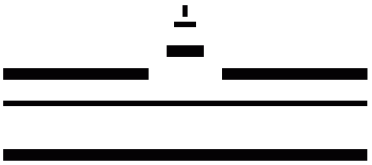
Westfälische Wilhelms-Universität Münster



Bachelorarbeit

Analyse raumzeitlicher Signale am Beispiel der Rayleigh-Bénard Konvektion

Analysis of spatiotemporal signals on the example of Rayleigh-Bénard convection

Tim Wilhelm Kroll
Matrikel-Nr.:370480
tim.kroll@wwu.de

Münster, Mai 2014

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen der Analyseverfahren **2**
2.1 Principal Component Analyse 2
2.1.1 Beschreibung der PCA 2
2.2 Singulärwertzerlegung 4
2.3 Independent Component Analyse 5
2.3.1 Grundlagen der ICA 5
2.3.2 Messung der Nicht-Gaussizität 6
2.3.3 FastICA-Algorithmus 7
2.4 Dynamic Mode Decomposition 8
2.4.1 Koopman Operator 8
2.4.2 Berechnung der Dynamic Mode Decomposition 9

3 Analyse zeitlicher Signale **10**
3.1 PCA 10
3.1.1 Kreisbewegung 10
3.1.2 PCA des Haken-Zwanzig Modell 13
3.2 ICA 13
3.3 DMD 16

4 Analyse raumzeitlicher Signale **18**
4.1 ICA 18
4.2 Lineare Kopplung 19
4.3 Nichtlineare Kopplung 21

5 Rayleigh-Bénard Konvektion **25**
5.1 Grundlagen der Rayleigh-Bénard Konvektion 25
5.2 Randbedingungen 26
5.3 Entdimensionalisierung 26
5.4 Analyse der Rayleigh-Bénard Konvektion 27
5.4.1 PCA 29
5.4.2 DMD 31

6 Zusammenfassung und Ausblick **36**

A PCA in Python **40**

B DMD in Python **40**

C Fast-ICA in Python **41**

D Erzeugung der Grundmuster und Umwandlung der Muster **42**

E Vektor-Matrix-Umwandlung für die Konvektionssimulation **43**

1 Einleitung

Bei komplexen Systemen handelt es sich um Systeme, welche aus einer Vielzahl von kleineren nichtlinear gekoppelten dynamischen Einheiten bestehen und daher einen hochdimensionalen Phasenraum besitzen. Viele komplexe Systeme neigen dazu, zeitliche, raumzeitliche oder auch funktionale Strukturen auszubilden, d.h trotz ihrer hohen Dimensionalität lassen Sie sich oft durch wenige relevante Freiheitsgrade bzw. Strukturen beschreiben. Ein Beispiel hierfür ist die Rayleigh-Bénard Konvektion, bei der sich für bestimmte Parametereinstellungen raumzeitliche Muster ausbilden. Eine Einführung in diese Thematik findet man in [Hak83], wo auch die Rayleigh-Bénard Konvektion behandelt wird.
Generell kann man den Zugang zu diesen Systemen in zwei verschiedene Ansätze aufteilen. Der erste Ansatz beruht auf dem bottom-up Prinzip. Hier kennt man bereits die Dynamik des Systems, in Form von (partiellen) Differentialgleichungen. Aus diesen Differentialgleichungen versucht man die sich ausbildenden Strukturen herzuleiten. Ein Problem ist hier, dass die Herleitung der Strukturen nicht immer möglich ist. Das zweite Problem ist, dass es Systeme gibt, deren Grundgleichungen nicht bekannt sind. In der Biologie und der Medizin steht man zum Beispiel vor diesem Problem.
Auf Grund dieser Probleme, gibt es einen zweiten Ansatz nach dem top-down Prinzip. Hier versucht man aus numerisch oder aber experimentell erzeugten Daten die wichtigen Strukturen zu identifizieren. Ein Beispiel dafür ist die Analyse von EEG-Daten. In [HUF99] werden zum Beispiel EEG-Daten mit einer Erweiterung der, in dieser Arbeit genutzten, Principal Component Analyse untersucht. Durch die Strukturen, welche man aus der Analyse erhält, wäre es dann zum Beispiel möglich die komplexen Systeme mit einer geringen Anzahl an Freiheitsgraden, also niedrigerdimensionaler, darzustellen.

Ziel dieser Arbeit ist es, verschiedene Analyseverfahren zur Dimensionsreduktion am Beispiel des Rayleigh-Bénard Systems auf ihre Eignung für die Untersuchung solcher komplexen Systeme zu testen. Zuerst werden hierfür die theoretischen Grundlagen der Principal Component Analyse(Abschnitt2.1), der Independent Component Analyse(Abschnitt2.3) und der Dynamic Mode Decomposition(Abschnitt2.4) dargestellt, sowie auch ihre numerische Umsetzung. Außerdem wird der Zusammenhang zwischen Principal Component Analyse und Singulärwertzerlegung herausgearbeitet(Abschnitt2.2).
In Kapitel 3 werden die Verfahren dann beispielhaft auf synthetisch erzeugte Zeitreihen angewendet, um so Vor- und Nachteile der Verfahren darzulegen. Nach den zeitlichen Signalen werden die Verfahren in Kapitel 4 ebenfalls auf raumzeitliche Signale angewendet. In diesem Kapitel wird festgestellt, dass die Independent Component Analyse für unsere Anwendungszwecke nicht geeignet ist.
Zuletzt werden in Kaptiel 5 die grundlegenden Gleichungen der Rayleigh-Bénard Konvektion erklärt, bevor die Verfahren auf eine zweidimensionale Simulation derselbigen Konvektion angewendet werden.

2 Grundlagen der Analyseverfahren

In diesem Kapitel werden die Grundlagen von drei verschiedenen Analyseverfahren erarbeitet. Namentlich sind dieses die Principal Component Analyse, im weiteren mit 'PCA' abgekürzt, die Independent Component Analyse,im weiteren auch 'ICA' genannt und die Dynamic Mode Decomposition, auch als 'DMD' abgekürzt. Diese Analyseverfahren werden in verschiedenen wissenschaftlichen Disziplinen angewandt. Es handelt sich bei ihnen um Verfahren, welche Muster in hochdimensionalen Zeitreihen sowie auch raumzeitlichen Signalen finden. Mit Hilfe diese Muster lassen sich die dynamischen Systeme, welche den jeweiligen Signalen zu Grunde liegen, in reduzierter Form beschreiben. Diese reduzierte Form ist mit einem niedrigerdimensionalen Unterraum des jeweiligen betrachteten Systems gleichzusetzen.

2.1 Principal Component Analyse

Die PCA wird zum Beispiel in der Turbulenzforschung [BHL93] und der Meteorologie [PV94] angewandt, aber auch in Biomechanik [Daf04]. Um nur wenige Beispiele zu nennen. Die Principal Component Analyse wurde zum ersten Mal von Karhunen [Kar46] und Loève [Loe65] beschrieben und ist daher auch als Karhunen-Loève Entwicklung bekannt. In der Literatur finden sich jedoch noch weitere Bezeichnungen, z.B. proper orthogonal decomposition, singular value decomposition oder empirical orthogonal function- Zerlegung für die PCA bzw. äquivalente Verfahren. In dieser Arbeit wird im weiteren der Terminus Principal Component Analyse bzw. PCA genutzt.

2.1.1 Beschreibung der PCA

Die folgenden Darstellungen orientieren sich an [Hut]. Es wird angenommen, dass ein System aus N verschiedenen reellwertigen Zeitreihen $q_i(t)$ betrachtet wird. Diese Zeitreihen sind für raumzeitliche Signale an bestimmten Punkten im Raum definiert, was eine 1-zu-1 Zuordnung der Zeitreihen an Raumpunkten zulässt.
Um einen guten mathematischen Zugang zu diesen Zeitreihen zu haben, werden sie als ein einzelner N-dimensionaler Vektor $\vec{q}(t)$ dargestellt:

$$\vec{q}(t) = \begin{pmatrix} q_1(t) \\ q_2(t) \\ . \\ . \\ . \\ q_N(t) \end{pmatrix} = \begin{pmatrix} q_1(t) \\ 0 \\ . \\ . \\ . \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ q_2(t) \\ . \\ . \\ . \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ . \\ . \\ . \\ q_N(t) \end{pmatrix} \tag{2.1}$$

Der Vektor $\vec{q}(t)$ besteht aus einer Summe von N einzelnen Vektoren, welche nicht zwangsweise linear unabhängig von einander sind. Zuerst wird nun ein N-dimensionales orthogonales Basissystem mit den Basisvektoren $\vec{v_i}$, im weiteren auch Moden genannt, definiert. Auf Grund der Orthogonalität der Moden gilt:

$$\vec{v_i} \cdot \vec{v_j} = \delta_{ij} \tag{2.2}$$

Der Zeitreihenvektor $\vec{q}(t)$ kann vollständig über dieses System beschrieben werden. Da die Basisvektoren zeitunabhängig sind, findet sich die zeitliche Abhängigkeit in den Projektionen $\eta_i(t) = \vec{q}(t) \cdot \vec{v_i}$ von $\vec{q}(t)$ auf die Moden $\vec{v_i}$ wieder.

Unter der Annahme, dass die Zeitreihen Redundanzen enthalten, sollte es möglich sein, $\vec{q}(t)$ mit $M < N$ Basisvektoren zu beschreiben.
Als ein Maß für die Genauigkeit der Beschreibung mit nur M Basisvektoren wird die Fehlerfunktion E, als zeitlich gemittelte quadratische Abweichung, definiert:

$$E = \frac{\langle(\vec{q}(t) - \sum_{i=1}^{M<N} \eta_i(t)\vec{v_i})^2\rangle}{\langle\vec{q}^2(t)\rangle}. \tag{2.3}$$

Hier steht $\langle ...\rangle$ für die Mittelung über den Zeitraum T, definiert als $\frac{1}{T}\int_0^T (...)dt$. Die hier definierte Fehlerfunktion kann wie folgt über eine Korrelationsfunktion dargestellt werden:

$$\begin{aligned}
E &= \frac{\langle(\vec{q}(t) - \sum_{i=1}^{M<N} \eta_i(t)\vec{v_i})^2\rangle}{\langle\vec{q}^2(t)\rangle} \\
&= \frac{\langle(\sum_{i=1}^{N} \eta_i(t)\vec{v_i} - \sum_{i=1}^{M<N} \eta_i(t)\vec{v_i})^2\rangle}{\langle\vec{q}^2(t)\rangle} \\
&= \frac{\langle(\sum_{i=M+1}^{N} \eta_i(t)\vec{v_i})^2\rangle}{\langle\vec{q}^2(t)\rangle} \\
&= \frac{\langle(\sum_{i=M+1}^{N} (\vec{q}(t)\cdot\vec{v_i})\vec{v_i})^2\rangle}{\langle\vec{q}^2(t)\rangle} \\
&= \frac{\langle(\sum_{i=M+1}^{N} (\vec{q}(t)\cdot\vec{v_i})\vec{v_i})(\sum_{j=M+1}^{N} (\vec{q}(t)\cdot\vec{v_j})\vec{v_j})\rangle}{\langle\vec{q}^2(t)\rangle} \\
&= \sum_{i,j=M+1}^{N} \frac{\langle(\vec{q}(t)\cdot\vec{v_i})\vec{v_i})(\vec{q}(t)\cdot\vec{v_j})\vec{v_j})\rangle}{\langle\vec{q}^2(t)\rangle} \\
&= \sum_{i,j=M+1}^{N} \frac{\langle(\vec{q}(t)\cdot\vec{v_i})(\vec{q}(t)\cdot\vec{v_j})\rangle\vec{v_i}\cdot\vec{v_j}}{\langle\vec{q}^2(t)\rangle} \\
&= \sum_{i=M+1}^{N} \vec{v_i}\boldsymbol{C}\vec{v_i}
\end{aligned} \tag{2.4}$$

Im letzten Schritt wurde die Kovarianzmatrix $\boldsymbol{C}$ eingeführt:

$$C_{ij} = \frac{\langle q_i(t)q_j(t)\rangle}{\langle\vec{q}^2(t)\rangle} \tag{2.5}$$

Außerdem wurde die Orthogonalität der Moden ausgenutzt und dass $\vec{q}(t)$ in einem beliebigen vollständigen orthogonalen Basissystem immer korrekt dargestellt werden kann.

Das Ziel ist es mit möglichst wenig Basisvektoren das Signal möglichst gut anzunähern, also die Fehlerfunktion zu minimieren. Hierzu muss die Fehlerfunktion E nach einer Mode $\vec{v}_k$ variiert werden. Durch die geforderte Nebenbedingung, dass die Moden normiert sind, erhalten wir folgende Lagrangefunktion $\mathcal{L}$ mit Lagrangemultiplikatoren λ_i:

$$\mathcal{L} = \sum_{i=M+1}^{N} \vec{v_i}\boldsymbol{C}\vec{v_i} - \sum_{i=M+1}^{N} \lambda_i(\vec{v_i}^2 - 1) \tag{2.6}$$

Da die Moden $\vec{v_i}$ zeitunabhängig sind, reduziert sich die Euler-Lagrange-Gleichung in diesem Fall wie folgt auf:

$$\frac{\partial \mathcal{L}}{\partial \vec{v}_k} = 2\boldsymbol{C}\vec{v_k} - 2\lambda_k \vec{v}_k \stackrel{!}{=} 0\,,\ \forall k = M+1, ..., N \tag{2.7}$$

Somit lässt sich das Problem auf das dargestellte Eigenwertproblem zurückführen:

$$\boldsymbol{C}\vec{v}_k = \lambda_k \vec{v}_k \tag{2.8}$$

Durch Lösung dieses Eigenwertproblems erhält man ein neues orthogonales Basissystem, in welchem die Eigenvektoren als Basisvektoren die stärksten Anteile des Signals repräsentieren. Der Anteil eines Eigenvektors am gesamten Signal wird durch den zugehörigen Eigenwert bestimmt. Verschwinden einer oder mehrere Eigenwerte, lässt sich das Signal in einem niedrigerdimensionalen Unterraum darstellen. Hierzu wählt man eine Rotationsmatrix $\boldsymbol{R}$ mit den Eigenvektoren als Spaltenvektoren, welche man auf das Signal $\vec{q}(t)$ anwendet:

$$\boldsymbol{R} = [\vec{v}_1, \vec{v}_2, ...\vec{v}_N] \tag{2.9}$$

Bei raumzeitlichen Signalen stellen die Eigenvektoren die räumlichen Moden des Systems dar. Mit anderen Worten sind dieses die Grundmuster, welche in der zeitlichen Entwicklung des räumlichen Signals auftreten.

Für große räumliche Dimensionen lässt sich die Berechnung der PCA auch umformulieren, um die numerischen Kosten zu reduzieren. Hierzu wird angenommen, dass das System an N_x Punkten im Raum und an N_t in der Zeit diskretisiert ist. Laut [SMH05] lässt sich die PCA nun als Berechnung einer räumlichen Korrelationsmatrix ausdrücken. Diese räumliche Korrelationsmatrix $\boldsymbol{C}$ ist dann definiert als:

$$C_{ij} = \frac{\langle q_i(x) q_j(x) \rangle}{\langle \vec{q}^2(x) \rangle} \tag{2.10}$$

Hierbei ist $i, j \in [1, N_t]$ und $q_i = [x_1(t_i), x_2(t_i), ..., x_{N_x}(t_i)]$. Bezeichnet man nun die Eigenwerte dieser Matrix $\boldsymbol{C}$ mit λ_i und die zugehörigen Eigenvektoren mit $\vec{c^i} = [c_1^i, c_2^i, ..., c_{N_t}^i]$, lassen sich die Eigenfunktionen $\phi_i(x)$, also die gesuchten räumlichen Moden, wie folgt berechnen:

$$\phi_i(x) = \frac{1}{\lambda_i N_t} \cdot [c_1^i, c_2^i, ..., c_{N_t}^i] \cdot \begin{bmatrix} q_1(x) \\ ... \\ q_{N_t}(x) \end{bmatrix} \tag{2.11}$$

Eine beispielhafte Implementation der PCA in Python befindet sich in Anhang A.

