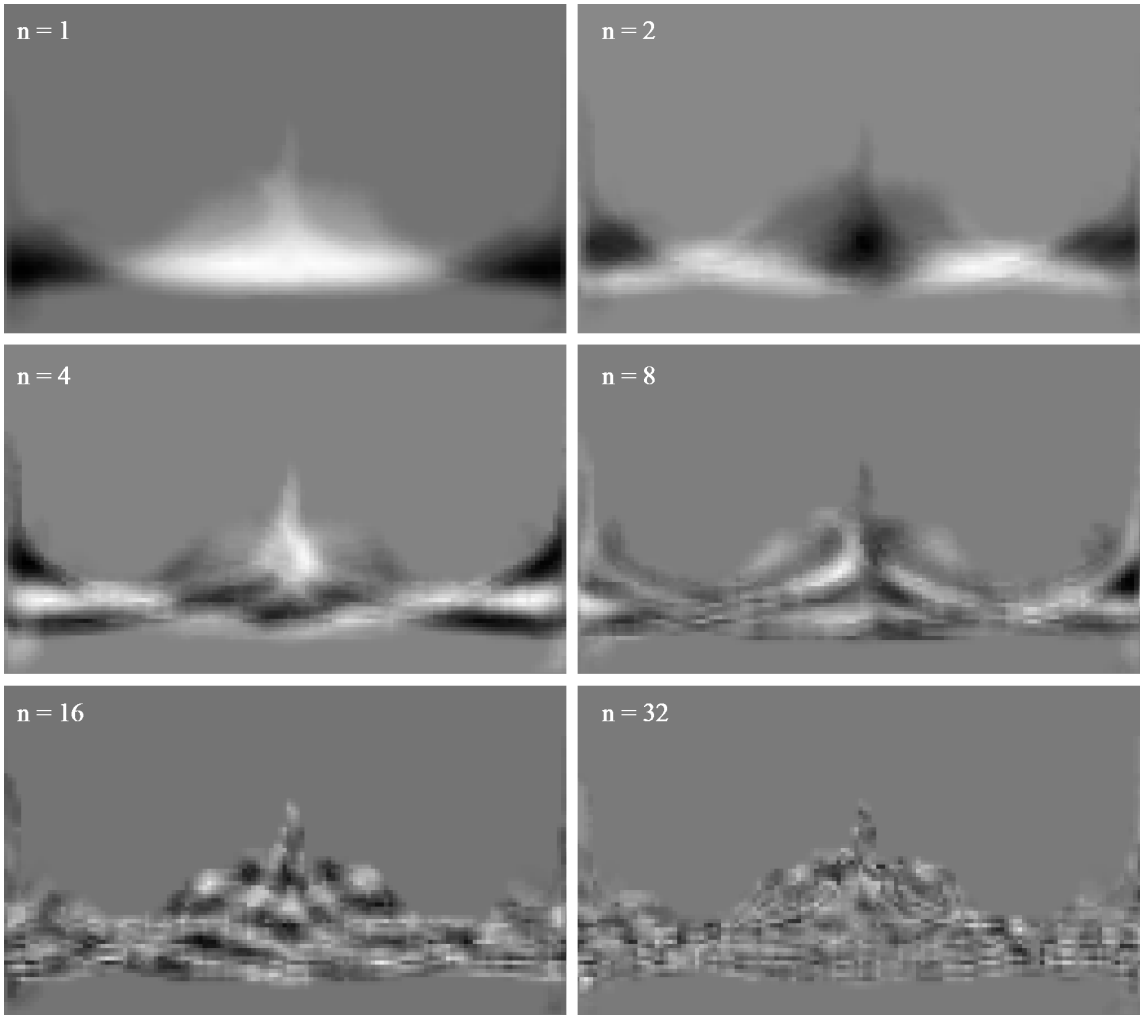

Abbildung 3.11: Visualisierung der Hauptkomponenten einer Testzeitreihe. Interpretation als Schwingungsmoden: Die n-te Hauptkomponente beschreibt für kleine n die Orte der Dynamik niederfrequenter Moden, für große n beschreibt sie die Orte der Dynamik hochfrequenter Moden.

genau dem Optimierungsziel der PCA entspricht[12], vice versa weisen die zeitlichen Verläufe der Merkmale, welche den höheren Hauptkomponenten zugeordnet sind, höhere Frequenzen und kleinere Amplituden auf. Abbildung 3.12 illustriert diesen Sachverhalt und stellt für alle n Merkmalsvektoren einer Testzeitreihe die maximale, minimale und mittlere Amplitude dar. Abbildung 3.13 zeigt für ausgewählte Basisvektoren $\vec{b}_n$ die Zeitentwicklungen der zugehörigen Merkmale, d.h. die n-ten Einträge über allen Merkmalsvektoren entlang der Zeit. Zu erkennen sind die schwingungsähnlichen Amplitudenverläufe mit steigender Frequenz sowie immer kleiner werdender Amplituden für wachsende n.

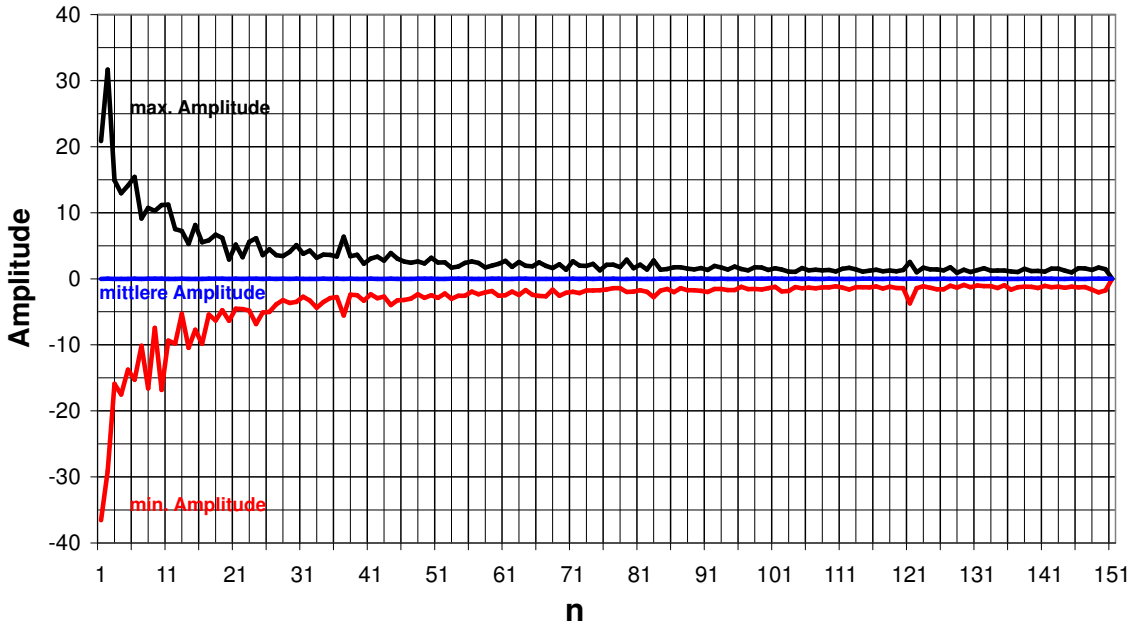

Abbildung 3.12: Maximale, minimale und mittlere Amplitude der Zeitentwicklung des n-ten Merkmals einer Testzeitreihe.

Der Modellansatz besteht darin, die Zeitentwicklungen der Merkmale durch analytische Funktionen zu substituieren. Dabei werden die Schwingungsverläufe durch die Summe von Harmonischen angenähert, d.h. die Zeitenwicklung der Amplitude eines Merkmals wird durch akkumulierte Sinusschwingungen beschrieben, deren Frequenzen ganzzahlige Vielfache einer Grundfrequenz sind[13]. Die Gesamtheit aller Teilschwingungen bildet dann eine Approximation der Zeitentwicklung der Fluidoberfläche, wobei sich die Teilschwingungen nach dem Superpositionsprinzip in ihrer Überlagerung nicht beeinflussen. Dieses Modell lässt sich aufgrund der Darstellung der Datenvektoren, wie sie nach der Hauptkomponentenanalyse vorliegen, leicht anwenden. Ein Datenvektor ist nach der Ausführung der PCA in folgender Form

[12] Anm.: Die Amplitude des Merkmals beschreibt die Stärke der korrespondierenden Hauptkomponente bezüglich des rekonstruierten Datensatzes (vgl. Gl. 3.2). Hohe Amplituden in den ersten Merkmalen spiegeln den Fakt wider, dass hohe Varianzen durch die ersten Hauptkomponenten repräsentiert werden.

[13] Anm.: Dieses Approximationsmodell ist u.a. in der Akustik bekannt (vgl. [Kut04]).

