Analyse schaltender und driftender Dynamik mit neuronalen Netzen

vorgelegt von
Dipl.-Inform. Jens Kohlmorgen
aus Hamburg

Vom Fachbereich 13 – Informatik –
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor-Ingenieur
– Dr.-Ing. –

genehmigte Dissertation

Promotionsausschuß:

Vorsitzender: Prof. Dr. K. Obermayer
Berichter: Prof. Dr. S. Jähnichen
Berichter: Prof. Dr. K. Pawelzik

Tag der wissenschaftlichen Aussprache: 9. Juli 1998

Berlin 1998

D 83

Vorwort

Kennzeichnend für die vorliegende Arbeit, kennzeichnend aber auch für Herrn Kohlmorgen ist das Bestreben solide theoretische Resultate zur Verbesserung technischer Systeme konkret zur Anwendung zu bringen. Im Falle dieser Arbeit liegen die theoretischen Ergebnisse auf dem Gebiet des maschinellen Lernens. Herr Kohlmorgen entwickelt ein Lernverfahren zur Segmentierung und Identifizierung schaltender und driftender Dynamiken und beweist dabei hervorragende Kenntnisse in einem breiten Fächerspektrum, welches von der Neuroinformatik über die theoretische Physik chaotischer dynamischer Systeme bis zur Analyse physiologischer Signale reicht. Und aus diesem reichen Kenntnisstand heraus entwickelt er die konkreten und praktisch hochrelevanten Anwendungen seiner Resultate. Beeindruckend sind seine Ergebnisse in den Analysen von kontinuierlichen Sprachsignalen und biologischen Systemen. Höhepunkt aber sind die Analysen von EEG-Daten aus einem konkreten Schlaflabor an der Freien Universität Berlin. Die Zeitauflösung wurde um ca. zwei Größenordnungen verbessert und seine Driftextraktion zwischen Schlafstadien gab Anlaß zu einer neuen Interpretation dieser Drift als Aufmerksamkeitsparameter.

Die Arbeit ist in vielerlei Hinsicht bemerkenswert und betritt in vielen Punkten wissenschaftliches Neuland. Ich hoffe, daß es vielen Lesern dieser Arbeit ähnlich viel Freude bereiten wird, dieses Neuland zu betreten und zu verstehen.

Berlin, im November 1998 Prof. Dr. Stefan Jähnichen

Vorwort des Autors

Bei der Untersuchung von Zeitreihen dynamischer Systeme wird meist vorausgesetzt, daß die zugrundeliegende Dynamik stationär ist, und zwar in dem Sinne, daß der funktionale Zusammenhang zwischen den Daten über die Zeit konstant bleibt. In vielen natürlichen Systemen läßt sich jedoch ein nichtstationäres Verhalten beobachten. Dies näher zu untersuchen gab die Motivation für diese Arbeit. Die Idee war es, ein Verfahren zu entwerfen, mit dem Zeitreihen schaltender oder driftender Systeme analysiert werden können.

Das daraufhin entwickelte Verfahren wurde dann zunächst an künstlich erzeugten Zeitreihen getestet und schließlich auch an gemessenen Zeitreihen natürlicher Systeme, Sprache und Hirnaktivität, erprobt. Die erzielten Resultate ermutigten zu einer Weiterentwicklung des Verfahrens speziell im Hinblick auf eine Anwendung zur Analyse physiologischer Daten des Wach-Schlaf-Übergangs.

All denen, die mich bei der Erarbeitung dieser Dissertation unterstützt haben, möchte ich an dieser Stelle meinen herzlichen Dank ausdrücken. Insbesondere möchte ich mich bei Prof. Dr. Stefan Jähnichen bedanken, der mir ermöglichte, an einem gleichermaßen aktuellen wie interessanten Thema zu arbeiten. Mein ganz besonderer Dank gilt auch Dr. Klaus-Robert Müller und Prof. Dr. Klaus Pawelzik für die fruchtbare Zusammenarbeit seit nunmehr vielen Jahren. Ich hoffe auch in Zukunft auf eine gute Zusammenarbeit und viele anregende Diskussionen mit ihnen. Nicht zuletzt geht mein Dank auch an Dr. Jörn Rittweger für die gelungene Zusammenarbeit bei der Untersuchung der Wach-Schlaf-Daten.

Berlin, im März 1998 Jens Kohlmorgen

Kurzfassung

In dieser Arbeit wird ein Verfahren zur unüberwachten Segmentierung und Identifizierung von Zeitreihen vorgestellt. Es eignet sich zur Analyse von Systemen, denen eine nichtstationäre, schaltende oder driftende Dynamik zugrunde liegt; ein Phänomen, das in vielen Bereichen der Natur beobachtet werden kann. Als Beispiele hierfür seien die Sprache oder physiologische Systeme genannt – zwei Bereiche, die in dieser Arbeit exemplarisch untersucht werden. Aber auch andere komplexe Systeme, wie etwa die Entwicklung der Finanzmärkte, lassen das Vorhandensein einer solchen Dynamik vermuten.

Der Ansatz basiert auf *hard* oder *soft competitive learning* und einem Wettbewerbsvorteil auf zeitlich benachbarten Daten. Im Wettbewerb stehen dabei Prädiktoren (neuronale Netze), die lernen, die Daten einer Zeitreihe vorherzusagen. Jeder einzelne Prädiktor kann dabei nur eine stationäre Dynamik vorhersagen. Die Prädiktoren spezialisieren sich daher während des Lernvorgangs auf solche Abschnitte der Zeitreihe, in denen die Dynamik weitgehend stationär ist. Dies wird durch die Verwendung tiefpaßgefilterter Vorhersagefehler erreicht, die sich aus der Annahme einer selten schaltenden Dynamik herleitet. Eine sich kontinuierlich verändernde Dynamik kann anschließend mit dem in dieser Arbeit ebenfalls vorgeschlagenen Drift-Algorithmus modelliert werden. Die Anwendung auf physiologische Wach/Schlaf-Daten (EEG, EOG, Atmung) zeigt, daß das Verfahren die Veränderungen in der Dynamik sehr fein auflöst und selbstständig eine Einteilung in Wach- und Schlafphasen liefert, die gut mit der manuellen Segmentation eines Mediziners übereinstimmt.

Schlagwörter: Zeitreihen, Segmentation, Identifikation, Vorhersage, neuronale Netze, nichtstationäre Systeme, nichtlineare Dynamik, schaltende Dynamik, driftende Dynamik.

Abstract

In this work, a method for unsupervised segmentation and identification of time series is presented. It can be used to analyze dynamical systems with non-stationary switching or drifting behavior – a phenomenon observable in many natural, real-world domains. As examples we examine the dynamics of speech and physiological systems, but also many other complex systems, e.g. the financial markets, are expected to show such type of behavior.

The ansatz is based on *hard* or *soft competitive learning* and a competitive advantage for temporally adjacent data points in a time series. The competition is performed by a set of (neural network) predictors that compete for the prediction of the data points. Each predictor is capable of predicting only *stationary* dynamics. Therefore, the predictors specialize during a training phase on those segments of the data where the dynamics is stationary to a large extent. This is achieved by means of a low-pass filter on the prediction errors, which can be motivated by the assumption of a low switching rate between different dynamical modes. A continuously alternating dynamics can then be modeled by a drift segmentation algorithm, which is also presented in this work. An application to physiological wake/sleep data (EEG, EOG, respiration) shows that the proposed method detects dynamical changes with a high resolution. Moreover, it autonomously yields a segmentation into wake and sleep stages which is in good agreement with the manual segmentation of a medical expert.

Keywords: time series, segmentation, identification, prediction, neural networks, non-stationary systems, non-linear dynamics, switching dynamics, drifting dynamics.

Inhaltsverzeichnis

1 Einleitung **11**

2 Zeitreihenvorhersage mit neuronalen Netzen **17**

2.1 Vorhersage von Zeitreihen 17

2.2 Neuronale Netze mit radialen Basisfunktionen 19

2.2.1 Architektur des Moody-Darken-Netzes 20

2.2.2 Lernen im Moody-Darken-Netz 22

3 Konkurrierende Prädiktoren **29**

3.1 Identifizierung und Segmentierung 30

3.1.1 Die Problemstellung 31

3.1.2 Competing Experts 32

3.1.3 Einbindung der Moody-Darken Netze 35

3.1.4 Erweiterung des Ansatzes 38

3.2 Beispiele 40

3.2.1 Zwei chaotische Abbildungen 41

3.2.2 Die Mackey-Glass Differentialgleichung 47

3.2.3 Sprachdaten 50

3.2.4 Physiologische Daten 58

4 Theoretische Fundierung der Methode 63

4.1 Maximum Likelihood 66

4.2 Der Tiefpaß-Filter 67

4.3 Spezialfall: Hard Competition 70

5 Annealed Competition of Experts (ACE) 71

5.1 Soft Competition 72

5.2 Beispiel: Mackey-Glass 77

5.3 Beispiel: Data Set D 78

6 Erkennung linearer Transienten 83

6.1 Segmentierung mit linearer Drift 83

6.2 Beispiel: Sprachdaten 88

7 Erkennung nichtlinearer Transienten 91

7.1 Hidden Markov Modell für variable Drift 91

7.2 Der Drift-Segmentationsalgorithmus 94

7.3 Driftendes Chaos 96

7.4 Driftende Mackey-Glass Dynamik 99

8 Analyse von Wach/Schlaf-Daten **101**

8.1 Schaltende Dynamik 103

8.1.1 Segmentation von EOG und Atmung 103

8.1.2 EEG Segmentation 106

8.2 Prädiktion 108

8.3 Driftende Dynamik 109

8.3.1 Atmungsdaten 109

8.3.2 EEG 110

8.3.3 Interpretation der Drift-Segmentation 111

9 Zusammenfassung und Diskussion **119**

Literatur **125**

Kapitel 1

Einleitung

Der Wunsch, die Vorgänge in der Natur zu verstehen, beflügelte von je her die Naturwissenschaften nach Gesetzmäßigkeiten zu suchen, die das Verhalten von beobachteten Phänomenen erklären. So kann man beispielsweise für ein einfaches Pendel, das sich um eine feste Achse dreht, eine Gleichung aufstellen, die die Dynamik des Systems vollständig beschreibt. Wenn die Anfangsbedingungen bekannt sind, läßt sich die Dynamik vorhersagen und das vermutete Gesetz durch ein Experiment bestätigen. Bei vielen komplexen dynamischen Systemen hat sich jedoch gezeigt, daß eine solche Modellbildung äußerst schwierig ist, da sehr viele Einflußgrößen zu berücksichtigen sind, von denen man nicht immer weiß, wie sie auf das System einwirken. Beispiele hierfür finden sich in den unterschiedlichsten Bereichen: der Meteorologie, der Strömungstheorie, der Gehirn- und Herzforschung oder der Ökonomie.

Man ist daher bei komplexen dynamischen Systemen daran interessiert, zeitabhängige Meßgrößen (Zeitreihen) des Systems zu untersuchen, um auf diese Weise Einsicht in bislang unbekannte Mechanismen zu erhalten oder die zukünftige Entwicklung des Systems vorherzusagen. Gerade die Vorhersage von Zeitreihen spielt in vielen Bereichen eine wichtige Rolle. Je nach Anwendungsgebiet kann sie als Planungs- und Steuerungshilfe, als Indikator von Fehlentwicklungen, zur Früherkennung und Vorwarnung, aber auch zur Spekulation herangezogen werden.

Klassische Methoden zur Analyse von Zeitreihen verwenden meist lineare Modelle, wie zum Beispiel ARMA-Modelle, Kalman Filter usw. (Box *et al.* 1994 [4], Priestley 1988 [56], Honerkamp 1990 [20]). Ein wichtiger Schritt, der über

lineare Vorhersagemodelle hinausging, war das TAR Modell (Tong *et al.* 1980 [65]), in dem abhängig vom Systemzustand zwischen zwei linearen ARMA Modellen gewählt wird. Zwei weitere Entwicklungen ebneten ebenfalls den Weg zu nichtlinearen Modellen:

1. Im Zusammenhang mit Untersuchungen deterministisch chaotischer Systeme führte man Verfahren zur Rekonstruktion des Phasenraumes ein, wobei hier die Methode der time-delay Einbettung hervorzuheben ist. Diese erlaubt es, die geometrische Struktur der unterliegenden Dynamik aus Zeitreihen zu rekonstruieren (Packard *et al.* 1980 [50], Takens 1981 [64]). Dadurch wurde eine wichtige Vorraussetzung für die phänomenologische Modellbildung deterministisch chaotischer Systeme geschaffen.

2. Auf dem Gebiet neuronaler Netze wurden leistungsfähige Lernalgorithmen gefunden, mit denen der Raum potentieller, nichtlinearer Modelle durchsucht werden kann. Neuronale Netze sind universelle Funktionsapproximatoren. Es kann rigoros bewiesen werden, daß mit einer ausreichenden Komplexität des Netzes eine Funktion einer bestimmten Funktionenklasse mit einer vorgegebenen Präzision angenähert werden kann (White *et al.* 1992 in [72], Barron 1993 [1], Murata 1993 [47]). Mit neuronalen Netzen wurden erfolgreich geeignete Prädiktormodelle für Zeitreihen gefunden (Farmer *et al.* 1987 [14], Lapedes *et al.* 1987 [36], Weigend *et al.* 1994 [69]).

Vorhersage war bei diesen Entwicklungen oft weniger ein Selbstzweck als vielmehr eine Methode die Güte der gefundenen Modelle zu validieren, die neben ihrer Bedeutung für das Verständnis der zugrundeliegenden Dynamik auch für Anwendungen wie Kontrolle und Klassifikation Verwendung fanden.

Ein eindrucksvolles Zeugnis der Entwicklung auf diesem Gebiet stellt die *Santa Fe Time Series Prediction and Analysis Competition* dar (Weigend *et al.* 1994 [69]). In diesem Wettbewerb wurden Teile verschiedener Zeitreihen zur Verfügung gestellt, und die teilnehmenden Wissenschaftler aus verschiedenen Disziplinen waren aufgerufen, die ihnen unbekannten folgenden n Zeitschritte der jeweiligen Zeitreihen vorhersagen. Es zeigte sich, daß neuronale Netze für alle gegebenen Zeitreihen die besten Vorhersagen und somit die besten Modelle lieferten.

Bei der Modellierung von Zeitreihen ging man bisher weitgehend davon aus, daß die zugrundeliegenden Prozesse (die dynamischen Systeme) stationär sind, in dem

Sinne, daß die Signale von nur einer Quelle mit einem zeitlich konstanten Satz von Parametern stammen. Diese Annahme ist jedoch häufig nicht korrekt. Solchermaßen nichtstationäre Zeitreihen kommen sowohl in der industriellen Anwendung als auch in physikalischen (Kaneko 1989 [23]) oder biologischen Systemen (Pawelzik 1994 [52]) häufig vor.

Ein Beispiel sind multistabile dynamische Systeme. Steuerungssignale oder auch zufällige Störungen können solche Systeme von einem Attraktor in einen anderen „schalten“, wobei diese jeweils für sich genommen durch eine relativ einfache Dynamik beschreibbar sind, während das Gesamtsystem inklusive der Störungen ein wesentlich komplexeres Modell erfordert. Weiterhin kommen schwer vermeidbare Drifts in der Dynamik experimenteller Systeme häufig vor. Diese sind beispielsweise verursacht durch schlecht kontrollierbare Schwankungen von Parametern und sind daher ein großes Hindernis bei der Datenanalyse. Es gibt erst wenige Ansätze der Zeitreihenanalyse, die mit Nichtstationaritäten dieses Typs umzugehen vermögen. Während es konzeptuell unproblematisch ist, Zeitreihen niedrigdimensionaler stationärer dynamischer Systeme vorherzusagen, versagen herkömmliche Modelle, wenn die Zeitreihe von unterschiedlichen Dynamiken stammt, zum Beispiel wenn das zugrundeliegende System seine Dynamik in der Zeit ändert.

Bei der Zeitreihenanalyse ist es also wünschenswert zu wissen, ob man es mit einer oder mehreren Signalquellen zu tun hat. Um dies zu überprüfen, werden u.a. statistische Tests eingesetzt (Box *et al.* 1994 [4]), die jedoch bisher nicht zum Standardwerkzeug der Datenanalyse mit neuronalen Netzen gehören. Vielmehr gehen sowohl klassische Ansätze als auch Methoden mit neuronalen Netzen in der Regel davon aus, daß das Signal von nur einer Quelle erzeugt wird, die ihre Dynamik mit der Zeit nicht ändert, d.h. daß das System weder schaltet, noch daß die Dynamik driftet.

Zu den wenigen Beispielen von neuronalen Ansätzen, die Nichtstationaritäten explizit berücksichtigen, gehören neben dem von uns entwickelten *Annealed Competition of Experts* (ACE) Algorithmus (Kohlmorgen *et al.* 1994 [28], Kohlmorgen *et al.* 1995 [29], Müller *et al.* 1995 [45], Pawelzik *et al.* 1996 [53]) die Arbeiten von Cacciatore *et al.* 1994 [6] und Weigend *et al.* 1995 [70].

Der ACE Algorithmus ist ein robuster, unüberwachter Lernalgorithmus zur Modellierung und Analyse von schaltenden dynamischen Systemen. Er wurde erfolgreich sowohl auf schaltende Zeitreihen mit chaotischer Dynamik, als auch auf Sprachsignale, physiologische Daten und auf Daten aus der Hirnforschung

angewandt (Kohlmorgen *et al.* 1994 [28], Kohlmorgen *et al.* 1995 [29], Müller *et al.* 1995 [45], Pawelzik *et al.* 1996 [53]). Der ACE Algorithmus berücksichtigt das Schalten der Dynamik durch eine explizite statistische Modellierung. Drei Grundprinzipien werden hierbei angewandt: (a) der Einsatz eines Ensembles von Prädiktoren, d.h. eines Ensembles von neuronalen Netzen, die die Zeitreihe vorhersagen, (b) die Verwendung tiefpaßgefilterter Fehler und (c) die sukzessive Verstärkung des Wettbewerbslernens. Wir gehen also nicht von *einem* neuronalen Netz aus, das die Zeitreihe vorhersagt oder analysiert, sondern ein *Ensemble* von Netzwerken tritt in Konkurrenz um den Datensatz. Der Wettbewerb wird adiabatisch durch einen Temperaturparameter gesteuert: zu Beginn des Lernens sind alle Prädiktoren am Lernen beteiligt, eine kollektive Approximation der Zeitreihe wird gefunden. Je weiter der Temperaturparameter gesenkt wird, desto stärker wird die Konkurrenz und Diversifikation der Einzelnetze. Nach Konvergenz des Algorithmus bleiben nur noch diejenigen Prädiktoren übrig, die sich korrekt auf *eine* der Signalquellen spezialisiert haben. Die Verlierernetze dürfen im Verlauf des Algorithmus nicht weiterlernen und fallen aus dem Wettbewerb heraus. Dabei wird die statistische Abhängigkeit aufeinanderfolgender Zeitschritte der Zeitreihe in unserer Modellierung durch einen *Tiefpaß* berücksichtigt. Weiterhin zeigte es sich, daß eine *adiabatische* Verstärkung des Wettbewerbs für eine Vermeidung lokaler Minima wichtig sein kann (Yuille *et al.* 1994 [73]). Außerdem wird durch geeignete Wahl des Temperaturparameters der Grad des Details bestimmt, mit dem Einzel-Dynamiken unterschieden werden sollen (Rose *et al.* 1990 [59]). Der ACE Algorithmus ist ein unüberwachter Analyse-Algorithmus, d.h. es wird keine Vorkenntnis über die Anzahl der Signalquellen, die Art der Dynamik oder die Statistik der Schaltzeitpunkte benötigt.

Der Ansatz von Cacciatore *et al.* 1994 [6] verwendet im Gegensatz zum ACE Algorithmus einen zusätzlichen Klassifikator zur Auswahl der Prädiktoren. Dieser muß sämtliche vorhandenen Dynamiken selbstständig unterscheiden können und damit in ungünstigen Fällen die gesamte Komplexität der Zeitreihe beherrschen. Cacciatore *et al.* konnten mit Ihren Ansatz lineare Signalquellen segmentieren. Der ACE Ansatz realisiert dagegen immer eine *divide-and-conquer* Strategie auf der Basis der individuellen Fehler der Prädiktoren und wurde erfolgreich auf komplexe, nichtlineare Probleme angewandt.

Weigend *et al.* 1995 [70] betrachten ebenfalls die Modellierung schaltender Dynamik. Es wird der EM Algorithmus und ein Ensemble von Netzen (*Mixtures of Experts*, Jacobs *et al.* 1991 [22]) angewandt und ebenfalls ein Wettbewerb zwischen den Einzelnetzen um die Daten induziert. Auch hier gibt es einen globalen

Klassifikator wie bei dem Ansatz von Cacciatore *et al.* 1994 [6], ebenfalls mit dem bereits beschriebenen Nachteil. Die Varianz der Schätzung der Einzelnetze entspricht hier jedoch dem Temperaturparameter im ACE Algorithmus, allerdings hat jedes Einzelnetz eine individuelle Varianz, die ebenfalls adaptiert wird. Bei diesem Ansatz wird dadurch nach unterschiedlichen Varianzen segmentiert. Die Verwendung individueller Varianzen bedeutet jedoch, daß auch ein stationäres Signal nach Vorhersagbarkeit bzw. Rauschanteil segmentiert wird und von einem Ensemble von Prädiktoren modelliert wird (siehe Weigend *et al.* 1995 [70]). Im Gegensatz dazu segmentiert unser Ansatz nicht nach der Größe des Rauschanteils, sondern ausschließlich nach funktional unterschiedlichen Dynamiken.

Ein Schritt zu einer allgemeineren Modellierung der Übergangscharakteristik zwischen zwei Dynamiken stellt die Berücksichtigung von Drift dar (Kohlmorgen *et al.* 1996, 1997, 1998 [30, 31, 32, 33]). Anstelle des bisher modellierten Schaltens sollen damit auch langsame Übergänge in der Dynamik erkannt werden, die bei realen Daten, wie etwa Sprachdaten oder physiologischen Daten, zu erwarten sind. Diese langsamen Übergänge können mit den von uns entwickelten Verfahren modelliert werden.

Diese Arbeit ist wie folgt aufgebaut: Im Anschluß an diese Einleitung werden in Kapitel 2 zunächst die benötigten Grundlagen aus den Bereichen Zeitreihenvorhersage und neuronale Netze erläutert. In Kapitel 3 wird ein *unüberwachtes* Lernverfahren vorgestellt, mit dem es möglich ist, eine nichtstationäre Datensequenz in stationäre Abschnitte zu unterteilen (Segmentation) und dabei gleichzeitig die unterschiedlichen Dynamiken zu identifizieren. Der Ansatz basiert auf einem harten Wettbewerb (*hard competition*) eines Ensembles neuronaler Netze. Die Methode wird in Kapitel 4 theoretisch fundiert. In Kapitel 5 wird dann das ACE-Verfahren (*Annealed Competition of Experts*) vorgestellt, das auf *soft competition* basiert und eine Verbesserung gegenüber der *hard competition* Methode darstellt. In Kapitel 6 wird, aufbauend auf dem bisherigen Ansatz, ein neuer Segmentationsalgorithmus vorgestellt, mit dem es möglich ist, nicht nur ein Schalten, sondern auch ein Driften der Dynamik zu erkennen. Die Drift zwischen zwei dynamischen Zuständen wird dabei zunächst als zeitlich linearer Übergang modelliert. In Kapitel 7 wird dann die Erkennung nichtlinearer Übergänge behandelt. In Kapitel 8 erfolgt eine Anwendung der in dieser Arbeit entwickelten Methoden auf physiologische Wach/Schlaf-Daten (EEG, EOG, Atmung) aus einem Schlaflabor. Die neue Analysemethode ermöglicht dabei eine hohe zeitliche Auflösung der dynamischen Struktur der untersuchten Daten. In Kapitel 9 erfolgt schließlich eine abschließende Diskussion und Bewertung der Ergebnisse.

Kapitel 2

Zeitreihenvorhersage mit neuronalen Netzen

Dieses Kapitel führt in die benötigten Grundlagen der Zeitreihenvorhersage und neuronalen Netze ein. Bezüglich der neuronalen Netze beschränken wir uns dabei auf die Beschreibung von Netzen mit radialen Basisfunktionen (*RBF-Netze*), da wir in den Beispielen und Anwendungen ausschließlich dieses Modell verwenden. Dieses Modell wurde deshalb gewählt, weil es ein einfaches und schnelles Lernverfahren dafür gibt, daß sich als sehr robust und unproblematisch erwiesen hat. Grundsätzlich sind die in dieser Arbeit vorgestellten Methoden zur Analyse nichtstationärer Zeitreihen jedoch unabhängig von der Wahl der verwendeten Funktionsapproximatoren. Es sei daher bereits an dieser Stelle darauf hingewiesen, daß die RBF-Netze nur als *ein* Beispiel aus einer Vielzahl von möglichen Regressionsmethoden zu sehen sind, die für unsere Verfahren verwendet werden können.

2.1 Vorhersage von Zeitreihen

Als Zeitreihe bezeichnet man eine zeitliche Folge von Werten, die in einem mathematischen oder physikalischen System erzeugt bzw. gemessen werden. Sei $\{x(t)\}$, $t = 1, \ldots, T$, eine Zeitreihe, die von einem dynamischen System erzeugt wird. Der Einfachheit halber nehmen wir an, daß x eine skalare Größe ist; die Behandlung von Vektoren erfolgt analog dazu. Wir nehmen weiter an, daß $\{x(t)\}$ eine

Projektion der Dynamik in einem höherdimensionalen Zustandsraum ist. Wenn wir davon ausgehen, daß die Dynamik in einem solchen Zustandsraum deterministisch ist, müssen wir versuchen, den Zustandsraum anhand der $\{x(t)\}$ zu rekonstruieren. Dann nämlich wäre es möglich, den Determinismus des Systems zu nutzen und die Zeitreihe vorherzusagen.

Eine Methode, den Zustands- oder Phasenraum zu rekonstruieren, wurde von Packard *et al.* [50] eingeführt und von Takens [64] mathematisch fundiert. Hierbei wird ein Zustandsvektor $\vec{x}(t)$ wie folgt definiert:

$$\vec{x}(t) = \begin{pmatrix} x(t) \\ x(t-\tau) \\ x(t-2\tau) \\ \vdots \\ \vdots \\ x(t-(d-1)\tau) \end{pmatrix} \tag{2.1}$$

wobei τ eine Verzögerungszeit und d die Einbettungsdimension ist. Wenn die Dynamik des Systems auf einem Attraktor[1] der Dimension D stattfindet, ist eine notwendige Bedingung für den Determinismus von $\vec{x}$:

$$d \geq D \tag{2.2}$$

Ist die Einbettungsdimension d groß genug gewählt, so daß $\vec{x}(t)$ den Zustand des Systems zum Zeitpunkt t eindeutig beschreibt, dann läßt sich für Punkte auf dem Attraktor folgende Gleichung aufstellen:

$$x(t+p) = f^*(\vec{x}(t)) \tag{2.3}$$

Dabei ist f^* eine Abbildung, mit der sich die Zeitreihe $\{x(t)\}$ vorhersagen läßt. p ist die Vorhersagezeit. Takens hat gezeigt, daß eine obere Schranke für die Einbettung d existiert, innerhalb derer sich eine kontinuierliche Funktion f^* finden läßt:

$$d \leq 2D + 1 \tag{2.4}$$

Die Verzögerungszeit τ ist bei unendlich langen Zeitreihen beliebig wählbar, nicht jedoch, wenn nur eine begrenzte Anzahl an Daten vorliegt. τ sollte im konkreten Fall nicht zu klein gewählt werden, da sonst $x(t) \approx x(t-\tau)$ gilt, womit die

[1] Ein Attraktor ist eine Teilmenge des Zustandsraums, die die stationäre Lösung des Systems für $t \to \infty$ darstellt.

Information der Zeitreihe unzureichend repräsentiert wäre. Ein zu großer Wert von τ läßt dagegen den kausalen Zusammenhang zwischen aufeinanderfolgenden Werten verlorengehen und ist daher ebenfalls ungeeignet. Liebert, Pawelzik und Schuster [37] haben z. B. eine Methode beschrieben, mit der sich τ und d gleichzeitig und optimal bestimmen lassen.

Es mag vielleicht überraschen, daß ein funktionaler Zusammenhang zwischen Werten einer Zeitreihe besteht, etwa einer Zeitreihe, die von komplizierten nichtlinearen Differentialgleichungen erzeugt wurde. Daß solch eine Abbildung existiert, ist jedoch eine Konsequenz aus Takens' Theorem. Leider liefert es keinen Hinweis darüber, wie die Funktion f^* aussehen könnte.

Wie in vielen Publikationen [7, 36, 41, 72] inzwischen belegt wurde, sind neuronale Netze sehr gut dazu geeignet, den funktionalen Zusammenhang f^* anhand der Daten aus der Zeitreihe zu „erlernen". Es hat sich gezeigt, daß neuronale Netze den bisherigen konventionellen Methoden zur Funktionsapproximation überlegen sein können [69]. Insbesondere können neuronale Netze auch komplizierte nichtlineare Funktionen mit (im Prinzip) beliebiger Genauigkeit approximieren [18, 21].

2.2 Neuronale Netze mit radialen Basisfunktionen

Zur Vorhersage von Zeitreihen eignen sich inbesondere neuronale Netze mit radialen Basisfunktionen (*Radial Basis Function (RBF) Networks*). Im folgenden wird das von Moody und Darken [41, 42] eingeführte RBF-Netz mit lokalen rezeptiven Feldern vorgestellt.[2] Der wesentliche Vorteil dieses Ansatzes liegt in der schnellen und zuverlässigen Konvergenz des Lernverfahrens begründet. Darüber hinaus unterscheidet sich das Moody-Darken-Netz vom „traditionellen" *Multi-Layer Perceptron* [60] in zwei wichtigen Punkten:

1. der Verwendung **lokaler** Aktivationsfunktionen in der Hidden-Schicht: Jede Unit der Hidden-Schicht liefert nur in einem eng begrenzten Bereich im Raum der Eingabevektoren eine nennenswerte Aktivation. Dies ist einer der Gründe dafür, daß das Lernverfahren relativ schnell konvergieren kann.

[2] Die üblichen Grundbegriffe zu neuronalen Netzen [3, 19, 60] werden im folgenden als bekannt vorausgesetzt.

2. der Benutzung eines **hybriden** Lernverfahrens, und zwar einer Kombination aus *supervised-* und *unsupervised learning*.

   - *supervised learning* (Lernen mit „Lehrer"). Hierbei wird vorausgesetzt, daß die korrekte Ausgabe zu einem Eingabevektor $\vec{x}$ bekannt ist. Die Ausgabe $f(\vec{x})$ des Netzes wird mit der korrekten Ausgabe $f^*(\vec{x})$ verglichen und die Netzparameter werden daraufhin entsprechend verändert.[3] Auf diese Weise wird die Output-Schicht des Moody-Darken-Netzes trainiert.

   - *unsupervised learning* (Lernen ohne „Lehrer"). Ist keine Information über eine korrekte Ausgabe vorhanden, spricht man von unsupervised learning. Das Netz sucht selbstständig eine Abbildung der Eingabedaten. Gesteuert wird es dabei nur durch die Daten selbst. Es geht in diesem Fall darum, die Eingabedaten zu strukturieren (*Clustering*). Von dem Netz wird erwartet, die Eingabemuster selbstständig in Kategorien einzuteilen, und die einem Muster zugeordnete(n) Kategorie(n) in geeigneter Form auszugeben. Die Hidden-Schicht des Moody-Darken-Netzes wird *unsupervised* trainiert.

2.2.1 Architektur des Moody-Darken-Netzes

Dem Netz liegt eine einfache Architektur zugrunde (Abb.2.1). Es ist ein feed-forward Netz mit nur einer Hidden-Schicht und vollständiger Verknüpfung zur Ein- und Ausgabeschicht.

Die Input-Units dienen lediglich zur Aufnahme des Eingabevektors $\vec{x}$ und haben keine weitere Funktionalität. Jede Hidden-Unit i hat als einstellbare Parameter ein **lokales Erregungszentrum** $\vec{\mu}_i$ (Vektor) im Definitionsbereich der Eingabevektoren $\vec{x}$ und einen skalaren **Radius** σ_i, der den Abstand zum Erregungszentrum bestimmt, innerhalb dessen ein Eingabevektor noch eine nennenswerte Aktivation der Unit bewirkt. Die Aktivationsfunktion für jede Hidden-Unit

$$g_i(\vec{x}) = \exp\left(-\frac{\|\vec{x} - \vec{\mu}_i\|^2}{\sigma_i^2}\right) \tag{2.5}$$

[3] Auch wenn lediglich die (binäre) Information, ob die Netzausgabe korrekt ist, verwendet wird, handelt es sich um eine Form des *supervised learning*.

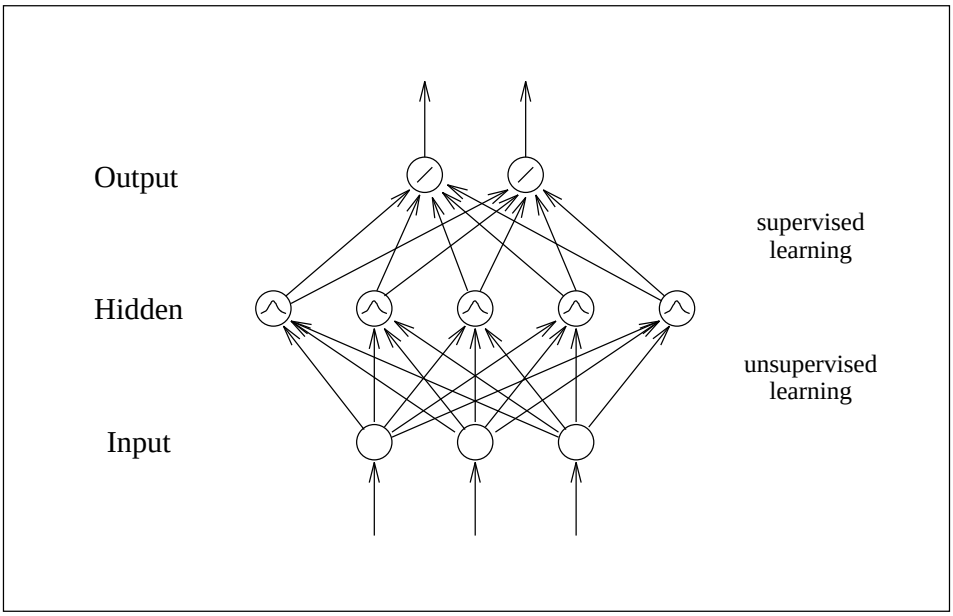


Abbildung 2.1: *Die Architektur des Moody-Darken-Netzes.*

erreicht ein Maximum von 1 für $\vec{x} = \vec{\mu}_i$, d.h. wenn $\vec{x}$ genau im Erregungszentrum von Unit i liegt. Die Aktivation fällt rasch gegen Null ab, je weiter $\vec{x}$ von $\vec{\mu}_i$ entfernt liegt, wobei als Abstandsmaß die euklidische Distanz $\|\vec{x} - \vec{\mu}_i\|$ dient.

Die Aktivationen der über den Argumentbereich der Eingabevektoren verteilten rezeptiven Felder werden in jeder einzelnen *linearen* Output-Unit j einfach durch gewichtetes Aufsummieren zusammengefaßt und normiert:

$$f_j(\vec{x}) = \frac{\sum_i w_{ij}\, g_i(\vec{x})}{\sum_i g_i(\vec{x})} \tag{2.6}$$

Die Ausgabe eines Netzes mit m Output-Units ist $f(\vec{x}) = (f_1(\vec{x}), \ldots, f_m(\vec{x}))$, bestehend aus m skalaren Teilfunktionen gemäß Gl.(2.6). Man erhält so einen gewichteten Mittelwert der w_{ij} derjenigen rezeptiven Felder, die einer Eingabe $\vec{x}$ jeweils am nächsten sind. Das ergibt eine „weiche“ Interpolation der Funktionswerte zwischen den Gewichtungen w_{ij} der rezeptiven Zentren (Abb.2.2). Im Falle skalarer Zeitreihen wird nur *eine* Output-Unit benötigt.

Wie in Abb.2.2 veranschaulicht, realisiert das Moody-Darken-Netz eine Funktionsapproximation durch Interpolation von **Stützwerten** w_{ij} zwischen den **Stützstellen** $\vec{\mu}_i$.

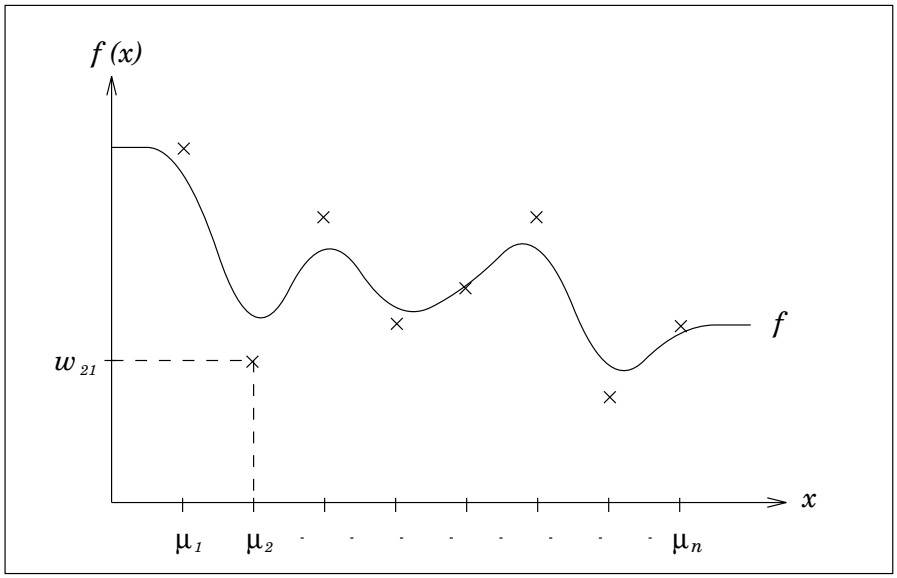


Abbildung 2.2: *Interpolation zwischen den Stützstellen $\vec{\mu}_i$ für $f : \mathcal{R} \to \mathcal{R}$.*

2.2.2 Lernen im Moody-Darken-Netz

Im vorigen Abschnitt wurde die Funktionsweise des Moody-Darken-Netzes erklärt: Die Zielfunktion wird durch „weiche“ Interpolation mit Hilfe von Stützstellen und Stützwerten approximiert. Das setzt geeignete Werte für die Parameter $\vec{\mu}_i, \sigma_i, w_{ij}$ voraus. Im folgenden wird nun das Lernverfahren beschrieben, mit dem diese Werte anhand der gegebenen Daten (aus der Zeitreihe) bestimmt werden.

Bestimmung der $\vec{\mu}_i$ (unsupervised)

Die $\vec{\mu}_i$ sollen die Funktion von *Stützstellen* im Raum der Eingabevektoren haben. Dabei ist es sinnvoll, daß in den Bereichen, in denen sich viele Eingabevektoren befinden, auch viele Stützstellen vorhanden sind. Dort kann dann die Zielfunktion besonders gut angenähert werden. Entsprechend sollen da, wo sich wenige oder keine Eingabevektoren befinden, auch nur wenige bzw. keine Stützstellen vorhanden sein. Gewünscht ist also eine effiziente Verteilung der Stützstellen entsprechend der Häufigkeitsverteilung der Eingabevektoren. Da diese Problemstellung unabhängig von den Zielvektoren ist, wird eine *unsupervised learning* Methode benötigt. Im Grunde kann hierzu jedes beliebige *Clustering*-Verfahren eingesetzt

werden. Ein Überblick über solche Verfahren findet sich z.B. in [11]. Wir verwenden für unseren Ansatz das im folgenden beschriebene *hard competitive learning*, da es sehr einfach und schnell ist. Die Methode wird auch als on-line Version des *K-means clustering* Algorithmus bezeichnet [3].

- Zunächst erfolgt die Initialisierung der $\vec{\mu}_i$ mit möglichst repräsentativen Eingabevektoren. Dazu genügt es im allgemeinen, die $\vec{\mu}_i$ mit zufällig ausgewählten, aber verschiedenen Vektoren $\vec{x}$ des Trainingsdatensatzes vorzubesetzen. Auf diese Weise sind die $\vec{\mu}_i$ dann bereits über den in Frage kommenden Definitionsbereich verteilt.

- Anschließend wird **hard competitive learning** durchgeführt:

  1. Ein beliebiger Vektor $\vec{x}$ wird aus der Menge der Eingabedaten ausgewählt (Zufallsauswahl).
  2. Nur die Hidden-Unit mit dem kürzesten Abstand $\|\vec{x} - \vec{\mu}_i\|$ zum Eingabevektor darf lernen (*winner-takes-all*), sie ist der Gewinner des „Wettbewerbs“ für die Eingabe $\vec{x}$. Die Lernregel für den Gewinner lautet:
  $$\Delta\vec{\mu}_i = \eta\,(\vec{x} - \vec{\mu}_i), \qquad 0 < \eta < 1 \tag{2.7}$$
  Das bedeutet: Für diese Unit wird $\vec{\mu}_i$ ein Stück in Richtung des Vektors $\vec{x}$ verschoben. Wie weit der Vektor $\vec{\mu}_i$ verschoben wird, hängt von der Lernrate η ab.

