SCHRIFTENREIHE DES INSTITUTS FÜR
ANGEWANDTE INFORMATIK | AUTOMATISIERUNGSTECHNIK
KARLSRUHER INSTITUT FÜR TECHNOLOGIE (KIT)

Band 50

Herausgeber

F. HOFFMANN
E. HÜLLERMEIER

Dortmund | 27. – 28. November 2014

PROCEEDINGS **24. WORKSHOP** COMPUTATIONAL INTELLIGENCE

KIT Scientific Publishing

F. Hoffmann, E. Hüllermeier (Hrsg.)

Proceedings 24. Workshop Computational Intelligence

Dortmund, 27. – 28. November 2014

Schriftenreihe des
Instituts für Angewandte Informatik / Automatisierungstechnik
am Karlsruher Institut für Technologie
Band 50

Eine Übersicht aller bisher in dieser Schriftenreihe erschienenen Bände finden Sie am Ende des Buches.

PROCEEDINGS **24. WORKSHOP** COMPUTATIONAL INTELLIGENCE

Dortmund, 27. – 28. November 2014

F. Hoffmann
E. Hüllermeier
(Hrsg.)

KIT Scientific Publishing

Impressum

Scientific Publishing

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

KIT Scientific Publishing is a registered trademark of Karlsruhe Institute of Technology. Reprint using the book cover is not allowed.

www.ksp.kit.edu

Print on Demand 2014

ISSN 1614-5267
ISBN 978-3-7315-0275-3
DOI: 10.5445/KSP/1000043427

Inhaltsverzeichnis

Abstands- und Monotoniemaße für Regressionsmodelle mit heterogenen Lerndaten

Wolfgang Doneit[1], **Ralf Mikut**[1], **Tim Pychynski**[2], **Markus Reischl**[1]

[1]Karlsruher Institut für Technologie, Institut für Angewandte Informatik
E-Mail: {wolfgang.doneit}{ralf.mikut}{markus.reischl}@kit.edu
[2]Karlsruher Institut für Technologie, Institut für Thermische Strömungsmaschinen
E-Mail: {tim.pychynski}@kit.edu

1 Einführung

Bei der Struktur- und Parameteroptimierung technischer Systeme durch Evolutionäre Algorithmen oder andere numerische Verfahren sind meist viele Iterationen zur Güteberechnung notwendig. Wenn diese Güteberechnung einen hohen Aufwand bedeutet, z.B. bei notwendigen Experimenten oder Simulationen mit Finite-Elemente-Modellen bzw. numerischen Strömungssimulationen mit CFD (Computational Fluid Dynamics), werden zunehmend recheneffizientere Regressionsmodelle (z.B. Künstliche Neuronale Netze) zur Fitnessapproximation eingesetzt [1]. Die Bildung dieser Modelle erfordert eine zuverlässige und ausreichend große Datenbasis, um den zulässigen Parameterraum vollständig und gleichförmig abzudecken. Diese Anforderungen werden in der Realität wegen der aufwändigen Datenerhebung, der Fusion heterogener oder widersprüchlicher Datenquellen sowie der Überrepräsentation etablierter Bereiche häufig verletzt. Das führt unter anderem dazu, dass Ersatzmodelle in schlecht abgedeckten Parameterbereichen große Fehler aufweisen oder lokal überangepasst sind (engl. Overfitting). Folglich schlagen sie oft irreführende Werte für die nächste Optimierungsiteration vor. Beispielsweise wird in [2] ein Datensatz aus dem Bereich des Turbomaschinenbaus vorgestellt, mit Hilfe dessen die Beziehung zwischen verschiedenen Parametern und dem Durchflussverhalten von Labyrinthdichtungen modelliert wird. Der Datensatz fasst Messungen aus mehreren Quellen zusammen, die Eingangsgrößen aus unterschiedlichen Bereichen erfassen.

Um die Zuverlässigkeit von Regressionsmodellen zu bewerten, werden beispielsweise Kenngrößen genutzt, die robust gegen Ausreißer sind [3]. Eine sorgfältige Auswahl der Merkmale ist für Regressionen [4] und Klassifikationen [5] wichtig, um die Modellkomplexität gering zu halten. In [6] wird

die Vertrauenswürdigkeit der Prognosen von Neuronalen Netzen anhand der Datendichte betrachteter Merkmalsräume bewertet.

Zur Bewertung der Qualität von Regressionsmodellen werden Informationstheoretische Maße („shortest data description“ [7], „minimum message length“ [8]), Verfahren zur Abschätzung von Approximationen [9, 10] und Kreuzvalidierungsverfahren eingesetzt [4, 11, 12]. Allerdings neigt beispielsweise die Kreuzvalidierung bei inhomogenen Datensätzen dazu, die Modellfehler zu unterschätzen.

Dieser Beitrag betrachtet Auswirkungen von Overfitting, die von Kreuzvalidierungsverfahren nicht erfasst werden können und durch große Schwankungen der geschätzten Ausgangsgröße die Anwendung eines Regressionsmodells negativ beeinflussen.

Abschnitt 2 erklärt, was Overfitting ist und wie Kreuzvalidierungen es erkennen. Abschnitt 3 stellt eine Methode vor, die Overfitting anhand des Verlaufs der geschätzten Ausgangsgröße eines Regressionsmodells erkennen und Kreuzvalidierungen ergänzen kann. Die Methode wird in Abschnitt 4 angewandt und die Ergebnisse diskutiert. Dabei wird sowohl ein in [13] empfohlener Datensatz aus dem UCI Machine Learning Repository, als auch ein eigener Datensatz aus dem Turbomaschinenbau verwendet.

2 Regressionsanalyse

2.1 Übersicht

Eine Regressionsanalyse stellt einen funktionellen Zusammenhang zwischen reellwertigen Eingangsvariablen $\mathbf{x}$ und einer Ausgangsvariable y auf. Eine Regressionsanalyse nutzt Polynome, Künstliche Neuronale Netze (KNN) o.ä., die entsprechend eines Datensatzes (Lerndaten) angepasst werden und ein Regressionsmodell darstellen. Ein Datensatz D für Regressionsanalysen besteht aus N Datentupeln. Jedes Datentupel beinhaltet $s+1$ Einzelmerkmale, die aus s Eingangsvariablen und einer Ausgangsvariable bestehen. Jeder Eingangsvariablenvektor $\mathbf{x}_i \in \mathbb{R}^s, i = 1...N$ stellt einen Punkt im Merkmalsraum dar. Der Datensatz ordnet jedem Punkt eine Ausgangsvariable y_i zu.

Ein Regressionsmodell liefert für einen beliebigen Eingangsvariablenvektor $\mathbf{x}$ einen Schätzwert $\hat{y}$. Bei der Anpassung werden die Struktur und Parameter des Modells gemäß eines Gütekriteriums gewählt, beispielsweise zur Minimierung der Wurzel des mittleren, quadrierten Fehlers

$$\text{RMSE} = \sqrt{\frac{1}{N} \cdot \sum_{\text{i=1}}^{N} (\hat{y}_\text{i} - y_\text{i})^2}. \tag{1}$$

Die Fehlermaße beziehen sich auf die Abweichungen der wahren Ausgangsgrößen von den Schätzungen des Regressionsmodells an allen N Punkten, die in den Lerndaten enthalten sind (engl. „in-sample error“).

2.2 Overfitting

Mit einer genügend komplexen Struktur können viele Regressionsmodelle beliebige Zusammenhänge abbilden, z.B. KNN mit einer hinreichend großen Zahl an Neuronen in einer verdeckten Schicht. Der in-sample error (IE) kann dadurch sehr gering werden. Regressionsmodelle werden häufig dazu verwendet, eine Vorhersage (Prädiktion) für die Ausgangsgröße an einem Punkt zu liefern, der nicht im Datensatz erfasst ist. Der IE sagt nichts über die Güte von solchen Prädiktionen aus. Um ein Regressionsmodell hinsichtlich seiner Prädiktionsfähigkeit zu bewerten, müssen die oben genannten Fehlermaße auf Daten angewandt werden, die nicht zur Modellbildung verwendet worden sind (Testdaten). Die Abweichungen der wahren Ausgangsgrößen von den Schätzungen des Regressionsmodells an den Punkten der Testdaten werden „out-of-sample error“ (OE) genannt. Dafür kann der zugrundeliegende Datensatz in zwei Teile geteilt werden. Ein Teildatensatz wird zur Modellbildung verwendet, der andere zur Modellvalidierung. Vor allem in technischen und naturwissenschaftlichen Anwendungen ist die Datenerhebung oft zeit- und kostenintensiv und die Datensätze enthalten entsprechend nur wenige oder im Merkmalsraum heterogen verteilte Daten. Für eine zuverlässige Modellbildung müssen daher alle Daten miteinbezogen werden.

Bei Datensätzen mit wenigen Datentupeln kann eine Kreuzvalidierung verwendet werden. Bei einer k-fachen Kreuzvalidierung wird der Datensatz in k gleichgroße Teile aufgeteilt. Anschließend werden k-1 Teile zur Modellbildung und 1 Teil zur Modellvalidierung verwendet. Nach k Iterationen diente jeder Teildatensatz genau einmal zur Modellvalidierung. Der Mittelwert der OEs liefert eine Aussage über die Prädiktionsfähigkeit der verwendeten Modellstruktur. Ein gutes Modell hat einen möglichst geringen IE und einen Fehlerquotienten

$$Q_{\text{CV,quot}} = \frac{OE}{IE}, \tag{2}$$

der möglichst nahe bei 1 liegt [4]. Ein hoher Fehlerquotient beurteilt ein Modell als zu komplex und erkennt Overfitting.

Dieser Beitrag untersucht, wann eine Kreuzvalidierung ein Modell zu gut bewertet und ein vorhandenes Overfitting nicht erkennt. Es wird eine Methode vorgestellt, die Overfitting beliebiger Modelle erkennen kann, indem sie den Verlauf der vom Modell geschätzten Ausgangsgröße zwischen zwei benachbarten Punkten des Lerndatensatzes untersucht.

3 Methoden

3.1 Interpolationsvalidierung

Die Interpolationsvalidierung beruht auf der Annahme, dass die geschätzte Ausgangsgröße guter Modelle in Bereichen zwischen zwei Punkten der Lerndaten $\mathbf{x}_a$ und $\mathbf{x}_b$ kein unerwartetes Verhalten zeigt, sondern näherungsweise linear, bzw. monoton interpoliert.

Dazu werden ν vom Modell geschätzte Ausgangsgrößen $\hat{y}_{\mathrm{a,j}}, j = 1, ..., \nu$ an Stellen betrachtet, die gleichmäßig verteilt auf der Gerade zwischen $\mathbf{x}_\mathrm{a}$ und $\mathbf{x}_\mathrm{b}$ im Merkmalsraum liegen. $\hat{y}_{\mathrm{x}_a} = \hat{y}_{\mathrm{a,1}}$ ist die vom Modell geschätzte Ausgangsgröße am Punkt $\mathbf{x}_\mathrm{a}$. $\hat{y}_{\mathrm{x}_b} = \hat{y}_{a,\nu}$ ist die vom Modell geschätzte Ausgangsgröße am Punkt $\mathbf{x}_\mathrm{b}$. Außerdem gilt

$$\hat{y}_{\mathrm{max}} = \max_{j=1}^{\nu}(\hat{y}_{\mathrm{a,j}}), \tag{3}$$

$$\hat{y}_{\mathrm{min}} = \min_{j=1}^{\nu}(\hat{y}_{\mathrm{a,j}}). \tag{4}$$

Die maximale und minimale Ausgangsgröße der gesamten Lerndaten sind y_{max} und y_{min}. Für ein Regressionsproblem gilt:

$$y_{\mathrm{max}} \neq y_{\mathrm{min}}. \tag{5}$$

In Anlehnung an [14] werden sogenannte Interpolationsindikatoren $Q_{\mathrm{IV,1}}$, $Q_{\mathrm{IV,2}}$ und $Q_{\mathrm{IV,3}}$ berechnet, die den Verlauf der geschätzten Ausgangsgröße auf Besonderheiten untersuchen. Gesucht ist ein Gütekriterium $Q_{\mathrm{IV,total}}$, was eine Aussage über ein lokales Overfitting liefert. Der Wertebereich von $Q_{\mathrm{IV,total}}$ sowie der Interpolationsindikatoren liegt zwischen 0 und 1. Je kleiner der Wert von

$$Q_{\mathrm{IV,total}} = \max\left(\frac{1}{3}\cdot(Q_{\mathrm{IV,1}} + Q_{\mathrm{IV,2}} + Q_{\mathrm{IV,3}}), Q_{\mathrm{IV,min}}\right), \tag{6}$$

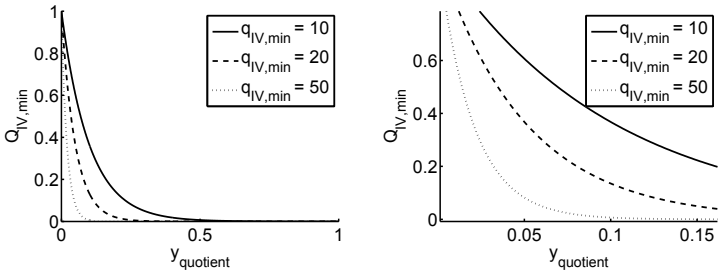
desto eher zeigt das untersuchte Regressionsmodell an der betrachteten Stelle im Merkmalsraum Overfitting. $Q_{\text{IV,min}}$ verhindert, dass Interpolationen mit geringen Schwankungen zu schlecht bewertet werden. Das heißt bei einem größeren Quotienten

$$y_{\text{quotient}} = \frac{\hat{y}_{\max} - \hat{y}_{\min}}{y_{\max} - y_{\min}} \tag{7}$$

werden schlechtere Bewertungen durch die Interpolationsindikatoren zugelassen:

$$Q_{\text{IV,min}} = \exp(-q_{\text{IV,min}} \cdot y_{\text{quotient}}). \tag{8}$$

Um ein geeignetes $q_{\text{IV,min}}$ zu finden, wird die Kurvenschar aus Bild 1 betrachtet.



(a) (b)

Bild 1: (a) zeigt den Verlauf von $Q_{\text{IV,min}}$ bei unterschiedlichen Werten für $q_{\text{IV,min}}$. (b) zeigt einen vergrößerten Teil von (a).

Aufgrund des Kurvenverlaufs wird $q_{\text{IV,min}} = 20$ gewählt. Damit ergibt sich ein angemessener Kompromiss zwischen einer hohen Fehlertoleranz bei geringen Schwankungen der geschätzten Ausgangsgröße und einer geringen Fehlertoleranz bei großen Schwankungen der geschätzten Ausgangsgröße.

Die einzelnen Interpolationsindikatoren berechnen sich folgendermaßen:

$$Q_{\text{IV,1}} = \begin{cases} 1, & \text{falls } \hat{y}_{\max} = \hat{y}_{\min} \\ \frac{|\hat{y}_{\mathbf{x}_b} - \hat{y}_{\mathbf{x}_a}|}{|\hat{y}_{\max} - \hat{y}_{\min}|} & \text{sonst.} \end{cases} \tag{9}$$

$Q_{\text{IV,1}}$ untersucht ob im Verlauf der geschätzten Ausgangsgröße zwischen $\mathbf{x}_\text{a}$ und $\mathbf{x}_\text{b}$ lokale Minima oder Maxima existieren. Dabei gilt

$$|\hat{y}_{\max} - \hat{y}_{\min}| \geq |\hat{y}_{\mathbf{x}_b} - \hat{y}_{\mathbf{x}_a}|. \tag{10}$$

Ein weiteres Kriterium bewertet die Steigung der Regression zwischen $\mathbf{x}_a$ und $\mathbf{x}_b$:

$$Q_{\mathrm{IV,2}} = \begin{cases} 1, & \text{falls } \max_{j=1}^{\nu-1}(|\hat{y}_{\mathrm{a,j+1}} - \hat{y}_{\mathrm{a,j}}|) = 0 \\ \left(\frac{|\hat{y}_{\mathbf{x}_b} - \hat{y}_{\mathbf{x}_a}|}{\nu \cdot \max_{j=1}^{\nu-1}(|\hat{y}_{\mathrm{a,j+1}} - \hat{y}_{\mathrm{a,j}}|)} \right)^{\frac{1}{q_{\mathrm{IV,2}}}}, & \text{sonst.} \end{cases} \tag{11}$$

Mit $q_{IV,2} > 1$ wird reguliert, wie empfindlich $Q_{\mathrm{IV,2}}$ große Steigungen bestraft. Je größer $q_{\mathrm{IV,2}}$ gewählt wird, desto toleranter ist $Q_{\mathrm{IV,2}}$.

Die Abweichungen der vom Regressionsmodell geschätzten Ausgangsgrößen und einer linearen Interpolation von $\hat{y}$ zwischen $\mathbf{x}_a$ und $\mathbf{x}_b$ werden von folgendem Kriterium bewertet:

$$Q_{\mathrm{IV,3}} = \max \left(0, 1 - \frac{\max_{j=1}^{\nu}(|\hat{y}_{\mathrm{lin,j}} - \hat{y}_{\mathrm{j}}|)}{|\hat{y}_{\mathbf{x}_b} - \hat{y}_{\mathbf{x}_a}|} \right). \tag{12}$$

Die lineare Interpolation entspricht

$$\hat{y}_{\mathrm{lin,j}} = \hat{y}_{\mathbf{x}_a} + \frac{\hat{y}_{\mathbf{x}_b} - \hat{y}_{\mathbf{x}_a}}{\nu - 1} \cdot (j-1), j = 1...\nu. \tag{13}$$

Die Interpolationsvalidierung ist kein Verfahren, um beispielsweise Überanpassung an Messrauschen o.ä. zu erkennen. Solches Overfitting wird bereits von Kreuzvalidierungsverfahren vermieden. Das Ziel der Interpolationsvalidierung ist die Vermeidung modellabhängiger Fehlschlüsse in der Parameteroptimierung technischer Systeme. Von Interesse sind also nur Schwankungen der geschätzten Ausgangsgröße, die solche Optimierungsprozesse stören.

In der Praxis werden oft an Stellen, an denen das Regressionsmodell Prädiktionen liefern soll, Aussagen über die Zuverlässigkeit benötigt. Gemeinsam mit einem weiteren Interpolationsindikator, der die Abweichungen $(\hat{y}_{\mathbf{x}_a} - y_{\mathbf{x}_a})$ und $(\hat{y}_{\mathbf{x}_b} - y_{\mathbf{x}_b})$ berücksichtigt, können die vorliegenden Bewertungsmaße eine solche Einschätzung der Zuverlässigkeit geben.

3.2 Finden nächster Nachbarn

Die Punkte $\mathbf{x}_{\mathrm{a}}$ und $\mathbf{x}_{\mathrm{b}}$, zwischen denen das Interpolationsverhalten validiert wird, müssen zunächst im Datensatz identifiziert werden. Es ist davon auszugehen, dass ein Punkt $\mathbf{x}_{\mathrm{m}}$ im Merkmalsraum bekannt ist, für dessen Umgebung eine Aussage bezüglich Overfitting benötigt wird. $\mathbf{x}_{\mathrm{a}}$ und $\mathbf{x}_{\mathrm{b}}$ müssen so gewählt werden, dass in der Nähe einer Gerade zwischen $\mathbf{x}_{\mathrm{a}}$ und $\mathbf{x}_{\mathrm{b}}$

lediglich $\mathbf{x}_\mathrm{m}$ liegt, Punkte aus dem Lerndatensatz dürfen sich dort nicht befinden. Dazu werden Winkel im Euklidischen Raum benötigt. Der Winkel zwischen zwei Vektoren $\mathbf{v}_1$ und $\mathbf{v}_2$ berechnet sich durch:

$$\measuredangle(\mathbf{v}_1, \mathbf{v}_2) = \arccos\left(\frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{|\mathbf{v}_1| \cdot |\mathbf{v}_2|}\right). \tag{14}$$

Mit der folgenden Auswahl der Datentupel wird sichergestellt, dass keine anderen Punkte die Regression im Bereich zwischen $\mathbf{x}_\mathrm{a}$ und $\mathbf{x}_\mathrm{b}$ verzerrt haben und ein nicht-lineares Verhalten rechtfertigen: Der Merkmalsraum wird normiert, und die N Datentupel werden ihrer Distanz zu $\mathbf{x}_\mathrm{m}$ (gemessen z.B. durch den Euklidischen Abstand) nach sortiert als $\mathbf{x}_\mathrm{a}$ angenommen und unter den übrigen Datentupeln im Lerndatensatz ein Punkt $\mathbf{x}_\mathrm{search}$ gesucht, für den gilt:

$$\measuredangle((\mathbf{x}_\mathrm{m} - \mathbf{x}_\mathrm{a}), (\mathbf{x}_\mathrm{search} - \mathbf{x}_\mathrm{a})) < \psi, \tag{15}$$

$$d(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{search}) > d(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{m}). \tag{16}$$

Werden mehrere Punkte gefunden, die den Bedingungen genügen, wird der Punkt mit der geringsten Euklidischen Distanz zu $\mathbf{x}_\mathrm{a}$ gewählt.

Anschließend wird geprüft ob für alle Punkte $\mathbf{x}_\mathrm{verify}$ mit

$$\measuredangle((\mathbf{x}_\mathrm{a} - \mathbf{x}_\mathrm{search}), (\mathbf{x}_\mathrm{verify} - \mathbf{x}_\mathrm{b})) < \psi_\mathrm{verify} \tag{17}$$

folgende Bedingung gilt:

$$d(\mathbf{x}_\mathrm{verify}, \mathbf{x}_\mathrm{search}) \geq d(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{search}). \tag{18}$$

Ist das der Fall, wird $\mathbf{x}_\mathrm{search}$ als $\mathbf{x}_\mathrm{b}$ für den aktuellen Punkt $\mathbf{x}_\mathrm{a}$ ausgewählt. Außerdem werden alle Punkte $\mathbf{x}_\mathrm{i}$ als potentielle $\mathbf{x}_\mathrm{a}$ ausgeschlossen, für die gilt:

$$\measuredangle((\mathbf{x}_\mathrm{i} - \mathbf{x}_\mathrm{a}), (\mathbf{x}_\mathrm{m} - \mathbf{x}_\mathrm{a})) > \psi_\mathrm{ignore}. \tag{19}$$

Bei der Anwendung in Abschnitt 4 wird mit $\psi = 30°$, $\psi_\mathrm{verify} = 20°$ und $\psi_\mathrm{ignore} = 120°$ gerechnet.

Die Suche nach geeigneten Paaren $(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{b})$ ist beendet, wenn entweder jeder Punkt $\mathbf{x}_\mathrm{i}$ als $\mathbf{x}_\mathrm{a}$ betrachtet wurde oder s Paare gefunden wurden.

Falls keine Paare gefunden werden, besteht die Möglichkeit, dass $\mathbf{x}_\mathrm{m}$ in einem Extrapolationsbereich oder einem sehr gut abgedeckten Bereich des Merkmalsraums liegt.

Alle beschriebenen Methoden wurden in die MATLAB-Toolbox GaitCAD [15] integriert.

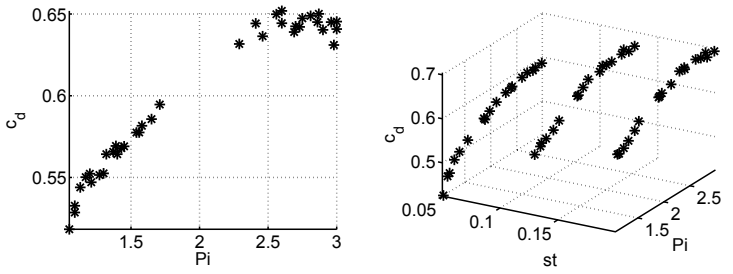
4 Anwendung

4.1 Durchflussverhalten von Labyrinthdichtungen

In [16] werden Data-Mining-Methoden genutzt, um das Systemverhalten von Labyrinthdichtungen in Turbomaschinen zu verstehen. Das von geometrischen, strömungsmechanischen und thermodynamischen Systemparametern abhängige Durchflussverhalten der Dichtungen wird mathematisch abgebildet, damit Vorhersagen für zukünftige Systemkonfigurationen gemacht werden können. In [17] bilden Regressionsmodelle durch KNN die Basis für eine Optimierung von Labyrinthdichtungen, um rechenintensive numerische Strömungssimulationen zu vermeiden und die Optimierung mit vertretbarem Zeitaufwand durchzuführen.