2.2 Singulärwertzerlegung

Die Singulärwertzerlegung ist ein Verfahren aus der linearen Algebra um eine beliebige reelle Matrix zu faktorisieren [BSMM05]. Auf Grund der englischen Bezeichnung 'Singular Value Decomposition' wird diese Zerlegung auch SVD abgekürzt.
Man betrachtet eine reelle Matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ mit Rang r. Die r singulären Werte

d_i dieser Matrix sind definiert als die Quadrat-Wurzeln der positiven Eigenwerte λ_i von $\boldsymbol{A}^T\boldsymbol{A}$ bzw. von $\boldsymbol{A}\boldsymbol{A}^T$. Es gilt also $d_i = \sqrt{\lambda_i}$. Die zu diesen Eigenwerten λ_i und der Matrix $\boldsymbol{A}^T\boldsymbol{A}$ gehörenden Eigenvektoren $\vec{v}_i$ heißen Rechtssingulärvektoren und die zu $\boldsymbol{A}\boldsymbol{A}^T$ gehörigen Eigenvektoren $\vec{u}_i$ werden Linkssingulärvektoren genannt. Die Links- und Rechtssingulärvektoren können jeweils in Matrizen $\boldsymbol{V}$ und $\boldsymbol{U}$ geschrieben werden, so dass gilt:

$$\boldsymbol{V} = (\vec{v}_1, \vec{v}_2, ..., \vec{v}_n) \tag{2.12}$$

$$\boldsymbol{U} = (\vec{u}_1, \vec{u}_2, ..., \vec{u}_m)\,. \tag{2.13}$$

Es gilt hier, dass die Eigenvektoren zu $d_i = 0$ Nullvektoren sind und in $\boldsymbol{V}$ die Indizes $n - r$ bis n und in $\boldsymbol{U}$ die Indizes $m - r$ bis m haben. Aus diesen beiden Matrizen $\boldsymbol{U}$ und $\boldsymbol{V}$, sowie der Diagonalmatrix $\boldsymbol{\Sigma} = \mathrm{diag}\,(d_1, d_2, ..., d_r)$ lässt sich die Matrix $\boldsymbol{A}$ wie folgt faktorisieren:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*. \tag{2.14}$$

Diese Darstellung der Matrix $\boldsymbol{A}$ ist ihre sogenannte Singulärwertzerlegung. Diese Singulärwertzerlegung ist äquivalent zur PCA, da man die Eigenvektoren und Eigenwerte der PCA durch sie berechnen kann. Für Datensätze, welche zentriert wurden, ist die Singulärwertzerlegung äquivalent zur PCA, denn $\boldsymbol{A}^T\boldsymbol{A}$ entspricht dann der Kovarianzmatrix $\boldsymbol{C}$, aus dem vorherigen Abschnitt. Somit entsprechen die d_i mit $i \leq r$ den Eigenwerten der PCA und die zugehörigen $\vec{v}_i$ ihren Eigenvektoren.

2.3 Independent Component Analyse

Wenngleich die PCA in ihrer Herleitung anders motiviert wurde, ist sie ein Verfahren, welches die statistische Unabhängigkeit zwischen mehreren Variablen maximiert. In der PCA wird als Maß für die statistische Abhängigkeit die Korrelation angesetzt. Somit erhalten wir durch die PCA Variablen, welche maximal unkorreliert sind. Diese Unkorreliertheit ist jedoch im Allgemeinen nicht mit statistischer Unabhängigkeit gleichzusetzen, da es bei unkorrelierten Daten immer noch Abhängigkeiten höherer Ordnungen geben kann.
Diese statistischen Abhängigkeiten höherer Ordnung zu minimieren ist die Motivation der ICA, welche von C. Jutten und P. Herault entwickelt wurde [JH91]. Die Darstellungen im Folgenden orientieren sich an [HO00], wo der FastICA-Algorithmus ebenfalls beschrieben wird.

2.3.1 Grundlagen der ICA

Das grundsätzliche Problem der ICA ist ähnlich gesetzt zu dem der PCA. Wir haben einen Signalvektor $\vec{x}(t)$ mit Komponenten $x_i(t)$ und nehmen an, dass diese Komponenten sich aus der Überlagerung von N Grundsignalen $s_i(t)$ zusammensetzen. Diese Überlagerung wird als linear angenommen, sodass gilt:

$$x_i(t) = \sum_j a_{ij} s_j(t). \tag{2.15}$$

Die Koeffizienten a_{ij} lassen sich als Komponenten einer Matrix $\boldsymbol{A}$ auffassen, womit sich das Problem auch wie folgt schreiben lässt:

$$\vec{x}(t) = \boldsymbol{A}\vec{s}(t). \tag{2.16}$$

Um nun die Grundsignale $\vec{s}(t)$ zu erhalten, muss man die Mischungsmatrix $\boldsymbol{A}$ kennen und invertieren, sodass gilt:

$$\vec{s}(t) = \boldsymbol{W}\vec{x}(t),\ \boldsymbol{W} = \boldsymbol{A}^{-1} \tag{2.17}$$

Die Problematik besteht nun darin, dass weder die Mischungsmatrix $\boldsymbol{A}$ noch die Grundsignale $s_i(t)$ bekannt sind.

Für die ICA werden nun verschiedene Annahmen gemacht, um dann iterativ die Matrix $\boldsymbol{W}$ zu bestimmen. Zum einen wird angenommen, dass die Wahrscheinlichkeitsverteilung der Grundsignale nicht gaussförmig sind. Wie wir später sehen werden, ist diese Annahme wichtig für den Algorithmus, da dieser ab zwei gaussförmig-verteilten Grundsignalen nicht mehr konvergiert. Zum anderen besteht die Annahme, dass die Mischungsmatrix $\boldsymbol{A}$ quadratisch ist.

Da die ICA keine Aussagen über die Varianz der Signale $s_i(t)$ machen kann, ist es möglich vor dem eigentlichen Algorithmus die Varianzen auf 1 zu normieren.

2.3.2 Messung der Nicht-Gaussizität

Da der Algorithmus auf der Annahme der Nichtgaussizität der Grundsignale beruht, wird ein Maß für die Gaußförmigkeit bzw. Nichtgaußförmigkeit der Signale gebraucht. Generell gibt es einige verschiedene zu einander äquivalente Arten die Gaußförmigkeit zu bestimmen. In diesem Abschnitt werden zwei Arten, namentlich die Kurtosis und die Negentropie, vorgestellt, um dann die Vorteile dieser beiden zu einem neuen Maß zu vereinen.

Die Kurtosis ist der Kumulant vierter Ordnung einer Wahrscheinlichkeitsverteilung. Für eine Variable y ist sie definiert als:

$$\mathrm{kurt}(y) = \langle y^4 \rangle - 3(\langle y^2 \rangle)^2. \tag{2.18}$$

Die Kurtosis ist so definiert, dass sie für gaussförmige Variablen verschwindet und für fast alle nichtgaussförmigen Variablen von Null verschieden ist. Die zwei großen Vorteile der Kurtosis sind, dass sie einfach numerisch umzusetzen ist und dass sie theoretisch leicht zu handhaben ist, da gilt:

$$\mathrm{kurt}(x_1 + x_2) = \mathrm{kurt}(x_1) + \mathrm{kurt}(x_2) \tag{2.19}$$

und

$$\mathrm{kurt}(\alpha x_1) = \alpha^4 \mathrm{kurt}(x_1) \tag{2.20}$$

Jedoch hat die Kurtosis den Nachteil, dass sie sehr sensitiv auf Ausreisser in den Daten reagiert. Demnach ist sie kein robustes Maß für Nichtgaussizität.

Das zweite Maß ist die Negentropie und basiert auf der informationstheoretischen Definition der Entropie. Die Entropie H ist eine Art Maß für die Menge an Informationen, welche in einer Variablen enthalten sind. Sie ist für einen Zufallsvektor $\vec{y}$ mit der Wahrscheinlichkeitsdichte $f(\vec{y})$ wie folgt definiert:

$$H(\vec{y}) = -\int f(\vec{y}) \log(f(\vec{y})) d\vec{y}. \tag{2.21}$$

Ein normalverteilter Vektor hat die höchste Entropie im Vergleich mit Zufallsvektoren, welche dieselbe Varianz haben, jedoch anderen Wahrscheinlichkeitsverteilungen folgen. Um ein Maß zu finden, welches für einen normalverteilten Vektor verschwindet und ansonsten immer negativ ist, kann die Negentropie J definiert werden:

$$J(\vec{y}) = H(\vec{y}_{gauss}) - H(\vec{y}). \tag{2.22}$$

Die Negentropie steht im Vergleich zur Kurtosis auf einem solideren theoretischen Fundament. Die Nachteile der Negentropie treten in den Bereichen auf, in denen die Kurtosis ihre Stärken hat und die Kurtosis besitzt Stärken in den Bereichen, in welchen die Negentropie Schwächen besitzt. Zum einen ist es numerisch aufwendig die Negentropie zu nutzen, da man entweder schon die gesamte Wahrscheinlichkeitsdichte $f(\vec{y})$ kennen muss, oder aber diese möglichst gut approximieren muss. Zum anderen ist die Negentropie ein robusteres Maß für Nichtgaussizität im Verleich zur Kurtosis.

Daher ist es sinnvoll die Negentropie auf eine Weise zu approximieren, welche die Vorteile der Kurtosis mit einbringt.

Es ist möglich die Negentropie mit Momenten höherer Ordnungen zu approximieren, jedoch sind diese Näherungen oftmals, wie die Kurtosis, kein robustes Maß. In [Hyv98] wurden daher neue Approximationen der Negentropie auf Basis des maximum-entropy Prinzip entwickelt. Diese Näherungen haben allgemein die Form:

$$J(y) \approx \sum_{i=1}^{p} k_i(\langle G_i(y)\rangle - \langle G_i(y_{gauss})\rangle)^2. \tag{2.23}$$

Hier werden mit k_i positive Konstanten bezeichnet, y und y_{gauss} sind Zufallsvariablen ohne Mittelwert und besitzen auf eins normierte Varianz. y_{gauss} folgt zudem einer Gauß-Verteilung. Diese Forderungen können durch Vorverarbeitung im Algorithmus mit eingebunden werden.

Folgende Funktionen haben sich als gute Näherungen erwiesen:

$$G_1(u) = \frac{1}{a_1} \log\cosh(a_1 u) \tag{2.24}$$

mit einer Konstanten $1 \leq a_1 \leq 2$.

$$G_2(u) = -\exp\left(-\frac{u^2}{2}\right). \tag{2.25}$$

2.3.3 FastICA-Algorithmus

Die Idee des FastICA-Algorithmus ist es, einen zufälligen Vektor $\vec{w}$ auszusuchen, welcher einen Spaltenvektor der inversen Mischungsmatrix $\boldsymbol{W}$ darstellt. Für eine optimale Schätzung von $\vec{w}$ entspricht $\vec{w}^T\vec{x}$ nun einem Grundsignal $\vec{s_i}$, was einer Maximierung der Nichtgaussizität entspricht. Ist die Nichtgaussizität noch nicht maximiert, wird die Richtung ermittelt, in welcher sich die Nichtgaussizität am stärksten ändert. Mit einer Art Newtonverfahren wird so eine Richtung mit höherer Nichtgaussizität ausgewählt. Dieses geschieht solange bis der Vektor $\vec{w}$ sich nicht mehr ändert. Das Ablaufschema des Algorithmus ist folgendes:

1. Ein (zufälliger) Startvektor $\vec{w}$ wird gewählt
2. $\vec{w}_{old} = \vec{w}$

3. $\vec{w}^+ = \langle \vec{x} \cdot g(\vec{w}^T \cdot \vec{x}) \rangle - \langle g^{'}(\vec{w}^T \cdot \vec{x}) \rangle \cdot \vec{w}$
4. $\vec{w} = \vec{w}^+ / ||\vec{w}^+||$
5. Vergleich von $\vec{w_{old}}$ und $\vec{w}$ über das Skalarprodukt der beiden Vektoren
6. Ist die Differenz von 1 und dem Skalarprodukt kleiner als eine Toleranzgrenze ϵ entspricht $\vec{w}$ einer Komponente der inversen Mischungsmatrix. Ansonsten werden Schritt 2-5 wiederholt.

Sollen nun alle Komponenten von $\boldsymbol{W}$ bzw. alle Grundsignale $\vec{s_i}$ bestimmt werden, muss nach Schritt 2 der aktuelle Vektor $\vec{w}$ in Bezug zu den bereits bestimmten Komponenten von $\boldsymbol{W}$ dekorelliert werden. Dieses geht nach Gram-Schmidt wie folgt:

$$\vec{w}_{p+1} = \vec{w}_{p+1} - \sum_{j=1}^{p} (\vec{w}_{p+1} \cdot \vec{w}_j) \cdot \vec{w}_j \tag{2.26}$$

$$\vec{w}_{p+1} = \frac{\vec{w}_{p+1}}{||\vec{w}_{p+1}||} \tag{2.27}$$

In der Praxis werden die Daten vorverarbeitet, bevor der FastICA-Algorithmus angewendet wird. Diese Vorverarbeitung besteht aus einer Zentrierung, indem der Mittelwert der Zeitreihen von den jeweiligen Zeitreihen subtrahiert wird. Danach folgt eine PCA um die Daten in einen niedrigerdimensionalen Unterraum zu projizieren.

Eine beispielhafte Implementation des Fast-ICA Algorithmus in Python befindet sich in AnhangC.

2.4 Dynamic Mode Decomposition

Das dritte Analyseverfahren, namentlich Dynamic Mode Decomposition, basiert im Vergleich zu den anderen beiden Verfahren auf einem grundlegend unterschiedlichen Ansatz. Die Idee der Dynamic Mode Decomposition ist es ein System in Eigenmoden des Koopman-Operators zu entwickeln. Dieser Operator wird im folgenden Teil definiert.

2.4.1 Koopman Operator

Wir folgen hier der Definition des Koopman-Operators aus [CWRH]. Zur Definition des Operators wird ein diskretes dynamisches System definiert, welches zum Beispiel durch diskretisierte Messung eines kontinuierlichen Systems erzeugt werden kann. Der Koopman-Operator kann jedoch in gleicher Weise auch für kontinuierliche Systeme definiert werden.

Das zu untersuchende dynamische System wird definiert als:

$$x_{k+1} = \boldsymbol{A} x_k. \tag{2.28}$$

x_i befindet sich auf einer beliebigen endlichdimensionalen Mannigfaltigkeit M. Für eine beliebige skalarwertige Funktion $g(x) : M \to \mathbb{R}$ ist der Koopman-Operator U definiert als:

$$Ug(x) = g(\boldsymbol{A}x). \tag{2.29}$$

Dieser Operator ist sozusagen das Pendant zum quantenmechanischen Zeitentwicklungsoperator. Er ist linear für beliebige Funktionen $g(x)$.