Unter Umständen kann sogar schon eine günstige Initialisierung der $\vec{\mu}_i$ ausreichen, um eine brauchbare Approximierung der Zielfunktion zu erreichen. Dann kann auf das anschließende *competitive learning* verzichtet werden. Ob dies im Einzelfall zutrifft, kann jedoch nicht ohne weiteres festgestellt werden. Daher sind in der Regel die Punkte 1 und 2 so lange zu wiederholen, bis eine stationäre Verteilung der $\vec{\mu}_i$ erreicht ist. Dies läßt sich insbesondere dadurch forcieren, indem man η im Laufe des Lernverfahrens von einem anfangs großen Wert gegen Null konvergieren läßt,

$$\eta(t+1) = \eta(t)\,\zeta \tag{2.8}$$

ζ läßt sich z.B. dadurch bestimmen, indem man sinnvolle Werte für η am Anfang ($t = 0$) und am Ende ($t = T$) des Lernvorgangs vorgibt, etwa $\eta(0) = 0.8$ und

$\eta(T) = 0.008$. Dies sind die beiden Werte, die wir auch für die meisten Experimente verwendet haben (mit $T = 10 \ldots 40$). Man erhält dann:

$$\zeta = \left(\frac{\eta(T)}{\eta(0)} \right)^{\frac{1}{T}} \tag{2.9}$$

Bestimmung der σ_i (ad hoc)

Die σ_i bestimmen die Größe der rezeptiven Felder, und damit den Grad ihrer gegenseitigen Überlappung. Je mehr sich die Felder überlagern, desto weicher wird der Funktionsverlauf. Moody und Darken schlagen zur Bestimmung der σ_i eine Ad-hoc-Berechnung anstelle eines adaptiven Lernverfahrens vor. Die σ_i berechnen sich danach ausschließlich aus den zuvor gelernten Stützstellen $\vec{\mu}_i$, und zwar für jede Hidden-Unit i als Mittelwert der euklidischen Abstände von $\vec{\mu}_i$ zu den P nächstgelegenen Stützstellen $\vec{\mu}_j$:

$$\sigma_i = \frac{1}{P} \sum_{j \in \mathcal{N}_i^P} \| \vec{\mu}_j - \vec{\mu}_i \| \tag{2.10}$$

wobei $\mathcal{N}_i^P$ die Menge der Indizes der P nächstgelegenen Stützstellen darstellt. Es hat sich jedoch gezeigt, daß $P = 1$ oftmals völlig ausreichend ist [26]. σ_i gibt dann genau den Abstand von $\vec{\mu}_i$ zur nächstgelegenen Stützstelle an. Damit vereinfacht sich Gl.(2.10) zu

$$\sigma_i = \min_{j \neq i} \left\{ \| \vec{\mu}_j - \vec{\mu}_i \| \right\} \tag{2.11}$$

Bestimmung der w_{ij} (supervised)

Die Stützwerte w_{ij} in der Output-Schicht werden – nachdem die Stützstellen $\vec{\mu}_i$ und die Radien σ_i bestimmt wurden – mit einfachem *Backpropagation* [60] trainiert:

$$\Delta w_{ij} = -\eta \, \frac{\partial E}{\partial w_{ij}}. \tag{2.12}$$

Mit dem Fehler

$$E = \frac{1}{2} \sum_j (f_j^*(\vec{x}) - f_j(\vec{x}))^2 \tag{2.13}$$

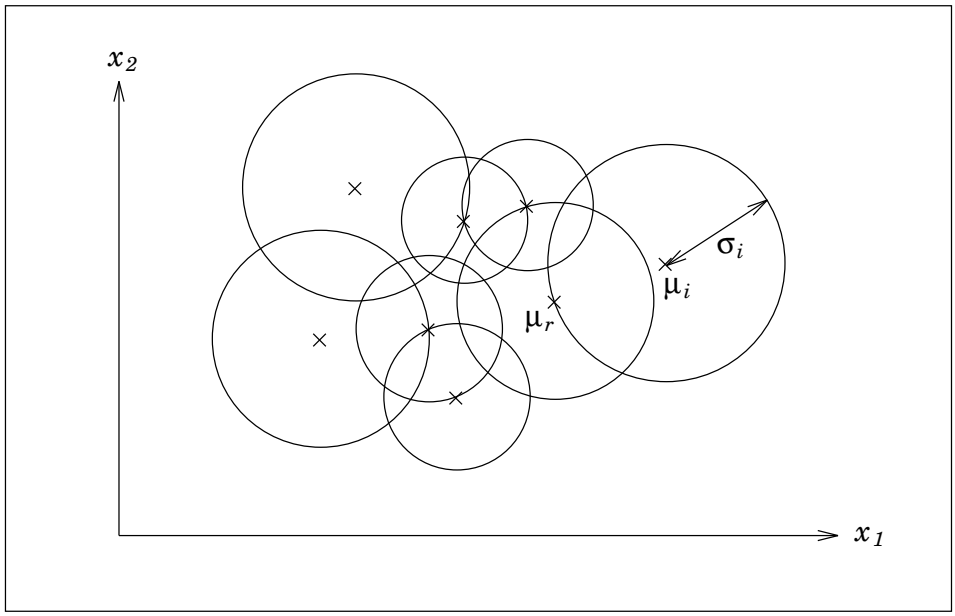


Abbildung 2.3: *Beispiel für Radien σ_i im $\mathcal{R}^2$ nach Gl.(2.11).*

eines Trainingspaares $(\vec{x}, f^*(\vec{x}))$, mit $f^*(\vec{x}) = (f_1^*(\vec{x}), ..., f_m^*(\vec{x}))$, ergibt sich für jede Output-Unit j durch Einsetzen von Gl.(2.6) die folgende Lernregel:

$$\Delta w_{ij} = \eta \left(f_j^*(\vec{x}) - f_j(\vec{x}) \right) \frac{g_i(\vec{x})}{\sum_i g_i(\vec{x})} \tag{2.14}$$

Die Änderungen können für jedes Trainingspaar sofort durchgeführt werden (*online training*).

Diskussion des Lernverfahrens

Die zuvor beschriebene Methode, die Parameter $\vec{\mu}_i, \sigma_i, w_{ij}$ zu bestimmen, wurde von Moody und Darken [41, 42] vorgeschlagen. Prinzipiell können die Parameter von RBF-Netzen natürlich auch auf andere Weise bestimmt werden. Eine mögliche Variante ist es, einen Gradientenabstieg nicht nur für die Stützwerte w_{ij} durchzuführen, sondern für alle Parameter. Diese, insbesondere bei *Multi-Layer Perceptrons* [60] übliche Methode hat jedoch den Nachteil, daß die partiellen Ableitungen für die Parameter $\vec{\mu}_i$ und σ_i nichtlinear sind, womit sich ein bekanntermaßen schwieriges Problem einstellt: es gilt zu vermeiden, in lokalen

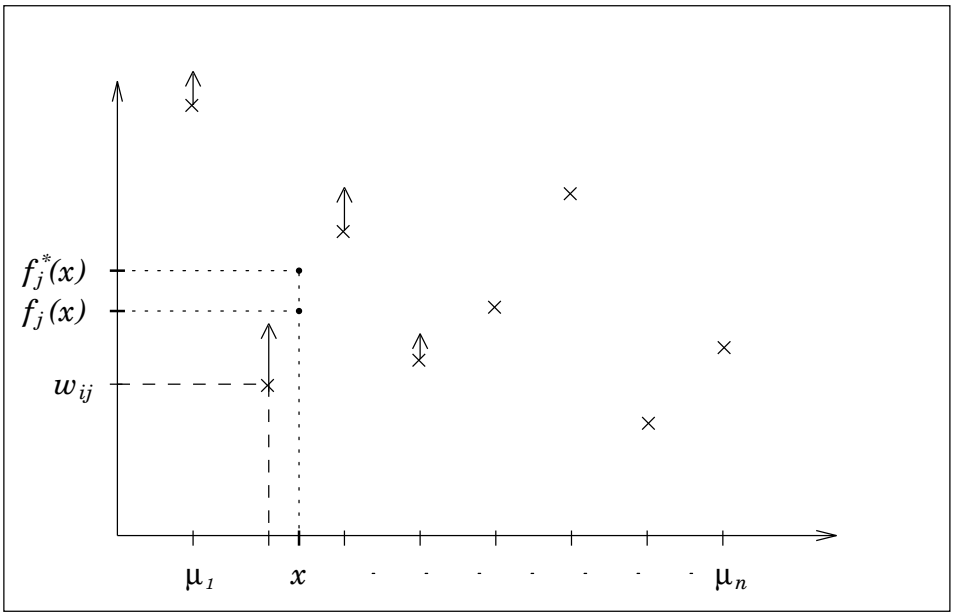


Abbildung 2.4: *Veranschaulichung eines Lernschrittes für eine Output-Unit.*

Minima der Fehlerfunktion E steckenzubleiben. Eine sinnvolle Stragtegie kann daher die folgende Kombination zu sein: zunächst mit dem oben beschriebenen Verfahren von Moody und Darken für eine gute „Initialisierung“ aller Parameter zu sorgen, und dann mit einem Gradientenabstieg eine Feineinstellung sämtlicher Parameter vorzunehmen. Diese Vorgehensweise kann noch einmal eine deutliche Verbesserung der zuvor erreichten Vorhersagegenauigkeit bewirken.[4]

Eine andere Möglichkeit ist es, auf einen Gradientenabstieg vollständig zu verzichten und die bezüglich des Trainingsfehlers E optimalen Stützwerte w_{ij} direkt zu berechnen. Die exakte Lösung dieses lediglich *quadratischen* Optimierungsproblems kann im allgemeinen durch Berechnung der *Pseudo-Inversen* gefunden werden (siehe [3], pp. 92 ff.). Diese Methode setzt allerdings eine feste, vorgegebene Trainingsmenge voraus (*batch learning*). Außerdem ist man eigentlich nicht an der Minimierung des Trainingsfehlers E interessiert, sondern vielmehr an der Minimierung des Generalisierungsfehlers einer Validierungsmenge, deren Daten jedoch nicht zur Optimierung verwendet werden dürfen. Bei iterativen Lernverfahren, wie etwa dem on-line Training der Stützwerte gemäß Moody und Darken, kann man zu diesem Zweck z.B. die *early stopping* Methode verwenden [3]. Dies ist bei der direkten Berechnung der Parameter jedoch nicht möglich.

[4]Persönliche Kommunikation mit A. Röbel und G. Rätsch.

Alternativen gibt es, wie bereits erwähnt wurde, auch bei der Bestimmung der Stützstellen $\vec{\mu}_i$ durch ein *Clustering*-Verfahren. Eine Vielzahl von Varianten steht hier zur Auswahl [11]. Das beschriebene *hard competitive learning* zeichnet sich vor allem durch seine Einfachheit aus. Andere Verfahren, z. B. der *K-means clustering* Algorithmus [3], können ebenfalls angewandt werden.

Kapitel 3

Konkurrierende Prädiktoren zur Analyse schaltender Dynamik

Neuronale Netze werden dazu verwendet, unbekannte Beziehungen in vorgegebenen oder gemessenen Daten zu erkennen. Eine wichtige Voraussetzung für die erfolgreiche Anwendung solcher Systeme ist jedoch eine gewisse Konsistenz der Daten. Insbesondere wird meist davon ausgegangen, daß es sich um Daten aus *stationären* Systemen handelt, d.h. daß die Relationen, die in den Daten enthalten sind, über die Zeit konstant bleiben müssen.

Wenn die Daten jedoch verschiedenen Ursprungs sind, zum Beispiel weil das zugrundeliegende System seine Dynamik ändert, ist es erforderlich, mehrere, voneinander verschiedene, funktionale Zusammenhänge in den Daten zu berücksichtigen. Die klassischen Methoden in der Zeitreihenanalyse und im Bereich neuronaler Netze berücksichtigen diesen Fall nicht und müssen daher an inkonsistenten Daten zwangsläufig scheitern. Ein naheliegender Ansatz um das Problem zu lösen, wäre es, den Daten zusätzliche Informationen mitzugeben, die dazu geeignet sind die Daten zu unterscheiden. Das setzt wiederum voraus, daß Informationen über die Änderungen der Dynamik explizit vorhanden sind (Beispiel: gelabelte Sprachdaten). In der Praxis ist das aber oft nicht der Fall, und ein nachträgliches *Labeling* der Daten ist zumeist sehr teuer oder gar nicht möglich.

Im folgenden wird ein **unüberwachtes** Lernverfahren vorgestellt, mit dem es ohne zusätzliche Informationen möglich ist, eine nichtstationäre Datensequenz zu **segmentieren** und dabei gleichzeitig die unterschiedlichen Dynamiken zu **identifizieren**.

3.1 Identifizierung und Segmentierung schaltender Dynamik

Die grundlegende Idee, ein Verfahren zur unüberwachten Identifizierung und Segmentierung schaltender Dynamik zu entwickeln, ist es, ein *Ensemble* von neuronalen Netzen zu verwenden, die in Konkurrenz zueinander stehen (*competing experts*) [28]. Die Netze sollen sich dabei im Laufe der Trainingsphase auf eine Dynamik spezialisieren und diese vorhersagen lernen.

Competing Experts bzw. *Mixtures of Experts* wurden ursprünglich von Jacobs *et al.* [22] vorgeschlagen, um das Erlernen von konsistenten, widerspruchsfreien Datenmengen zu vereinfachen. Sie erreichen dies durch eine adaptive Aufteilung des Raumes der Eingabevektoren an die einzelnen Netze. Jedes Netz wird dadurch zu einem lokalen Experten. In unserem Fall geht es jedoch um das Problem, die Quellen einer nichtstationären, schaltenden Dynamik zu identifizieren und dabei gleichzeitig die gegebene, inkonsistente Datenmenge in Abschnitte zu unterteilen, in denen die Dynamik stationär ist (Segmentieren). Dabei muß insbesondere das Problem sich überlappender Eingabedatenräume berücksichtigt werden.

Unabhängig von unserer Arbeit haben sich Cacciatore und Nowlan [6] ebenfalls mit schaltenden nichtstationären Systemen beschäftigt und dabei den Ansatz von Jacobs *et al.* weiterentwickelt. Ihre *Mixtures of Controllers* Architektur basiert auf einem Markov-Prozeß, der zu einem rekurrenten Ansatz führt. Obwohl ihre Architektur, ebenso wie unsere, für ein unüberwachtes Lernen der unterschiedlichen Dynamiken ausgelegt ist, konnten sie dies in [6] lediglich bei einer einfachen Aufgabe tatsächlich erreichen: zwei lineare Dynamiken wurden von zwei linearen Netzen richtig segmentiert. In Fällen etwas komplizierterer Dynamik konnte ihr Lernverfahren dagegen ohne die zusätzliche Information darüber, welche Dynamik jeweils vorliegt, nicht zu einer Lösung konvergieren. Außerdem berichten sie von unzureichender Stabilität und zu langsamer Konvergenz ihres Verfahrens. Sie machen leider keine Angaben darüber, was im Falle überzähliger Netze passiert, da sie nur Probleme mit zwei Netzen und zwei Dynamiken beschrieben haben. Somit ist auch nicht geklärt, ob ihr Verfahren auch noch mit mehr als zwei Netzen funktioniert, da dann nicht nur eine Ja/Nein-Entscheidung bei der Auswahl des Netzes getroffen werden muß.

Unser Lernverfahren leidet nicht unter diesen Problemen. Es konvergiert erfahrungsgemäß sehr schnell zu einer stabilen Lösung (sofern es eine gibt), ohne

daß Informationen über die jeweils vorherrschende Dynamik mitgegeben werden müssen. Überzählige Netze fallen dabei automatisch aus dem Lernprozeß.

3.1.1 Die Problemstellung

Sei $\{f_l^*\},\ l = 1, \ldots, L$ eine Menge von Funktionen. Dann ist $\{x(t)\},\ t = 1, \ldots, T$ eine Zeitreihe von wechselnden Anwendungen $l(t)$ dieser Funktionen, wenn

$$x(t) = f_{l(t)}^*(\vec{x}(t)) \tag{3.1}$$

mit

$$\vec{x}(t) = (x(t-\tau),\ x(t-2\tau),\ \ldots,\ x(t-d\tau)) \tag{3.2}$$

gilt. Das bedeutet, daß zu jedem Zeitpunkt t durch l eine Funktion $f_{l(t)}^*$ bestimmt wird, die den Wert $x(t)$ in Abhängigkeit von d vorangegangenen Werten der Zeitreihe festlegt. Die vorangegangenen Werte werden dabei mit einem zeitlichen Abstand τ „abgetastet“ (siehe Abb. 3.1). $\{x(t)\}$ ist dann eine nichtstationäre Zeitreihe, deren Schaltcharakteristik $l(t)$ ist.

Das Ziel ist nun, die Parameter einer Menge von Prädiktoren (neuronalen Netzen) $\{f_k\}, k = 1, \ldots, K$, und eine Segmentation $k(t)$ zu finden, so daß der mittlere Vorhersagefehler für die Zeitreihe minimal wird. Dies soll nur aus den Daten der Zeitreihe ermittelt werden, weitere Informationen über $\{f_l^*\}$ und $l(t)$ stehen nicht zur Verfügung. Die Prädiktoren sollen sich dabei auf je eine (unbekannte) Dynamik f_l^* spezialisieren. Mit der Segmentation $k(t)$ soll dann zu jedem Zeitpunkt t der jeweils zuständige Prädiktor ausgewählt werden, so daß

$$x(t) \approx f_{k(t)}(\vec{x}(t)) \tag{3.3}$$

gilt. Überzählige Prädiktoren ($K > L$) sollen außerdem aus dem Wettbewerb herausfallen und in der endgültigen Segmentation nicht mehr vorkommen.

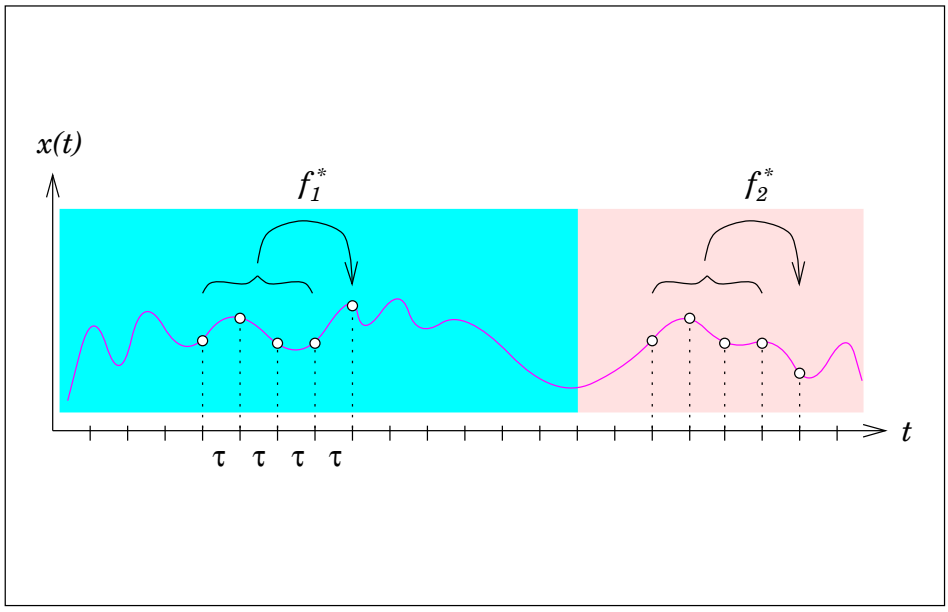


Abbildung 3.1: *Schematische Darstellung einer nichtstationären, schaltenden Zeitreihe. Der funktionale Zusammenhang f^* der Daten $x(t)$ ändert sich an bestimmten Zeitpunkten.*

3.1.2 Competing Experts

Die Idee ist es, ein Team von Experten so um die Daten konkurrieren zu lassen, daß sich jeder Experte auf eine Dynamik spezialisiert. Dazu verwenden wir *hard competition*: Ein Experte (Prädiktor) wird in jedem Lerndurchgang nur mit den Daten trainiert, für die er bereits eine bessere Vorhersage liefert als seine Konkurrenten. Wenn dann ein Experte erst einmal einen bestimmten Bereich der Zeitreihe gut vorhersagen kann, erhält er außerdem einen Wettbewerbsvorteil für zeitlich benachbarte Daten. Damit soll es den Experten ermöglicht werden, sich zum einen in den gefundenen Nischen zu behaupten und zum anderen ihre Zuständigkeit darüber hinaus zu erweitern. Grundlage dieser Vorgehensweise ist die Annahme, daß sich die Dynamik des Systems nicht sofort mit jedem Zeitschritt ändert, sondern einen gewissen Abschnitt lang stationär bleibt, dann den Attraktor wechselt, und auf dem neuen Attraktor wiederum eine Weile verbleibt. Der Algorithmus unseres Lernverfahrens besteht daher aus den folgenden Ablaufschritten:

1. Für alle Prädiktoren f_k werden die Vorhersagefehler $e_k(t)$ bezüglich aller

gegebenen Trainingspaare $(\vec{x}(t), x(t))$ berechnet:

$$e_k(t) = \Big(x(t) - f_k(\vec{x}(t)) \Big)^2 \tag{3.4}$$

2. Anschließend werden die Fehler über die Zeit tiefpaßgefiltert, so daß sie nun auch davon abhängen, wie gut ein Prädiktor die zeitlich benachbarten Werte vorhersagen kann:

$$E_k(t) = \int_{-\infty}^{\infty} G(t - t')\, e_k(t')\, dt' \tag{3.5}$$

Im Falle diskreter Zeitschritte erhält man somit

$$E_k(t) = \sum_{t'=1}^{T} G(t - t')\, e_k(t') \tag{3.6}$$

Dabei ist G eine Funktion, die die Filtercharakteristik festlegt. Verschiedene Filterfunktionen kommen hierzu in Betracht, z.B. die Gaußfunktion

$$G(x) = \exp\left(- \frac{x^2}{\Delta^2} \right) \tag{3.7}$$

Mit Δ läßt sich die Größe des gewünschten „Zeitfensters“ einstellen. Eine andere Möglichkeit ist es, die Fehler $e_k(t)$ in einem explizit festgelegten Zeitfenster zu mitteln; man erhält so den *gleitenden Durchschnitt*

$$E_k(t) = \frac{1}{2\,\Delta + 1} \sum_{t'=-\Delta}^{\Delta} e_k(t + t') \tag{3.8}$$

Sowohl Gl. (3.7) als auch Gl. (3.8) sind akausale Filter, das heißt sie verwenden – bezüglich des Zeitpunkts t – auch zukünftige Werte der Zeitreihe. Sind diese Daten nicht vorhanden, muß das Zeitfenster entsprechend begrenzt werden. Welche Filterfunktion am besten geeignet ist, hängt im Prinzip von dem zu untersuchenden dynamischen System ab. In den in dieser Arbeit untersuchten Systemen haben sich beide oben genannten Filter als gleichermaßen geeignet erwiesen und lieferten sehr ähnliche Ergebnisse. Die Verwendung von Gl. (3.8) ist daher wegen des geringeren Rechenaufwands von Vorteil.

3. *Hard competition*: Für jeden Zeitpunkt t wird der Prädiktor $k(t)$ mit dem kleinsten Fehler $E_{k(t)}(t)$ bestimmt:

$$k(t) = \text{indexmin} \Big\{ E_k(t) \Big\} \tag{3.9}$$

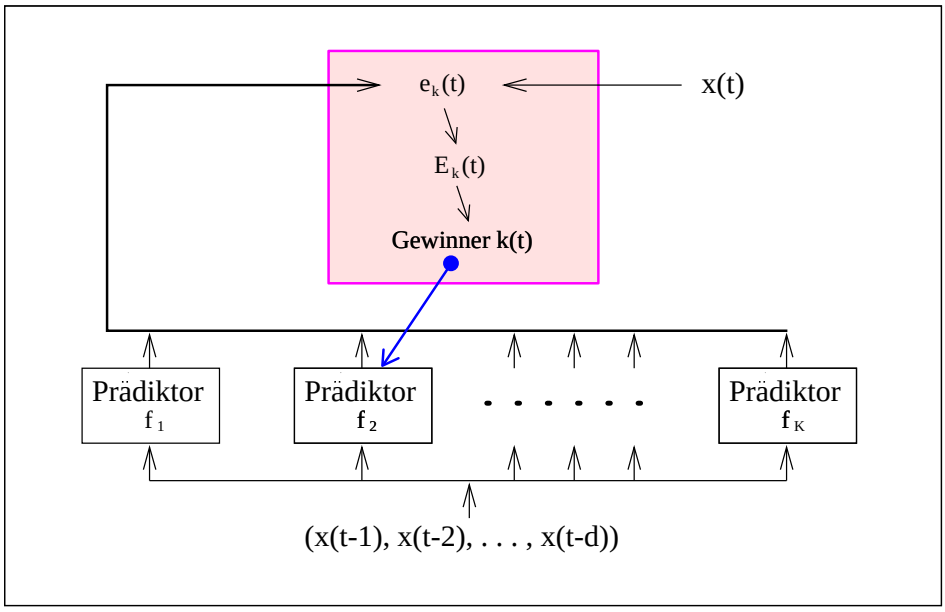


Abbildung 3.2: *Die Architektur unseres Ansatzes besteht aus einem Ensemble konkurrierender Prädiktoren. Jeder Prädiktor erhält dieselben Daten und liefert eine Vorhersage für $x(t)$. Die resultierenden Vorhersagefehler $e_k(t)$ werden über die Zeit tiefpaßgefiltert, $E_k(t)$, und nur der Prädiktor mit dem kleinsten Fehler $E_k(t)$ darf auf den zugehörigen Daten trainieren (in diesem Beispiel: f_2. $\tau = 1$).*

Nur mit diesem Prädiktor wird dann ein Lernschritt auf dem zugehörigen Trainingspaar $(\vec{x}(t), x(t))$ durchgeführt. Mit Gl.(3.9) ist darüber hinaus eine Definition für die Segmentationsfunktion $k(t)$ gegeben. Sie gibt an, welcher Prädiktor zum Zeitpunkt t für die Vorhersage von $x(t)$ zuständig ist. Wenn die Prädiktoren erst einmal eine bestimmte Dynamik repräsentieren, gibt $k(t)$ dann auch den dynamischen Modus zum jeweiligen Zeitpunkt an.

4. Die Schritte 1 bis 3 sind so lange zu wiederholen, bis sich keine Verbesserungen des Gesamt-Vorhersagefehlers

$$E_{RMS} = \sqrt{\frac{1}{T}\sum_{t=1}^{T} e_{k(t)}(t)} \tag{3.10}$$

mehr erreichen lassen. E_{RMS} wird als *Root-Mean-Squared Error* (RMSE) bezeichnet [3].

Der oben beschriebene, iterative Algorithmus ist unabhängig davon, welche Funktionsapproximatoren als Prädiktoren benutzt werden. Im Prinzip können dazu al-

le Modelle verwendet werden, die ein iteratives, datengetriebenes Lernverfahren benutzen. Auch ist es denkbar, verschiedene Prädiktoren in einem Team zusammenzubringen, z.B. lineare und nichtlineare Prädiktoren. Dabei ist jedoch darauf zu achten, daß alle verwendeten Prädiktoren eine etwa gleichgroße Lerngeschwindigkeit haben, weil sonst langsam lernende Prädiktoren von vornherein aus dem Wettbewerb herausgedrängt werden könnten.

3.1.3 Einbindung der Moody-Darken Netze

Bei der Wahl der Prädiktoren haben wir uns für die RBF-Netze, die bereits in Kapitel 2.2 beschrieben wurden, entschieden. Sie haben insbesondere den Vorteil, daß sie sich in der oben beschriebenen Wettbewerbsphase mit sehr geringem Rechenaufwand trainieren lassen, so daß die gesamte Trainingsprozedur auf einer SUN 10 typischerweise nur wenige Sekunden bis eine Minute dauert. Die Trainingszeit hängt dabei insbesondere von der Größe der Trainingsmenge bzw. der Länge der Zeitreihe ab.

Parameterisierung

Die Anzahl der Netze, die für das Wettbewerbslernen anzusetzen sind, hängt davon ab, wieviele Einzeldynamiken man erwartet bzw. auflösen möchte. Werden mehr Netze als benötigt verwendet, dann fallen die überflüssigen Netze entweder automatisch aus dem Wettbewerb heraus oder sie spezialisieren sich auf einzelne Daten an den Übergängen. Die Möglichkeit, daß sich zwei Netze auf dieselbe Dynamik spezialisieren, ist aufgrund der *hard competition* praktisch ausgeschlossen (*soft competition* siehe Kapitel 5). Werden andererseits zu wenig Netze verwendet, dann werden sich einige Netze zwangsläufig auf die Vorhersage von zwei oder mehr Einzeldynamiken spezialisieren und dadurch typischerweise so etwas wie das arithmetische Mittel der entsprechenden Zielfunktionen, $\sum_i f_i^*$, erlernen. Man sollte also immer etwas mehr Prädiktoren ansetzen als man Dynamiken erwartet oder auflösen möchte.

Die Anzahl an Input- und Output-Units wird allein durch die Einbettungsdimension d und die Dimension der Werte $x(t)$ in der Zeitreihe, die bisher als skalare Größen betrachtet wurden, bestimmt. Allgemein gilt: Sind die $x(t)$ n-dimensionale Vektoren, so werden $n \times d$ Input-Units und n Output-Units benötigt.

Die Einbettungsdimension d und das Abtast-Interval τ können bei einer stationären Dynamik z.B. mit der Methode von Liebert, Pawelzik und Schuster [37] gefunden werden. Im Falle einer nichtstationären, schaltenden Dynamik kann die Einbettung für die Einzeldynamiken jedoch meist wesentlich kleiner sein als dies für ein globales Modellieren der Zeitreihe erforderlich wäre [55]. Bevor man die Einzeldynamiken nicht durch Segmentation isoliert hat, kann man die jeweils passende Einbettung aber nicht berechnen. Als Heuristik kann jedoch gelten, daß die Einbettung so gewählt werden sollte, daß sich die Struktur des Signals in den Komponenten des Einbettungsvektors in den Grundzügen wiederfindet.

Wieviele Hidden-Units verwendet werden, ist dagegen im Prinzip frei wählbar. Je mehr Hidden-Units verwendet werden, desto komplexere Funktionsverläufe können realisiert werden. Wenn jedoch nur wenige Trainingsdaten zur Verfügung stehen, dürfen nicht zu viele Hidden-Units angesetzt werden, da sonst das bekannte Problem des *overfitting* auftritt [3]: die gegebenen Trainingsdaten können zwar sehr gut reproduziert werden (kleiner Vorhersagefehler auf der Trainingsmenge), die Vorhersage neuer Daten ist aber dennoch mit einem großen Fehler behaftet (großer Vorhersagefehler auf einer Testmenge).

Um ein *overfitting* bei neuronalen Netzen zu vermeiden und eine gute Generalisierungsfähigkeit zu erreichen, wurden eine Reihe von Kriterien und Techniken zur *Modellselektion* und *Regularisierung* entwickelt [3, 47, 48], die hier eingesetzt werden können. Insbesondere kann die Methode des *early stopping* [3] leicht in das Lernverfahren der *competing experts* integriert werden, indem man einen Teil der gegebenen Daten als Testmenge zurückhält, die nicht für das Training verwendet wird, und nur so lange trainiert, bis ein Minimum des Fehlers auf den *Testdaten* erreicht wird. Der Lernalgorithmus ist dann also dahingehend abzuändern, daß in Schritt 3 nur auf den verbliebenen Trainingsdaten gelernt wird, und in Schritt 4, Gl. (3.10), nur die Fehler der Testdaten berücksichtigt werden.

Wir haben jedoch festgestellt, daß die von uns verwendeten Moody-Darken RBF-Netze grundsätzlich relativ robust gegen *overfitting* sind. Das liegt zum einen an der Normierung in Gl. (2.6), die dafür sorgt, daß die Funktionswerte zumindest immer zwischen den Stützwerten liegen, und zum anderen an den sich überlappenden Gauß-Glocken der Basisfunktionen, die sich aus Gl. (2.10) bzw. Gl. (2.11) ergeben (siehe auch Abb. 2.3). Zudem macht es grundsätzlich keinen Sinn, mehr Hidden-Units (Basisfunktionen) zu verwenden, als Trainingsdaten pro Netz vorhanden sind, da es sonst mehr Cluster-Zentren als Daten gäbe. Da die Trainingsmenge für ein einzelnes Netz jedoch in unserem Fall während des Trainings

variiert, haben wir das Clustering immer im voraus auf der gesamten Trainingsmenge durchgeführt. Mit den so gewonnenen Basisfunktionen wurden dann alle Netze initialisiert. Im Competing-Experts-Training wurden schließlich nur noch die Gewichte der Output-Units adaptiert. Diese Vorgehensweise ermöglicht zudem einen deutlich geringeren Rechenaufwand in der Wettbewerbsphase. Bei den von uns durchgeführten Experimenten wurde die Anzahl der Hidden-Units systematisch im Bereich 2–200 gesucht: die kleinsten mittleren Vorhersagefehler lagen schließlich im Bereich 6–50 Hidden-Units.

Initialisierung

Damit kein Netz in der Wettbewerbsphase von vornherein durch eine ungünstige Initialisierung benachteiligt wird und deshalb sofort aus dem Wettbewerb herausfällt, wird folgende Ausgangssituation geschaffen: Alle Netze erhalten dieselbe Größe, sie verfügen also über dieselbe Anzahl von Hidden-Units. Darüber hinaus erhalten sämtliche Netze, wie oben bereits erwähnt, dieselben rezeptiven Zentren $\vec{\mu}_i$ und Radien σ_i, und zwar auf die Weise, wie schon in Abschnitt 2.2.2 beschrieben wurde:

- *Hard competition* der $\vec{\mu}_i$ über alle $\vec{x}(t)$ der Zeitreihe.
- Ad-hoc-Berechnung der σ_i nach Gl.(2.11).

Somit sind die Netze bis auf die Gewichte der Output-Schicht, w_{ij}, identisch. Nur die w_{ij} bestimmen in den RBF-Netzen den tatsächlichen Funktionsverlauf, die $\vec{\mu}_i$ und σ_i sorgen lediglich dafür, daß der Argumentbereich der Eingabevektoren $\vec{x}(t)$ sinnvoll mit rezeptiven Feldern bedeckt wird. Damit brauchen in Punkt 3 unseres Lernverfahrens nur noch die Gewichte w_{ij} des Gewinners $k(t)$ (gemäß Gl. (2.14)) angepaßt werden. Die Parameter $\vec{\mu}_i$ und σ_i bleiben dagegen während der gesamten Lernprozedur unverändert.

Damit der Wettbewerb um die Daten der Zeitreihe funktionieren kann, dürfen jedoch keine völlig identischen Netze an den Start gehen. Sie müssen sich ein wenig voneinander unterscheiden, so daß es tatsächlich immer nur *einen* Gewinner gibt. Das Problem unterschiedliche, aber dennoch sinnvolle Initialisierungen für den Start zu finden, wurde so gelöst:

- Zunächst werden die w_{ij}, für alle Netze gleich, jeweils auf die Mittelwerte der j-ten Komponenten der Vektoren $x(t)$ gesetzt (bei skalaren Zeitreihen gibt es dementsprechend nur eine Komponente):

$$w_{ij} = \frac{1}{T} \sum_{t=1}^{T} x_j(t) \tag{3.11}$$

- Für den *ersten* Durchgang der Trainingsphase wird die Zeitreihe $\{x(t)\}$ in k möglichst gleichgroße Abschnitte unterteilt. Jedes Netz darf nun auf dem eigenen Abschnitt einmal trainieren: ein Lernschritt für jedes $x(t)$ in dem Abschnitt. Die Lernrate η sollte dabei sehr klein sein, damit die $x(t)$ nicht wirklich gut gelernt werden und sich die w_{ij} lediglich ein wenig voneinander unterscheiden. Denn die Netze sollen sich nicht wirklich auf den vorgegebenen Abschnitt spezialisieren. Die Hoffnung ist aber, daß die Netze Abschnitte erhalten, in denen bereits *eine* stationäre Dynamik dominiert. Auf diese Weise soll jedes Netz bereits in der Initialisierungsphase eine sinnvolle Ausrichtung erhalten.

Im zweiten und jedem folgenden Durchgang erfolgt dann das *hard competition* Lernverfahren gemäß den Punkten 1 bis 4: Nur das Netz mit dem kleinsten $E_k(t)$ darf einen Lernschritt auf dem zugehörigen Trainingsdatum durchführen.

3.1.4 Erweiterung des Ansatzes

Bei der praktischen Erprobung des neuen Verfahrens wurden folgende Verbesserungen am Algorithmus vorgenommen:

Die Lernrate η und die Größe Δ zur Festlegung des Zeitfensters sollten am Anfang sehr klein sein, und erst im Laufe des Lernverfahrens schrittweise vergrößert werden. Mit dem zunächst kleinen η wird verhindert, daß sich die Netze gleich zu Beginn an den ersten, mehr oder weniger zufällig zugeteilten Daten festsetzen und diese so gut gelernt werden, daß sie später nicht mehr an andere Netze vergeben werden können. Die Netze haben stattdessen mehr Zeit, sich für eine Dynamik zu entscheiden. Daraus folgt aber auch, daß die Netze am Anfang noch recht undifferenziert sind, d.h. sie unterscheiden sich nur geringfügig voneinander und haben noch keine klare Ausrichtung auf eine bestimmte Dynamik. Es macht daher keinen Sinn in diesem Stadium schon allzu große Wettbewerbsvorteile für

benachbarte Daten einzuräumen, da sonst zu beobachten ist, daß ein oder zwei Netze relativ schnell alle Daten der Zeitreihe für sich beanspruchen – unabhängig davon, wieviele Dynamiken tatsächlich vorhanden sind. Δ sollte daher zunächst auch nur einen kleinen Wert haben und erst mit zunehmender Trainingsdauer – und damit auch zunehmender Differenzierung – erhöht werden. Wir sind dabei nach dem folgenden Fahrplan vorgegangen, wobei die verwendeten Indizes t und T einzelne Iterationen des Lernverfahrens bezeichnen:

$$\Delta(t+1) = \Delta(t) + \zeta_\Delta \qquad \eta(t+1) = \eta(t)\,\zeta_\eta \tag{3.12}$$

ζ_Δ und ζ_η können dadurch festgelegt werden, indem man sinnvolle Werte für Δ und η für den Anfang ($t = 0$) und das Ende ($t = T$) des Lernverfahrens wählt. Man erhält dann:

$$\zeta_\Delta = \frac{\Delta(T) - \Delta(0)}{T} \qquad \zeta_\eta = \left(\frac{\eta(T)}{\eta(0)}\right)^{\frac{1}{T}} \tag{3.13}$$

Insbesondere die Wahl des finalen Zeitfensters $\Delta(T)$ ist problemabhängig. Hierzu muß abgeschätzt werden, wie lang die stationären Abschnitte der Dynamik mindestens sind. Hier sollte ein eher kleiner Wert angesetzt werden, der jedoch bei stark verrauschten Daten oder zur Unterscheidung ähnlicher Dynamiken problematisch sein kann, da beides die Identifikation des Operationsmodus erschwert. Wir verwendeten in unseren Anwendungen Zeitfenster $\Delta(T)$ im Bereich 3 bis 40 und $\Delta(0) = 0$. Für die Lernrate wurde meist $\eta(0) = 0.03$ und $\eta(T) = 0.3$ angesetzt. Grundsätzlich sollte $\eta(T)$ nicht größer als 0.5 sein, keinesfalls aber größer als 1.5, da sonst das RBF-Lernverfahren nicht mehr konvergiert.

Folgende Regel wurde zusätzlich benutzt, um das Verfahren zu beschleunigen:

> *Hat ein Netz erstmalig eine bestimmte Anzahl H von gewonnenen Trainingsdaten unterschritten, wird es aus dem Wettbewerb genommen und nimmt nicht mehr am Lernverfahren teil.*

Für die betroffenen Netze brauchen somit keine weiteren Berechnungen mehr durchgeführt zu werden. Es ergibt sich darüber hinaus ein weiterer, positiver Effekt: Ohne diese Regel kann es vorkommen, daß ein Netz, das ab einem bestimmten Zeitpunkt keine Daten mehr zugeteilt bekommt und damit eigentlich aus dem

Wettbewerb herausgefallen ist, im späteren Verlauf dennoch wieder Daten zum Trainieren erhält. Das liegt daran, daß die in dem Zeitfenster gemittelten Fehler E_k an den Segmentationsgrenzen, wegen der dort vorhandenen Inkonsistenz der Daten, immer relativ hoch sind. Dort ist es dann für Netze, die eigentlich gar nicht mehr benötigt werden, möglich, Daten zu „stehlen“, und zwar dann, wenn sie die Mittelwerte der beiden am Schalten beteiligten Dynamiken gelernt haben (siehe dazu auch Abb. 3.4(d)). Ein solcher Segmentierungsfehler kann mit der oben genannten Regel verhindert werden. Wir haben in den Experimenten entweder $H = 0$ oder $H = 1$ verwendet.

Bei einigen Versuchen haben wir Zeitreihen verwendet, die abschnittsweise reines Rauschen enthielten. Es gab also Abschnitte, auf denen eine Vorhersage prinzipiell nicht möglich war. Das Lernverfahren bestimmt jedoch immer ein Gewinner-Netz für jeden Datenpunkt, also auch Gewinner für Daten, die nicht vorhersagbar sind. Das führt dazu, daß die Netze immer wieder von ihrer eigentlich zu erlernenden Vorhersagefunktion abgebracht werden. Selbst dann, wenn sämtliche Netze einen Datenpunkt mit sehr hohem Fehler vorhersagen (z.B. auch *outliers* [3]), muß das Netz, dessen Vorhersage „am wenigsten schlecht“ ist, auf dem Datenpunkt trainieren. Um zu verhindern, daß die Netze auf diese Weise beim Lernen gestört werden, wurde folgende, zusätzliche Bedingung für einen Lernschritt eingeführt:

$$e_{k(t)}(t) < \theta \tag{3.14}$$

Es erfolgt somit nur noch dann ein Lernschritt auf einem Datenpunkt, wenn der Vorhersagefehler für diesen Datenpunkt bereits eine bestimmte Grenze, gegeben durch den Wert θ, unterschritten hat.

3.2 Beispiele

In diesem Teil werden einige Experimente vorgestellt, in denen das oben beschriebene Verfahren zur Identifizierung und Segmentierung von nichtstationären, schaltenden Zeitreihen verwendet wurde. Dabei wird zunächst ein einfaches Beispiel vorgestellt, das zur Illustration der Funktionsweise unseres Ansatzes dienen soll. Anschließend wird ein nichtstationäres System von Mackey-Glass Differentialgleichungen [38] behandelt. Danach werden reale Daten aus der Spracherkennung und biologische Daten untersucht.