Die erforderlichen Daten werden aus Forschungsprojekten, Dissertationen, Diplomarbeiten und anderen wissenschaftlichen Veröffentlichungen zusammengetragen. Messobjekte und Zielsetzung der Messungen sowie der zugehörigen Arbeiten unterscheiden sich teilweise erheblich. Außerdem divergiert die Aktualität der Messungen. Manche Arbeiten, aus denen Messdaten entnommen wurden, liegen bereits über 40 Jahre zurück, andere lediglich wenige Jahre. Durch das Zusammentragen der Quellen entsteht ein großer Datensatz, mit Hilfe dessen das Durchflussverhalten von Labyrinthdichtungen modelliert wird.

Aufgrund der verschiedenen Quellen weist der Datensatz eine ungleichmäßige Verteilung der Punkte im Merkmalsraum auf. Um die Auswirkungen der ungleichmäßigen Verteilung auf die Modellqualität zu untersuchen, werden Messungen simuliert, die auf der empirischen Korrelation nach Dörr [18] zur Vorhersage von Leckageströmen in Durchblicklabyrinthdichtungen basieren. Die Daten ordnen den Eingangsgrößen st und Pi, auf deren Bedeutung in diesem Beitrag nicht weiter eingegangen wird, die Ausgangsgröße cd zu, welche den sogenannten Durchflussbeiwert und damit die Zielgröße der Problemstellung darstellt. In der Ausgangsgröße wird ein mögliches Messrauschen berücksichtigt. Aus den simulierten Daten werden zwei Datensätze generiert, welche eine ungleichmäßige Verteilung der Daten im Merkmalsraum aufweisen und zur datengetriebenen Modellierung und Validierung verwendet werden. Datensatz D_1 beinhaltet nur die Eingangsgröße Pi und die Ausgangsgröße cd, Datensatz D_2 beinhaltet beide Eingangsgrößen st und Pi und die Ausgangsgröße cd. Die Bilder 2(a) und 2(b) zeigen die Datensätze.



(a) (b)

Bild 2: Visualisierung der Datensätze D_1 und D_2. (a) zeigt die Ausgangsgröße cd über der Eingangsvariablen Pi im Datensatz D_1. (b) zeigt die Ausgangsgröße cd über den Eingangsvariablen st und Pi im Datensatz D_2.

4.1.1 Datensatz D_1

Künstliche Neuronale Netze mit unterschiedlicher Komplexität (Anzahl an Neuronen in der verdeckten Schicht) werden auf D_1 angelernt. Bild 3 zeigt die Modelle mit den geschätzten Ausgangsgrößen und die Lerndaten.

Um das bestangepasste Modell ohne Overfitting zu finden, wird für jede Modellkomplexität eine 10-fache Kreuzvalidierung durchgeführt. Außerdem wurde für die drei Punkte im Merkmalsraum $\mathbf{x}_{\text{T1}} = Pi_{\text{T1}} = 1.5$, $\mathbf{x}_{\text{T2}} = Pi_{\text{T2}} = 2.1$ und $\mathbf{x}_{\text{T3}} = Pi_{\text{T3}} = 2.5$ eine Interpolationsvalidierung ihrer nächsten Nachbarn für die in Bild 3 gezeigten Modelle durchgeführt.

Neuronen	IE	OE	$\frac{OE}{IE}$	$Q_{\text{IV,total,T1}}$	$Q_{\text{IV,total,T2}}$	$Q_{\text{IV,total,T3}}$
1	0.007	0.007	1.059	1	0.94	0.99
3	0.005	0.006	1.33	1	0.96	0.98
5	0.004	0.005	1.136	0.98	0.85	1
7	0.006	0.008	1.323	0.93	0.03	0.74

Tabelle 1: Die mittleren Fehler der zehnfachen Kreuzvalidierungen. Das Regressionsmodell mit 5 Neuronen in der verdeckten Schicht kann als bestes Modell identifiziert werden.

Tabelle 1 zeigt den mittleren IE, OE und den gemittelten Quotienten $\frac{OE}{IE}$ der Kreuzvalidierung (gemittelter $RMSE$) sowie $Q_{\text{IV,total}}$ für $\mathbf{x}_{\text{T1}}$, $\mathbf{x}_{\text{T2}}$ und $\mathbf{x}_{\text{T3}}$ eines über alle Lerndaten angelernten Modells (vgl. Bild 3). KNN mit 5 Neuronen in der verdeckten Schicht werden als gute Regressionsmodelle identifiziert, da sowohl der IE als auch der OE minimal werden bei einem Quotienten $\frac{OE}{IE} \approx 1$. Ein Overfitting wird bei der Modellkomplexität von der Kreuzvalidierung nicht erkannt. Das unerwartete Verhalten vom Mo-

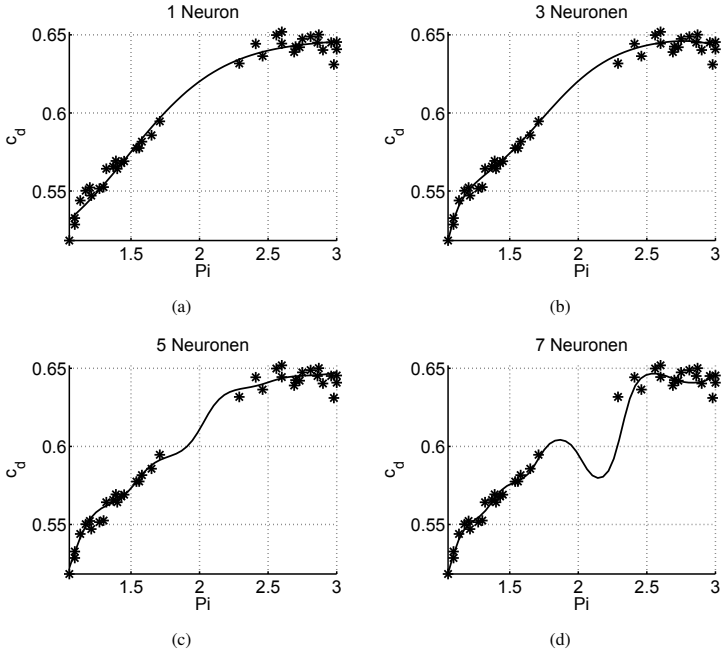


Bild 3: Regressionsmodelle über Datensatz D_1 mit Hilfe von KNN verschiedener Komplexität

dell mit 7 Neuronen im nicht abgedeckten Bereich zeigt sich durch ein niedriges $Q_{\text{IV,total,T2}}$.

Beim Anlernen von KNN werden die initialen Gewichte der Neuronen zufällig gewählt, um das Finden eines globalen Optimums zu ermöglichen. Dadurch kann sich der Verlauf der geschätzten Ausgangsgröße von zwei Modellen gleicher Komplexität unterscheiden, obwohl die gleichen Lerndaten verwendet wurden. Als Beispiel hierfür zeigt Bild 4 wie Bild 3(c) ein Modell, das durch ein KNN mit 5 Neuronen in der verdeckten Schicht auf D_1 angelernt wurde. Im Intervall $Pi = [1.8; 2.4]$ zeigt das Modell aus Bild 4 ein unerwartetes Verhalten.

Die Interpolationsvalidierung des Modells liefert $Q_{\text{IV,total,T1}} = 0.99$, $Q_{\text{IV,total,T2}} = 0.38$ und $Q_{\text{IV,total,T3}} = 1$. Auch hier resultiert das unerwartete Verhalten im nicht abgedeckten Bereich in einem niedrigen $Q_{\text{IV,total,T2}}$. Damit ergänzt die Interpolationsvalidierung die Kreuzvalidierung, die eine generelle (nicht modellspezifische) globale Aussage für im Datensatz erfasste Merkmalsbereiche liefert.

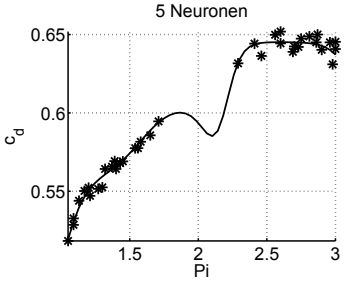


Bild 4: Regressionsmodell über Datensatz D_1. Das Modell hat einen auffälligen Verlauf im Bereich der Eingangsgröße, der nicht durch Daten abgedeckt wird.

4.1.2 Datensatz D_2

Künstliche Neuronale Netze mit unterschiedlicher Komplexität (Anzahl s_{neuron} an Neuronen in der verdeckten Schicht) werden auf D_2 angelernt. Bild 5 zeigt die Modelle mit den geschätzten Ausgangsgrößen und die Lerndaten.

Um das bestangepasste Modell ohne Overfitting zu finden, wird für jede Modellkomplexität eine 10-fache Kreuzvalidierung durchgeführt. Außerdem wird für die vier Punkte im Merkmalsraum aus Tabelle 2 eine Interpolationsvalidierung ihrer nächsten Nachbarn für die in Bild 5 gezeigten Modelle durchgeführt.

	$\mathbf{x}_{\text{T1}}$	$\mathbf{x}_{\text{T2}}$	$\mathbf{x}_{\text{T3}}$	$\mathbf{x}_{\text{T4}}$
st	0.09	0.05	0.13	0.17
Pi	2	1.7	1.7	2.4

Tabelle 2: Punkte im Merkmalsraum, deren Umgebung von der Interpolationsvalidierung untersucht werden

Bild 6 zeigt die Lage der Punkte im Merkmalsraum sowie die Verbindungslinien der identifizierten nächsten Nachbarn im Lerndatensatz.

Tabelle 3 zeigt die Ergebnisse der Kreuzvalidierung und der Interpolationsvalidierung. Die Kreuzvalidierungen lassen aufgrund minimaler OE und $\frac{OE}{IE}$ auf Modelle durch KNN mit 12 bzw. 10 Neuronen als optimale Lösung schließen. Die Interpolationsvalidierung deutet allerdings auf ein Overfitting bei derartigen Modellen in der Umgebung von $\mathbf{x}_{\text{T1}}$ und $\mathbf{x}_{\text{T4}}$ hin. Das wird durch Bild 5 bestätigt.

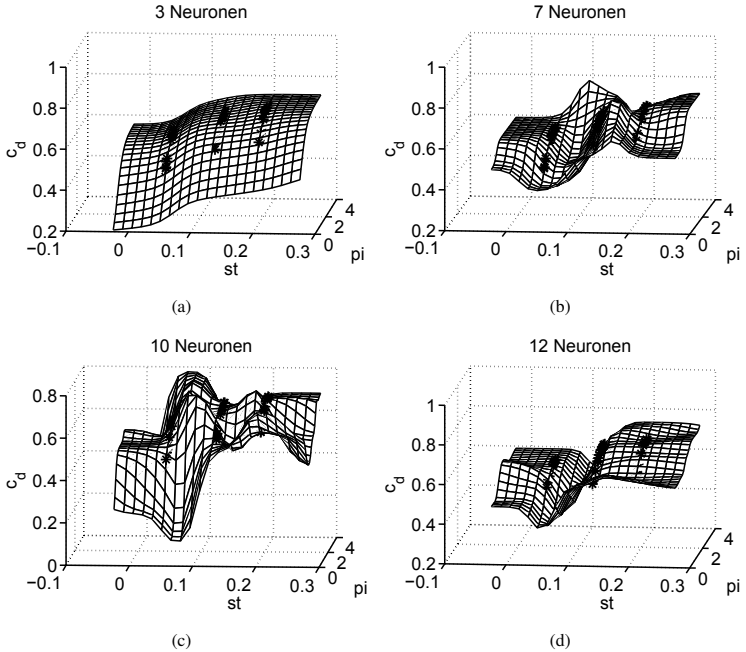


Bild 5: Regressionsmodelle über Datensatz D_2 mit Hilfe Künstlicher Neuronaler Netze verschiedener Komplexität

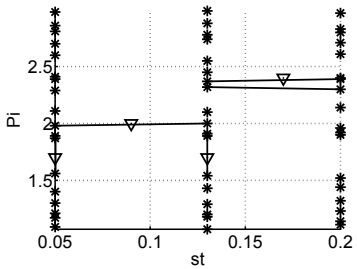


Bild 6: Merkmalsraum von D_2. Punkte für die Interpolationsvalidierung sind durch Dreiecke gekennzeichnet. Verbindungen zwischen ausgewählten Punkten $\mathbf{x}_a$ und $\mathbf{x}_b$ des Lerndatensatzes gemäß Abschnitt 3.2 sind durch Linien gekennzeichnet.

4.2 Setzverhalten von Beton (Datensatz D_3)

In [19] wird versucht, das Setzverhalten von Beton mit Hilfe von KNN zu modellieren. Damit soll unter anderem der Einfluss verschiedener Bestand-

s_{neuron}	IE	OE	$\frac{OE}{IE}$	$Q_{\text{IV,total,T1}}$	$Q_{\text{IV,total,T2}}$	$Q_{\text{IV,total,T3}}$	$Q_{\text{IV,total,T4}}$
3	7.8	10.3	1.5	0.82	0.94	0.93	0.92
7	6.1	6.7	1.4	0.3	0.94	0.95	0.3
10	5.5	5.7	1.2	0.27	0.98	0.95	0.26
12	4.9	5.5	1.2	0.27	0.95	0.86	0.58
14	4.0	6.6	1.9	0.54	0.94	0.94	0.27

Tabelle 3: Ergebnisse der Kreuzvalidierungen und der Interpolationsvalidierungen. IE und OE in $[10^3]$.

teile des Betons auf das Setzverhalten untersucht werden. Für die Anwendung der Interpolationsvalidierung sind Details über den Anwendungsfall nicht nötig. Der Datensatz D_3 ist dem UCI Maschine Learning Repository entnommen und diente als Grundlage für [19]. Er besteht aus 103 Datentupeln mit 7 Eingangsgrößen und einer Ausgangsgröße. Tabelle 4 zeigt eine Übersicht über die Eingangsgrößen des Datensatzes.

Merkmal	min	max	mean	median
x_1	137	374	229.894	248
x_2	0	193	77.974	100
x_3	0	260	149.015	164
x_4	160	240	197.168	196
x_5	4.4	19	8.54	8
x_6	708	1049.9	883.979	879
x_7	640.6	902	739.605	742.7

Tabelle 4: Übersicht über die Eingangsgrößen

Künstliche Neuronale Netze mit unterschiedlicher Komplexität (Anzahl s_{neuron} an Neuronen in der verdeckten Schicht) werden auf D_3 angelernt. Um das bestangepasste Modell ohne Overfitting zu finden wird für jede Modellkomplexität eine 10-fache Kreuzvalidierung durchgeführt.

$\mathbf{x}$	x_1	x_2	x_3	x_4	x_5	x_6	x_7
$\mathbf{x}_{\text{T1}}$	150	120	130	190	8	870	730
$\mathbf{x}_{\text{T2}}$	200	60	52	200	8	800	750

Tabelle 5: Punkte für die Untersuchung auf Overfitting

Tabelle 5 zeigt zwei Punkte im Merkmalsraum, in deren Umgebung die beiden Modelle hinsichtlich Overfitting untersucht werden. $\mathbf{x}_{\text{T1}}$ liegt in einem gut abgedeckten Bereich des Merkmalsraums, $\mathbf{x}_{\text{T2}}$ in einem schlecht abgedeckten. Das heißt die Datentupel von D_3 sind im normierten Euklidischen Raum weiter von $\mathbf{x}_{\text{T2}}$ entfernt als von $\mathbf{x}_{\text{T1}}$.

s_{neuron}	IE	OE	$\frac{OE}{IE}$	$Q_{\text{IV,total,T1}}$	$Q_{\text{IV,total,T2}}$
4	6.126	7.458	1.253	0.84	0.76
5	5.069	6.407	1.263	0.85	0.78
6	4.984	6.512	1.326	0.56	0.39
10	4.979	6.897	1.447	0.55	0.38

Tabelle 6: Ergebnisse der Kreuzvalidierungen und Interpolationsvalidierung

Die Tabelle 6 zeigt die Ergebnisse der Kreuzvalidierungen sowie der Interpolationsvalidierungen. Auf Basis der Kreuzvalidierungen können KNN mit 5 Neuronen in der verdeckten Schicht als bestangepasste Modelle erkannt werden. Das wird auch von der Interpolationsvalidierung bestätigt, die hier hohe Werte auch für den schlecht abgedeckten Bereich aufweist. Interessant ist der Unterschied zwischen Modellen mit 5 Neuronen und 6 Neuronen in der verdeckten Schicht. Während bei der Kreuzvalidierung der Unterschied gering ausfällt, zeigt die Interpolationsvalidierung einen deutlichen Unterschied. Die Interpolationsvalidierung bietet damit eine nützliche Ergänzung zu Kreuzvalidierungsverfahren.

5 Zusammenfassung

In diesem Beitrag wird die Möglichkeit untersucht, Overfitting bei Regressionsmodellen zu erkennen, deren Komplexität und Anpassung von einer Kreuzvalidierung als angemessen bewertet werden. Anhand von verschiedenen Datensätzen und Regressionsmodellen wird gezeigt, dass es möglich ist, Overfitting am Verlauf der geschätzten Ausgangsgröße eines Modells zu identifizieren. Von besonderem Interesse ist diese Methode bei Datensätzen mit mehr als zwei Eingangsgrößen, wie es bei der Optimierung von technischen Systemen oft der Fall ist.

Ein kritischer Punkt der Methode ist das Finden von Datenpunkten, zwischen denen der Verlauf der geschätzten Ausgangsgröße eines Modells untersucht wird. Auf der einen Seite muss geprüft werden, wie allgemeingültig die verwendeten Parameter sind, bzw. wie sie datensatzspezifisch (abhängig von N, s, etc.) angepasst werden können. Auf der anderen Seite muss man unterscheiden zwischen den Anwendungsfällen:

- lokale Bewertung eines Regressionsmodells an einem beliebigen bekannten Punkt $\mathbf{x}_{\text{m}}$ hinsichtlich Overfitting oder Zuverlässigkeit und
- globale Bewertung eines Regressionsmodells hinsichtlich Overfitting oder Zuverlässigkeit.

Für eine globale Bewertung müssen zunächst Punkte im Merkmalsraum identifiziert werden, deren Umgebung untersucht wird. Besonders nicht abgedeckte Bereiche sind hierbei von Interesse. Es werden demnach Methoden benötigt, um solche Bereiche in hochdimensionalen Merkmalsräumen zu finden.

Des Weiteren muss geprüft werden, ob die Parameter der Interpolationsvalidierung allgemein gültig sind, bzw. wie sie datensatzspezifisch (abhängig von N, s, etc.) angepasst werden können.

Literatur

[1] Jin, Y.: A Comprehensive Survey of Fitness Approximation in Evolutionary Computation. *Soft Computing* 9 (2005) 1, S. 3–12.

[2] Pychynski, T.; Blesinger, G.; Mikut, R.; Dullenkopf, K.; Bauer., H.-J.: Modelling the Leakage Behaviour of Labyrinth Seals Using Data Mining Methods. In: *Proc., ASME TURBO EXPO; Glasgow*. 2010.

[3] Daszykowski, M.; Kaczmarek, K.; Vander Heyden, Y.; Walczak, B.: Robust Statistics in Data Analysis - A Review: Basic Concepts. *Chemometrics and Intelligent Laboratory Systems* 85 (2007) 2, S. 203–219.

[4] Hawkins, D. M.: The Problem of Overfitting. *Journal of Chemical Information and Computer Sciences* 44 (2004) 1, S. 1–12.

[5] Reischl, M.; Alshut, R.; Mikut, R.: On Robust Feature Extraction and Classification of Inhomogeneous Data Sets. In: *Proc., 20. Workshop Computational Intelligence*, S. 124–143. KIT Scientific Publishing. 2010.

[6] Protzel, P.; Kindermann, L.; Tagscherer, M.; Lewandowski, A.: Abschätzung der Vertrauenswürdigkeit von Neuronalen Netzprognosen bei der Prozessoptimierung. In: *Proc., VDI-Berichte*, Bd. 1626, S. 335–339. 2000.

[7] Rissanen, J.: Modeling by Shortest Data Description. *Automatica* 14 (1978), S. 465–471.

[8] Wallace, C. S.; Boulton, D. M.: An Information Measure for Classification. *Computer Journal* 11 (1968), S. 185–194.

[9] Papadopoulos, G.; Edwards, P.; Murray, A.: Confidence Estimation Methods for Neural Networks: A Practical Comparison. *IEEE Transactions on Neural Networks* 12 (2001) 6, S. 1278–1287.

[10] Rivals, I.; Personnaz, L.: Construction of Confidence Intervals for Neural Networks Based on Least Squares Estimation. *Neural Networks* 13 (2000) 4-5, S. 463–484.

[11] Moore, A. W.: Cross-Validation for Detecting and Preventing Overfitting. *School of Computer Science Carnegie Mellon University* (2001).

[12] Tetko, I. V.; Livingstone, D. J.; Luik, A. I.: Neural Network Studies. 1. Comparison of Overfitting and Overtraining. *Journal of Chemical Information and Computer Sciences* 35 (1995) 5, S. 826–833.

[13] Hoffmann, F.; Mikut, R.; Kroll, A.; Reischl, M.; Nelles, O.; Schulte, H.; Bertram, T.: Computational Intelligence: State-of-the-Art Methoden und Benchmarkprobleme. In: *Proc., 22. Workshop Computational Intelligence*, S. 15–29. KIT Scientific Publishing. 2012.

[14] Reuter, W.: *Entwicklung einer neuen Methode zur Bewertung von Überanpassung von datenbasierten Modellen mit Lerndaten*. Studienarbeit, Karlsruher Institut für Technologie (KIT). 2013.

[15] Mikut, R.; Burmeister, O.; Braun, S.; Reischl, M.: The Open Source Matlab Toolbox Gait-CAD and its Application to Bioelectric Signal Processing. In: *Proc., DGBMT-Workshop Biosignalverarbeitung, Potsdam*, S. 109–111. 2008.

[16] Pychynski, T.: *Anwendung von Data Mining Methoden zur Analyse von Turbomaschinenkomponenten am Beispiel des Durchflussverhaltens von Labyrinthdichtungen*. Diplomarbeit, Karlsruher Institut für Technologie (KIT). 2009.

[17] Braun, E.; Pychynski, T.; Bauer, H.-J.: An Opensource Framework for Multi-Objective Flow Optimization as Applied to Labyrinth Seals. In: *Proc., 15th International Symposium on Transport Phenomena and Dynamics of Rotating Machinery, ISROMAC-15*. 2014.

[18] Dörr, L.: *Modellmessungen und Berechnungen zum Durchflussverhalten von Durchblicklabyrinthen unter Berücksichtigung der Übertragbarkeit*. Dissertation, Universität Karlsruhe (TH). 1985.

[19] Yeh, I.: Exploring Concrete Slump Model Using Artificial Neural Networks. *Journal of Computing in Civil Engineering* 20 (2006) 3, S. 217–221.

Detektion von Extrapolation

Tobias Ebert, Julian Belz, Oliver Nelles

Department Maschinenbau
Universität Siegen
D-57068 Siegen, Germany
E-Mail: tobias.ebert@uni-siegen.de

Die Detektion von Extrapolation ist ein seit langem betrachtetes Problem und bereits 1988 bemerkte Brooks etwas wichtiges zu der Fachliteratur, in der Extrapolation thematisiert wird:

"Although many researchers recognize the importance of determining if a new point is an extrapolation, the statistical literature is not specific when discussing definitions of the domain, nor does it provide procedures to determine if new prediction points are extrapolation" [1].