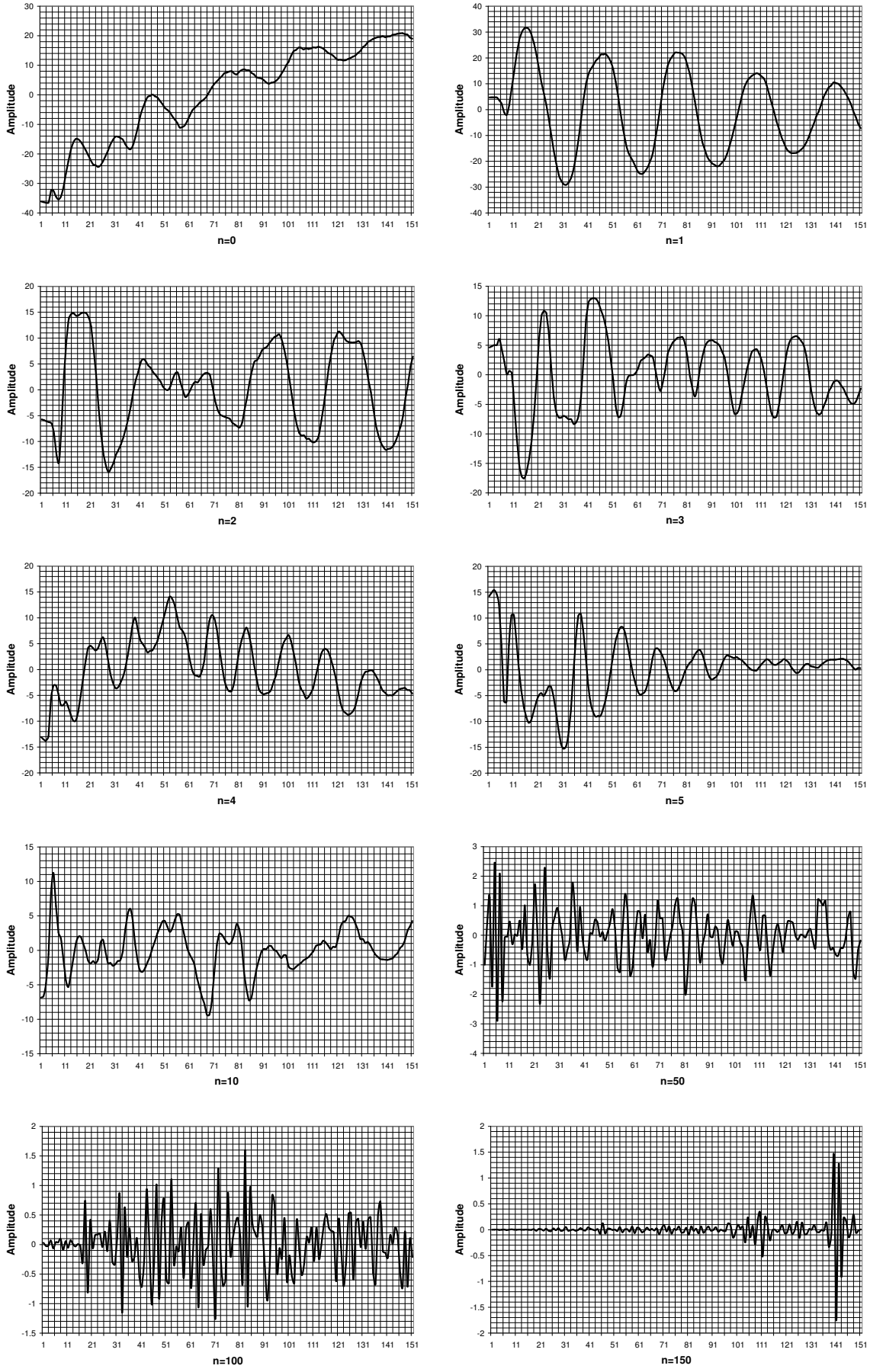

Abbildung 3.13: Zeitlicher Verlauf der Amplitude des n-ten Merkmals einer Testzeitreihe.

gegeben (vgl. Gl. 3.3):

$$\tilde{\vec{x}}_j = \sum_{i=0}^{M-1} a_i \cdot \vec{b}_i \tag{3.82}$$

Die Merkmale a_i werden nun durch die stetigen Funktionen $y_i(t)$ angenähert:

$$a_i \approx y_i(t = j) \tag{3.83}$$

Dabei werden die Funktionen $y_i(t)$ wie folgt definiert:

$$y_i(t) \equiv o_i + \hat{y}_i \cdot \sum_{n=0}^{n_i} \sin(2\pi n f_i t + \varphi_i) \tag{3.84}$$

Hierbei definiert n_i die Anzahl der Harmonischen, $\hat{y}_i$ die Amplitude der Grundschwingung, f_i die Grundfrequenz, φ_i die Phase und o_i einen Offset-Wert.

Um das Modell flexibler zu gestalten, wird im Folgenden die Amplitudenfunktion $y_i(t)$ um zusätzliche Terme erweitert. Der Schwingungsvorgang der Fluidoberfläche wird als Relaxationsprozess aufgefasst – ein Prozess, welcher nach Anregung mit fortschreitender Zeit in einen Gleichgewichtszustand übergeht. Zunächst wird die Amplitudenfunktion $y_i(t)$ mit einem Exponentialterm erweitert:

$$y_i(t) = o_i + e^{-t\delta_i} \cdot \hat{y}_i \cdot \sum_{n=0}^{n_i} \sin(2\pi n f_i t + \varphi_i) \tag{3.85}$$

Der Term $e^{-t\delta_i}$ mit δ_i als Dämpfungskonstante erzielt das Abklingen des Schwingungsvorgangs, $y_i(t)$ wird zur *gedämpften Schwingung*. Im nächsten Schritt wird ein Exponentialterm mit der Konstanten β_i zur Bedämpfung der relativen Spektralanteile der Harmonischen eingeführt. Die Funktion $y_i(t)$ wird dann zur Summe von gedämpften Partialschwingungen:

$$y_i(t) = o_i + e^{-t\delta_i} \cdot \hat{y}_i \cdot \sum_{n=0}^{n_i} e^{-(n-1)\beta_i} \sin(2\pi n f_i t + \varphi_i) \tag{3.86}$$

Abbildung 3.14 zeigt beispielhaft die Zeitentwicklung einer Oszillation $y_i(t)$ aus überlagerten Harmonischen mit $n_i = 9$, $\hat{y}_i = 1$, $f_i = 1/100$, $\delta_i = 0.004$, $\beta_i = 0.2$, $\varphi_i = 0$ und $o_i = 0$ sowie deren Linienspektrum.

Der Datenvektor $\vec{x}_j$ kann nun mithilfe von M Paaren von Funktionen $y_i(t)$ und Hauptkomponenten b_i approximiert werden:

$$\tilde{\vec{x}}_j \approx \sum_{i=0}^{M-1} y_i(t = j) \cdot \vec{b}_i \tag{3.87}$$

$$\tilde{\vec{x}}_j \approx \sum_{i=0}^{M-1} \left(o_i + e^{-t\delta_i} \cdot \hat{y}_i \cdot \sum_{n=0}^{n_i} e^{-(n-1)\beta_i} \sin(2\pi n f_i j + \varphi_i) \right) \cdot \vec{b}_i \tag{3.88}$$

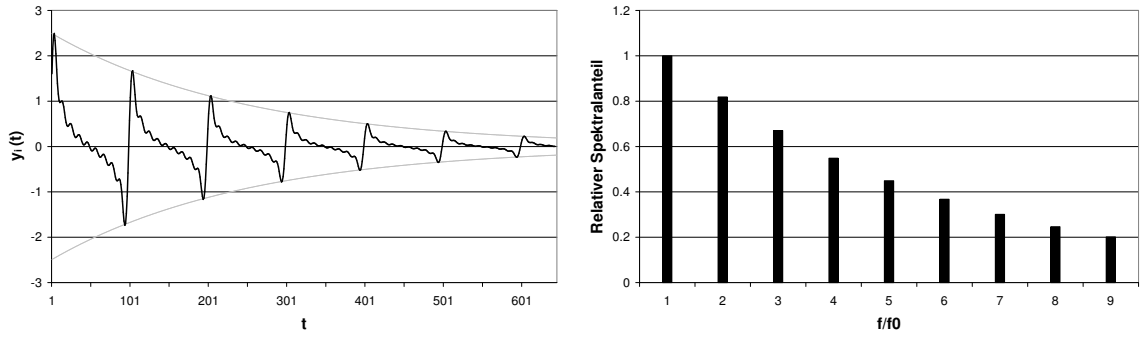

Abbildung 3.14: Links: Amplitudenfunktion $y_i(t)$ als Überlagerung von Harmonischen. Rechts: Relative spektrale Anteile der Partialschwingungen von $y_i(t)$ bezogen auf die Grundfrequenz f_0.

4 Prototypische Umsetzung

Um die die PCA in der Praxis anwenden und ihre Eigenschaften studieren zu können, wurde ein wissenschaftlicher Prototyp auf OpenGL-Basis entwickelt. Die Implementierung der Analysesoftware erfolgte in C++ und setzt auf dem *GLUT*-Framework[1] [KR01], welches die Anbindung an das Fenstersystem mitsamt OpenGL-Kontext bereitstellt und das Eventhandling von Tastatur- und Mausereignissen verwaltet sowie der *GLUI* [2] [RSB06], zur Bereitstellung der graphischen Benutzeroberfläche mit einer Reihe vordefinierter Steuerelemente, auf. Beide Frameworks stellen für kleine bis mittlere Projekte einen guten Kompromiss aus Funktionalität und einfacher Einbindung dar. Die Emulation der nativen Betriebssystem-Steuerelemente durch OpenGL-Pendants macht sie zudem plattformunabhängig.

Für die notwendigen mathematischen Berechnungen, insbesondere für die Matrizenoperationen sowie die SVD-Dekomposition, wurde auf die Bibliotheken des lehrstuhleigenen CGV-Frameworks zurückgegriffen.

4.1 Layout

Der Prototyp ist aufgrund der Restriktionen des GLUI-Frameworks als Mehr-Fenster-Anwendung realisiert[3], wodurch dem Nutzer eine freie Anordnung der einzelnen Unterfenster auf dem Bildschirm ermöglicht wird. Die Applikation ist in zwei primäre Ansichten gegliedert: In einem Menüfenster werden alle Programmfunktionalitäten verwaltet, während in einem OpenGL-Fenster die Zeitreihen visualisiert werden.

4.1.1 Menüfunktionen

Vom Menüfenster lassen sich sämtliche Funktionen des Prototyps erreichen und alle relevanten Unterfenster aufrufen. Zur platzsparenden Gestaltung der Bedienelemente wurden bei der Implementierung

[1] Anm.: Die Abkürzung GLUT steht für Open**GL** **U**tility **T**oolkit.

[2] Anm.: Die Abkürzung GLUI steht für Open**GL** **U**ser **I**nterface Library.

[3] Anm.: GLUT bietet neben dem standardmäßigen OpenGL-Viewport lediglich ein Kontextmenü als Steuerelement an, während sich in durch GLUI erzeugten Fenstern neben den vorgefertigten Steuertypen keine weiteren OpenGL-Inhalte darstellen lassen

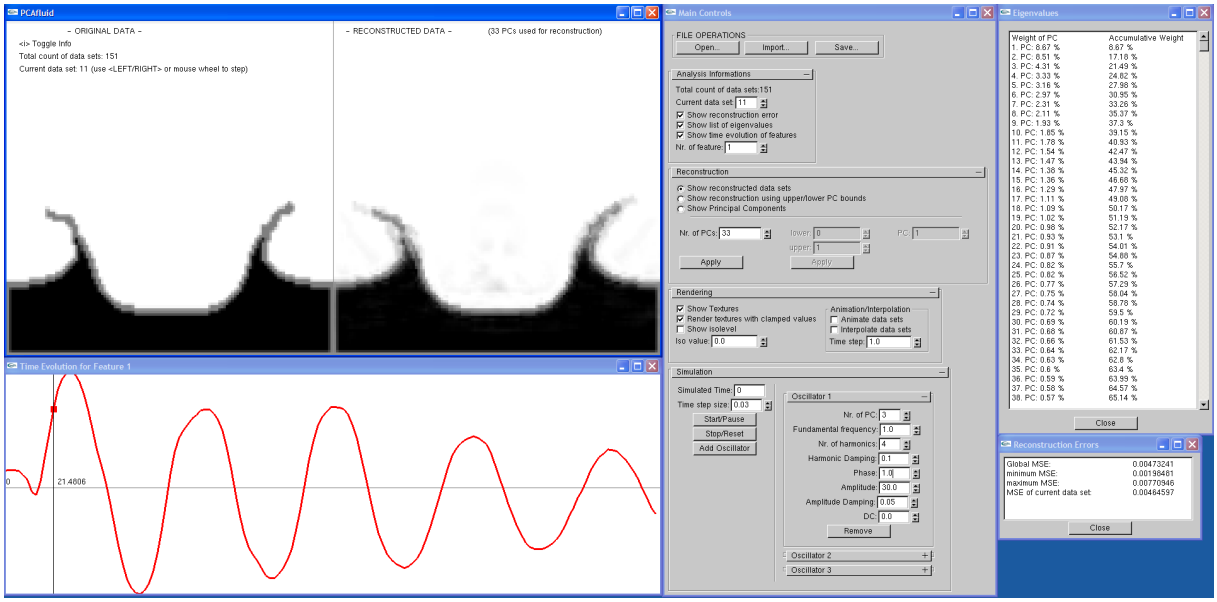

Abbildung 4.1: Die Hauptansicht der Anwendung.