3.2.1 Zwei chaotische Abbildungen

Im folgenden wird beschrieben, wie eine nichtstationäre Zeitreihe von einem Ensemble, das aus sechs Prädiktoren besteht, segmentiert wird. Die Zeitreihe wird durch eine alternierende Anwendung von zwei eindimensionalen Abbildungen erzeugt: der Sinusfunktion

$$f_1(x) = \sin(\pi\, x) \tag{3.15}$$

und der doppelt-logistischen Abbildung

$$f_2(x) = f_{log}(f_{log}(x)), \text{ mit } f_{log}(x) = 4\, x\, (1 - x). \tag{3.16}$$

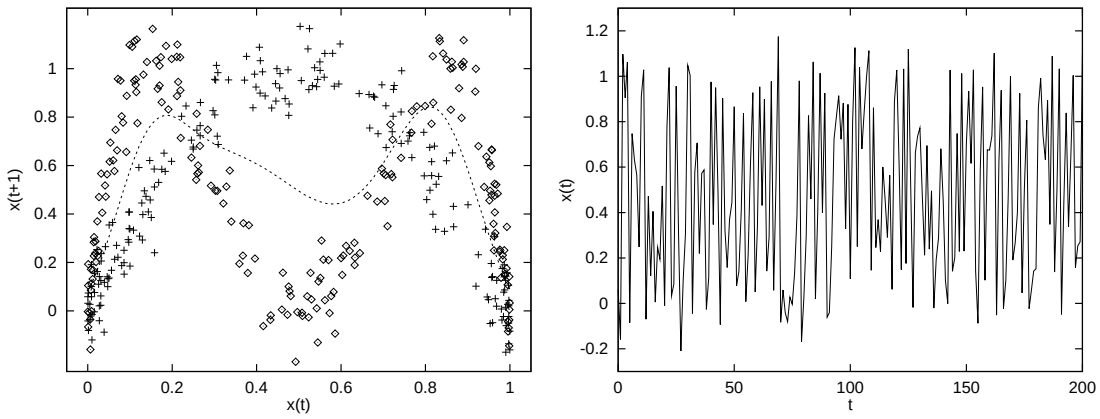
Wir verwenden dabei eine Zeitreihe $\{x(t)\}$, die alle 50 Zeitschritte, $t = 1, \ldots, 400$, mit Ausnahme von $t = 300$, die Dynamik wechselt. Dieses Schaltverhalten wird repräsentiert durch $l(t)$:

$$x(t+1) = f_{l(t)}(x(t)), \quad \text{mit } l(t) \in \{1, 2\} \tag{3.17}$$

Darüber hinaus wird den Daten normalverteiltes Rauschen ($\sigma = 0.1$, entspricht einem Signal-Rausch-Abstand von 11 dB) hinzuaddiert. Abbildung 3.3(a) zeigt die auf diese Weise generierten 400 Trainingsdaten $x(t+1)$ in Abhängigkeit der vorangehenden Werte $x(t)$, wobei für die Daten, die mit der logistischen Abbildung erzeugt wurden, andere Symbole verwendet werden (Kreuze), als für die Daten, die mit der doppelt-logistischen Abbildung erzeugt wurden (Rauten). Es ist leicht zu erkennen, daß man diese Daten nicht angemessen durch *eine* Funktion repräsentieren bzw. mit einem einzelnen Prädiktor vorhersagen kann. Die richtige Vorhersage eines Wertes kann nur gelingen, wenn man zunächst die beiden Dynamiken unterscheiden kann. Dem Team von Prädiktoren steht jedoch nur die *ungelabelte* Zeitreihe aus Abb. 3.3(b) zur Verfügung, d.h. die Information, welche Dynamik jeweils vorherrscht (Rauten, Kreuze), ist nicht explizit gegeben. In dem Plot sieht man die Daten, mit Linien verbunden, für $t = 1, \ldots, 200$: das Schalten der Dynamik an den Zeitpunkten $t = 50, 100, 150$ ist optisch nicht zu erkennen.

Das Lernverfahren wird folgendermaßen durchgeführt:

- Als Prädiktoren verwenden wir 6 RBF-Netze mit jeweils 10 Hidden-Units. Sie sollen aus $x(t)$ den Wert von $x(t+1)$ vorhersagen.



(a) (b)

Abbildung 3.3: *(a) Die Trainingsdaten* $(x(t), x(t+1))$ *wurden unter Verwendung von Gl. (3.17) generiert, d.h. sie wurden mit Hilfe der beiden Abbildungen* f_1 *(Kreuze) bzw.* f_2 *(Rauten) erzeugt und zusätzlich verrauscht. Es ist offensichtlich, daß alle Daten zusammengenommen nicht von einer einzigen Funktion repräsentiert werden können. Trainiert man dennoch ein einzelnes neuronales Netz auf allen Daten, so erhält man eine Mittelwertkurve, die das zugrundeliegende System nur sehr unzureichend beschreibt (gestrichelte Linie). (b) Dieselben Daten als Zeitreihenplot* $(t, x(t))$*. Die Dynamik wechselt alle* 50 *Zeitschritte die zugrundeliegende chaotische Abbildung, was jedoch rein optisch nicht zu erkennen ist.*

- Der Fehler $E_k(t)$ wird als *gleitender Durchschnitt* (Gl.(3.8)) der Fehler $e_k(t)$ über ein variables Zeitfenster Δ definiert: zu Beginn des Lernverfahrens ist $\Delta = 0$ und damit $E_k(t) = e_k(t)$, am Ende gilt $\Delta = 7$ (d.h. es wird über 15 Fehler e_k akausal gemittelt). Dazwischen wird Δ sukzessive gemäß Gl.(3.12) erhöht.

- Die Lernrate η beträgt anfangs 0.03, sie wird im Verlauf des Trainings bis auf 0.3 erhöht, um die Konvergenz des Lernverfahrens nach erfolgter Spezialisierung zu beschleunigen. Um gegen das Rauschen stabil zu bleiben, wurde die Lernrate jedoch nicht noch größer gewählt.

Mit dieser Ausgangssituation ist es in nur *drei* Iterationen möglich, die 400 Daten der Zeitreihe korrekt zu segmentieren und gleichzeitig die zugrundeliegenden Dynamiken zu erfassen.

Abbildung 3.4 zeigt die Verteilung der Gewinner, (a) vor dem ersten Trainings-

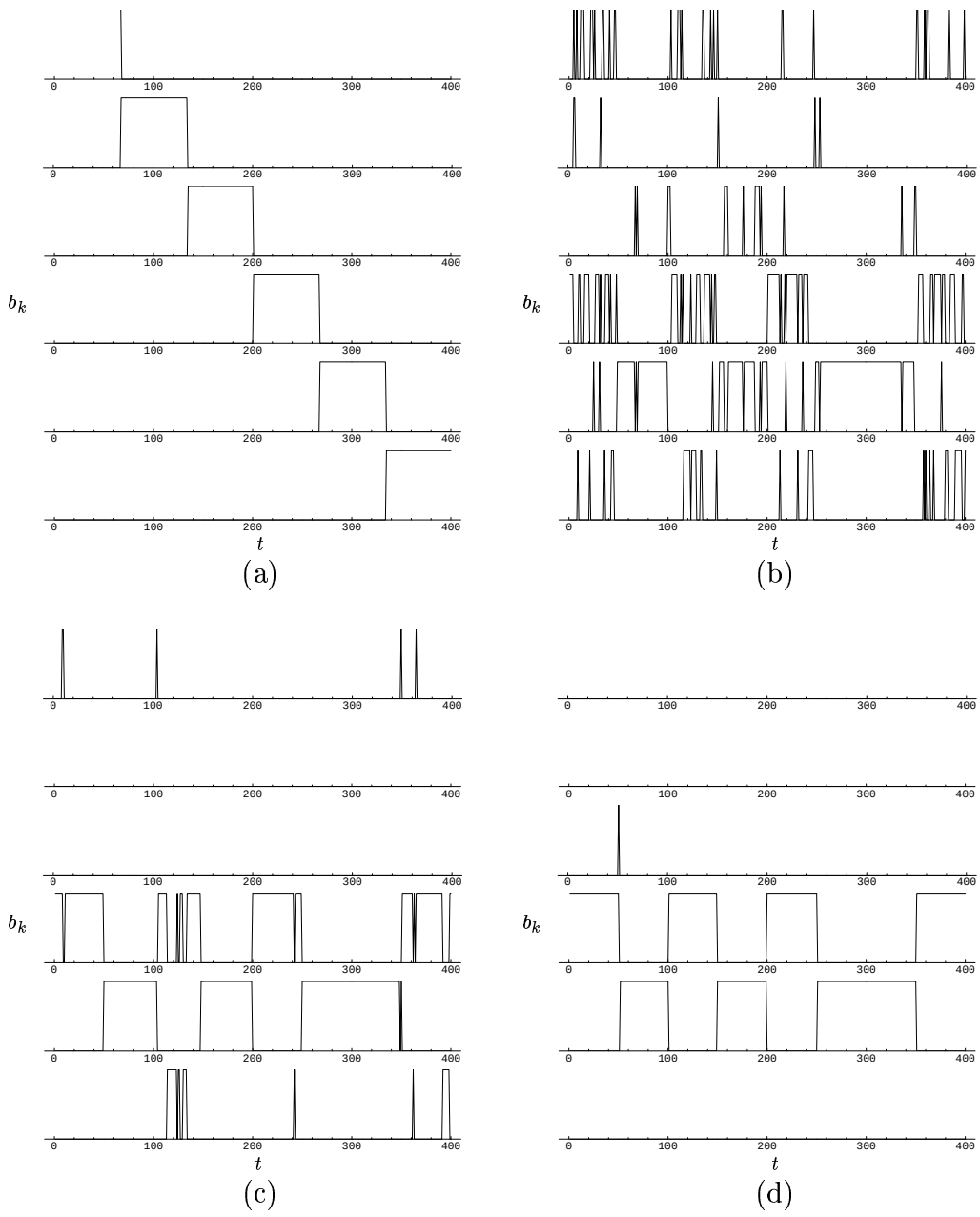


Abbildung 3.4: *(a) Initiale Verteilung der* 400 *Daten an die* 6 *Prädiktoren, vor dem ersten Trainingsdurchgang. (b)–(d) Verteilung der Gewinner nach der ersten (b), zweiten (c) und dritten (d) Iteration der Trainingsphase.*

durchgang, sowie nach der ersten (b), zweiten (c) und dritten (d) Iteration des Algorithmus. Die Bilder zeigen dabei nicht unmittelbar die Segmentationsfunktion $k(t)$, sondern für jedes der sechs Netze einzeln ($k = 1, \ldots, 6$, von oben nach unten), ob ein Datum zum Zeitpunkt t gewonnen wurde (d.h. $k(t) = k$), oder nicht:

$$b_k(t) = \begin{cases} 1 & \text{, falls } k(t) = k \\ 0 & \text{, sonst} \end{cases} \tag{3.18}$$

Eine Ausnahme bildet Abbildung 3.4(a). Hierbei handelt es sich nicht um eine Gewinnerverteilung, sondern um die gleichmäßige Aufteilung der Daten zur Initialisierung für den ersten Trainingsdurchgang (siehe Seite 38). In der Abfolge (b)–(d) sieht man sehr schön, wie sich die richtige Segmentation allmählich herausbildet. Es wird herausgefunden, daß *genau zwei* unterschiedliche Dynamiken der Zeitreihe zugrunde liegen: Nur zwei Netze (4 und 5) können sich bis zum Ende behaupten und haben die Daten unter sich aufgeteilt. Darüber hinaus wurden auch die gesuchten Schaltpunkte $t = 50, 100, 150, 200, 250, 350$ gefunden. Einziger Schönheitsfehler: Obwohl Netz 3 im zweiten Durchgang bereits aus dem Wettbewerb gefallen war (c), konnte das Netz zum Schluß wieder einen Datenpunkt am Übergang $t = 50$ erhalten (d). Auch wenn hier ohnehin offensichtlich ist, daß Netz 3 für die Segmentation keine Rolle spielt, könnte die erfolgte Reaktivierung explizit durch die Wahl von $H > 0$ verhindert werden (siehe dazu Seite 39).

Was haben die sechs Netze nun tatsächlich gelernt, wie sehen inbesondere die von Netz 4 und 5 gelernten Funktionen aus? Dies zeigt Abbildung 3.5. Wie schon in Abbildung 3.4 wird hier der zeitliche Verlauf des Trainings in vier Bildern festgehalten. Abbildung 3.5(a) zeigt den identischen Funktionsverlauf aller sechs Netze zu Beginn des Lernverfahrens. Die Netze wurden so initialisiert, daß sie konstant den Mittelwert aller Daten vorhersagen (Mittelwertprädiktoren, siehe Seite 37). Erst nach dem ersten Trainingsdurchgang zeigen sich gewisse Unterschiede (b). Mit zunehmender Spezialisierung wird dann deutlich, für welche Dynamik sich die Netze entschieden haben (c). Schließlich können sich die Netze 4 und 5 durchsetzen (d) und bieten eine gute Näherung für die beiden Ausgangsfunktionen f_1 und f_2. Man beachte, daß dazu lediglich die 400 verrauschten Daten aus Abb. 3.3(a) zur Verfügung standen.

Abbildung 3.6 zeigt die Fehler $e_k(t)$ und $E_k(t)$ am Ende des Lernverfahrens. Interessant ist vor allem Abb.3.6(a): An den kammartigen Fehlerverläufen der Netze 4 und 5 erkennt man, daß diese auch die jeweils konkurrierende Dynamik an bestimmten Punkten mit niedrigem Fehler $e_k(t)$ vorhersagen können. Der Grund

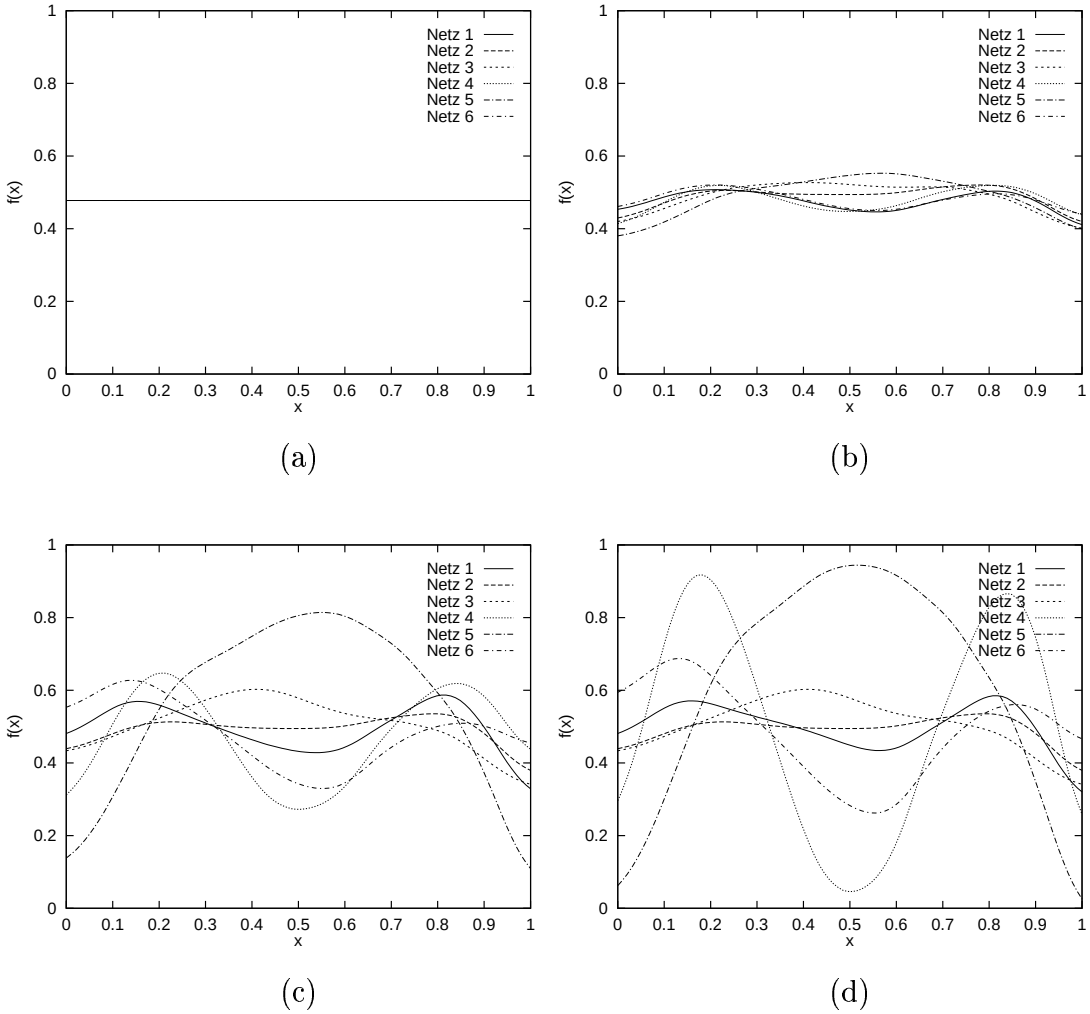


Abbildung 3.5: *Die von den sechs Netzen im Laufe des Trainings erlernten Funktionen. (a) Die Ausgangssituation: Alle sechs Netze sind zunächst Mittelwertprediktoren. (b)–(d) Die erlernten Funktionen nach jedem Trainingsdurchgang.*

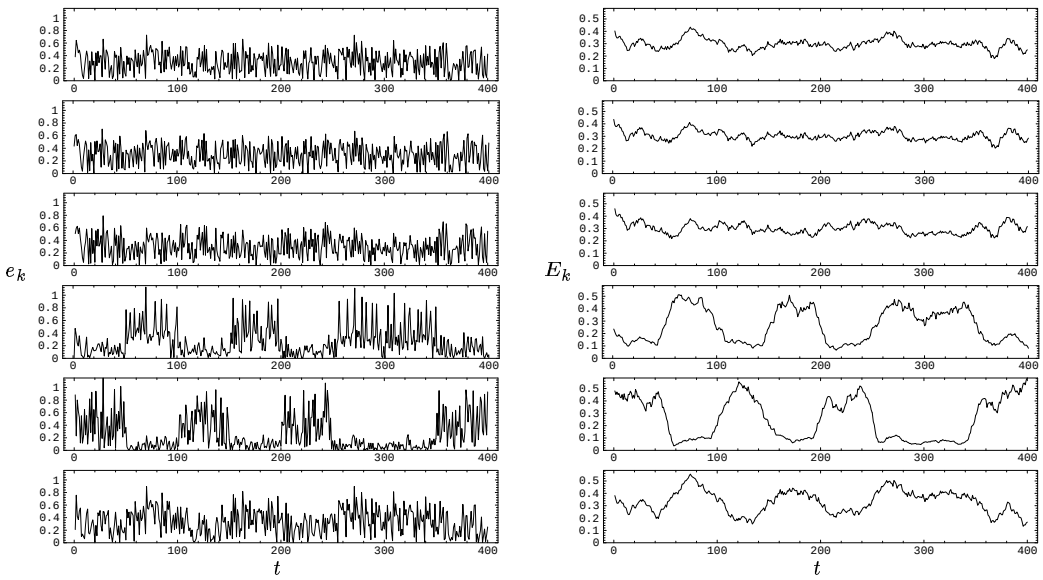


Abbildung 3.6: *Die Fehler $e_k(t)$ (a) und $E_k(t)$ (b) der sechs Netze, am Ende der Trainingsphase. Beide Darstellungen lassen bereits die erzielte Segmentierung erkennen.*

dafür ist, daß f_1 und f_2 an vier Stellen identische Funktionswerte liefern. Ein Datenpunkt in der Nähe solcher Schnittpunkte kann demnach zu beiden Dynamiken gehören. Erst durch die Einbeziehung der zeitlichen Umgebung kann entschieden werden, welche Dynamik tatsächlich vorliegt. Deshalb ist die Verwendung des Tiefpaßfilters $E_k(t)$ notwendig, um die gewünschte Segmentation zu erzielen (Abb. 3.6(b)).

Es ist also festzuhalten, daß die gestellte Aufgabe mit unserem Lernverfahren nahezu perfekt gelöst wurde: es wurde herausgefunden,

- **daß** dem System alternierende Dynamiken zugrunde liegen,
- **wie viele** und
- **welche** es sind, und schließlich noch
- **wann** jede Dynamik auftritt und
- **wo** die Übergänge sind.

Damit ist das der Zeitreihe zugrundeliegende System vollständig beschrieben. An diesem Beispiel sollte deutlich geworden sein, wie unser Verfahren funktioniert. Es war deshalb besonders anschaulich, weil dem System *eindimensionale* Abbildungen zugrunde lagen. Dadurch war es möglich, in Abbildung 3.5 darzustellen, welche Funktionen die RBF-Netze jeweils repräsentieren. Dies ist in dem nachfolgend beschriebenen Experiment nicht mehr der Fall.

3.2.2 Die Mackey-Glass Differentialgleichung

Die Mackey-Glass Differentialgleichung [38]

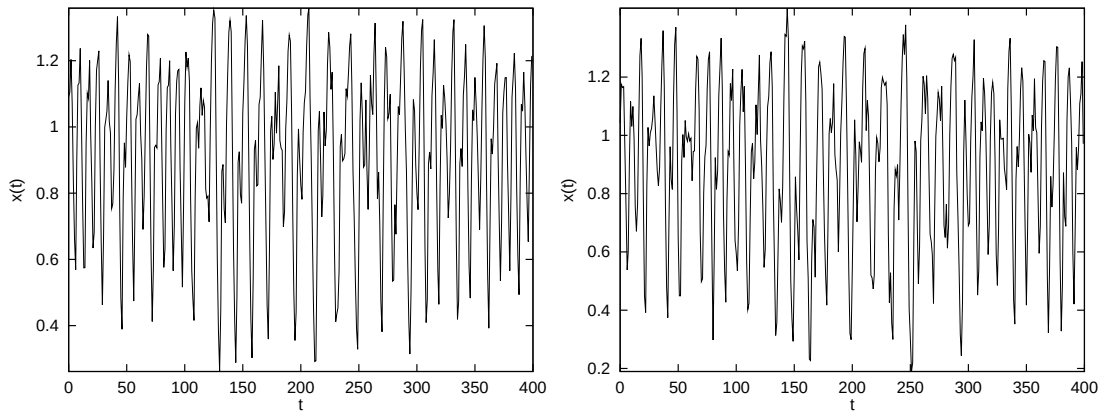
$$\frac{dx(t)}{dt} = -b\,x(t) + \frac{a\,x(t-t_d)}{1+x(t-t_d)^{10}} \tag{3.19}$$

ist ein Modell für die Regulation der Produktion von Blutzellen. Sie ist besonders interessant, weil der Zustandsraum wegen der Zeitverzögerungskonstante t_d *unendlich-dimensional* ist. Außerdem hat sie sich in vielen Untersuchungen zur Zeitreihenvorhersage als Referenz etabliert. Für dieses Experiment wurden deshalb auch die üblichen Parameter zur Vorhersage verwendet, insbesondere $a = 0.2$ und $b = 0.1$ [7].

Um ein nichtstationäres System zu erhalten, wurde der Parameter t_d in der Weise variiert, daß die ersten 100 Trainingsdaten mit $t_d = 17$, die zweiten mit $t_d = 23$, die dritten mit $t_d = 30$ und die vierten schließlich wieder mit $t_d = 17$ generiert wurden. Zur Rekonstruktion des Phasenraums wurden die ebenfalls üblichen Werte $\tau = 6$ und $d = 6$ gewählt (siehe Gl.(3.2), Seite 31).

Die auf diese Weise erzeugte Zeitreihe ist in Abbildung 3.7(a) zu sehen. Die hier dargestellten 400 Daten $x(t)$ $(t = 1, \ldots, 400)$ wurden im zeitlichen Abstand $\tau = 6$ abgetastet. Sie bilden die Trainingsmenge für ein Team von fünf RBF-Prädiktoren, die aufgrund der gewählten Einbettung, $d = 6$, sechs zeitverzögerte Input-Units besitzen. Desweiteren sollen 40 Hidden-Units ausreichen, um den sechsdimensionalen Raum der Eingabedaten mit rezeptiven Feldern abzudecken. Diese 40 Hidden-Units werden zu je einer Output-Unit zusammengeführt, die den nächsten Wert der Zeitreihe vorhersagen soll.

Dem optischen Anschein nach ist in Abbildung 3.7(a) bereits zu vermuten, daß die ersten und letzten 100 Daten einer anderen Dynamik folgen als die mittleren 200 Daten. Es wurde daher noch eine zweite Zeitreihe erzeugt, und zwar mit



(a) (b)

Abbildung 3.7: *Nichtstationäre Zeitreihen aus der Mackey-Glass Differentialgleichung: (a) unverrauscht, (b) verrauscht.*

denselben Parametern wie zuvor, aber zusätzlich wurde nun bei jedem Zeitschritt gleichverteiltes Rauschen im Interval $[-0.05, 0.05]$ hinzugefügt. Dieses Rauschen wurde bereits bei der Generierung der Zeitreihe hinzuaddiert, so daß ein stochastisches System entstand. Die daraus resultierende Zeitreihe läßt nun eher eine einzelne Dynamik vermuten (Abbildung 3.7(b)).

Zum Lernverfahren:

- Für beide Zeitreihen wird ein Trainingslauf über 10 Iterationen durchgeführt.
- Der Fehler $E_k(t)$ wird als *gleitender Durchschnitt* (Gl.(3.8)) der Fehler $e_k(t)$ über ein Zeitfenster Δ definiert: zu Beginn des Lernverfahrens ist $\Delta = 0$, am Ende ist $\Delta = 20$. Dazwischen wird Δ wieder gemäß Gl.(3.12) linear erhöht.
- Die Lernrate η beträgt anfangs 0.2 und wird im Verlauf des Trainings bis auf 1.5 erhöht.
- Sobald ein Prädiktor keine Trainingsdaten mehr gewinnt, scheidet er aus dem Wettbewerb aus (d.h. $H = 1$, siehe Seite 39).

Abbildung 3.8 zeigt die Fehler $e_k(t)$ und die Verteilung der Gewinner am Ende des Lernverfahrens. Dabei zeigen (a) und (b) das Ergebnis für die unverrauschte

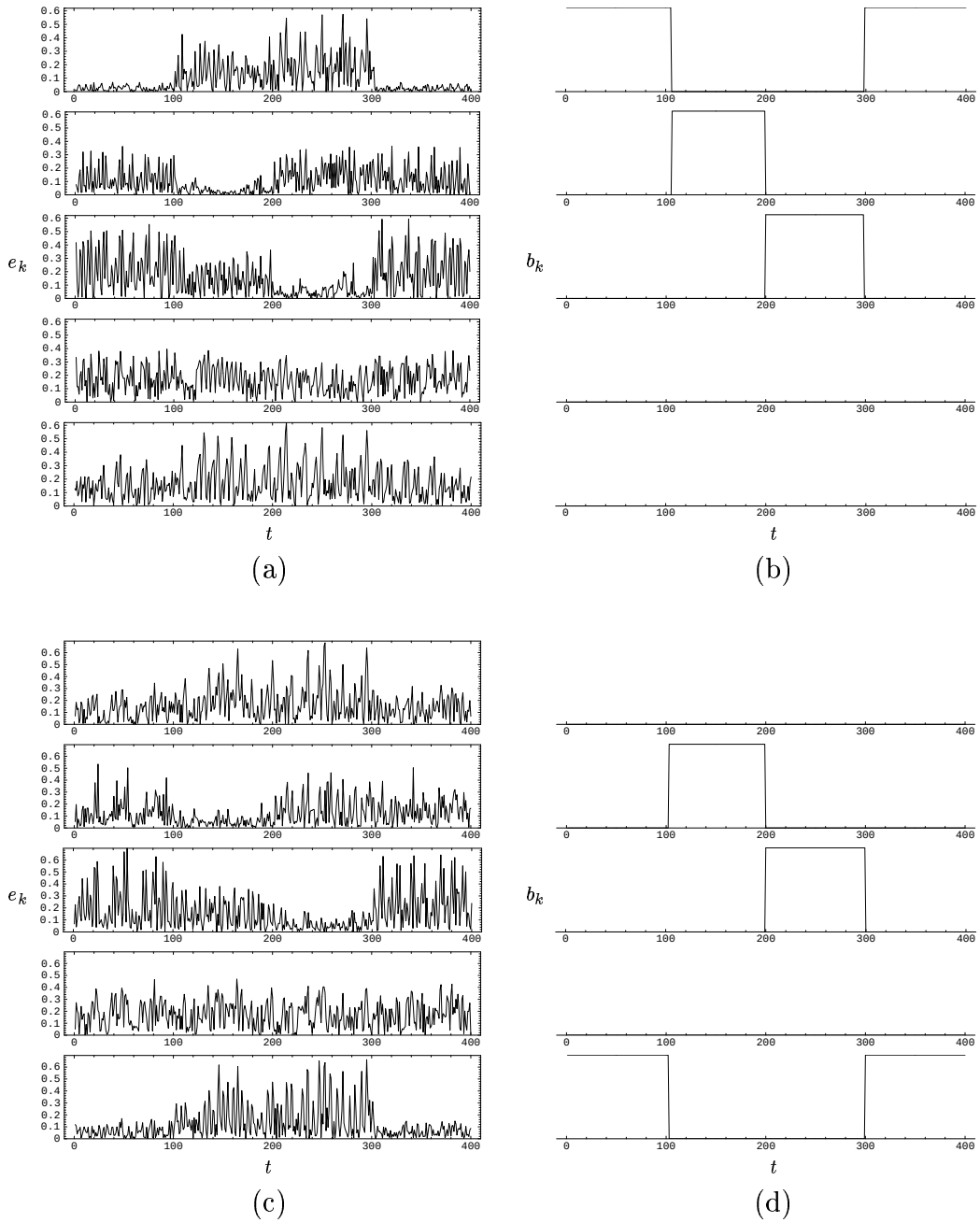


Abbildung 3.8: *Die Fehler $e_k(t)$ und die Verteilung der Gewinner, für die unverrauschte Zeitreihe (a)+(b), und für die verrauschte Zeitreihe (c)+(d). In beiden Fällen wird eine perfekte Segmentierung erreicht.*

Zeitreihe, (c) und (d) das Ergebnis für die verrauschte Zeitreihe. In beiden Fällen bleiben drei Prädiktoren übrig, die sich die Daten wie gewünscht untereinander aufteilen. Jeder Prädiktor hat sich auf eines der drei chaotischen Systeme spezialisiert. Das ist beachtlich, wenn man bedenkt, daß alle drei Systeme durch nur insgesamt 400 Daten repräsentiert werden. Darüber hinaus liefert das Verfahren die korrekte Segmentierung, und zwar auch für das verrauschte System. Dies zeigt, wie auch schon bei den chaotischen Abbildungen, daß das Verfahren auch bei verrauschten Daten stabil segmentieren kann.

3.2.3 Sprachdaten

Eine sehr interessante Anwendung unserer Methode ist die Segmentierung von natürlich gesprochener Sprache. Es besteht jedoch nicht die Absicht, mit unserem Ansatz tatsächlich professionelle Spracherkennung zu betreiben. Wir wollen aber versuchen, Sprachdaten zu segmentieren, um zu sehen, inwieweit es mit diesem Ansatz möglich ist, gemessene Zeitreihen aus *natürlichen* (realen) Systemen zu segmentieren. Im Falle der Sprachdaten handelt es sich darüber hinaus nicht mehr nur um skalare Größen, sondern um Frequenzspektren.

Das Segmentieren von Sprachdaten in einzelne Laute ist erfahrungsgemäß ein schwieriges Problem, da die Übergänge zwischen den Lauten in der Regel nicht eindeutig zu erkennen sind — weder von menschlichen noch von maschinellen Segmentierern [68]. In solchen Fällen werden die Segmentgrenzen dann mehr oder weniger willkürlich festgelegt. Wenn wir unser Verfahren auf Sprachdaten anwenden, gehen wir von der Annahme aus, daß der Sprache eine *schaltende* Dynamik zugrunde liegt. Ein ausgeprägtes Schaltverhalten kann man bei gesprochener Sprache im allgemeinen jedoch nicht erwarten. Bei fließend gesprochenen Sätzen ist eher mit kontinuierlichen Übergängen in der Dynamik zu rechnen; in unseren Experimenten trifft diese Annahme daher eher bei den gesprochenen Vokalfolgen als bei den Sätzen zu. So lange sich jedoch solche Übergänge lediglich über eine kleinere Zeitspanne hinstrecken, sollte unser Verfahren dennoch erfolgreich anwendbar sein.

Die in der Sprache enthaltenen Laute unterscheiden sich in ihrem Frequenzspektrum, das durch *short-time Fast-Fourier-Transformation* (im folgenden mit FFT abgekürzt) der gemessenen Amplitudenkurve berechnet werden kann. Die zeitliche Abfolge dieser Spektren ist der Gegenstand unserer Untersuchung. Die

Prädiktoren sollen anhand von d vorangegangenen Frequenzspektren das nächste Spektrum vorhersagen. Wir gehen somit weiter davon aus, daß für einzelne Laute hierbei ein deterministischer und stationärer Zusammenhang besteht, d.h. wir erwarten, daß jeder Laut durch bestimmte Spektrenfolgen charakterisiert werden kann.

FFT-Daten von gesprochenen Vokalen „AEIOU“

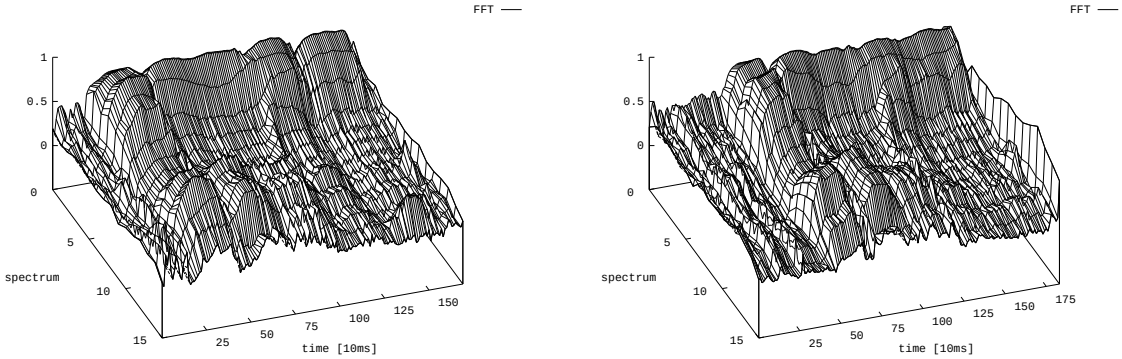
Die Sprachdaten für dieses Experiment bestehen aus Sequenzen von 16-dimensionalen mel-scale FFT-Spektren im zeitlichen Abstand von 10 ms (siehe dazu Abbildung 3.9). Diese Fourier-Spektren wurden aus kontinuierlich gesprochenen Vokalen „AEIOU“ errechnet, die mit einer *sampling*-Rate von 16 kHz aufgenommen wurden. Die insgesamt 20 Aufnahmen wurden von *einem* männlichen Sprecher gesprochen, sie unterscheiden sich aber dennoch stark durch unterschiedliche Betonung und Geschwindigkeit.[1]

Ziel des Experiments ist es, die korrekte Segmentierung einer solchen FFT-Zeitreihe zu erhalten. Jeder Prädiktor soll sich auf einen der Vokale spezialisieren oder aus dem Wettbewerb ausscheiden. Da in den Sequenzen auch Pausen enthalten sind (am Anfang oder am Ende), soll es auch einen Prädiktor geben, der sich auf solche Pausen spezialisiert.

Es wurden 8 RBF-Prädiktoren mit 30 Hidden-Units zum Segmentieren der Daten verwendet. Jeder Prädiktor soll anhand von 4 vorangegangenen FFT-Spektren das nächste Spektrum vorhersagen. Somit sind die Prädiktoren gewissermaßen mit einem Kurzzeit-Gedächtnis von 40 ms ausgestattet. Im Hinblick auf die Einbettungsparameter bedeutet dies, daß $\tau = 1$ und $d = 4$ zu wählen sind. Daraus ergibt sich ein 16×4 dimensionaler Eingabevektor. Jeder Prädiktor benötigt also 64 Input-Units und 16 Output-Units.

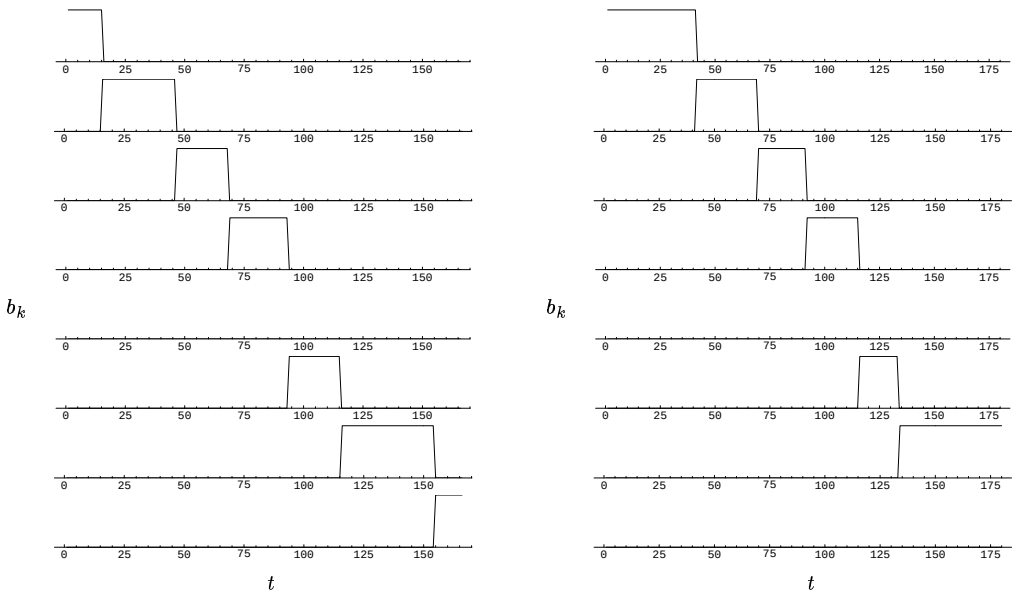
Das Wettbewerbslernen wurde über 20 Iterationen durchgeführt. Die Lernrate η wurde währenddessen von 0.05 auf 0.5 erhöht. Als Fehler $E_k(t)$ wurde der *gleitende Durchschnitt* berechnet, mit $\Delta = 0$ am Anfang und $\Delta = 10$ am Ende des Lernverfahrens.

[1] Wir möchten uns an dieser Stelle bei der Gruppe von Prof. Waibel bedanken, die uns die Aufnahmen ermöglicht hat.



(a) (b)

Abbildung 3.9: *Short-time FFT-Spektren von kontinuierlich gesprochenen Vokalen „AEIOU“: (a) Trainingssequenz, (b) Testsequenz.*



(a) (b)

Abbildung 3.10: *Verteilung der Gewinner nach* 20 *Lerndurchgängen für (a) die Trainingssequenz, (b) die Testsequenz.*

Das Ergebnis der Lernphase zeigt Abbildung 3.10(a). Fünf Prädiktoren repräsentieren jeweils einen Vokal, zwei Prädiktoren haben sich auf die Pausen am Anfang und am Ende der Sequenz spezialisiert und ein Prädiktor fiel aus dem Wettbewerb heraus. Die gefundenen Segmentationen wurden mit Hand-Segmentationen verglichen und es zeigte sich eine sehr gute Übereinstimmung. Um die Generalisierungsfähigkeit des erfolgreich trainierten Teams zu testen, wurden die 8 Netze als Prädiktoren für die Dynamik der übrigen 19 Sprachsequenzen eingesetzt. In 13 Fällen ergab sich dabei eine völlig korrekte Segmentierung der Daten entsprechend den darin enthaltenen Vokalen. Eine der Testsequenzen zeigt Abbildung 3.9(b), die zugehörige Segmentierung zeigt Abbildung 3.10(b). Man beachte, daß die Segmentierung der unbekannten Testsequenzen kein zusätzliches Training erfordert. Es zeigt sich also, daß mit unserem Lernverfahren tatsächlich die Vokalcharakteristik aller fünf Vokale gelernt wurde. Dieses Ergebnis wurde erreicht, obwohl nur ein einziger Datensatz zum Training verwendet wurde (Abbildung 3.9(a)). In den übrigen 6 Fällen enthielten die Segmentierungen jeweils einen kleinen Abschnitt, der falsch zugeordnet wurde. Der Grund dafür war die ähnliche Aussprache von E und I, bzw. O und U, was z. B. dazu führte, daß ein E-Prädiktor auch einen Teil des I-Abschnitts zugeteilt bekam.

Es war hier jedoch nicht die Absicht, das Problem der Vokalerkennung zu lösen. Man kann Vokale auch einfach anhand der Formanten unterscheiden (siehe z. B. [68]). Es wurde jedoch gezeigt, daß unser Verfahren in der Lage ist, auch Zeitreihen von Fourier-Spektren richtig zu segmentieren. Dabei handelte es sich um Fourier-Spektren realer Sprache, bei denen die Segmentgrenzen und die einzelnen Dynamiken nicht künstlich erzeugt worden sind, wie etwa beim Mackey-Glass System in Abschnitt 3.2.2. Es soll hier die vielseitige Anwendbarkeit unseres Ansatzes deutlich werden.

FFT-Daten von kontinuierlich gesprochenen Sätzen

Die mit den Vokaldaten gemachten Erfahrungen ermunterten uns dazu, den nächsten Schritt zu probieren: das Segmentieren ganzer, kontinuierlich gesprochener Sätze. Dazu untersuchten wir *short-time* FFT-Daten, die uns die Gruppe von Prof. Waibel zur Verfügung gestellt hatte. Es handelt sich dabei um eine Anzahl von deutschen Sätzen (wie z.B. „Am blauen Himmel ziehen die Wolken“), die in derselben Weise wie die Vokaldaten erzeugt und aufbereitet wurden, jedoch mit einem anderen Sprecher. Die Sätze wurden deutlich, aber ohne Pausen zwischen den Wörtern ausgesprochen. Die daraus erstellten Fourier-Sequenzen

wurden dann zunächst von Hand segmentiert und anschließend noch einmal maschinell nachgelabelt. Das Ergebnis zeigt Abbildung 3.11 für die beiden Sätze, die wir für unsere Experimente ausgewählt haben. Somit war es möglich, unsere Simulationsergebnisse mit der vorgegebenen Segmentierung zu vergleichen.