Obwohl also eine umfangreiche Sammlung von Methoden und Algorithmen entstanden war, um Extrapolation für verschiedene Randbedingungen zu erkennen, gab es damals wie heute keine allgemeingültige Vorgehensweise, wie Extrapolation zu behandeln ist und ab wann genau man von Extrapolation sprechen kann. Viele verschiedene Forschungsbereiche sind durch das Problem der Erkennung und dem richtigen Umgang mit Extrapolationsproblemen betroffen. Es ist verständlich, dass dabei viele unterschiedliche Auffassungen, Nomenklaturen und Algorithmen entstanden sind, um diese Probleme zu beschreiben und zu lösen. Eine verbreitete Definition von Extrapolation findet sich in folgendem Zitat, das häufig so oder in ähnlicher Form zitiert wird:

"An extrapolation is an inference made about a system's behavior in a new range of variables, from experience in an old (familiar) range" [2].

Wie diese Definition angewendet wird, ist jedoch höchst unterschiedlich. Der Grund dafür liegt in der Kontroverse, was genau der alte oder ursprünglichen Beobachtungsbereich ist, wie er begrenzt ist und wo die Extrapolation beginnt. Beispiele für Anwendungsfälle sind:

- Beschreibung des Raumes oder der Fläche, der durch eine Menge von Punkten besetzt ist [3]. (Mustererkennung)
- Aufprägen eines charakteristischen Verhaltens, jenseits eines ursprünglichen Beobachtungsbereiches. (Regelungstechnik)

- Gegeben sind mehrere Reihen von Daten, z.B. eine alte und eine neue Datenreihe. Die Frage die sich nun stellt: Sind diese zwei oder mehr Datensätzen einander ähnlich bzw. beschreiben sie den gleichen Wertebereich? Sind sie in der Lage, sich gegenseitig zu beschreiben? (Modellbildung)
- Erkennen von Kollision zweier Körper/Hüllen. (Computatergrafik)
- Die Entscheidung, ob eine Messung zu einer bekannten Referenz zugewiesen wird, oder ob eine neue Referenz geschaffen wird [4]. (Strukturüberwachung)
- Erkennung neuer Ereignisse [5, 6]. (Signalklassifizierung)

Es ist ein wiederkehrendes Problem datenbasierter Verfahren, eine oder mehrere Gruppen von Objekten auszuwerten. Alle oben genannten Probleme haben ein Verfahren zur Lösung gemeinsam: Der Aufbau einer Hülle oder Grenze aus einer gegebenen Menge von Objekten und Anwendung dieser Hülle, um Extrapolation zu erkennen.

1 Definitionen von Extrapolation und Interpolation

Bevor die verschiedenen Arten von Algorithmen vorgestellt werden, ist es wichtig zu verstehen, dass viele der folgenden Algorithmen auf sehr unterschiedlichen Meinungen beruhen, was Extrapolation ist, bzw. ab wann diese auftritt. Eine Untersuchung der verbreitetsten Definitionen und der zugehörigen Algorithmen zeigt einige Ähnlichkeiten, aber auch Unterschiede:

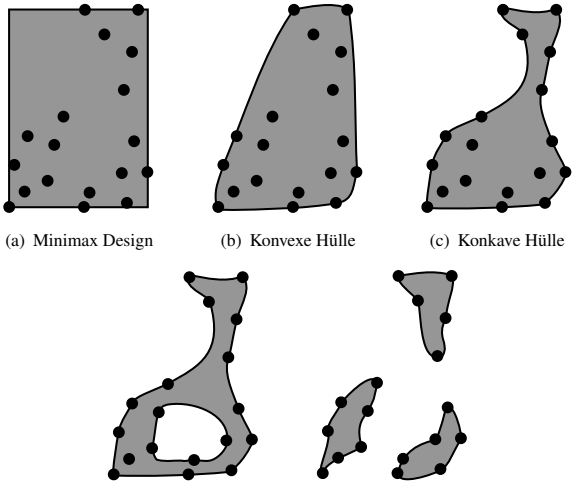
1. Ist es erforderlich, dass die Hülle um die Objekte von konvexer Form ist? Oder ist die Form der Grenze beliebig?
2. Gibt es nur eine Gruppe von Objekten und alles darüber hinaus wird als Extrapolation angesehen, oder gibt es mehrere Gruppen und der leere Raum zwischen diesen Gruppen wird als eine Region von Extrapolation angesehen? Kann es ‘Löcher’ in der Interpolationsregion geben, die als Extrapolation angesehen werden können?
3. Wie ist der Bereich der Interpolation definiert? Durch eine Art von Hülle? Sollte die Dichte der Datenobjekte eine Rolle spielen bei

der Bestimmung des Extrapolationsgebietes? Oder sollte ein statistischer Ansatz angewendet werden um Extrapolation zu identifizieren?

Wie Autoren Inter- und Extrapolation definieren ist sehr unterschiedlich, kann aber kategorisiert werden. Wenn wir die oben genannten Bedingungen betrachten und die Beschreibungen der Methoden in verschiedenen Publikationen vergleichen, in denen ‘extrapolation’, ‘novelty detection’ oder ‘anomaly detection’ explizit erwähnt wird, treten deutlich abgegrenzte Fälle von Extrapolationsdefinitionen hervor.

Definition I - Rechteckige Hülle

Die intuitivste Methode um Extrapolation zu ermitteln, ist durch die Definition des Minimums und Maximums der Objekte als Grenze zwischen Interpolation und Extrapolation. Während viele Autoren diese Spezifikation



(a) Minimax Design (b) Konvexe Hülle (c) Konkave Hülle

(d) Konkave Hülle mit Loch (e) Konkave Hülle für mehrere Gruppen

Abbildung 1: Welche Art von Hülle ist die beste Beschreibung einer Grenze, um ein Innen/Interpolation von einem Außen/Extrapolation für eine gegebene Menge von Objekten zu trennen?

anwenden und ein Überschreiten der Grenze einfach nur als Extrapolation bezeichnen [2, 7, 8], bezeichnen andere Autoren es als '*bounding box*', '*rectangular hull*' [1], '*hypercube*' [9], oder '*minimax design*' [10].

Definition II - Konvexe Hülle

Eine gegenüber des ersten Falles aufwendigere, aber sehr gängige Definition für ein Interpolationsgebiet ist, wenn eine Gruppe von Objekten oder einer Reihe von Daten als Basis dient, um den Wertebereich für die Interpolation zu ermitteln. Dieser Wertebereich wird dabei durch eine Grenze von konvexen Form umschlossen; der Bereich jenseits dieser Grenze oder Hülle ist dann das Extrapolationsgebiet. Die Grenze wird in der Literatur unter andrem benannt als '*independent variable hull*' [11] und '*regressor variable hull*' [12]. Üblicherweise aber wird es als '*minimal convex polyhedron*', oder einfach und am häufigsten nur als '*convex hull*' bezeichnet. Zahlreiche Autoren definieren ein derart eingeschlossenen Gebiet als Interpolationsgebiet [1, 13, 14, 15, 16, 17, 18, 19].

In seltenen Fällen benennen Autoren die Region zwischen mehreren Gruppen von Objekten als Extrapolation: [4] ist ein Beispiel, wo eine Clustering-Technik verwendet wird, um neue Referenzen (Datenpunkte im Extrapolationsgebiet) für Schadenserkennung unter wechselnden Umwelt- und Betriebsbedingungen zur Zustandsüberwachung zu identifizieren.

Definition III - Eine Hülle von beliebiger oder konkaver Form

Ein Fall, der in engem Zusammenhang mit der obigen Definitionen steht, ist der Folgende: Eine Gruppe von Objekten dient als Basis, um den Wertebereich für die Interpolation zu ermitteln, aber die Hülle um eine Gruppe von Objekten ist nicht zwangsweise von konvexer Form. Beispiele, bei denen eine konkave Hülle angewendet und als Interpolationsbereich definiert wird, lassen sich in [20, 21, 9, 22] finden.

Definition IV - Statistischer Ansatz

Eine Reihe von Methoden berechnen nicht eine explizite Hülle, sondern machen einen statistischen Ansatz zur Erkennung von Ausreißern für die Erkennung neuer Ereignisse ('*novelty detection*'). Wann ein Objekt ein

neues Ereignis ist oder nicht, wird dann in der Regel von einer Wahrscheinlichkeit beschrieben. [23] formalisiert die Frage, "*Ist das ein Punkt der Extrapolation?*" zu einem Test: "*Würden wir glauben, dieser Punkt wäre aus der gleichen Verteilung wie die Traingsdaten erzeugt worden, oder aus einer Gleichverteilung?*". In [5] findet sich eine detailreiche Übersicht über solche Ansätze.

Definition V - Dichte der Daten

Es gibt auch Beispiele, in denen weder explizite Hüllen, noch statistische Ansätze verwendet werden. [24] definiert den Bereich, in dem keine Objekte liegen, als Extrapolation und in [20] wird die lokale Dichte von Trainingsdaten mit radialen Basisfunktion-Netzen angenähert, um Extrapolation zu erkennen. Oder wie [21] es formuliert: "*Having this in mind, it is now clear that interpolation can be achieved only in regions of the space where there are enough data to interpolate*". Nach dieser Definition ist also nicht allein die bloße Existenz von umgebenden Daten entscheidend. Die Nähe der Daten im Verhältnis zur Dynamik des zugrundeliegenden Prozesses ist es ebenso, um zwischen Inter- und Extrapolationsgebieten zu unterscheiden.

Die richtige Definition von Extrapolation

Aus den vielen oben aufgeführten, höchst unterschiedlichen Definitionen und Meinungen, was Extrapolation ist (siehe Bild 1), wird offensichtlich, dass es keine allgemein anwendbare Interpretation gibt, um Extrapolation von Interpolation zu unterscheiden. Erst das vorliegende Problem bestimmt, welche Interpretation die am besten geeignetste ist. Wenn nur eine grobe Abschätzung benötigt wird, ist ein rechteckige Hülle ausreichend. Anders verhält es sich, wenn der Bereich der Interpolation klar definiert sein muss: Wenn eine äußere Grenze möglichst gut abgebildet werden muss, sollte eine konvexe Hülle bevorzugt werden. In extremen Fällen reicht eine äußere Hülle nicht aus, z.B. wenn Teilräume ohne Daten zu ausgedehnt sind. Für derart dünn besetzte Räume bietet sich die Verwendung einer konkaven Hülle an. Für Anwendungen, in denen es wichtig ist zu wissen, ob ein Bereich zuverlässig ist oder nicht, muss ein statistischer oder die Datendichte abschätzender Ansatz gewählt werden.

2 Methoden und Algorithmen

Wie aus dem vorherigen Abschnitt hervorgeht, wurden viele Algorithmen entwickelt und über die Jahre verfeinert, um das Problem der Hüllberechnung und der Detektion neuer Ereignisse ('novel detection') anzugehen. Die erste Generation von Algorithmen im Bereich der Hüllberechnung konzentrierte sich darauf, wie eine konvexe Hülse zu berechnen ist. Mehrere markante Strategien [25, 26, 27, 28] entstanden aus diesem Prozess und sind in vielen Bereichen noch immer Stand der Technik. Diese Strategien bilden die Grundlage zur Lösung neuerer Probleme der datengetriebenen Verfahren: Wie Probleme mit vielen Eingangsdimensionen, großen Datenmengen oder nicht-konvexen Grenzen zu behandeln sind.

2.1 Einfache Algorithmen

Verschiedene Methoden und Algorithmen werden heute als Randerscheinung wahrgenommen, weil sie überholt, veraltet oder in ihren Fähigkeiten eingeschränkt sind. Jedoch sind die meisten dieser Verfahren einfach und leicht zu implementieren. Sie werden deshalb hier als Referenzrahmen für die späteren Entwicklungen von hüllbildenden Methoden vorgestellt.

Die intuitivste Methode um Extrapolation zu ermitteln, ist durch die Definition einer oberen und unteren Grenze der Objekte, als Abgrenzung zwischen Interpolation und Extrapolation. Diese Methode wird als Minmax Design, '*bounding box*' oder sehr häufig als '*rectangular box*' bezeichnet [1, 20, 8]. Eine anspruchsvollere Version einer rechteckigen Hülle ist die orthogonale konvexe Hülle, die auch als geradlinige konvexe Hülle bekannt ist. In einer euklidischen Metrik ist eine Menge $Z \subset \mathbb{R}^n$ als orthogonal konvex definiert, wenn für jede Zeile, die zu einer der Achsen des kartesischen Koordinatensystems parallel ist, der Schnittpunkt von Z mit der Zeile leer, ein Punkt oder ein einzelnes Intervall ist [29, 30, 31, 32]. Obwohl dieses Verfahren nicht so genau ist wie eine echte konvexe Hülle, weist die orthogonal konvexe Hülle Vorteile für eine schnelle Berechnung auf, da die Hüllkonstruktion auf einfache binäre Vergleiche der Koordinaten reduziert wird.

Andere wichtige Methoden sind folgende: Bei Problemen mit einer sehr großen Anzahl von Objekten hat [1] ein Screening-Verfahren eingeführt. Eine andere, einfache Methoden ist die kleinste umschließende Kugel [33, 34] und der minimale oder Loewner Ellipsoid [35, 36], der am nützlichsten für normalverteilte Daten ist.

2.2 Algorithmen für konvexe Hüllen

Die etablierten Methoden, um Hüllen zu erstellen und Extrapolation zu erkennen, sind solche für konvexe Hüllen. Die ersten Algorithmen zur Berechnung der konvexen Hülle waren '*Grahams Scan*' [25] und '*Jarvis March*' [26], die von [28] weiter verfeinert wurden. All diese Algorithmen sind jedoch limitiert auf niedrigdimensionale Probleme. Für die konvexe Hülle von Problemen höherer Dimensionen ist es oft notwendig, eine große Anzahl von Simplexen zu berechnen. Sehr verbreitet ist daher [37], der, um den Arbeitsaufwand zu reduzieren, einen Quickhull Algorithmus [27] für n Dimensionen einführte.

Neuere Entwicklungen konzentrieren sich auf anspruchsvollere Anwendungsfelder, wie dynamische konvexe Hüllen [38], genäherte Hüllen für große Datenmengen [39] und konvexe Hüllen für Anwendungen mit vielen Dimensionen [40]. Da Rechenzeit für umfangreichere Probleme eine entscheidende Einschränkung darstellt, entwickeln mehrere Autoren Verbesserungen der Leistungsfähigkeit der Algorithmen durch Parallelisierung [41, 42] und durch angepasste Algorithmen, die besonderen Eigenschaften einer GPU ('*graphics processing unit*') für Innenpunktidentifikation [43, 44] ausnutzen.

2.3 Algorithmen für konkave Hüllen

Ein sehr aktives Feld für algorithmische Geometrie ist die Berechnung von konkaven Hüllen. Es gibt verschiedene Ansätze um eine Grenze von beliebiger Form zu berechnen. Die wichtigsten Design-Konzepte für Algorithmen zur Berechnung konkaver Hüllen basieren auf '*nearest-neighbor*' Techniken, Kernel-Funktionen oder auf einer konvexe Hülle und ihrer Delaunay-Triangulation. Obwohl es keine hüllbildenden Algorithmen sind, gibt es auch einen Bereich der statistischen Methoden für die Erkennung neuer Ereignisse, die das gleiche Problem behandeln. Eine bekannte Möglichkeit, konkave Hüllen zu konstruieren, sind α-shapes, die in [45] vorgestellt worden sind. Sie sind eine Verallgemeinerung der konvexen Hülle, wobei α einen einzustellenden Parameter darstellt. Für $\alpha \rightarrow 0$ nähert sich eine α-Hülle der gewöhnlichen konvexen Hülle an. α-shapes werden aus Voronoi-Diagrammen konstruiert und sind ursprünglich für 2D-Mustererkennung verwendet worden. Später ist das Konzept auf mehrere Dimensionen erweitert worden [46].

Eng mit den α-shapes verwandt sind die so genannten χ-shapes [47]. Der χ-shapes Algorithmus ist einfach, flexibel und effizient in der Konstrukti-

on eines möglicherweise nicht-konvexen, einfachen Polygons. Dieses Polygon, das die Form einer Gruppe von Eingangspunkten in der Ebene charakterisiert, wird in diesem Zusammenhang als eine charakteristische Form bezeichnet. Anstelle von Voronoi-Diagrammen baut der Algorithmus auf der Delaunay-Triangulation der Punkte auf. Alternative Ansätze für ähnliche Ergebnisse basieren auf der Grundlage von '*Grahams Scan*' [3, 48] und besitzen die gleichen Einschränkungen. Ein weiterer Ansatz wird vorgestellt in [49], der mit einer konvexen Hülle beginnt und dann mit einem gewählten Maximalabstand benachbarter Punkte eine konkave Hülle herausschält.

Als Leonard RBF-Netze ('*radial basis function networks*') verwendete, um die Dichte von Trainingsdaten darzustellen [20], führte er damit erstmalig Kernel-Methoden im Bereich der Extrapolationserkennung ein. Mit der Einführung von Support-Vektor-Maschinen (SVM) ist eine neue Klasse von Detektionsverfahren entwickelt worden. Vor allem das Debüt der Ein-Klassen SVMs [50] bot ein leistungsfähiges Werkzeug für die Erkennung 'neuer' Objekte. Ein Nachteil der Ein-Klassen SVMs ist die schwierige Deutung der entstehenden Grenze: Eine exakte Hülle wird nicht berechnet, stattdessen muss eine Schwelle festgelegt werden, um zwischen innen und außen zu unterscheiden. Da keine exakte geometrische Beschreibung der Grenze vorliegt, muss diese zudem aufwendig berechnet werden. Wenn nicht nur Informationen über Objekte auf der Innenseite, sondern auch im Außenbereich, verfügbar sind, ist es möglich und außerordentlich sinnvoll, diese zusätzliche Information zu verwenden. Eine Möglichkeit ist es, die Hülle mit einer Zwei-Klassen SVM zu beschreiben. [22] erweitert das Konzept einer Zwei-Klassen SVM zur Hüllenbeschreibung in einem iterativen Algorithmus. Es wird gezeigt, wie vorteilhaft diese Herangehensweise ist, um einen Entwurfsraum in einem iterativen '*design of experiment*' (DoE) zu erkunden.

Um unabhängige Hüllen für mehrere Gruppen von Objekten zu berechnen, schlägt [51] einen Algorithmus vor, um mehrere Gruppen von Objekten mit dem '*shared nearest neighbours*' (SNN) Cluster-Algorithmus zu trennen [52] und berechnet anschließend eine seperate Hülle für jede Gruppe durch Anwendung eines '*k-nearest-neighbors*' (kNN) Ansatzes. Dieser kNN Ansatz wird erweitert in [53]. Eine Berechnung von Grenzpunkten mit rückwärtsgerichtetem kNN Algorithmus wird in [54] gezeigt. Obwohl hier keine konkrete Hülle erzeugt wird, ist die Fähigkeit, die Grenzpunkte von mehreren Clustern zu isolieren, sehr hilfreich.

Es gibt auch andere Konzepte, um das Problem der Erzeugung konkaver Hüllen anzugehen: Der '*power crust*' Ansatz aus [55, 56] verwendet me-

diale Achsen-Transformation, um eine Darstellung des Körpers als die unendliche Vereinigung seiner maximalen inneren Kugeln zu schaffen. Für den speziellen Fall von Systemen, deren Daten wie an einer Kette aufgereiht im Eingangsraum verteilt sind, haben [9] ihren Ansatz auf parametrischer Kurvenmodellierung aufgebaut. Ihre Idee ist es, das parametrische Kurvenmodell mit einem Hyperschlauch zu umgeben, um die Grenze zu beschreiben. Aus dem Bereich der statistischen Ansätze präsentiert [57] eine Möglichkeit zur Feststellung von Extrapolation als statistischen Test: Ob ein Objekt aus der ursprünglichen Datenverteilung stammt, oder aus einer Gleichverteilung als Nullhypothese. Eine ausgezeichnete Studie für andere statistische Ansätze wird in [5] gegeben. Die gleichen Autoren haben auch eine Übersicht erstellt, in der neuronale Netze verwendet werden [6].

Mit dem Aufkommen von Algorithmen für konkaven Hüllen erkannte [3] die Notwendigkeit von Bewertungskriterien, um Konkavität von Hüllen zu messen. Diese vorgeschlagenen Bewertungskriterien für konvexe und konkave Hüllalgorithmen wurden in [49] weiter ausgearbeitet.

2.4 Vergleiche von Algorithmen

Im Gegensatz zu der großen Menge von Publikationen, in denen Algorithmen vorgestellt werden, ist die Liste der Publikationen, in denen diese Algorithmen verglichen und ausgewertet, sehr kurz. Die erste bemerkenswerte erschien in [27], gefolgt von einem Vergleich von Algorithmen für sequentielle Delaunay-Triangulation in [58]. 1997 präsentierte [59] einen Vergleich von Algorithmen für konvexe Hüllen, betreffend des Problems, Ecken und Flächen korrekt zu zählen. Ein Vergleich der klassischen Algorithmen für konvexe Hüllen - '*Grahams Scan*', '*Jarvis March*' und der '*Quickhull Algorithm*' - kann in [60] gefunden werden. Zuletzt vergleicht [49] den '*swinging arm*' [3], Moreiras KNN-basierte Lösung [51] und χ-shapes [47].

2.5 Umgang mit Extrapolation

In deutlicher Abgrenzung zu Autoren, die einfach feststellen, Extrapolation sollte vermieden werden, gibt es auch eine Reihe von Publikationen über Lösungen. Eine Vereinheitlichung der bestehenden Literatur über nichtlineare Extrapolationsmethoden für skalare Sequenzen ist in [61] aufgeführt. [62] führt asymptotische Extrapolation sowie [8] Extrapolation

mit Splines ein. In [19] wird ein neuronales Netz gezeigt, das zu Extrapolation in der Lage ist. [10] schlägt robuste Vorhersagen und Extrapolations-Designs für fehlspezifizierte verallgemeinerte lineare Regressionsmodelle vor. Zuletzt führte [63] Extrapolation mit '*natural neighbors*' mit Hilfe von 'Geisterpunkten' ein.

3 Zusammenfassung

Aus den oben aufgeführten, zahlreichen und in der Fachliteratur genutzten Definitionen, geht hervor, dass keine für alle Zwecke geeignete Interpretation von Extrapolation existiert. Aus den vielfältigen Anwendungen, in denen dieses Problem auftritt, entstanden sehr differenzierte Meinungen, wie man innen (Interpolation) von außen (Extrapolation) trennt. Dieser Dissens ist nach wie vor existent. Die gezeigte Übersicht über gängige Definitionen gibt eine Hilfestellung, welche Interpretation angemessen ist, um für einen gegebenen Anwendungsfall Extrapolation zu erkennen, zu beschreiben und welche weiterführenden Publikationen und Algorithmen herangezogen werden sollten.

Literatur

[1] Brooks, D. G.; Carroll, S. S.; Verdini, W. A.: Characterizing the domain of a regression model. *The American Statistician* 42 (1988) 3, S. 187–190.

[2] Matalas, N.; Bier, V.: B. Extremes, Extrapolation, And Surprise. *Risk Analysis* 19 (1999) 1, S. 49–54.

[3] Galton, A.; Duckham, M.: What is the region occupied by a set of points? In: *Geographic Information Science*, S. 81–98. Springer. 2006.

[4] Moll, J.; Kraemer, P.; Fritzen, C.: Compensation of environmental influences for damage detection using classification techniques. *Proceedings of 4th European Workshop on Structural Health Monitoring* (2008), S. 1080–1087.

[5] Markou, M.; Singh, S.: Novelty detection: a review—part 1: statistical approaches. *Signal processing* 83 (2003) 12, S. 2481–2497.

[6] Markou, M.; Singh, S.: Novelty detection: a review—part 2:: neural network based approaches. *Signal processing* 83 (2003) 12, S. 2499–2521.

[7] Mcdaniel, M. A.; Busemeyer, J. R.: The conceptual basis of function learning and extrapolation: Comparison of rule-based and associative-based models. *Psychonomic bulletin & review* 12 (2005) 1, S. 24–42.

[8] Gluhovsky, I.; Vengerov, D.: Constrained multivariate extrapolation models with application to computer cache rates. *Technometrics* 49 (2007) 2.