sogenannte *Rollouts* verwendet, welche durch das GLUI-Framework als Alternative zu den bekannteren *Tabs* bereitgestellt werden. Die Zuordnung der einzelnen Steuerelemente zu den Rollouts erfolgt nach funktionaler Aufgabe, es lassen sich dabei fünf Hauptbereiche festlegen:

- Dateifunktionen
- Analyse-Informationen
- Rekonstruktion
- Rendering
- Simulation

Dateifunktionen

Der PCA-Algorithmus mit sämtlichen zugehörigen Hilfsstrukturen wird durch eine selbstverwaltende Klasse gekapselt, welche u.a. ein Interface zum Laden und Speichern bereitstellt. Neben allen relevanten Variablen können insbesondere die Hauptkomponenten und Merkmalsvektoren in eine Datei gesichert werden. Dies erfolgt zustandsbasiert, sodass bei erneutem Laden alle Nutzereingaben, wie z.B. die Anzahl der zur Rekonstruktion verwendeten Hauptkomponenten, wiederhergestellt werden können.

Das Importieren von Zeitreihen wird über das Hauptmenü verwaltet und erfolgt über ein separates Menü; hier können die einzulesenden Datensätze ausgewählt und angepasste Import-Optionen angewendet

werden:

- **Anzahl der Dimensionen:** Die Anwendung kann 2D- und 3D-Datensätze verarbeiten und darstellen (vgl. Abschnitt Rendering). Hier kann festgelegt werden, welcher der beiden Modi aktiv sein soll.
- **Transponieren der Datensätze:** Die Datenvektoren der Zeitreihe können vor Anwendung der PCA transponiert werden. Dies ist gerade bei hochauflösenden Datensätzen bezüglich des Speicher- und Rechenaufwandes günstiger, da die zu berechnenden Matrizen kleiner als im nicht transponierten Fall sind.
- **Normalisieren der Datensätze:** Die Werte der Zeitreihe können vor der Anwendung der PCA-Analyse auf das Intervall $[-1, 1]$ normalisiert werden. Diese Funktion schafft eine Basis zur Vergleichbarkeit der Analyse-Ergebnisse verschiedener Zeitreihen, z.B. bezüglich der Fehlerbetrachtung.
- **Vorfilterung:** Die Daten können vor Anwendung der PCA einer Schwellwertfilterung unterzogen werden. Hierbei kann der Nutzer einen Abstandswert definieren: Ist der Betrag des skalaren Wertes an einem Gitterpunkt des Datensatzes kleiner als dieser Abstandswert, so wird der Gitterpunkt der Fluidoberfläche zugeordnet (vgl. Kap. 4.3).

Analyse-Informationen

Über den Menüpunkt „Analyse-Informationen“ können verschiedenste Informationen zur analysierten Zeitreihe abgefragt werden, u.a.:

- **Rekonstruktionsfehler:** Bei der Fehlerbetrachtung werden minimale, maximale und mittlere Fehler ermittelt, als Fehlermaß dient die mittlere quadratische Abweichung (vgl. Kap. 5).
- **Varianzanteile:** Es werden die durch die ersten n Eigenwerte erklärten absoluten sowie kumulativen Anteile der Gesamtvarianz einer Zeitreihe aufgelistet.
- **Zeitentwicklung der Merkmale:** Über ein Unterfenster kann die Amplitude eines jeden Merkmals über der Zeit visualisiert werden. Die Skalierung der Amplitudenachse erfolgt dabei automatisch auf das optimale Werte-Intervall.

Rekonstruktion

In dieser Sektion wird die Anzahl der zur Analyse und Rekonstruktion verwendeten Hauptkomponenten festgelegt. Die Rekonstruktion kann einerseits mithilfe der ersten M Eigenvektoren $\vec{b}_i$ ($i = 0 \ldots M-1$) erfolgen, andererseits lässt sich durch den Nutzer ein Intervall $[i_s, i_e]$ angeben, welches die Eigenvektoren $\vec{b}_j$ ($j \in [i_s, i_e]$) definiert, die zur Rekonstruktion verwendet werden sollen. Diese Funktionalität leistet z.B. die Darstellung von Bereichen geringer Energie, d.h. geringem Varianzanteil des korrespondierenden Eigenvektors. Bezüglich der Fluidzeitreihen sind dies die Gebiete hoher räumlicher Frequenz (vgl. Abb. 3.11).

Als weitere Option lassen sich die ermittelten Hauptkomponenten selbst visualisieren. Erfolgte die Analyse ohne eine Transposition der Daten, so zeigen sie die Gebiete hoher und niedriger Dynamik.

Rendering

Über diesen Menüpunkt kann die Art der Visualisierung eingestellt werden: Zweidimensionalen Zeitreihen können entweder über Texturen, welche die *level sets* als Graustufenbild repräsentieren, bzw. über ein *Marching Squares*-Verfahren mit justierbarem Isowert visualisiert werden. Bei der Texturendarstellung erfolgt ein automatisches Mapping des Werteintervalls der Daten auf den normierten Grauwertbereich: $[\max(\vec{x}_t), \min(\vec{x}_t)] \rightarrow [0, 1]$. Als weitere Option ist vor dem Anwenden des Mappings ein Begrenzen – sogenanntes *clamping* – der Daten auf das Intervall $[-1, 1]$ möglich[4]. Dies ermöglicht in vielen Fällen ein ausgeglichenes Helligkeitsverhältnis zwischen den Frames einer Animation.

Für die Visualisierung dreidimensionaler Zeitreihen ist ein *Marching Cubes*-Algorithmus mit frei einstellbarem Isowert implementiert.

Neben der Animation der Zeitreihe kann ein Interpolationsmodus mit regelbarer Zeitschrittweite t aktiviert werden. Hierbei wird die Variable t gleichzeitig als Parameter zur linearen Interpolation zwischen zwei Datenvektoren $\vec{x}_i$ und $\vec{x}_{i+1}$ genutzt: Für $t \in [i, i+1)$ wird der interpolierte Datenvektor $\vec{x}_t = (1 - [i + 1 - t]) \cdot \vec{x}_i + [i + 1 - t] \cdot \vec{x}_{i+1}$ berechnet. Die Interpolation komprimierter Zeitreihen ist dabei im Raum der Merkmale möglich (vgl. Kap. 3.2.10).

Simulation

Für die Anwendung wurde die Simulation nach dem Oszillatormodell aus Kapitel 3.4.1 implementiert, welche aus diesem Untermenü heraus gesteuert werden kann. Der Nutzer kann eine beliebige Anzahl

[4] Anm.: Das Intervall $[-1, 1]$ für die Isowerte hat sich in der Praxis als zufriedenstellend gezeigt; das Augenmerk liegt ohnehin auf der Kontur mit dem Wert Null.

von Oszillator-Modulen generieren, welche jeweils mit einer Hauptkomponente verknüpft werden. Für jedes Modul können folgende Parameter eingestellt werden (vgl. Kap. 3.4.1):

- Grundfrequenz
- Anzahl der Harmonischen
- Spektrale Dämpfung
- Phase
- Amplitude
- Amplitudendämpfung
- Bias

4.1.2 Visualisierung der Zeitreihen

Neben dem Hauptmenü existiert ein OpenGL-Fenster zur Visualisierung der Zeitreihen. Hier werden Ursprungs- und rekonstruierte Zeitreihen in einer Doppelansicht dargestellt, um dem Nutzer ein unmittelbares visuelles Feedback über die Qualität der Rekonstruktion zu geben. Die Navigation durch die einzelnen Frames des Datensatzes kann, insofern nicht der Animationsmodus aktiv ist, über Mausrad und Cursortasten erfolgen.

Für die Darstellung von dreidimensionalen Daten wird eine Trackball-Kamerasteuerung genutzt, sodass ein freies Drehen und Zoomen der Daten-Objekte möglich ist.

Eine Anzeige-Überlagerung gibt Aufschluss über diverse Informationen wie den aktuellen Datensatz, das Kompressionsverhältnis oder die Anzahl der verwendeten Hauptkomponenten.

4.2 Beschreibung des Datensatzformates

Zum Testen der Analysesoftware wurden mit dem Fluid-Simulator des Lehrstuhls verschiedene zwei- und dreidimensionale Testdatensätze generiert. Die Datensätze definieren die Fluidoberfläche mithilfe einer Distanzfunktion, d.h. die implizite Funktion $\Phi(\underline{x}) = \Delta$, deren Funktionswert den vorzeichenbehafteten Abstand zur Oberfläche beschreibt (vgl. Kap. 3.1). Dabei ist das relevante *level set* Γ_0 ebenjene Region, bei der Δ gleich Null ist. Abbildung 4.2 verdeutlicht beispielhaft die Struktur der Datensätze: Die Distanzen liegen in einem rectilinearen Gitter mit konstanter Gitterweite vor. Dabei wird für jede Zelle der minimale Abstand des Zellmittelpunktes zur Oberfläche gespeichert, wobei Werte außerhalb

des Fluids ein positives und Werte innerhalb des Fluids ein negatives Vorzeichen besitzen.

Die Distanzwerte der Datensätze werden in einem Binärformat gespeichert. Am Anfang einer solchen Binärdatei werden die Gitter-Dimensionen sowie die Zellgröße definiert, danach folgen sequentiell die Werte des Gitters. In einer Datei wird genau ein Datenvektor abgelegt.

2,5	2,5	2,5	2,9	2,9	2,5	2,5	2,5
1,5	1,5	1,6	2,1	2,1	1,6	1,5	1,5
0,5	0,5	0,7	1,4	1,4	0,7	0,5	0,5
-0,5	-0,5	0,0	0,5	0,5	0,0	-0,5	-0,5
-1,5	-1,4	-0,7	-0,5	-0,5	-0,7	-1,4	-1,5
-2,5	-2,1	-1,6	-1,5	-1,5	-1,6	-2,1	-2,5
-3,5	-2,9	-2,5	-2,5	-2,5	-2,5	-2,9	-3,5
-4,3	-3,8	-3,5	-3,5	-3,5	-3,5	-3,8	-4,3

Abbildung 4.2: Schematische Repräsentation eines *level set*-Datenvektors. Eingezeichnet ist der errechnete Fluidrand.