Aus Abbildung 3.11 ist ersichtlich, daß die einzelnen Laute nur über wenige *frames* (FFT-Spektren) andauern. Das heißt, daß die Sprachdynamik sich (verständlicherweise) sehr schnell ändert – im Gegensatz zu den Vokaldaten. Für unser Lernverfahren bedeutet dies wiederum, daß der Tiefpaßfilter $E_k(t)$, der den Prädiktoren Wettbewerbsvorteile auf zeitlich benachbarten Daten verschafft, nur sehr begrenzt zum Einsatz kommen kann. Denn die Verwendung des Filters ist nur dann sinnvoll, wenn davon ausgegangen werden kann, daß jede Einzeldynamik eine gewisse Zeitlang stationär bleibt, benachbarte Daten also mit hoher Wahrscheinlichkeit derselben Dynamik unterliegen. Diese Voraussetzung trifft hier nur in sehr begrenztem Umfang zu, was die Ausgangssituation für unser Verfahren beeinträchtigt.

Desweiteren sollte man sich vor Augen führen, daß in den ca. 230 *frames* eines Satzes über 20 verschiedene Laute enthalten sind. Ein Laut besteht dabei meist aus nur 4 bis 11 FFT-Spektren. Jeden dieser Laute allein anhand der 230 *frames* als eine eigenständige Dynamik zu identifizieren ist also äußerst schwierig. Um die Sätze in so viele Laute, wie vorgegeben, zerlegen zu können, wählten wir ein Ensemble von 60 Prädiktoren. Damit entfallen auf jeden Prädiktor am Anfang des Lernverfahrens durchschnittlich vier *frames* zum Trainieren, was etwa der minimalen Dauer eines Lautes entspricht. Auf diese Weise soll erreicht werden, daß die meisten Prädiktoren im *ersten* Lerndurchgang hauptsächlich Daten *eines* Lautes erhalten.

Die Prädiktoren sollen anhand der zwei vorangegangenen FFT-Spektren das nächste Spektrum vorhersagen: Drei aufeinanderfolgende Spektren werden meist zum selben Laut gehören, ein viel längeres „Gedächtnis“ wäre schon nicht mehr hilfreich. Da auf jeden Laut ohnehin nur sehr wenige *frames* (d.h. Trainingsdaten) entfallen, benötigt jeder Prädiktor auch nur wenige Hidden-Units, wenn er nur einen einzigen Laut, nicht aber gleich mehrere erkennen soll. Acht bis zehn Hidden-Units haben sich daher als sinnvoll erwiesen.

Das Lernverfahren wurde über $T = 20$ Iterationen durchgeführt ($\eta = 0.5, \Delta(0) = 1, \Delta(T) = 4$). Abbildung 3.12 zeigt das Ergebnis für die beiden Sätze „Am blauen Himmel ziehen die Wolken“ und „Über die Felder weht ein Wind“. Es ist ange-

geben, welchen Prädiktoren die Daten zugeteilt wurden. Am Ende blieben von den 60 Netzen 23 bzw. 25 übrig.

Die durch das Lernverfahren zugeteilten Abschnitte wurden zum Vergleich den vorgegebenen Labels aus Abb. 3.11 zugeordnet. Bei beiden Sätzen ergibt sich eine stärkere Segmentierung als beim Labeln von Hand, das Verhältnis beträgt etwa 3 zu 2. Aber die Segmentgrenzen der Vorgabe lassen sich dennoch wiederfinden. Es ergibt sich eine mittlere Abweichung von ca. 2 *frames*. Eine Bewertung dieses Resultats ist jedoch schwierig, denn auch eine Segmentierung von Hand stellt nicht unbedingt eine optimale Einteilung dar. Eine Segmentierung kann erst dann als optimal bezeichnet werden, wenn sie eine (weitgehend) fehlerfreie Weiterverarbeitung der Daten in einem Spracherkennungssystem ermöglicht. Ob dies eher mit der vorgegebenen Einteilung von Hand oder unserer automatischen Methode der Fall ist, bleibt an dieser Stelle offen, da wir kein Spracherkennungssystem bauen, sondern hier lediglich die Einsatzmöglichkeiten unseres Segmentierungsansatzes auf realen Daten untersuchen und aufzeigen wollen.

Bei genauer Betrachtung von Abbildung 3.12 ist festzustellen, daß sich die Prädiktoren offenbar nicht derart auf bestimmte Laute spezialisiert haben, daß für einen Laut immmer derselbe Prädiktor zuständig wäre. Beispielsweise hat Netz 17 in Satz 1 das „L“ im Wort BLAUEN zugeteilt bekommen, nicht jedoch das „L“ in HIMMEL (Netz 45) und WOLKEN (Netz 14). Das mag daran gelegen haben, daß das „L“ jedesmal etwas anders ausgesprochen wurde. Auch bei anderen Netzen zeigt sich, daß die erhoffte *Identifizierung* der Laute nicht erreicht wurde. Dies ist nicht so verwunderlich: Jedem Netz stehen am Ende des Lernverfahrens durchschnittlich nur 10 *frames* zur Verfügung, die jeweils die zuletzt gültige Trainingsmenge darstellen. Die Trainingsmengen der Netze sind hier also extrem klein.

Trotz der schwierigen Problemstellung hat sich jedoch gezeigt, daß unser Verfahren grundsätzlich in der Lage ist, Sprachdaten zu *segmentieren*. Wenn man beabsichtigt, das Verfahren tatsächlich in ein Spracherkennungssystem zu integrieren, sollte man versuchen, das Verfahren dahingehend zu erweitern, *vortrainierte* Prädiktoren einzusetzen, die sich bereits auf einzelne Laute spezialisiert haben. Interessant wäre in dem Zusammenhang auch die Verwendung anderer Prädiktoren, die speziell für Sprachdaten geeignet sind, z. B. durch die Einbindung von Hidden-Markov-Modellen [57]. Diese werden bereits erfolgreich in Spracherkennungssystemen eingesetzt.

```
            Satz 1

AM BLAUEN HIMMEL ZIEHEN DIE WOLKEN
230 frames
6 words
AM       44      61      3
           ?   A   M
           3   4  11
BLAUEN   62      99      5
           B   L  AU   E   N
           4   6  17   4   7
HIMMEL  100     128      4
           H   I   M   L
           6   4   7  11
ZIEHEN  129     161      4
           T   S  IE   N
           6   7  12   8
DIE     162     169      2
           D  IE
           3   5
WOLKEN  170     229      5
           V   O   L   K  NG     SI
           7   9  10  11  19     4
```

```
            Satz 2

UEBER DIE FELDER WEHT EIN WIND
229 frames
6 words
UEBER   62      77      3
         UEH   V  AE
           5   5   6
DIE     78      90      2
           D  IE
           5   8
FELDER  91     137      5
           F  AE   L   D  AE
          14  10  10   4   9
WEHT   138     159      3
           V  EH   T
           7   8   7
EIN    160     177      3
          AI   I   N
           5   6   7
WIND   178     228      4
           V   I   N  TH     SI
           4  10  17  16     4
```

Abbildung 3.11: *Zwei handgelabelte Sätze. Aus den Mikrophon-Aufnahmen wurden Sequenzen von* 230 *bzw.* 229 *short-time Fourier-Spektren (frames) errechnet. Anschließend wurden die Laute eines Satzes den durchnumerierten frames von Hand zugeordnet. So erkennt man, durch welche frames ein Laut repräsentiert wird, z. B. ist das „W“ im Wort WOLKEN in den sieben Frames 170–176 enthalten.*

Satz 1

net	from	to	label
14	0	36	-
55	37	39	-
45	40	40	-
21	41	47	-
12	48	52	A
55	53	60	M
29	61	65	B
17	66	74	L
18	75	83	A
19	84	89	U
22	90	92	E
24	93	100	N
26	101	106	H
27	107	113	I
29	114	119	M
45	120	122	L
32	123	133	LT
33	134	137	S
35	138	146	SI
36	147	150	I
38	151	157	IN
35	158	163	ND
36	164	165	D
41	166	168	I
42	169	171	I
17	172	178	W
45	179	190	O
14	191	198	L
55	199	201	K
5	202	203	K
54	204	204	K
52	205	206	K
54	207	215	N
55	216	225	G
14	226	239	-

Satz 2

net	from	to	label
1	0	22	-
59	23	41	-
1	42	54	-
59	55	56	-
14	57	62	-
16	63	64	UE
17	65	72	B
26	73	75	AE
20	76	83	D
36	84	84	IE
22	85	90	IE
23	91	98	F
17	99	109	AE
27	110	123	L
31	124	131	D
40	132	135	AE
33	136	137	AE
17	138	145	W
36	146	148	EH
37	149	151	EH
55	152	152	EH
31	153	160	T
32	161	164	A
40	165	171	I
42	172	174	N
36	175	176	N
35	177	183	W
45	184	190	I
48	191	193	I
49	194	197	N
50	198	199	N
1	200	207	N
52	208	212	T
53	213	217	T
55	218	220	H
1	221	237	-
59	238	239	-

Abbildung 3.12: *Die Verteilung der Gewinner auf den short-time FFT-Sequenzen beider Sätze. Die durch das Lernverfahren zugeteilten Abschnitte wurden zum Vergleich den vorgegebenen Labels zugeordnet. Die zuvor von Hand bestimmten Segmentgrenzen lassen sich, bis auf kleine Abweichungen, in der unüberwacht erlernten Segmentierung wiederfinden.*

3.2.4 Physiologische Daten

Eine weitere interessante Anwendung unseres Verfahrens ergab sich aus der Untersuchung von Zeitreihen lokaler Feldpotentiale (LFP) aus dem visuellen Kortex der Katze [16], welche im Max-Planck-Institut für Hirnforschung in Frankfurt gemessen wurden. Abbildung 3.13 zeigt einen Ausschnitt aus den Daten. Die Vermutung ist, daß sich in der Zeitreihe stochastisches und oszillatorisches Verhalten abwechseln [51]. Man geht davon aus, daß zwischen diesen zwei unterschiedlichen Dynamiken immer wieder hin und her geschaltet wird. Es war daher zu untersuchen, ob unser Segmentierungsalgorithmus ein Resultat liefert, das diese Erwartung bestätigt.

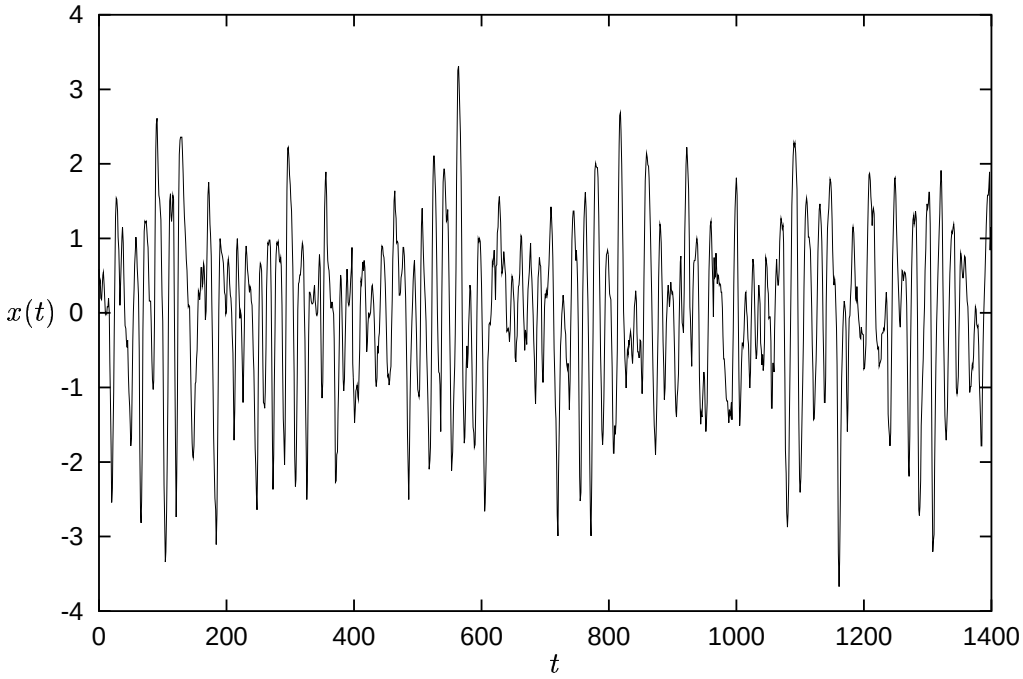


Abbildung 3.13: *Ausschnitt aus den Feldpotentialdaten (LFP).*

Wir verwendeten dazu ein Team von vier Prädiktoren mit jeweils zehn Hidden-Units. Für die Einbettungsparameter τ und d wurden verschiedene Werte eingesetzt. Abbildung 3.14 zeigt das Ergebnis für $\tau = 5$ und $d = 5$: Tatsächlich teilen sich hier im wesentlichen zwei Prädiktoren die insgesamt 1900 Daten. Das ist beachtlich, denn das Zeitfenster des Tiefpaßfilters war während der gesamten Trainingsphase ($T = 20$ Iterationen, $\eta(0) = 0.1$, $\eta(T) = 0.35$) nur sehr klein (Δ

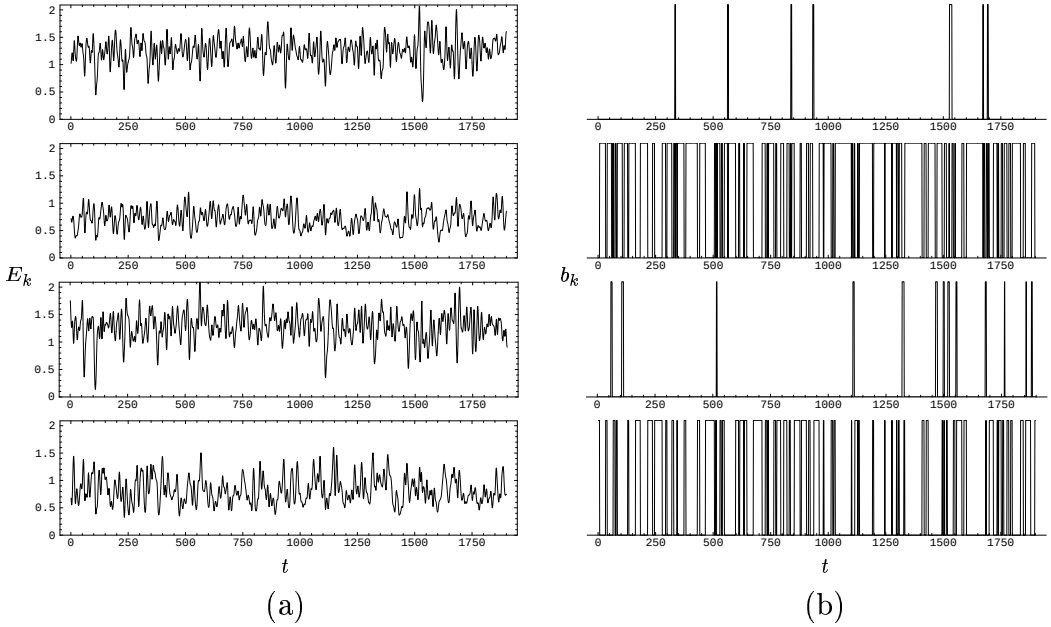
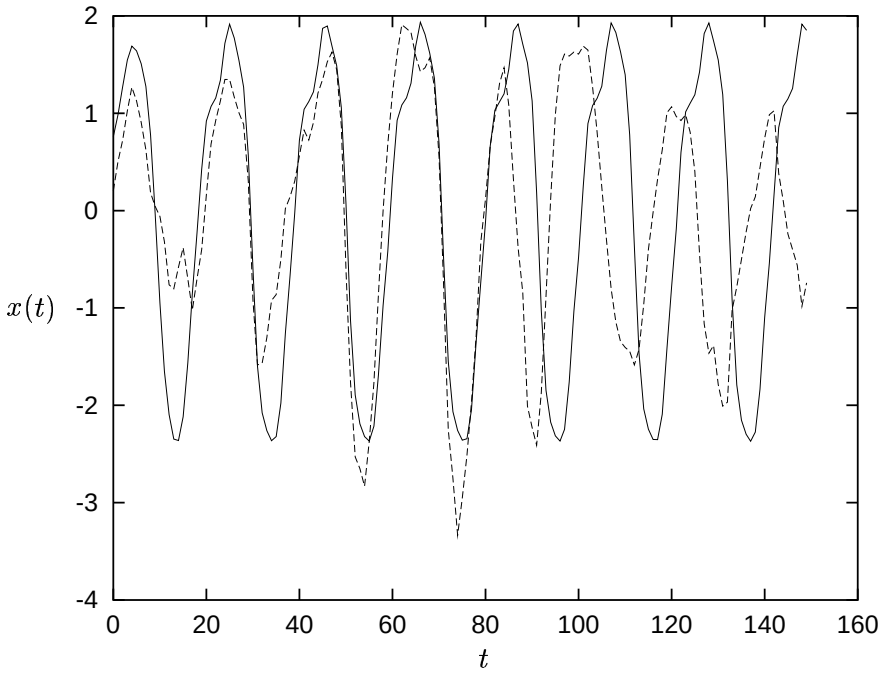


Abbildung 3.14: *Ergebnis des Lernverfahrens für die LFP-Daten. (a) Die Fehler $E_k(t)$ der 4 Netze und (b) die Verteilung der Gewinner, jeweils am Ende der Trainingsphase. Im wesentlichen teilen sich zwei Netze die gesamten Daten. Die beiden anderen Netze liefern im Durchschnitt deutlich höhere Fehler.*

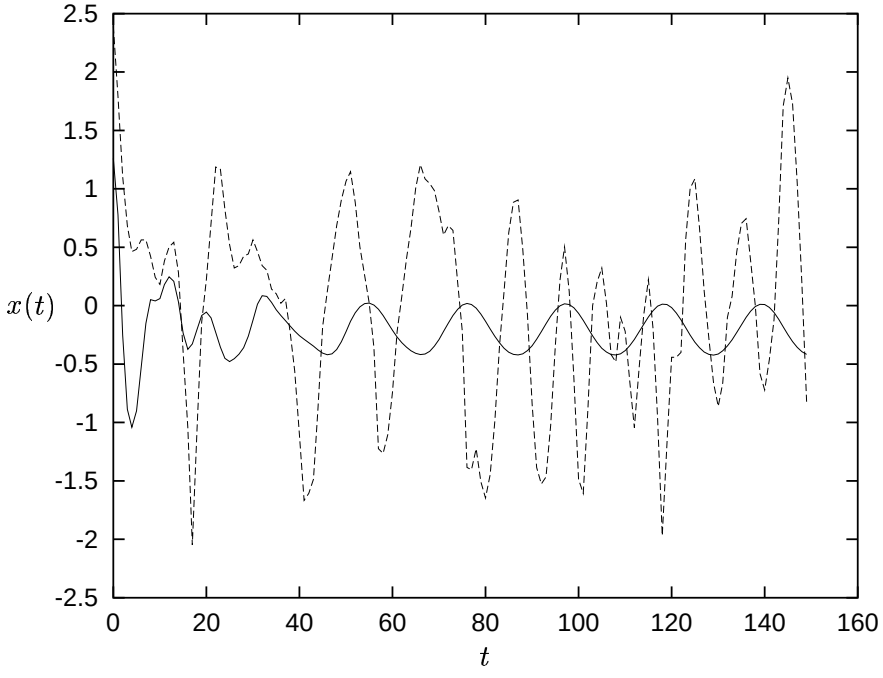
wurde von 0 bis 4 erhöht), so daß der Verdrängungseffekt für die Prädiktoren nur gering war. Trotzdem erhalten zwei der vier Prädiktoren am Ende kaum noch Daten. Ein wichtiger Punkt bei diesem Experiment: nur die ersten 1000 Daten wurden tatsächlich zum Trainieren verwendet, die restlichen 900 Daten sind Testdaten. In Abbildung 3.14(b) zeigt sich jedoch kein großer Unterschied: auch die Testdaten werden im wesentlichen an zwei Prädiktoren verteilt.

Dieses Ergebnis ist ein Anzeichen dafür, daß tatsächlich *zwei* unterschiedliche Dynamiken in der Zeitreihe vorkommen. Die beiden Gewinner-Prädiktoren wurden dann auf zwei Punkte der Zeitreihe aufgesetzt, einer auf eine Stelle mit oszillatorischem Verhalten und der andere auf eine Stelle mit stochastischem Verhalten. Beide Prädiktoren wurden von diesen Startpunkten aus iteriert, das heißt die vorhergesagten Daten der Prädiktoren werden wieder auf die Prädiktoren selbst zurückgekoppelt, so daß sie die weiteren Vorhersagen auf ihre zuvor gemachten Vorhersagen basieren und nicht auf die tatsächlichen Daten. Das Ergebnis zeigt Abb. 3.15: Der eine Prädiktor reproduziert sehr schön die Grundfrequenz und

Amplitude der Oszillationen. Der andere Prädiktor schwingt sich dagegen sehr schnell auf den Mittelwert des vermutlich stochastischen Datenabschnitts ein und es gelingt ihm im Gegensatz zu dem anderen Prädiktor nicht einmal im Ansatz, dem Verlauf des Signals zu folgen. Dieses Verhalten – das Vorhersagen des Mittelwerts – ist in der Tat zu erwarten, wenn der Signalverlauf nicht vorhersagbar ist. Die Vermutung, daß den Hirndaten zwei unterschiedliche Dynamiken zugrunde liegen, konnte somit erhärtet werden.



(a)



(b)

Abbildung 3.15: *Vergleich der iterierten Vorhersagen der beiden Gewinner-Prädiktoren (durchgezogene Linien) mit dem tatsächlichen Signalverlauf (gestrichelte Linien): (a) für den Prädiktor des oszillatorischen Abschnitts und (b) für den Prädiktor des stochastischen Abschnitts. Letzterer kann die Dynamik nicht reproduzieren.*

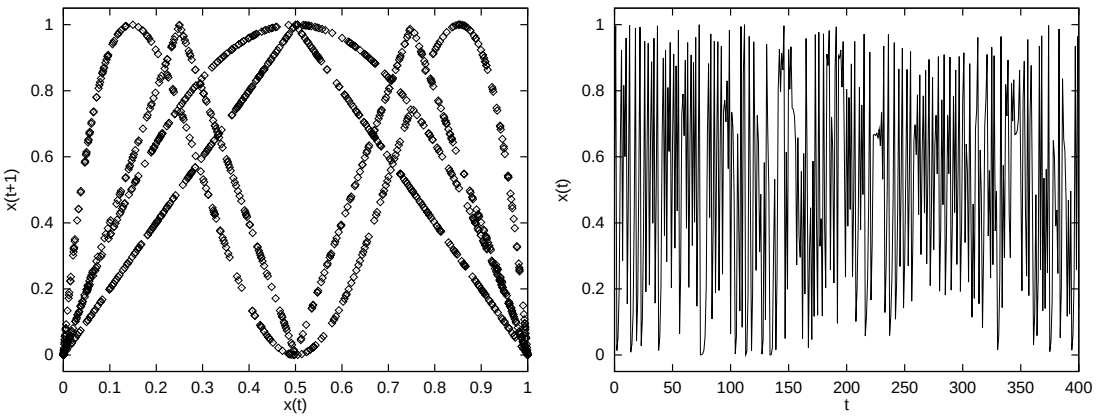
Kapitel 4

Theoretische Fundierung der Methode

In diesem Kapitel wird die Methode der *Competing Experts*, die im vorigen Kapitel intuitiv konstruiert wurde, mit Mitteln der Wahrscheinlichkeitsrechnung motiviert. Die grundlegenden Ideen des Ansatzes sollen dazu außerdem an einem Beispiel für schaltende Dynamik illustriert werden. Betrachtet wird dazu eine Zeitreihe $\{x(t)\}$, die von vier chaotischen Abbildungen f_l, $l = 1, 2, 3, 4$, durch die Rekursion $x(t+1) = f_l(x(t))$ erzeugt wird (mit $x(0) = 0.5289$). Alle 100 Zeitschritte schaltet das System in einen anderen Operationsmodus l und eine andere Funktion f_l wird gewählt, um die nächsten 100 Werte der Zeitreihe zu erzeugen. Die vier verwendeten Funktionen sind (siehe Abb.4.1)

$f_1(x) =$	$4x(1-x)\,,\ x \in [0,1]$	(logistic map)
$f_2(x) =$	$f_1(f_1(x))$	(double logistic map)
$f_3(x) =$	$2x,$ für $x \in [0,.5)$ und $2(1-x),$ für $x \in [.5,1]$	(tent map)
$f_4(x) =$	$f_3(f_3(x))$	(double tent map)

Wie in Abb. 4.1(a) zu sehen ist, kann man zu einem Zeitpunkt t die jeweils vorliegende Dynamik l und damit auch $x(t+1)$ nicht allein aus dem Wert $x(t)$ bestimmen, da alle vier Einzeldynamiken auf demselben Interval, $x(t) \in [0,1]$, operieren. Eine Möglichkeit, die vier Dynamiken zu unterscheiden, ist es, eine größere Einbettung zu wählen, d.h. mehrere zeitverzögerte Koordinaten $(x(t), x(t-1), \ldots, x(t-(d-1)))$ zu verwenden (Packard *et al.* [50]). Die größere Einbettung ermöglicht zwar prinzipiell die Vorhersage von $x(t+1)$ für die



(a) (b)

Abbildung 4.1: *(a) Die ersten 1200 Daten der Zeitreihe* $\{x(t)\}$*, 300 Daten für jede der vier Abbildungen* f_l*. Dargestellt ist* $x(t+1)$ *versus* $x(t)$*, wodurch die vier Funktionen leicht zu erkennen sind. (b) Die ersten 400 Daten* $x(t)$ *als Zeitreihe dargestellt. Die vier Einzeldynamiken lassen sich in dieser Darstellung nicht mehr unterscheiden.*

gesamte Zeitreihe durch ein *einziges* neuronales Netz, allerdings ohne explizite Erkennung der Moden l. Der Übergang in einen solchen höherdimensionalen Eingangsdaten-Raum erfordert jedoch eine erheblich kompliziertere Vorhersagefunktion für $x(t+1)$ als zur Modellierung und Vorhersage der Einzeldynamiken f_l erforderlich wäre (Abb. 4.2).

Auf diese Weise werden also weder die Einzeldynamiken vernünftig repräsentiert, noch wird die Struktur des dynamischen System offengelegt. Eine adäquate Repräsentation der Zeitreihe sollte daher aus einer Segmentation und Identifikation der Einzeldynamiken bestehen. Jede Einzeldynamik wird dazu von einem neuronalen Netz (Prädiktor) repräsentiert, das die entsprechende Dynamik vorherzusagen lernt. Es gilt also, ein Ensemble von Prädiktoren so zu trainieren, daß sie sich auf die Vorhersage derjenigen Daten der Zeitreihe spezialisieren, die zu jeweils einer Einzeldynamik gehören.

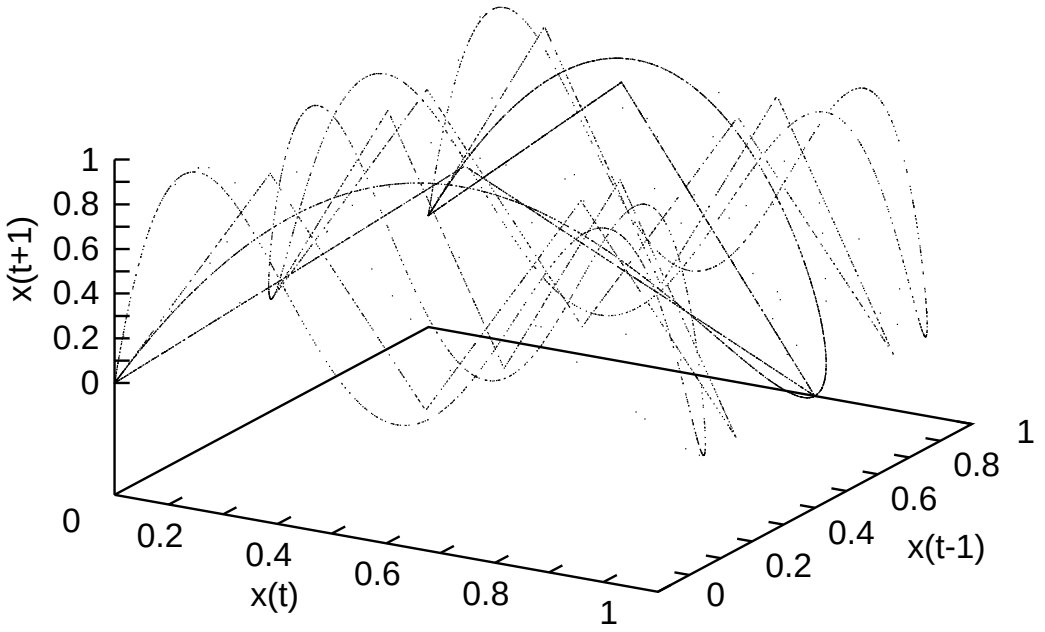


Abbildung 4.2: *Das zu erlernende Vorhersageproblem für $x(t+1)$ (Ordinatenachse) aus einer zweidimensionalen Einbettung $(x(t), x(t-1))$ des aus vier chaotischen Abbildungen bestehenden Systems. Die Zuordnung ist an einigen Schnittpunkten nicht eindeutig. Die Menge der unentscheidbaren Punkte würde bei verrauschten Daten (typisch bei realen Meßdaten) gravierend zunehmen. Diese im Grunde also noch zu kleine Einbettung, zeigt jedoch bereits die enorme Zunahme der Komplexität des Vorhersageproblems verglichen mit dem Problem, die vier eindimensionalen Funktionen einzeln zu approximieren.*

4.1 Maximum Likelihood

Gegeben sei eine Menge von Prädiktoren f_i, $i = 1, ..., n$. Unter der Annahme einer normalverteilten Abweichung zwischen beobachteter und vorhergesagter Größe, ist die Wahrscheinlichkeit, daß ein einzelner Prädiktor i einen gegebenen Datenpunkt $x(t)$ erzeugt hat, gegeben durch

$$p(x(t) \mid i) = Ke^{-\beta(x(t)-f_i)^2} \tag{4.1}$$

wobei K der Normalisierungsterm der Gaußverteilung ist. Da wir weiter annehmen, daß für jeden Datenpunkt immer genau einer der Experten zuständig ist, gilt für die Wahrscheinlichkeit, daß ein Datenpunkt $x(t)$ von dem Ensemble $\{i\}$ erzeugt wurde,

$$p(x(t) \mid \{i\}) = \sum_i p(x(t) \mid i)p(i). \tag{4.2}$$

Für die *a priori* Wahrscheinlichkeit der Experten wird

$$p(i) = 1/n \tag{4.3}$$

angenommen, d.h. *a priori* sollen alle Experten gleich wahrscheinlich sein. Nun soll die Wahrscheinlichkeit, daß das Ensemble die gegeben Daten produziert hat, maximiert werden (*Maximum Likelihood*), indem die Parameter der Experten geeignet eingestellt werden. Ein (lokales) Maximum kann durch ein Gradientenverfahren gefunden werden. Hierzu wird die *log-likelihood* $\log L = \log(p(x(t) \mid \{i\}))$ maximiert. Für die erste Ableitung bezüglich eines Experten f_i ergibt sich damit

$$\frac{\partial \log L}{\partial f_i} \propto \left[\frac{e^{-\beta(x(t)-f_i)^2}}{\sum_j e^{-\beta(x(t)-f_j)^2}}\right](x(t) - f_i). \tag{4.4}$$

Gemäß Bayes' Theorem ist der Term in eckigen Klammern die *a posteriori* Wahrscheinlichkeit $p(i \mid x(t))$, daß Experte i die richtige Wahl für den gegebenen Datenpunkt $x(t)$ ist. Darum kann man auch einfach

$$\frac{\partial \log L}{\partial f_i} \propto p(i \mid x(t))(x(t) - f_i) \tag{4.5}$$

schreiben. Die gewonnene Lernregel gewichtet also den individuellen Vorhersagefehler eines Experten mit der Wahrscheinlichkeit seiner Zuständigkeit im Vergleich zu den anderen Experten.

Es zeigt sich jedoch, daß die Lernregel in Gl.(4.4) in den meisten Fällen nicht ausreichend ist, um die richtige Segmentierung und damit auch einen niedrigen

Vorhersagefehler zu erhalten. Das liegt daran, daß bisher keine konkrete Annahme über das Schaltverhalten des dynamischen Systems gemacht wurde, die in die Lernregel eingehen würde. Ohne eine solche Annahme kommt eine Vielzahl von dynamischen Systemen als Erzeuger einer gegebenen Zeitreihe in Frage. Ein extremes Beispiel wäre ein System, bei dem jeder Wert der Zeitreihe von einer anderen, neuen Dynamik erzeugt wird.

Bisher lassen sich die Möglichkeiten, das zugrundeliegende dynamische System zu modellieren, zwar bereits einschränken durch (a) die Anzahl der verwendeten Experten und (b) die Klasse der Funktionen, die ein Experte repäsentieren kann (z. B. lineare Funktionen). Trotzdem ist es möglich, eine Zeitreihe durch prinzipiell unterschiedliche Lösungsansätze zu modellieren und dabei eventuell in lokalen Minima der Fehlerfunktion steckenzubleiben, wie in Abb.4.3(a) am Beispiel der chaotischen Abbildungen illustriert wird. Dazu wurden sechs Moody-Darken-Prädiktoren $f_i, i = 1, ..., 6$, mit 20 RBF-Zentren auf den ersten 1200 Daten der Zeitreihe gemäß Gl.(4.4) trainiert. Das Lernverfahren liefert zwar ein Modell für die Dynamik der Zeitreihe, es ist jedoch ein prinzipiell anderes als das Erzeugersystem.

Die Menge der möglichen Lösungen muß also auf die eine, gewünschte Lösung reduziert werden. Diese ist dadurch charakterisiert, daß das dynamische System *selten schaltet*. Man muß daher die Annahme seltenen Schaltens explizit in der Lernregel berücksichtigen, wenn man ausschließlich zu einer solchen Lösung gelangen will. Wie im folgenden gezeigt wird, führt dies zu der Verwendung eines Tiefpaß-Filters.

4.2 Der Tiefpaß-Filter

Die Wahrscheinlichkeit, daß eine Datensequenz $\sigma_t^\Delta = (x(t-\Delta), \ldots, x(t+\Delta))$ der Zeitreihe $\{x(t)\}$ durch eine gegebene Sequenz $\vec{s}_t^\Delta = (l(t-\Delta), \ldots, l(t+\Delta))$ von Prädiktoren $f_{l(t)}$ erzeugt wurde, ist gegeben durch

$$p(\sigma_t^\Delta \mid \vec{s}_t^\Delta) = \prod_{t'=t-\Delta}^{t+\Delta} p(x(t') \mid l(t')), \tag{4.6}$$

wobei

$$p(x(t) \mid l(t)) = K e^{-\beta(x(t)-f_{l(t)})^2} \tag{4.7}$$

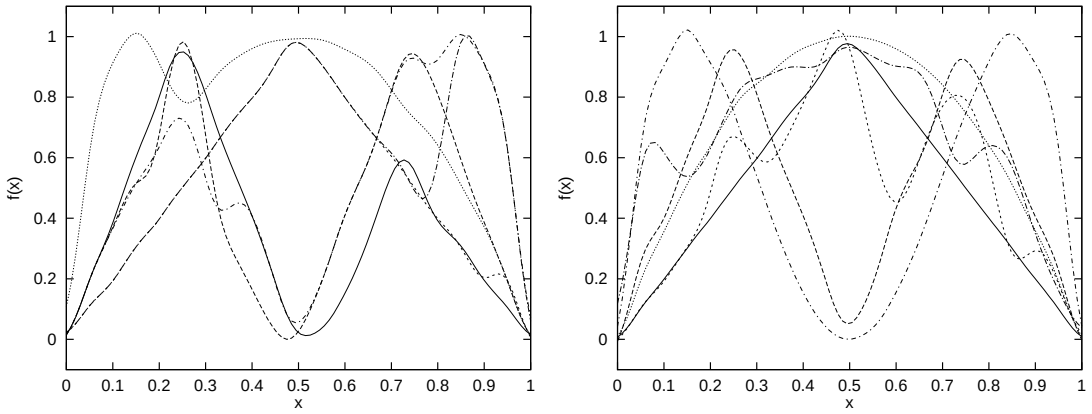


Abbildung 4.3: *(a) Ergebnis des Lernverfahrens für die vier chaotischen Abbildungen ohne Berücksichtigung der Information, daß zeitlich benachbarte Daten mit hoher Wahrscheinlichkeit zur selben Dynamik gehören. Die Prädiktoren erlernen Funktionen, die beliebig aus den Zweigen der vier zu erlernenden Funktionen f_l zusammengesetzt sein können (vgl. Abb. 4.1(a)). Denn der Vorhersagefehler läßt sich auf verschiedene Arten minimieren. Daß das gesuchte dynamische System selten schaltet, wird nicht berücksichtigt. (b) Die einzige Lösung, die ein seltenes Schalten zwischen den Prädiktoren ermöglicht, besteht darin, daß jede Funktion f_l vollständig von je einem Prädiktor approximiert wird. Überzählige Prädiktoren werden dann nicht benötigt, sie können daher beliebige Funktionen repräsentieren.*

entsprechend Gl.(4.1) gilt. Umgekehrt ist die Wahrscheinlichkeit, daß eine bestimmte Sequenz von Prädiktoren für die Erzeugung einer gegebenen Datensequenz zuständig war, gemäß Bayes' Theorem gegeben durch

$$p(\vec{s}_t^{\,\Delta} \mid \sigma_t^\Delta) = \frac{p(\sigma_t^\Delta \mid \vec{s}_t^{\,\Delta}) p(\vec{s}_t^{\,\Delta})}{\sum_{\vec{s}'^{\,\Delta}} p(\sigma_t^\Delta \mid \vec{s}'^{\,\Delta}) p(\vec{s}'^{\,\Delta})}, \tag{4.8}$$

wobei die Summe über alle möglichen Sequenzen $\vec{s}'^{\,\Delta}$ läuft. Unter der Annahme, daß die Dynamik selten schaltet, haben Sequenzen, die ein Schalten der Dynamik enthalten, folglich eine sehr geringe Wahrscheinlichkeit. Sie werden daher im folgenden vernachlässigt, d.h. es wird $p(\vec{s}^{\,\Delta}) = 0$ angenommen, falls nicht alle Komponenten der Sequenz gleich sind. Die verbleibenden n möglichen Sequenzen $\vec{s}^{\,\Delta} = (i, \ldots, i)$, $i = 1, \ldots, n$, die nun jeweils für einen einzelnen Prädiktor stehen, werden wiederum als *a priori* gleichwahrscheinlich betrachtet, d.h. $p(\vec{s}^{\,\Delta}) = 1/n$. Die *a posteriori* Wahrscheinlichkeit, daß ein Prädiktor i eine gegebene Sequenz σ_t^Δ erzeugt hat, vereinfacht sich damit aus Gl.(4.8) zu

$$p(i \mid \sigma_t^\Delta) = \frac{p(\sigma_t^\Delta \mid i)}{\sum_{j=1}^n p(\sigma_t^\Delta \mid j)}. \tag{4.9}$$

Mit Gl.(4.6) erhält man

$$p(i \mid \sigma_t^\Delta) = \frac{\prod_{t'=t-\Delta}^{t+\Delta} p(x(t') \mid i)}{\sum_{j=1}^n \prod_{t'=t-\Delta}^{t+\Delta} p(x(t') \mid j)} \tag{4.10}$$

und mit Gl.(4.7)

$$p(i \mid \sigma_t^\Delta) = \frac{e^{-\beta \sum_{t'=t-\Delta}^{t+\Delta} (x(t')-f_i)^2}}{\sum_{j=1}^n e^{-\beta \sum_{t'=t-\Delta}^{t+\Delta} (x(t')-f_j)^2}}. \tag{4.11}$$

Ersetzt man nun in der Lernregel in Gl.(4.5) die *a posteriori* Wahrscheinlichkeit, daß ein Prädiktor einen *einzelnen* Datenpunkt $x(t)$ erzeugt hat, $p(i \mid x(t))$, durch die Wahrscheinlichkeit, daß er auch alle Daten in der zeitlichen Nachbarschaft $[t - \Delta, t + \Delta]$ erzeugt hat, $p(i \mid \sigma_t^\Delta)$, dann werden die Prädiktoren nicht mehr auf die Vorhersage zeitlich beliebig verteilter Datenpunkte optimiert, sondern auf die Vorhersage zusammenhängender Sequenzen. Denn nur derjenige Prädiktor kann sich mit dem größten Lernschritt an einen Datenpunkt adaptieren, der auch in der zeitlichen Nachbarschaft des Datenpunktes bessere Vorhersagen als die anderen Prädiktoren liefert. Auf diese Weise werden die Daten der Zeitreihe überhaupt erst in einen zeitlichen Zusammenhang gebracht. Die Berücksichtigung dieses zeitlichen Zusammenhangs ist, wie bereits dargestellt wurde, notwendig, um den vorhandenen Spielraum bei der Zuweisung der Daten an die Prädiktoren

einzuschränken. In diesem Fall wird, wie gesagt, das Auffinden von Systemen mit niedriger Schaltrate begünstigt.

Die Verwendung von $p(i \mid \sigma_t^\Delta)$ anstelle von $p(i \mid x(t))$ führt lediglich dazu, daß die Einzel-Vorhersagefehler

$$e_i^t = (x(t) - f_i)^2 \tag{4.12}$$

in der Lernregel Gl.(4.4) durch eine gleitende Summe (Tiefpaß)

$$E_i^t = \sum_{t'=t-\Delta}^{t+\Delta} e_i^{t'} \tag{4.13}$$

ersetzt werden müssen. Unter Verwendung von Gl. (4.13) läßt sich $p(i \mid \sigma_t^\Delta)$ nun auch einfach in der Form

$$p(i \mid \sigma_t^\Delta) = \frac{e^{-\beta E_i^t}}{\sum_{j=1}^n e^{-\beta E_j^t}} \tag{4.14}$$

als *softmax* Funktion [5] über die tiefpaßgefilterten Fehler E_i^t schreiben.

4.3 Spezialfall: Hard Competition

Der Parameter β bestimmt den Grad des Wettbewerbs um die Trainingsdaten. Für den Spezialfall $\beta \to \infty$, gilt

$$p(k(t) \mid \sigma_t^\Delta) = 1 \tag{4.15}$$

für den Prädiktor $k(t)$ mit dem kleinsten tiefpaßgefilterten Fehler E_i^t,

$$k(t) = \text{indexmin}\left\{ E_i^t \right\}, \tag{4.16}$$

und

$$p(i \mid \sigma_t^\Delta) = 0 \tag{4.17}$$

für alle übrigen Prädiktoren $i \neq k(t)$. Mit $\beta \to \infty$ macht folglich nur derjenige Prädiktor einen Lernschritt zum Zeitpunkt t, der den kleinsten Vorhersagefehler E_i^t hat. Dies entspricht exakt dem *hard competition* Lernverfahren in Kapitel 3: Gl.(3.9) ist bei Verwendung des gleitenden Durchschnitts aus Gl.(3.8) äquivalent zu Gl.(4.16).