[9] Rejer, I.; Mikolajczyk, M.: A hypertube as a possible interpolation region of a neural model. In: *Artificial Intelligence and Soft Computing–ICAISC 2006*, S. 123–132. Springer. 2006.

[10] Wiens, D. P.; Xu, X.: Robust prediction and extrapolation designs for misspecified generalized linear regression models. *Journal of Statistical Planning and Inference* 138 (2008) 1, S. 30–46.

[11] Cook, R. D.: Detection of influential observation in linear regression. *Technometrics* (1977), S. 15–18.

[12] Montgomery, D. C.; Peck, E. A.: *Introduction to linear regression analysis*. New York: John Wiley & Sons, 2nd edition Aufl. 1992.

[13] Bacchelli Montefusco, L.; Casciola, G.: Algorithm 677 C 1 surface interpolation. *ACM Transactions on Mathematical Software (TOMS)* 15 (1989) 4, S. 365–374.

[14] Patel, M. H.: A linear program to detect extrapolation in predicting new responses of a multiple linear regression model. *Computers & industrial engineering* 28 (1995) 4, S. 787–791.

[15] Baranyi, J.; Ross, T.; McMeekin, T.; Roberts, T.: Effects of parameterization on the performance of empirical models used inpredictive microbiology'. *Food Microbiology* 13 (1996) 1, S. 83–91.

[16] Haffner, P.: Escaping the convex hull with extrapolated vector machines. In: *Advances in Neural Information Processing Systems*, S. 753–760. 2001.

[17] King, G.; Zeng, L.: The dangers of extreme counterfactuals. *Political Analysis* 14 (2006) 2, S. 131–159.

[18] Loh, W.-Y.; Chen, C.-W.; Zheng, W.: Extrapolation errors in linear model trees. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1 (2007) 2, S. 6.

[19] Klesk, P.: Construction of a Neurofuzzy Network Capable of Extrapolating (and Interpolating) With Respect to the Convex Hull of a Set of Input Samples in. *Fuzzy Systems, IEEE Transactions on* 16 (2008) 5, S. 1161–1179.

[20] Leonard, J.; Kramer, M.; Ungar, L.: Using radial basis functions to approximate a function and its error bounds. *Neural Networks, IEEE Transactions on* 3 (1992) 4, S. 624–627.

[21] Verleysen, M.: Learning high-dimensional data. In: *NATO Advanced Research Workshop on Limitations and Future Trends in Neural Computing*. 2001.

[22] Kampmann, G.; Kieft, N.; Nelles, O.: Support Vector Machines for Design Space Exploration. In: *Proceedings of the World Congress on Engineering and Computer Science*, Bd. 2. 2012.

[23] Hooker, G.: Diagnosing extrapolation: Tree-based density estimation. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, S. 569–574. ACM. 2004.

[24] Barnard, E.; Wessels, L.: Extrapolation and interpolation in neural network classifiers. *Control Systems, IEEE* 12 (1992) 5, S. 50–53.

[25] Graham, R. L.: An efficient algorith for determining the convex hull of a finite planar set. *Information processing letters* 1 (1972) 4, S. 132–133.

[26] Jarvis, R. A.: On the identification of the convex hull of a finite set of points in the plane. *Information Processing Letters* 2 (1973) 1, S. 18–21.

[27] Clarkson, K. L.; Shor, P. W.: Applications of random sampling in computational geometry, II. *Discrete & Computational Geometry* 4 (1989) 1, S. 387–421.

[28] Chan, T. M.: Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry* 16 (1996) 4, S. 361–368.

[29] Montuno, D. Y.; Fournier, A.: *Finding xy Convex Hull of a Set of xy Polygons*. Computer Systems Research Group, University of Toronto. 1982.

[30] Nicholl, T. M.; Lee, D.-T.; Liao, Y.-Z.; Wong, C.-K.: On the XY convex hull of a set of XY polygons. *BIT Numerical Mathematics* 23 (1983) 4, S. 456–471.

[31] Ottmann, T.; Soisalon-Soininen, E.; Wood, D.: On the definition and computation of rectilinear convex hulls. *Information Sciences* 33 (1984) 3, S. 157–171.

[32] Karlsson, R. G.; Overmars, M. H.: Scanline algorithms on a grid. *BIT Numerical Mathematics* 28 (1988) 2, S. 227–241.

[33] Welzl, E.: *Smallest enclosing disks (balls and ellipsoids)*. Springer. 1991.

[34] Gärtner, B.: Fast and robust smallest enclosing balls. In: *Algorithms-ESA'99*, S. 325–338. Springer. 1999.

[35] Titterington, D.: Estimation of correlation coefficients by ellipsoidal trimming. *Applied Statistics* (1978), S. 227–234.

[36] Silverman, B.; Titterington, D.: Minimum covering ellipses. *SIAM Journal on Scientific and Statistical Computing* 1 (1980) 4, S. 401–409.

[37] Barber, C. B.; Dobkin, D. P.; Huhdanpaa, H.: The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)* 22 (1996) 4, S. 469–483.

[38] Brodal, G. S.; Jacob, R.: Dynamic planar convex hull. In: *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, S. 617–626. IEEE. 2002.

[39] Kavan, L.; Kolingerova, I.; Zara, J.: Fast approximation of convex hull. In: *ACST*, S. 101–104. 2006.

[40] Khosravani, H. R.; Ruano, A. E.; Ferreira, P. M.: A simple algorithm for convex hull determination in high dimensions. In: *Intelligent Signal Processing (WISP), 2013 IEEE 8th International Symposium on*, S. 109–114. IEEE. 2013.

[41] Cintra, M.; Llanos, D. R.; Palop, B.: Speculative parallelization of a randomized incremental convex hull algorithm. In: *Computational Science and Its Applications–ICCSA 2004*, S. 188–197. Springer. 2004.

[42] González-Escribano, A.; Llanos, D. R.; Orden, D.; Palop, B.: Parallelization alternatives and their performance for the convex hull problem. *Applied mathematical modelling* 30 (2006) 7, S. 563–577.

[43] Tang, M.; Zhao, J.-y.; Tong, R.-f.; Manocha, D.: GPU accelerated convex hull computation. *Computers & Graphics* 36 (2012) 5, S. 498–506.

[44] Tzeng, S.; Owens, J. D.: Finding convex hulls using Quickhull on the GPU. *arXiv preprint arXiv:1201.2936* (2012).

[45] Edelsbrunner, H.; Kirkpatrick, D.; Seidel, R.: On the shape of a set of points in the plane. *Information Theory, IEEE Transactions on* 29 (1983) 4, S. 551–559.

[46] Edelsbrunner, H.; Mücke, E. P.: Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)* 13 (1994) 1, S. 43–72.

[47] Duckham, M.; Kulik, L.; Worboys, M.; Galton, A.: Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition* 41 (2008) 10, S. 3224–3236.

[48] Xu, J.; Feng, Y.; Zheng, Z.; Qing, X.: A concave hull algorithm for scattered data and its applications. In: *Image and Signal Processing (CISP), 2010 3rd International Congress on*, Bd. 5, S. 2430–2433. IEEE. 2010.

[49] Park, J.-S.; Oh, S.-J.: A New Concave Hull Algorithm and Concaveness Measure for n-dimensional Datasets. *Journal of Information Science & Engineering* 28 (2012) 3.

[50] Schölkopf, B.; Williamson, R. C.; Smola, A. J.; Shawe-Taylor, J.; Platt, J. C.: Support Vector Method for Novelty Detection. In: *NIPS*, Bd. 12, S. 582–588. 1999.

[51] Moreira, A.; Santos, M. Y.: Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points (2007).

[52] Ertoz, L.; Steinbach, M.; Kumar, V.: A new shared nearest neighbor clustering algorithm and its applications. In: *Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining*, S. 105–115. 2002.

[53] Chau, A. L.; Li, X.; Yu, W.; Cervantes, J.; Mejía-Álvarez, P.: Border samples detection for data mining applications using non convex hulls. In: *Advances in Soft Computing*, S. 261–272. Springer. 2011.

[54] Xia, C.; Hsu, W.; Lee, M. L.; Ooi, B. C.: BORDER: efficient computation of boundary points. *Knowledge and Data Engineering, IEEE Transactions on* 18 (2006) 3, S. 289–303.

[55] Amenta, N.; Choi, S.; Kolluri, R. K.: The power crust. In: *Proceedings of the sixth ACM symposium on Solid modeling and applications*, S. 249–266. ACM. 2001.

[56] Amenta, N.; Choi, S.; Kolluri, R. K.: The power crust, unions of balls, and the medial axis transform. *Computational Geometry* 19 (2001) 2, S. 127–153.

[57] Hooker, G.: *Diagnostics and extrapolation in machine learning*. Dissertation, stanford university. 2004.

[58] Su, P.; Drysdale, R. L. S.: A comparison of sequential Delaunay triangulation algorithms. In: *Proceedings of the eleventh annual symposium on Computational geometry*, S. 61–70. ACM. 1995.

[59] Avis, D.; Bremner, D.; Seidel, R.: How good are convex hull algorithms? *Computational Geometry* 7 (1997) 5, S. 265–301.

[60] Chadnov, R.; Skvortsov, A.: Convex hull algorithms review. In: *Science and Technology, 2004. KORUS 2004. Proceedings. The 8th Russian-Korean International Symposium on*, Bd. 2, S. 112–115. IEEE. 2004.

[61] Sidi, A.: *Practical extrapolation methods: Theory and applications*. Nr. 10. Cambridge University Press. 2003.

[62] Lopez, S.: An effective parametrization for asymptotic extrapolations. *Computer methods in applied mechanics and engineering* 189 (2000) 1, S. 297–311.

[63] Bobach, T.; Farin, G.; Hansford, D.; Umlauf, G.: Natural neighbor extrapolation using ghost points. *Computer-Aided Design* 41 (2009) 5, S. 350–365.

Zur automatisierten Auswahl signifikanter Regressoren für die Identifikation eines dynamischen Ladedruckmodells

Matthias Kahl[1], Andreas Kroll[1], Robert Kästner[2], Manfried Sofsky[2]

[1]Universität Kassel, FB Maschinenbau, FG Mess- und Regelungstechnik
E-Mail: {matthias.kahl, andreas.kroll}@mrt.uni-kassel.de
[2]IAV GmbH, Bereich Powertrain Mechatronik Systeme, Abteilung Aktoren und Sensoren, Team Applikationstools
E-Mail: {robert.kaestner, dr.manfried.sofsky}@iav.de

1 Einleitung

Ein großer Vorteil der datengetriebenen Modellbildung ist eine weitgehende Automatisierbarkeit des Modellierungsprozesses durch die bereitgestellten Methoden. Doch obliegt es meist dem Anwender eine Auswahl derjenigen Regressoren zu treffen, welche die interessierende Ausgangsgröße im Wesentlichen beschreiben. Bei der Modellierung dynamischer Systeme beinhaltet dies sowohl die Auswahl relevanter physikalischer Größen als auch die Festlegung von Zeithorizonten oder Termen, in denen die Größen eingehen, resultierend in einer großen Anzahl potentieller Regressoren. Für große Mehrgrößensysteme erweist sich dabei ein Ausschließen irrelevanter Regressoren als zunehmend schwierig. Doch ist es von großem Interesse möglichst spärlich parametrierte Modelle zu erhalten, um bspw. einen modellbasierten Reglerentwurf zu ermöglichen oder die Rechenkomplexität beim Einsatz für eine Echtzeitsimulation gering zu halten.

Das aus der linearen Regressionsanalyse bekannte und als Regressorselektion bezeichnete Problem wurde bereits im Bereich der Statistik und dem Data-Mining untersucht. Es wurden Methoden entwickelt, die eine automatisierte Auswahl signifikanter Regressoren für statische Modelle ermöglichen (s. hierzu bspw. [1], [2], [3]). Diese Methoden hielten bei der nichtlinearen dynamischen Modellbildung bisher nur gering Einzug, da dort Effekte den Daten inhärent sind, die nicht durch die Statistik abgedeckt werden und keine ausreichende Transparenz beim Einsatz der Methoden gewährleistet ist. So wird bspw. vorausgesetzt, dass alle relevanten Regressoren in der Kandidatenmenge enthalten sind und zwischen diesen keine Korrelation besteht – praktisch kann dies allerdings nicht gewährleistest werden. Eine Evaluierung der Methoden für den Einsatz bei

der dynamischen Modellbildung ist daher erstrebenswert, um Modelle mit statistisch abgesicherter Struktur bei einer gleichzeitig erweiterten Automatisierung des Modellierungsprozesses zu erhalten.

Der vorliegende Beitrag behandelt exemplarisch zwei Regressorselektionsverfahren – den LASSO-Schätzer [4] und die schrittweise Regression mit dem partiellen F-Test als Bewertungskriterium [5] – im Kontext der Systemidentifikation. Diese Verfahren sind modellbasiert, d.h. sie treffen eine Auswahl signifikanter Regressoren basierend auf einem gewählten Modellansatz, der linear in seinen Parametern (LiP) sein muss. Die Regressorselektion mit den Verfahren entspricht folglich einer Termselektion bei festgelegtem Modellansatz. Betrachtet wird in diesem Beitrag die Anwendung der Verfahren für affine und polynomiale dynamische Modellansätze sowie die Eignung der auf einem affinen Ansatz basierenden Selektion zur Modellierung mittels lokal affiner Fuzzy-Takagi-Sugeno-(TS-)Modelle. Dabei ist eine Bewertung der rekursiven Modellauswertung, welche bei Simulationsaufgaben auftritt, von primärer Bedeutung.

Im folgenden Abschnitt 2 wird zunächst auf die betrachteten Modellansätze und den Identifikationsprozess eingegangen, bevor in Abschnitt 3 eine Beschreibung der hier betrachteten Selektionsverfahren folgt. Anschließend erfolgt in Abschnitt 4 ein Vergleich dieser Verfahren anhand eines am Pkw aufgenommenen Datensatzes mit dem Ziel für eine Simulation geeignete Modelle zu generieren und Rückschlüsse auf die Eignung der Verfahren für den Einsatz im Bereich der dynamischen Modellbildung zu ziehen. Der Beitrag schließt mit einer Zusammenfassung in Abschnitt 5 ab.

2 Modellansatz und Identifikationsprozess

Der Identifikationsprozess gliedert sich in mehrere Schritte, welche die Erzeugung informationsreicher Daten und deren Vorverarbeitung (Vor-Identifikation), die Festlegung einer initialen Modellklasse und der Struktur eines Modellkandidatens (Strukturidentifikation), die Ermittlung der Modellparameter (Parameteridentifikation) und die Modellvalidierung betreffen. Im Folgenden soll kurz auf die Schritte der Struktur- und Parameteridentifikation sowie der Modellbewertung eingegangen werden.

2.1 Betrachtete Modellansätze

Betrachtet werden in diesem Beitrag zunächst Modellansätze, die linear in ihren Parametern und in folgender allgemeiner Form darstellbar sind:

$$\hat{y}(k) = \boldsymbol{\varphi}^T(k) \cdot \boldsymbol{\Theta},\ k = 1, \dots, N\,. \tag{1}$$

Hierin sind $\hat{y}(k)$ der geschätzte Modellausgang, $\boldsymbol{\varphi}^T(k)$ der Regressionsvektor, $\boldsymbol{\Theta}$ der Parametervektor und k die diskrete Zeit bzw. die „Beobach-

tungsnummer“ bei einem statischen System. Im Fall eines statischen Modellansatzes fasst der Regressionsvektor die Eingangsgrößen $\varphi_i(k)$, $i = 1, \dots, p$ des Systems zusammen. Bei der dynamischen Modellbildung wird der Kandidatenraum üblicherweise um die vergangenen Werte der Ausgangsgröße sowie der Eingangsgrößen erweitert. Mit der Annahme, dass das dynamische Systemverhalten durch eine lineare/affine Differenzengleichung beschrieben werden kann, findet häufig der sogenannte ARX-Modellansatz (AutoRegressiv mit eXterner Eingangsgröße) Anwendung:

$$\hat{y}(k|k-1) = \boldsymbol{\varphi}^T(k) \cdot \boldsymbol{\Theta}, \tag{2}$$

$$\text{mit } \boldsymbol{\varphi}(k) = \begin{bmatrix} 1 \\ y(k-1) \\ \vdots \\ y(k-n) \\ u(k-1-T_\tau) \\ \vdots \\ u(k-m-T_\tau) \end{bmatrix} \text{ und } \boldsymbol{\Theta} = \begin{bmatrix} \Theta_0 \\ \Theta_1 \\ \vdots \\ \Theta_n \\ \Theta_{n+1} \\ \vdots \\ \Theta_{n+m} \end{bmatrix}. \tag{3}$$

Hierbei sind u die Eingangsgröße und T_τ die Totzeit [6]. Im dynamischen Fall wird also der aktuelle Modellausgang aus den zurückliegenden Argumenten der Ein- und Ausgangsgröße bis zum Zeitpunkt $k-1$ berechnet. Eine Erweiterung dieser Darstellung auf einen Prozess mit mehreren Eingängen (MISO) ist direkt möglich. In diesem Beitrag werden ausschließlich MISO-Systeme betrachtet, da Systeme mit mehreren Ein- und Ausgangsgrößen (MIMO) meist in mehrere MISO-Systeme überführt werden.

Für den Fall, dass ein nichtlinearer Zusammenhang zwischen den Modelleingangsgrößen und der Modellausgangsgröße angenommen wird, kann das ARX-Modell zu einem NARX-Modell (Nichtlineares ARX-Modell) erweitert werden [7]. Dazu wird der lineare Zusammenhang des ARX-Modells ersetzt durch eine nichtlineare Funktion $f(\cdot)$:

$$\hat{y}(k|k-1) = f(\boldsymbol{\varphi}(k), \boldsymbol{\Theta}). \tag{4}$$

Insbesondere bei der in diesem Beitrag betrachteten Klasse der polynomialen Modelle, welche einen Fall der sogenannten Kolmogorov-Gabor-Modelle [7] darstellen, ist $f(\cdot)$ ein Polynom der Ordnung l, wodurch sich Gl. (4) schreiben lässt als

$$\hat{y}(k|k-1) = \boldsymbol{\varphi}^{\boldsymbol{T}}(k) \cdot \boldsymbol{\Theta} = \Theta_0 + \sum_{i_1=1}^{n+m} \varphi_{i_1}(k) \cdot \Theta_{i_1} \tag{5}$$

$$+\sum_{i_1=1}^{n+m}\sum_{i_2=1}^{i_1}\varphi_{i_1}(k)\cdot\varphi_{i_2}(k)\cdot\Theta_{i_1 i_2}$$
$$+\cdots+\sum_{i_1=1}^{n+m}\sum_{i_2=1}^{i_1}\ldots\sum_{i_l=1}^{i_{l-1}}\varphi_{i_1}(k)\cdot\ldots\cdot\varphi_{i_l}(k)\cdot\Theta_{i_1\ldots i_l},$$

mit

$$\varphi_i(k)=\begin{cases}y(k-i), & 1\le i\le n\\ u(k-i+n-T_\tau), & n<i\le n+m.\end{cases} \tag{6}$$

Als dritte Modellklasse werden Fuzzy-TS-Modelle betrachtet. Ein TS-Modell beschreibt den Zusammenhang zwischen Ein- und Ausgangsgrößen mittels semi-linguistischen WENN-DANN-Regeln der Form [8]:

$$R_i\text{: WENN } \boldsymbol{Z} \text{ IST } \boldsymbol{A}_i \text{ DANN } \hat{y}_i(k|k-1)=\boldsymbol{\varphi}^T(k)\cdot\boldsymbol{\Theta}_i,\ i=1;\ldots;c. \tag{7}$$

Hierbei sind $\boldsymbol{Z}=F(\boldsymbol{z})$ die fuzzifizierten Prämissenvariablen $\boldsymbol{z}$, $\boldsymbol{A}_i$ die eingangsseitige Fuzzy-Referenzmenge der i-ten Regel, definiert über eine multivariate Zugehörigkeitsfunktion $\mu_i(\boldsymbol{z})$. Die Festlegung der Zugehörigkeiten μ_i bewirkt eine unscharfe Partitionierung des Eingangsgrößenraums, welche ein erstes Teilproblem der Identifikation von TS-Modellen darstellt. In diesem Beitrag erfolgt die Partitionierung mittels des Gustafson-Kessel-Algorithmus (s. hierzu [6], [9]). Dabei resultieren orthogonale Zugehörigkeiten der Form

$$\mu_i(\boldsymbol{z})=\left[\sum_{j=1}^{c}\left(\frac{\|\boldsymbol{z}-\boldsymbol{v}_i\|_{\boldsymbol{A}_i}}{\|\boldsymbol{z}-\boldsymbol{v}_j\|_{\boldsymbol{A}_j}}\right)^{\frac{2}{\nu-1}}\right]^{-1}, \tag{8}$$

mit den Clusterzentren $\boldsymbol{v}_i$, $\boldsymbol{v}_j$ und dem Unschärfeparameter $\nu>1$. Prinzipiell realisieren Fuzzy-TS-Modelle eine nichtlineare statische Funktion. Durch Verwendung lokaler ARX-Modelle nach Gl. (2) und (3) können diese zu einem zeitdiskreten dynamischen Modell erweitert werden. Das Gesamtübertragungsverhalten des TS-Modells ergibt sich dann in dem hier beschriebenen Fall durch Überlagerung von c Teilmodellen zu

$$\hat{y}(k|k-1)=\sum_{i=1}^{c}\mu_i(\boldsymbol{z})\cdot\hat{y}_i(k|k-1). \tag{9}$$

Bei den bisherigen Ausführungen wurde angenommen, dass die Modellausgangsgröße $\hat{y}$ aus gemessenen Größen berechnet werden kann. In diesem Fall spricht man auch von einer Einschrittprädiktion. Bei der simula-

tiven Nutzung bzw. rekursiven Auswertung eines Modells stehen die vergangenen Ausgangsgrößen des Systems allerdings nicht zur Verfügung und die prädizierten Werte müssen dem Modell zeitverzögert wieder zugeführt werden. D.h. der Regressionsvektor ergibt sich zu

$$\boldsymbol{\varphi}(k) = [1; \hat{y}(k-1); \dots; \hat{y}(k-n); u(k-1-T_\tau); \dots; u(k-m-T_\tau)]. \tag{10}$$

In diesem Fall spricht man von einem OE-(Output-Error, Ausgangsfehler-) Modell bzw. Nonlinear-Output-Error-(NOE-)Modell.

2.2 Auswahl der Modellstruktur

Die Auswahl einer geeigneten Modellstruktur beinhaltet bei der Modellierung dynamischer Systeme sowohl die Auswahl relevanter physikalischer Einflussgrößen als auch die Festlegung der Dynamikordnung, mit der die Größen eingehen. In diesem Fall spricht man auch von Modellordnungsselektion. D.h. für die hier betrachteten Modellansätze besteht das Ziel darin, die maximale Zeitverzögerung n der Ausgangsgröße und die maximalen Zeithorizonte m sowie eine mögliche Totzeit T_τ der Eingangsgrößen zu ermitteln [10].

Wird hingegen explizit die Auswahl derjenigen Modellterme, die das Ausgangsverhalten eines Systems am besten beschreiben, adressiert, spricht man von Termselektion oder allgemein von Modellstrukturselektion. Bei Verwendung bspw. eines polynomialen Modellansatzes mit einer Ordnung der Nichtlinearität $l = 2$ würde dies sowohl der Auswahl der linearen als auch der bilinearen und quadratischen Terme entsprechen. Die Anzahl an potentiellen Regressoren kann dabei abhängig vom gewählten Modellansatz für dynamische Modelle sehr groß werden. So ergibt sich bspw. in der im Rahmen dieser Arbeit betrachteten Fallstudie eine Kandidatenmenge von 665 Regressoren bei Betrachtung eines polynomialen NARX-Ansatzes mit $l = 2$ für jeweils 5 zurückliegende Werte für jede der 6 Eingangsgrößen sowie der Ausgangsgröße.