Die Eigenfunktionen $\phi_j(x)$ und die zugehörigen Eigenwerte λ_j sind für $j = 1, 2...$ definiert als:

$$U\phi_j(x) = \lambda_j \phi_j(x) \tag{2.30}$$

2.4.2 Berechnung der Dynamic Mode Decomposition

Da die, dem System zu Grunde liegende Dynamik $\boldsymbol{A}$ als a priori nicht bekannt angenommen wird, und somit auch der Koopman-Operator nicht bekannt ist, muss eine Berechnung der Koopman-Moden approximativ durchgeführt werden. Diese, auf approximierten Koopman-Moden beruhende, Dekomposition ist als Dynamic Mode Decomposition bekannt. Der im Folgenden beschriebene Algorithmus, basiert auf [Sch10] und nutzt eine Singular-Value Decomposition. Hier werden zuerst die Zeitreihen $x_i(t)$ mit $i = 1, 2, ..., N$ in zwei Matrizen $\boldsymbol{X}$ und $\boldsymbol{Y}$ geschrieben:

$$\boldsymbol{X} = (x_1, x_2, ..., x_{N-1}) \tag{2.31}$$

und

$$\boldsymbol{Y} = (x_2, x_3, ..., x_N). \tag{2.32}$$

Für die Matrix $\boldsymbol{X}$ wird dann eine Singular Value Decomposition durchgeführt und man erhält die drei Matrizen $\boldsymbol{U}$,$\boldsymbol{\Sigma}$,$\boldsymbol{V}$. Nun berechnet man eine neue Matrix $\tilde{\boldsymbol{A}}$:

$$\tilde{\boldsymbol{A}} = \boldsymbol{U}^* \boldsymbol{Y} \boldsymbol{V} \boldsymbol{\Sigma}^{-1}. \tag{2.33}$$

Die Eigenwerte λ_i von $\tilde{\boldsymbol{A}}$ sind Annäherungen an die Eigenwerte des Koopman-Operators und werden DMD-Eigenwerte genannt. Aus den zugehörigen Eigenvektoren $\vec{w}_i$ lassen sich die zugehörigen DMD-Eigenvektoren $\vec{v}_i$ berechnen:

$$\vec{v}_i = \boldsymbol{U}\vec{w}_i. \tag{2.34}$$

Für viele Bereiche, in denen die DMD Anwendung findet, ist diese Berechnung nicht praktikabel. Wenn die zeitliche Dimension N_t kleiner ist, als die räumliche DimensionN_x, erzeugt der beschriebene Algorithmus Eigenvektoren, deren Dimension der zeitlichen Dimension entspricht. Aus diesen Eigenvektoren lassen sich daher keine räumlichen Moden ableiten. Gerade bei fluiddynamischen Simulationen ist jedoch dieses der Fall. Vor allem bei räumlich dreidimensionalen Simulationen trifft schnell man auf dieses Problem.
In [CTR12] wird beschrieben, wie man obigen Algorithmus abändert, um dieses Problem zu umgehen. Zuerst diagonalisiert man $\boldsymbol{X}^*\boldsymbol{X}$, sodass gilt:

$$\boldsymbol{X}^*\boldsymbol{X} = \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^*. \tag{2.35}$$

Aus $\boldsymbol{\Sigma}$ und $\boldsymbol{V}$ berechnet man

$$\boldsymbol{U} = \boldsymbol{X}\boldsymbol{V}\boldsymbol{\Sigma}^{-1}. \tag{2.36}$$

Die so berechneten Matrizen $\boldsymbol{U}$,$\boldsymbol{V}$ und $\boldsymbol{\Sigma}$ setzt man dann in Gleichung 2.33 ein und verfährt weiter, wie in dem normalen Algorithmus. Somit muss man nur eine $N_t \times N_t$-Matrix diagonalisieren. Durch Berechnung von 2.34 erhält man schließlich Eigenvektoren der Dimension N_x.

Eine beispielhafte Implementation der beiden DMD-Varianten in Python befindet sich in Anhang B.

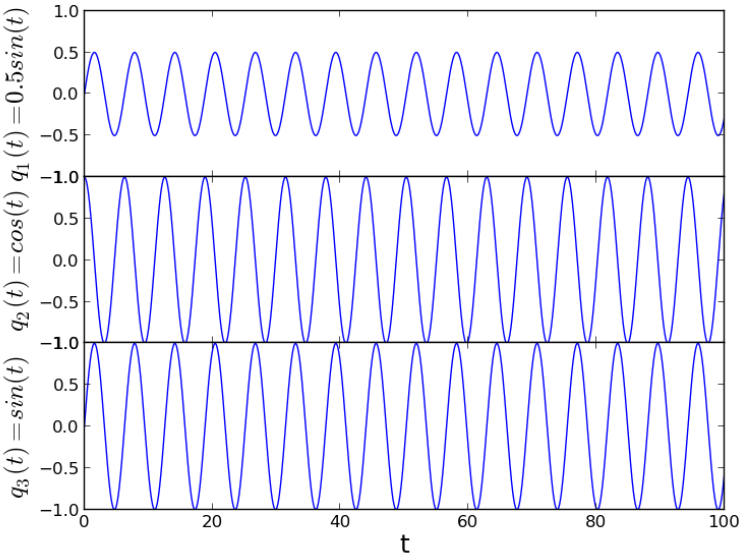


Abbildung 1: Zeitreihen der Kreisbewegung

3 Analyse zeitlicher Signale

3.1 PCA

Hier wird die PCA beispielhaft auf zeitliche Signale angewandt, um die Möglichkeit der Dimensionsreduktion darzustellen und anhand eines zweiten Systems die Unterschiede zur DMD zu zeigen.

3.1.1 Kreisbewegung

Zuerst wird, wie auch in [Daf04], eine Kreisbewegung untersucht, welche beliebig im 3-dimensionalen kartesischen Raum liegt und somit über drei verschiedene Zeitreihen $q_i(t)$ dargestellt wird(siehe Abbildung 1und 3):

$$q_1(t) = 0.5\sin(t)\ ,\ q_2(t) = \cos(t)\ ,\ q_3(t) = \sin(t) \tag{3.1}$$

Bei Durchführung der PCA wird nun erwartet, dass von drei Eigenwerten lediglich zwei von null verschieden sind. Somit könnte man die Bewegung in einem neuen Koordinatensystem darstellen, in welchem zwei der drei Achsen in der Ebene liegen, welche von der Kreisbewegung aufgespannt wird.

Das Eigenwertspektrum der Analyse ist in Abbildung 4 dargestellt. Tatsächlich erhält man nur zwei von null verschiedene Eigenwerte. Die zu den insgesamt drei Eigenwerten zugehörigen Eigenvektoren ermöglichen es, das System in einen zweidimensionalen Unterraum zu projizieren. Hierzu wählt man eine Matrix mit den Eigenvektoren als Spaltenvektoren. Durch Multiplikation dieser Matrix mit dem Eingangssignal $\vec{q}(t)$ wird $\vec{q}(t)$ auf den Unterraum projiziert. Die Komponenten $\eta_i(t)$ des Vektors im Unterraum und die resultierende Kreisbewegung sind in Abbildung 5 und 2 dargestellt.

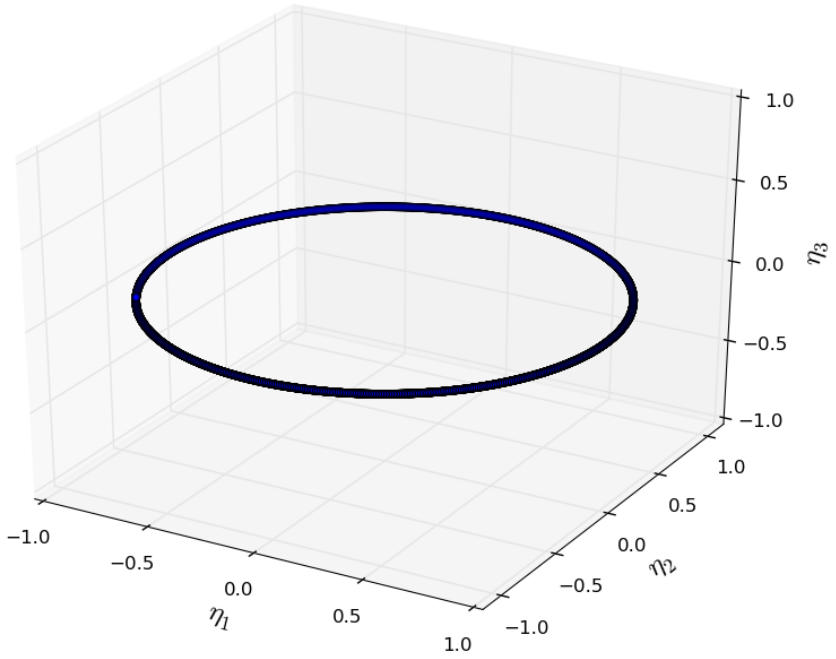


Abbildung 2: 3D Plot der Kreisbewegung

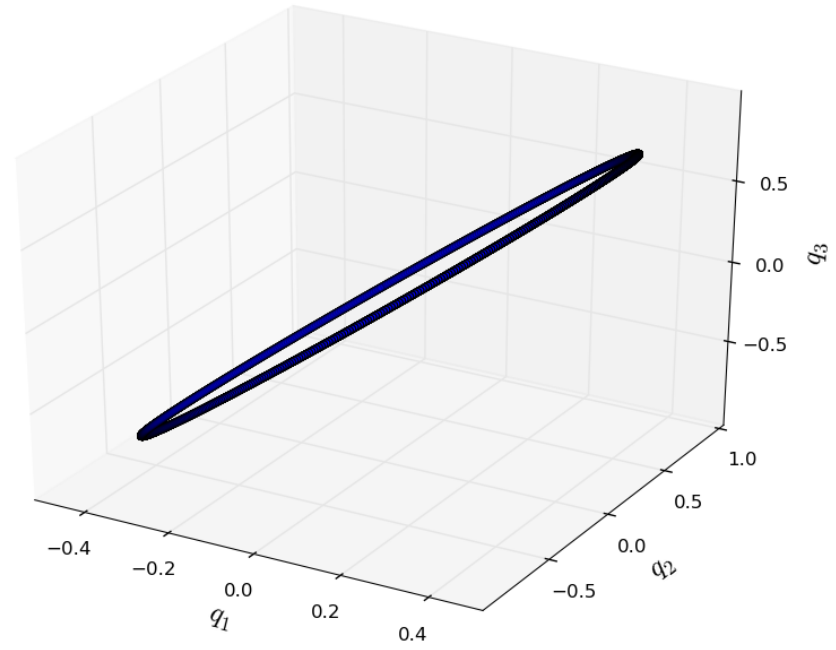


Abbildung 3: 3D Plot der Kreisbewegung

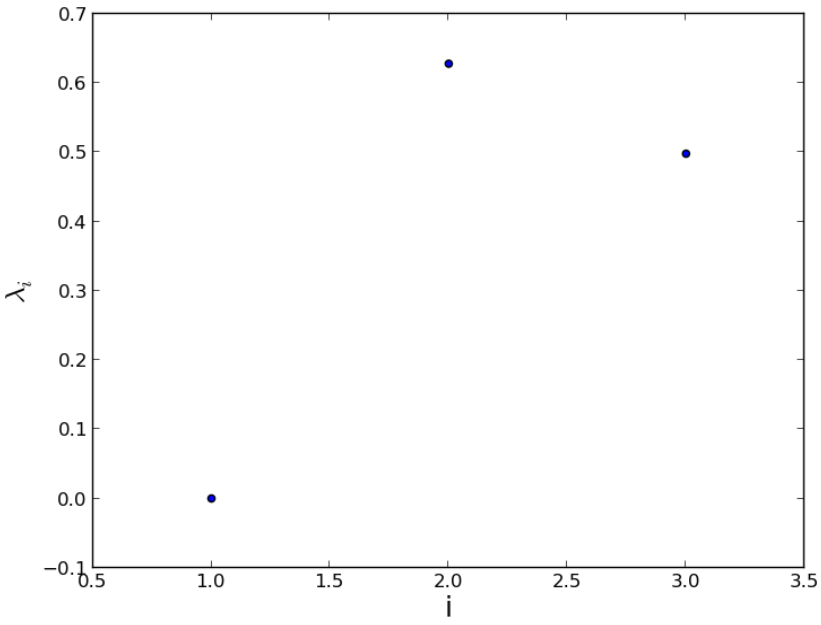


Abbildung 4: Eigenwerte von $\vec{q}(t)$

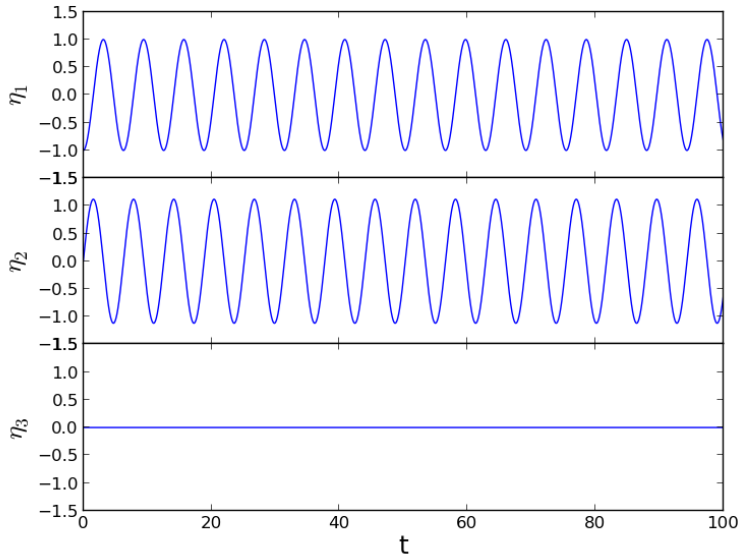


Abbildung 5: Zeitreihen der Kreisbewegung in den Moden v_i

3.1.2 PCA des Haken-Zwanzig Modell

Als nächstes werden die Zeitreihen des Haken-Zwanzig Modells untersucht. Das Haken-Zwanzig Modell ist ein System aus zwei nichtlinear gekoppelten Differentialgleichungen erster Ordnung in den Variablen u und s:

$$\dot{u} = \epsilon \cdot u - u \cdot s \tag{3.2}$$

$$\dot{s} = -\gamma \cdot s + u^2 \tag{3.3}$$

Dieses Differentialgleichungssystem wurde mit einem Runge-Kutta Verfahren vierter Ordnung mit folgender Parameterwahl simuliert:

- Schrittweite $s = 0.01$
- Gesamtzeit $t = 100$
- $\epsilon = 0.01$
- $\gamma = 15$
- Startpunkt $u(0) = s(0) = 1$

In Abbildung 6 und 7 sind der Phasenraum und die Trajektorien der beiden Variablen dargestellt.

Führt man für diese Simulation die PCA durch, erhält man zwei Eigenwerte mit den zugehörigen Eigenvektoren. Diese sind beliebig in den Raum gelegt dargestellt in Abbildung 9.

3.2 ICA

Um die Funktion der ICA zu demonstrieren, wird sie auf ein Beispiel angewendet, welches nichtorthogonale Grundsignale hat. Anhand dieses Beispiels wird ein Fall aufgezeigt, bei welchem die PCA im Gegensatz zur ICA Signale nicht korrekt rekonstruiert.

Die beiden Signale sind:

$$s_1 = \sin(2t) \tag{3.4}$$

$$s_2 = \text{sign}(\sin(3t)) \tag{3.5}$$

Beiden Signale werden zusätzlich mit einem normalverteilten Rauschen überlagert, welches eine Amplitude von $0,2$ hat.

Die beobachteten Signale sind Mischungen aus den beiden Grundsignalen:

$$x_1(t) = s_1(t) + s_2(t) \tag{3.6}$$

$$x_2(t) = 0,5s_1(t) + 2s_2(t) \tag{3.7}$$

Die Signale sind zusammen mit den gemischten Signalen und den jeweiligen Rekonstruktionen von PCA und ICA in Abbildung 10 abgebildet.

Man erkennt, dass die PCA die Signale nicht richtig rekonstruiert. Die ICA hingegen gibt die zugrundeliegenden Signale $s_i(t)$ korrekt wieder, bis auf Skalierung und Vorzeichen, was ein inhärentes Problem der ICA ist.