4.3 Vorfilterung

Die implementierte Analysesoftware bietet eine Option zur Filterung der Datensätze während des Imports an. Bei dieser *Treshold*-Filterung werden die skalaren Distanzwerte x_i eines Datenvektors über einen vom Nutzer einstellbaren Schwellwert ξ modifiziert. Es gilt:

$$\text{filt}(x_i, \xi) \equiv \begin{cases} 1 & \text{, wenn } \vec{x}_i > \xi \\ -1 & \text{, wenn } -\vec{x}_i > \xi \\ 0 & \text{sonst} \end{cases} \tag{4.1}$$

Werte, welche betragsmäßig größer als der von außen eingestellte Wert ξ sind, werden demnach unter Beibehaltung des Vorzeichens auf 1 respektive -1 gesetzt. Werte, die betragsmäßig kleiner als ξ sind, werden auf den Wert null abgebildet. Abbildung 4.3 zeigt beispielhaft die Werte des Datensatzes aus Kapitel 4.2 und den Verlauf der Fluidoberfläche nach einer solchen Schwellwertfilterung.

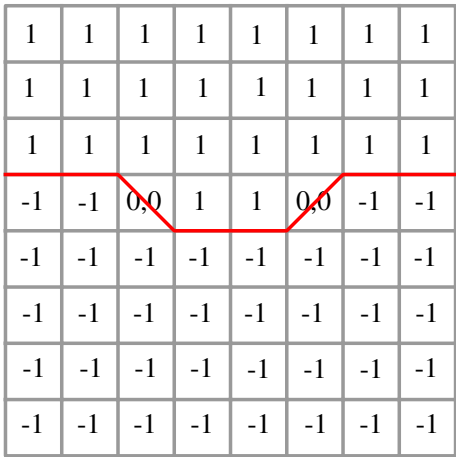

Abbildung 4.3: Schematische Repräsentation eines *level set*-Datenvektors nach der Schwellwert-Vorfilterung. Eingezeichnet ist der errechnete Fluidrand.

5 Ergebnisse

5.1 Ergebnisse der PCA

Die Anwendung des entwickelten Software-Prototypen zeigt die Fähigkeit der Dimensionsreduktion durch die Hauptkomponentenanalyse. Abbildung 5.1 verdeutlicht beispielhaft die Rekonstruktionsqualität eines durch die PCA komprimierten dreidimensionalen Datensatzes unter verschiedenmächtigen Hauptkomponentenbasen. Je größer die Dimension ebendieser ist, desto genauer können feine Details rekonstruiert werden.

Es bestätigt sich der in Kapitel 3.2.5 eingeführte Parameter ϵ^* als Gütekriterium für die Rekonstruktion, d.h. der errechenbare Residual-Fehler fällt umgekehrt proportional zur ansteigenden akkumulierten Varianz der den ersten Hauptkomponenten zugehörigen Eigenwerte. Bei Denis Serre (vgl. [Ser02]) findet sich ein Wert für ϵ^* von ca. 70 Prozent als ausreichend für eine visuell ansprechende Rekonstruktion. Nimmt man diesen Wert als Grundlage, kann für die untersuchten Testzeitreihen ein mittlerer Wert von ca. 3:1 als Kompressionsrate ermittelt werden. Bei einer günstigeren Varianzverteilung der Eigenwerte könnten entsprechend höhere Kompressionsraten erreicht werden.

Bezogen auf Fluidoberflächensimulationen kann in diesem Sinne auch formuliert werden: Zeitreihen mit niedriger (zeitlicher) Dynamik in hohen räumlichen Frequenzen lassen sich im Allgemeinen mit weniger Hauptkomponenten darstellen als Zeitreihen mit hoher Dynamik in hohen Frequenzen. Ebenjenen Gebieten hoher räumlicher Frequenz sind die Hauptkomponenten zugeordnet, deren korrespondierende Eigenwerte nur noch einen geringen Anteil an der akumulierten Varianz ausmachen.

Die Abbildungen 5.2 und 5.3 zeigen die Rekonstruktion von Zeitreihenausschnitten unterschiedlicher Dynamik in hohen Frequenzen, d.h. kleinen räumlichen Strukturen.

Wird als Kriterium zur Einschätzung der Rekonstruktionsqualität anstelle der 70 Prozent-Schwelle aus [Ser02] lediglich die visuell überzeugende und plausible Anmutung herangezogen, so lassen sich für Zeitreihen mit durchschnittlicher[1] Verteilung der Varianzen über die Eigenwerte mitunter Kompressionsraten von 6:1 und mehr ausmachen. Tabelle 5.1 verdeutlicht diesen Sachverhalt und zeigt die Kom-

[1] Anm.: Die Durchschnittlichkeit ist in diesem Falle die durchschnittliche Verteilung der Eigenwerte, welche aus der Untersuchung der Gesamtheit der Testzeitreihen mithilfe der Analysesoftware hervorgeht.

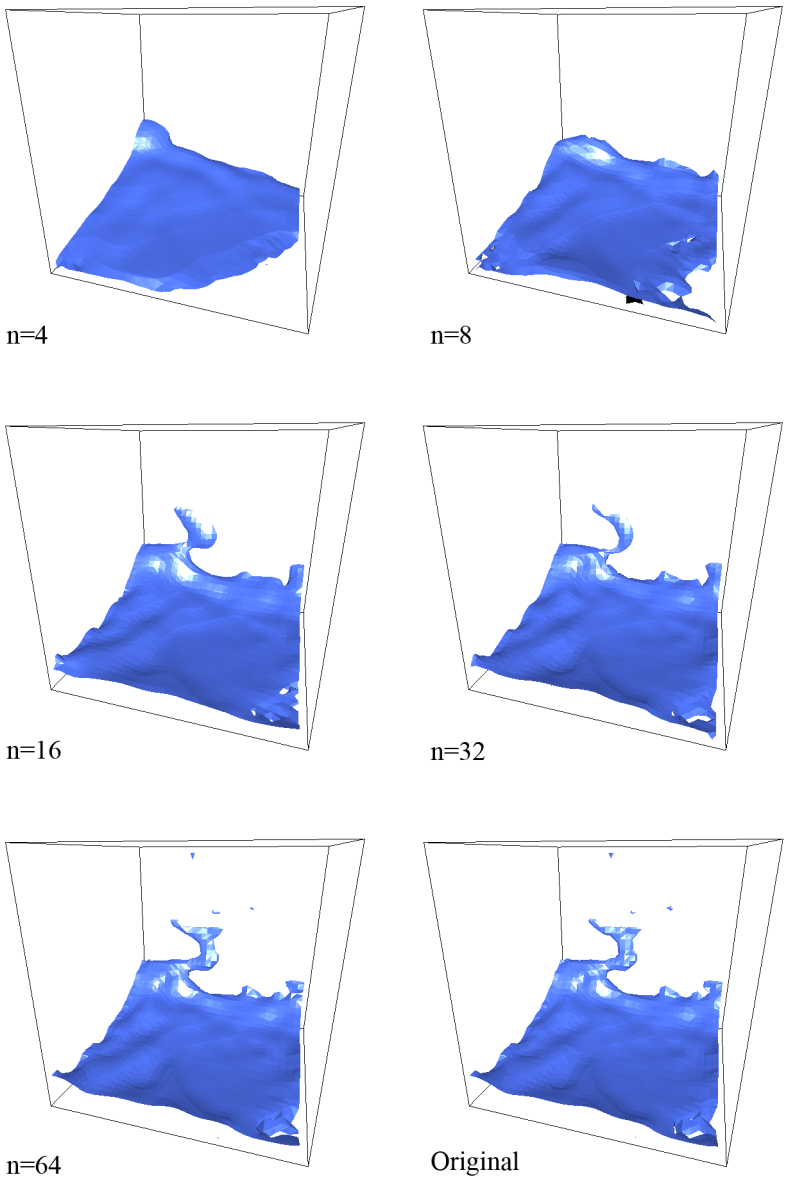

Abbildung 5.1: Rekonstruktionen einer 3D-Zeitreihe bestehend aus 136 Datensätzen in Abhängigkeit der Anzahl n der Hauptkomponenten.

pressionsraten der untersuchten 2D-Zeitreihen bezogen auf den von Serre vorgeschlagenen statischen Wert für ϵ^* sowie für einen vorgeschlagenen individuellen Wert, bei dem die visuelle Qualität als Referenz herangezogen wurde. Die 3D-Zeitreihe konnte aufgrund der Speicherproblematik großer Datenmengen (vgl. Kap. 6.2) nur transponiert analysiert werden, somit ist hier keine Vergleichbarkeit zu den zweidimensionalen Zeitreihen gegeben.

Zeitreihe	T	N ($\epsilon \geq 70\%$)	K_ϵ	N (visuell)	K_v
1	200	64	3,06	39	5,02
2 (sh. Abb. 5.2)	300	81	3,59	60	4,85
3	200	70	2,8	60	3,26
4	133	39	3,36	22	5,96
5 (sh. Abb. 5.3)	151	48	3,09	20	7,43
∅	196.8	60,4	3,18	40,2	5,30

Tabelle 5.1: Vergleich der Kompressionsraten auf Grundlage der 70-Prozent-Schwelle nach [Ser02] sowie einer visuellen Beurteilung. (Mit T..Anzahl der Zeitschritte, N..Anzahl der verwendeten Hauptkomponenten, K_ϵ..Kompressionsrate bei $\epsilon \geq 70\%$, K_v..Kompressionsrate bei visueller Beurteilung.)