Damit wurde gezeigt, daß sich das intuitiv entwickelte Lernverfahren, das in Abschnitt 3.1.2 beschrieben wurde, auch mit Mitteln der Wahrscheinlichkeitsrechnung motivieren läßt. Darüber hinaus ergibt sich daraus die im nächsten Kapitel folgende Erweiterung des Ansatzes.

Kapitel 5

Annealed Competition of Experts (ACE)

Im letzten Kapitel wurde die Methode der *Competing Experts* mit Mitteln der Wahrscheinlichkeitsrechnung fundiert. Daraus läßt sich nun eine Erweiterung des Verfahrens ableiten, die anstelle des reinen *winner-takes-all* Wettbewerbs einen kontinuierlichen Übergang vom gemeinsamen Lernen zum exklusiven Lernen während der Trainingsphase vorsieht. Es wird gezeigt, daß damit, im Gegensatz zur *hard competition*, auch noch kleine Unterschiede in der Dynamik aufgelöst werden können. Der Rechenaufwand steigt dafür jedoch erheblich an.

Wie bereits im vorigen Kapitel erwähnt, bestimmt der Parameter β den Grad des Wettbewerbs um die Trainingsdaten. Für den Spezialfall $\beta \to \infty$ ergibt sich die *hard competition* aus Abschnitt 3.1.2. Ein prinzipielles Problem von hartem Wettbewerb (*winner-takes-all*) ist, daß er ohne eine gute Initialisierung der Parameter – hier der Experten – leicht in lokalen Optima steckenbleiben kann [10].

In Abschnitt 3.1.3 wurde zwar eine Methode angegeben, die für eine sinnvolle Initialisierung der Experten sorgen soll, nämlich eine äquidistante Segmentierung der Daten im ersten Trainingsdurchgang (siehe Seite 37). Dies reicht jedoch nicht immer aus, um eine ausreichende Verteilung der Experten zu gewährleisten. Insbesondere dann nicht, wenn es gilt, auch kleinere Unterschiede in der Dynamik aufzulösen, wie in Abb.5.1(a) am Beispiel von *logistic* und *tent map* gezeigt wird. Im Unterschied zu Abb. 4.3(a) sorgt hier zwar ein Tiefpaß von $\Delta = 3$ für die richtige Zuordnung der Funktionsäste, aber die *hard competition*, d.h. $\beta \to \infty$,

verhindert hier die Unterscheidung von *logistic* und *tent map*, die zusammen von *einem* Prädiktor repräsentiert werden.

5.1 Soft Competition

Um das Initialisierungsproblem systematisch zu lösen, kann das *hard competition* Lernverfahren durch einen anfangs „weichen" Wettbewerb (*soft competition*) ersetzt werden, bei dem der „Temperaturparameter" $T = 1/\beta$ adiabatisch „abgekühlt" wird (*deterministic annealing*). Die Methode wurde von Rose *et al.* in [59] mit Mitteln der statistischen Mechanik beschrieben und zum *Clustering* von Daten verwendet. Angewendet auf die *Competing Experts*, bedeutet dies, daß ein *Clustering* im Funktionenraum der Experten durchgeführt wird.

In einem *Annealing*-Verfahren müßte man im Prinzip bei einer gegebenen Temperatur unendlich lange trainieren, bevor man die Temperatur unendlich geringfügig senken darf. Für die praktische Anwendung wurde stattdessen die folgende Heuristik verwendet:

Annealed Competition of Experts (ACE)

1. Beginne mit zufallsinitialisierten Prädiktoren und mit $T \approx 2\sigma^2$, wobei σ^2 für die Varianz der Trainingsdaten steht.

2. Trainiere $c = 10$ Trainingsepochen. In jeder Epoche sind dazu zunächst, jeweils für alle Prädiktoren f_i und alle Zeitschritte t, die Größen e_i^t, E_i^t und $p(i \mid \sigma_t^\Delta)$ zu berechnen (Gl. (4.12), Gl. (4.13), Gl. (4.14)). Die Lernregel für einen Parameter w eines Prädiktors f_i, bezüglich eines Datenpunkts $x(t)$, ist dann gegeben durch

$$\Delta w = \eta \, p(i \mid \sigma_t^\Delta) \, (x(t) - f_i) \, \frac{\partial f_i}{\partial w}. \tag{5.1}$$

   Dabei ist $\partial f_i / \partial w$ durch die Architektur des gewählten Prädiktors festgelegt. Für die Lernrate η und den Tiefpaß-Parameter Δ können, im Gegensatz zur *hard competition* in Kapitel 3, *konstante* Werte gewählt werden. Wir haben z. B. meist $\eta = 0.1$ und Δ je nach Anwendung im Bereich 3 bis 40 verwendet.

3. Wenn nach $c = 10$ Trainingsepochen der Gesamtvorhersagefehler (RMSE)

$$\hat{E}_{RMS} = \sqrt{\frac{1}{T} \sum_{t=1}^{T} \sum_{i=1}^{K} p(i \mid \sigma_t^{\Delta})\ e_i^t} \tag{5.2}$$

nicht mindestens $p = 1\%$ kleiner wurde, kühle ab: $T := \gamma \cdot T$, mit $\gamma \approx 0.96$. Sonst kühle nicht ab.

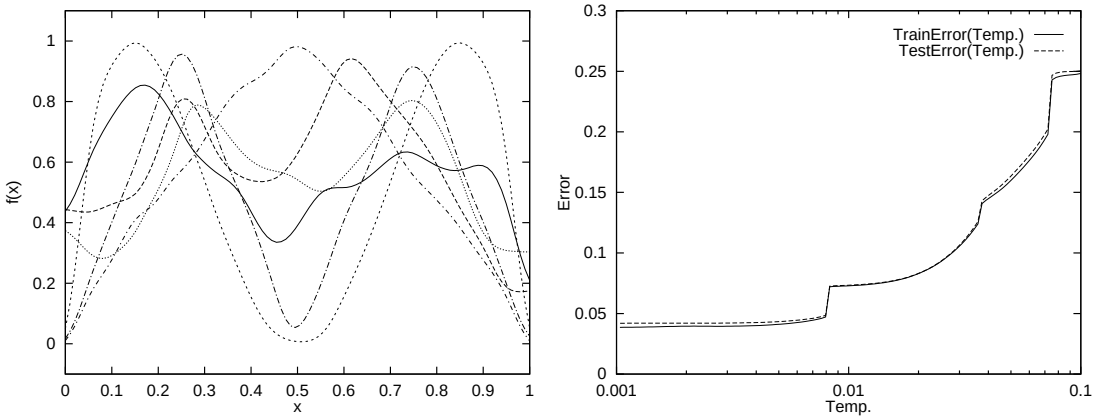
4. Wiederhole die Punkte 2 und 3, solange $T > 2\sigma^2 \cdot \varepsilon$, mit $\varepsilon \approx 10^{-4}$, d.h. so lange bis *hard competition* erreicht wird.

Das Lernverfahren beginnt demnach mit $T \gg 0$, wodurch *alle* Prädiktoren auf *allen* Daten in nahezu gleicher Weise trainieren, denn für die Größe der Lernschritte, die durch $p(i \mid \sigma_t^{\Delta})$ bestimmt wird, gilt anfangs $p(i \mid \sigma_t^{\Delta}) \approx 1/n$. Erst das Verkleinern von T bewirkt eine Differenzierung der Lernschrittweiten für die Prädiktoren, und zwar nach der Fähigkeit die Daten in der zeitlichen Umgebung des betrachteten Datenpunktes vorherzusagen. Je kleiner der Vorhersagefehler, desto größer ist der Lernschritt.

Je kleiner T wird, desto stärker wird der Verdrängungswettbewerb durch die zunehmende Polarisierung der Lernschrittfaktoren $p(i \mid \sigma_t^{\Delta})$ auf die Werte null und eins (*hard competition*). Dies zwingt die Prädiktoren zu einer Spezialisierung auf unterschiedliche Abschnitte der Zeitreihe.

Bei einer adiabatischen Abkühlung von T erfolgt die Spezialisierung bei bestimmten Temperaturen, bei denen sich die Vorhersagefunktionen der Prädiktoren abrupt separieren. Dabei wird die Struktur der in der Zeitreihe enthaltenen Einzeldynamiken genauer aufgelöst. Dies wird in Abb. 5.2 sehr klar am Beispiel der chaotischen Abbildungen verdeutlicht. Diese Phasenübergänge gehen deshalb auch mit einem sprunghaften Abfall des Gesamtvorhersagefehlers für die Zeitreihe einher (Abb. 5.1(b)). Nachdem die Struktur des dynamischen Systems vollständig aufgelöst wurde, treten keine weiteren Phasenübergänge mehr auf und es gibt nur noch eine geringfügige Verbesserung des Vorhersagefehlers, wenn T weiter gegen Null geht.

Nach dem Training mit ACE kann es sein, daß sich mehrere Prädiktoren auf dieselbe Dynamik spezialisiert haben, und zwar dann, wenn mehr Prädiktoren verwendet wurden als Einzeldynamiken in der Zeitreihe vorhanden sind und eine weitere Differenzierung nicht mehr möglich ist (wie z.B. in Abb. 5.2(d)). Ob und



(a) (b)

Abbildung 5.1: *(a) Ergebnis der reinen hard competition ($\beta \to \infty$) für die vier chaotischen Abbildungen. Obwohl eine sinnvolle Initialisierungsstrategie für die Experten verwendet wurde, hat sich nur ein Prädiktor auf logistic und tent map spezialisiert. Der Grund dafür ist die vergleichsweise Ähnlichkeit der beiden Funktionen. Eine Unterscheidung dieser beiden Funktionen durch das Ensemble ist damit nicht mehr möglich. Auch bleibt der Vorhersagefehler für beide Funktionen höher als es möglich wäre, da ein einzelner Prädiktor, wie hier, bestenfalls die Mittelwerte aus beiden Funktionen repräsentieren kann. (b) Mittlere Vorhersagefehler (RMSE) auf Trainings- und Testdaten, während der Trainingsphase mit deterministic annealing. Je weiter die Temperatur T im Laufe des Trainings gesenkt wird, desto kleiner werden die Fehler. Die Stufen in den Kurven zeigen die Phasenübergänge an.*

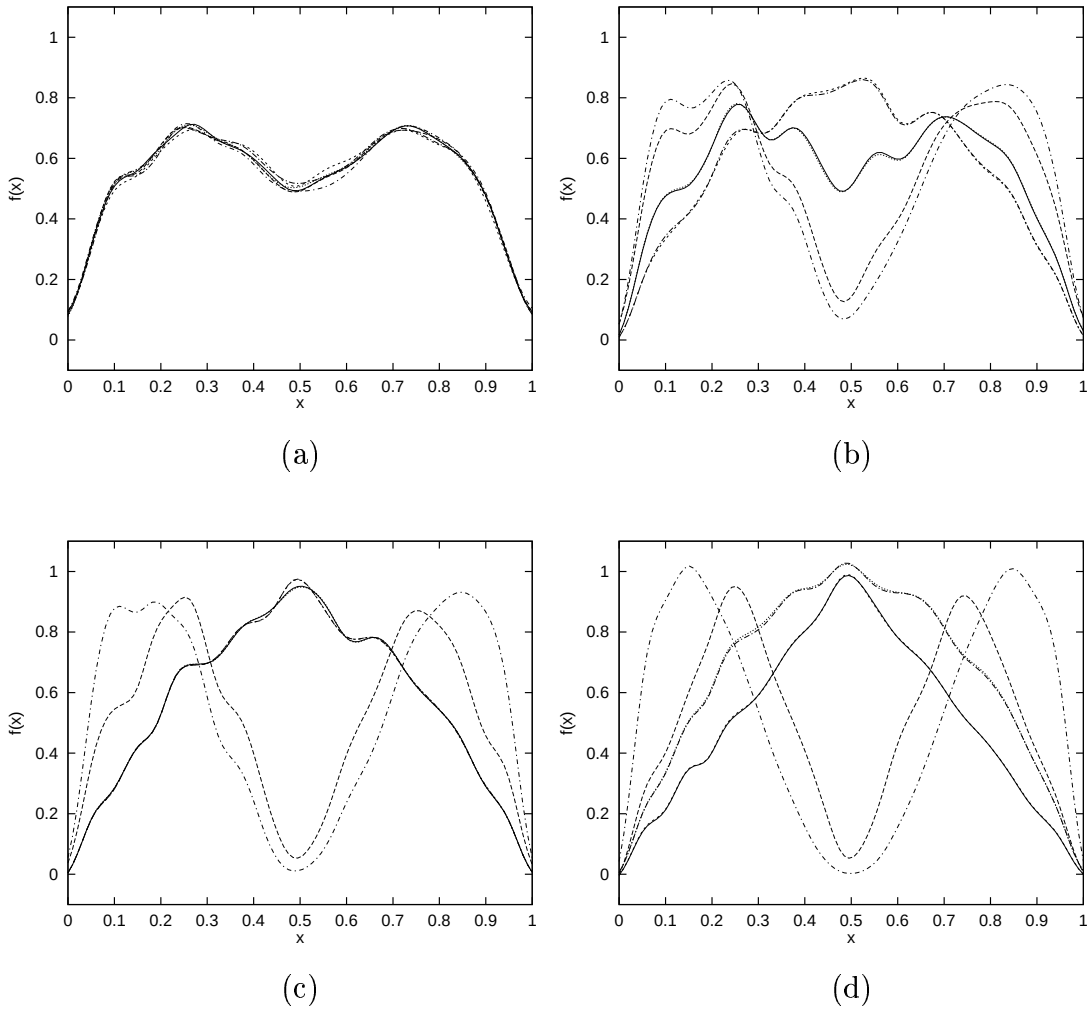


Abbildung 5.2: *Die von den sechs Prädiktoren f_i erlernten Funktionen bei Verwendung von ACE: (a) vor dem ersten und (b)-(d) nach jedem von insgesamt drei Phasenübergängen. Nach jedem Phasenübergang, der an bestimmten Stellen während des Absenkens der Temperatur T ausgelöst wird, werden die vier Zielfunktionen genauer aufgelöst. Nach dem letzten Phasenübergang haben sich je zwei Prädiktoren auf die logistic und tent map spezialisiert.*

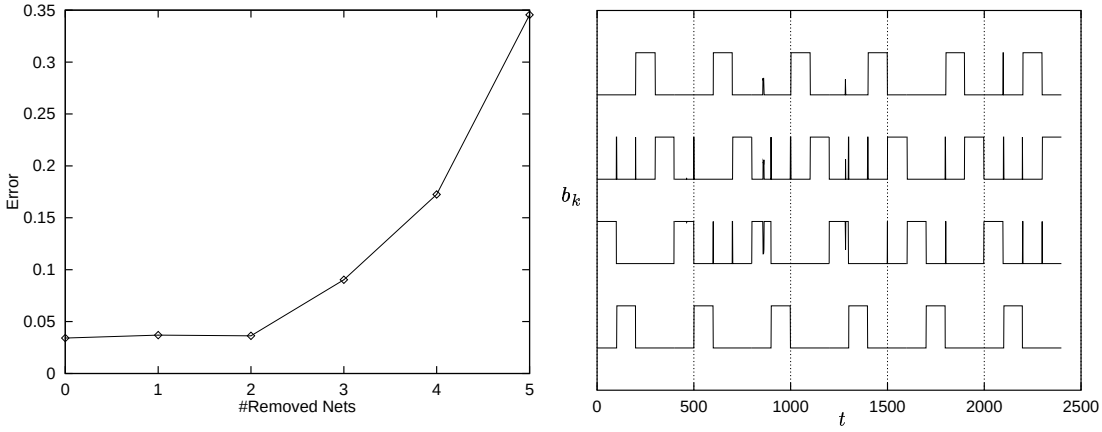
welche Experten überflüssig sind, kann anhand des Vorhersagefehlers auf folgende Weise bestimmt werden:

ENTFERNEN ÜBERFLÜSSIGER PRÄDIKTOREN

1. Berechnung des Gesamtvorhersagefehlers (RMSE) für das gesamte Ensemble.
2. Berechnung des Gesamtvorhersagefehlers (RMSE) für alle n Teilmengen, die aus nur $n-1$ Prädiktoren bestehen, und zwar jeweils durch Weglassen des i-ten Prädiktors.
3. Auswahl der Teilmenge mit dem niedrigsten Fehler.
4. Wiederholung der Punkte 2 und 3 für die gewählte Teilmenge bis nur noch ein Prädiktor übrig ist.

Die gewonnene Sequenz der Vorhersagefehler für die ausgewählten Teilmengen (mit $n-1$ bis 1 Prädiktoren), zeigt dann wie sich der Vorhersagefehler erhöht, wenn man sukzessive den jeweils am wenigsten benötigten Prädiktor wegläßt (Abb. 5.3(a)). Man wählt dann die Teilmenge aus, für die der Vorhersagefehler gar nicht oder nur geringfügig größer ist als für das gesamte Ensemble, und die dazu die wenigsten Prädiktoren benötigt. Dies ist typischerweise an einem sprunghaften Anstieg des Vorhersagefehlers zu erkennen, der dann auftritt, sobald ein *benötigter* Prädiktor entfernt wird, nachdem bereits alle Prädiktoren, die überflüssig sind, entfernt wurden.

Abb. 5.3(b) zeigt im Fall der chaotischen Abbildungen, daß die vier verbleibenden Experten dann sehr schön die Struktur der Dynamik reproduzieren. Die Fehler an den Schaltzeitpunkten rühren daher, daß die Segmentation auf den tiefpaßgefilterten Vorhersagefehlern basiert. Diese sind nämlich an den Schaltzeitpunkten auch für die beiden am Schalten beteiligten Prädiktoren relativ hoch, da sie für den Tiefpaß zur Hälfte Daten vorhersagen müssen, für die sie nicht zuständig sind. Dort ist es dann möglich, daß ein anderer Prädiktor, der mehr oder weniger den Mittelwert der beiden beteiligten Dynamiken gelernt hat, *im Mittel* den kleinsten Vorhersagefehler in dem Zeitfenster macht und so Datenpunkte an den Schaltpunkten zugeteilt bekommt. In dem vorliegenden Fall stellt insbesondere der Prädiktor der *double tent map* (zweiter von oben) in gewisser Näherung auch einen Prädiktor für das arithmetische Mittel von *double logistic map* und der *tent*



(a) (b)

Abbildung 5.3: *(a) Die Zunahme des Vorhersagefehlers für die chaotischen Abbildungen, wenn man wiederholt den jeweils am wenigsten benötigten Prädiktor entfernt. Es zeigt sich, daß man zwei der sechs trainierten Netze entfernen kann, ohne den Vorhersagefehler nennenswert zu erhöhen. Damit wird korrekt angezeigt, daß vier unterschiedliche Dynamiken in der Zeitreihe enthalten sind. Diese werden von den vier verbleibenden Prädiktoren repräsentiert. (b) Die Segmentation $k(t)$ der Zeitreihe zeigt, daß die vier verbleibenden Prädiktoren die Struktur der Dynamik richtig reproduzieren. An den Schaltzeitpunkten führt jedoch die Verwendung der tiefpaßgefilterten Vorhersagefehler E_i^t, die hier das Kriterium zur Segmentation sind, teilweise zu Fehlern (Erläuterung siehe Text).*

map (bzw. auch der *logistic map*) dar, und er erhält daher die entsprechenden Transienten zugeteilt. In den folgenden Kapiteln wird der Viterbi-Algorithmus [57] zur Segmentation verwendet, der diesen speziellen, aber systematischen Effekt nicht aufweist.

5.2 Beispiel: Mackey-Glass

Als ein Anwendungsbeispiel für ACE soll die verrauschte Zeitreihe des Mackey-Glass Systems aus Abschnitt 3.2.2 verwendet werden. In der Mackey-Glass Differentialgleichung [38]

$$\frac{dx(t)}{dt} = -0.1\ x(t) + \frac{0.2\ x(t-t_d)}{1 + x(t-t_d)^{10}} \tag{5.3}$$

wurde der Parameter t_d dabei so gewählt, daß die ersten 100 Trainingsdaten mit $t_d = 17$, die zweiten mit $t_d = 23$, die dritten mit $t_d = 30$ und die vierten schließlich wieder mit $t_d = 17$ generiert wurden. Die mit $\tau = 6$ abgetastete Zeitreihe schaltet somit alle 100 Zeitschritte in einen anderen Modus $l = 1, 2, 3, 1$ (Abb. 5.4(a)). Zur Rekonstruktion des Phasenraums wurde wieder die Einbettungsdimension $d = 6$ gewählt (siehe Gl.(3.2)).

Sechs RBF-Netze mit 40 Zentren wurden zur Vorhersage der Zeitreihe angesetzt. Der verwendete Tiefpaß hat die Größe $\Delta = 10$. Während des ACE-Trainings ($\eta = 0.1$, $T = 0.1 \ldots 0.0001$, $\gamma = 0.96$) erfolgen zwei Phasenübergänge, die anzeigen, daß eine Struktur im dynamischen System erkannt wurde (Abb. 5.4(b)). Der zweite Übergang (bei $T \approx 0.0007$) wird deutlicher, wenn kleinere Netze mit weniger Zentren verwendet werden, was jedoch zu einer schlechteren Vorhersagegüte führt.

Das Entfernen von bis zu drei Netzen nach dem Ende des Trainings führt nicht zu einer signifikanten Erhöhung des Vorhersagefehlers, was korrekt anzeigt, daß bereits drei Prädiktoren ausreichen, um das System vollständig zu beschreiben (Abb. 5.4(c)). Die Segmentation durch diese drei Prädiktoren gibt das Schaltverhalten des Systems perfekt wieder (Abb. 5.4(d)).

5.3 Beispiel: Data Set D

Die Annahme von Stationarität ist in vielen Fällen der Datenanalyse problematisch. Der ACE Ansatz stellt ein Methode zur *Analyse*, aber auch eine Methode zur *Vorhersage* von Systemen dar, die Nichtstationaritäten in Form von Sprüngen in den Systemparametern enthalten. In diesem Abschnitt wird die Bedeutung des Ansatzes für die Vorhersage von Zeitreihen demonstriert. Dennoch muß an dieser Stelle betont werden, daß mit dem Ansatz zwar eine sehr gute Vorhersagequalität *zwischen* zwei Schaltzeitpunkten erreicht werden kann, die Schaltzeitpunkte selbst können aber nicht vorhergesagt werden, da wir annehmen, daß sie externe Einflußgrößen sind, die von der Einzeldynamik des jeweiligen Operationsmodus unabhängig sind. Zur Vorhersage der Schaltzeitpunkte müßte dann zusätzlich eine Statistik des Schaltverhaltens erstellt werden, aus der man die Wahrscheinlichkeit des Schaltens berechnen könnte. Das Aufstellen oder gar Erlernen einer solchen Statistik erfordert aber eine wesentlich größere Menge an Daten (die ausreichend viele Schaltvorgänge enthalten muß), als für ACE erforderlich ist. Daher ist es

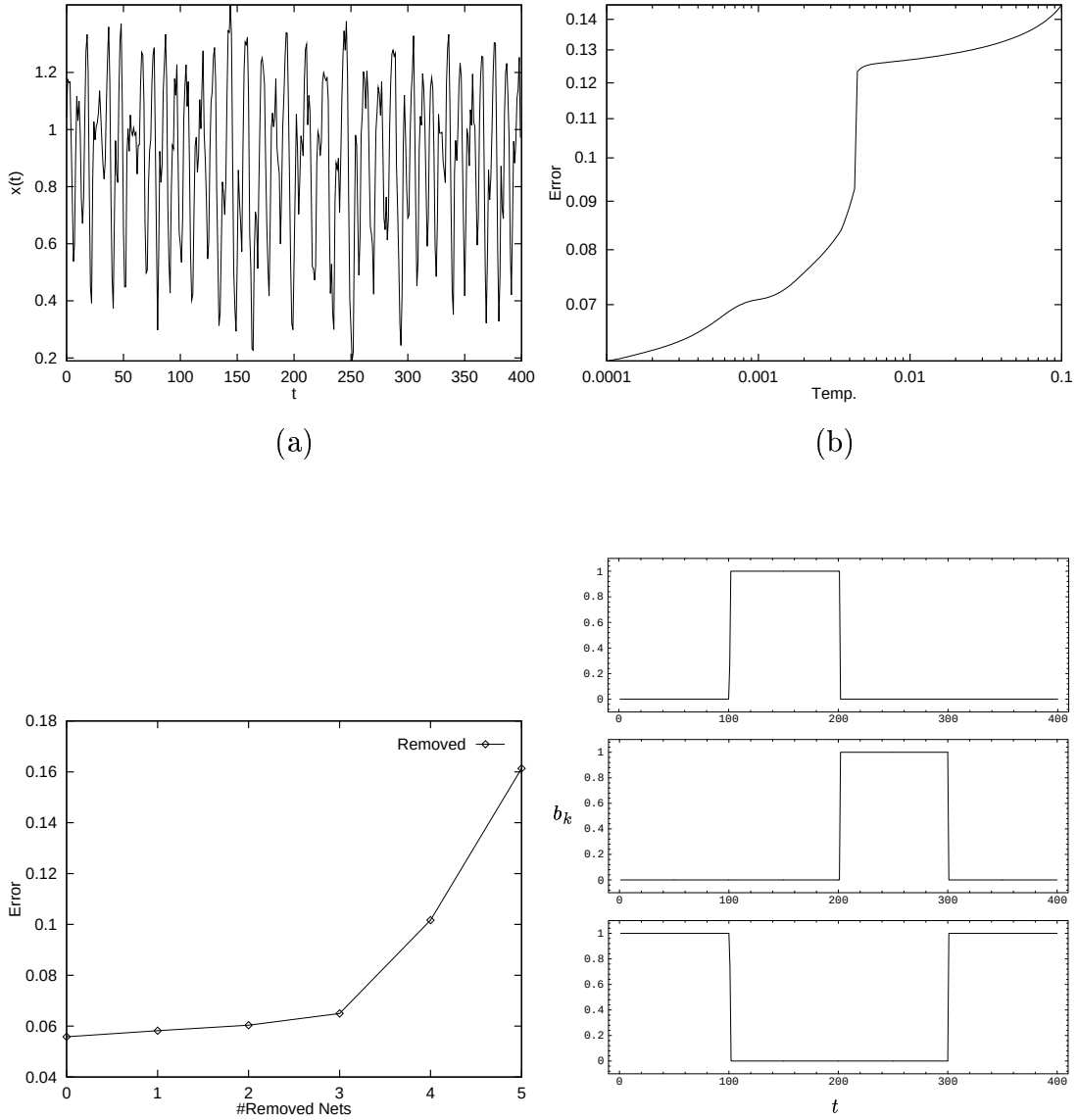


Abbildung 5.4: *(a) Die verrauschte Mackey-Glass Zeitreihe enthält drei unterschiedliche Dynamiken. (b) Verlauf des Vorhersagefehlers (RMSE) in Abhängigkeit der sinkenden Temperatur T, während des ACE-Trainings. (c) Zunahme des Vorhersagefehlers beim sukzessiven Entfernen von Prädiktoren nach dem Training. Bis zu drei Prädiktoren können entfernt werden, ohne daß der Fehler deutlich zunimmt. (d) Korrekte Segmentation der Zeitreihe durch die drei verbleibenden Prädiktoren.*

sinnvoll, dieses Problem separat, z. B. auf der Basis einer ACE-Segmentation, zu betrachten. Ansätze, die ein Schalten modellieren, das von der Entwicklung der jeweiligen Einzeldynamik abhängig ist, finden sich in [2, 6].

Im folgenden wird die Anwendung von ACE auf die Vorhersage des Datensatzes D der *Santa Fe Time Series Competition* [69] beschrieben. Dieser skalare Benchmark-Datensatz wurde von einem neundimensionalen, periodisch getriebenen, dissipativen dynamischen System erzeugt, das vier Potentialmulden besitzt und außerdem eine leichte Drift der Parameter aufweist. Gegeben sind 10000 Datenpunkte, deren Fortsetzung vorhergesagt werden soll. Das System hat die Eigenschaft, eine zeitlang in einer Potentialmulde zu laufen und dann in eine andere zu springen. Die Idee ist nun, das Vorhersageproblem zu unterteilen und je einen Prädiktor für die Vorhersage in einer Potentialmulde zu gewinnen. Die Hoffnung dabei ist, damit das Gesamtproblem in vier kleinere Probleme zu unterteilen und so die Vorhersagequalität zu verbessern.

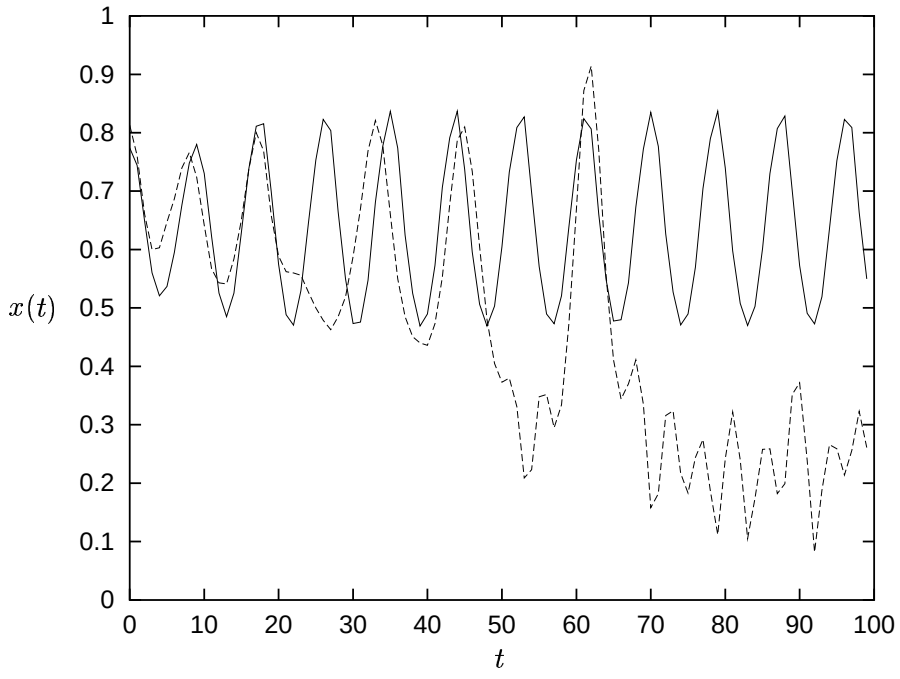
Sechs RBF-Prädiktoren mit 40 Zentren wurden mit ACE trainiert, um aus 20 vorangehenden Datenpunkten den nächsten Punkt vorherzusagen, das heißt die Einbettung ist $d = 20$ und $\tau = 1$. Als Tiefpaß wurde $\Delta = 5$ gewählt. Die weiteren ACE-Parameter waren $\eta = 0.1$, $T = 0.1 \ldots 0.0001$, und $\gamma = 0.96$. Der Trainingsdatensatz wurde auf die letzten 2000 Datenpunkte von Datensatz D begrenzt um die Rechenzeit in Grenzen zu halten und um das Problem der leichten Parameterdrift, die kontinuierlich über den gesamten Datensatz geht, gering zu halten. Nach der Trainingsphase waren die Datenpunkte der Zeitreihe den Prädiktoren ohne erkennbare Struktur zugeordnet, mit häufigem Wechsel des Prädiktors.

Die Vorhersage der Fortsetzung des Datensatzes D wurde nun einfach durch Iterieren desjenigen Prädiktors bewerkstelligt, der für die Vorhersage der letzten Datenpunkte von Datensatz D zuständig war. Das Iterieren erfolgt dabei einfach durch Zurückkoppeln des vorhergesagten Wertes $x(t+1)$ auf den zeitverzögerten Eingabevektor und Vorhersage des nächsten Wertes $x(t+2)$, so daß der Prädiktor die weiteren Vorhersagen auf seine bereits gemachten Vorhersagen basiert (*autonomous prediction*). Die wahre Fortsetzung der Zeitreihe wird dabei also nicht verwendet. Die Vorhersage der Fortsetzung wird schließlich mit der wahren Fortsetzung verglichen um den Vorhersagefehler zu berechnen.

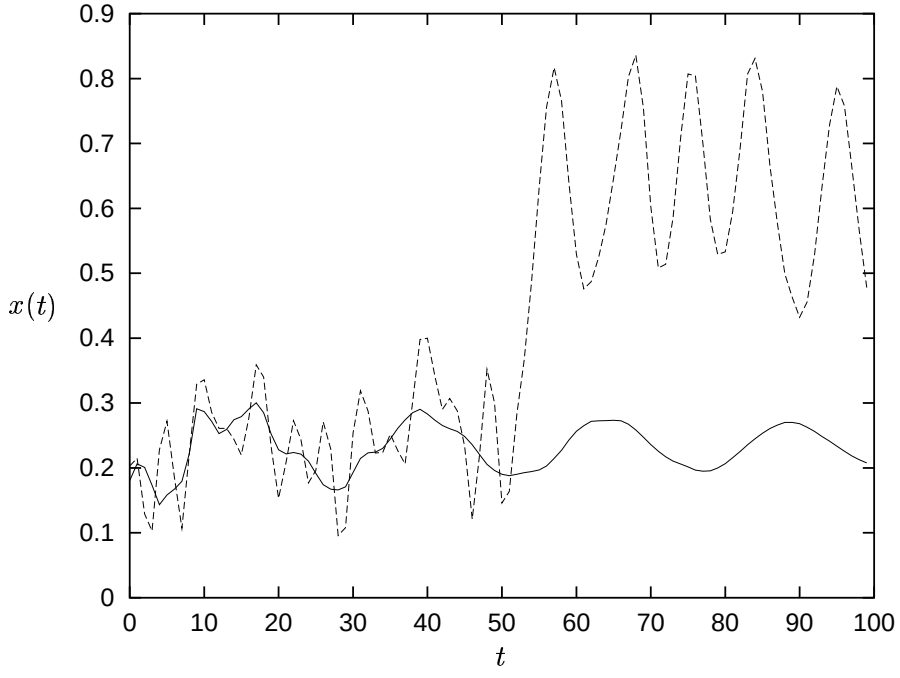
Die Methode erwies sich als recht erfolgreich zur Vorhersage der ersten 50 Zeitschritte der wahren Fortsetzung (Abb. 5.5(a)). Nach 50 Zeitschritten erfolgt offenbar ein Sprung in eine andere Potentialmulde, der grundsätzlich nicht mit

dem ACE Ansatz vorhergesagt werden kann (siehe oben), da sich der Prädiktor selbst auf eine stationäre Dynamik einschwingt und eine Vorhersage des Schaltens nicht stattfindet. Trotzdem ist es interessant herauszufinden, ob nicht ein anderer Prädiktor sich darauf spezialisiert hat, die Dynamik nach dem Sprung fortzusetzen, indem man ihn an einer Stelle nach dem Sprung aufsetzt und iteriert. Tatsächlich gibt es einen solchen Prädiktor mit ebenfalls einer guten Vorhersageleistung, wiederum bis zum nächsten Sprung in der Dynamik (Abb. 5.5(b)).

Ein Vergleich mit dem besten Ergebnis der *Santa Fe Competition* für Datensatz D zeigt die Stärke unseres Ansatzes. Die Sieger des Benchmark-Wettbewerbs, Zhang und Hutchinson [69, pp. 219–241], verwendeten ein einzelnes, sehr komplexes, neuronales Netz, das 100 Stunden Trainingszeit auf einer Connection Machine CM-2 mit 8192 Prozessoren benötigte, und sie erreichten einen Vorhersagefehler von $E_{RMS} = 0.0665$ (Root-Mean-Squared Error) bezogen auf die ersten 25 Zeitschritte, da die Vorhersage danach zusammenbrach. Für denselben Zeitabschnitt erreicht der ACE Ansatz einen Fehler von nur $E_{RMS} = 0.0596$, das ist eine Verbesserung um 10% bei einer Trainingszeit von nur 2.5 Stunden auf einer SUN 10/20GX Workstation.



(a)



(b)

Abbildung 5.5: *Vorhersage (durchgezogene Linie) der Fortsetzung von Datensatz D (gestrichelte Linie) durch RBF-Netze, die mit ACE trainiert wurden. Der ACE Ansatz zerlegt die Dynamik in einfachere Vorhersageprobleme, so daß jeder Prädiktor bestimmte Segmente der Zeitreihe am besten vorhersagen kann: (a) Vorhersage des Anfangs der Fortsetzung (unter Wettbewerbsbedingungen), (b) Vorhersage eines anderen Teilstücks durch einen anderen Prädiktor.*

Kapitel 6

Erkennung linearer Transienten

Der in Kapitel 5 vorgestellte ACE Algorithmus, sowie dessen Spezialfall, die *hard competition*, können *schaltende* Dynamik segmentieren und identifizieren. Wenn jedoch das dynamische System nicht unmittelbar von einem Zeitpunkt zum nächsten den Operationsmodus wechselt, sondern die Dynamik über einen gewissen Zeitraum von A nach B *driftet*, kann dieses Verhalten bisher nur durch ein Schalten angenähert werden. In vielen realen Systemen, wie zum Beispiel bei den Wach/Schlaf-Übergängen in Kapitel 8, ist jedoch zu vermuten, daß sich die Dynamik nicht immer schlagartig ändert, sondern daß sich die Änderung über eine gewisse Zeitspanne hinzieht. In diesem Kapitel wird daher eine Segmentierungsmethode vorgestellt, die im Anschluß an das ACE Verfahren angewendet wird und nicht nur ein Schalten, sondern auch lineare Übergänge in der Dynamik erkennen und darstellen kann.

6.1 Segmentierung mit linearer Drift

Die Erkennung und Modellierung linearer Übergänge erfolgt in zwei Schritten. Zuerst wird der ACE Algorithmus angewendet. Dieser setzt voraus, daß die Einzeldynamiken des Systems jeweils über einen gewissen Zeitraum stationär sind, bevor sich die Dynamik ändert. Diese Annahme ist notwendig gewesen, um den Lösungsraum einzuschränken und das Verfahren auf das Auffinden selten schaltender Modelle der Dynamik zu beschränken.

Mit dem ACE-Verfahren erhält man eine schaltende Segmentation der gegebenen Zeitreihe und eine Menge von Prädiktionsexperten für die einzelnen Segmente. Im Fall einer Drift zwischen zwei Operationsmoden wird das Interval, in dem die Drift auftritt, in mehrere Segmente unterteilt, für deren Vorhersage verschiedene Prädiktoren zuständig sind (Abb. 6.1). Ein einzelner Prädiktor ist nicht in der Lage, die nichtstationäre Drift zu modellieren, da sie die Eigenschaft hat, den funktionalen Zusammenhang der Daten kontinuierlich zu verändern. Wenn nicht genügend Prädiktoren vorhanden sind, die sich auf den Driftbereich spezialisieren können, erfolgt im einfachsten Fall ein harte Segmentation des Driftintervals zwischen den Prädiktoren der beiden beteiligten Operationsmoden. In jedem Fall wird die Drift also lediglich durch ein Schalten zwischen zwei oder mehreren Prädiktoren approximiert.

Der zweite Schritt besteht daher in der Anwendung eines neuen Segmentationsalgorithmus, der auch eine Drift zwischen zwei stationären Dynamiken modellieren kann. Dies geschieht durch eine Kombination der beiden Prädiktoren, f_i und f_j, die für die Vorhersage der beiden stationären Dynamiken zuständig sind. Um die Drift zu modellieren, können die Vorhersagen der beiden Prädiktoren dabei prinzipiell durch eine beliebige nichtlineare und zeitabhängige Abbildung $g(f_i, f_j, t)$ kombiniert werden. Ohne *a priori* Wissen darüber, wie die Drift zwischen zwei stationären Dynamiken in einer konkreten Anwendung tatsächlich vonstatten geht, wollen wir die Drift im folgenden jedoch einfach durch eine zeitabhängige Linearkombination der Prädiktoren modellieren:

$$f(\vec{x}_t) = (1 - a(t))\, f_i(\vec{x}_t) + a(t)\, f_j(\vec{x}_t), \tag{6.1}$$

wobei das Mischungsverhältnis $a(t)$ in dem Drift-Interval $[t_a, t_b]$ von Null nach Eins läuft. $\vec{x}_t = (x_{t-\tau}, x_{t-2\tau}, \ldots, x_{t-d\tau})$ ist der Vektor der zeitverzögerten Koordinaten der Zeitreihe $\{x_t\}$. Für den nachfolgend beschriebenen Segmentationsalgorithmus kann und muß eine feste Form für die Übergangscharakteristik $a(t)$ gewählt werden (für variable Driftverläufe siehe Kapitel 7). Hierbei kann *a priori* Wissen über das zu untersuchende dynamische System einfließen und so der Lösungsraum von vornherein eingeschränkt werden. Wir beschränken uns im folgenden auf die Betrachtung linearer Transienten

$$a(t) = \frac{t - t_a}{t_b - t_a}. \tag{6.2}$$

Der Segmentationsalgorithmus führt eine *vollständige Suche* nach der optimalen Segmentation mit dem kleinsten Vorhersagefehler durch. Dabei wird sowohl ein Schalten der Dynamik als auch die oben beschriebene lineare Drift berücksichtigt.