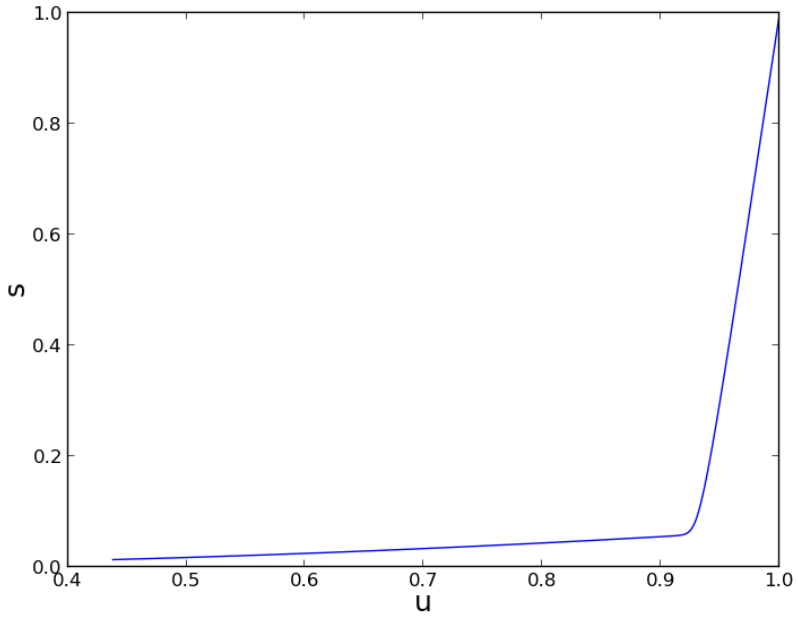


Abbildung 6: Phasenraum des Haken-Zwanzig-Modells für die angegebenen Parameter

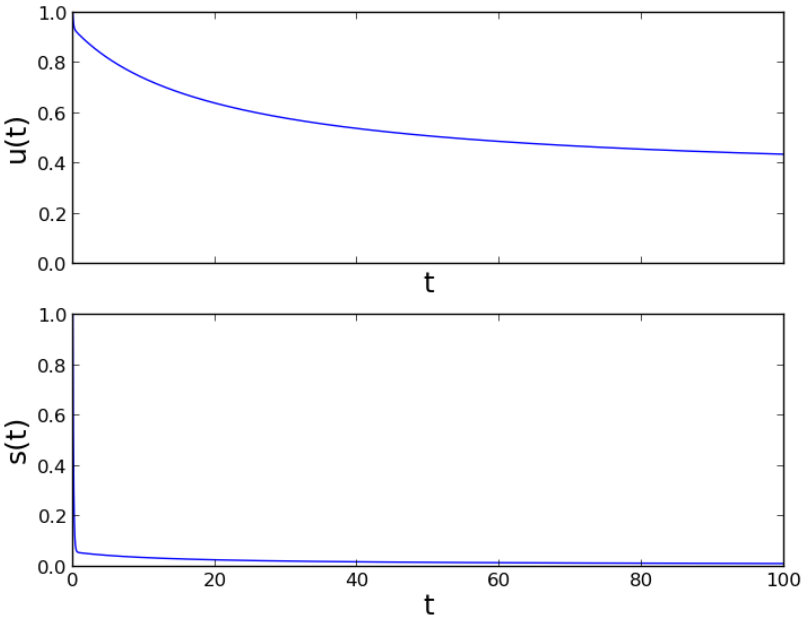


Abbildung 7: Zeitreihen von $u(t)$ und $s(t)$

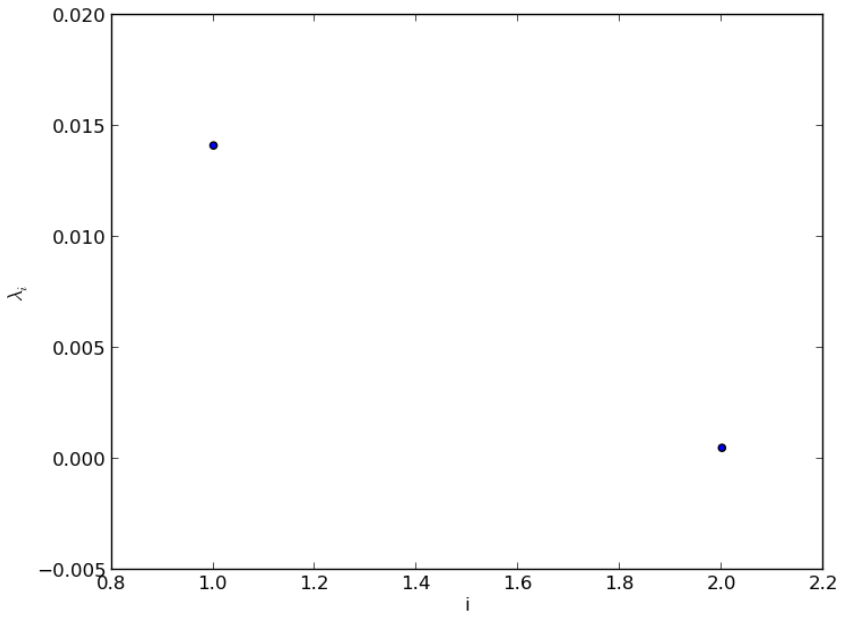


Abbildung 8: PCA-Eigenwerte des Haken-Zwanzig-Modells

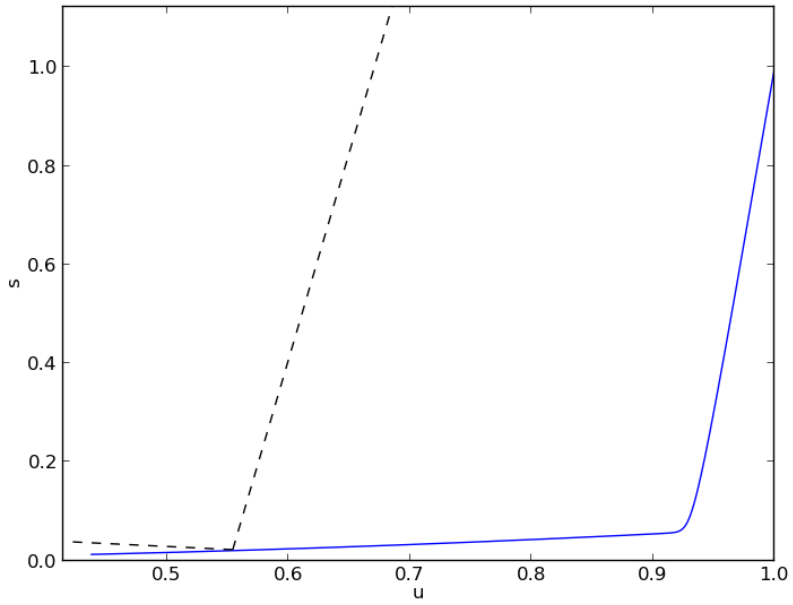


Abbildung 9: Phasenraum des Haken-Zwanzig-Modells mit den PCA-Eigenvektoren (gestrichelt)

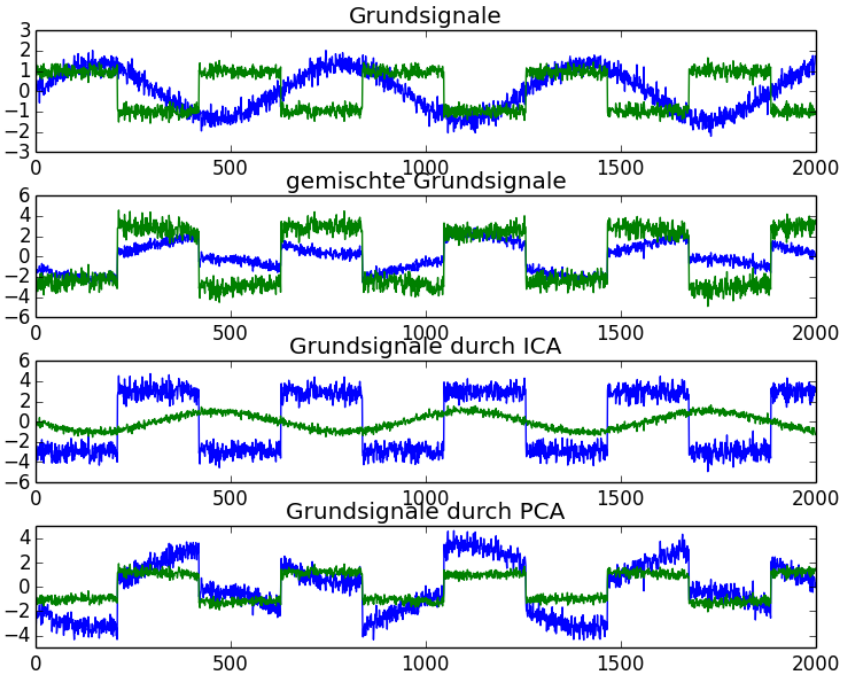


Abbildung 10: Grundsignale, gemischte Signale, PCA und ICA Rekonstruktion

3.3 DMD

Nun wird die DMD auch auf das in 3.1.2 simulierte Haken-Zwanzig Modell angewendet. Mit der DMD ergeben sich auch zwei Eigenwerte, welche nur einen Realteil besitzen und keinen Imaginärteil (siehe Abb. 11). Diese beiden Eigenwerte liegen betragsmäßig nah aneinander, was ein Anzeichen dafür ist, dass die DMD die schneller abklingende Variable s anders als die PCA bewertet. In Abbildung12 sind die zugehörigen DMD-Eigenvektoren abgebildet. Auch hier ist zu erkennen, dass die DMD insgesamt andere Moden erkennt, als die PCA.

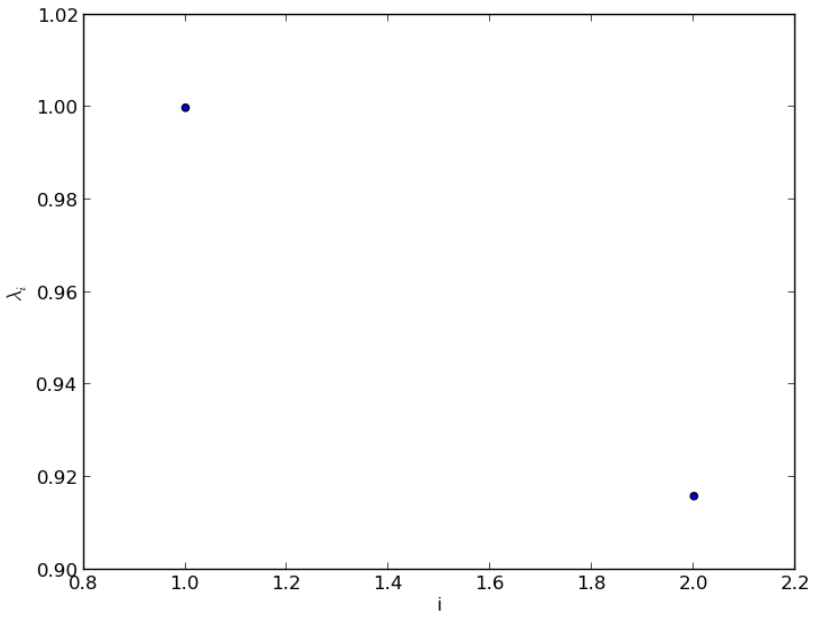


Abbildung 11: DMD-Eigenwerte des Haken-Zwanzig-Modells

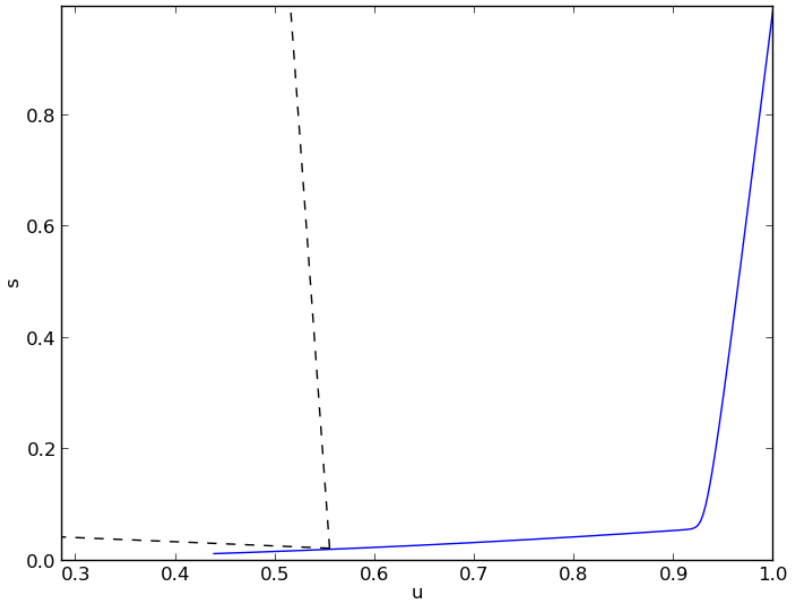


Abbildung 12: Phasenraum des Haken-Zwanzig-Modells mit den DMD-Eigenvektoren (gestrichelt)

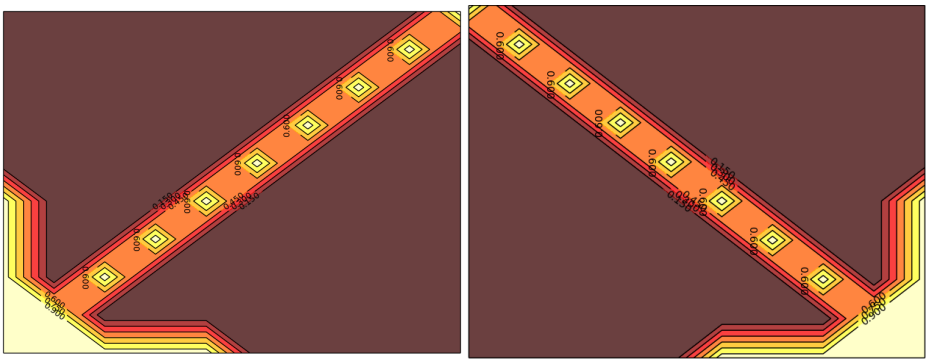


Abbildung 13: Muster 1(links) und 2(rechts)

4 Analyse raumzeitlicher Signale

Um die Anwendung der PCA und der DMD auf raumzeitliche Signale zu verdeutlichen, werden zwei selbst gewählte orthogonale räumliche Moden einmal linear, als harmonischer Oszillator und einmal nichtlinear, mit dem Haken-Zwanzig-Modell, miteinander gekoppelt.

In beiden Fällen werden als die zwei Grundmuster zwei Pfeile (Abbildung 13) gewählt, welche jeweils über eine 10x10 Matrix dargestellt werden. Die Erzeugung und Umwandlung dieser Muster in Python ist in Anhang D dargestellt. Für die Analyse werden diese beiden Matrizen jeweils nach einem eindeutigem und umkehrbaren Schema in einen Vektor umgewandelt. Ebenso werden die Eigenvektoren danach wieder in Matrizen umgewandelt. Auf Grund der Größe der Matrizen ergibt die PCA 100 Eigenwerte λ_i mit zugehörigen Eigenvektoren $\vec{v}_i$.

4.1 ICA

Wie oben erwähnt, werden die raumzeitlichen Signale nur mit der PCA und der DMD analysiert. Dieses ist damit begründet, dass die ICA nicht für die angegebenen Zwecke sinnvoll ist. Die raumzeitlichen Grundmuster, welche durch die Analyse ermittelt werden sollen, entsprächen in der Theorie der ICA den Eigenvektoren der ermittelten invertierten Mischungsmatrix $\boldsymbol{W}$. Da in der Praxis zuerst eine PCA durchgeführt wird, um das Problem dimensionell zu reduzieren, berechnet die ICA nicht die Eigenvektoren der eigentlichen inversen Mischungsmatrix $\boldsymbol{W}$, sondern die einer Matrix $\boldsymbol{W'}$. Für diese Matrix $\boldsymbol{W'}$ gilt:

$$\vec{s}(t) = \boldsymbol{W'}\vec{x'}(t) = \boldsymbol{W'}\boldsymbol{A}\vec{x}(t) = \boldsymbol{W}\vec{x}(t), \tag{4.1}$$

mit der Matrix $\boldsymbol{A}$ der PCA Eigenvektoren und dem Signalvektor $\vec{x}(t)$. Für die echte inverse Mischungsmatrix müsste man also das Produkt von $\boldsymbol{A}$ und $\boldsymbol{W'}$ bestimmen. Da die Reihenfolge der Eigenvektoren von $\boldsymbol{W'}$ nicht eindeutig bestimmt werden kann, kann dieses Produkt und somit $\boldsymbol{W}$ auch nicht eindeutig bestimmt werden. Somit lassen sich nicht die der eigentlichen Dynamik zu Grunde liegenden raumzeitlichen Muster bestimmen.