Fehlerbetrachtung

Fehlermaße

Als Fehlermaß wird die mittlere quadratische Abweichung (kurz *MSE* für *mean square error*) herangezogen, da sie einerseits dem Minimierungsziel der Hauptkomponentenanalyse entspricht und andererseits keine statistischen Ausreißer in den vorliegenden Testzeitreihen zu erwarten sind. Dabei beschreibt der MSE den Mittelwert der quadrierten Einträge des Differenzvektors von Ursprungsdaten $\vec{x}_t \in \mathbb{R}^N$ und rekonstruierten Daten $\tilde{\vec{x}}_t \in \mathbb{R}^N$. Er lässt sich wie folgt berechnen:

$$\text{MSE}(t) \equiv \frac{1}{N} \cdot \sum_{n=0}^{N-1} (\vec{x}_{t,n} - \tilde{\vec{x}}_{t,n})^2 = \frac{< \vec{x}_t - \tilde{\vec{x}}_t, \vec{x}_t - \tilde{\vec{x}}_t >}{N} \tag{5.1}$$

Man sieht, dass der MSE hier den mittleren Fehler eines Datenvektors $\vec{x}_t$ der Zeitreihe $X = \{\vec{x}_t | t = 0..T-1\}$ definiert. Um die Fehleranalyse zu verfeinern und weitere Aussagen über die Qualität der Rekonstruktion zu treffen, werden folgende weitere Fehlermaße definiert:

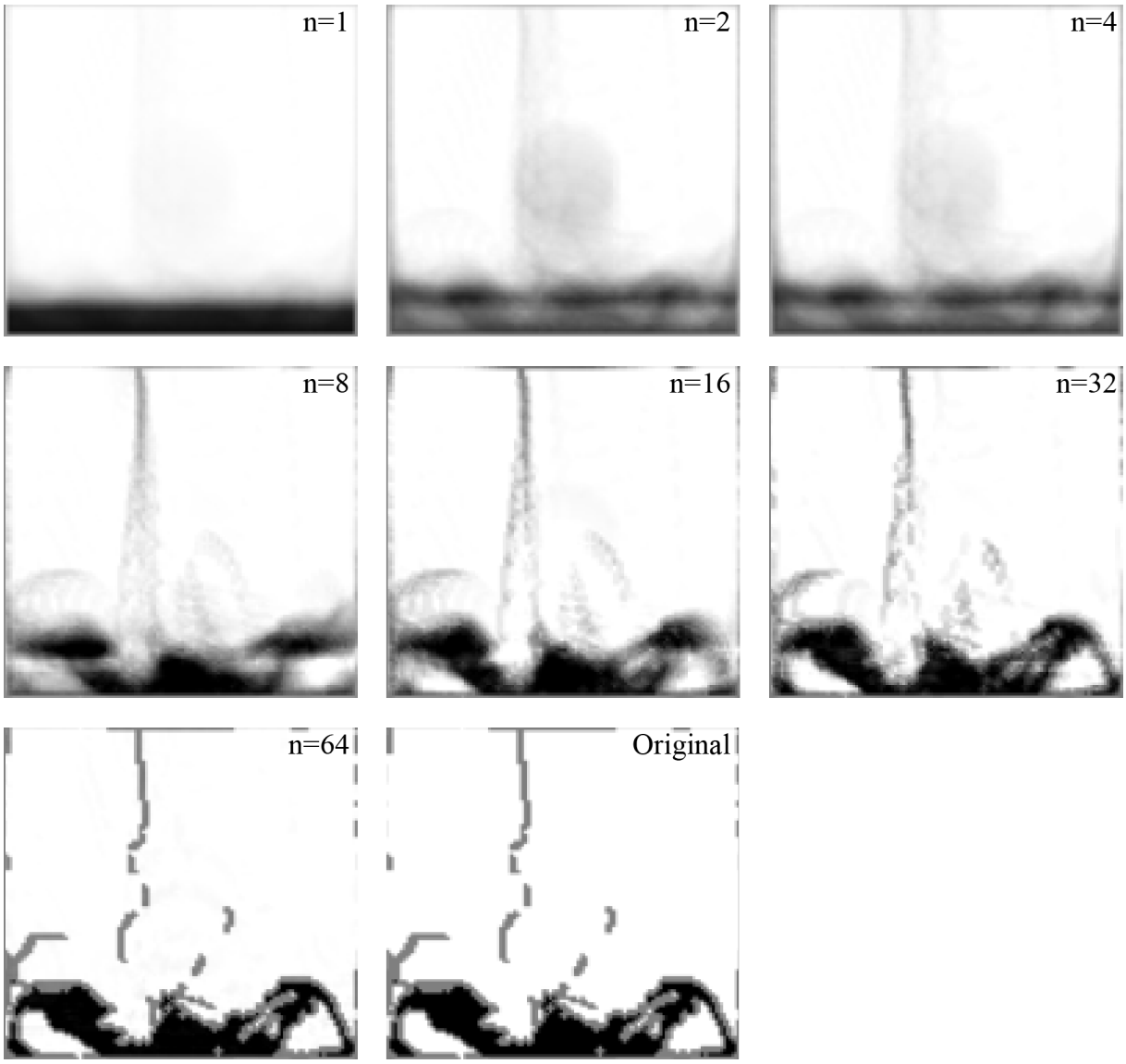

Abbildung 5.2: Rekonstruktionen einer 2D-Zeitreihe mit großer Dynamik in hohen Frequenzen.

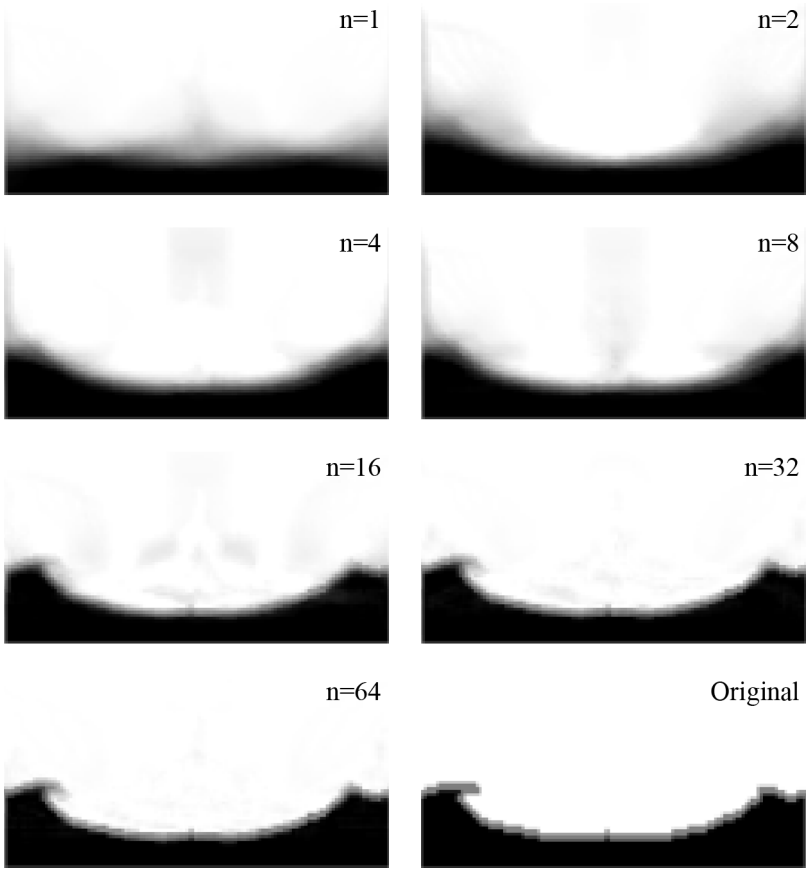

Abbildung 5.3: Rekonstruktionen einer 2D-Zeitreihe mit niedriger Dynamik in hohen Frequenzen.

Minimaler MSE

Dieser Wert stellt den kleinsten aller MSE der einzelnen Datenvektoren einer Zeitreihe dar:

$$\text{MSE}_{min} \equiv \min(\{\text{MSE}(0), \text{MSE}(1), \ldots, \text{MSE}(T-1)\})$$

Mittlerer MSE

Dieser Wert geht aus der Mittelung aller MSE der Datenvektoren der Zeitreihe hervor:

$$\text{MSE}_{mean} \equiv \frac{1}{T} \cdot \sum_{t=0}^{T-1} \text{MSE}(t)$$

Maximaler MSE

Dieser Wert ist gegensätzlich zum minimalen MSE als der größte auftretende MSE einer Zeitreihe definiert:

$$\text{MSE}_{max} \equiv \max(\{\text{MSE}(0), \text{MSE}(1), \ldots, \text{MSE}(T-1)\})$$

Auswertung

Abbildung 5.4 zeigt die Auswertung der rekonstruierten Testzeitreihen bezüglich ihrer MSE. Zur Vergleichbarkeit wurden die Zeitreihen zunächst normiert (vgl. Kap. 4.1.1), anschließend wurden minimaler, mittlerer und maximaler MSE ermittelt. Man erkennt, dass maximaler und mittlerer MSE dabei nah beieinander liegen.

Weiterhin zeigt die Verwendung der in Kapitel 4.3 vorgestellten Schwellwertfilterung deutlich kleinere Fehler bei einer Rekonstruktion mit wenigen Hauptkomponenten als bei der Rekonstruktion ohne Filterung. Abbildung 5.5 sowie Tabelle 5.2 verdeutlichen diesen Sachverhalt genauer.

Auch beeinflusst die Schwellwertfilterung die Verteilung der Varianzanteile der Eigenwerte (vgl. Abb. 5.6). Durch sie ergibt sich ein größerer Anstieg der kumulativen Varianz für die ersten Hauptkomponenten.

5.2 Ergebnisse der Kern-PCA

Die Kern-PCA zeigt für Beispielrechnungen (vgl. Kap. 3.3.4), dass eine für die Kompression günstigere Verteilung der Eigenwerte als bei der klassischen PCA erreicht werden kann. Daraus wird geschlossen,