Die Suche wird folgendermaßen durchgeführt: Für eine gegebene Zeitreihe, die nicht notwendigerweise die Trainingsmenge des ACE Algorithmus sein muß, macht jeder trainierte Prädiktor f_i eine Vorhersage für jeden Zeitpunkt t. Dies ergibt eine $N \times T$ Matrix von Expertenvorhersagen versus Zeitschritten. Diese Matrix kann zur Berechnung der Vorhersagefehler aller möglicher Segmentationen der Zeitreihe benutzt werden, einschließlich Drifts beliebiger Länge, aber fester Übergangscharakteristik (zum Beispiel Gl. (6.2)). Die Segmentation mit dem kleinsten Vorhersagefehler stellt dann das Ergebnis dar (siehe Abb. 6.1) – wobei zusätzlich die Annahme berücksichtigt wird, daß die Wahrscheinlichkeit zu einem anderen Prädiktor zu wechseln (durch Driften oder Schalten), kleiner ist als die Wahrscheinlichkeit beim aktuell zuständigen Prädiktor zu bleiben (Annahme seltener Übergänge). Dieses Problem kann effizient mit dynamischer Programmierung (*dynamic programming*) gelöst werden:

Unter Verwendung des Vorhersagefehlers eines einzelnen Prädiktors

$$e_n(t) = (x_t - f_n(\vec{x}_{t-\tau}))^2, \tag{6.3}$$

mit $\vec{x}_{t-\tau} = (x_{t-\tau}, \ldots, x_{t-d\tau})$, sowie der Transitionskosten für den Übergang von Prädiktor i zu Prädiktor n

$$s_{i,n} = \begin{cases} 0 & ; \text{falls } i = n \\ \alpha & ; \text{sonst} \end{cases} \tag{6.4}$$

und des Vorhersagefehlers der linearen Drift von Prädiktor i zu Prädiktor n im Interval $[t', t]$

$$l_{i,n}(t', t) = \sum_{k=t'+1}^{t-1} \left(x_k - \left[\frac{k-t'}{t-t'} f_n(\vec{x}_{k-\tau}) + \left(1 - \frac{k-t'}{t-t'} \right) f_i(\vec{x}_{k-\tau}) \right] \right)^2 \tag{6.5}$$

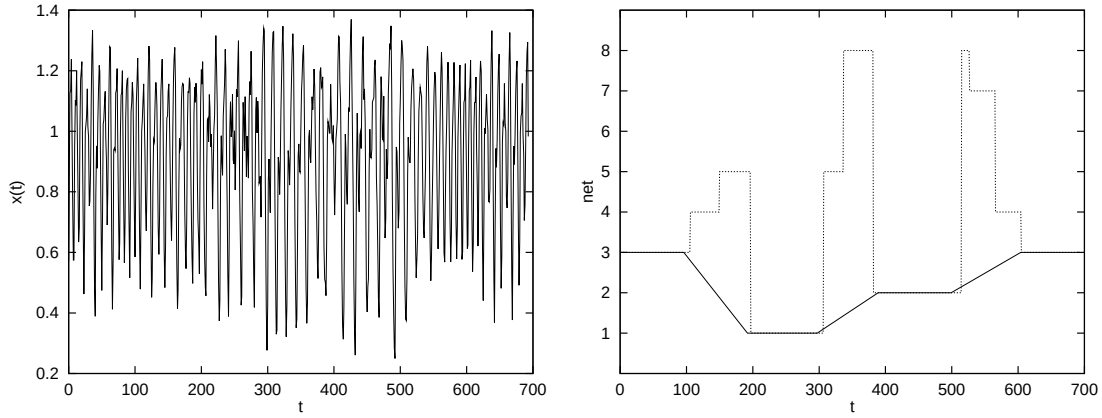
wird die Kostenfunktion $C_n(t)$ für alle Prädiktoren rekursiv über alle Zeitschritte, $t = 1, ..., T$, der Zeitreihe berechnet (Abb. 6.2):

- Initialisierung (für $t = 1$)

$$C_n(1) = e_n(1) \tag{6.6}$$

- Rekursion (für $t = 2, ..., T$)

$$C_n(t) = e_n(t) + \min_{1 \le i \le n,\ 1 \le t' < t} \left\{ C_i(t') + s_{i,n} + l_{i,n}(t', t) \right\} \tag{6.7}$$



(a) (b)

Abbildung 6.1: *(a) Eine driftende Mackey-Glass Zeitreihe. Drei Operationsmoden, A, B und C, werden durch Verwendung unterschiedlicher Systemparameter* $\tau = 17, 23, 30$ *erzeugt (aus Gl. (3.19)). Nach 100 Zeitschritten in Modus A driftet die Dynamik linear nach Modus B. Die Drift dauert wiederum 100 Zeitschritte. Danach arbeitet das System 100 Zeitschritte stationär in Modus B, woraufhin es nach Modus C driftet, usw. (b) Gepunktete Linie: Die Segmentation der Zeitreihe durch das ACE Training, auf der Basis der tiefpaßgefilterten Vorhersagefehler. Dargestellt ist die Sequenz der Prädiktoren in Abhängigkeit der Zeit. Die Driftabschnitte werden von mehreren, aufeinanderfolgenden Prädiktoren vorhersagt, da die Drift nicht angemessen repräsentiert werden kann. Durchgezogene Linie: Die Segmentation mit linearer Drift. Zwischen* $t = 100$ *und* 200 *bezeichnet die Linie zum Beispiel eine lineare Drift von Prädiktor 3 direkt zu Prädiktor 1 (und nicht etwa über Prädiktor 2). Das Verhalten des dynamischen Systems wird durch den neuen Algorithmus perfekt reproduziert.*

Schließlich erhält man die Segmentation mit den niedrigsten Vorhersagekosten (Vorhersagefehler plus Summe der Transitionskosten)

$$C^* = \min_n \left\{ C_n(T) \right\} \tag{6.8}$$

als Sequenz der in C^* vorkommenden Prädiktoren und Drifts. Die Transitionskosten α legen fest, um wieviel sich der Vorhersagefehler mindestens verbessern muß, damit jeweils ein Wechsel des Prädiktors erlaubt wird. Damit werden Segmentationen bevorzugt, die selten schalten oder driften. Eine Bevorzugung zwischen Schalten und Driften gibt es dagegen *a priori* nicht, d.h. die Dauer einer Transition wird allein durch den Vorhersagefehler bestimmt.

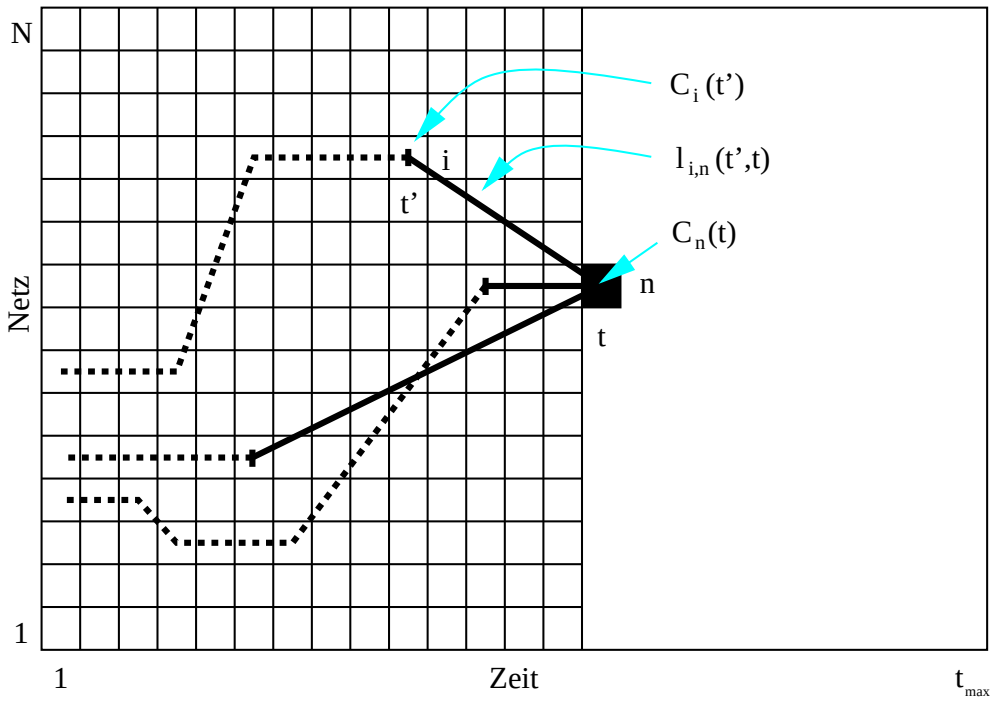
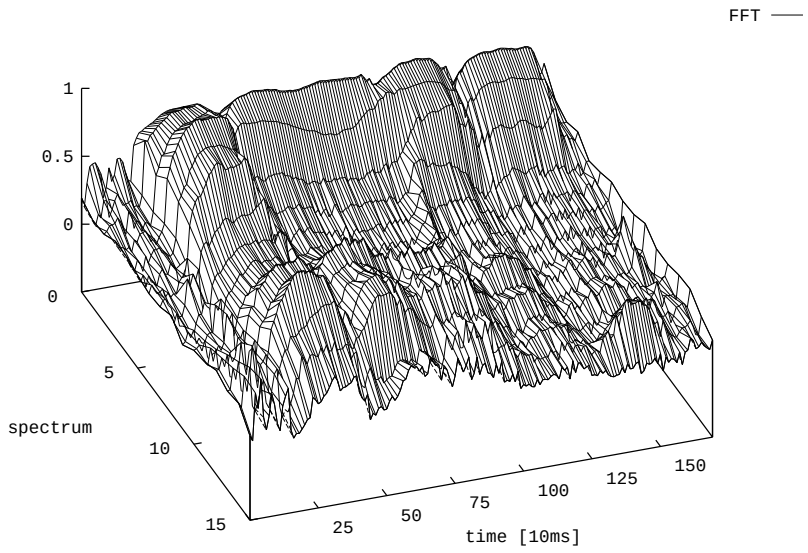


Abbildung 6.2: *Illustration des Induktionsschritts: Die beste Segmentation mit den niedrigsten Vorhersagekosten $C_n(t)$, die zum Zeitpunkt t bei Prädiktor n endet, läßt sich dadurch finden, indem man alle bisher berechneten besten Segmentationen, für alle $t' = 1, ..., t-1$ und alle Endpunkte $i = 1, ..., N$, betrachtet und jeweils die Kosten der linearen Transition zu Prädiktor n zum Zeitpunkt t hinzuaddiert. Von diesen Segmentationen ist dann diejenige mit den kleinsten Vorhersagekosten auszuwählen.*

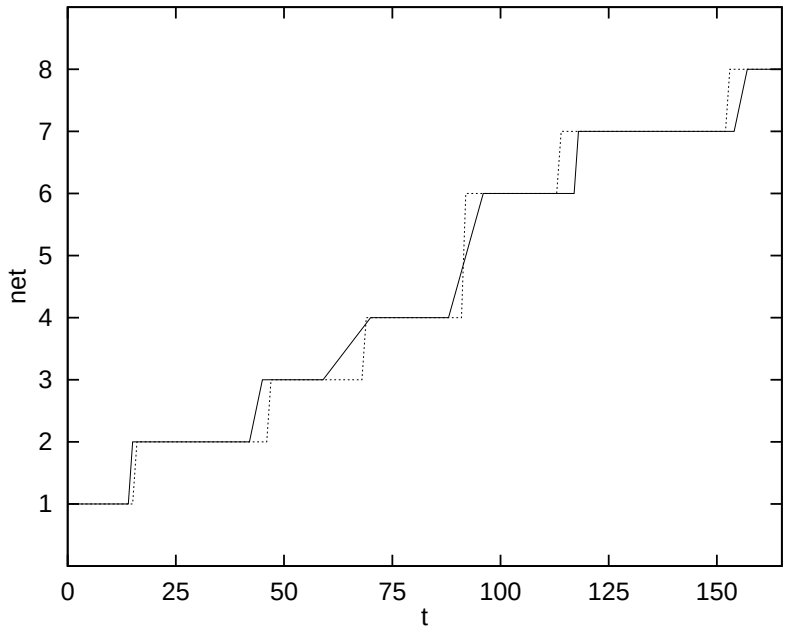
6.2 Beispiel: Sprachdaten

Die praktische Relevanz, Drift zu modellieren, soll am Beispiel von Sprachdaten verdeutlicht werden. Die Daten bestehen aus 16-dimensionalen mel-scale FFT Koeffizienten von kontinuierlich gesprochenen Vokalen "AEIOU" (Abb. 6.3(a), vgl. Abschnitt 3.2.3). Die Vorhersage erfolgt hier also nicht auf dem Signal selbst, sondern im 16-dimensionalen Fourier-Raum. Acht Prädiktoren wurden mit *hard competition* auf einem 16×4-dimensionalen Eingangsraum ($m = 4, \tau = 1$) darauf trainiert, den nachfolgenden FFT-Koeffizienten-Vektor vorherzusagen [45]. Nach dem Training haben sich fünf Prädiktoren jeweils auf A, E, I, O und U spezialisiert (gepunktete Linie in Abb. 6.3(b)), zwei Prädiktoren (1 und 8) repräsentieren die Ruhephasen am Anfang und am Ende, und ein Prädiktor (5) ist aus dem Wettbewerb herausgefallen und ist nicht mehr beteiligt.

Danach wurde der Drift-Segmentationsalgorithmus auf alle trainierten Prädiktoren angewendet. Vier der sechs Transienten (durchgezogene Linie in Abb. 6.3(b)) wurden nunmehr als lineare Drift zwischen zwei Prädiktoren modelliert: über 30 ms von A nach E, 110 ms von E nach I, 80 ms von I nach O, und 30 ms von U zur Ruhepause. Die beiden anderen Transienten – der Beginn des Vokals A und der Übergang von O nach U – werden als Schalten von einem Zeitschritt zum nächsten (10 ms) modelliert. Die Drift-Segmentation hat eine deutlich höhere Vorhersagegenauigkeit an den Übergängen als die harte Segmentation, zum Beispiel reduziert sich der Vorhersagefehler auf dem 110 ms E/I-Driftsegment um den Faktor 0.38 und auf dem 80 ms I/O-Driftsegment um den Faktor 0.64. Das zeigt, daß die lineare Drift ein besseres Modell für die Transienten der Vokaldaten ist als das Schalten.



(a)



(b)

Abbildung 6.3: *(a) Die Entwicklung der 16 short-time FFT-Koeffizienten für die gesprochene Vokalfolge AEIOU. (b) Gepunktete Linie: Die Segmentation der 16-dimensionalen Zeitreihe auf der Basis der tiefpaßgefilterten Vorhersagefehler (harte Segmentation ohne Drift). Durchgezogene Linie: Segmentation mit linearer Drift. Die Drift von Netz 3 zu Netz 4 ist besonders prägnant.*

Kapitel 7

Erkennung nichtlinearer Transienten

In diesem Kapitel wird eine Segmentationsmethode vorgestellt, mit der man nicht nur lineare Übergänge in der Dynamik modellieren kann, wie es im vorigen Kapitel beschrieben wurde, sondern auch variable, nichtlineare Verläufe der Drift zwischen zwei Operationsmoden. Dabei wird die Form der Drift *adaptiv* aus den Daten der Zeitreihe bestimmt.

Die Vorgehensweise zur Erkennung variabler, nichtlinearer Drift erfolgt zunächst wie im Fall der linearen Drift. Der erste Schritt ist die Anwendung des ACE Trainingsverfahrens, um Prädiktionsexperten für die Zeitreihe zu erhalten (vgl. Kapitel 6). Der zweite Schritt besteht jedoch in der Anwendung eines neuen Segmentationsalgorithmus, der Drift zwar auch, wie in Kapitel 6, als eine gewichtete Superposition zwischen zwei Prädiktoren, f_i und f_j modelliert,

$$f(\vec{x}_t) = (1 - a(t))\, f_i(\vec{x}_t) + a(t)\, f_j(\vec{x}_t), \quad 0 \leq a(t) \leq 1, \tag{7.1}$$

wobei aber der Driftverlauf $a(t)$ nicht linear sein muß, sondern adaptiv aus den Daten bestimmt wird.

7.1 Hidden Markov Modell für variable Drift

Im folgenden wird ein Hidden Markov Modell (HMM) [57] unter Verwendung der trainierten Prädiktoren definiert, das als Modell für das driftende dynamische

System verwendet wird. Für eine detailierte Beschreibung von Hidden Markov Modellen sei auf [57] und die Referenzen darin verwiesen. Das Hidden Markov Modell ermöglicht es uns, den Viterbi-Algorithmus [57] zur Analyse driftender Dynamik einzusetzen. Ein Hidden Markov Modell besteht aus

1. einer Menge S von Zuständen,
2. einer Matrix $A = \{p_{\hat{s},s}\}$ von Übergangswahrscheinlichkeiten zwischen den Zuständen,
3. einer Wahrscheinlichkeitverteilung $p(y|s)$ der beobachteten Größe y für jeden Zustand s, die in diesem Fall eine kontinuierliche Dichtefunktion darstellt,
4. einer Menge $\pi = \{\pi_s\}$ von Anfangswahrscheinlichkeiten der Zustände.

Der entscheidende Punkt unseres Ansatzes ist die Konstruktion der Menge von Zuständen, S. Sie soll daher als erstes betrachtet werden.

Gegeben sei eine Menge P von „puren“ Zuständen. Jeder Zustand $s \in P$ repräsentiert einen Prädiktor $k(s)$, der jeweils allein für die Vorhersage zuständig ist. Gegeben sei außerdem eine Menge M von „gemischten“ (Drift-)Zuständen, wobei jeder Zustand $s \in M$ eine bestimmte Mischung von zwei Prädiktoren $i(s)$ und $j(s)$ repräsentiert. Für einen gegebenen Zustand $s \in S, S = P \cup M$, wird die Vorhersage des Gesamtsystems dargestellt durch

$$g_s(\vec{x}_t) = \begin{cases} f_{k(s)}(\vec{x}_t) & ; \text{falls } s \in P \\ a(s) f_{i(s)}(\vec{x}_t) + b(s) f_{j(s)}(\vec{x}_t) & ; \text{falls } s \in M \end{cases} \tag{7.2}$$

Für jeden gemischten Zustand $s \in M$ müssen die Koeffizienten $a(s)$ und $b(s)$, sowie die Prädiktorindizes $i(s)$ und $j(s)$ definiert werden. Um den Rechenaufwand später in Grenzen zu halten, darf die Anzahl der Mischungszustände nicht beliebig groß sein. Daher werden nur Drifts zwischen je zwei Prädiktoren des zuvor trainierten Ensembles erlaubt, so daß $0 < a(s) < 1$ und $b(s) = 1 - a(s)$ (vgl. Gl. (7.1)). Darüber hinaus muß die Menge M endlich sein, so daß also nur eine endliche Menge von a-Koeffizienten verwendet werden kann. Der Einfachheit halber erfolgt die notwendige Diskretisierung in gleichgroßen Schritten,

$$a_r = \frac{r}{R+1} \ , \ r = 1, \ldots, R. \tag{7.3}$$

R gibt dabei die Anzahl der Zwischenstufen an. Eine gegebene Auflösung R zwischen je zwei von N Prädiktoren erfordert demnach die folgende Gesamtanzahl an gemischten Zuständen: $|M| = R \cdot N \cdot (N-1)/2$. Für die Auflösung $R = 32$ und $N = 8$ Prädiktoren, zum Beispiel, werden $|M| = 896$ gemischte Zustände benötigt, plus $|P| = N = 8$ pure Zustände, bei denen die Prädiktoren die Vorhersage allein durchführen.

Als nächstes wird die Transitionsmatrix $A = \{p_{\hat{s},s}\}$ definiert. Sie bestimmt die Übergangswahrscheinlichkeiten zwischen den Zuständen. Im Prinzip kann diese Matrix durch ein Trainingsverfahren adaptiv aus den Daten der Zeitreihe bestimmt werden, zum Beispiel mit dem Baum-Welch Algorithmus [57]. In diesem Fall ist das jedoch wegen der immensen Größe der Matrix praktisch nicht möglich. Für die oben genannten, typischen Parameter zum Beispiel, besteht die Matrix A aus $(896+8)^2 = 817216$ Elementen, die mit dem Verfahren geschätzt werden müßten. Eine solch große Anzahl freier Parameter ist für jede adaptive Methode im allgemeinen tödlich, da in den meisten Fällen zu wenig Daten für eine ausreichend genaue Bestimmung der Parameter zur Verfügung stehen dürften. Daher wird hier eine feste Übergangsmatrix verwendet. Auf diese Weise fließt *a priori* Wissen über das dynamische System ein und der Suchraum wird eingeschränkt. In dem hier vorgestellten Ansatz sollen nur Lösungen berücksichtigt werden, bei denen die Dynamik schaltet oder kontinuierlich zwischen zwei Prädiktoren driftet, und zwar so, daß ein Schalten genauso wahrscheinlich ist wie eine (monotone) Drift zwischen zwei Prädiktoren. Genauer: die Wahrscheinlichkeit $p_{\hat{s},s} = p_{switch}$ wird für das Schalten zwischen zwei puren Zuständen $\hat{s}, s \in P$, angesetzt, wobei p_{switch} ein freier Parameter ist, und

$$p_{\hat{s},s} = (p_{switch})^{\frac{1}{R+1}} \tag{7.4}$$

wird für Mischungszustände verwendet, deren Mischungskoeffizienten sich nur um eine Stufe unterscheiden, d.h.

$$|a(\hat{s}) - a(s)| = \frac{1}{R+1}. \tag{7.5}$$

Alle anderen Zustandsänderungen werden durch Setzen von $p_{\hat{s},s} = 0$ verboten. Die Definitionen von $p(y \mid s)$ und π folgen den bereits bei ACE gemachten Annahmen. Entsprechend Gl. (4.1) und Gl. (4.3), werden eine Gaußverteilung

$$p(y \mid s) = Ke^{-\beta(y-g_s)^2}, \tag{7.6}$$

und gleichwahrscheinliche Anfangszustände, $\pi_s = |S|^{-1}$, angenommen.

Für das nun vollständig definierte Hidden Markov Modell eines driftenden dynamischen Systems kann schließlich der für Hidden Markov Modelle übliche Viterbi-Algorithmus [57] angewendet werden. Er liefert diejenige Sequenz von Zuständen (die Sequenz von Prädiktoren oder Mischungen von zwei Prädiktoren), die mit der größten Wahrscheinlichkeit eine gegebene Zeitreihe generiert haben könnte. In diesem Fall ergibt das die Drift-Segmentation der Zeitreihe. Dabei wird die Annahme berücksichtigt, daß Änderungen in der Dynamik entweder als seltenes Schalten oder als kontinuierliche Drift auftreten.

7.2 Der Drift-Segmentationsalgorithmus

Der Viterbi-Algorithmus kann besonders effizient und ohne numerische Probleme implementiert werden, wenn man anstelle der Wahrscheinlichkeit P die logarithmische Wahrscheinlichkeit $\log(P)$ einer gegebenen Zustandssequenz berechnet, wodurch mit Summen statt mit Produkten gerechnet werden kann [57]. Verwendet man darüber hinaus -log(P), so kann man $C = -\log(P)$ als die *Kosten* einer Sequenz betrachten.

Folgende Größen werden daher im folgenden verwendet: $C_s(t)$ sind die Kosten derjenigen Sequenz, mit der eine gegebene Zeitreihe $x_{t_0}, \ldots, x_t$ von dem Hidden Markov Modell *am wahrscheinlichsten* hätte erzeugt werden können, und deren Zustand zum Zeitpunkt t der Zustand s ist. $C_s(t)$ stellt folglich das Minimum der Kosten *aller möglichen* Sequenzen dar und ist die Summe der quadrierten Vorhersagefehler,

$$e_s(t) = (x_t - g_s(\vec{x}_{t-\tau}))^2, \tag{7.7}$$

zuzüglich der Summe der Transitionskosten, $T(\hat{s}, s) = -\log(p_{\hat{s},s})$, der wahrscheinlichsten Sequenz. Diese Sequenz kann zusammen mit $C_s(t)$ durch rekursives Berechnen der Kosten von t_0 bis t, für alle $s \in S$, gewonnen werden:

$$C_s(t_0) = e_s(t_0), \tag{7.8}$$

$$C_s(t') = e_s(t') + \min_{\hat{s} \in S} \left\{ C_{\hat{s}}(t'-1) + T(\hat{s}, s) \right\}, \quad t' = t_0 + 1, \ldots, t. \tag{7.9}$$

Das Induktionsprinzip dieser Berechnungsmethode wird in Abb. 7.1 erläutert (*dynamic programming*). Da der Zustand, in dem sich das System zum Zeitpunkt $t = T$ am Ende der betrachteten Zeitreihe befindet, beliebig sein darf, sind die Kosten der wahrscheinlichsten Sequenz schließlich gegebenen durch

$$C^* = \min_{s \in S} \left\{ C_s(T) \right\}, \tag{7.10}$$

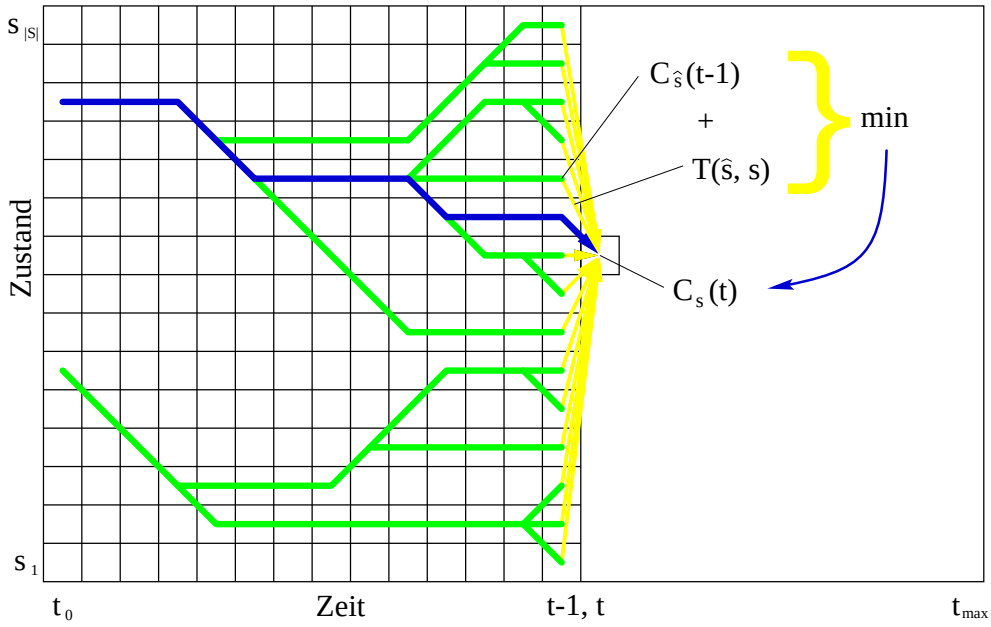


Abbildung 7.1: *Illustration des Viterbi-Algorithmus: Von allen möglichen Zustandssequenzen, die zum Zeitpunkt t in einem bestimmten Zustand s enden, sei die Sequenz $(q_0, \ldots, q_{t-1}, q_t), q_t = s$, diejenige, die am wahrscheinlichsten die Zeitreihe zwischen t_0 und t generiert haben könnte (dicke, schwarze Linie). Dann ist die kürzere Sequenz $(q_0, \ldots, q_{t-1})$ zwangsläufig diejenige Sequenz, die am wahrscheinlichsten die Zeitreihe zwischen t_0 und $t-1$ generiert haben könnte, und zwar von allen Sequenzen, die auch in q_{t-1} enden. Wenn umgekehrt die wahrscheinlichsten Sequenzen für alle Endzustände $\hat{s}$ zum Zeitpunkt $t-1$ bekannt sind (dicke, graue Linien), kann die wahrscheinlichste Sequenz, die zum Zeitpunkt t in s endet, demnach bereits schon durch alleiniges Betrachten dieser, um einen Zeitschritt kürzeren, Sequenzen gefunden werden (Induktionsschritt).*

Die zugehörige Folge von Zuständen, die an C^* beteiligt sind, stellt das Segmentationsergebnis dar: eine Folge von Prädiktoren oder Mischungen von Prädiktoren, die die Zeitreihe $x_{t_0}, \ldots, x_T$ mit der größten Wahrscheinlichkeit generiert haben könnte.

7.3 Driftendes Chaos

Zur Illustration des Drift-Segmentationsalgorithmus wird ein driftendes System der vier chaotischen Abbildungen aus Kapitel 4 untersucht. Gegeben ist eine Zeitreihe $\{x_t\}$, mit $x_{t+1} = f_l(x_t)$. Vier Operationsmoden, $l = 1, 2, 3, 4$, sind gegebenen durch

$$
\begin{array}{lll}
f_1(x) = & 4x(1-x)\,,\ x \in [0,1] & \text{(logistic map)} \\
f_2(x) = & f_1(f_1(x)) & \text{(double logistic map)} \\
f_3(x) = & 2x,\ \text{für } x \in [0,.5) \text{ und } 2(1-x),\ \text{für } x \in [.5,1] & \text{(tent map)} \\
f_4(x) = & f_3(f_3(x)) & \text{(double tent map)}
\end{array}
$$

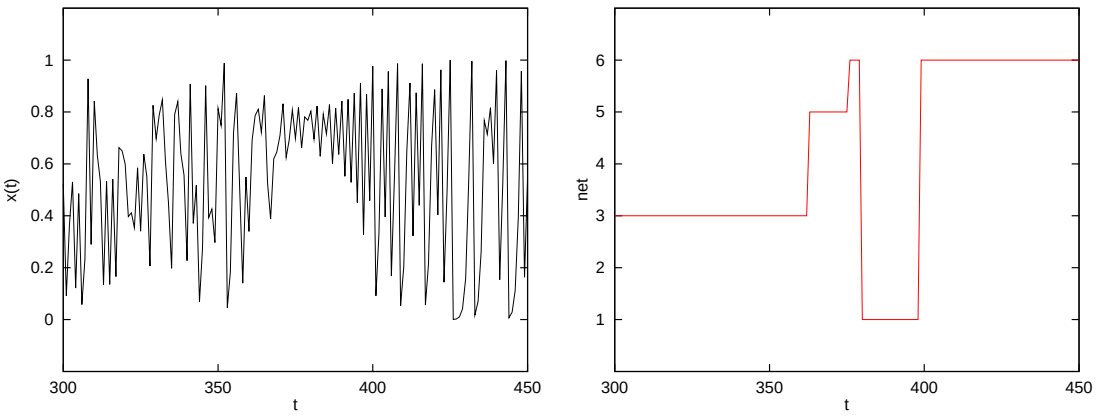
Während der ersten 50 Zeitschritte wird f_1 iteriert, mit $x_0 = 0.5289$. Nach $t = 50$ Zeitschritten driftet die Dynamik von f_1 nach f_2,

$$f(x_t) \;=\; (1 - a(t))\, f_1(x_t) \;+\; a(t)\, f_2(x_t), \qquad a(t) = \frac{t - t_a}{t_b - t_a}, \tag{7.11}$$

mit $t_a = 50$ und $t_b = 100$. Die Drift ist hier also linear über der Zeit und dauert weitere 50 Zeitschritte. Danach läuft die Dynamik für die nächsten 50 Zeitschritte stationär in f_2, woraufhin das System linear nach f_3 driftet, usw. Bei $t = 350$ driftet das System schließlich von f_4 nach f_1 zurück und der Zyklus startet bei $t = 400$ von vorn, siehe Abb. 7.2(a).

Für die ersten 1200 Daten dieser Zeitreihe wurde das ACE Verfahren mit einem Ensemble von sechs RBF-Prädiktoren $f_i(x_t)$, $i = 1, ..., 6$, mit 20 Basisfunktionen durchgeführt. Der verwendete Tiefpaß hat die Größe $\Delta = 3$. Nach dem Training haben sich vier Prädiktoren (6, 2, 4 und 3) auf je eine der vier chaotischen Abbildungen (f_1, f_2, f_3 bzw. f_4) spezialisiert. Die übrigen beiden Prädiktoren (1 und 5) haben sich dagegen auf die Drift-Abschnitte spezialisiert. Die Drift kann jedoch durch ein Schalt-Modell nicht angemessen repräsentiert werden (Abb. 7.2(b)).

Als nächstes wurde der Drift-Segmentationsalgorithmus auf alle sechs Prädiktoren angewendet. Er reproduziert die Entwicklung der Dynamik perfekt, auch



(a) (b)

Abbildung 7.2: *(a) Ein Ausschnitt aus den Trainingsdaten, der die Drift von f_4 nach f_1 zeigt. Zunächst wird f_4 von $t = 300$ bis $t = 350$ iteriert. Dann erfolgt eine Drift von f_4 nach f_1 zwischen $t = 350$ und $t = 400$. Nach $t = 400$ wird f_1 iteriert. (b) Die Segmentation nach dem ACE Training (auf der Grundlage der tiefpaßgefilterten Fehler) kann die Drift nicht repräsentieren. Die stationären Abschnitte, f_4 in $[300, 350]$ und f_1 in $[400, 450]$, werden zwar angemessen von den Prädiktoren 3 bzw. 6 repräsentiert, der dazwischenliegende Drift-Abschnitt wird aber zwischen vier Prädiktoren aufgeteilt, einschließlich Netz 1 und 5. Eine bessere Repräsentation der Drift ist hier mit einer schaltenden Segmentation nicht möglich.*

auf den Testdaten im Interval $[1200, 2400]$, wie die Abb. 7.3(a) für die Auflösung $R = 32$ zeigt: es wird eine lineare Drift zwischen vier Operationsmoden gefunden. Diese Driftform wurde hierbei, im Gegensatz zu dem Verfahren in Kapitel 6, dem Algorithmus nicht vorgegeben, sondern allein aus den Daten der Zeitreihe bestimmt. Abb. 7.3(b) zeigt den Treppen-Effekt bei einer zu kleinen Auflösung $R = 3$.

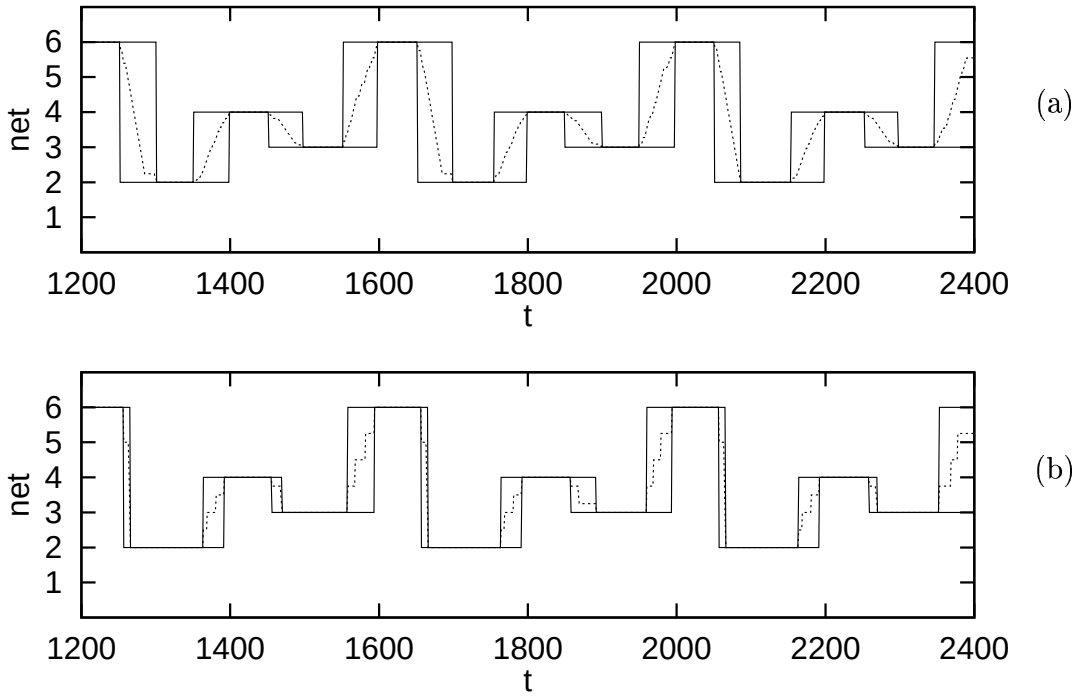


Abbildung 7.3: *(a) Das Resultat des Drift-Segmentationsalgorithmus für die Testdaten im Interval* $[1200, 2400]$. *Die Auflösung ist* $R = 32$. *Dargestellt ist die Sequenz von Prädiktoren als Funktion über der Zeit* t. *Dabei zeigen die Boxen Mischungen von zwei Prädiktoren an. Die beiden beteiligten Prädiktoren werden hierbei durch die obere und untere Begrenzung der Box festgelegt. Die gepunktete Linie in einer Box zeigt darüber hinaus die zeitliche Entwicklung des Mischungskoeffizienten* $a(t)$ *an. Zum Beispiel ist zwischen* $t = 1350$ *und* 1400 *eine Drift von Netz* 2 *zu Netz* 4 *dargestellt, die sehr schön die vorhandene lineare Drift des Systems wiedergibt. Tatsächlich gibt die gesamte Segmentation der Testdaten das zeitliche Verhalten des dynamischen Systems nahezu perfekt wieder. (b) Die Segmentation der Testdaten bei Verwendung einer niedrigeren Auflösung* $R = 3$. *Die lineare Drift kann nunmehr lediglich durch* 3 *Mischungsstufen angenähert werden, da der Algorithmus keine weiteren Mischungsverhältnisse mehr berücksichtigt.*

7.4 Driftende Mackey-Glass Dynamik

In diesem Abschnitt wird die Anwendbarkeit der Drift-Analyse-Methode auf ein *hochdimensionales* chaotisches System untersucht. Dazu wird ein Mackey-Glass System mit *driftender Dynamik* betrachtet.

Mit Hilfe der Mackey-Glass Differentialgleichung [38] (vgl. Abschnitt 3.2.2)

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-t_d)}{1+x(t-t_d)^{10}} \tag{7.12}$$

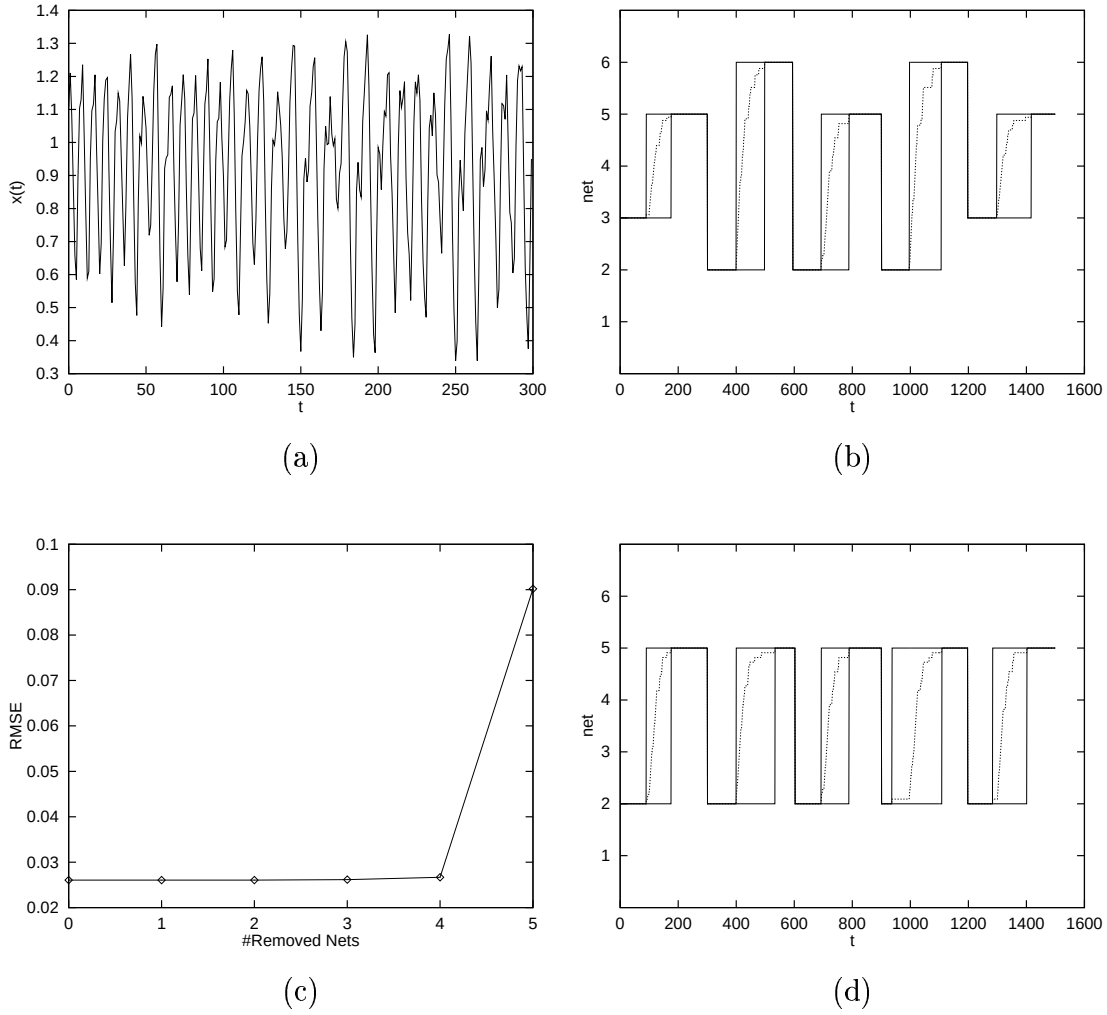
werden zwei Operationsmoden, A und B, durch Verwendung zweier unterschiedlicher Parameter, $t_d = 17$ für A und $t_d = 23$ für B, festgelegt. Nachdem das System 100 Zeitschritte in Modus A operiert (in Bezug auf eine Abtastrate $\tau = 6$), driftet die Dynamik zu Modus B. Die Drift dauert wiederum 100 Zeitschritte. Sie erfolgt durch das Mischen der Gleichungen für $t_d = 17$ und 23 während der Integration von Gl. (7.12). Die Mischung wird gemäß Gl. (7.1) unter Verwendung eines exponentiell abfallenden Mischungsverhältnisses erzeugt,

$$a(t) = \exp\left(\frac{-4\,t}{100}\right), \quad t = 1, \ldots, 100. \tag{7.13}$$

Nach der Driftphase arbeitet das System für weitere 100 Zeitschritte stationär in Modus B und *schaltet* schließlich bei $t = 300$ zurück zu Modus A, woraufhin der Zyklus wieder von vorn anfängt (Abb. 7.4(a)).

Der ACE Algorithmus wurde auf die ersten 1500 Datenpunkte der so generierten Zeitreihe angewendet. Dazu wurden sechs RBF-Prädiktoren $f_i(\vec{x}_t)$, $i = 1, ..., 6$, mit je 40 Basisfunktionen verwendet. Die Einbettungsdimension ist $m = 6$.