Um eine Auswahl signifikanter Regressoren zu treffen, existieren verschiedene Verfahren, die sich in sogenannte Filter-, Wrapper- und Embedded-Verfahren einteilen lassen [2], [3]. Filter-Methoden werden als Vorverarbeitungsverfahren verstanden, die relevante Regressoren oder deren Kombinationen vor der eigentlichen Identifikation des Modells selektieren und somit unabhängig von dem verwendeten Modellierungsansatz sind. Der Selektionsprozess findet alleinig auf dem verwendeten Datensatz mittels statistischer oder informationstheoretischer Kriterien, wie der Partialkorrelation oder dem Transinformationsgehalt, statt. Im erweiterten Sinn können auch die in [11] untersuchte Varianzanalyse oder das in [12]

vorgestellte Verfahren zur Modellordnungsselektion basierend auf einer Clusteranalyse als Filter-Methoden verstanden werden, da sie modellunabhängig arbeiten.

Wrapper-Methoden hingegen bewerten anhand der Approximations- oder Generalisierungseigenschaften eines Modells den Nutzen einer betrachteten Regressorteilmenge. Es erfolgt also ein Vergleich verschiedener Modelle, für deren Identifikation jeweils eine andere Regressorteilmenge verwendet wurde. Soll zusätzlich die Modellkomplexität Gegenstand der Bewertung sein, können bspw. Informationskriterien, wie das AIC oder BIC, Anwendung finden [2]. Beispiele für Wrapper-Methoden sind die schrittweise Regression in Kombination mit dem F-Test oder die in [10] und [13] vorgestellten Verfahren.

Die Embedded-Methoden verbinden die Regressorselektion mit dem Modelltraining [2]. Dabei wird nur ein Modell betrachtet. Zu den Embedded-Methoden können bspw. das Prunen bei künstlichen Neuronalen Netzen oder bestimmte Regularisierungsverfahren gezählt werden, wie LASSO oder non-negative garrote, bei denen durch Einbringen von Nebenbedingungen oder Straftermen in die Parameteridentifikation die Parameter nicht-signifikanter Regressoren zu Null geschätzt werden, was einem Ausschluss des korrespondierenden Regressors entspricht [2].

Um diejenigen Regressoren aus einer gegebenen Kandidatenmenge zu finden, die im Sinne der bei Filter- oder Wrapper-Verfahren verwendeten Kriterien eine optimale Lösung darstellen, müsste jede mögliche Kombination von Regressoren beurteilt werden. Dieses Vorgehen wird vollständige Enumeration genannt und verlangt bei p potentiellen Regressoren das Bewerten von $2^p - 1$ Modellen. Ein Nachteil der vollständigen Enumeration ist der exponentielle Anstieg der Rechenkomplexität bei hochdimensionalen Problemen, wie es bei dynamischen Mehrgrößensystemen der Fall ist. Eine Durchführung der Methode ist dann kaum in annehmbarer Zeit realisierbar, weshalb andere schrittweise oder heuristische Suchstrategien Anwendung finden, die eine Lösung meist nahe dem globalen Optimum in einem tragbaren Zeitraum finden können [2]. Zu diesen Suchstrategien gehören bspw. die Vorwärtsselektion, bei der in jedem Iterationsschritt ein individueller Regressor ausgewählt und dessen Einfluss auf die Ausgangsgröße beurteilt wird, die Rückwärtsselektion, die gegenläufig zur Vorwärtsselektion agiert – also zu Beginn alle potentiellen Regressoren auswählt und in jedem Schritt denjenigen verwirft, der den geringsten Einfluss auf die Ausgangsgröße aufweist. Eine Kombination dieser Verfahren stellt die schrittweise Selektion dar. Bei dieser Suchstrategie werden in jedem Iterationsschritt sowohl der Mehrwert eines neuen Regressors als auch alle zuvor aufgenommen Regressoren beurteilt und ggf. verworfen. Auch (meta-)heuristische Verfahren, wie evolutionäre

Algorithmen oder Schwarmverfahren, können genutzt werden, um den Suchraum zu explorieren [2].

2.3 Parameteridentifikation

Um die Parameter eines affinen ARX- und polynomialen NARX-Modellansatzes zu ermitteln, kann der gewöhnliche Least-Squares-(LS-)Schätzer

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{k=1}^{N} \big(y(k) - \hat{y}(k|k-1)\big)^2 \tag{11}$$

verwendet werden. Dabei werden die Parameter des Modells so bestimmt, dass die Quadratsumme der Residuen minimal wird. Dieses Optimierungsproblems kann explizit gelöst werden. Die Parameter der Teilmodelle eines TS-Modells können simultan (global) in analoger Weise geschätzt werden [6]. Bei Verwendung von OE-Modellen ist das resultierende Schätzproblem nichtlinear in einem Teil der Parameter und muss iterativ durch nichtlineare numerische Optimierungsverfahren gelöst werden [6].

2.4 Modellbewertung

Die Beurteilung der generierten Modelle erfolgt in diesem Beitrag anhand einer quantitativen und qualitativen Beurteilung des Verlaufs der Ausgangsgröße. Zur quantitativen Beurteilung wird der mittlere quadratische Fehler (Mean Squared Error, MSE)

$$J_{\mathrm{MSE}} = \frac{1}{N} \sum_{k=1}^{N} \big(y(k) - \hat{y}(k|k-1)\big)^2 \tag{12}$$

verwendet. Des Weiteren wird als dimensionsloses Fehlermaß der normalisierte MSE (Normalized Mean Squared Error, NMSE) betrachtet [14]:

$$J_{\mathrm{NMSE}} = \frac{\sum_{k=1}^{N} \big(y(k) - \hat{y}(k|k-1)\big)^2}{\sum_{k=1}^{N} (y(k) - \bar{y})^2} \text{ mit } \bar{y} = \frac{1}{N} \sum_{k=1}^{N} y(k). \tag{13}$$

3 Betrachtete Regressorselektionsverfahren

Nachstehend werden die allgemeinen Annahmen für die Modellordnungsselektion und die Grundlagen je eines Wrapper-Ansatzes – der schrittweisen Regression mit dem partiellen F-Test als Bewertungskriterium – und eines Embedded-Verfahrens – des LASSO-Schätzers – beschrieben.

3.1 Allgemeine Annahmen für Modellordnungsselektion

Die betrachteten Verfahren sind modellbasiert, d.h. abhängig vom gewählten Modellansatz, der linear in seinen Parametern sein muss. Mit dem Ziel, die signifikanten Regressoren eines Prozesses

$$y(k) = f(\boldsymbol{\varphi}(k), \boldsymbol{\Theta}) + e(k), \tag{14}$$

mit dem unabhängigen und identisch verteiltem Rauschsignal $e(k)$, zu ermitteln, muss gewährleistet sein, dass der gewählte Modellansatz die Funktion $f(\cdot)$ approximieren kann. Nach [10] kann das System (14), mit der Annahme, dass die Funktion $f(\cdot)$ hinreichend glatt ist, damit eine Taylorentwicklung in der Umgebung D eines gegebenen Arbeitspunktes des betrachteten Systems bis hin zur 2. Ordnung gültig ist, durch Linearisierung um diesen Arbeitspunkt durch

$$y(k) = \Theta_0 + \sum_{i=1}^{n} \varphi_i(k) \cdot \Theta_i + e(k) \tag{15}$$

respektive

$$y(k) = \Theta_0 + \sum_{i=1}^{n} \varphi_i(k) \cdot \Theta_i + \sum_{i=1}^{n}\sum_{j=1}^{n} \varphi_i(k) \cdot \varphi_j(k) \cdot \Theta_{ij} + e(k) \tag{16}$$

angenähert werden. Für den Fall, dass ein Regressor φ_i signifikanten Einfluss auf die Ausgangsgröße y des Prozesses hat, sollte dieser auch einen signifikanten Beitrag in den linearisierten Näherungen (15) und (16) leisten. Mit Verwendung der Modellansätze (2) und (5) mit bilinearen und quadratischen Termen und Anwendung des LASSO-Schätzers und der schrittweisen Regression entspricht die Regressorselektion damit einer Termselektion. Da (2) und (5) globale Modellansätze darstellen, wird angenommen, dass der Bereich D den gesamten Betriebsbereich für die gegebenen Beobachtungen des betrachteten Prozesses darstellt und die so selektierten Regressoren global signifikant sind. Für TS-Modelle gilt diese Annahme i. Allg. nicht.

3.2 Schrittweise Regression

Bei der schrittweisen Regression (SWR) werden ausgehend von einem Initialmodell, welches den Regressor enthält, der am stärksten mit der Ausgangsgröße korreliert, sukzessive weitere signifikante Regressoren in das Modell aufgenommen oder nicht-signifikante aus diesem ausgeschlossen [5]. D.h. es wird abwechselnd eine Vorwärts- und eine Rückwärtsselektion durchgeführt. Die Beurteilung der Signifikanz erfolgt über einen partiellen F-Test. Das Grundprinzip des F-Testes ist zu untersuchen,

welchen zusätzlichen Anteil an der mittleren Quadratsumme der Residuen ein einzelner Regressor erklärt. Dazu werden die Hypothesen

$$H_0: \Theta_i = 0 \text{ und } H_1: \Theta_i \neq 0, \tag{17}$$

mit Θ_i ist Parameter des linearen Regressionsmodells, formuliert. D.h. ein Verwerfen der Nullhypothese im Rahmen des Tests während der Vorwärtsselektion entspricht einer Aufnahme des Regressors in das Modell, da dessen Koeffizient in diesem Fall einen Wert ungleich Null hat. Bei Durchführung der Rückwärtsselektion würde der Regressor bei Annahme der Nullhypothese wieder aus dem Modell entfernt werden.

Die Durchführung einer SWR erfolgt in den folgenden drei Schritten [5]:

1. Initialisierung: Bestimme das initiale Modell
2. Vorwärtsselektion: Bestimme für jeden nicht im Modell befindlichen Regressor den partiellen *F*-Wert und den korrespondierenden *p*-Wert. Resultieren *p*-Werte, die unterhalb eines festgelegten Signifikanzniveaus liegen, wähle den Regressor mit dem kleinsten *p*-Wert in das Modell.
3. Rückwärtsselektion: Bestimme für jeden im Modell befindlichen Regressor den partiellen *F*-Wert und den korrespondierenden *p*-Wert. Resultieren *p*-Werte, die oberhalb eines festgelegten Signifikanzniveaus liegen, entferne den Regressor mit dem größten *p*-Wert aus dem Modell.

Der Algorithmus iteriert solange bis kein Regressor aus der Kandidatenmenge mehr den Eingangstest besteht und kein Regressor mehr aus dem Modell ausgeschlossen wird.

Die schrittweise Regression gehört zu der Klasse der lokalen Suchalgorithmen. Abhängig von der Bildung des Initialmodells und davon wie die einzelnen Regressoren in das Modell aufgenommen oder daraus entfernt werden, können sich daher für dieselbe Kandidatenmenge unterschiedliche Modelle ergeben [15]. Bei Verwendung der SWR kann daher nicht garantiert werden, die global optimale Modellstruktur zu erhalten. Damit der im Rahmen der schrittweisen Regression durchgeführte F-Test zur Überprüfung der Signifikanz einzelner Regressoren vertrauenswürdige Aussagen liefert, müssen die Annahmen, die für eine lineare Regressionsanalyse gefordert werden, gelten [5]. Bspw. muss die dem Prozess unterliegende Störgröße einer Normalverteilung folgen und darf nicht autokorreliert sein. Zudem sollte keine Korrelation zwischen den betrachteten Regressoren bestehen. Insbesondere die Verletzung letzterer Annahme führt dazu, dass der Signifikanztest seine Aussagekraft verliert [15].

3.3 LASSO

LASSO (Least Absolute Shrinkage and Selection Operator) gehört zu den Regularisierungsverfahren. Im Bereich der Systemidentifikation wurde LASSO bspw. in [16] zur Termselektion von polynomialen NARMAX-(Nonlinear AutoRegressive, Moving Average eXogenous)-Modellen genutzt. Bei dem LASSO-Schätzer wird die L_1-Norm angewandt auf den Parametervektor der Schätzung als Strafterm verwendet, wodurch Parameterwerte nicht-signifikanter Regressoren exakt zu Null geschätzt werden [4]. Formal ist der LASSO-Schätzer definiert durch das Optimierungsproblem [1]:

$$\widehat{\boldsymbol{\theta}}_{\text{lasso}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{2}\sum_{k=1}^{N}\left(y(k) - \sum_{i=1}^{p}\varphi_i(k)\cdot\Theta_i\right)^2 + \lambda\sum_{i=1}^{p}|\Theta_i|\,, \quad (18)$$

mit dem Regularisierungsfaktor λ. Bei dieser Formulierung wird davon ausgegangen, dass die Regressoren standardisiert und mittelwertbefreit vorliegen und der konstante Term durch $\widehat{\Theta}_0 = \bar{y}$ geschätzt wird:

$$\frac{1}{N}\sum_{k=1}^{N}\varphi_i(k) = \bar{\varphi}_i = 0, \quad \frac{1}{N}\sum_{k=1}^{N}(\varphi_i(k) - \bar{\varphi}_i)^2 = 1, \quad \forall\, i = 1, \dots, p. \quad (19)$$

D.h. es erfolgt eine Schätzung ohne affinen Term. Aufgrund der Beschränkung des Parametervektors durch die L_1-Norm ist die Lösung von (18) nichtlinear von $y(k)$ abhängig und kann somit nicht geschlossen angegeben werden. Daher werden zur Lösung von (18) numerische Verfahren verwendet, von denen einige bspw. in [1] beschrieben werden.

Der Regularisierungsfaktor λ legt den Grad der Beschränkung fest. Mit steigendem λ steigt die Anzahl der zu Null geschätzten Koeffizienten und die Modellkomplexität sinkt. Um eine geeignete Wahl von λ zu treffen, wird üblicherweise eine Kreuzvalidierung für eine abfallende Sequenz an Werten von λ, ausgehend von dem Wert $\lambda_{\max}$, für den der resultierende Parametervektor $\widehat{\boldsymbol{\theta}}_{\text{lasso}} = \mathbf{0}$ ist, durchgeführt, die es erlaubt den (Einschritt-)Prädiktionsfehler zu schätzen [1]. Dazu wird der gegebene Trainingsdatensatz in K ungefähr gleich große Datensätze $F_1, \dots, F_K$ zufällig aufgeteilt, von denen $K-1$ als Trainingsdatensätze und ein v-ter als Validierungsdatensatz F_v dienen. Es werden K Durchläufe durchgeführt, bei denen eine Modellbildung auf den Trainingsdaten und eine Auswertung auf dem Validierungsdatensatz erfolgt, wobei $v = 1,2, \dots, K$ in den jeweiligen Durchläufen ist. Als Maß für die Prädiktionsgüte kann bspw. der über die Kreuzvalidierung gemittelte MSE verwendet werden [1]. Sei $\kappa: \{1, \dots, N\} \mapsto \{1, \dots, K\}$ eine Funktion, die jede Beobachtung einem der

Datensätze $F_1, \ldots, F_K$ zuweist und $\hat{y}^{-v}$ die geschätzte Ausgangsgröße des Modells, das ohne den *v*-ten Datensatz generiert wurde. Der über die Kreuzvalidierung gemittelte MSE ergibt sich dann für jedes λ zu

$$\mathrm{MSE}_{\mathrm{CV}}(\lambda) = \frac{1}{N}\sum_{k=1}^{N}\left(y(k) - \hat{y}^{-\kappa(k)}(k,\lambda)\right)^2. \tag{20}$$

Als Entwurfsparameters λ wird der Wert gewählt, für den $\mathrm{MSE}_{\mathrm{CV}}(\lambda)$ minimal ist:

$$\lambda_{\mathrm{minMSE}} = \underset{\lambda\in\{\lambda_{\min},\ldots,\lambda_{\max}\}}{\operatorname{argmin}} \mathrm{MSE}_{\mathrm{CV}}(\lambda)\,. \tag{21}$$

Alternativ wird oft eine sogenannte „One-standard-error-Regel“ in Verbindung mit der Kreuzvalidierung angewandt, bei der eine Auswahl des spärlichsten Modells erfolgt, dessen Prädiktionsfehler nicht über einem Standardfehler (Standard Error, SE) des Prädiktionsfehlers des besten Modells liegt [1]:

$$\mathrm{MSE}_{\mathrm{CV}}(\lambda_{1\mathrm{SE}}) \leq \mathrm{MSE}_{\mathrm{CV}}(\lambda_{\mathrm{minMSE}}) + \mathrm{SE}(\lambda_{\mathrm{minMSE}}). \tag{22}$$

Der Standardfehler wird durch

$$\mathrm{SE}(\lambda) = \sqrt{\frac{\sum_{v=1}^{K}(\mathrm{MSE}_v(\lambda) - \mathrm{MSE}_{\mathrm{CV}}(\lambda))^2}{(K-1)\cdot K}}, \tag{23}$$

mit MSE_v ist der MSE auf dem *v*-ten Validierungsdatensatz, geschätzt. Eine Anwendung von LASSO in Verbindung mit der Kreuzvalidierung ist für dynamische LiP-Modelle direkt möglich. In diesem Fall sind in jeder Zeile der Regressionsmatrix die nötigen Werte vorhanden, um eine Einschrittprädiktion durchzuführen und ein Vertauschen der Zeilen im Rahmen der Kreuzvalidierung kann ohne Fehler durchgeführt werden. Für OE-Modelle gilt dies nicht.

Bezüglich einer konsistenten Selektion von Regressoren mittels LASSO sollte keine Multikollinearität vorliegen. Für den Fall, dass einzelne irrelevante Regressoren stark mit den Regressoren des wahren Systems korreliert sind, verhält sich LASSO indifferent bei der Selektion [1]. D.h. der Schätzer ist nicht in der Lage, den wahren vom irrelevanten Regressor zu unterscheiden.

4 Fallstudie Ladedruck

Im Folgenden werden die beiden o.g. Regressorselektionsmethoden an realen Messdaten erprobt. Im Vordergrund steht dabei die Modellierung des Ladedrucks eines Diesel-Pkw-Motors. Die Selektion signifikanter

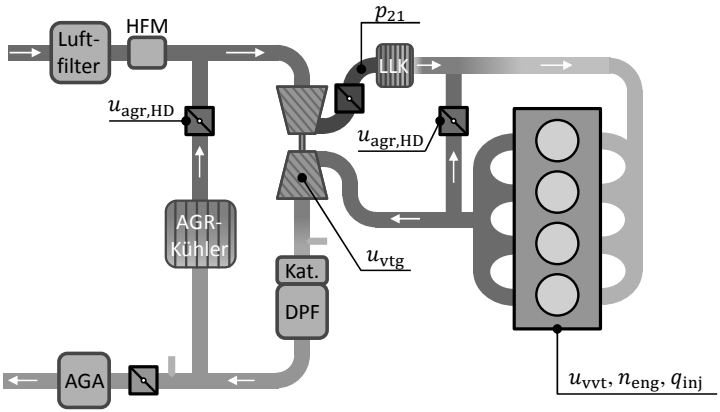
Regressoren erfolgt durch Anwendung des LASSO-Schätzers und der schrittweisen Regression zur Termselektion von ARX- und polynomialen NARX-Modellen. Da das Ziel OE-Modelle sind, wird also ein Ersatzproblem durch Verwendung des ARX-Ansatzes bei der Selektion gelöst, damit der Rechenaufwand in einem durchführbaren Rahmen gehalten wird bzw. die Verfahren überhaupt angewandt werden können. Eine Optimierung der Modellparameter für eine rekursive Modellauswertung folgt nach der Selektion. Die Auswahl für den affinen Modellansatz wird als Basis für die Modellierung lokal affiner Fuzzy-TS-Modelle genutzt.

4.1 Prozess und Daten

Der verwendete Datensatz wurde während einer Überland-Messfahrt aufgezeichnet. Als Versuchsträger diente ein moderner aufgeladener Dieselmotor mit VTG-(variable Turbinengeometrie-)Abgasturbolader, Ladeluftkühler (LLK), Hoch- und Niederdruck-Abgasrückführung (HD- und ND-AGR) und variablem Ventiltrieb (VVT). Eine vollständige Übersicht der betrachteten Motorkonfiguration mit den für die Modellbildung betrachteten Größen ist in Abbildung 1 dargestellt.

Zur Modellierung des Ladedrucks (p_{21} in Abbildung 1) wurde eine Vorauswahl physikalisch relevanter Eingangsgrößen getroffen: Zur Beschreibung des Motorbetriebspunktes werden die Einspritzmenge q_{inj} und die Motordrehzahl n_{Eng} herangezogen. Die VTG-Stellung u_{vtg} des Abgasturboladers beeinflusst im Wesentlichen den Druckaufbau. Aufgrund von Druckausgleichsvorgängen bei Veränderung der AGR-Rate kommt es zu dynamischen Querkopplungen auf den Ladedruck, weshalb sowohl die Hochdruck- als auch die Niederdruck-AGR-Ventilstellung ($u_{\mathrm{agr,HD}}$ und $u_{\mathrm{agr,ND}}$) mit in die Betrachtung aufgenommen werden [17]. Des Weiteren wird der Einfluss der VVT-Stellung u_{vvt} berücksichtigt, da sie den effektiven mittleren Massenstrom durch den Motor und damit auch den Ladedruck beeinflusst.

Der verwendete Datensatz besteht aus insgesamt $N = 92180$ Datenpunkten. Die Aufzeichnung der Messwerte erfolgte mit einer Abtastzeit von $T_0 = 10\,\mathrm{ms}$. Für die Anwendung der Verfahren und die anschließende Überprüfung der generierten Modelle wird der vorhandene Datensatz in einen Trainings- (ersten 50 %) und einen Testdatensatz (letzten 50 %) aufgeteilt. Da die Messwerte während einer Messfahrt und damit im geschlossenen Regelkreis aufgezeichnet wurden, besteht eine Abhängigkeit zwischen ihnen. Zudem kann nicht garantiert werden, dass alle Betriebspunkte des Systems angefahren werden. Insbesondere die Annahme für die Regressorselektion, dass keine Multikollinearität zwischen den Regressoren besteht, wird bei der dynamischen Modellierung gebrochen, da die



AGA : Abgasanlage
AGR : Abgasrückführung
DPF : Dieselpartikelfilter
HFM : Heißfilmluftmassenmesser
Kat. : Katalysator
LLK : Ladeluftkühler

Abbildung 1: betrachtete Motorkonfiguration

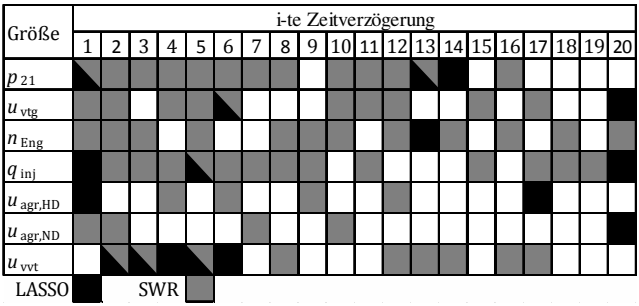
einzelnen Zeitschritte der physikalischen Größen stark korreliert sind.

4.2 Modellansatz 1. Ordnung

Als potentielle Regressoren für das ARX-Modell werden die jeweils zeitverzögerten Werte der Ausgangsgröße und der Eingangsgrößen bis zu einer Dynamikordnung von 20 betrachtet, da die wahre Systemordnung nicht bekannt ist und eine Totzeit nicht ausgeschlossen werden kann. Dies führt auf eine Kandidatenmenge von 140 potentiellen Regressoren. Für die Auswahl einer geeigneten Untermenge an Regressoren mittels LASSO wird der Regularisierungsparameter λ mittels Kreuzvalidierung und der One-standard-error-Regel aus einer Menge von 100 logarithmisch verteilten Werten für λ gewählt. Zur Durchführung wird die MATLAB-Funktion *lasso* verwendet. Das Signifikanzniveau bei der schrittweisen Regression wird im Rahmen der Vorwärtsselektion zu 0,05 und der Rückwärtsselektion zu 0,1 gewählt. Zur Durchführung wird die MATLAB-Funktion *stepwisefit* verwendet. Die Selektionsergebnisse für die gewählte Parametrierung sind in Tabelle 1 dargestellt.