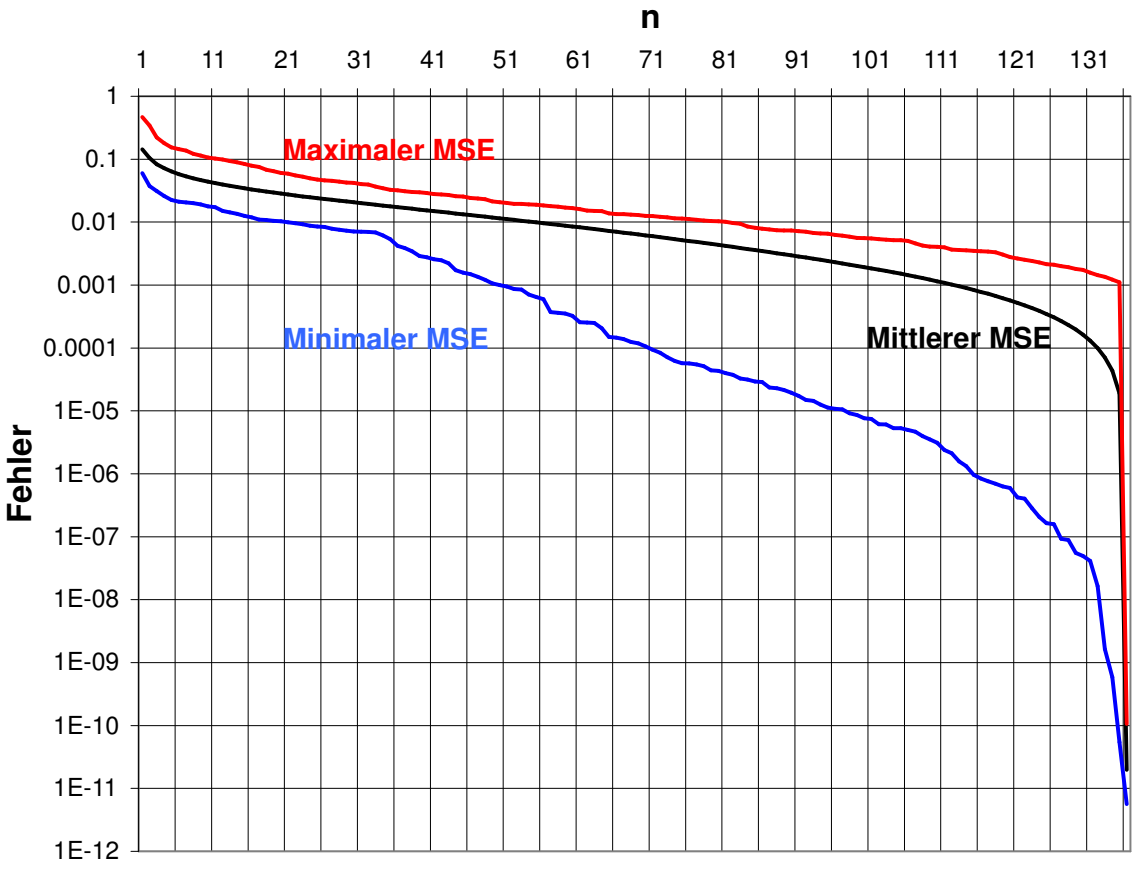

Abbildung 5.4: Maximaler, minimaler und gemittelter MSE der Testzeitreihen (gemittelt über die Gesamtheit der Zeitreihen) in Abhängigkeit der Anzahl n der verwendeten Hauptkomponenten.

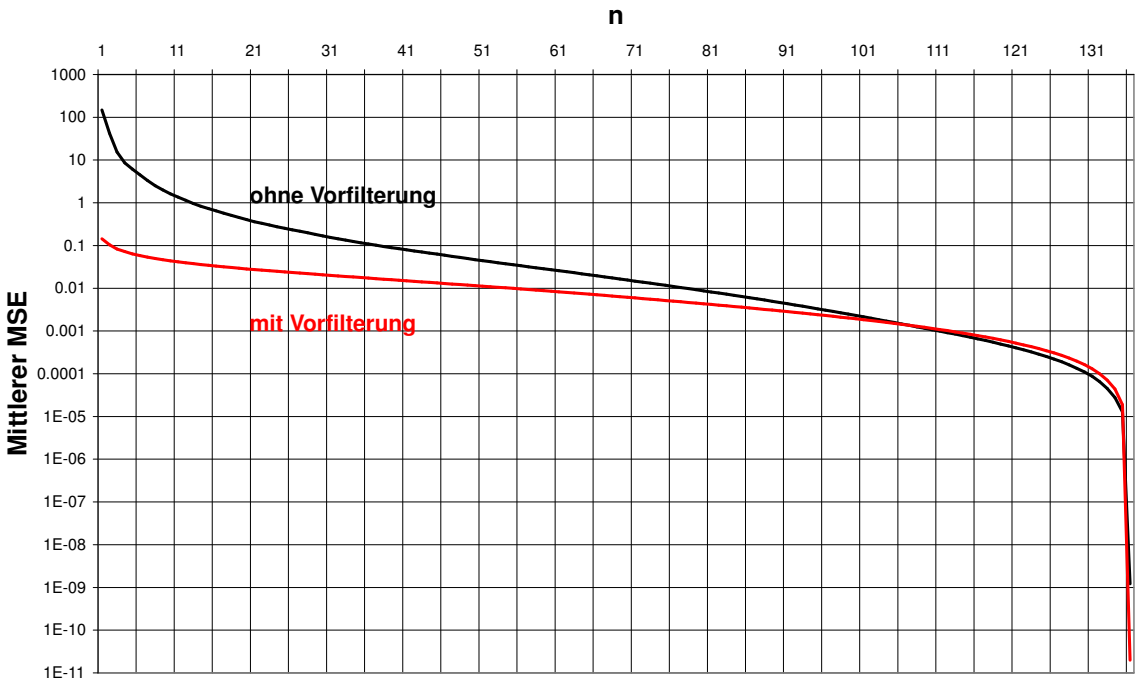

Abbildung 5.5: Mittlerer MSE der Testzeitreihen (gemittelt über die Gesamtheit aller Zeitreihen) in Abhängigkeit der Vorfilterung sowie der Anzahl n der verwendeten Hauptkomponenten.

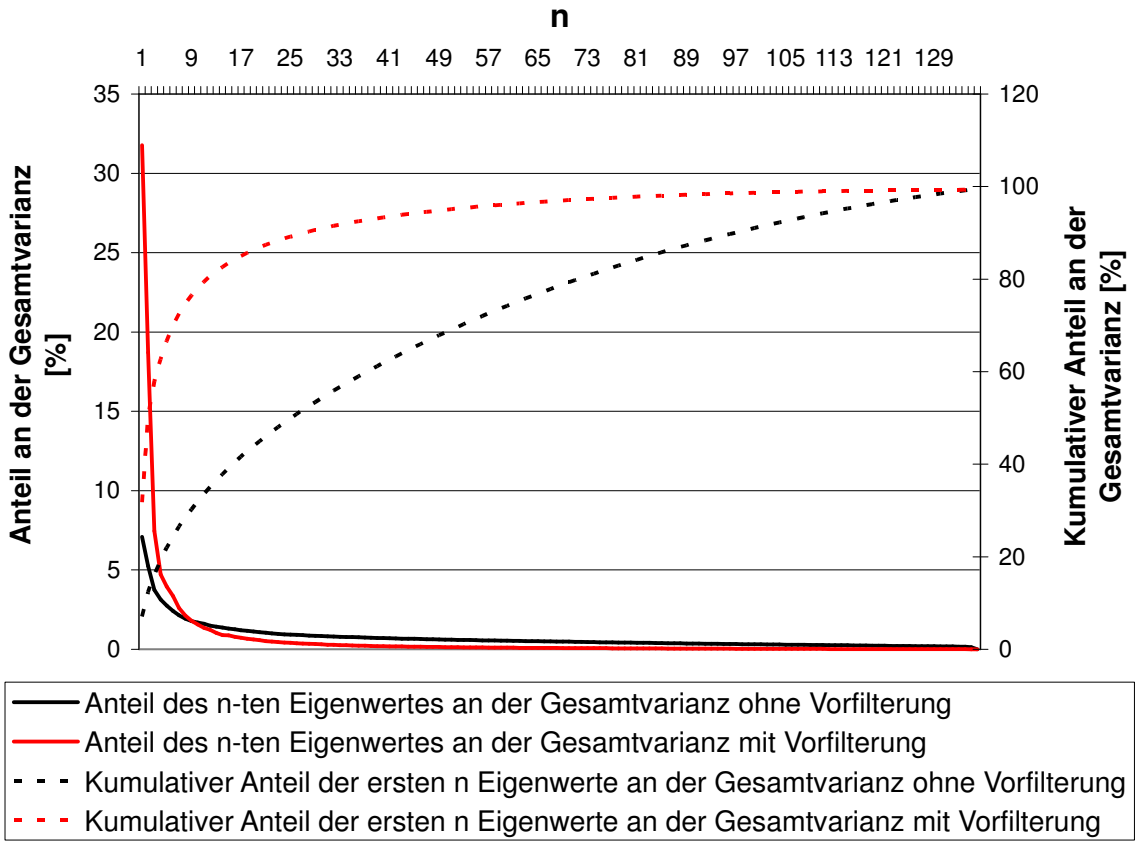

Abbildung 5.6: Anteile der Eigenwerte an der Gesamtvarianz in Abhängigkeit der Vorfilterung.

Anzahl der Haupkomponenten	MSE ohne Vorfilterung	MSE mit Vorfilterung
1	147.8754	0.143877
2	42.30488	0.10443192
4	8.547068	0.07195824
8	2.516271	0.04950984
16	0.64514574	0.0330747
32	0.14343882	0.019445538
64	0.021833214	0.007504626
128	0.000161377	0.000232265

Tabelle 5.2: Vergleich des mittleren MSE der Testzeitreihen (gemittelt über die Gesamtheit der Zeitreihen) in Abhängigkeit der Anzahl der verwendeten Hauptkomponenten mit und ohne Vorfilterung.

dass sie, unter der Annahme der Wahl des optimalen Kerns, zumindest im Bildraum $\mathcal{H}$ bessere Ergebnisse als die PCA im Ursprungsraum liefern kann und somit die Beschreibung der Daten mit noch weniger Hauptkomponenten auskommt.

5.3 Ergebnisse der interaktiven Simulation

Die Anwendung des vorgestellten heuristischen Simulations-Modells (vgl. Kap. 3.4.1) auf die Testdatenreihen des Lehrstuhlfluidsimulators zeigt, dass plausible und visuell überzeugende Animationen von Fluidoberflächen auf der Grundlage ausgewählter Zeitreihen extrapoliert werden können. Hierzu sind verschiedenste Parameter durch den Anwender zur Laufzeit manipulierbar (vgl. Kap. 4). Da die Simulation jedoch auf der Analyse von Trainingsdatensätzen beruht und die Zeitentwicklungsfunktionen der Merkmalsvektoren durch das Modell approximiert werden, sind nur solche Schwingungsvorgänge simulierbar, welche auf den Ausprägungen der gefundenen Hauptkomponenten beruhen. Dabei ähneln die Zeitentwicklungen der Merkmale von gleichmäßigen Schwingungsvorgängen eher überlagerten Sinusschwingungen, als die von chaotischen Vorgängen, und sind somit besser approximierbar. Die Praxis zeigt, dass für solche Schwingungen meist wenige Paare von Harmonischen und Hauptkomponenten genügen, um visuell ansprechende Animationen zu erzeugen.

6 Diskussion

6.1 Einschätzung der Ergebnisse

6.1.1 PCA

Die Ergebnisse der PCA zeigen, dass trotz der Dimensionsreduktion wichtige Merkmale der Fluidbewegung wie Turbulenzen erhalten bleiben, in Abhängigkeit der Dimension der reduzierten Basis.

Sie stellt ein stabiles Verfahren dar, was sich im nahen Beieinanderliegen von mittlerem und maximalem MSE widerspiegelt. Es sollten folglich keine einzelnen Datenvektoren einer Zeitreihe signifikant größere Fehler aufweisen, als durch den mittleren vorgegeben. Somit bleibt die Rekonstruktionsqualität in ihrer Gesamtheit auf einem gleichmäßigen Niveau. Für einzelne Datenvektoren kann jedoch eine wesentlich höhere Rekonstruktionsqualität erreicht werden, da ein relativ großer Abstand zwischen mittlerem und minimalem MSE vorliegt, wenn eine durchschnittliche Anzahl von Hauptkomponenten zur Rekonstruktion verwendet wurde (vgl. Abb. 5.4).