Nach dem Training haben sich die Netze 2 und 3 auf Modus A und die Netze 5 und 6 auf Modus B spezialisiert, wie in der Driftsegmentation in Abb. 7.4(b) zu sehen ist. Es zeigt sich, daß vier der sechs Netze enfernt werden können ohne den mittleren Vorhersagefehler (RMSE) zu erhöhen (Abb. 7.4(c), siehe Abschnitt 5.1). Daraus läßt sich schließen, daß zwei Prädiktoren ausreichen, um das dynamische System vollständig zu beschreiben. Die Segmentation mit den Netzen 2 und 5 reproduziert dann sehr schön die zeitliche Struktur der Dynamik, wie in Abb. 7.4(d) zu sehen ist.



(a) (b) (c) (d)

Abbildung 7.4: *(a) Die driftende Mackey-Glass Zeitreihe. Das dynamische System operiert in Modus A während der ersten 100 Zeitschritte. Dann driftet die Dynamik zu Modus B während der nächsten 100 Zeitschritte und bleibt dann stationär in Modus B. Nach* $t = 300$*, schaltet das System zurück in Modus A und der Zyklus beginnt von vorn. (b) Die resultierende Drift-Segmentation verwendet vier Netze. Zwei Experten haben sich auf Modus A, zwei weitere auf Modus B spezialisiert. (c) Die Zunahme des Vorhersagefehlers, wenn Prädiktoren nach und nach entfernt werden. Es können bis zu vier Prädiktoren aus dem Ensemble entfernt werden ohne den Vorhersagefehler zu erhöhen. (d) Die beiden verbleibenden Prädiktoren modellieren das dynamische Verhalten des Systems nahezu perfekt.*

Kapitel 8

Anwendung: Analyse von physiologischen Wach/Schlaf-Daten

Schlaf ist ein physiologischer Zustand, in dem sich ein Mensch etwa während eines Drittels seiner Lebenszeit befindet. Seine Bedeutung wird oft unterschätzt. Alle Schlafstörungen beeinflußen die Wachsamkeit, entweder direkt (gestörte Wachsamkeit) oder indirekt (gestörter Schlaf). Dem Übergang zwischen Wachen und Schlafen kommt dabei besondere Bedeutung zu. Vom neurophysiologischen Standpunkt aus betrachtet, erfolgt dabei eine Reorganisation des neuronalen Netzes in der retikularen Formation des Hirnstamms, welche für die Regulation und Integration der kardiovaskulären, respiratorischen und somatomotorischen Systeme, sowie der Wachsamkeit verantwortlich ist [35]. Diese Reorganisation kann man als Übergang in einen anderen „Operationsmodus“ betrachten.

Physiologische Studien in der retikularen Formation des Hirnstamms haben gezeigt, daß die Neuronen dieser Netzwerke ihren Operationsmodus in Abhängigkeit von der Art des afferenten Inputs sowie vom Grad der Neuronenaktivität verändern [35, 8]. Dies gilt für die Entladungsmuster einzelner Neuronen, aber auch für die Konnektivität zwischen den Neuronen [25]. Der Übergang zwischen den verschiedenen dynamischen Moden des neuronalen Netzes korrespondiert mit dem Übergang zwischen verschiedenen physiologischen Zuständen. Eine besonders prägnante Zustandsänderung, die von großem praktischen Interesse ist, ist der Übergang vom Wachen zum Schlafen.

Beim Einschlafen erfahren praktisch alle physiologischen Funktionen eine fundamentale Veränderung: das Herz schlägt langsamer, Blutdruck und Muskeltonus vermindern sich, das Bewußtsein wird getrübt und die kortikale Aktivität verlangsamt sich. Die Schlaf-Polysomnographie beinhaltet daher die simultane Aufzeichnung mehrerer Organsysteme, wie zum Beispiel kortikale Aktivität (Elektroenzephalogramm, EEG), Augenbewegungen (Elektrookulogramm, EOG), Herzaktivität (Elektrokardiogramm, EKG) und Atmung. Alle diese Größen können relativ einfach gemessen werden.

Die Analyse des Einschlafzeitpunkts erfolgt klassischerweise durch das EEG. In der klinischen Praxis wird Schlaf seit den Arbeiten von Davis und Loomis [8] allein mit der Verlangsamung der kortikalen Aktivität identifiziert. Dies ist jedoch eine eingeschränkte Sicht: Von Economo war der erste, der zwischen kortikalem Schlaf und Organschlaf unterschied [25]. Es sei an dieser Stelle daran erinnert, daß sich zum Beispiel auch die Atmungsmuster während des Einschlafens und zwischen den Schlafphasen verändern [39]. Dies wurde von Magnussen bereits herausgefunden bevor die Ära des EEG begann [66].

Die derzeitige klinische Analyse der Schlafstadien beschränkt sich in der zeitlichen Auflösung auf den Bereich 10–20 Sekunden und verwendet insbesondere EEG Aufzeichnungen. Folglich sind sie von geringem Wert, wenn Auflösungen unter 10 Sekunden erforderlich sind oder auch dann, wenn artefaktfreie EEG Messungen nicht möglich sind.

Unsere Absicht ist es, den Einschlafvorgang und seine Dynamik mit hoher zeitlicher Auflösung in verschiedenen Organsystemen zu untersuchen. Es ist bekannt, daß Schlaf in den unterschiedlichen Organsystemen nicht mit absoluter Synchronität stattfindet [58]. Durch eine höhere Zeitauflösung hoffen wir daher neue Einsichten die dynamische Struktur des Schlafs und seiner Störungen zu erlangen.

Im folgenden wird die Analyse von EEG, EOG und Atmung mit der zuvor beschriebenen Methode der *Competing Experts* beschrieben. Die Daten stammen von Aufzeichungen des Mittagsschlafs gesunder Probanden, die am Institut für Physiologie der Freien Universität Berlin gemacht wurden. Die automatische Analyse verwendet dabei keinerlei medizinisches Expertenwissen und basiert jeweils auf einzelnen Meßgrößen, da die zeitlichen Verzögerungen in den verschiedenen physiologischen Größen nicht berücksichtigt werden kann. Es wird zum einen eine harte Segmentation (schaltende Dynamik) berechnet, insbesondere für den Vergleich mit der manuellen Segmentation eines Mediziners, und zum anderen

wird auch die Drift-Segmentation bestimmt, um die dynamische Struktur der Übergänge aufzulösen. Hierbei ist noch anzumerken, daß die manuelle Segmentation, im Gegensatz zur maschinellen Untersuchung, immer unter Verwendung von sechs physiologischen Größen erstellt wurde, nämlich EEG, EOG, EKG, Herzrate, Blutdruck und Atmung.

8.1 Schaltende Dynamik

8.1.1 Segmentation von EOG und Atmung

Abb. 8.1 zeigt Aufzeichnungen von (a) der Augenbewegung (EOG) und (b) der Atmung, die im folgenden untersucht werden. Der Übergang vom Wachen zum Schlafen erfolgt bei ca. $t = 5000$ (Auflösung 0.1 s), was durch typische langsame Augenbewegungen (Slow Eye Movement, SEM) angezeigt wird, die sich mit einiger Übung aus den EOG-Daten ablesen lassen. Der entsprechende Übergang läßt sich in der Atmung jedoch nur schwer festlegen. Die beiden Signale wurden mit dem *hard competition*-Ansatz analysiert. Die vier angesetzten RBF-Netze haben 50 Zentren, die Einbettungsdimension ist $d = 2$ und die Abtastung $\tau = 5$. Die resultierenden Segmentationen, Abb. 8.1(c) für das EOG und Abb. 8.1(d) für die Atmung, finden ähnliche Zeitpunkte für den Wach/Schlaf-Übergang und sind in guter Übereinstimmung mit der Diagnose eines medizinischen Experten.

Um systematischere Ergebnisse zu erhalten, wurden Atmungsdaten von fünf Probanden, von denen jeweils drei Aufzeichnungen des Mittagsschlafs vorhanden waren, analysiert und verglichen. Alle Probanden waren gesunde, 20-40 Jahre alte Nichtraucher. Jede der insgesamt 15 Zeitreihen wurde, wie oben bereits beschrieben, mit dem *hard competition* Verfahren analysiert, diesmal wurden jedoch sechs Prädiktoren verwendet. Die für die einzelnen Zeitreihen erhaltenen RBF-Experten wurden dann dazu verwendet auch jeweils sämtliche anderen vorhandenen Zeitreihen zu segmentieren, ohne die Experten neu zu trainieren. So läßt sich überprüfen, ob die Experten auch dazu geeignet sind, sinnvolle Segmentationen auf neuen Daten desselben oder sogar eines anderen Probanden zu liefern. Es wird also die Generalisationsfähigkeit der RBF-Experten getestet. Die Ergebnisse lassen sich somit in drei Klassen einteilen: 1. Segmentationen der Zeitreihe, auf der die Experten trainiert wurden, 2. Segmentationen der anderen beiden

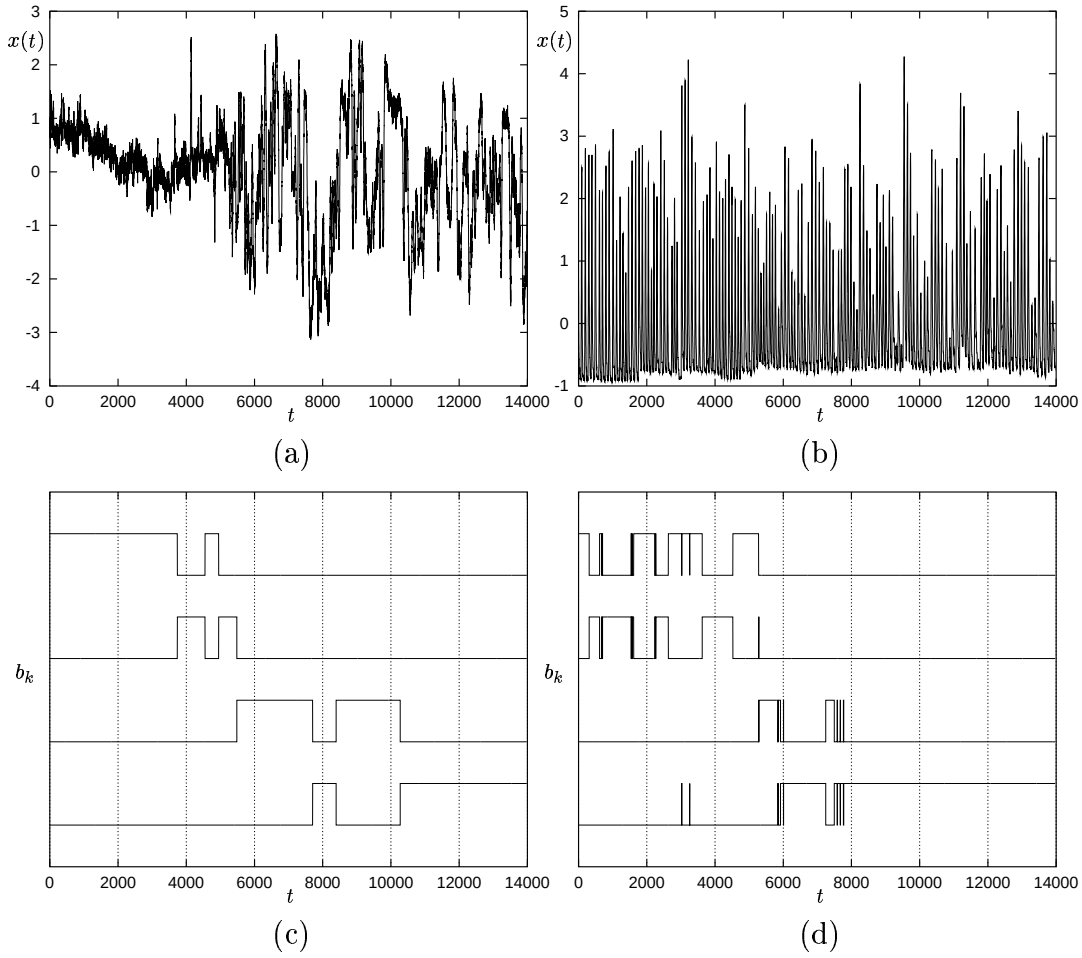


Abbildung 8.1: *Die Segmentation von (a) Augenbewegungen und (b) Atmungssignal durch vier RBF-Experten ist in (c) und (d) dargestellt. In beiden Fällen teilen sich je zwei Experten die Wach- bzw. Schlafphase. Der Übergang zwischen Wachen und Schlafen nach ca. 5000 Zeitschritten (500 Sekunden) wird aus beiden Signalen richtig erkannt.*

Klasse	korrekt	S1/S2	W1/W2	W2/S1	W2/S2	W1/S1	W1/S2	1+2	1+2+3
1.	69.70	9.38	7.25	9.16	1.01	2.82	0.65	79.08	86.33
2.	51.80	12.12	11.32	12.85	1.58	7.57	2.72	63.92	75.24
3.	34.02	11.70	13.81	15.92	4.72	14.87	4.93	45.72	59.53

Tabelle 8.1: *Vergleich der Segmentationsergebnisse für die Atmung mit den manuellen Segmentationen eines Mediziners, in drei Klassen aufgeschlüsselt (Erläuterung siehe Text). Alle Angaben in Prozent.*

Zeitreihen *desselben* Probanden, ohne neu zu trainieren, und 3. Segmentationen der Zeitreihen aller übrigen Probanden, ebenfalls ohne neu zu trainieren.

Die Übereinstimmung der Segmentationsergebnisse mit den manuellen Segmentationen eines Mediziners wird in Tabelle 8.1 für die drei Klassen detailliert aufgeschlüsselt. Alle Angaben sind in Prozent (100% = gesamte Zeitreihe). Die manuelle Segmentation unterscheidet zwischen Wachen mit geöffneten und geschlossenen Augen (W1 bzw. W2), sowie Schlafstadium I und II (S1 bzw. S2). Andere Schlafstadien werden bei keinem der Experimente erreicht, da der Mittagsschlaf dafür zu kurz ist. Die Zuordnung (*labeling*) der einzelnen RBF-Experten zu jeweils einer dieser vier Wach/Schlaf-Stadien erfolgt nach dem Training von Hand, da die *Bedeutung* der gelernten Dynamiken nicht allein aus den Daten abgeleitet werden kann, sondern zusätzliches Wissen erfordert.

Die Spalte „korrekt“ gibt den Prozentsatz der exakten Übereinstimmung zwischen automatischer und manueller Segmentation bezüglich der vier Stadien an. Die Spalte „S1/S2“ gibt den prozentualen Anteil der Verwechselung von S1 und S2 an. Die folgenden Spalten geben entsprechend die anderen möglichen Abweichungen an (W1/W2, W2/S1, etc.). Spalte „1+2“ gibt die prozentuale Übereinstimmung an, wenn man die Abweichung S1/S2 nicht als Fehler bewertet (Summe aus Spalte 1 und 2). Spalte „1+2+3“ bewertet zusätlich die Abweichung W1/W2 nicht als Fehler und gibt somit die Übereinstimmung in der einfacheren Einteilung in Wachen (W1 und W2) und Schlafen (S1 und S2) an (Summe der Spalten 1, 2 und 3).

Offensichtlich lassen sich die trainierten RBF-Experten nicht zur Untersuchung neuer Daten, auf denen nicht zuvor trainiert wurde, verwenden, ohne einen deutlichen Anstieg des Segmentationsfehlers zu erhalten. Dies gilt sowohl für Aufzeichnungen beim selben Probanden, als auch insbesondere für Aufzeichnungen

bei anderen Probanden. Es ist also von Vorteil, die Prädiktoren für jede Zeitreihe neu zu trainieren, was im übrigen auch keinen großen, zusätzlichen Rechenaufwand darstellt. Auf den trainierten Zeitreihen ergibt sich dagegen eine gute Übereinstimmung mit dem manuellen Ergebnis (Klasse 1).

Eine harte Segmentation des Atmungssignals (Zeitreihe rs11) ist beispielhaft in Abb. 8.2, oben, dargestellt. Man sieht, daß sich zwei RBF-Prädiktoren, Netz 2 und Netz 6, auf die Wachphasen W1 spezialisiert haben, zwei weitere Prädiktoren, 3 und 4, sind für die Schlafphasen zuständig, und Netz 1 modelliert jeweils den Übergang zwischen diesen Phasen (W2). Es gibt in diesem Fall jedoch keine klare Unterscheidung zwischen S1 und S2. Um die Generalisierungsfähigkeit zu testen, wurden die auf rs11 trainierten Netze u.a. zur Segmentation von Datensatz rs13 (gleicher Proband) verwendet (Abb. 8.3, oben). Wieder sind Netz 2 und Netz 6 für die Wachphase zuständig und die Netze 3 und 4 für die Schlafphase. Netz 1 ist meist für die kurzen Aufwachphasen zuständig (W2), erscheint aber auch mehrmals, wenn in der manuellen Segmentation S1 oder S2 angezeigt werden.

Dieselben Segmentationen sind in Abb. 8.4 und Abb. 8.5 noch einmal in der Überlagerung von automatischer und manueller Segmentation zu sehen, wobei die Netze den entsprechenden Phasen zugeordnet wurden. In Abb. 8.4 (rs11) sind 76% völlig korrekt segmentiert. Wenn man nur die Einteilung in Wachen und Schlafen betrachtet sind es sogar 96%. Die Differenz rührt daher, daß die Netze nicht zwischen S1 und S2 unterscheiden können. Dies aus dem Atmungssignal zu erkennen, ist jedoch prinzipiell äußerst schwierig. Die manuelle Segmentation basiert, wie bereits erwähnt, dagegen auf der Betrachtung aller physiologischer Daten, insbesondere auch dem EEG. Wie zu Beginn bereits erwähnt wurde, können dadurch auch *systematische* Abweichungen zwischen manueller und maschineller Segmentation herrühren, da sich die Stadien in EEG und Atmung nicht synchron verändern. Das Ergebnis für die Segmentation der Testdaten rs13 beträgt entsprechend 58% bzw. 87% (Wach/Schlaf). Für die reine Wach/Schlaf-Unterscheidung ergibt sich also auch hier eine gute Übereinstimmung.

8.1.2 EEG Segmentation

Im EEG kann der Einschlafzeitpunkt genauer und einfacher bestimmt werden als im Atmungssignal. Es ist hier auch einfacher, die Schlafstadien zu unterscheiden. Dies zeigen die Segmentationsergebnisse für das EEG in Tabelle 8.2, wobei in diesem Fall jedoch nur Daten von einem Probanden vorlagen.

Exp.	korrekt	S1/S2	W1/W2	W2/S1	W2/S2	W1/S1	W1/S2	1+2	1+2+3
eeg11									
11	70.87	21.25	4.32	0.00	0.64	0.91	2.01	92.12	96.44
12	67.31	10.51	13.27	0.00	1.47	7.44	0.00	77.82	91.09
13	60.93	21.47	8.27	3.78	2.19	3.37	0.00	82.40	90.67
eeg12									
11	67.04	27.51	4.43	1.01	0.00	0.00	0.00	94.55	98.99
12	83.63	2.56	3.01	3.47	0.00	7.32	0.00	86.19	89.21
13	57.65	26.39	7.53	2.67	0.00	5.76	0.00	84.04	91.58
eeg13									
11	72.77	21.45	4.29	0.71	0.78	0.00	0.00	94.22	98.51
12	69.82	8.15	2.21	4.36	1.42	14.04	0.00	77.97	80.18
13	73.92	12.87	8.32	1.53	1.17	2.19	0.00	86.78	95.10

Tabelle 8.2: *Vergleich der EEG Segmentationsergebnisse mit den manuellen Segmentationen eines Mediziners. Alle Angaben in Prozent. Für jede der drei EEG Zeitreihen (eeg11, eeg12, eeg13) wurden RBF-Experten trainiert und anschließend jeweils zur Segmentation aller drei Datensätze verwendet. Die Segmentationen unterscheiden zwischen Wachen mit geöffneten und geschlossenen Augen (W1 bzw. W2), sowie Schlafstadium I und II (S1 bzw. S2). Die Spalte „korrekt“ gibt den Prozentsatz der Übereinstimmung zwischen automatischer und manueller Segmentation bezüglich aller Stadien an. Die Spalte „S1/S2“ gibt den prozentualen Anteil der unterschiedlichen Zuordnung von S1 und S2 an. Die folgenden Spalten geben entsprechend die anderen möglichen Abweichungen an (W1/W2, W2/S1, etc.). Spalte „1+2“ gibt die Übereinstimmung an, wenn man die Abweichung S1/S2 nicht als Fehler bewertet (Summe aus Spalte 1 und 2). Spalte „1+2+3“ bewertet zusätlich die Abweichung W1/W2 nicht als Fehler und gibt somit die Übereinstimmung in der Einteilung in Wachen (W1 und W2) und Schlafen (S1 und S2) an.*

Abb. 8.6 zeigt ein Beispiel für eine EEG Segmentation (eeg11). Es wurden acht RBF-Netze mit sechs Zentren und einer Einbettung $d = 4$ verwendet. Man sieht, daß sich die Netze 6 und 7 auf die Wachphasen spezialisiert haben, jedoch ohne angemessen zwischen W1 und W2 zu unterscheiden. Netz 2 hat sich auf Artefakte spezialisiert und Netz 4 auf die Schlafphase, wobei wiederum nicht zwischen S1 und S2 unterschieden wird. Der Einschlafzeitpunkt wurde jedoch sehr genau gefunden. Auch das kurze Aufwachen bei $t = 7000$ wurde gut erkannt. Die Überlagerung der beiden Segmentationen ist in Abb. 8.8 dargestellt: 70.87% korrekte Segmentation bzw. 96.44% bei einfacher Wach/Schlaf-Unterscheidung (vgl. Tabelle 8.2).

Um auch hier die Generalisierungsfähigkeit zu testen, wurde u.a. das EEG-Signal eeg13 von den Netzen segmentiert, die zuvor auf eeg11 trainiert wurden (Abb. 8.7, oben). Wieder sind Netz 6 und 7 für die Wachphase zuständig, der Einschlaf- und Aufwachzeitpunkt wird sehr genau gefunden, und die kurzen Aufwachphasen zwischendurch werden teilweise gefunden. S1 und S2 werden wieder durch Netz 4 repräsentiert. Netz 5, das in eeg11 fast gar nicht beteiligt war (W2), zeigt nun ebenfalls Aufwachphasen an. Die Überlagerung der beiden Segmentationen ist in Abb. 8.9 dargestellt: 60.93% korrekte Segmentation bzw. 90.67% bei einfacher Wach/Schlaf-Unterscheidung (Tabelle 8.2).

Abgesehen von der Unterscheidung von S1 und S2, sind die Übereinstimmungen mit den manuellen Segmentationen recht gut, insbesondere was den Zeitpunkt des Einschlafens und Aufwachens angeht.

8.2 Prädiktion

Nachdem eine vernünftige Segmentation in Wach- und Schlafphasen erreicht wurde, stellt sich nun die Frage, welche Dynamiken die Prädiktoren tatsächlich gelernt haben bzw. wie gut die Signaldynamik von den Prädiktoren gelernt wurde.

Um die Vorhersageleistung beispielsweise der Prädiktoren für das Atmungssignal zu illustrieren, wurden ein Wach- und ein Schlaf-Prädiktor auf einen Wach- bzw. Schlafzeitpunkt der Zeitreihe aufgesetzt und auf den selbst vorhergesagten Daten iteriert. Die Iterationen liefern dann Langzeitvorhersagen des Signals. Dies zeigt Abb. 8.10 für je 500 Zeitschritte (50 Sekunden) des Atmungssignals. Obwohl die Prädiktoren nur auf die Vorhersage des nächsten Datenpunkts trainiert wurden,

haben sie die jeweilige Dynamik so gut gelernt, daß sinnvolle Vorhersagen noch für mindestens 180 Zeitschritte (18 Sekunden) für die Wachdynamik und 350 Zeitschritte (35 Sekunden) für die Schlafdynamik möglich sind, bevor die Signale jeweils aus der Phase laufen. Jeder der beiden Prädiktoren schwingt sich auf ein periodisches Signal ein, das jeweils in Wellenform und Frequenz dem Wach- bzw. Schlafsignal entspricht. Lediglich die Amplitudenschwankungen der realen Signale werden dabei nicht berücksichtigt.

Für das EEG ergibt sich dagegen nur eine sehr grobe Repräsentation der Dynamik, da die RBF-Netze hier nur sechs Zentren besitzen. Diese Beschränkung ist erforderlich, um zu verhindern, daß die Netze den hohen Rauschanteil im EEG mitlernen. Die iterierte Vorhersage stimmt deshalb schon nach zwei bis drei Zeitschritten nicht mehr mit dem EEG Signal überein (Abb. 8.11). Daß dennoch sehr gute Segmentationsresultate mit diesen Netzen erzielt werden konnten (Tabelle 8.2), ist dadurch zu erklären, daß die Segmentation nur eine *Klassifikation* verschiedener Dynamiken erfordert, nicht aber deren exakte Rekonstruktion. Die Einzeldynamiken müssen für die Segmentation also lediglich so modelliert werden, daß man sie voneinander unterscheiden kann. Verwendet man anstelle der RBF-Netze jedoch lediglich *lineare* Prädiktoren, so erhält man Segmentationen, in denen die Struktur der Dynamik nicht mehr so gut in Wach- und Schlafzustände aufgelöst wird, wie es bei Verwendung der RBF-Netze der Fall ist.

8.3 Driftende Dynamik

Im folgenden wird der Verlauf der dynamischen Drift im Atmungssignal und im EEG untersucht. Wir verwenden dazu den Drift-Segmentationsalgorithmus mit adaptivem Mischungskoeffizienten aus Kapitel 7.

8.3.1 Atmungsdaten

Die Drift-Segmentation von Atmungssignal rs11 in Abb. 8.2 zeigt in der Wachphase zuerst eine Mischung von Netz 6 und Netz 2, mit starker Gewichtung von Netz 2 (Phase W1). Dann erfolgt eine sehr langsame Drift von Netz 2 zu Netz 3, das den Schlafzustand repräsentiert. Am Einschlafzeitpunkt erfolgt dann eine

schnelle Änderung des Mischungsverhältnisses zugunsten von Netz 3, das schließlich ab $t = 5000$ die Dynamik allein modelliert. Das erste kurze Aufwachen wird in der Atmung nicht erkannt, beim zweiten wird eine vorübergehende Drift zu Netz 5 angezeigt. Das Aufwachen am Ende wird als Drift von Netz 3 zu Netz 6 dargestellt, wobei Netz 6 den Wachzustand repräsentiert. Dabei ist Abb. 8.2 nicht so zu interpretieren, daß die Drift auch über die Netze 4 und 5 läuft: An einer Drift sind grundsätzlich immer nur die beiden Netze beteiligt, die durch die obere und untere horizontale Begrenzungslinie angegeben sind.

Insgesamt wird die Struktur der Dynamik gut wiedergegeben. Es ist jedoch festzustellen, daß eine Drift über weite Bereiche der gesamten Zeitreihe gefunden wird, was ein wenig im Widerspruch zu der Annahme des ACE-Algorithmus steht, daß es hinreichend lange stationäre Abschnitte für die Einzeldynamiken gibt. Hier sollte man sich über die Grenzen des Algorithmus im Klaren sein: eine kontinuierliche Drift über die gesamte Zeitreihe kann nicht angemessen repräsentiert werden. Sie kann bestenfalls näherungsweise beschrieben werden, siehe hierzu auch die Diskussion in Kapitel 9.

Um die Generalisierungsfähigkeit zu testen, wurde wiederum Atmungssignal rs13 mit den Prädiktoren von rs11 segmentiert. Hierbei wird schließlich nur noch Wachen und Schlafen unterschieden (Abb. 8.3). Darüber hinaus ist praktisch keiner der Prädiktoren mehr allein für die Vorhersage des Signals zuständig. Wachen und Schlafen werden jeweils durchgehend als Mischungen zweier Prädiktoren repräsentiert. Dies zeigt, daß die von den Prädiktoren auf rs11 gelernten Dynamiken in den Daten rs13 nicht mehr in der gelernten Form vorhanden sind. Die Prädiktoren eignen sich also in diesem Fall nicht sehr gut zur Segmentation der neuen Daten und sollten daher neu trainiert werden.

8.3.2 EEG

Die Drift-Segmentation von EEG-Signal eeg11 in Abb. 8.6 (Mitte) zeigt eine beeindruckend detaillierte Struktur im EEG. Der Einschlafzeitpunkt, entsprechend der manuellen Segmentation bei $t = 4000$, wird als exponentiell abfallende Driftkurve von einem Wach-Prädiktor, Netz 7, zu einem Schlaf-Prädiktor, Netz 4, modelliert. Im Gegensatz dazu wird das Aufwachen bei $t = 9000$ durch eine leichte Drift zu Netz 7 eingeleitet, die bis zum Aufwachzeitpunkt $t = 9500$ beibehalten wird. Dort springt das Mischungsverhältnis schlagartig zugunsten von

Netz 7 um. Schließlich wird die Wachphase ab $t = 10000$ durch eine Mischung der Wach-Prädiktoren 2 und 7 repräsentiert. Diese Segmentation ist in guter Übereinstimmung mit der manuellen Segmentation und zeigt darüber hinaus eine interessante Struktur der EEG-Dynamik in hoher zeitlicher Auflösung.

Um auch hier die Generalisierungsfähigkeit des Drift-Algorithmus zu überprüfen, wurden die RBF-Prädiktoren von eeg11 auf eeg13 angewendet (Abb. 8.7). Das Resultat zeigt wieder sehr interessante Übereinstimmungen mit der manuellen Segmentation. Insbesondere werden die beiden kurzen Aufwachphasen vor und nach $t = 10000$ sehr gut wiedergegeben. W1 wird wie zuvor als Mischung von Netz 2 und 7 repräsentiert. S1 und S2 werden ebenfalls wieder von Schlaf-Netz 4 modelliert. Bei $t = 6500$ wird das kurze Aufwachen des Probanden erkannt. Das letzte Aufwachen schließlich bei $t = 12000$ wird als kurzer, exponentieller Drift-Übergang von Netz 4 zu Netz 7 dargestellt.

8.3.3 Interpretation der Drift-Segmentation

Im Vergleich mit den beiden Schalt-Segmentationen (manuell und maschinell) zeigt die Drift-Segmentation wesentlich mehr Details der dynamischen Struktur der Signale, insbesondere in Bezug auf die Geschwindigkeit und Form der Veränderungen. Es ist ein naheliegender Gedanke, den Mischungskoeffizienten zwischen Wach- und Schlaf-Prädiktor als Maß der Wachsamkeit zu interpretieren. Ob diese Interpretation physiologisch gerechtfertigt ist, ist ein interessanter Ausgangspunkt für weitere Untersuchungen, für die allerdings neue, speziell auf diese Fragestellung ausgerichete Schlaf-Experimente nötig sind. Das Auffinden eines verläßlichen Wachsamkeitsmaßes kann jedenfalls in vielen Bereichen von großer Bedeutung sein, zum Beispiel bei der Diagnose von Schlafstörungen oder der Wachsamkeitsüberwachung von Autofahrern.

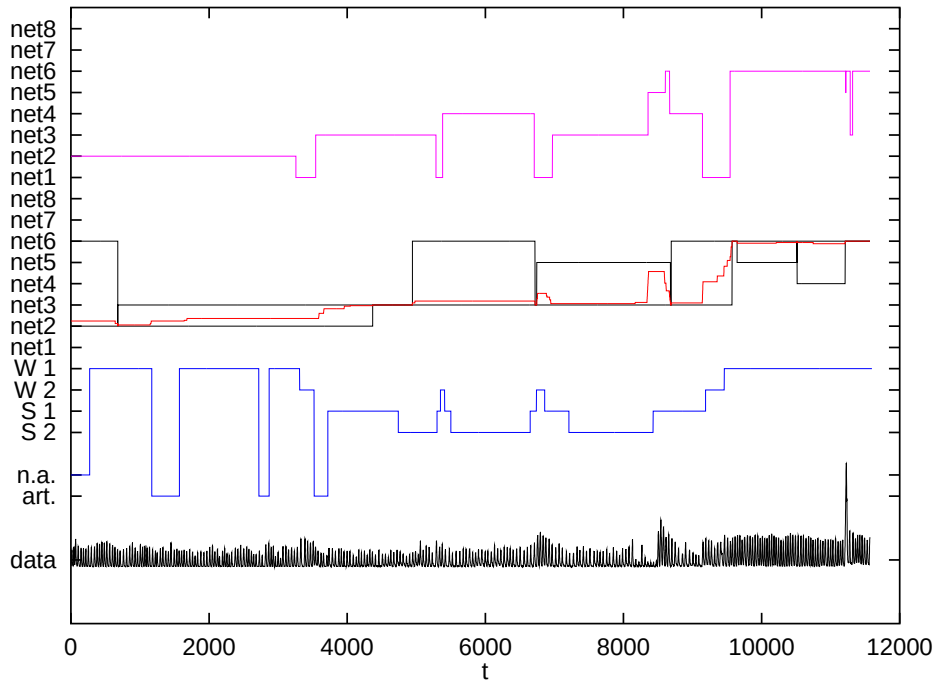


Abbildung 8.2: *Harte (oben), Drift- (Mitte) und manuelle Segmentation (unten) für das Atmungssignal rs11 (Zeit in 0.1s). W1 und W2 zeigen Wachen mit offenen/geschlossenen Augen an, S1 und S2 Schlafstadium I und II (n.a.: nicht ausgewertet, art.: Artefakte). Die gepunktete Linie in der Drift-Segmentation zeigt die Entwicklung des Mischungskoeffizienten $a(t)$ jeweils zwischen den zwei Netzen, die durch die obere und untere Begrenzungslinie angegeben sind, z. B. wird zwischen $t = 500$ und 4500 eine Drift von Netz 2 zu Netz 3 angezeigt.*

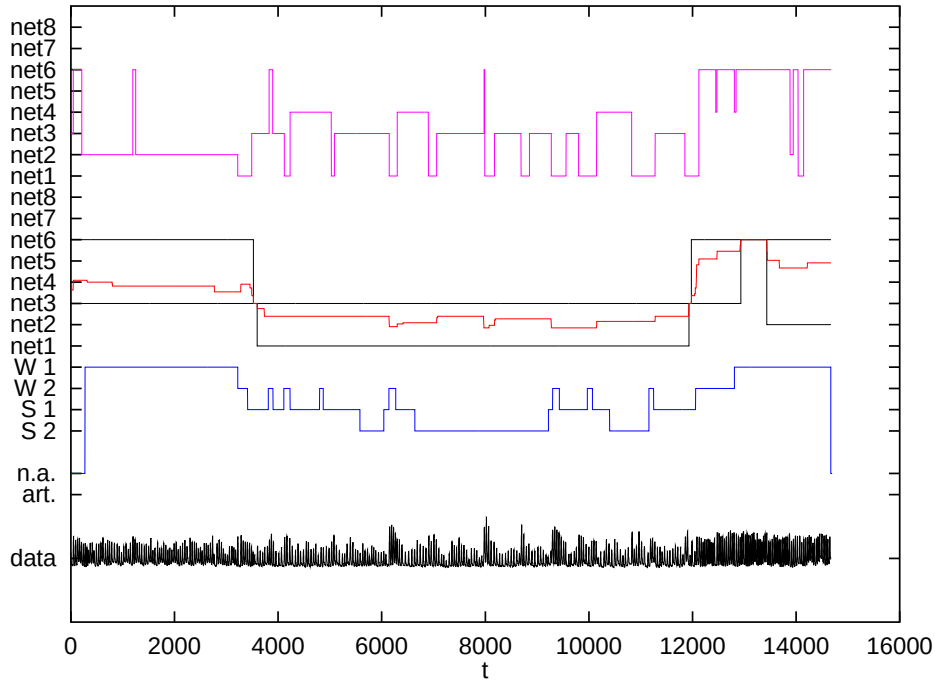


Abbildung 8.3: *Der Segmentationsplot für die Testdaten rs13 (siehe Text).*

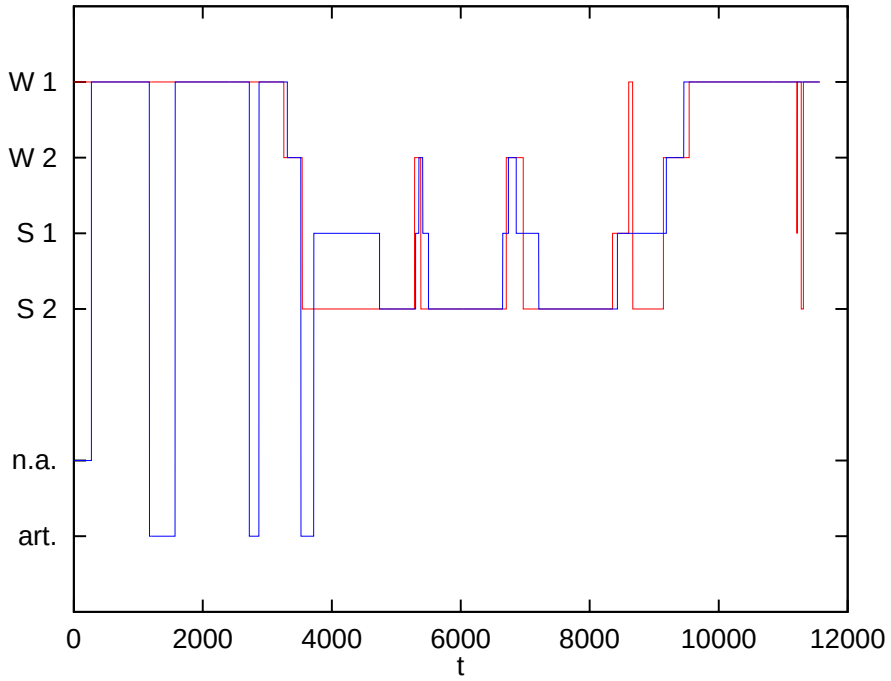


Abbildung 8.4: *Vergleich des Segmentationsresultats für das Atmungssignal rs11 (gepunktete Linie) mit der manuellen Segmentation (durchgezogene Linie). Mit Ausnahme der Unterscheidung von S1 und S2, zeigt sich eine gute Übereinstimmung der Ergebnisse.*

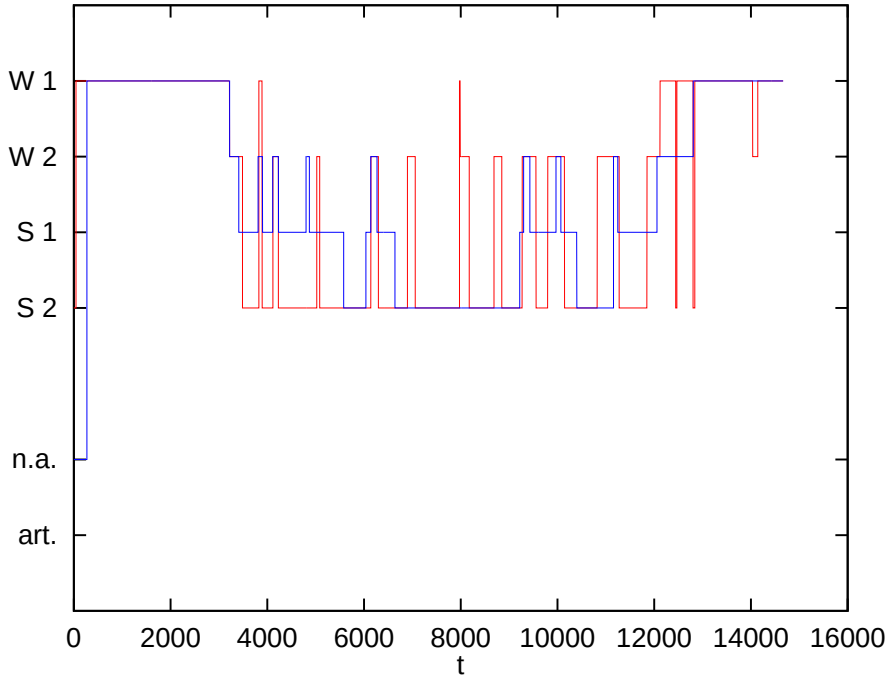


Abbildung 8.5: *Vergleich der Segmentationen für den Generalisierungstest auf Atmungssignal rs13. Die maschinelle Segmentation (gepunktete Linie) zeigt mehr Aufwachphasen an, als die manuelle Segmentation (durchgezogene Linie).*

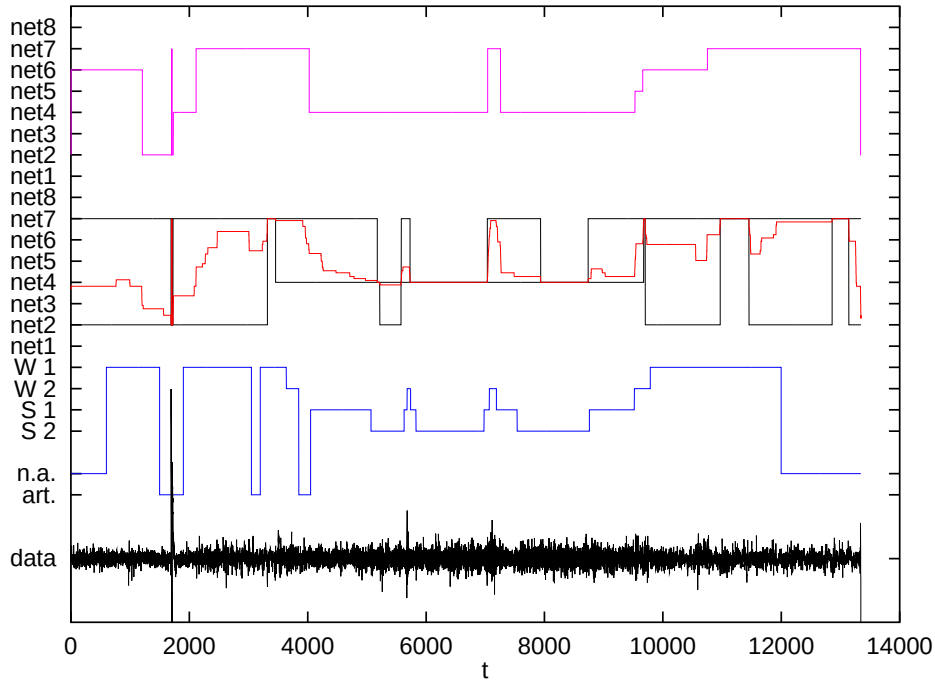


Abbildung 8.6: *Vergleich von harter Segmentation (oben), Drift-Segmentation (Mitte) und manueller Segmentation eines Mediziners (unten) für eeg11. Nur ein EEG-Kanal (o1) wird für unsere Methode verwendet (Zeit in 0.1s).*