Hinsichtlich der Selektion zeigen sich die verschiedenen Eigenschaften der Verfahren bei Vorliegen nicht-idealer Daten. So nimmt LASSO mit 17 Regressoren im Vergleich zur schrittweisen Regression mit 68 Regressoren eine spärliche Anzahl an Modelltermen auf, was vermutlich auf die Multi-

Tabelle 1: Selektionsergebnisse für affinen Modellansatz mittels LASSO und SWR



kollinearität in der betrachteten Kandidatenmenge zurückzuführen ist. Ein Rückschluss auf die Modellordnung oder eine Totzeit ist in beiden Fällen nicht möglich, da beide Verfahren keine Intervalle selektieren und eine Interpretation aus systemtheoretischer Sicht entfällt. Hinsichtlich der Modellgüte zeigt sich allerdings, dass die resultierende Auswahl der Terme sich als geeignet erweist, um die gemessene Ausgangsgröße sowohl auf den Trainings- als auch den Testdaten zu approximieren (Tabelle 2).

Beide Verfahren minimieren im Zuge der Durchführung den Prädiktionsfehler – LASSO bei der Auswahl des Regularisierungfaktors und die SWR in jedem Iterationsschritt. Dass dieses Vorgehen hinsichtlich einer rekur-

Tabelle 2: Zusammenfassung der Gütewerte für die rekursive Modellauswertung auf Trainings- und Testdaten für selektierten OE-Modellansatz 1. Ordnung, NOE-Ansatz 2. Ordnung sowie den TS-(NOE-)Ansatz

Modellansatz	Selektions-verfahren	$\dim(\boldsymbol{\theta})$	Datensatz	MSE in hPa^2	NMSE
1. Ordnung	SWR	68+1	Training	4506,24	0,039
			Test	5116,33	0,130
	LASSO	17+1	Training	4965,24	0,043
			Test	5447,96	0,138
2. Ordnung	SWR	310+1	Training	1545,25	0,014
			Test	1527,28	0,039
	LASSO	29+1	Training	2970,36	0,026
			Test	2328,64	0,059
TS	LASSO	54	Training	3824,69	0,034
			Test	3842,89	0,097

siven Auswertung der Modelle nicht zu einer optimalen Auswahl führt, ist in Abbildung 2 zu erkennen: Abbildung 2(a) zeigt den Verlauf des MSE auf den Trainings- und Testdaten in Abhängigkeit von λ bei rekursiver Modellauswertung. Dabei wurden die Parameter der für jedes λ selektierten Regressoren mittels LS-Schätzung ermittelt. Abbildung 2(b) zeigt entsprechend die Verläufe des MSE für die Modelle, die in den einzelnen Iterationen der schrittweisen Regression resultieren. In beiden Fällen zeigt sich, dass eine Hinzunahme von Regressoren zwar auch bei rekursiver Auswertung der Modelle eine Verbesserung erzielt, eine Reduktion der Modellkomplexität allerdings möglich ist. So ergibt sich bei zunehmender Komplexität der Modelle ab $\lambda = 0{,}3$ bzw. Iteration Nr. 32 ein annährend konstanter Verlauf des MSE. Dieser Aspekt ist aus pragmatischer Sicht von Bedeutung, da die Rechenkomplexität der Verfahren im Wesentlichen von der Bewertung unterschiedlicher Regressorteilmengen auf Basis einer Einschrittprädiktion abhängt. Bei rekursiver Auswertung würde sich ein erheblich größerer Rechenaufwand ergeben.

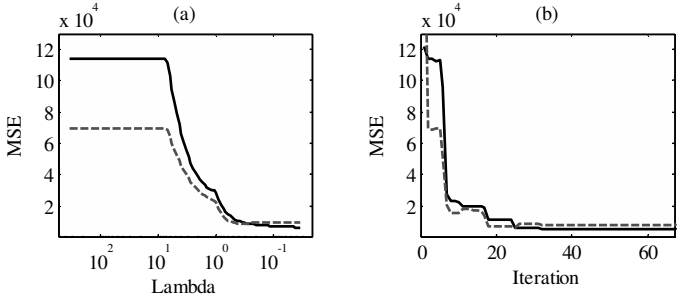


Abbildung 2: Verlauf des MSE auf den Trainings- (–) und Testdaten (- -) bei rekursiver Modellauswertung für die durch LASSO selektierten affinen ARX-Modelle in Abhängigkeit von λ (a) und die durch SWR in den einzelnen Iterationen selektierten affinen ARX-Modelle (b)

4.3 Modellansatz 2. Ordnung

Da die Verfahren bei dem betrachteten Anwendungsproblem keinen Rückschluss auf die Modellordnung und eine mögliche Totzeit zulassen, wird im Folgenden eine Kandidatenmenge aus den jeweils 5 zurückliegenden Werten der Ausgangsgröße und der Eingangsgrößen und sowohl bilinearen als auch quadratischen Terme dieser Werte gewählt. Eine Betrachtung von lediglich 5 zurückliegenden Werten wird vorgenommen, um den Rechenaufwand zu beschränken. Zudem wird vermutet, dass bei der Selektion des Polynomansatzes 1. Ordnung zusätzliche Terme zur Annäherung von Nichtlinearitäten in das Modell aufgenommen wurden, und zur Beschrei-

bung der Dynamik eigentlich weniger zeitverzögerte Werte benötigt werden. Dieser Punkt ist allerdings unklar und kann nur durch einen Vergleich der resultierenden Modellgüte abgewogen werden. Die getroffene Auswahl führt auf eine Kandidatenmenge von 665 Termen. Die Parametrierung der Selektionsverfahren erfolgt analog zur Selektion bei Betrachtung des Polynomansatzes 1. Ordnung. Die Ergebnisse bezüglich der resultierenden Modellgüte sind in Tabelle 2 zusammengefasst. Auf eine detaillierte Darstellung der Selektionsergebnisse wird aufgrund der großen Kandidatenmenge an dieser Stelle verzichtet. Es lässt sich allerdings vermerken, dass LASSO fast ausschließlich Werte zum Zeitpunkt $(k-1)$ und $(k-5)$ auswählt. LASSO selektiert in diesem Fall ein Modell mit 30 Termen mit guten Approximationseigenschaften, die schrittweise Regression hingegen ein Modell mit 311 Termen, welches eine entsprechend bessere Modellgüte aufweist. Im Vergleich zum affinen Modellansatz zeigt sich, dass trotz Vernachlässigung einer Dynamik höherer Ordnung in der Kandidatenmenge, die Dynamik durch den polynomialen Ansatz erfasst werden kann, was auf die Redundanz der einzelnen Zeitschritte zurückzuführen ist. Eine Betrachtung der Verläufe des MSE für eine rekursive Modellauswertung analog zu Abbildung 2 zeigt, dass auch hier eine geringere Modellkomplexität ausreicht (Abbildung 3). Insbesondere zeigt sich, dass bei Verwendung der schrittweisen Regression in den einzelnen Iterationsschritten teilweise Modelle resultieren, die zu einer instabilen Simulation führen und für welche entsprechend keine Ermittlung der Modellgüte möglich war (s. bspw. Bereich zwischen Iteration Nr. 7 und 35 in Abbildung 3(b)). Das kleinste Modell, für welches eine Auswertung möglich war und welches verhältnismäßig geringe Fehlermaße aufweist, beinhaltet 35 Regressoren.

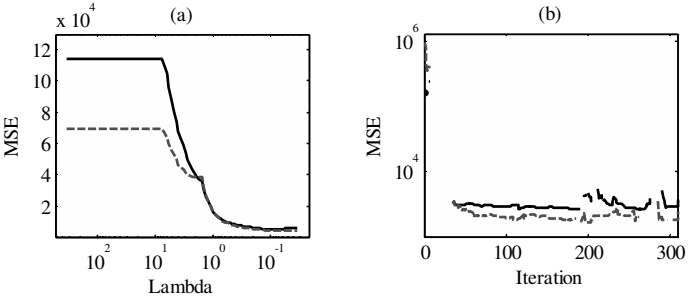


Abbildung 3: Verlauf des MSE auf den Trainings- (–) und Testdaten (- -) bei rekursiver Modellauswertung für die durch LASSO selektierten polynomialen NARX-Modelle in Abhängigkeit von λ (a) und die durch SWR in den einzelnen Iterationen selektierten polynomialen NARX-Modelle (b)

4.4 TS-Ansatz

Abschließend soll die Eignung der durch LASSO selektierten Auswahl an Regressoren zur Modellierung lokal affiner TS-Modelle untersucht werden. Dazu wird das affine Modell beispielhaft mit 17 Regressoren als lokaler Ansatz der Teilmodelle gewählt. Das durch die schrittweise Regression selektierte affine Modell wird aufgrund der hohen Anzahl an Parametern nicht betrachtet. Um eine geeignete Partitionierung des Eingangsraums zu erhalten, wird eine GK-Clusterung im Produktraum vorgenommen. Um lokale Konvergenz bei der Clusterung auszuschließen, wird diese 15-mal zufallsinitialisiert und das NARX-Modell mit dem geringsten Fehler auf den Testdaten ausgewählt. Bei der Initialisierung werden die Prototypen (statistisch) gleichverteilt im Raum angeordnet. Der Unschärfeparameter wurde beispielhaft zu $\nu = 1{,}1$ oder $\nu = 1{,}3$ sowohl für die Clusterung als auch das Modell gewählt, da sich ein Unschärfeparameter in diesem Wertebereich für die Modellbildung als günstig erwiesen hat [18]. Um eine Auswahl der Clusteranzahl hinsichtlich der Modellbildung zu treffen, wurde die Clusteranzahl zu $c \in \{2, \dots, 15\}$ gewählt und eine anschließende Auswertung der resultierenden NARX-Modelle vorgenommen (Abbildung 4). Es zeigt sich, dass keine eindeutige Auswahl der Clusteranzahl anhand des MSE für eine rekursive Modellauswertung getroffen werden kann. Um den Rechenaufwand gering zu halten, wurde eine geringe Clusteranzahl von $c = 3$ präferiert. Um Überlagerungseffekte, die aus einer Vergrößerung der Unschärfe resultieren, zu vermeiden, wurde im Weiteren $\nu = 1{,}1$ gewählt. Diese Kombination führte auch nach einer OE-Optimierung zu guten Ergebnissen (Tabelle 2).

Um die Eignung der Auswahl mittels LASSO für die Modellierung von TS-Modellen abzusichern, wird im Folgenden eine punktuelle Variation der Regressionsgleichung vorgenommen. Zuerst werden sukzessive diejenigen Regressoren entfernt, für die $\sum_{i=1}^{c} \left|\theta_i^{opt}\right|$ im jeweils resultierenden NOE-Modell minimal ist. Dabei werden die standardisierten Parameter betrachtet, wie sie direkt aus der OE-Optimierung resultieren. Da die standardisierten Parameter betrachtet werden, äußert sich dadurch der lineare Einfluss des jeweiligen Regressors auf die Ausgangsgröße. Dieses Vorgehen wurde der Einfachheit halber angewandt. Auch andere Kriterien, wie bspw. die in [19] vorgestellte Sensitivitätsanalyse, können für eine Modellreduktion eingesetzt werden. Auch eine Hinzunahme von Regressoren wird durchgeführt. Dabei wird nur die Hinzunahme von vergangenen Ausgangsgrößen untersucht, da diese den größten Einfluss auf die Ausgangsgröße aufweisen. Der jeweils resultierende MSE ist in Abbildung 5 dargestellt. Die ermittelten Ergebnisse stellen eine punktuelle Auswahl dar und lassen daher keine allgemeingültige Aussage zu. Allerdings zeigte sich für das gegebene Problem auf den Testdaten keine signifikante Verbesserung durch

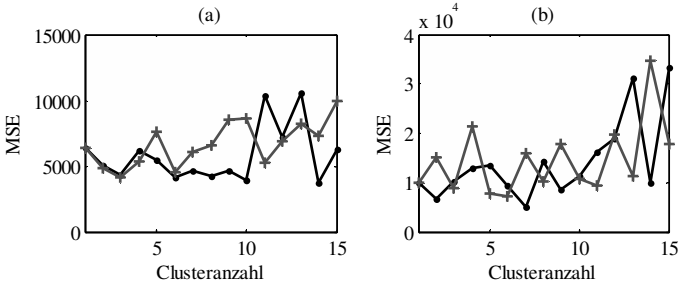


Abbildung 4: Verläufe des MSE bei rekursiver Modellauswertung der resultierenden NARX-Modelle für $\nu = 1{,}1$ (•) und $\nu = 1{,}3$ (+) auf Trainings- (a) und Testdaten (b).

die Hinzunahme von vergangenen Ausgangsgrößen, was auf die Redundanz der gewählten Regressoren zurückzuführen ist. Eine Reduktion der Regressoranzahl hingegen scheint möglich. Allgemein zeigt sich bei diesem Anwendungsproblem, dass die Regressoren eine geeignete Auswahl zur Modellbildung lokal affiner Modelle darstellen.

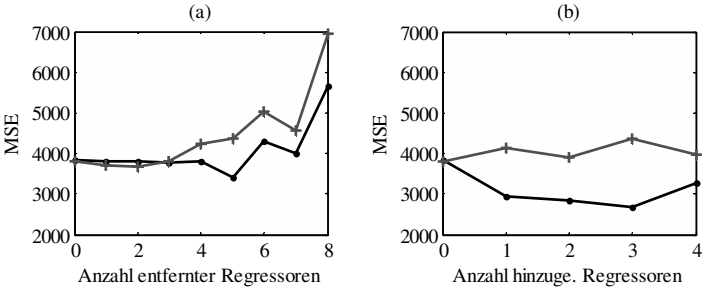


Abbildung 5: Verlauf des MSE auf Trainings- (•) und Testdaten (+) bei rekursiver Modellauswertung bei Entfernen (a) und Hinzuahme (b) einzelner Regressoren des lokal affinen NOE-Modells

4.5 Diskussion

In der betrachteten Fallstudie zeigte sich, dass die Verfahren keine Ermittlung der Modellordnung bei nicht-idealen Bedingungen der Daten ermöglichen. Insbesondere ist aufgrund der vorliegenden Multikollinearität nicht garantiert, dass die Verfahren eine vertrauenswürdige Auswahl an Regressoren treffen. So zeigen sich starke Abweichungen der Selektionsergebnisse bei LASSO und der SWR. Zudem kann nicht davon ausgegangen werden, dass die in Kapitel 3.1 beschriebene allg. Annahme der

Approximierbarkeit durch Linearisierung über den gesamten Betriebsbereich für den betrachteten Datensatz gültig ist. Eine Betrachtung der Modellgüte jedoch zeigt, dass die gewählten Regressoren gut geeignet sind, um den Verlauf der Ausgangsgröße auch bei einer rekursiven Modellauswertung zu approximieren (Abbildung 6). Auch eine Verwendung der durch LASSO selektierten affinen Modellstruktur bei einem lokal affinen TS-Modell erwies sich als praktikabel. Hinsichtlich einer Selektion von Regressoren eines Modells für eine simulative Nutzung zeigte sich in dieser Arbeit, dass bei Verwendung der Methoden eine geeignete aber nicht optimale Auswahl resultiert. Es existieren bereits Verfahren, die anstelle der Bewertung einer Einschrittprädiktion die Selektion auf Basis einer rekursiven Auswertung vornehmen (s. bspw. [13]). Damit ist allerdings ein erheblicher Rechenaufwand verbunden. Bei Verwendung des in [13] vorgestellten Algorithmus in eigenen Untersuchungen konnte zudem festgestellt werden, dass das Verfahren keine guten Ergebnisse liefert, wenn die Hinzunahme eines autoregressiven Terms zu einer instabilen Simulation führt. In diesem Fall konvergiert der Algorithmus lokal und es werden keine autoregressiven Terme selektiert. Um diese Aussagen verallgemeinern zu können, müssen weitere Untersuchungen durchgeführt werden.

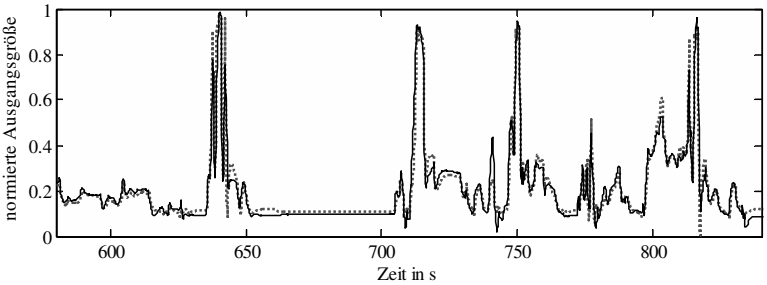


Abbildung 6: Verlauf der normierten gemessenen Ausgangsgröße (--) und der normierten Modellausgabe des durch LASSO selektierten polynomialen OE-Modells 2. Ordnung (–) auf den Testdaten

5 Zusammenfassung und Ausblick

In dem vorliegenden Beitrag wurden zwei statistische Verfahren zur automatisierten Selektion signifikanter Regressoren im Kontext der datengetriebenen dynamischen Modellbildung untersucht – der LASSO-Schätzer und die schrittweise Regression mit dem partiellen F-Test als Selektionskriterium. Eine Erprobung der Verfahren wurde anhand realer Daten eines Diesel-Pkw-Motors vorgenommen. Dabei wurden die Verfahren zur Termselektion bei polynomialen ARX-Modellen 1. und 2. Ordnung ver-

wendet, und eine Eignung der Selektion basierend auf dem Ansatz 1. Ordnung zur Modellierung eines TS-Modells untersucht. Auch wenn die Auswahl keine Interpretation aus systemtheoretischer Sicht zulässt, weisen die selektierten Modelle gute Approximationseigenschaften für das betrachtete Anwendungsproblem auf.

Die durchgeführten Untersuchungen sind als punktuell zu betrachten und sollten in Zukunft an weiteren Beispielen verifiziert werden. Hier könnte die Verwendung anderer Selektionsverfahren zu besseren Ergebnissen führen. Auch könnte es zielführend sein, sich in Zukunft den Verfahren unterliegenden Annahmen zu widmen, um eine Absicherung der Lösung für das betrachtete Anwendungsproblem zu erhalten.

6 Literatur

[1] T. Hastie, R. Tibshirani, J. Friedmann, *The elements of statistical learning: Data mining, inference, and prediction*, 2. Auflage. New York: Springer, 2009.

[2] R. May, G. Dandy, H. Maier, „Review of input variable selection methods for artificial neural network," *The Artificial Neural Networks - Methodological Advances and Biomedical Applications*, K. Suzuki, Hrsg., InTech, 2011.

[3] I. Guyon, A. Elisseeff, „An introduction to variable and feature selection," in *Journal of Machine Learning Research*, Band 3, S. 1157-1182, 2003.

[4] R. Tibshirani, „Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society*, Band B *(Methodological)*, S. 267-288, 1996.

[5] N.-R. Draper, H. Smith, *Applied regression analysis*, 3. Auflage. New York: Wiley, 1998.

[6] A. Kroll, *Computational Intelligence: Eine Einführung in Probleme, Methoden und technische Anwendungen*, München: Oldenbourg, 2013.

[7] A. Janczak, *Identification of nonlinear systems using neural networks and polynomial models: A block-oriented approach*, Berlin: Springer, 2005.

[8] A. Kroll, „Zur Modellierung unstetiger sowie heterogener nichtlinearer Systeme mittels Takagi-Sugeno-Fuzzy-Systemen," in *Proc. 20. Workshop Computational Intelligence*, F. Hoffmann und E. Hüllermeier, Hrsg., 1.-3. Dezember 2010, Dortmund, S. 64-79.

[9] R. Babuska, *Fuzzy modelling for control*, Boston: Kluwer Academic, 1998.

[10] H. L. Wei, S. A. Billings, J. Liu, „Term and variable selection for non-linear system identification," *International Journal of Control*, Band 77, Nr. 1, S. 86-110, 2004.

[11] I. Lind, „Regressor selection in system identification using ANOVA," Dissertation, Department of Electrical Engineering, Linköping university, 2006.

[12] B. Feil, J. Abonyi, F. Seifert, „Model order selection of nonlinear input-output models – a clustering based approach," *Journal of Process Control*, Band 6, Nr. 14, S. 593-602, 2004.

[13] L. Piroddi, W. Spinelli, „An identification algorithm for polynomial NARX models based on simulation error minimization," *International Journal of Control*, Band 76, Nr. 17, S. 1767-1781, 2003.

[14] H. Kötter, H. Sequenz, „Stationäre Motorvermessung mit verschiedenen Methoden und Modellen," in *Elektronisches Management motorischer Fahrzeugantriebe: Elektronik, Modellbildung, Regelung und Diagnose für Verbrennungsmotoren, Getriebe und Elektroantriebe*, R. Isermann, Hrsg. Wiesbaden: Viewig + Teubner, 2010, Kapitel 6, S. 130-166.

[15] D. C. Montgomery, E. A. Peck, G. G. Vining, *Introduction to linear regression analysis*, 4. Auflage. New Jersey: Wiley, 2006.

[16] S. L. Kukreja, J. Lofberg, M. J. Brenner, „A least absolute shrinkage and selection operator (LASSO) for nonlinear system identification," *System Identification*, Band 14, Nr. 1, S. 814-819, 2006.

[17] M. Hafner, „Dynamische Motorvermessung", in Modellgestützte Steuerung, Regelung und Diagnose von Verbrennungsmotoren, R. Isermann, Hrsg. Berlin: Springer, 2003, Kapitel 10, S. 153-162.

[18] A. Kroll, „On choosing the fuzziness parameter for identifying TS models with multidimensional membership functions," *Journal of Artificial Intelligence and Soft Computing Research*, Band 1, Nr. 4, S. 283-300, 2011.

[19] D. Saez, R. Zufiiga, „Takagi-Sugeno Fuzzy model structure selection based on new sensitivity analysis," in *Proc. of the 14th IEEE International Conference on Fuzzy Systems*, Reno, 2005, S. 501-506.

Machine Learning in Predictive Filtering

Jan H. Schoenke, Werner Brockmann

University of Osnabrück
Albrechtstr. 28, 49069 Osnabrück
Tel.: (0541) 969 2439
Fax: (0541) 969 2799
E-Mail: {jschoenk, Werner.Brockmann}@uni-osnabrueck.de

Abstract

This paper considers the Simultaneous State and Parameter Estimation Problem (SSPEP) for non-linear dynamic systems and its stability issues due to a possibly harmful interaction between the two estimation task. Common approaches to state estimation in general and the SSPEP in particular are reviewed. A new approach to the SSPEP is proposed to avoid the harmful interaction in SSPEPs. Experiments with simulated data highlight properties of the proposed approach and compare the estimation performance on a SSPEP with related work. The proposed approach appears to be robust to missing knowledge for initialization.

1 Introduction

On-line state estimation of non-linear dynamic systems from noisy measurements is a challenging task. The estimator has to reduce the noise in the measurements of the state variables and has to overcome a limited sensor resolution for an unbiased and delay-free state estimate. At the same time the estimation has to be stable to enable reliability in the respective application. In addition the processing of the estimator has to comply with real-time constraints in order to keep up with the continuous on-line measurements. The task is even more challenging if the dynamics of the underlying system are partially or completely unknown. The key requirements for delay-free on-line state estimation this paper hence focuses on are:

- stability
- limited prior knowledge
- real-time applicability

Simple filter algorithms like moving average use only past measurements to estimate the state of the system. Their ability to reduce the measurement

noise is limited and only increases as they are allowed to produce delayed estimates. More elaborated methods like Kalman filters enhance the measurements with a model of the dynamic system. These methods are able to perform optimal delay-free state estimates as long as the model is accurate.

If there is no or only an inaccurate model avaliable for the design of the state estimator, a model of the system can be build on-line. But in order to build an accurate model of the system, accurate data i.e. state estimates are required. So optimal state estimates require an accurate model. And building a model of the system requires optimal or at least accurate state estimates. To solve this vicious circle, the Simultaneous State and Parameter Estimation Problem (SSPEP) arises.

The basic concept of all SSPEP approaches consists of concurrent state and parameter estimators. These estimators are coupled by a global iterative on-line estimation scheme that allows the state estimator to use the model of the parameter estimator to form its state estimations and vise versa. If the coupling of the estimators builds a feedback loop between the estimators, the stability of the whole estimation is limited beyond the stability of the individual estimators.