6.1.2 Kern-PCA

Aufgrund des Urbildproblems und der damit verbundenen Schwierigkeiten bei der Rekonstruktion (vgl. Kap. 3.3.6) erfolgte keine Implementierung der Kern-PCA. Folglich kann noch keine Aussage darüber getroffen werden, ob die Qualität der Rekonstruktion durch die Kern-PCA nach der Rückabbildung in den Ursprungsraum besser als die Rekonstruktionsqualität der PCA ist.

6.1.3 Simulation

Das implementierte Modell leistet eine einfache Beschreibung der Schwingungsvorgänge und kann Pseudosimulationen aus trainierten Fluidbewegungen generieren. Es können aus beschränkten Animationszeitrahmen unbeschränkte Simulationszeitrahmen erzeugt werden. Dabei erfolgt die Berechnung der Zeitentwicklungskoeffizienten vollständig im reduzierten Raum und ist in ihrer Komplexität unabhängig von der Größe des simulierten Raumes. Im Gegensatz zu Treuille (vgl. [TLP06]) ist keine direkte

Interaktion mit der Fluidoberfäche durch den Nutzer möglich, jedoch lassen sich die Modellvariablen zur Laufzeit anpassen und verändern. Das Modell vermag, bedingt durch das zugrundeliegende Konzept der superpositionierten Partialschwingungen, lediglich periodische Ausschwingvorgänge zu simulieren und eignet sich nicht für die physikalisch exakte Simulation. Wie auch in [TLP06] ist der Simulationsinhalt stets abhängig vom zugrundeliegenden Trainingsdatensatz, da er einzig auf den gewonnenen Hauptkomponenten beruht.

6.2 Bestehende Probleme und Lösungsansätze

Der aktuelle Prototyp lädt die Datensätze zur Analyse komplett in den Hauptspeicher, ebenso werden die Matrizen, welche zur SVD-Zerlegung benötigt werden, im Hauptspeicher verwaltet. Wie in Kapitel 3.2.3 ersichtlich, wächst die Kovarianzmatrix quadratisch zur Größe der Datenvektoren. Somit verursachen die Speicherkosten, gerade bei sehr großen Datensätzen, einen Flaschenhalseffekt und es kann zu Speicheradressierungsproblemen kommen, insbesondere bei dreidimensionalen Datensätzen. Für einen 3D-Datensatz über einem 128^3-Gitter müssten für die Kovarianzmatrix nach naivem Ansatz 17.59 TB an Speicher reserviert werden, wenn die skalaren Distanzwerte als floats mit 32 Bit angenommen seien. Die Implementierung des Prototypen nutzt deshalb die SVD-Methode auf die Matrix aller Datenvektoren an, um die Eigenwerte und Eigenvektoren der Kovarianzmatrix zu ermitteln, ohne diese jedoch explizit aufzustellen (vgl. Kap. 3.2.4). So kann einer Speicherknappheit in Maßen zuvorgekommen werden. Eine weitere mögliche Lösung besteht darin, die PCA auf die transponierte Matrix der mittelwertbefreiten Datenvektoren anzuwenden (vgl. Kap. 4.1.1). Bei der Rekonstruktion müssen die Daten entsprechend wieder zurücktransponiert werden. Dies bringt eine Speicherersparnis, insofern die zeitliche Auflösung kleiner als die räumliche ist[1] und wurde auf die 3D-Testzeitreihe angewendet.

Ebenfalls hohe Speicherkosten werden durch die Berechnung der Kernmatrix erzeugt, hier jedoch mit quadratischer Abhängigkeit zur Anzahl der Zeitschritte. Dies kann zu Problemen bei Simulationen führen, welche einen sehr langen Simulationszeitraum aufweisen.

Ein grundsätzlicher Nachteil der PCA-Methode besteht in ihrem Optimierungsziel; der Strukturierung nach größter Varianz. Nicht immer muss die Richtung der größten Bedeutsamkeit in einem Datensatz auf die Richtung der größten Varianz fallen; dann schlägt die PCA fehl. Auch müssen, abhängig von den Daten, die Achsen der gefunden Basis nicht zwangsläufig orthogonal zueinander stehen, obgleich

[1] Anm.: Dann allerdings wird die Semantik der Hauptkomponenten zweifelhaft: Sie bildet nun keine entlang der Zeit konstante Basis mit wechselnden Koeffizienten (Merkmalen) mehr, sondern eine entlang der Zeit variable Basis mit lokal konstanten Koeffizienten. Man könnte folglich die wechselnden Basen als Merkmale und die Menge der konstanten Koeffizienten als Hauptkomponenten auffassen.

es die Anwendbarkeit der PCA durch diese Konstruktionsweise einfach macht, da hier das Hin- und Rückabbilden in bzw. aus der reduzierten Basis allein das Transponieren der Abbildungsmatrix bedeutet. Hier können nichtlineare Verfahren wie z.B. die *Independent Component Analysis*, kurz ICA, unter Umständen bessere Ergebnisse liefern (vgl. [HO00]).

Im konkreten Anwendungsfall, der Nutzung der PCA zur Kompression von *level sets*, fällt die Problematik des Optimierungszieles derart ins Gewicht, dass eine Rekonstruktion ohne die in Kap. 4.3 beschriebene Vorfilterung beinahe unpraktikabel wird. Das Problem hierbei ist, dass mithilfe der PCA versucht wird, das gesamte Distanzfeld hinsichtlich des MSE zu optimieren, einschließlich der Bereiche abseits der Fluidgrenze, welche gegenüber der *level-set*-Kontur einen viel größeren Anteil an der Gesamtvarianz ausmachen. Somit werden unnötig viele Hauptkomponenten benötigt, um die Isolinie der Fluidoberfläche exakt zu rekonstruieren. Durch die Anwendung der Schwellwertfilterung werden die Bereiche außerhalb dieser Konturlinie auf einen konstanten Wert gesetzt und können dann mithilfe einer einzigen Hauptkomponente repräsentiert werden. Abbildung 6.1 zeigt den Vergleich der Rekonstruktion einer Zeitreihe mit und ohne vorangegangener Schwellwertfilterung.

Ein entscheidender Nachteil der Kern-PCA bei der Rekonstruktion ist der Umgang mit der Mittelwertbefreiung. Bisher wurde in den theoretischen Überlegungen stets von einer Zentriertheit der Daten ausgegangen bzw. konnte diese bei der Berechnung der Kernmatrix implizit erreicht werden (vgl. Kap. 3.3.5.1). In der Praxis jedoch muss die Mittelwertbefreiung während des Rekonstruktionsschrittes wieder rückgängig gemacht werden. Dazu müssen allerdings die Mittelwerte der einzelnen Datenkomponenten explizit bekannt sein. Dies ist durch das Verfahren aus Kapitel 3.3.6.1 nicht zu erreichen. Lösungen für diese Problematik findet man z.B. in approximierenden Verfahren, wie sie in Kapitel 3.3.6.2 beschrieben sind.

Auch ist die Wahl des „richtigen“ Kernes von entscheidender Bedeutung, d.h. es ist a-priori-Wissen über die Art der Daten notwendig oder es muss einen geeigneten Schätzer geben.

6.3 Ausblick

Die Untersuchungen an den Testdaten haben gezeigt, dass die PCA ein stabiles und einfaches Verfahren darstellt, um die Fluidoberflächenbewegungen mithilfe der Hauptkomponenten aus den Zeitreihen zu extrahieren. Dabei konnte das Verfahren durch den Einsatz einer einfachen Schwellwertfilterung entscheidend verbessert werden, wenngleich sie durch den festen Threshold die Fluidgrenzen stark auf das zugrundeliegende Abtastgitter quantisiert. Von diesem Punkt ausgehend sollten weitere Filter entwickelt und getestet werden, welche in der Nähe der Konturlinie eine feinere, womöglich adaptive Zuordnung

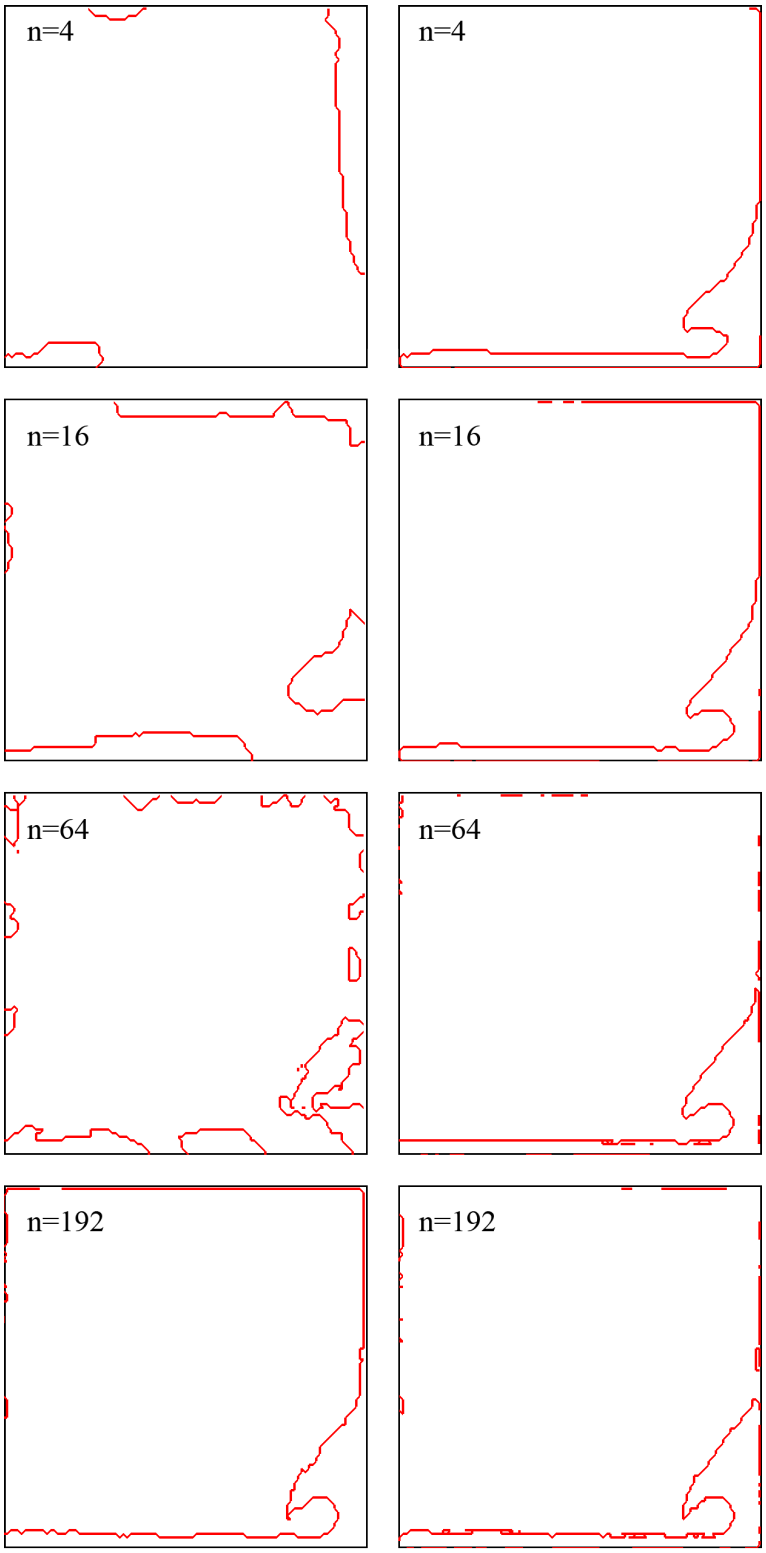

Abbildung 6.1: Rekonstruktionen einer 2D-Zeitreihe bestehend aus 200 Datenvektoren in Abhängigkeit der Anzahl n der Hauptkomponenten ohne (links) und mit Vorfilterung (rechts).