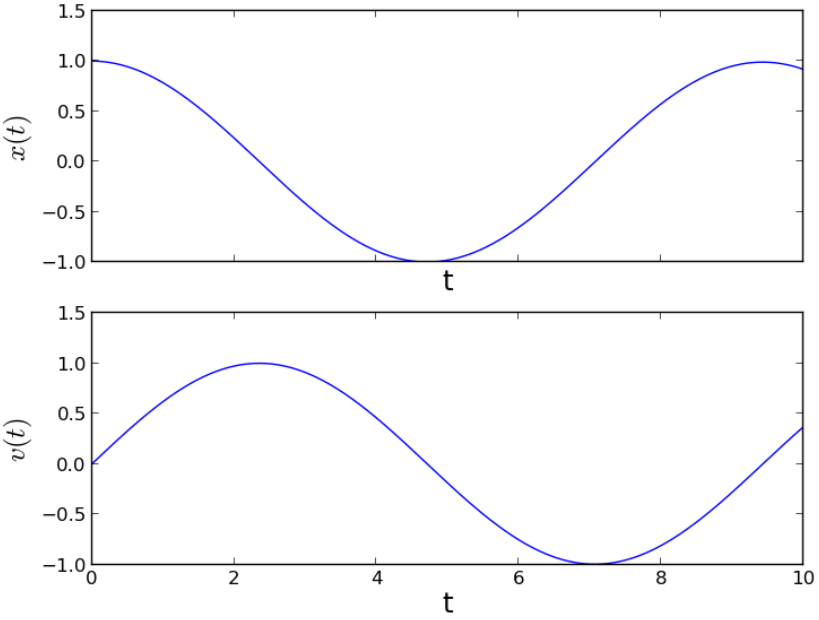


Abbildung 14: Zeitreihen x(t) und v(t) des harmonischen Oszillators

4.2 Lineare Kopplung

Als erstes Beispiel koppeln wir die beiden Muster an die Dynamik des harmonischen Oszillators. Hierzu wird der harmonische Oszillator über zwei gekoppelte Differentialgleichungen erster Ordnung dargestellt:

$$\dot{x} = -\omega \cdot v \tag{4.2}$$

$$\dot{v} = x \tag{4.3}$$

Um die beiden Muster mit den Differentialgleichungen zu koppeln, werden sie als Vektoren $\vec{m}_1$ und $\vec{m}_2$ dargestellt. Unser zu untersuchender Signalvektor ergibt sich hiernach zu:

$$\vec{q}(t) = x(t) \cdot \vec{m}_1 + v(t) \cdot \vec{m}_2 \tag{4.4}$$

Die beiden Zeitreihen $x(t)$ und $v(t)$ (siehe Abbildung 14) wurden jeweils mit einem Runge-Kutta-Verfahren vierter Ordnung erzeugt. Die Wahl von ω beeinflusst das Ergebnis nicht, da vor der Durchführung der Analyse die beiden Zeitreihen auf eins normiert wurden.

Das Eigenwertspektrum (Abbildung 15), welches die PCA liefert, hat nur zwei von null verschiedene Eigenvektoren λ_1 und λ_2. Da die zwei Eingangsmuster in der Vektordarstellung orthogonal zueinander sind, sollten die beiden zu λ_1 und λ_2 zugehörigen Eigenvektoren $\vec{v}_1$ und $\vec{v}_2$ die zuvor gewählten Muster sein. Es zeigt sich, dass die Eigenvektoren $\vec{v}_1$ und $\vec{v}_2$ (Abbildung 17) in guter Übereinstimmung mit den gewählten Mustern sind.

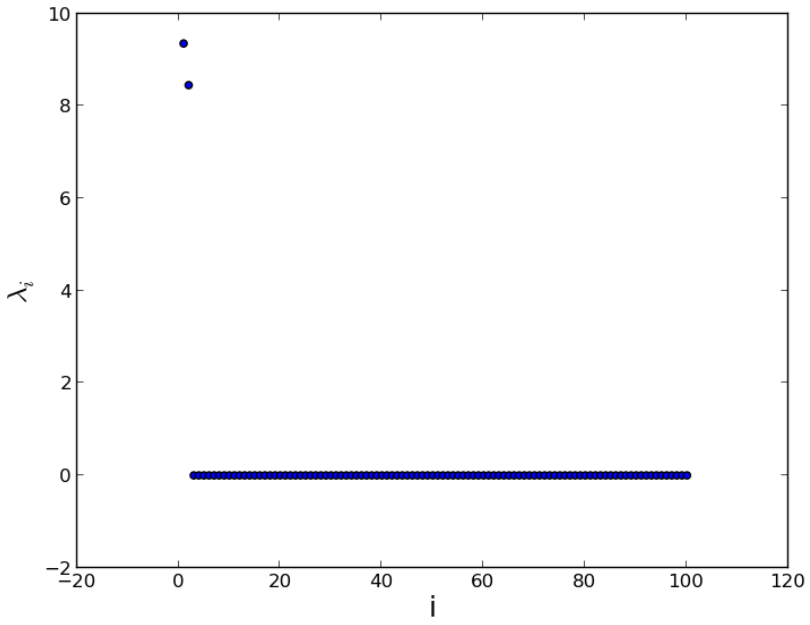


Abbildung 15: PCA-Eigenwertspektrum von $\vec{q}(t)$

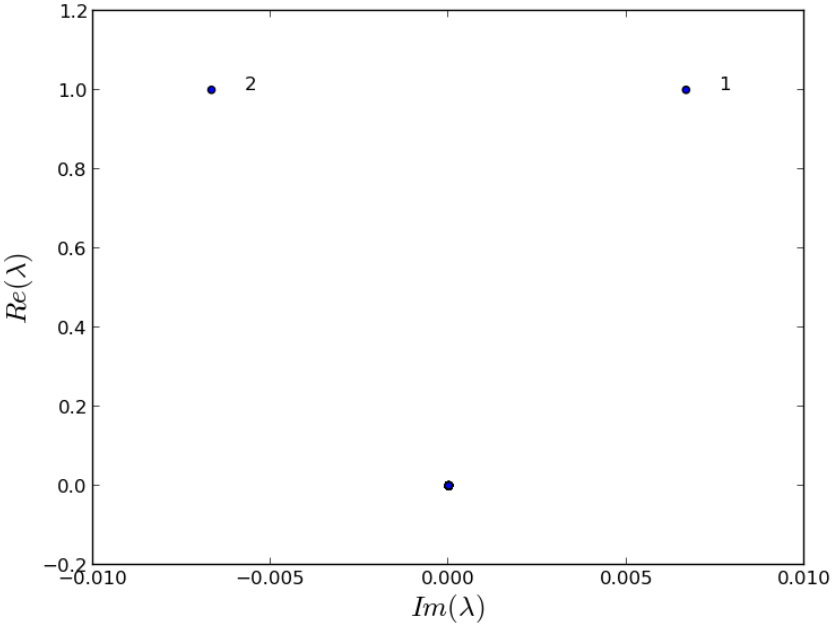


Abbildung 16: DMD-Eigenwertspektrum von $\vec{q}(t)$

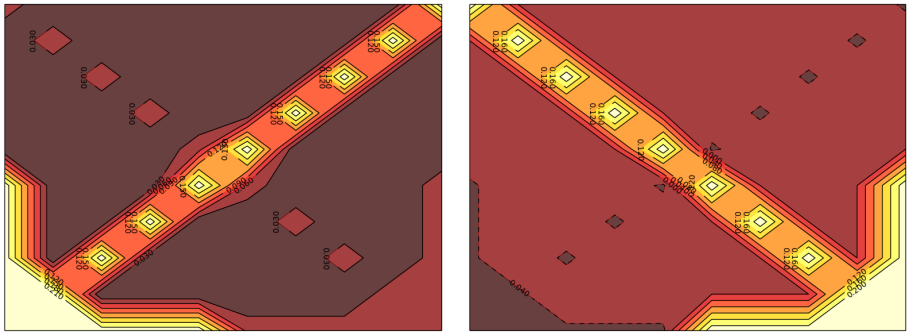


Abbildung 17: PCA-Eigenvektor $\vec{v}_1$(links) und $\vec{v}_2$(rechts) des harm. Oszillators.

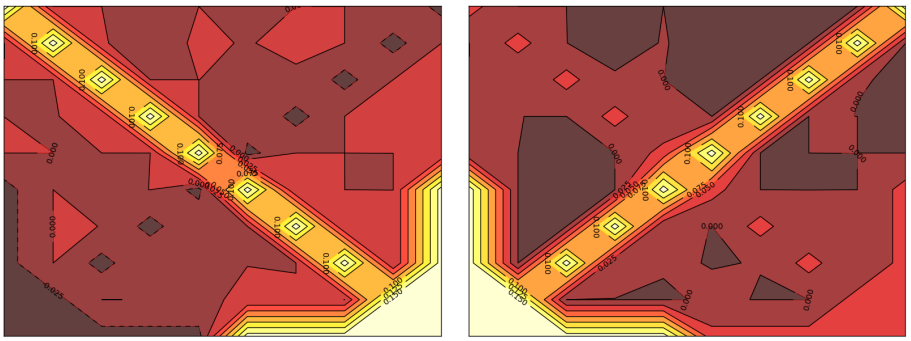


Abbildung 18: Real-(links) und Imaginäranteil (rechts des DMD-Eigenvektors $\vec{v}$ des harm. Oszillators.

Das Eigenwertspektrum der DMD in Abb.16 besitzt zwei komplexe von null verschiedene Eigenwerte, markiert in der Abbildung durch 1 und 2. Ihre Realanteile sind gleich groß und ihre Imaginäranteile weisen entgegengesetzte Vorzeichen auf. Sie besitzen jeweils denselben komplexwertigen Eigenvektor $\vec{v}$. Die Imaginäranteile der Eigenwerte zeigen, dass die Eigenvektoren ein oszillatorisches Verhalten besitzen. In der Abbildung 18 sind der Real- und der Imaginäranteil von $\vec{v}$ dargestellt. Man erkennt, dass die Anteile die gewählten Grundmuster widerspiegeln.

4.3 Nichtlineare Kopplung

Im folgenden wird, wie im vorigen Abschnitt, ein raumzeitliches Signal als Überlagerungen zweier orthogonaler Muster untersucht, welche an zwei verschiedene Zeitreihen gekoppelt sind. Dieses Mal wird für die Zeitreihen das Haken-Zwanzig-Modell, welches in 3.1.2 eingeführt wurde, gewählt. Somit sind hier die beiden Muster nichtlinear gekoppelt im Gegensatz zur linearen Kopplung beim harmonischen Oszillator.

Die expliziten genutzten Zeitreihen entsprechen denen in Abschnitt 3.1.2. Der zu analysierende Vektor mit den Mustervektoren $\vec{m}_i$ ist in diesem Fall:

$$\vec{q}(t) = u(t) \cdot \vec{m}_1 + s(t) \cdot \vec{m}_2. \tag{4.5}$$

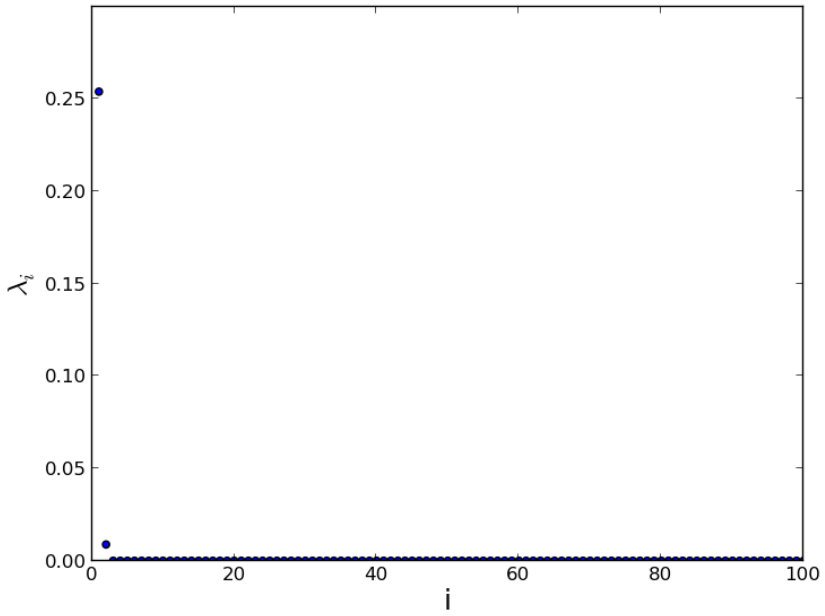


Abbildung 19: Eigenwerte der PCA für das Haken-Zwanzig-Modell

Am Phasendiagramm dieses Modells erkennt man, dass in diesem Parameterbereich eines der beiden Muster das System dominiert. Daher wird erwartet, dass wir zwar, wie beim harmonischen Oszillator, zwei von Null verschiedene Eigenwerte erhalten, jedoch diese sich in ihrem Betrag sehr stark unterscheiden.

Diese Annahme erweist sich nach der PCA als richtig. Es existieren wieder zwei PCA-Eigenwerte λ_1 und λ_2, welche nicht verschwinden (Abbildung 19). Die zugehörigen PCA-Eigenvektoren (Abb.20) entsprechen wieder größtenteils den Grundmustern. Jedoch enthält der zweite PCA-Eigenvektor noch nicht verschwindende Anteile des ersten PCA-Eigenvektors.

Die DMD ergibt zwei Eigenwerte mit von null verschiedenen Realanteilen (Abb.21), welche keine Imaginäranteile besitzen. Also zeigen die zugehörigen DMD-Eigenvektoren kein oszillatorisches Verhalten, was der gewählten Dynamik entspricht. Die zugehörigen DMD-Eigenvektoren (Abb. 22) spiegeln die Grundmuster besser wider als die PCA-Eigenvektoren, da hier der zweite Eigenvektor keinen Anteil des ersten enthält. Für das gewählte nichtlineare System ist die DMD also besser geeignet als die PCA. Dieses wird darüber erklärt, dass die DMD die dynamisch relevanten Moden berechnet und die PCA die Moden berechnet, welche die meiste Energie besitzen.

Um die Güte der beiden Analyseverfahren besser darzustellen, wurden jeweils die Beträge der Skalarprodukte zwischen den beiden Grundmustern m_i und den beiden Moden v_i der PCA und der DMD berechnet. Hierfür wurden die Muster in Vektorform dargestellt und alle auf eins normiert. Es ergeben sich folgende Werte:

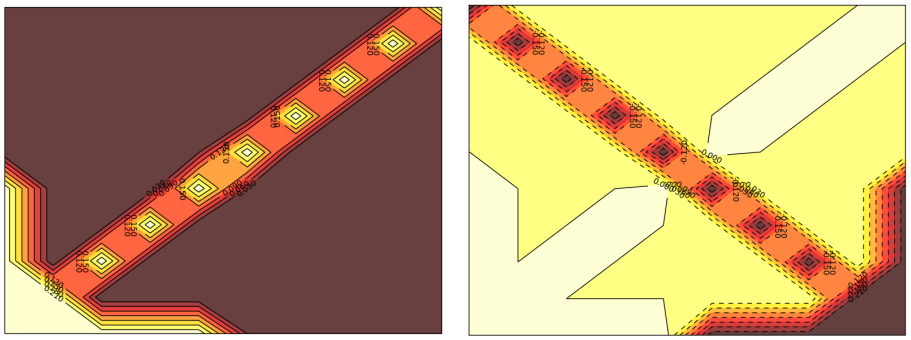


Abbildung 20: PCA-Eigenvektoren v_1 (links) und v_2 (rechts) des Haken-Zwanzig-Modells.