A poor state prediction of the parameter estimator causes a poor state estimate in the next step that may bias the training of the parameter estimator and thus further reduces its prediction accuracy. Starting with a poor state estimate leads to the same harmful interaction and possibly self-reinforcing estimation performance reduction. What further complicates the harmful interaction situation is that it is usually impossible to determine whether an error between a state estimate and a measurement is caused by a poor state estimate of the previous time step or by a poor prediction of the model. This ambiguity may even increase the harmful interaction between the state estimator and the parameter estimator, if the used approach blames the wrong estimator. In this paper a new architecture is presented to overcome the problem of this harmful interaction and to provide stable simultaneous state and parameter estimation even in the absence of prior knowledge about a system model.

The rest of the paper is organized as follows. In Section 2 methods for state estimation, parameter estimation and approaches to the SSPEP are are reviewed. In section 3 the proposed architecture is presented in detail. Experiments to highlight properties of the reviewed methods and to compare approaches to the simultaneous state and parameter estimation problem are shown in section 4.1. In section 5 the results are discussed, and section 6 concludes the paper.

2 Related work

2.1 State Estimation

Methods for state estimation can be divided into model-free and model-based approaches. Model-free approaches only use past measurements for state estimation. A simple yet powerful model-free filter method for state estimation is Exponential Smoothing (ES) [1]. It is computationally cheap, requires only little memory and its design only requires knowledge about time scales of the dynamics of the observed system. The ES filter belongs to the class of Infinite Impulse Response (IIR) filters. IIR filters are optimal for the class of ARIMA models [2], but are possibly unstable due to an internal feedback loop.

Another simple and powerful method for state estimation is the Moving Average (MA) [2]. The MA has still low computational and memory demands, but is expensive compared to ES. The MA filter belongs to the class of Finite Impulse Response (FIR) filters. In [3] a family of Unbiased FIR (UFIR) filters with arbitrary delay is discussed that generalizes the MA, since the estimation delay of the MA always equals the half of its filter length. The prior knowledge requirements of ES, MA and UFIR are equal. FIR Filters are optimal for the class of ARMA models [2] and BIBO stable, which is a valuable property for any practical application. An other appealing property of FIR filters is their ability to produce state estimates with a constant delay.

A common disadvantage of ES, MA and UFIR filter is their low-pass behavior, that is useful for noise reduction, but also dumps higher frequencies of the signal. They thus do not allow for observation of all the system dynamics. Therefore Savitzky and Golay presented in [4] a class of FIR filters that are able to conserve the system dynamics even in turning and inflection points, while simultaneously reducing the noise. And they allow in principal to compensate the delay of the filter. The design of Savitzky-Golay (SG) filters requires prior knowledge about the time scale of the system dynamics and the general behavior of the system on this time scale like monotony or maximal number of turning points.

Model-based approaches use past measurements and an additional model of the system dynamics for state estimation. This kind of estimator dates back to the Kalman Filter (KF) [5] for linear systems which in addition to the system model requires knowledge about the noise properties in from of covariance matrices. The KF handles a state estimation and a covariance matrix of the state estimation in order to recursively minimize the squared

error between the state estimates and measurements. Since the KF only works well with accurate prior knowledge and suffers from stability issues, there has been various extensions of the basic KF. In the direct line of the KF there is the Extended Kalman Filter (EKF) for non-linear systems and the Unscented Kalman Filter (UKF) [6] for improved stability concerning the covariance propagation. The line of H_∞ filters form a version of the KF that is robust to model and covariance errors and thus applicable to a wider range of task where less accurate prior knowledge is available [7]. Recent developments in the field of Optimal FIR (OFIR) filters completely abandon the prior knowledge about the noise by estimating the noise properties from an set of past measurement data, thus their applicability only depends on prior knowledge about the system dynamics [3, 8, 9].

2.2 Parameter Estimation

The on-line parameter estimation task requires a given model structure and a set of adjustable parameters within this structure that are tuned to fit the available data of the observed system. The model structure may be a formula representing physical laws and forces that drive the system under observation. In such a case, the estimated parameters usually carry clear physical interpretations like friction, torque or amplitude and the parameter estimation only needs to fine-tune the parameters. In general, any prior knowledge about the observed system can be used to find a model structure that captures most of the system dynamics leaving only as few as possible parameters for the estimation. But this may become a tedious work.

If there is no or not enough prior knowledge to formulate a system specific model structure, the parameter estimation task becomes a function approximation task on universal function approximators. Thus the designer has to specify both the model structure and an on-line learning algorithm. As limited prior knowledge is a key requirement in this paper, the related work on parameter estimation focuses on function approximation.

Function approximators for on-line learning can be categorized by the effect of their parameters. There are approximators that are Linear In the Parameters (LIP) and Non-Linear In the Parameters (NLIP) [10]. A common structure for LIP and NLIP approximators is shown in equation (2), where $\alpha \in \mathbb{R}^k$ contains the parameters that affect the output linearly, $\beta \in \mathbb{R}^m$ contains the parameters that affect the output non-linearly and $\phi : \mathbb{R}^n \to \mathbb{R}^k$ is an non-linear mapping from input space to (linear) parameter space[1].

[1]The Multiple Input Single Output (MISO) system in equation (2) can easily be extended to a Multiple

As long as β is adjusted by the learning algorithm, the update of the approximator behaves NLIP. If only α is adjusted by the learning algorithm, the update of the approximator behaves LIP. So every NLIP approximator can be seen as a LIP approximator as long as β remains unchanged by learning.

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^d \tag{1}$$
$$f(\mathbf{x}) = \alpha \cdot \phi(\mathbf{x}, \beta) \tag{2}$$

Examples for NLIP approximators are Multi Layer Perceptrons (MLP), Radial Basis Functions (RBF) and Fuzzy Systems. The advantage of NLIP approximators is their ability to achieve an accurate approximation with less parameters compared to a LIP approximator. A major drawback of NLIP approximators is the necessity to solve a non-linear optimization problem for any parameter update causing high computational demands that even can violate real-time constraints.

Examples for LIP approximators are Polynomials, Wavelets and Splines. RBF and Fuzzy Systems with fixed function centers and premises, respectively, are used as LIP approximator, too. LIP approximator allow a computational cheap evaluation and parameter update, but possibly suffer from the *curse of dimensionality*. This means their number of parameters may grow exponentially with the number of input dimensions. The design of a LIP approximator requires prior knowledge about the range of the state variables and the overall complexity of the mapping that describes the dynamics of the system. Due to their low computational complexity coupled with good approximation quality, LIP approximators are optimally suited for on-line function approximation concerning real-time applicability and prior knowledge demands. Thus only on-line learning algorithms for LIP approximators will be reviewed next.

In general a learning algorithm $l : \mathbb{R}^k \times \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^k$ for LIP approximators maps a current parameter vector $\alpha_i \in \mathbb{R}^k$ and an learning datum $(\mathbf{x}_i, y_i)$ that contains an instance $\mathbf{x}_i \in \mathbb{R}^n$ and a label $y_i \in \mathbb{R}$ to a new parameter vector $\alpha_{i+1} \in \mathbb{R}^k$. The lower index i represents the current time step. Methods for on-line learning can be divided into first and second order algorithms. A general formal representation for nearly all first and second order learning algorithms is shown in equations (3) and (4).

$$\alpha_{i+1} = a \cdot \alpha_i + b \cdot (y_i - \hat{y}_i) \cdot S_i \phi(\mathbf{x}) \tag{3}$$
$$S_{i+1} = c \cdot S_i + d \cdot S_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_i) S_i \tag{4}$$

Input Multiple Output (MIMO) system by using one MISO system for each output. So for simplicity all formal descriptions concerning approximators are presented here for MISO systems.

Here α_i represents the parameter vector and S_i the corresponding second order information of the algorithm. For first order algorithms S_i is constant and thus only the update equation (3) is used. $\hat{y}_i$ is the evaluation of the approximator using α_i and $\mathbf{x}_i$, y_i is the desired label of the learning date $(\mathbf{x}_i, \mathbf{y}_i)$ and $\mathbf{x}_i$ the corresponding instance. By specifying a, b, c and d a learning algorithm is chosen and possible tuning parameters of the algorithm are set.

Examples for first order algorithms are Gradient Descent (GD) [11], Passive Aggressive (PA) [12] and Incremental Risk Minimization Algorithm (IRMA) [13]. They only use the current learning date to update the parameter vector α_i, are computationally cheap and can adapt even to changing target functions. But they are prone to noise and also suffer from fatal forgetting when used on trajectory-based learning data. As first order algorithms are prone to noise, they need prior knowledge about the noise-level of the learning data.

Examples for second order algorithms are Recursive Least Squares (RLS) [14], Gaussian Herding (GH) [15] and the Second Order Perceptron (SOP) [16]. Second order algorithms store additional information to estimate second order derivatives of the loss function and adapt the parameter update of α_i to the data at hand. They are robust to noise and converge more rapidly than first order algorithms, but are computationally more expensive than first order algorithms. They can handle trajectory-based learning data and require no prior knowledge about any property of the system under observation. As an example for the notion in equations (3) and (4), the RLS algorithm is provided by $a = c = 1$ and $b = d = \frac{1}{1+\phi(\mathbf{x}_i)^T S_{i,j} \phi(\mathbf{x}_i)}$

2.3 Simultaneous State and Parameter Estimation

Mainly there are two classes of approaches for simultaneous state and parameter estimation. On the one hand the twofold estimation problem can be rewritten as one state estimation problem by adding the system parameters to the state vector. This way even linear systems may cause non-linear estimation problems and thus drastically increase the complexity of the problem. As the two estimation tasks are handled by only on estimator, the estimation tasks have fully and up-to-date information about each other, which enables feedback and thus harmful interaction between them. This kind of approaches is also called *joint estimation*.

On the other hand separate estimators for state and parameters are used that interact by using each others current estimates to form their own next esti-

mate. Although this approach allows a control of the interaction between the estimation tasks, feedback is still enabled due to the mutual estimation usage. This kind of approach is also called *dual estimation*.

Both approaches are analyzed in [17] using Kalman filter techniques as estimators and MLPs as approximators. Their results show a higher accuracy and faster convergence rate for the *dual estimation* approach. The *joint estimation* approach can be found in recent applications like [18], too, where prior knowledge about the system model allows simplifications that strongly reduce the complexity of the estimation problem.

A common drawback of the *joint estimation* and *dual estimation* approach is their possible instability due to harmful interaction. A stable estimation with these approaches requires prior knowledge about the system state and the system parameter in order to form a initialization for the whole estimation that stably converges.

An other recent approach presented in [19] disables feedback between the two estimation task by only allowing a one way interaction between them. The parameters of the system are estimated on a moving window of past measurements. The state estimator uses the parameter estimate to form its state estimate, but does not feedback its state estimate to the parameter estimator. This way a harmful interaction is impossible and thus a stable estimation is provided. A drawback of this approach is its need for prior knowledge about a limited and convex subspace of the parameter space that should be searched for the optimal parameter vector. This subspace is rasterized to form a finite set of possible solutions. For each possible solution a observer is designed and the best operating observer is used as the solution. This results in high computational demands for the processing of all observers and may contradict real-time constraints.

In summary, model-free state estimators can only reduce noise in measurements for delayed estimates and model-based state estimators require prior knowledge about the system in form of an accurate model. The *joint estimation* and *dual estimation* approaches to simultaneous state and parameter estimation suffer stability issues due to harmful interaction. The approach in [19] is stable concerning harmful interaction, but this stability comes for the cost of prior knowledge about the system parameters and large computational costs that limit its real-time applicability.

Thus – to the best of our knowledge – there is no approach to on-line simultaneous state and parameter estimation that is stable, real-time applicable and at the same time needs no prior knowledge about system parameters

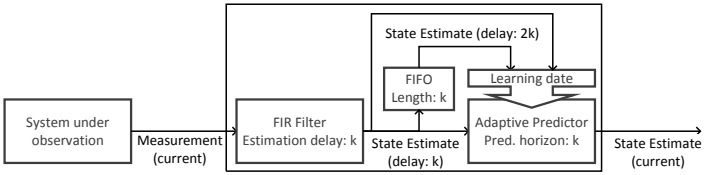


Figure 1: Scheme for the coupling of the state and parameter estimator within the proposed approach.

while producing delay-free estimates. Hence our aim is to find an architecture that avoids feedback between the two estimation tasks of the SSPEP like e.g. in [19] without the need for prior knowledge about the system parameters and low computational demands.

3 Feedback-free Approach for Simultaneous State and Parameter Estimation

3.1 General Concept

Our approach uses two separate estimators for state and parameter estimation. They are coupled in a estimator-predictor scheme that is shown in Figure 1. The state estimator uses a moving window of past measurements to build a model-free state estimate with fixed delay. In order to guarantee a fixed delay of the state estimate of k time steps a FIR filter is used.

The parameter estimator uses the consecutive state estimates of the FIR filter and a FIFO storage of length k to produce learning data for an on-line learning function approximator. Thus the function approximator behaves like a predictor for the state of the system and can therefore compensate the delay of the state estimates provided by the FIR filter. To achieve high prediction accuracy and at the same time meet real-time requirements, a LIP approximator is used for function approximation. And as the learning data for the predictor are gathered along a trajectory in the state space, a second order learning algorithms is used. In this approach the interaction of the two estimation tasks is one way, because the parameter estimator uses the state estimates of the FIR filter and the FIR filter only uses measured data to form its estimation. Thus a feedback loop is avoided and a harmful interaction is impossible.

3.2 Formal Description

In this section the proposed approach is formalized in a mathematical structure. The state of the system in time step i is denoted by $\mathbf{x}_i$ and the respective single state variables by $x_{i,j}$. The corresponding estimates are denoted by $\hat{\mathbf{x}}_i$ and $\hat{x}_{i,j}$, respectively, and the measurements by $\mathbf{y}_i$ and $y_{i,j}$ The structure of the moving window state estimator is shown in equation (6).

$$\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,n}) \tag{5}$$

$$\hat{x}_{i-k,j} = \sum_{p=1}^{P} a_{j,p} \cdot y_{i-p,j} \tag{6}$$

$$\hat{\mathbf{x}}_{i-k} = (\hat{x}_{i-k,1}, \ldots, \hat{x}_{i-k,n}) \tag{7}$$

For each state variable estimate $\hat{x}_{i,j}$, the set of parameters $a_{j,p}$ defines the FIR filter coefficients for the estimation. The learning data scheme for the second order on-line learning algorithm is shown in equation (8). For each state variable $x_{i,j}$ in time step i a learning date is generated that has the same delay between instance and label as the state estimator and uses the data from the FIFO storage.

$$(\hat{\mathbf{x}}_{i-2k}, \hat{x}_{i-k,j}) \tag{8}$$

So the learning of each predictor f_j for the single state variable estimates $\hat{x}_{i,j}$ follows equation (9) and provides in each time step i a parameter vector $\alpha_{i,j}$ for the individual predictor f_j. All single parameter vectors $\alpha_{i,j}$ are gathered in A_i for compact representation. In the same way $\mathbf{l}$ gathers all learning algorithms.

$$\alpha_{i+1,j} = l_j(\alpha_{i,j}, \hat{\mathbf{x}}_{i-2k}, \hat{x}_{i-k,j}) \tag{9}$$

$$A_i = (\alpha_{i,1}, \ldots, \alpha_{i,n}) \tag{10}$$

$$\mathbf{l} = (l_1, \ldots, l_n) \tag{11}$$

$$A_{i+1} = \mathbf{l}(A_i, \hat{\mathbf{x}}_{i-2k}, \hat{\mathbf{x}}_{i-k}) \tag{12}$$

The state estimation of the current state of the system $\hat{\mathbf{x}}_i$ is obtained by the evaluation of all predictors $\mathbf{f} = (f_1, \ldots, f_n)$ on the current state estimate $\hat{\mathbf{x}}_{i-k}$ of the filter with the current parameter vectors $\alpha_{i,j}$ as shown in equation (13).

$$\mathbf{f}(\mathbf{x}_i) = (f_1(\alpha_{i,1}, \mathbf{x}_i), \ldots, f_n(\alpha_{i,n}, \hat{\mathbf{x}}_i)) \tag{13}$$

$$\hat{\mathbf{x}}_i = \mathbf{f}(\hat{\mathbf{x}}_{i-k}) \tag{14}$$

3.3 Algorithmic Implementation

In Algorithm 1 the single steps of the simultaneous state and parameter estimation scheme in Figure 1 are presented. The algorithm receives consecutive state measurements $\mathbf{y}_i$ and produces consecutive state estimates $\hat{\mathbf{x}}_i$. The setup up of the algorithm incorporates the specification of the FIR filter coefficients $a_{j,p}$ and the corresponding estimation delay k, the predictors $\mathbf{f}(\mathbf{x})$ and learning algorithms $\mathbf{l}(A_i, \hat{\mathbf{x}}_{i-2k}, \hat{\mathbf{x}}_{i-k})$. In addition, the FIFO storage to produce the learning dates should already be filled with consecutive state estimations.

The estimation process starts by reading the current measurement of the system state $\mathbf{y}_i$ and estimating filtered and thus the delayed state of the system $\hat{\mathbf{x}}_{i-k}$. The state estimate $\hat{\mathbf{x}}_{i-k}$ is added to the FIFO storage and a learn step is done. So the predictor can may use of the current learning datum to predict the current state of the system $\hat{\mathbf{x}}_i$ and finally the oldest state estimate is removed from the FIFO storage.

The architecture presented in our approach is similar to [19] aiming for a stable estimation by avoiding harmful interaction. But here the state instead of the parameters is estimated on a moving window and no prior knowledge about system parameters is required. An other similarity can be found in the comparison with the *dual estimation* approach in [17]. The use of the KF there in fact equals the standard RLS with forgetting algorithm, which is a second order on-line learning algorithm as required in our approach.

Data: data stream of state measurements $\mathbf{y}_i$
Result: state estimations $\hat{\mathbf{x}}_i$
state estimator: $a_{j,p}$, state estimation delay: k,
state predictor: $\mathbf{f}(\mathbf{x})$, learning algorithm: $\mathbf{l}(\alpha, \mathbf{x}, y)$,
FIFO storage containing the last k measurements of $\mathbf{y}$
while *new datum is available* **do**
  read current measurement $\mathbf{y}_i$
  calculate state estimate $\hat{\mathbf{x}}_{i-k}$
  add state estimate $\hat{\mathbf{x}}_{i-k}$ to FIFO storage
  do one learning step $\alpha_i = \mathbf{l}(\alpha_{i-1}, \hat{\mathbf{x}}_{i-2k}, \hat{\mathbf{x}}_{i-k,j})$
  predict current state estimate $\hat{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}, \alpha_i)$
  remove state estimate $\hat{\mathbf{x}}_{i-2k}$ from FIFO storage
end

Algorithm 1: Description of the approach to simultaneous state and parameter estimation for an algorithmic implementation.

4 Experimental Comparisons

4.1 Experimental Design

The experiments to compare our simultaneous state and parameter estimation approach to related work are conducted in three steps in order to work out the principal characteristics of the individual parts of our approach and their cooperation. The experimental setup for all of these experiments is deliberately simple to focus on the properties of the methods and described in section 4.2. First, the state estimation capabilities of different model-free state estimators are examined and compared to a EKF with perfect prior knowledge as a state of the art standard model-based method. This yields the filter type which will be used by our approach here. Second, state predictors are investigated concerning their capability to accurately represent the system dynamics. Last, the proposed approach is compared to related work. The only related work approach allowing a fair comparison to our approach here is the *dual estimation* approach. Although the *joint estimation* is commonly used in recent applications, it requires prior knowledge about the system for model simplifications or is outperformed by *dual estimation* as shown in [17]. The separation approach in [19] drops out because of prior knowledge about the system, too, as discounting a finite number of system models at runtime is far more easy than building a system model by regression. The comparison with the *dual estimation* approach is even more appealing, since it allows a similar initialization and has thus equal prior knowledge demands compared to our approach.

4.2 Experimental Setup

The experiments are performed on a rather simple non-linear dynamic system: a pendulum. The pendulum is also often used as a standard benchmark problem and thus offers comparability to related works. It is simulated using the differential equation (15) within an implementation by the standard numpy ode solver. Following starting conditions are used: $\theta(t=0) = 3.13\,\text{rad}$, $\dot{\theta}(t=0) = 0\,\text{rad/s}$ with a fixed simulation time step size of $0.005\,\text{s}$ and parameters $g = 9.81, l = 0.2$.

$$\ddot{\theta}(t) \;=\; -\frac{g}{l} \cdot \theta(t) \tag{15}$$

In many real world applications only a angular sensor is available to observe a joint angle in a system. For the simulated pendulum this simple

Table 1: Optimized estimation delay and corresponding state estimation performance of the MA filter and SG filters of different degrees and the performance of the EKF state estimator.

Filter	Opt. Delay	RMSE
MA/UFIR	6	0.0739 ± 0.00032
SG degree:3	20	0.0593 ± 0.00011
SG degree:5	31	0.0578 ± 0.00012
SG degree:7	43	0.05622 ± 0.000029
SG degree:9	51	0.0569 ± 0.00021
SG degree:11	54	0.0570 ± 0.00090
EKF	–	0.05507 ± 0.000014

sensor model is applied to obtain realistic measurement data. The measurements of the angular sensor are simulated by disturbing and discretizing the ground truth data. The noise is white with a zero-mean and a magnitude of 0.1% of the amplitude of the angle range. The discretizing raster contains 13056 values, a resolution modern angular sensors can provide.

The measurements for the velocity $\dot{\theta}$ are calculated by computing finite differences of the measurements of the angle θ. This way the quality of angle measurements is much higher than for velocity, thus making the velocity estimation the more interesting and challenging part. In order to keep this section compact, only the results concerning the velocity measurement will be worked on. All following experiments concerning state estimation in the above setup are repeated 1000 times to make the results independent of noise influence.

4.3 State Estimation

The experiments to state estimation compare the model-free state estimators discussed in section 2.1 with each other and to a EKF state estimator with perfect prior knowledge as a reference. The EKF is setup following [17]. The performance of the methods is measured by the Root Mean Squared Error (RMSE) between the state estimates and the ground truth data. The RMSE is calculated on a time interval containing two periods of the simulated pendulum, thus covering every possible system state. Table 1 contains the resulting RMSE performance of the methods for an optimized state estimation delay.

For MA filters the filter length p also determines the delay of the state estimate k as $2k + 1 = p$, so the only design parameter for this filter is its

delay k. As shown in [3], UFIR filters show the best noise reduction for $2k + 1 = p$ and in this case behave like usual MA filters. Thus the two filters are treated as one method in the experimental results.

SG filters behave the same way as UFIR filters concerning the filter length p and delay k, but offer the polynomial approximation degree d as an additional free parameter. The performance of the SG filters shows a minimum for a SG filter of degree 7 which nearly reaches the estimation accuracy of the EKF reference. The optimized estimation delay of the SG filters increases with an increasing polynomial degree, because the more parameters of the polynomial has to be estimated and thus more data is needed. But with an increasing estimation delay and thus filter length, the computational demands increase as well. So there is a trade-off between estimation accuracy and real-time constraints.

Looking at the optimized estimation delay, the performance of all FIR filters generally increases as they are allowed to estimate more past values. For MA filters the dumping of the signal is crucial. So there is a sharp optimum for the delay that also limits the accuracy of their state estimation.

The SG filter is able to widely conserve the signal dynamics and can therefore benefit from an increased filter length. The filter length is limited by a dumping of the signal due to a low degree polynomial approximation that cannot represent the system dynamics within the filter length. But a high degree polynomial approximation of the signal within the filter length may lead to a poor noise reduction. Thus only low degree SG filters allow for a effective noise reduction of the signal. As SG filters are FIR filters, they are BIBO stable and need only prior knowledge about the time scale of the system dynamics. So they meet all of the key requirements this paper focuses on and thus are well suited for the state estimation task within the proposed approach.