ermöglichen, um so die Fluidoberfläche glatter zu extrahieren. Auch sind Weiterentwicklungen der Visualisierungsalgorithmen denkbar, z.B. zur Aufbereitung von Mikrostrukturen auf der Fluidoberfläche. Zum aktuellen Zeitpunkt liefert ein rudimentäres Marching-Cubes-Verfahren die Visualisierung der Isoflächen dreidimensionaler Datensätze; für die Visualisierung von zweidimensionalen Zeitreihen ist ein Marching-Squares-Algorithmus sowie eine texturbasierte Darstellung implementiert.

Des Weiteren steht noch die Implementierung des Kern-PCA-Verfahrens aus. Trotz der bestehenden Problematik der Rückabbildung aus dem Hochdimsenionalen während der Rekonstruktion, ist die reine Darstellung der Hauptkomponenten und Merkmalsvektoren nicht minder uninteressant, wenngleich auch hier die Herausforderung besteht, in welcher Form die Hauptkomponenten mit potentiell unendlicher Dimension dargestellt werden sollten.

Einen weiteren Schritt, ausgehend vom aktuellen Stand der Arbeit, stellt die Weiterentwicklung des Simulationsmoduls dar. Zur Zeit lässt sich die Simulation lediglich durch die Justierung der technischen Parameter zur Laufzeit manipulieren. Eine direkte Nutzerinteraktion sei anzustreben, man vergleiche die interaktive Einflussnahme durch den Anwender in [TLP06].

7 Zusammenfassung

In der vorliegenden Arbeit wurde die wohlbekannte mathematische Methode der Hauptkomponentenanalyse in den Kontext der Computergraphik gebracht, um mit ihrer Hilfe Fluidgrenzflächensimulationen zu analysieren und zu komprimieren. Dabei wurden die theoretischen Grundlagen sowie der Algorithmus selbst detailliert vorgestellt und praktisch in einer prototypischen Anwendung umgesetzt. Durch die Analyse der Testzeitreihen und der Extraktion der Merkmalsvektoren konnten Eigenschaften gefunden werden, welche es erlaubten, ein Simulationsmodell basierend auf der Überlagerung harmonischer Partialschwingungen zu definieren. Es lassen sich damit eigenständige Simulationen generieren, welche die Merkmale der analysierten Testzeitreihen aufweisen, und deren Parameter sich zur Laufzeit durch den Nutzer einstellen lassen. Weiterhin wurde die Kernmethode in Verbindung mit der PCA als nichtlineares Verfahren untersucht. Dabei wurde der unter dem Namen Kern-PCA bekannte Algorithmus in der Theorie vorgestellt und bestehende Probleme sowie Lösungen aufgezeigt. Auf eine prototypische Umsetzung wurde aufgrund der Schwierigkeiten bei der Rekonstruktion verzichtet.

Literaturverzeichnis

[And63] ANDERSON, T.W.: Asymptotic theory for principal component analysis. In: *The Annals of Mathematical Statistics* 34 (1963), S. 122–148

[Bak05] BAKER, Kirk: *Singular Value Decomposition Tutorial*. March 2005. – `http://www.ling.ohio-state.edu/~kbaker/pubs/Singular_Value_Decomposition_Tutorial.pdf`

[CST03] CRISTIANINI, Nello ; SHAWE-TAYLOR, John ; BERTHOLD, Michael (Hrsg.) ; HAND, David J. (Hrsg.): *Intelligent Data Analysis*. Springer-Verlag Berlin/Heidelberg, 2003. – ISBN 3–540–43060–1

[DKKR07] DIMITROV, Darko ; KNAUER, Christian ; KRIEGEL, Klaus ; ROTE, Günter: Upper and lower bounds on the quality of the PCA bounding boxes. In: *In International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision - WSCG 2007*, 2007, S. 185–192

[GBT04] GLARDON, Pascal ; BOULIC, Ronan ; THALMANN, Daniel: PCA-Based Walking Engine Using Motion Capture Data. In: *Computer Graphics International Conference* (2004), S. 292–298. – ISSN 1530–1052

[Gow66] GOWER, J.C.: Some distance properties of latent root and vector methods used in multivariate analysis. In: *Biometrika* 53 (1966), S. 325–338

[HLB96] HOLMES, Philip ; LUMLEY, John L. ; BERKOOZ, Gal: *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge University Press, 1996

[HO00] HYVÄRINEN, Aapo ; OJA, Erkki: Independent component analysis: algorithms and applications. In: *Neural Networks* 13 (2000), S. 411–430

[Jef67] JEFFERS, J.N.R.: Two case studies in the application of principal component analysis. In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 16 (1967), S. 225–236

[Jol02] JOLLIFFE, I.T.: *Principal Components Analysis*. Second Edition. Springer-Verlag New York, 2002 (Springer Series in Statistics). ISSN 0172–7397

[Kil96] KILGARD, Mark J.: *The OpenGL Utility Toolkit (GLUT) - Programming Interface - API Version 3*. November 1996. – http://www.opengl.org/resources/libraries/glut/glut-3.spec.pdf

[KPK01] KIM, K. I. ; PARK, S. H. ; KIM, H. J.: Kernel Principal Component Analysis for Texture Classification. In: *IEEE Signal Processing Letters* 8 (2001), February, Nr. 2, S. 39–41. – ISSN 1070–9908

[KR01] KILGARD, Mark J. ; ROBINS, Nate: *GLUT - The OpenGL Utility Toolkit*. November 2001. – http://www.opengl.org/resources/libraries/glut/

[Kut04] KUTRUFF, Heinrich: *Akustik: Eine Einführung*. S. Hirzel Verlag Stuttgart, April 2004. – ISBN 3777612448

[Lum70] LUMLEY, John L.: *Stochastic Tools In Turbulence. Volume 12. Applied Mathematics And Mechanics*. Cambridge University Press, 1970

[Mar06] MARTIN, Shawn: An Approximate Version of Kernel PCA. 0 (2006), December, S. 239–244. ISBN 0–7695–2735–3

[MMK04] MÜLLER, Gero ; MESETH, Jan ; KLEIN, Reinhard: Fast Environmental Lighting for Local-PCA Encoded BTFs. In: *Computer Graphics International Conference* 0 (2004), S. 198–205. – ISSN 1530–1052

[OF02] OSHER, Stanley ; FEDKIW, Ronald: *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag New York, October 2002. – ISBN 0387954821

[Rad99] RADEMACHER, Paul: *GLUI - A GLUT-Based User Interface Library*. June 1999. – http://surfnet.dl.sourceforge.net/project/glui/Documentation/GLUI%202/glui_manual_v2_beta.pdf

[Rao64] RAO, C.R.: The use and interpretation of principal component analysis in applied research. In: *Sankhya Series A* 26 (1964), S. 329–358

[Row05] ROWLEY, C.W.: Model reduction for fluids, using balanced proper orthogonal decomposition. In: *Int. J. on Bifurcation and Chaos* 15 (2005), Nr. 3, S. 997–1013

[RSB06] RADEMACHER, Paul ; STEWART, Nigel ; BAXTER, Bill: *OpenGL User Interface Library*. July 2006. – http://glui.sourceforge.net/

[SA94] SREERAM, V. ; AGATHOKLIS, P.: On the properties of Gram matrix. In: *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on* 41 (1994), Mar, Nr. 3, S. 234–237. – ISSN 1057–7122

[Ser02] SERRE, Denis: *Graduate Texts in Mathematics*. Bd. 216: *Matrices - Theory and Applications*. Springer-Verlag New York, 2002. – ISBN 978–0–387–95460–8

[Shl03] SHLENS, Jon: *A Tutorial on Principal Components Analysis – Derivation, Discussion and Singular Value Decomposition*. March 2003. – `http://www.cs.princeton.edu/picasso/mats/PCA-Tutorial-Intuition_jp.pdf`

[Sir87] SIROVICH, Lawrence: Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations. III - Dynamics and scaling. In: *Quarterly of Applied Mathematics* 45 (1987), October, S. 561–571

[Smi02] SMITH, Lindsay I.: *A Tutorial on Principal Components Analysis*. February 2002. – `http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf`

[SMS99] SCHÖLKOPF, Bernhard ; MÜLLER, Klaus-Robert ; SMOLA, Alexander J.: Lernen mit Kernen – Support-Vektor-Methoden zur Analyse hochdimensionaler Daten. In: *Informatik – Forschung und Entwicklung* 14 (1999), Nr. 3, S. 154–163. – ISSN 0178–3564

[SS01] SCHÖLKOPF, Bernhard ; SMOLA, Alexander J.: *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, December 2001. – ISBN 0262194759

[TLP06] TREUILLE, Adrien ; LEWIS, Andrew ; POPOVIĆ, Zoran: Model Reduction for Real-time Fluids. In: *International Conference on Computer Graphics and Interactive Techniques* 25 (2006), Nr. 3, S. 826–834. – ISSN 0730–0301

[YZZL08] YAN, Su ; ZHAO, Jiu-Fen ; ZHAO, Jiu-Ling ; LI, Qing-Zhen: A Method for Image Classification based on Kernel PCA. In: *Proceedings of the Seventh International Conference on Machine Learning and Cybernetics* 2 (2008), July, S. 718–722. ISBN 978–1–4244–2095–7

Erklärungen zum Urheberrecht

GLUT unterliegt dem Copyright von Mark J. Kilgard. Für die Nutzung fallen keine Lizenzgebühren an.

GLUI unterliegt der GNU Lesser General Public License.