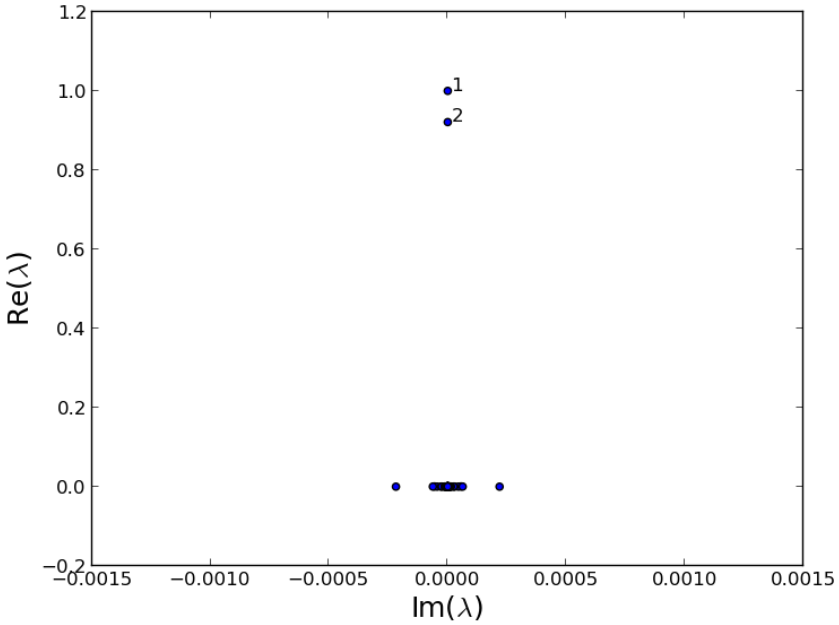


Abbildung 21: Eigenwerte der DMD für das Haken-Zwanzig-Modell

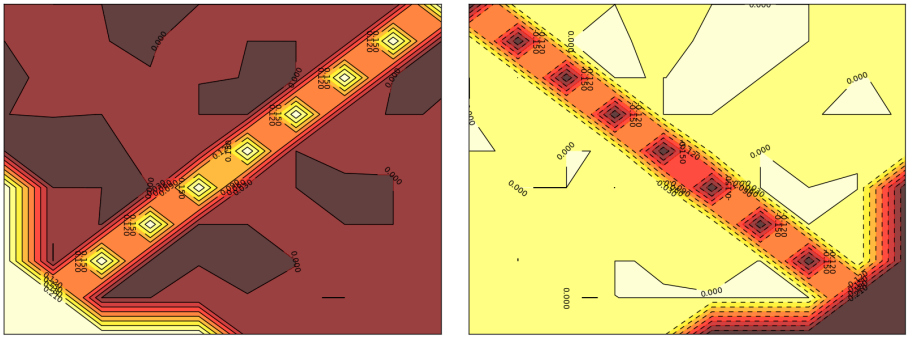


Abbildung 22: DMD-Eigenvektoren v_1 (links) und v_2 (rechts) des Haken-Zwanzig-Modells.

PCA:

	m_1	m_2
v_1	0.992976054072	0.118315493656
v_2	0.11831549366	0.992976054072

DMD:

	m_1	m_2
v_1	0.999121772871	0.0419008708206
v_2	0.0816720139946	0.996659260796

Man erkennt, dass die DMD-Moden jeweils die Grundmuster besser als die PCA-Moden repräsentieren. Außerdem enthalten die DMD-Moden signifikant weniger Anteile des jeweilig anderen Grundmusters, was die graphische Darstellung der Moden auch schon vermuten lässt.

5 Rayleigh-Bénard Konvektion

Bei der Rayleigh-Bénard Konvektion handelt es sich um die thermale Konvektion einer Flüssigkeit, welche sich zwischen zwei horizontalen Platten konstanter Temperatur in einem Schwerefeld befindet. Wobei die untere Platte eine höhere Temperatur als die obere Platte besitzt. Ist der Temperaturunterschied zwischen diese beiden Platten gering, wird die Wärme von der unteren Platte zur oberen Platte durch das Fluid geleitet (Konduktion). Ab einem kritischen Temperaturunterschied, welcher mit der (im späteren definierten) Rayleighzahl zusammenhängt, setzt sogenannte Konvektion ein, was bedeutet, dass die Wärme auch durch Bewegung des Fluids an sich transportiert wird.

5.1 Grundlagen der Rayleigh-Bénard Konvektion

Um dieses System theoretisch zu beschreiben, benötigt man drei Felder. Diese sind das skalare Temperaturfeld $T(\vec{x},t)$, das skalare Druckfeld $p(\vec{x},t)$ sowie das vektorielle Geschwindigkeitsfeld $\vec{u}(\vec{x},t)$ mit $\vec{u} = (u_x, u_y, u_z) \in \mathbb{R}^3$, $\vec{x} = (x,y,z) \in \mathbb{R}^3$ und $t \in \mathbb{R}$.

Die Evolution des Geschwindigkeitsfeldes wird über die Navier-Stokes Gleichung beschrieben mit der Schwerkraft als zusätzlichen Kraft:

$$\frac{\partial}{\partial t}\vec{u}(\vec{x},t) + \vec{u}(\vec{x},t)\cdot\nabla\vec{u}(\vec{x},t) = -\nabla p(\vec{x},t) + \nu\Delta\vec{u}(\vec{x},t) - g\frac{\rho}{\rho_0}\vec{e}_z \tag{5.1}$$

Es treten zusätzlich zu den bereits definierten Größen die kinematische Viskosität ν, die Dichte des Fluids ρ und die Erdbeschleunigung g auf. Die Nichtlinearität $\vec{u}\cdot\nabla\vec{u}$ beschreibt die Advektion des Fluids und der Druckterm ∇p ist nötig, um die Inkompressibilität des Fluids zu gewährleisten. Diese Gleichung ist über die temperaturabhängige Fluiddichte an die Temperatur gekoppelt. Zusätzlich lässt sich die Inkompressibilität des Fluids folgendermaßen ausdrücken:

$$\nabla\cdot\vec{u}(\vec{x},t) = 0 \tag{5.2}$$

Zuletzt wird die zeitliche Entwicklung der Temperatur aufgestellt:

$$\frac{\partial}{\partial t}T(\vec{x},t) + \vec{u}(\vec{x},t)\cdot\nabla T(\vec{x},t) = \kappa\Delta T(\vec{x},t) \tag{5.3}$$

Dieses ist eine Advektions-Diffusions-Gleichung, welche beschreibt wie die Temperatur mit der Geschwindigkeit des Fluids advektiert wird. κ steht für den Wärmeleitkoeffizienten des Fluides.
Boussinesq [Bou03] hat die Approximation eingeführt, dass die Dichte ρ in der Evolutionsgleichung des Geschwindigkeitsfeld, die einzige temperaturabhängige Größe ist. Die Koeffizienten ν und κ werden als temperaturunabhängig betrachtet. Des weiteren wird angenommen, dass die Dichte sich linear mit der Temperatur ändert. Das bedeutet, dass die betrachteten Temperaturschwankungen und die zugehörigen Dichteänderungen klein sind:

$$\rho = \rho_0[1 - \alpha(T - T_0)] \tag{5.4}$$

Hier bezeichnet ρ_0 die Dichte des Fluids bei einer Referenztemperatur T_0 und α den Wärmeausdehnungskoeffizienten. Die Grundgleichungen lassen sich mit dieser Approximation umformulieren:

$$\frac{\partial}{\partial t}\vec{u}\left(\vec{x},t\right)+\vec{u}\left(\vec{x},t\right)\cdot\nabla\vec{u}\left(\vec{x},t\right)=-\nabla p\left(\vec{x},t\right)+\nu\Delta\vec{u}\left(\vec{x},t\right)+\alpha gT\left(\vec{x},t\right)\vec{e}_z \quad (5.5)$$

$$\nabla\cdot\vec{u}\left(\vec{x},t\right)=0 \quad (5.6)$$

$$\frac{\partial}{\partial t}T\left(\vec{x},t\right)+\vec{u}\left(\vec{x},t\right)\cdot\nabla T\left(\vec{x},t\right)=\kappa\Delta T\left(\vec{x},t\right). \quad (5.7)$$

In dieser Form aufgeschrieben, heißen die Gleichungen Oberbeck-Boussinesq-Gleichungen.

5.2 Randbedingungen

Nachdem die Grundgleichungen des Systems nun aufgestellt wurden, ist es nötig die Randbedingungen des Rayleigh-Bénard-Systems zu spezifizieren. Wie bereits beschrieben, befindet sich das Fluid in diesem System zwischen zwei horizontalen Platten konstanten Abstandes h in z-Richtung. In x- und y-Richtung wird das System als unendlich weit ausgedehnt angesehen. Die untere Platte soll sich bei $z=0$ befinden und die obere bei $z=h$.
Für die Randbedingungen des Geschwindigkeitsfeldes gibt es zwei Möglichkeiten. Entweder wählt man 'no-slip'-Randbedingungen oder man wählt spannungsfreie Ränder. Im ersten Fall verschwindet das Geschwindigkeitsfeld in allen Komponenten für $z=0$ und $z=h$. Im zweiten Fall muss gelten:

$$u_z=\frac{\partial u_x}{\partial z}=\frac{\partial u_y}{\partial z}=0,\forall x,y,t \quad (5.8)$$

Die Komponenten u_x und u_y sind beliebig.
Die letzte Randbedingung ist, dass die beiden Platten eine konstante Temperatur besitzten. Es muss also gelten:

$$T(x,y,z=0,t)=T_u \quad (5.9)$$

und

$$T(x,y,z=h,t)=T_o \quad (5.10)$$

Wobei $T_u>T_o$ gelten muss und $T_u-T_o=\delta T$ ist.

5.3 Entdimensionalisierung

Um dieses System zu entdimensionalisieren, führt man eine Koordinatentransformation durch:

$$\frac{h}{\kappa}\vec{u}\rightarrow\vec{u}' \quad (5.11)$$

$$\frac{1}{\delta T}\rightarrow T' \quad (5.12)$$

$$\frac{h^2}{\kappa^2}p\rightarrow p' \quad (5.13)$$

$$\frac{1}{h}\vec{x} \to \vec{x}' \text{ bzw. } h\nabla \to \nabla' \text{ bzw. } h^2\Delta \to \Delta' \tag{5.14}$$

$$\frac{\kappa}{h^2}t \to t' \text{ bzw. } \frac{h^2}{\kappa}\frac{\partial}{\partial t'} \tag{5.15}$$

Im weiteren werden die Striche an den neuen Variablen weggelassen, um die Gleichungen leserlicher zu gestalten. Führt man nun noch die Prandtlzahl Pr und die Rayleighzahl Ra ein

$$Pr = \frac{\nu}{\kappa}\ ,\ Ra = \frac{\alpha g \delta T h^3}{\nu\kappa}, \tag{5.16}$$

lassen sich die Oberbeck-Boussinesq-Gleichungen entdimensionalisiert in Abhängigkeit von nur zwei Parametern darstellen.

$$\frac{\partial}{\partial t}\vec{u}\left(\vec{x},t\right)+\vec{u}\left(\vec{x},t\right)\cdot\nabla\vec{u}\left(\vec{x},t\right) = -\nabla p\left(\vec{x},t\right)+Pr\Delta\vec{u}\left(\vec{x},t\right)+PrRaT\left(\vec{x},t\right)\vec{e}_z \tag{5.17}$$

$$\nabla\cdot\vec{u}\left(\vec{x},t\right) = 0 \tag{5.18}$$

$$\frac{\partial}{\partial t}T\left(\vec{x},t\right) + \vec{u}\left(\vec{x},t\right)\cdot\nabla T\left(\vec{x},t\right) = \Delta T\left(\vec{x},t\right) \tag{5.19}$$

Da die Prandtl-Zahl Pr nur von den Konstanten ν und κ abhängt, ist es eine rein materialabhängiger Parameter, welcher das untersuchte Fluid festlegt. Die Rayleigh-Zahl Ra hingegen enthält mit der Systemhöhe h und der Temperaturdifferenz δT Parameter, welche extern beeinflusst werden können, ohne das Fluid zu tauschen. Im Experimenten kann somit durch alleinige Variation der Höhe des Systems oder der Temperaturdifferenz das Systemverhalten beeinflusst werden. Ab einer kritischen Rayleigh-Zahl Ra_c geht das Fluid von einem Zustand der Wärmeleitung in den Zustand der Konvektion über.

5.4 Analyse der Rayleigh-Bénard Konvektion

Nachdem die Funktionsweise der Principal Component Analyse und der Dynamic Mode Decomposition an einfachen Beispielen getestet wurden, werden diese Verfahren nun auf Daten einer Simulation der beschriebenen Konvektionsströmungen angewandt. Einziger Unterschied ist, dass die Simulation zweidimensional durchgeführt wurde. Dieses bedeutet, dass die y-Komponente der Felder entfällt.

Die Simulation wurde von J. Lülff zur Verfügung gestellt. Sie basiert auf einem Pseudospektralverfahren mit Volume-Penalization, welches im Rahmen von [Lue] entwickelt wurde. Es wurden folgende Parameter gewählt:

- $Ra = 10^6$
- $Pr = 1$
- räumliche Diskretisierung der Simulation: $256 * 64$

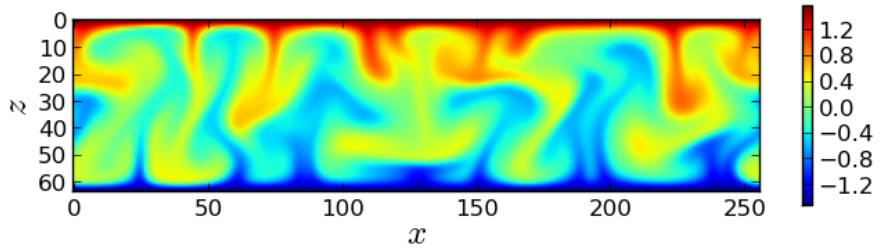


Abbildung 23: Snapshot des Temperaturfeldes bei Zeitschritt 22000.

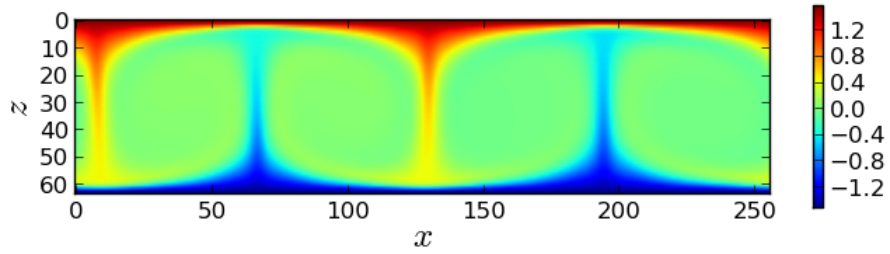


Abbildung 24: Snapshot des Temperaturfeldes bei Zeitschritt 160000.

Für die Analyse-Verfahren wird nun die Struktur untersucht, welche sich im Temperaturfeld ausbildet. Die bereitgestellte Simulation entwickelt nach einer gewissen Einschwingphase (siehe Abb. 23) einen, als stationär zu betrachtenden, Zustand (siehe Abb. 24). In den Abbildung ist die Temperatur farblich dargestellt. Rot entspricht heißeren Gebieten, Blau korrespondiert mit kälteren Gebieten.

In Abbildung 25 ist der mittlere Wärmetransport des Konvektionsprozesses in Abhängigkeit der Zeit dargestellt. Dieses ist als Maß für die Stationarität des Prozesses zu betrachten. Man erkennt, dass sich nach anfänglich starken Fluktuationen ein über längere Zeit stabiler Zustand ausbildet. Dieser Zustand wird mittels PCA und DMD analysiert. Bei beiden Analysen wird der Zeitraum von $t_a = 1600000$ bis $t_e = 1700000$ analysiert. Bei den Simulationsdaten entspricht das 100 Punkten in der Zeit. Somit muss im Falle der DMD die zweite Implementation gewählt werden. Die räumlichen Punkte werden so analysiert, dass nur jeder zweite Raumpunkt in die Analyse eingeht, um den Rechenaufwand zu verringern. Auch bei der PCA wird die Analyse mit der als zweites erwähnten Implementation durchgeführt, da erstere Implementation die Diagonalisierung einer $8192 * 8192$-Matrix benötigen würde. Die Funktion der zweiten Implementationen von PCA und DMD wurde vorher zusätzlich am Haken-Zwanzig Modell getestet und rekonstruierte beide Male die gewünschten Grundmuster. Die Implementation der Umwandlung der Matrizen in Python findet sich in Anhang E.