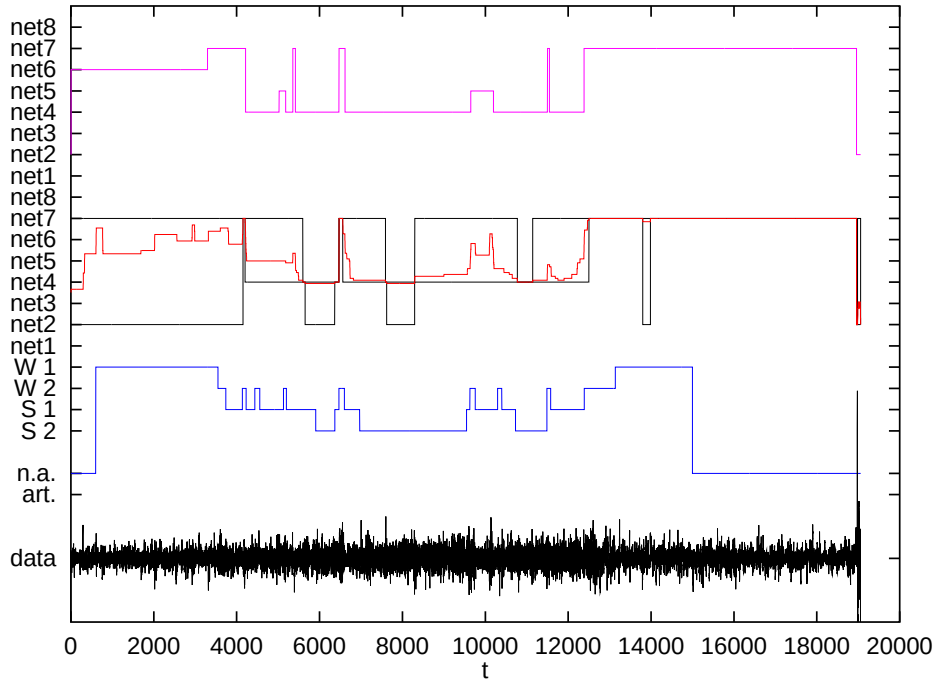


Abbildung 8.7: *Der Segmentationsplot für die EEG-Testdaten eeg13 (o1, 2000 Sekunden). Der Vergleich mit der manuellen Segmentation (unten) zeigt auch hier eine gute Übereinstimmung.*

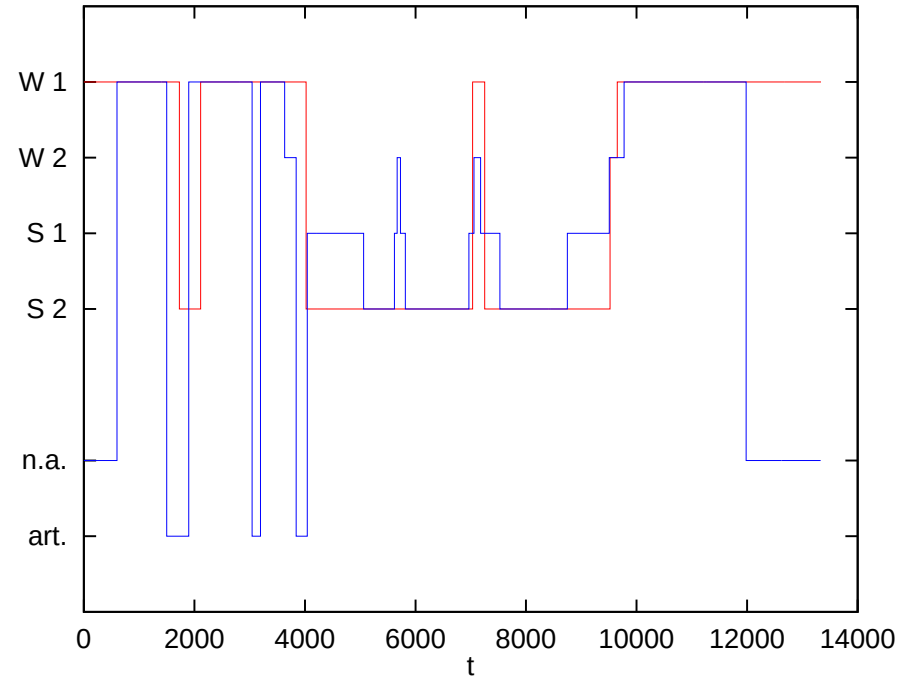


Abbildung 8.8: *Vergleich des Segmentationsresultats für das EEG-Signal eeg11 (gepunktete Linie) mit der manuellen Segmentation (durchgezogene Linie). Wieder zeigt sich, mit Ausnahme der Unterscheidung von S1 und S2, eine gute Übereinstimmung.*

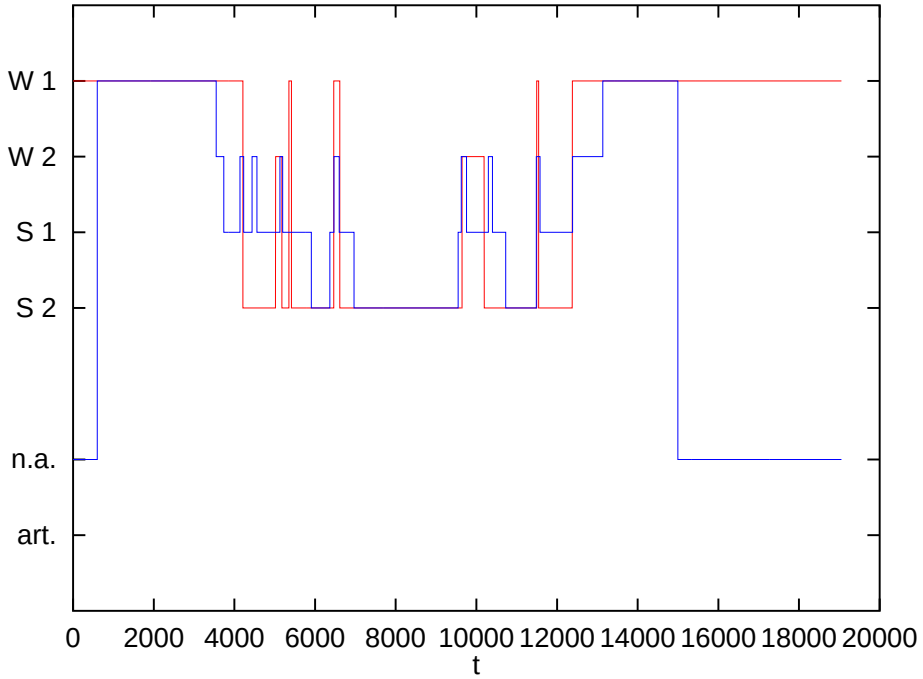
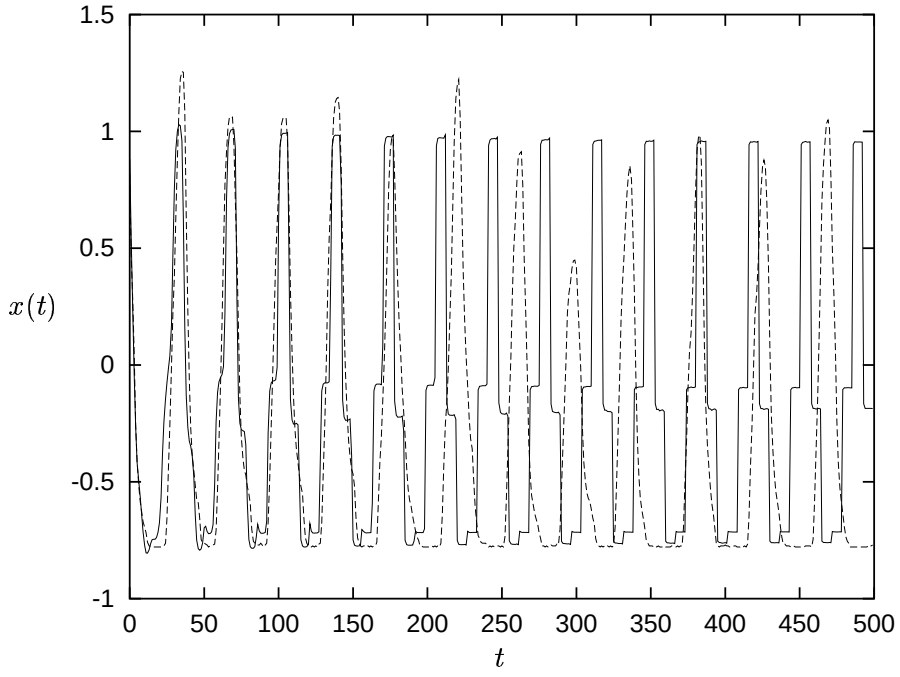
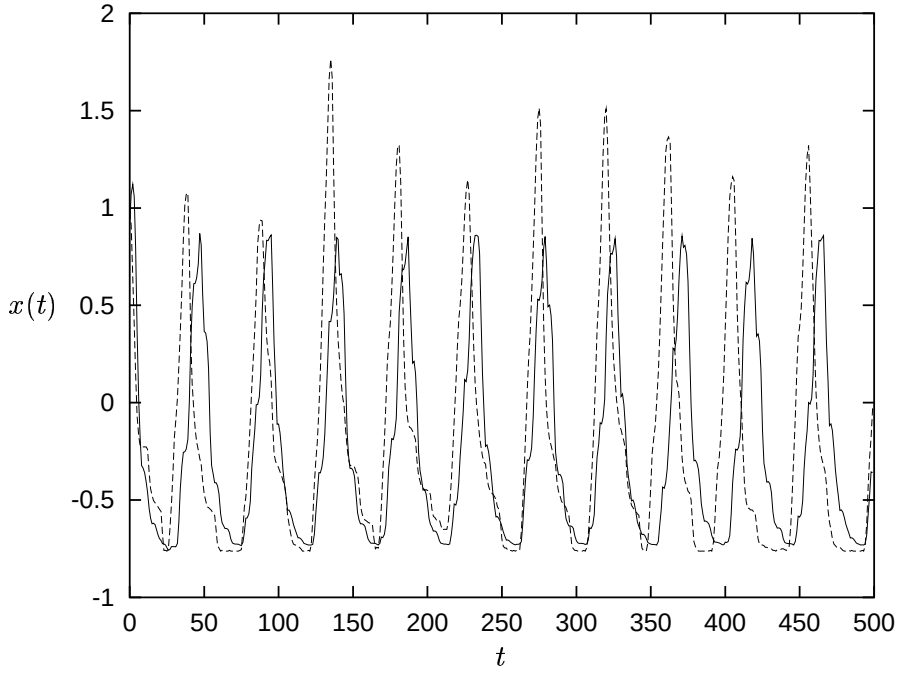


Abbildung 8.9: *Vergleich des gefundenen Segmentationsergebnisses für eeg13 mit der manuellen Segmentation. Es wurden hierbei die Netze verwendet, die zuvor auf eeg11 trainiert wurden.*

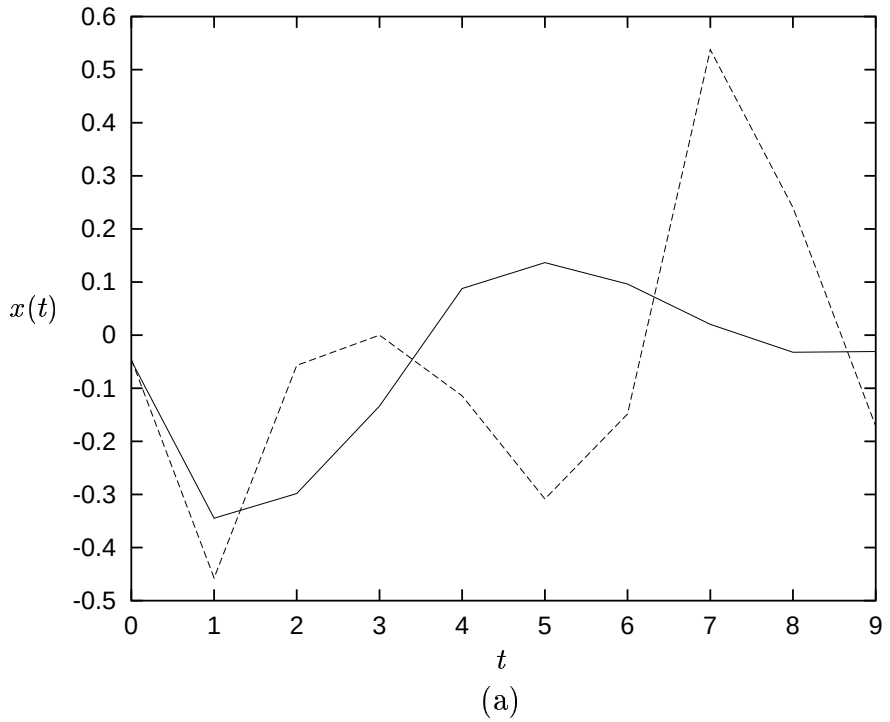


(a)

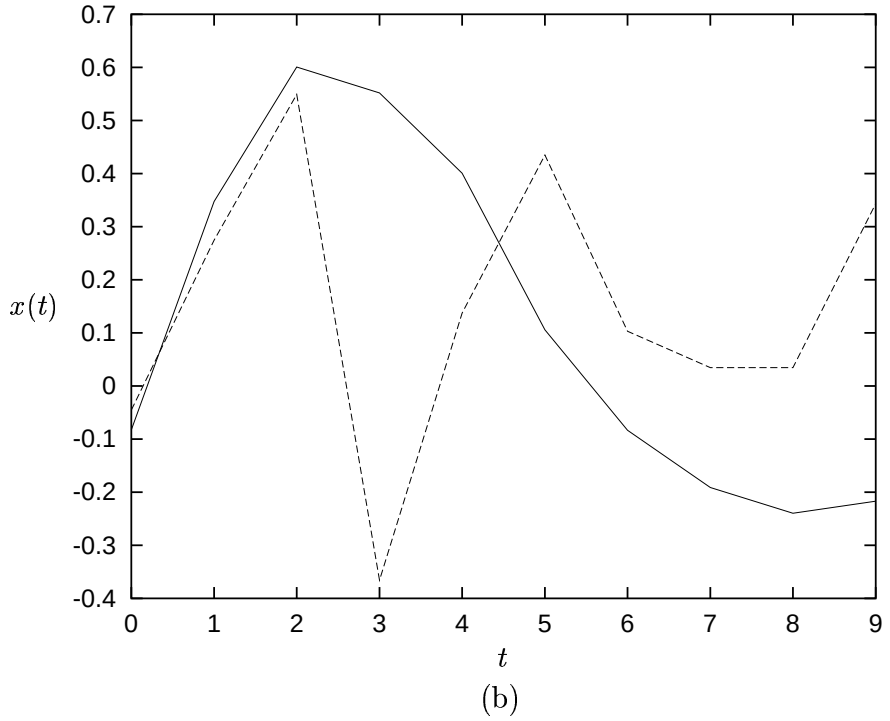


(b)

Abbildung 8.10: *Atmung. Das Iterieren der Prädiktoren für (a) die Wachphase und (b) die Schlafphase für 500 Zeitschritte (durchgezogene Linie) zeigt eine gute Übereinstimmung mit der wahren Fortsetzung des Atmungssignals (gestrichelte Linie) für 180 bzw. 350 Zeitschritte (ein Zeitschritt entspricht 0.1s).*



(a)



(b)

Abbildung 8.11: *EEG. Das Iterieren der Prädiktoren für (a) die Wachphase und (b) die Schlafphase im EEG (durchgezogene Linie) zeigt lediglich eine Übereinstimmung mit der wahren Fortsetzung des Signals (gestrichelte Linie) für die ersten 2 bzw. 3 Zeitschritte (ein Zeitschritt entspricht 0.1s).*

Kapitel 9

Zusammenfassung und Diskussion

In dieser Arbeit wurde ein Verfahren zur unüberwachten Segmentierung und Identifizierung von Zeitreihen vorgestellt. Es eignet sich zur Analyse von Systemen, denen eine nichtstationäre, schaltende oder driftende Dynamik zugrunde liegt; ein Phänomen, das in vielen Bereichen der Natur beobachtet werden kann. Als Beispiele hierfür seien die Sprache oder physiologische Systeme genannt – zwei Bereiche, die in dieser Arbeit exemplarisch untersucht wurden. Aber auch andere komplexe Systeme, wie etwa die Entwicklung der Finanzmärkte, lassen das Vorhandensein einer solchen Dynamik vermuten.

Unser Ansatz basiert auf *hard* oder *soft competitive learning* (ACE) und einem Wettbewerbsvorteil auf zeitlich benachbarten Daten. Im Wettbewerb stehen dabei Prädiktoren (neuronale Netze), die lernen, die Zeitreihe vorherzusagen. Jeder einzelne Prädiktor kann dabei nur eine *stationäre* Dynamik vorhersagen. Die Prädiktoren spezialisieren sich daher während des Lernvorgangs auf solche Abschnitte der Zeitreihe, in denen die Dynamik weitgehend stationär ist. Dies wird erreicht durch die Verwendung tiefpaßgefilterter Vorhersagefehler, die sich aus der Annahme selten schaltender Dynamik herleitet. Es sollte betont werden, daß die Annahme seltenen Schaltens ein entscheidender Punkt ist, um die gewünschte Segmentation zu erhalten. Ohne eine solche Annahme kommen eine Vielzahl von dynamischen Systemen als Erzeuger einer gegebenen Zeitreihe in Frage. Ein extremes Beispiel wäre ein System, bei dem jeder Wert der Zeitreihe von einer anderen, neuen Dynamik erzeugt wird.

Wie sich gezeigt hat, funktioniert das ACE-Verfahren sehr gut und robust, wenn die nichtstationäre Dynamik der Zeitreihe tatsächlich abschnittsweise stationär ist. So ergab sich im Falle der chaotischen Abbildungen und bei der schaltenden Mackey-Glass Gleichung die korrekte Segmentierung der Zeitreihe, sowie eine sehr gute Rekonstruktion bzw. Identifizierung der zugrundeliegenden Quellen. Dies wurde auch bei verrauschten Daten erreicht. Auch die Segmentierung kontinuierlich gesprochener Vokale lieferte sehr gute Ergebnisse. Bei den fließend gesprochenen Sätzen zeigte sich jedoch, daß Probleme auftreten, wenn nicht mehr davon ausgegangen werden kann, daß die Dynamik abschnittsweise stationär ist. Auch für den Fall, daß die stationären Abschnitte sehr kurz sind, ist ein Wettbewerbsvorteil auf zeitlich benachbarten Daten nur in begrenztem Umfang nützlich, da die Wahrscheinlichkeit groß ist, daß die zeitlich benachbarten Daten bereits zu einer anderen Dynamik gehören. Sind im Fall einer sich kontinuierlich verändernden Dynamik überhaupt keine stationären Abschnitte mehr vorhanden, so können die Prädiktoren bestenfalls ein arithmetisches Mittel der sich ständig verändernden Vorhersagefunktion auf den ihnen zugeteilten Abschnitten erlernen, wie es beispielsweise in Abb. 3.3(a) demonstriert wurde.

Eine sich kontinuierlich verändernde Dynamik kann jedoch mit dem in dieser Arbeit ebenfalls vorgeschlagenen Drift-Algorithmus modelliert werden. Hierfür sind aber Prädiktoren erforderlich, die die Basisdynamiken bereits repräsentieren. Zur Beschreibung driftender Dynamik muß also die Information über die Basisdynamiken entweder explizit durch bereits trainierte Prädiktoren oder implizit durch das Vorhandensein stationärer Abschnitte in der Zeitreihe gegeben sein. Im letzteren Fall müssen zunächst Prädiktoren mit dem ACE-Verfahren trainiert werden, bevor der Drift-Algorithmus angewendet werden kann. Der Drift-Algorithmus kann dann sowohl die Basisdynamiken als auch Mischungen von Dynamiken erkennen, je nachdem, welche Repräsentation die Zeitreihe besser beschreibt. Als Mischungen wurden dabei Linearkombinationen von zwei Prädiktoren verwendet. Prinzipiell sind jedoch auch beliebige Kombinationen von beliebig vielen Prädiktoren möglich. Eine solche Verallgemeinerung des Drift-Modells macht dann Sinn, wenn in einer Anwendung entsprechendes Vorwissen über die Art der Drift vorhanden ist.

Zur Segmentierung und Identifizierung schaltender Dynamik mit dem ACE Algorithmus ist es notwendig, daß die zu untersuchende Zeitreihe vollständig vorliegt. Der ACE Algorithmus ist also kein on-line Verfahren, d.h. er kann einen ankommenden Datenstrom nicht dynamisch fortlaufend analysieren, sondern er wird auf komplette Datenblöcke (*batches*) angewendet. Dies gilt auch für den

Drift-Algorithmus. Kehagias und Petridis haben jedoch eine *on-line* Variante unseres *hard competition* Verfahrens vorgeschlagen [24]. Sie berichten, daß sie damit die von uns definierten Benchmarks (schaltende Zeitreihen der vier chaotischen Abbildungen und das schaltende Mackey-Glass System) erfolgreich segmentieren konnten.

Eine weitere Variante unseres Verfahrens haben Fancourt und Principe [12, 13] vorgeschlagen. Sie verwenden eine Kohonen-Karte [34], die Nachbarschaftsbeziehungen zwischen den Prädiktoren definiert. Die Größe des Lernschrittes eines Prädiktors hängt bei diesem Ansatz nicht mehr von dessen eigenem Vorhersagefehler ab, sondern lediglich von seinem topologischen Abstand (auf der Kohonen-Karte) zu demjenigen Prädiktor, der den *kleinsten* Vorhersagefehler hat. Fancourt und Principe sehen den Vorteil darin, daß „ähnliche" Dynamiken auf diese Weise von benachbarten Prädiktoren repräsentiert werden [12]. Darüber hinaus verwenden sie einen asymmetrischen Tiefpaßfilter, der nur Werte aus der Vergangenheit berücksichtigt und dessen Zeitfenster während des Lernverfahrens adaptiert wird. Sie haben mit diesem Verfahren einen schaltenden FIR Prozess [71] untersucht. In der gewonnenen Segmentation wird jedoch deutlich häufiger geschaltet, als in dem tatsächlichen System. Dies wird dadurch erklärt, daß die Größe des Zeitfensters am Ende des Lernverfahrens einen sehr kleinen Wert angenommen hat [13]. Unser Lernverfahren zeigt dagegen, bedingt durch die Wahl eines *festen* Zeitfensters, immer eine Bevorzugung selten schaltender Modelle.

Der ACE Algorithmus macht keine Aussagen darüber, wann oder mit welcher Wahrscheinlichkeit das nächste Mal geschaltet wird. Dadurch ist der Vorhersagehorizont eingeschränkt, wenn es darum geht, die Fortsetzung einer gegebenen Zeitreihe vorherzusagen. Ein Beispiel dafür ist Datensatz D der Santa Fe Competition (siehe Abschnitt 5.3). Eine verläßliche Vorhersage des Signals ist maximal bis zum nächsten Schaltzeitpunkt möglich, der aber nicht vorhergesagt werden kann.

Dafür kommt der ACE Algorithmus aber andererseits mit sehr wenigen Daten zur Analyse des Schaltens aus. Mit wenigen Daten ist es praktisch auch nicht möglich, die Schaltwahrscheinlichkeit vorherzusagen, da in diesem Fall keine zuverlässige Statistik über das Schalten aufgestellt werden kann. Liegen jedoch sehr viele Daten vor, könnte ein Wahrscheinlichkeitsmodell des Schaltens geschätzt werden und so die Vorhersagefähigkeit verbessert werden. Als Weiterentwicklung des ACE Ansatzes wäre es daher denkbar, daß ein Schaltmodell gelernt werden kann. Verwendet man den ACE Algorithmus, so erhält man auf den gegebenen

Daten bereits eine Folge von Schaltzeitpunkten. Diese könnten nachfolgend zum Training eines übergeordneten Wahrscheinlichkeitsmodells des Schaltens eingesetzt werden.

Die Anwendung der entwickelten Methoden auf physiologische Wach/Schlaf-Daten (EEG, EOG, Atmung) hat gezeigt, daß die gefundene Segmentation der Meßdaten eine Einteilung in Wach- und Schlafphasen liefert, die gut mit der manuellen Segmentation eines Mediziners übereinstimmt. Dies ist deshalb besonders beachtlich, da für die maschinelle Analyse keinerlei medizinsches Expertenwissen über die Einteilung in Schlafstadien verwendet wurde. Darüber hinaus liefert unsere maschinelle Methode eine sehr hohe zeitliche Auflösung von unter einer Sekunde. Dies ist um ein bis zwei Größenordnungen besser als die Auflösung bisheriger Standardverfahren, die im Bereich 10–20 Sekunden liegt.

Bisher wurden die Meßkanäle mit unserer Methode einzeln untersucht. Eine Verbesserung der Ergebnisse läßt sich erwarten, wenn es gelänge, mehrere physiologische Kanäle gemeinsam zu untersuchen. Dazu muß jedoch zunächst ein Weg gefunden werden, die unterschiedlichen Zeitskalen der Signale zu integrieren.

Die Prozesse beim Übergang vom Wachen zum Schlafen sind in der Physiologie bisher nur schlecht verstanden. Die entwickelten Verfahren sind in der Lage, den Verlauf der dynamischen Drift beim Übergang vom Wachen zum Schlafen zu modellieren. Dies könnte zu einem besseren Verständnis der zugrundeliegenden physiologischen Prozesse führen, die für den Einschlafvorgang verantwortlich sind. Es wären sogar bessere Diagnosemöglichkeiten bei Schlafstörungen denkbar, wenn es gelänge, unterschiedliche Einschlafprofile für die verschiedenen Arten von Schlafstörungen zu bestimmen. Es wäre auch interessant zu untersuchen, inwieweit sich die Driftkurve als Maß für die Wachsamkeit des Probanden eignet. Hier ist eine potentielle Anwendung unserer Methode als Einschlafwarner, zum Beispiel im Straßenverkehr, zu sehen.

Bezüglich der generellen Anwendbarkeit der in dieser Arbeit vorgestellten Analysemethode läßt sich zusammenfassend also folgendes sagen: Die Methode eignet sich zur Untersuchung von Zeitreihen dynamischer Systeme, die in verschiedenen Moden operieren. Dabei müssen die einzelnen Moden jedoch jeweils eine gewisse Zeitlang stationär vorliegen, bevor das System in einen anderen Modus schaltet oder driftet. Es wird weiter vorausgesetzt, daß es in jedem Operationsmodus, abgesehen von einem Rauschanteil, einen funktionalen Zusammenhang in den Daten gibt, der sich dem Einbettungstheorem [64] entsprechend rekonstruieren läßt. Die

gesamte Zeitreihe muß außerdem zum Zeitpunkt der Analyse komplett vorliegen. Vorhersagen der zukünftigen Entwicklung der Zeitreihe sind insofern beschränkt, als daß eine Änderung des dynamischen Modus nicht vorhergesagt werden kann. Eine solche Änderung wird folglich in der Vorhersage nicht berücksichtigt. Die Frage, welche Art von Prädiktor für das Analyseverfahren am besten geeignet ist, kann nicht allgemeingültig beantwortet werden, da die Antwort grundsätzlich von der jeweils beabsichtigten Anwendung abhängt. Die vorgeschlagenen Segmentierungsmethoden sind jedoch prinzipiell unabhängig von der Wahl der Prädiktoren. Die von uns verwendeten RBF-Prädiktoren haben sich in unseren Anwendungen als robust und zuverlässig erwiesen. Zur Optimierung der Vorhersageleistung können darüber hinaus allgemeine Techniken zur Modellselektion bzw. Regularisierung eingesetzt werden [3, 47, 48].

Das ACE-Verfahren wurde bereits von der Hessischen Landesbank erfolgreich zur Analyse von Finanzdaten eingesetzt. Außerdem wird die Methode im Rahmen des laufenden EU-Projekts STORM (*Stochastic Non-Stationary Modelling Environment for Decision Processes in Financial Markets*) als zentrale Komponente des Datenanalyse-Moduls verwendet. Zudem soll der Ansatz in einem DFG-geförderten Projekt (*Identifikation nichtstationärer und überlagerter Quellen*) weiterentwickelt werden.

Literaturverzeichnis

[1] Barron, A. R. (1993). Universal approximation bounds for superposition of a sigmoidal function. *IEEE Transactions on Information Theory* **39** (3), 930–945.

[2] Bengio, Y., Frasconi, P. (1994). Credit assignment through time: Alternatives to backpropagation. In *NIPS '93: Advances in Neural Information Processing Systems* 6 (eds. J.D. Cowan, G. Tesauro, J. Alspector), Morgan Kaufmann, 75–82.

[3] Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*, Oxford University Press, NY.

[4] Box, G. E. P., Jenkins, G. M., Reinsel, G. C. (1994). *Time series analysis - forecasting and control*, Prentice Hall, NJ.

[5] Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing: Algorithms, Architectures and Applications* (eds. F. Fogelman Soulie, J. Herault), New York: Springer, 227–236.

[6] Cacciatore, T. W., Nowlan, S. J. (1994). Mixtures of Controllers for Jump Linear and Non-linear Plants. In *NIPS '93: Advances in Neural Information Processing Systems* 6 (eds. J.D. Cowan, G. Tesauro, J. Alspector), Morgan Kaufmann, 719–726.

[7] Casdagli, M. (1989). Nonlinear Prediction of Chaotic Time Series. *Physica D* **35**, 335–356.

[8] Davis, H., Davis, P. A., Loomis, A. L., Harvey, E. N., Hobart, G. (1937). Changes in Human Brain Potentials during the Onset of Sleep. *Science* **86**, 448–450.

[9] Deppisch, J. (1990). Vorhersagen chaotischer Zeitreihen mit neuronalen Netzen. Diplomarbeit, Universität Würzburg.

[10] DeSieno, D. (1988). Adding a Conscience to Competitive Learning. In *IEEE International Conference on Neural Networks* (San Diego 1988), New York: IEEE, vol. I, 117–124.

[11] Duda, R. O., Hart, P. E. (1973). *Pattern Classification and Scene Analysis*, Wiley, NY.

[12] Fancourt, C., Principe, J. C. (1996). A Neighborhood Map of Competing One Step Predictors for Piecewise Segmentation and Identification of Time Series. In *ICNN '96: Proc. of the Int. Conf. on Neural Networks*, vol. 4, 1906–1911.

[13] Fancourt, C., Principe, J. C. (1997). Temporal self-organization through competitive prediction. In *ICASSP '97: Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing*, vol. 4, 8–12.

[14] Farmer, J. D., Sidorowich, J. J. (1987). Predicting chaotic time series. *Phys. Rev. Letters* **59** (8), 845–848.

[15] Farmer, J. D., Sidorowich, J. J. (1988). Exploiting Chaos to Predict the Future and Reduce Noise. In *Evolution, Learning, and Cognition*, ed. W.C. Lee, Singapore: World Scientific, 277–330.

[16] Gray, C. M., Engel, A. K., König, P., Singer, W. (1989). Oscillatory Responses in Cat Visual Cortex Exhibit Inter-columnar Synchronization which Reflects Global Stimulus Properties, *Nature* **338**, 334.

[17] Hamilton, J. D. (1994). *Time Series Analysis*, Princeton University Press, Princeton, NJ.

[18] Hartman, E. J., Keeler, J. D., Kowalski, J. M. (1990). Layered neural networks with Gaussian hidden units as universal approximations. *Neural Computation* **2** (2), 210–215.

[19] Hertz, J. A., Krogh, A., Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, Redwood City.

[20] Honerkamp, J. (1990). *Stochastische Dynamische Systeme*, VCH-Verlag, Weinheim.

[21] Hornik, K., Stinchcombe, M., White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks* **2** (5), 359–366.

[22] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., Hinton, G. E. (1991). Adaptive Mixtures of Local Experts. *Neural Computation* **3**, 79–87.

[23] Kaneko, K. (1989). Chaotic but Regular Posi-Nega Switch among Coded Attractors by Cluster-Size Variation. *Phys. Rev. Letters* **63**, 219.

[24] Kehagias, A., Petridis, V. (1997). Time Series Segmentation using Predictive Modular Neural Networks. *Neural Computation* **9**, 1691–1710.

[25] Kleitman, N. (1963). *Sleep and Wakefulness*, The University of Chicago Press, Chicago.

[26] Kohlmorgen, J. (1992). Lernen stetiger Funktionen in hybriden neuronalen Netzen. Studienarbeit am Institut für Angewandte Informatik, Technische Universität Berlin.

[27] Kohlmorgen, J. (1994). Segmentierung und Identifizierung nichtstationärer Zeitreihen — Eine Anwendung mit neuronalen Netzen. Diplomarbeit am Institut für Angewandte Informatik, Technische Universität Berlin.

[28] Kohlmorgen, J., Müller, K.-R., Pawelzik, K. (1994). Competing Predictors Segment and Identify Switching Dynamics. In *ICANN '94: Proc. of the Int. Conf. on Artificial Neural Networks*, Springer London, 1045–1048.

[29] Kohlmorgen, J., Müller, K.-R., Pawelzik, K. (1995). Improving short-term prediction with competing experts. In *ICANN '95: Proc. of the Int. Conf. on Artificial Neural Networks*, EC2 & Cie, Paris, 2:215–220.

[30] Kohlmorgen, J., Müller, K.-R., Pawelzik, K. (1996). Analysis of Drifting Dynamics with Competing Predictors. In *ICANN '96: Proc. of the Int. Conf. on Artificial Neural Networks* (eds. C. von der Malsburg, W. von Seelen, J.C. Vorbrüggen, B. Sendhoff), LNCS 1112, Springer Berlin, 785–790.

[31] Kohlmorgen, J., Müller, K.-R., Pawelzik, K. (1997). Segmentation and Identification of Drifting Dynamical Systems. In *NNSP '97: IEEE Workshop on Neural Networks for Signal Processing* (eds. J. Principe, L. Giles, N. Morgan, E. Wilson), IEEE, 326–335.

[32] Kohlmorgen, J., Müller, K.-R., Rittweger, J., Pawelzik, K. (1997). Analysis of Wake/Sleep EEG with Competing Experts. In *ICANN '97: Proc. of*

the Int. Conf. on Artificial Neural Networks (eds. W.Gerstner, A.Germond, M.Hasler, J.-D. Nicoud), LNCS 1327, Springer Berlin, 1077–1082.

[33] Kohlmorgen, J., Müller, K.-R., Pawelzik, K. (1998). Analysis of Drifting Dynamics with Neural Network Hidden Markov Models. In *NIPS '97: Advances in Neural Information Processing Systems* 10, MIT Press, to appear.

[34] Kohonen, T. (1984). *Self-Organization and Associative Memory*. Springer Series in Information Sciences, Springer-Verlag, Berlin.

[35] Langhorst, P., Schulz, B., Schulz, G., Lambertz, M. (1983). Reticular formation of the lower brainstem. A common system for cardiorespiratory and somatomotor functions: discharge patterns of neighbouring neurons influenced by cardiovascular and respiratory afferents. *J.A.N.S.* **9**, 411–432.

[36] Lapedes, A., Farber, R. (1987). Nonlinear Signal Processing Using Neural Networks: Prediction and System Modelling. Technical Report LA-UR-87-2662, Los Alamos National Laboratory, Los Alamos, New Mexico.

[37] Liebert, W., Pawelzik, K., Schuster, H. G. (1991). Optimal Embeddings of Chaotic Attractors from Topological Considerations. *Europhysics Letters* **14**, 521.

[38] Mackey, M., Glass, L. (1977). Oscillation and Chaos in a Physiological Control System. *Science* **197**, 287.

[39] Magnussen, G. (1944). *Studies on the respiration during sleep. A contribution to the physiology of sleep function*. Lewis & Co Ltd., London.

[40] McLachlan, G. J., Basford, K. J. (1988). *Mixture models*, Marcel Dekker, NY and Basel.

[41] Moody, J., Darken, C. (1988). Learning with Localized Receptive Fields. In *Proceedings of the 1988 Connectionist Models Summer School* (Pittsburg 1988), eds. D. Touretzky, G. Hinton, and T. Sejnowski. San Mateo: Morgan Kaufmann, 133–143.

[42] Moody, J., Darken, C. (1989). Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Computation* **1**, 281–294.

[43] Müller, K.-R., Kohlmorgen, J., Pawelzik, K. (1994). Segmentation and Identification of Switching Dynamics with Competing Neural Networks. In *ICONIP '94: Proc. of the Int. Conf. on Neural Information Processing*, Seoul, 213–218.

[44] Müller, K.-R., Kohlmorgen, J., Pawelzik, K. (1994). The Use of Competing Neural Networks for Segmentation and Identification of Switching Dynamics. In *NOLTA '94: Kagoshima Symposium on Nonlinear Theory and its Applications*, 283–286.

[45] Müller, K.-R., Kohlmorgen, J., Pawelzik, K. (1995). Analysis of Switching Dynamics with Competing Neural Networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E78-A, No.10, 1306–1315.

[46] Müller, K.-R., Kohlmorgen, J., Rittweger, J., Pawelzik, K. (1995). Analysing Physiological Data from the Wake-Sleep State Transition with Competing Predictors. In *NOLTA '95: Las Vegas Symposium on Nonlinear Theory and its Applications*, 223–226.

[47] Murata, N., Yoshizawa, S., Amari, S. (1993). Learning curves, model selection and complexity of neural networks. In *NIPS '92: Advances in Neural Information Processing Systems* 5 (eds. S.J. Hanson, J.D. Cowan, C.L. Giles), Morgan Kaufmann, 607.

[48] Murata, N., Yoshizawa, S., Amari, S. (1994). Network information criterion — Determining the number of hidden units for an artificial neural network model. *IEEE Transactions on Neural Networks* **5**, 865–872.

[49] Nowlan, S. J. (1990). Maximum Likelihood Competitive Learning. In *NIPS '89: Advances in Neural Information Processing Systems* 2 (ed. D. Touretzky), Morgan Kaufmann, 574–582.

[50] Packard, N. H., Crutchfield J. P., Farmer, J. D., Shaw, R. S. (1980). Geometry from a Time Series. *Phys. Rev. Letters* **45**, 712–716.

[51] Pawelzik, K., Bauer, H.-U., Deppisch, J., Geisel, T. (1993). How Oscillatory Neuronal Responses Reflect Bistability and Switching of the Hidden Assembly Dynamics. In *NIPS '92: Advances in Neural Information Processing Systems* 5 (eds. S.J. Hanson, J.D. Cowan, C.L. Giles), Morgan Kaufmann, 977–984.

[52] Pawelzik, K. (1994). Detecting Coherence in Neuronal Data. In: Domany, E., Van Hemmen, J. L., Schulten, K. (Eds.), *Models of Neural Networks II*, Springer, 253–285.

[53] Pawelzik, K., Kohlmorgen, J., Müller, K.-R. (1996). Annealed Competition of Experts for a Segmentation and Classification of Switching Dynamics. *Neural Computation* **8** (2), 340–356.

[54] Pawelzik, K., Müller, K.-R., Kohlmorgen, J. (1996). Prediction of Mixtures. In *ICANN '96: Proc. of the Int. Conf. on Artificial Neural Networks* (eds. C. von der Malsburg, W. von Seelen, J.C. Vorbrüggen, B. Sendhoff), LNCS 1112, Springer Berlin, 127–132.

[55] Pawelzik, K., Müller, K.-R., Kohlmorgen, J. (1997). Divisive Strategies for Predicting Non-Autonomous and Mixed Systems, Tech. Report 1069, GMD.

[56] Priestley, M. B. (1988). *Non-linear and Non-Stationary Time-Series Analysis*, Academic Press, London.

[57] Rabiner, L. R. (1990). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Readings in Speech Recognition*, eds. A. Waibel, K. Lee. San Mateo: Morgan Kaufmann, 267–296.

[58] Rittweger, J., Lambertz, M., Langhorst, P. (1995). Interaction of brainstem oscillations is modified during state transition between wakefulness and sleep. In *First French-German Conference on Sleep Research and Sleep Medicine*, Trier.

[59] Rose, K., Gurewitz, E., Fox, G. (1990). Statistical Mechanics and Phase Transitions in Clustering. *Phys. Rev. Letters* **65**, 945–948.

[60] Rumelhart, D. E., McClelland, J. L., and the PDP Research Group (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Cambridge: MIT Press.

[61] Santamaria, J., Chiappa, K. H. (1987). The EEG of drowsiness in normal adults. *Journal of Clinical Neurophysiology* **4**, 327–382.

[62] Schuster, H. G. (1988). *Deterministic Chaos*, 2nd Edition, Physik Verlag, Weinheim.

[63] Shamma, J. S., Athans, M. (1992). Gain scheduling: potential hazards and possible remedies. *IEEE Control Systems Magazine* **12** (3), 101–107.

[64] Takens, F. (1981). Detecting Strange Attractors in Turbulence. In: Rand, D., Young, L.-S. (Eds.), *Dynamical Systems and Turbulence*, Springer Lecture Notes in Mathematics, **898**, 366–381.

[65] Tong, H., Lim, K. S. (1980). Threshold autoregression, limit cycles and cyclical data. *J. R. Stat. Soc. B* **42**, 245–268.

[66] Trinder, J., Whitworth, F., Kay, A., Wilkin, P. (1992). Respiratory instability during sleep onset. *J. Appl. Physiol.* **73**, 2462–2469.

[67] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K. and Lang, K. (1989). Phoneme Recognition Using Time-Delay Neural Networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing* **37**, 328–339.

[68] Waibel, A., Lee. K. (Eds.) (1990). *Readings in Speech Recognition*, San Mateo: Morgan Kaufmann.

[69] Weigend, A. S., Gershenfeld, N. A. (Eds.) (1994). *Time Series Prediction: Forecasting the Future and Understanding the Past*, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley.

[70] Weigend, A. S., Mangeas, M. (1995). Nonlinear gated experts for time series: discovering regimes and avoiding overfitting. *International Journal of Neural Systems* **6**, 373–399.

[71] Widrow, B., Stearns, S. D. (1985). *Adaptive Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall.

[72] White, H. (1992). *Artificial neural networks: approximation and learning theory*, Blackwell Publishers.

[73] Yuille, A. L., Stolorz, P., Utans, J. (1994). Statistical Physics, Mixtures of Distributions, and the EM Algorithm. *Neural Computation* **6**, 334–340.