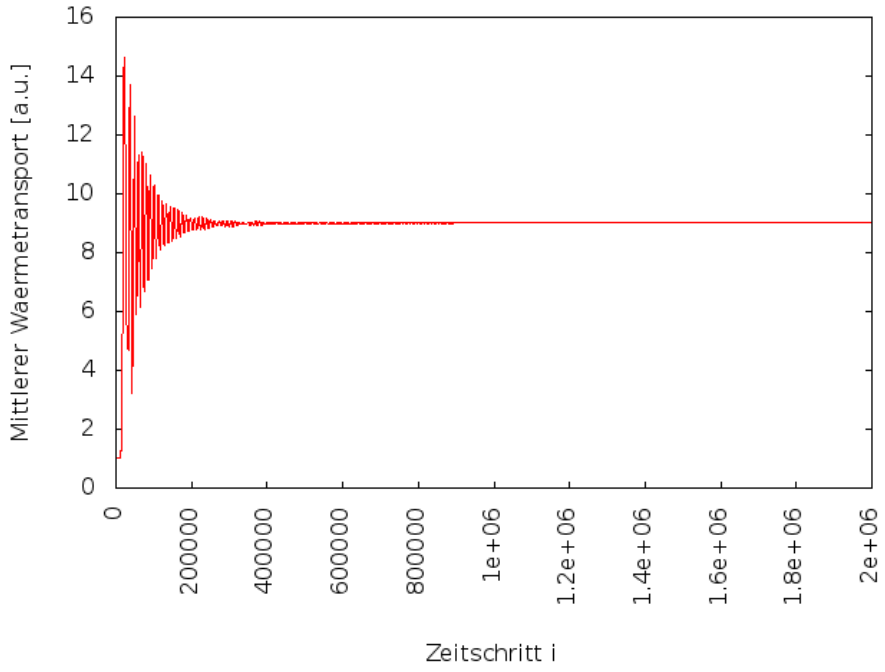


Abbildung 25: Mittlerer Wärmetransport der Konvektion in Abhängigkeit der Zeit.

5.4.1 PCA

Die Durchführung der Principal Component Analyse für die Simulation zeigt, dass der stationäre Zustand eine dominante Mode (siehe Abb. 31) hat. Diese Mode zeichnet sich dadurch aus, dass sie als einzige einen von null verschiedenen Eigenwert hat(siehe Abb. 26). Die PCA liefert somit genau den stationären Zustand wieder, welchen man auch in Abb. 24 erkennt und findet in den Zeitreihen keine weiteren relevanten Moden. Der PCA inhärent ist es aber, dass wir mit dieser Mode keine Aussage über die Dynamik des Systems machen können. Die Simulation zeigt zwar, dass diese Mode wirklich auftritt, jedoch oszilliert das simulierte System um diese Mode herum. Somit kann es lediglich im zeitlichen Mittel das System vernünftig beschreiben.

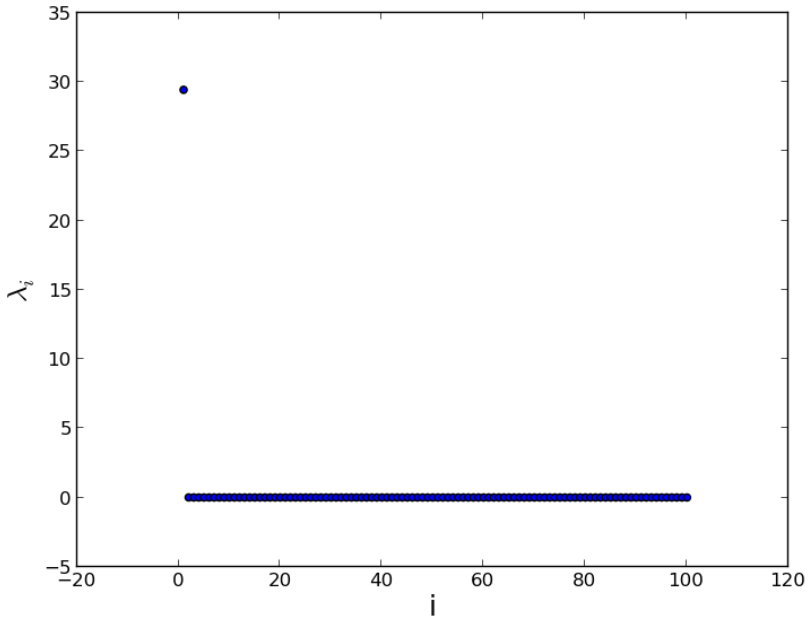


Abbildung 26: PCA-Eigenwerte für die Konvektionssimulation.

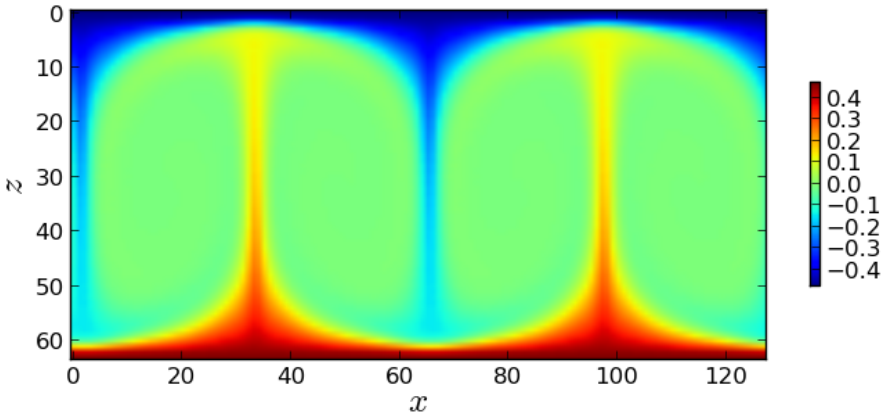


Abbildung 27: Mode des von null verschiedenen PCA-Eigenwertes.

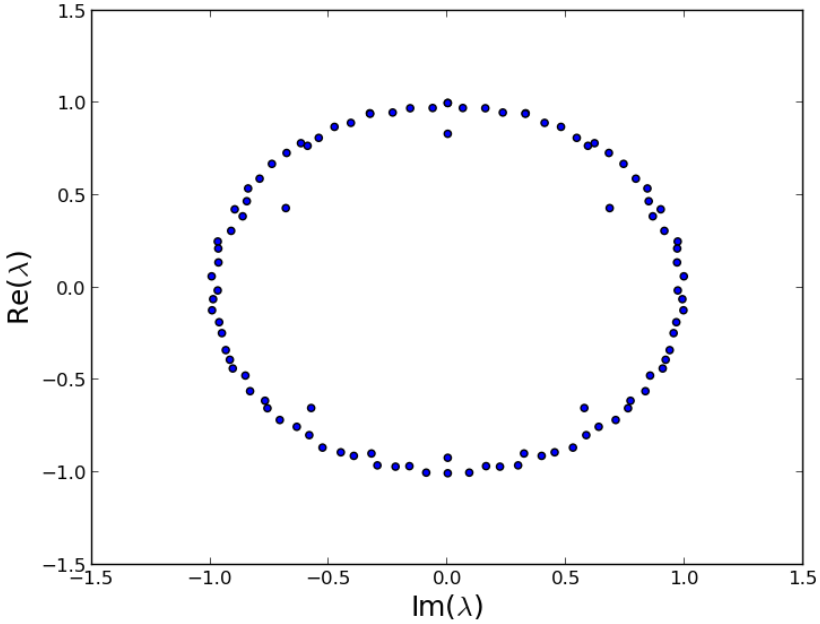
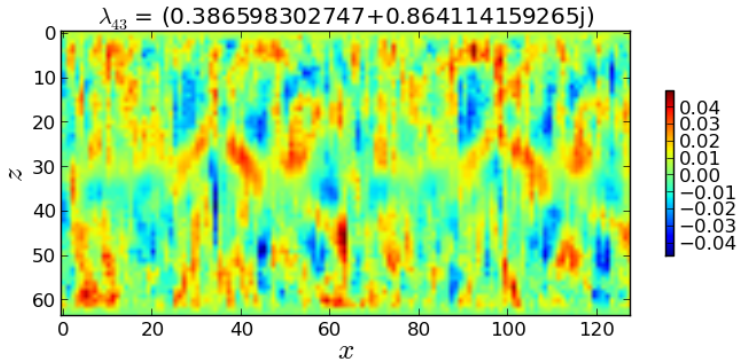


Abbildung 28: DMD-Eigenwerte für die Konvektionssimulation.

5.4.2 DMD

Bei der Analyse der Simulation mittels Dynamic Mode Decomposition erkennt man, dass es mehr als einen Eigenwert gibt, welcher ungleich null ist. Dieses erschwert die genaue Interpretation, welche Eigenvektoren für das Signal wichtig sind. Man kann jedoch überprüfen, in welchen Eigenvektoren sich Strukturen ausbilden, welche für die Dynamik wichtig sind und welche Eigenvektoren lediglich numerische Artefakte sind. Eigenvektoren, welche keine sichtbare Struktur haben und zudem noch Imaginäranteile besitzen, welche ungleich null sind, sind als numerisches Rauschen zu betrachten (siehe Abb.29).



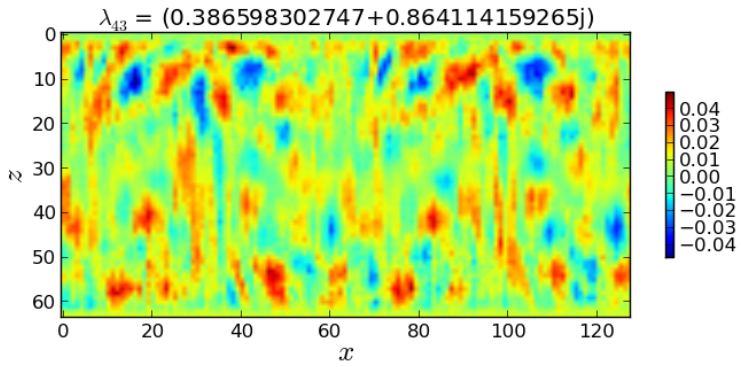


Abbildung 29: Real-(oben) und Imaginäranteil(unten) des DMD Eigenvektors mit Eigenwert $\lambda_{43} = 0.39 + 0.86i$

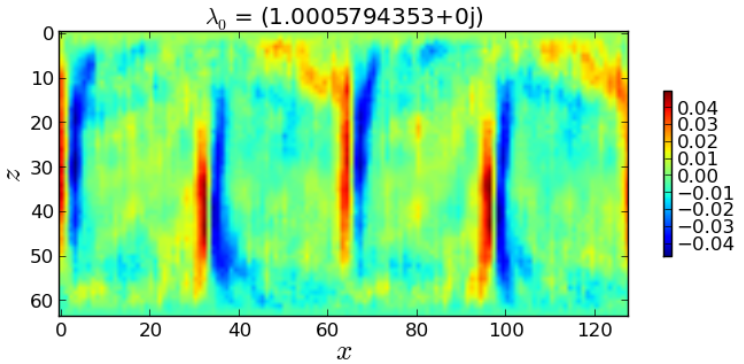


Abbildung 30: Realanteil der DMD-Mode mit $\lambda_0 = 1.00$.

Die DMD-Moden mit verschwindendem Imaginäranteil im Eigenwert und hohen Realanteil im Eigenwert sind als für die Dynamik wichtige Moden zu interpretieren. Die beiden Moden mit den höchsten rein reellen Eigenwerten gehören auch zu den Moden, welche gut erkennbare Strukturen besitzen und nicht als Rauschen gewertet werden sollten (siehe Abb. 30 und 31).

In dieser Analyse sind zwei weitere Moden mit komplexen Eigenwerten als für die Dynamik relevant einzustufen. Die eine Mode zeigt eindeutige Strukturen im Imaginäranteil, welcher über den komplexen Eigenwert zum Signal beitragen kann (Abb.32 und 32). Die andere Mode zeigt besitzt einen komplexen Eigenwert und zeigt Strukturbildung im Realanteil(siehe Abb.34).

Die DMD liefert somit mehr Moden, welche als für die Dynamik des Systems relevant angesehen werden müssen. Jedoch erweist sich die Interpretation der DMD-Ergebnisse schwieriger als die Interpretation der PCA-Mode. Im Gegensatz zur PCA können mit der DMD jedoch Aussagen über die Dynamik des Systems getätigt werden. Die DMD liefert nämlich komplexe Eigenwerte, deren Imaginäranteile für oszillatorisches Verhalten stehen und somit eine Dynamik widerspiegeln.

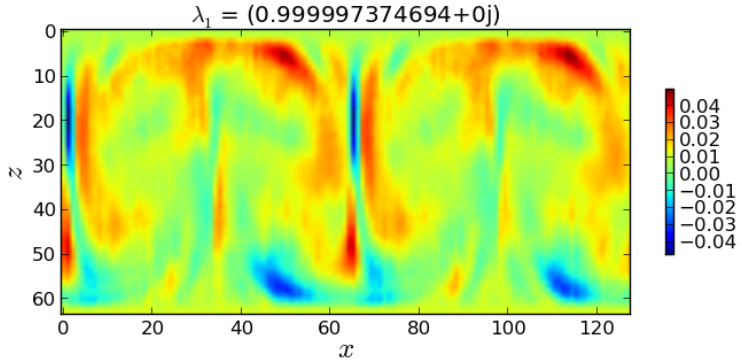


Abbildung 31: Realanteil der DMD-Mode mit $\lambda_1 = 1.00$.

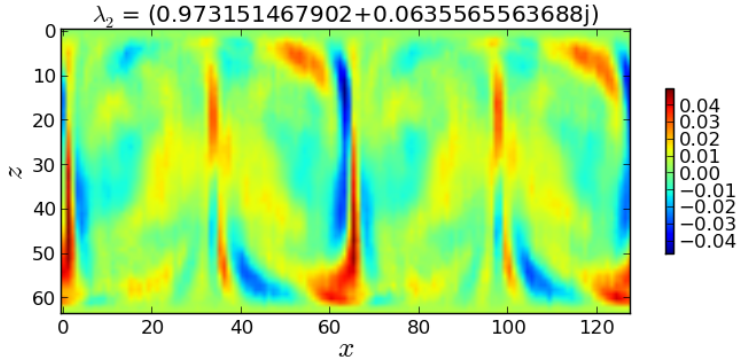


Abbildung 32: Imaginäranteil der DMD-Mode mit $\lambda_2 = 0.97 + 0.06i$.

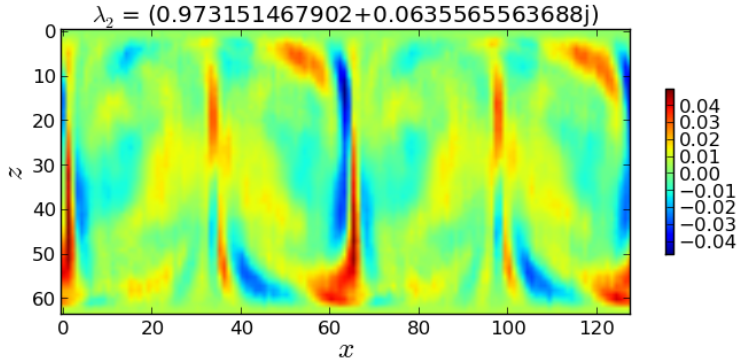


Abbildung 33: Realanteil der DMD-Mode mit $\lambda_3 = 0.97 + 0.06i$.

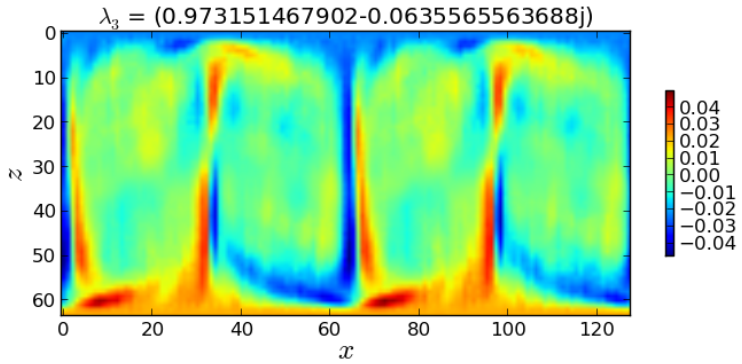


Abbildung 34: Realanteil der DMD-Mode mit $\lambda_1 = \lambda_3 = 0.97 - 0.06i$.

6 Zusammenfassung und Ausblick

Zunächst wurden in dieser Arbeit die Grundlagen für drei verschiedene Analyseverfahren gelegt. Namentlich sind diese Verfahren in der Forschung bekannt als die "Principal Component Analyse", die "Independent Component Analyse" und die "Dynamic Mode Decomposition". Diese Verfahren werden alle eingesetzt um komplexe Systeme zu analysieren und sind alle drei aus verschiedenen Ansätzen heraus motiviert. Zu den Grundlagen gehörten die theoretischen Motivationen der einzelnen Verfahren, sowie auch deren numerische Implementation. Für die Dynamic Mode Decomposition und die Principal Component Analyse wurde außerdem erklärt, wie man vorzugehen hat, falls große räumliche Dimensionen untersucht wurden und nur kleine zeitliche Dimensionen.
Danach wurde anhand von synthetisch erzeugten Zeitreihen gezeigt, welche Vor- und Nachteile die einzelnen Verfahren für die Anwendung auf zeitliche Signale besitzen. Unter anderem wurde gezeigt, dass die Independent Component Analyse, als Erweiterung der Principal Component Analyse und im Kontrast zu ihr, auch nicht-orthogonale Signale rekonstruieren kann. Außerdem wurde dazu mit Zeitreihen einer Kreisbewegung das Prinzip der Dimensionsreduktion für die Principal Component Analyse gezeigt.

Auf die Analyse von rein zeitlichen Signalen, schloss sich die Analyse raumzeitlicher Signale an. Hier wurde gezeigt, dass die Independent Component Analyse sich nicht bei der Anwendung auf raumzeitliche Signale für unsere Zwecke eignet. Die beiden anderen Analyseverfahren wurden dann auf zwei synthetische Beispiele angewendet. Zum einen wurden zwei orthogonale Grundmuster an die Orts- und Geschwindigkeitskomponente des klassischen harmonischen Oszillators gekoppelt um so zwei linear gekoppelte Moden zu untersuchen. Hier zeigte sich, dass beide Verfahren die Moden gut rekonstruieren. Der Vorteil der Dynamic Mode Decomposition lag darin, dass sie durch die komplexen Eigenwerte auch eine Aussage über die Oszillation tätigen kann.

Anschließend wurde mit dem Haken-Zwanzig-Modell ein nichtlinear gekoppeltes Differentialgleichungssystem untersucht, an welches auch wieder die zwei orthogonalen Grundmuster gekoppelt wurden. Hier konnte gezeigt werden, dass die Dynamic Mode Decomposition die Moden besser rekonstruiert als die Principal Component Analyse.

Zuletzt wurden die Grundlagen der Rayleigh-Bénard Konvektion erklärt, um danach eine zweidimensionale Simulation derselbigen mittels Principal Component Analyse und Dynamic Mode Decomposition zu analysieren. Hier wurde ein Teil der Simulation ausgewählt, welcher ein stationäres Verhalten zeigt. Wie schon bei dem Haken-Zwanzig Modell lieferten die beiden Verfahren unterschiedliche Ergebnisse. Ein Grund dafür ist mit Sicherheit, dass die Dynamic Mode Decomposition wirklich von der Dynamik des Systems ausgeht und nicht wie die Principal Component Analyse die Varianz betrachtet. Als Schwierigkeit der Dynamic Mode Decomposition stellte sich hier die Interpretation der Ergebnisse heraus. Da man im Gegensatz zur Principal Component Analyse keine verschwindenden Eigenwerte hat, ist es schwerer zu entscheiden, welche Moden nicht zum Signal beitragen. Der große Unterschied dieser Verfahren ist, dass die PCA mit ihrer gefundenen Mode das System zwar im Mittel gut beschreiben

kann, jedoch keine Informationen über die Dynamik liefert.

Um die Funktion der DMD an diesem System weiter zu testen, wäre es interessant zu versuchen mittels der ermittelten Moden die Dynamik zu rekonstruieren. Hier können die komplexen Eigenwerte Informationen über die Oszillationen der DMD-Moden geben. Eine Kopplung dieser Moden an zugehörige Oszillationen könnte mit der eigentlichen Simulation verglichen werden, um herauszufinden mit welchen Moden man die gesamte Dynamik des Systems am besten darstellt.

Für weitere Untersuchungen wäre es interessant die Änderung der ermittelten Moden in Abhängigkeit der Rayleigh-Zahl zu betrachten. Des Weiteren könnten die Analyse-Verfahren auf dreidimensionale Simulationen von Konvektionsprozessen angewendet werden.
Eine weitere Analyse der Vor- und Nachteile der einzelnen Verfahren wäre eine Möglichkeit, um besser einschätzen zu können, in welchen Bereichen welche Analyse Stärken gegenüber den anderen Verfahren besitzt. Zum Beispiel könnte man testen, wie gut die Dynamic Mode Decomposition nicht-orthogonale Moden rekonstruieren kann.

Abbildungsverzeichnis

1 Zeitreihen der Kreisbewegung 10
2 3D Plot der Kreisbewegung 11
3 3D Plot der Kreisbewegung 11
4 Eigenwerte von $\vec{q}(t)$ 12
5 Zeitreihen der Kreisbewegung in den Moden v_i 12
6 Phasenraum des Haken-Zwanzig-Modells für die angegebenen Parameter 14
7 Zeitreihen von $u(t)$ und $s(t)$ 14
8 PCA-Eigenwerte des Haken-Zwanzig-Modells 15
9 Phasenraum des Haken-Zwanzig-Modells mit den PCA-Eigenvektoren (gestrichelt) 15
10 Grundsignale, gemischte Signale, PCA und ICA Rekonstruktion . 16
11 DMD-Eigenwerte des Haken-Zwanzig-Modells 17
12 Phasenraum des Haken-Zwanzig-Modells mit den DMD-Eigenvektoren (gestrichelt) 17
13 Muster 1(links) und 2(rechts) 18
14 Zeitreihen x(t) und v(t) des harmonischen Oszillators 19
15 PCA-Eigenwertspektrum von $\vec{q}(t)$ 20
16 DMD-Eigenwertspektrum von $\vec{q}(t)$ 20
17 PCA-Eigenvektor $\vec{v}_1$(links) und $\vec{v}_2$(rechts) des harm. Oszillators. 21
18 Real-(links) und Imaginäranteil (rechts des DMD-Eigenvektors $\vec{v}$ des harm. Oszillators. 21
19 Eigenwerte der PCA für das Haken-Zwanzig-Modell 22
20 PCA-Eigenvektoren v_1 (links) und v_2 (rechts) des Haken-Zwanzig-Modells. 23
21 Eigenwerte der DMD für das Haken-Zwanzig-Modell 23
22 DMD-Eigenvektoren v_1 (links) und v_2 (rechts) des Haken-Zwanzig-Modells. 24
23 Snapshot des Temperaturfeldes bei Zeitschritt 22000. 28
24 Snapshot des Temperaturfeldes bei Zeitschritt 160000. 28
25 Mittlerer Wärmetransport der Konvektion in Abhängigkeit der Zeit. 29
26 PCA-Eigenwerte für die Konvektionssimulation. 30
27 Mode des von null verschiedenen PCA-Eigenwertes. 30
28 DMD-Eigenwerte für die Konvektionssimulation. 31
29 Real-(oben) und Imaginäranteil(unten) des DMD Eigenvektors mit Eigenwert $\lambda_{43} = 0.39 + 0.86i$ 32
30 Realanteil der DMD-Mode mit $\lambda_0 = 1.00$. 33
31 Realanteil der DMD-Mode mit $\lambda_1 = 1.00$. 34
32 Imaginäranteil der DMD-Mode mit $\lambda_2 = 0.97 + 0.06i$. 34
33 Realanteil der DMD-Mode mit $\lambda_3 = 0.97 + 0.06i$. 35
34 Realanteil der DMD-Mode mit $\lambda_1 = \lambda_3 = 0.97 - 0.06i$. 35

Literatur

[BHL93] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Rev. Fluid Mech*, pages 539–575, 1993.

[Bou03] J. Boussinesq. Théorie analytique de la chaleur. *volume 2. Gauthier-Villars,Paris*, 1903.

[BSMM05] I. N. Bronstein, K. A. Semendjajew, G. Musiol, and H. Mühlig. *Taschenbuch der Mathematik*. Verlag Harri Deutsch, Frankfurt am Main, 6. edition, 2005.

[CTR12] Kevin K. Chen, Jonathan H. Tu, and Clarence W. Rowley. Variants of dynamic mode decomposition: Boundary condition, koopman, and fourier analyses. *Journal of Nonlinear Science*, 22(6):887–915, 2012.

[CWRH] S. Bagheri P. Schlatter C. W. Rowley, I. Mezic and D.S. Henningson. Spectral analysis of nonlinear flows. *J. Fluid Mech.*

[Daf04] C.J.C Meijer O.G. Beek P.J. Daffertshofer, A Lamoth. Pca in studying coordination and variability: a tutorial. *Clinical Biomechanics*, 2004.

[Hak83] H. Haken. *Synergetics. an introduction.* 1983.

[HO00] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *Neural Netw.*, 13(4-5):411–430, May 2000.

[HUF99] A. Hutt, C. Uhl, and R. Friedrich. Analysis of spatiotemporal signals: A method based on perturbation theory. *Phys. Rev. E*, 60:1350–1358, Aug 1999.

[Hut] Methoden zur untersuchung der dynamik raumzeitlicher signale, author = Hutt, A., type = dissertation year = 2001 address = universität stuttgart.

[Hyv98] Aapo Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*, NIPS '97, pages 273–279, Cambridge, MA, USA, 1998. MIT Press.

[JH91] C. Jutten and P. Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24(1):1–10, July 1991.

[Kar46] K. Karhunen. Zur Spektraltheorie stochastischer Prozesse. *Annales Academiae Scientiarum Fennicae*, 37, 1946.

[Loe65] M. Loeve. *Fonctions aléatoires du second ordre...* 1965.

[Lue] Statistische eigenschaften turbulenter rayleigh–bénard-konvektion, author = Lülff J., type = diplomarbeit year = 2011 address = universität münster.

[PV94] G. Plaut and R. Vautard. Spells of Low-Frequency Oscillations and Weather Regimes in the Northern Hemisphere. *Journal of the Atmospheric Sciences*, 51(2):210–236, 1994.

[Sch10] P.J. Schmid. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech*, 2010.

[SMH05] TROYR. SMITH, JEFF MOEHLIS, and PHILIP HOLMES. Low-dimensional modelling of turbulence using the proper orthogonal decomposition: A tutorial. *Nonlinear Dynamics*, 41(1-3):275–307, 2005.

A PCA in Python

Die Funktion PCA benötigt als Argument eine Matrix A, welche aus den in Abschnitt 2.1 beschriebenen Signalvektoren besteht. Sie liefert dann in der Variable latent die PCA-Eigenwerte und in coeff die zugehörigen Eigenvektoren zurück. Je nach Variante ist die Matrix A so zu wählen, dass bei 'np.mean(A.T,axis=1)' entweder zeitlich oder aber räumlich gemittelt wird.

```
import numpy as np

def PCA(A):
        M = (A-np.mean(A.T,axis=1)).T
        latent,coeff = np.linalg.eig(np.cov(M))
        return latent, coeff
```

B DMD in Python

Die DMD ist wie folgt in Python implementiert. Hierbei enthält die Matrix M die Signalvektoren, sodass X und Y die in Abschnitt 2.4 beschriebenen Matrizen sind. Die DMD-Eigenvektoren und DMD-Eigenwerte befinden sich zum Schluss in eigenvectors bzw. eigenvalues.

```
import numpy as np

Y = np.delete(M,0,1)
X = np.delete(M,len(M.T)-1,1)

U,S,V = np.linalg.svd(X,full_matrices=False)
S = np.diag(S)
U_ = U.conj()
V = V.conj().T
S_ = np.linalg.inv(S)

A = np.dot(U_,np.dot(Y,np.dot(V,S_)))

eigenvalues, eigenvectors = np.linalg.eig(A)

for i in range(len(eigenvectors)):
        eigenvectors[:,i]=np.dot(U,eigenvectors[:,i])
```

Die zweite Variante sieht wie folgt aus:

```
import numpy as np

Y1 = np.delete(M,0,1)
X1= np.delete(M,len(M.T)-1,1)

F,S,V = np.linalg.svd(X1,full_matrices=False)
X_X=np.dot(X1.conj().T,X1)

lambdas,A=np.linalg.eig(X_X)
lambdas_=np.sqrt(np.diag(lambdas))
lambdas=np.linalg.inv(lambdas_)
U=np.dot(X1,np.dot(A,lambdas))
```

```
U_ = U.conj()
B = np.dot(U_.T,np.dot(Y1,np.dot(A,lambdas)))

eigenvalues, eigenvectors = np.linalg.eig(B)
```

C Fast-ICA in Python

Dieses Beispiel einer ICA-Implementation ist geeignet für einen Vektor X, welcher zwei verschiedene Signale besitzt. Diese zwei Signale bestehen aus zwei überlagerten Grundsignalen, welche am Ende in z ausgegeben werden.

```
import numpy as np

def PCA(A):
        M = (A-np.mean(A.T,axis=1)).T
        latent,coeff = np.linalg.eig(np.cov(M))
        return latent,coeff

def g(x):
        g=np.tanh(x)
        return g

def g_(x):
        g=(4*np.cosh(x)**2)/(np.cosh(2*x)+1)**2
        return g

latent, coeff = PCA(X)

A = np.dot(coeff,X.T)


D = np.diag(latent)
D = np.sqrt(D)
X = np.dot(D,A)


M=np.zeros((2,2))

for j in range(2):
        w = np.random.rand(2)
        w/=np.linalg.norm(w)
        w_old=w
        diff=1

        while(diff>0.0001):
                w = (X*g(np.dot(w_old.T,X))).mean(axis=1)
                w = w-g_(np.dot(w_old.T,X)).mean()*w
                w-=np.dot(np.dot(w,M[:j].T),M[:j])
                w/=np.linalg.norm(w)
                diff=np.abs(np.abs(np.dot(w,w_old))-1)
                w_old=w
        M[j:]=w
```

```
z=np.dot(M,X)
```

D Erzeugung der Grundmuster und Umwandlung der Muster

Die Funktion matrixtovector benötigt eine Matrix m als Argument und schreibt diese in einen Vektor vec, welcher daraufhin zurückgegeben wird. Die Funktion vectortomatrix wird mit einem Vektor der Dimension 100 als Argument aufgerufen und erzeugt daraus eine Matrix m der Dimension $10 \cdot 10$. Die Funktion createpatterns erzeugt die zwei Grundmuster aus Abschnitt 4 und gibt diese zurück.

```
import numpy as np

def matrixtovector(m):
        vec = np.zeros(len(m)*len(m))
        i = 0
        for j in range(len(m)):
                for k in range(len(m)):
                        vec[i]=m[j,k]
                        i=i+1
        return vec

def vectortomatrix(vec):
        m = np.zeros([10,10])
        i = 0
        for j in range(10):
                for k in range(10):
                        m[j,k]=vec[i]
                        i=i+1
        return m


def createpatterns():
        m1 = np.eye(10)
        m1[0:5,0:1]=1
        m1[0:1,0:5]=1
        m2 = np.zeros((10,10))

        for i in range(10):
                m2[9-i,i] = 1

        m2[0:5,9]=1
        m2[0:1,5:9]=1
        return m1,m2
```

E Vektor-Matrix-Umwandlung für die Konvektionssimulation

Die Funktion matrixtovector(M) erhält eine $256 \cdot 64$-Matrix M als Argument und liefert einen Vektor v der Größe $128 \cdot 64$ zurück. Hier wurde in den Zeilen jeweils nur jeder zweite Punkt in den Vektor geschrieben.
Die Funktion vectortomatrix(vec) erhält einen Vektor vec als Argument und liefert die zugehörige Matrix m zurück, welche die Dimension $128 \cdot 64$ besitzt.

```
def matrixtovector(M):
        v = np.zeros(M.size/2)
        dim = M.shape
        k=0
        for i in range(0,dim[0],2):
                for j in range(dim[1]):
                        v[k]=M[i,j]
                        k=k+1
        return v

def vectortomatrix(vec):
        m = np.zeros([128,64])
        i = 0
        for j in range(128):
                for k in range(64):
                        m[j,k]=vec[i]
                        i=i+1
        return m
```

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit mit dem Titel

"Analyse raumzeitlicher Signale am Beispiel der Rayleigh-Bénard Konvektion"

selbständig verfasst habe, und dass ich keine anderen Quellen und Hilfsmittel als die angegebenen benutzt habe und dass die Stellen der Arbeit, die anderen Werken – auch elektronischen Medien – dem Wortlaut oder Sinn nach entnommen wurden, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht worden sind.

(Datum, Unterschrift)