4.4 Parameter Estimation

In this section the prediction accuracy for different state predictors is compared for various prediction horizons. The accuracy of the predictor is measured by the minimal RMSE (mRMSE) between its prediction and the corresponding ground truth value over a full period of the pendulum.

As the underlying approximator of the predictor, a local and a global approximation structure are compared. The local approximation structure is a Grid-base Look-up Table (GLT) with linear interpolation and equally

Table 2: The table contains the prediction accuracy for the pendulum velocity $\dot{\theta}$ of local and global approximation structures on different prediction horizons k for optimized parameter values of the approximation structures.

$(\theta, \dot{\theta})_{i-k} \rightarrow \dot{\theta}_i$	#nodes		mRMSE	degree		mRMSE
k	θ	$\dot{\theta}$	GLT	θ	$\dot{\theta}$	Polynomial
1	8	10	$1.35 \cdot 10^{-4}$	11	1	$2.44 \cdot 10^{-8}$
3	8	19	$2.68 \cdot 10^{-4}$	13	1	$9.36 \cdot 10^{-8}$
5	8	19	$3.20 \cdot 10^{-4}$	13	1	$1.56 \cdot 10^{-7}$
10	8	19	$5.05 \cdot 10^{-4}$	15	1	$3.60 \cdot 10^{-7}$
20	15	19	$9.23 \cdot 10^{-4}$	19	1	$8.28 \cdot 10^{-6}$
30	15	20	$1.32 \cdot 10^{-3}$	8	6	$2.76 \cdot 10^{-6}$
40	20	20	$2.40 \cdot 10^{-3}$	11	5	$3.15 \cdot 10^{-5}$

distributed nodes along the input dimensions. Thus the only design parameters are the number of nodes along each input dimension.

The global approximation structure is a polynomial with mixed terms of the input variables. The design parameters here are the degrees of the polynomials in each input variable. If the number of nodes in the GLT along each input dimension is equal to the polynomial degree plus one of the corresponding input dimension in the polynomial, the two structures have the same total number of parameters and thus theoretically the same expressiveness, which is an appealing feature for comparing them. Table 2 shows the mRMSE results for the two approximation structures on different prediction horizons and optimized design parameters of the approximators.

The comparison of the GLT and polynomial approximator shows that in this case the polynomial is a much more accurate approximator. It beats the mRMSE performance of the GLT approximator by at least two orders of magnitude and needs on the same time a less total number of parameters for the approximation. The different prediction horizons show that the prediction gets harder as the prediction horizon increases. While the overall accuracy for the polynomial approximator is alway better than for the GLT approximator, the relative degradation of the accuracy with an increasing prediction horizon is better for the GLT approximator. Thus the GLT allows for a more easy and robust engineering with acceptable performance and the polynomial allows for a more accurate approximation with less computational afford due to the reduced total number of parameters.

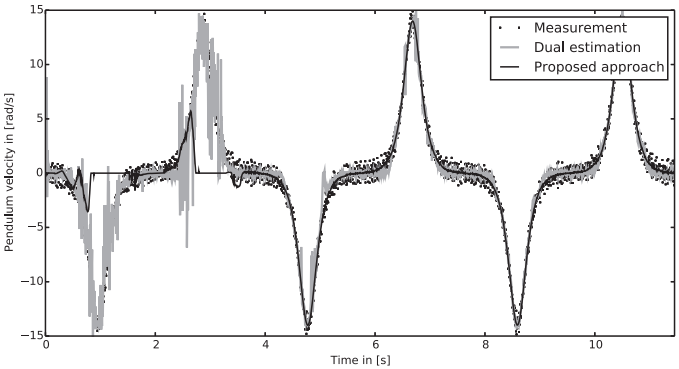


Figure 2: The state estimation for the velocity of the pendulum in a simultaneous state and parameter estimation task for the *dual estimation* and the proposed approach. The parameter estimators used in the compared approaches work on a GLT approximator.

4.5 Simultaneous State and Parameter Estimation

The experiments to compare the two approaches to the SSPEP are performed with a local and global approximation structure for the parameter estimator. In order to achieve high comparability of the approaches, the approximators they use share the same number of parameters and initialization. For the local approximation structure, a GLT with 15 nodes along each input dimension is used. The global approximation structure is a polynomial of degree 8 for each input dimension. All parameters of the approximators are initialized with $\alpha = \mathbf{0}$ and both approaches use the RLS on-line learning algorithm.

The *dual estimation* approach uses the same noise characteristics for the state estimating EKF as the EKF in section 4.3, thus assuming prior knowledge about the noise. The use of a SG filter of degree 7 with delay $k = 43$ would preform the best state estimate as shown on Table 1, but here the prediction accuracy has to be considered, too. So the state estimator for the proposed approach is a SG filter with delay $k = 30$ and degree 5 as a compromise between estimation and prediction accuracy.

The results for one run of the experiments with the GLT approximator are shown in Figure 2 over time. The figure shows the measurements and the estimations of the two approaches for the velocity of the pendulum over three periods right from the start. The *dual estimation* follows the

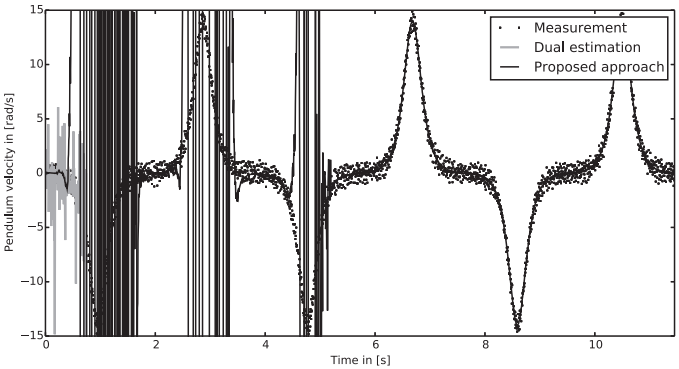


Figure 3: The state estimation for the velocity of the pendulum in a simultaneous state and parameter estimation task for the *dual estimation* and the proposed approach. The parameter estimators used in the compared approaches work on a polynomial approximator.

measurements right from the start, but amplifies the noise during the first period. The estimation becomes much more smooth during the second period, but still shows some peaks. Over the third period this approach shows a good noise reduction without peaks. So the overall estimation quality slowly increases over time.

The proposed approach seems to ignore the measurements during the first period and mainly the $\alpha = \mathbf{0}$ initialization drives the estimation. During the second and third period the estimation smoothly follows the measurements and shows a good noise reduction. So after a poor estimation while the predictor learns the system dynamics in the beginning, the approach nearly instantaneously is able to estimate the current state of the system very well.

Comparing this results with the behavior of the two approaches on a polynomial approximator as shown in Figure 3. The *dual estimation* approach shows a completely different behavior as it gets unstable during the first period and never gets back. The proposed approach remains stable, but shows a random output with heavy peaks during the first and second period that only follows the measurements for low velocities. In the third period, the approach shows a smooth estimation of the velocity without peaks. So the polynomial approximator needs more time and thus learning data to be able to represent the system dynamics.

On the long run the estimation accuracy of the *dual estimation* approach

Table 3: The table contains the prediction accuracy for the pendulum velocity $\dot{\theta}$ of the *dual estimation* and the proposed approach with GLT and polynomial approximators.

GLT	*dual estimation*	Porposed
1. Period	1.81 ± 0.040	5.188 ± 0.0027
2. Period	0.74 ± 0.042	0.068 ± 0.0011
3. Period	0.408 ± 0.0026	$0.0632 \pm 6.2 \cdot 10^{-4}$
4. Period	0.31 ± 0.015	$0.06238 \pm 2.2 \cdot 10^{-5}$
Polynomial	*dual estimation*	Porposed
1. Period	NaN	$11.2e8 \pm 8.9 \cdot 10^{8}$
2. Period	NaN	$9.0e3 \pm 8.9 \cdot 10^{3}$
3. Period	NaN	0.064 ± 0.0014
4. Period	NaN	$0.062325 \pm 9.7 \cdot 10^{-6}$

is limited by the prediction accuracy of the parameter estimator and thus nearly equals performance of the EKF in Table 1. The long run performance of the proposed approach is on the one hand limited by the estimation accuracy of the SG filter and the prediction accuracy of the predictor and thus beaten by the *dual estimation* approach.

These results are supported by the RMSE measurements over the single periods for both experiments that are shown in Table 3. For the GLT approximtor, the RMSE of the *dual estimation* approach starts with a moderate value and then slowly decreases. The RMSE of the proposed approach in the first period is very high, but drops below the RMSE of the *dual estimation* approach right in the second period and nearly reaches the optimal performance of the SG filter shown in Table 1. In the third, forth and following periods the RMSE nearly stays the same. For the polynomial approximator the drop of the RMSE happens in the third period and then again nearly stays the same in the forth and following periods.

The comparison of the two approximation structures shows that the Polynomial causes the *dual estimation* approach to become unstable, due to harmful interaction and poor state predictions during the initialization of the approximator. For the GLT approximator, the comparison of the two approaches shows that avoiding a inner feedback loop and thus possibly harmful interaction between the state and parameter estimator results in a faster total convergence of the estimation. The state estimation of our approach is very poor during the first learning phase that lasts the first period, i.e. the state space is run through for the first time. But once an already known situation appears the estimation accuracy drastically increases. The *dual estimation* approach offers a much better estimation during the first

period, but converges only slowly and shows peak errors even during the second and third period. On the long run, both approaches behave equally well. The estimation performance of our approach is limited by the FIR filter for state estimation and the predictor accuracy and the *dual estimation* approach only is limited by the approximation structure and prior knowledge about the noise.

5 Discussion

The experimental results show that our approach provides comparable estimation accuracy to the *dual estimation* approach and thus is able to deal with the simultaneous state and parameter estimation problem. Since any feedback interaction between the two estimators is discarded, our approach converges more quickly than the *dual estimation* approach and is always stable independent of the used approximation structure. Thus the engineering of the whole estimation system becomes easier as every single part of it can be designed without paying attention to the rest of the components and only concentrating on the application performance. The only dependence between the state and parameter estimator that has to be considered is the estimation delay of the state estimator, determining the prediction horizon. What further makes the engineering difficult are the principal limitations shown in section 4.3 and 4.4. While the state estimation of delayed estimates becomes more accurate with increasing the delay, the prediction accuracy decreases with an increasing prediction horizon. Thus a compromise between accurate state estimation and manageable state prediction is needed.

The experiments also showed that the proposed approach is only able to perform an acceptable state estimation in regions the used predictor already had some learning data before. This results in a crucial initialization phase with degraded estimation performance. Nevertheless the estimation remains stable and allows the performance to rise on the long run.

The computational demands of the proposed approach are low and fix and therefore it is real-time applicable. So our approach meets all of the key requirements and produces accurate delay-free state estimates without detailed prior knowledge about the system under observation and an easy engineering for the state estimation and predictor.

6 Conclusion

This paper presented an approach to the simultaneous state and parameter estimation problem that offers the same estimation quality as state of the art methods, but avoids instability issues. The approach follows a estimator-predictor scheme without feedback interaction between the estimator and predictor, thus making harmful interaction impossible and allowing for a stable estimation. The proposed approach needs no detailed prior knowledge about the system parameters and is real-time capable. The only general prior knowledge that is required is a estimation of the time scale of the system dynamics and the complexity of the dynamics of the system in oder to define the estimation delay of the state estimator and the expressiveness of the predictor.

References

[1] Gardner Jr, E. S.: Exponential smoothing: The state of the art—Part II. *International Journal of Forecasting* 22 (2006) 4, S. 637–666.

[2] Wei, W. W.-S.: *Time series analysis*. Addison-Wesley Publ. 1994.

[3] Shmaliy, Y. S.; Simon, D.: Iterative unbiased FIR state estimation: a review of algorithms. *EURASIP Journal on Advances in Signal Processing* 2013 (2013) 1, S. 1–16.

[4] Savitzky, A.; Golay, M. J.: Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry* 36 (1964) 8, S. 1627–1639.

[5] Kalman, R. E.: A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering* 82 (1960) 1, S. 35–45.

[6] Wan, E. A.; Van Der Merwe, R.: The unscented Kalman filter for nonlinear estimation. In: *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, S. 153–158. IEEE. 2000.

[7] Simon, D.: *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons. 2006.

[8] Kim, P.-S.: An alternative FIR filter for state estimation in discrete-time systems. *Digital Signal Processing* 20 (2010) 3, S. 935–943.

[9] Shmaliy, Y. S.: Linear optimal FIR estimation of discrete time-invariant state-space models. *IEEE Transactions on Signal Processing,* 58 (2010) 6, S. 3086–3096.

[10] Farrell, J. A.; Polycarpou, M. M.: *Adaptive approximation based control: Unifying neural, fuzzy and traditional adaptive approximation approaches*, Bd. 48. John Wiley & Sons. 2006.

[11] Ying, Y.; Pontil, M.: Online gradient descent learning algorithms. *Foundations of Computational Mathematics* 8 (2008) 5, S. 561–596.

[12] Crammer, K.; Dekel, O.; Keshet, J.; Shalev-Shwartz, S.; Singer, Y.: Online passive-aggressive algorithms. *The Journal of Machine Learning Research* 7 (2006), S. 551–585.

[13] Buschermohle, A.; Schoenke, J.; Rosemann, N.; Brockmann, W.: The Incremental Risk Functional: Basics of a Novel Incremental Learning Approach. In: *IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2013*, S. 1500–1505. IEEE. 2013.

[14] Hunt, K. J.: A survey of recursive identification algorithms. *Transactions of the Institute of Measurement and Control* 8 (1986) 5, S. 273–278.

[15] Crammer, K.; Lee, D. D.: Learning via gaussian herding. In: *Advances in Neural Information Processing Systems*, S. 451–459. 2010.

[16] Cesa-Bianchi, N.; Conconi, A.; Gentile, C.: A second-order perceptron algorithm. *SIAM Journal on Computing* 34 (2005) 3, S. 640–668.

[17] Haykin, S. S.; Haykin, S. S.; Haykin, S. S.: *Kalman filtering and neural networks*. Wiley Online Library. 2001.

[18] Liu, L.; Wang, L. Y.; Chen, Z.; Wang, C.; Lin, F.; Wang, H.: Integrated system identification and state-of-charge estimation of battery systems. *IEEE Transactions on Energy Conversion* 28 (2013) 1, S. 12–23.

[19] Chong, M. S.; Nešić, D.; Postoyan, R.; Kuhlmann, L.: Parameter and state estimation of nonlinear systems using a multi-observer under the supervisory framework. *arXiv preprint arXiv:1403.4647* (2014).

Safe Global Inter- and Extrapolation Using Local A Priori Knowledge

Jonas Schneider, Werner Brockmann

University of Osnabrück
Albrechtstraße 28, 49069 Osnabrück
Tel.: (0541) 969 2439
Fax: (0541) 969 2799
E-Mail: {joschnei, Werner.Brockmann}@Uni-Osnabrueck.de

Abstract Designers of adaptive systems for regression often have different kinds of a priori knowledge about the function that is to be learned. By using that knowledge, the learning process can be led to faster and better convergence and increased robustness. Those designers deal with the problem of how to effectively use that knowledge in the formulation of the learning problem. This paper deals with the formalization of using different kinds of local a priori knowledge for a specific form of local regularization of the learning problem. The GIESELA approach presented here enables the designer to utilize his local *absolute*, *relative* and *qualitative* a priori knowledge about the function to be learned in a robust and interpretable way and to apply this knowledge on a global scale. The approach is mathematically analyzed and demonstrated on a simple example.

1 Introduction

Nowadays on-line learning systems are more and more employed in even safety-critical application scenarios. Such on-line learning systems are required to enhance their functional behavior over time as more and more training data are incorporated. Due to the criticality of the application, such systems must guarantee a safe and robust behavior even when data are sparse, i.e. also already at the beginning of the learning process. But as the training data provide only local learning stimuli, sparsity of the training data may become a problem at any time in the learning process.

This contrasts the need to express different properties of the function in different regions of the input space, e.g. when a process enters a new state. It is therefore desirable to perform regional adaptations instead of local adaptations of the approximation based on a training datum and the expected regional properties of the function, i.e. to perform some kind of

limited extrapolation. This requires the designer to incorporate a priori knowledge of the underlying system to identify such regions and the corresponding functional properties. It also requires a method to utilize that knowledge during the learning process to accelerate the convergence of the approximator while still guaranteeing the desired overall properties of the approximator.

In this work the designer's a priori knowledge is divided into three categories:

absolute knowledge:	a priori knowledge about absolute local output values
relative knowledge:	a priori knowledge about relations between local output values
qualitative knowledge:	a priori knowledge about qualitative properties in certain regions of the output function, e.g. monotony or smoothness

Using this knowledge for accelerating the convergence of the approximator and guaranteeing a certain behavior presents certain challenges considering the safety of the learning behavior and the learned approximation. Hence the following three requirements arise:

- *robustness* of the learning process
- *a priori controllability* of the learning process
- *interpretability* of the (learned) knowledge

First, a method for incorporating a priori knowledge into the learning process must be *robust*. That means that small errors in the training data, i.e. noise, and also small errors in the a priori knowledge must not lead to an arbitrarily big error in the adaptation of the learning system because the designer's a priori knowledge can't be expected to be exact. The representation of that knowledge must be *interpretable* to enable the designer to convert his a priori knowledge into a form that can be used by the learning process at runtime. This means that the knowledge about the application can easily be converted into that representation. Vice-versa learned knowledge must be interpretable considering the underlying application in this representation.

A priori knowledge often arises from experience the designer has gathered about the application. Exact knowledge would require a total understanding of the application and all influences and thus the formulation of exact knowledge is not feasible. *A priori controllability* means that the designer is able to guide the complete learning process right from the beginning. If for example the approximator has to be bounded in a region of the input space, the designer must be able to express that to ensure a safe adaptation of the learning system and thus a safe operation of the application. This is necessary to enable the designer to formulate the a priori knowledge and understand how it is used by the learning process.

In this paper an approach to utilize the different kinds of a priori knowledge in local function approximators for 0-Order Takagi-Sugeno-Fuzzy-Systems (TS-0) is presented and investigated after the state of the art is discussed.

2 State of the Art

2.1 Learning Methods

Different works have already dealt with the problem of regression using a priori knowledge. Early works have concentrated on the case of off-line batch training, e.g. different extensions of Shepard's Inverse Distance Weighting [1]. Those works incorporate the a priori knowledge explicitly at the points of the training data [2]. These approaches lack the possibility to incorporate a priori knowledge at arbitrary locations in the input space but are reduced to the incorporation at positions where training data lie. Other methods focused on rectangle-based blending regression by k-nearest-neighbor interpolation [3]. Those methods have the undesirable behavior to produce discontinuities in the input space where the neighboring relationships change. Both types of methods require all training data to be known beforehand and can not easily be adapted when new training data are available as in an on-line scenario rendering these methods unsuitable for the application case discussed in this paper.
To easily incorporate new training data into the approximator on-line different learning rules for approximators with fixed parameter vector are known. These can be divided into first order algorithms such as Passive-Aggressive-Learning [4] and second order algorithms such as Recursive Least Squares [5][6] or the more complex Gaussian Herding [7]. A priori knowledge can be directly implemented into these algorithms by incorporating it directly into the learner's error function [8]. This requires the

designer to state an error function for each kind of a priori knowledge, i.e. absolute, relative and qualitative a priori knowledge, and each learner. He then has to determine the resulting error function. This is neither easily done and interpretable nor controllable since the changes of the learning process result indirectly from the minimized function and not from the a priori knowledge directly. Due to these problems, this paper focuses on algorithms that incorporate the designer's a priori knowledge independently from the chosen learning rule. A general framework to guide the designer in stating the error function is given by Learning From Hints and is discussed in the next section.

2.2 Learning From Hints

Learning From Hints [9] allows the designer to utilize his a priori knowledge during the learning process by transferring it into so called *hints*. A hint is any information available about the function, i.e. also absolute, relative or qualitative a priori knowledge, but also the training data themselves. For all available a priori knowledge and further information the designer formulates hints. These hints represent a subspace of the hypothesis space of the approximator in which all functions fulfilling the respective hint lie. A hint is formulated by the expected value of the error function measuring the fulfillment of the hint, e.g. $(g(x) - g(x'))^2$ the function measuring the fulfillment of an invariance hint, with g the function to be measured and input value x and $x' = x + a$ with invariance offset a. Taking the expected value of these error functions then measures the approximator's hypothesis' overall fulfillment of the given hint. The learning algorithm then tries to minimize the function given by the sum of all those hints' fulfillment measurement functions. Since the resulting error functional is a complex function due to a potentially large number of hints, local gradient methods are prone to converging to local minima. The incorporation of new training data also constantly changes the error function rendering an off-line precalculation of the function impossible. While absolute and relative a priori knowledge can be relatively easy formulated as hints, the formulation of qualitative a priori knowledge in terms of hints is difficult due to the need to define a continuous error function for properties that are measured binarily. Not only does every training datum represent a new hint, old hints based on the a priori knowledge might be changed, too. Hence the training process is neither robust nor a priori controllable. Another method for the incorporation of a priori knowledge is *regularization* which separates the learning based on the training data from the learning based on the a priori knowledge. This method is discussed in the following section.

2.3 Regularization

2.3.1 Global Regularization

Global regularization methods extend the error functional that the learner minimizes by an additive penalty term. A standard regularization approach is the *Tikhonov regularization* [10]. The Tikhonov regularization adds a penalty term consisting of a multiplication of a *Tikhonov matrix* Γ to the approximator's parameter vector ω. Let ω be the approximator's parameter vector, $\phi(x)$ the mapping function from input space to parameter space, f the approximator's hypothesis and (x_i, y_i) a training datum. In case of a simple Passive-Aggressive-Learner [4] this yields the following functional that is minimized by the learner:

$$\text{argmin}_{\omega_{t+1}} \underbrace{||\omega_t - \omega_{t+1}||^2}_{\text{Passive}} + \underbrace{||f(\phi(x_i), \omega_{t+1}) - y_i||^2}_{\text{Aggressive}} + \underbrace{||\Gamma\omega_{t+1}||}_{\text{Regularization}} \quad (1)$$

Usually, the matrix Γ is chosen as the identity matrix thus penalizing parameter vectors with greater norms [11]. The non-diagonal elements of the matrix Γ can be used to express desired dependencies between different parameters of the approximator. These non-diagonal entries of Γ can therefore be used to describe dependencies between arbitrary approximator parameters. Although this allows the incorporation of arbitrary relative a priori knowledge, the design of the matrix Γ is a difficult task if one wants to take all global dependencies into account because the design of the a priori knowledge takes place in the parameter space of the approximator and not the input space. Hence this approach does not offer an interpretable formulation of the a priori knowledge.

2.3.2 Local Regularization

Local regularization for TS-0 Fuzzy Systems with a normalized rule base has been introduced by the SILKE approach [12] and provides a method to design and apply local relative and qualitative a priori knowledge in local function approximators. Each rule is provided with a *template* that represents the desired relationships to the neighboring rules by interrelating the conclusion of a rule as a linear combination of the rule conclusion and the conclusions of the neighboring rules so that a template can be seen as a locally applied convolution matrix for each rule, as figure 1 exemplarily shows for a two dimensional grid-based local function approximator. The local a priori knowledge is incorporated by the choice of these local

templates. After each learning step the regularization then modifies every parameter, i.e. every rule conclusion, according to the adaptation by the learning rule and the parameter value determined by the local convolution with the template. The update ratio is given by a local adjustment rate $\alpha_i \in [0; 1]$.

Typically, the same template for each rule is chosen, e.g. an averaging template to obtain a smoother curvature of the resulting approximation. This allows the easy incorporation of relative a priori knowledge between neighboring nodes. Absolute a priori knowledge on the other hand cannot be expressed by those templates. Due to the ease of the formulation of the a priori knowledge as templates, this concept fulfills the requirement of interpretability. Nevertheless, the approach does not fulfill the requirement of a priori controllability. As long as the local adjustment rate α_i is not 1, i.e. the system incorporates the knowledge gained through the training data into the approximator, it is not guaranteed that the approximation matches the desired a priori knowledge. It may thus not be suitable for use in safety critical applications.

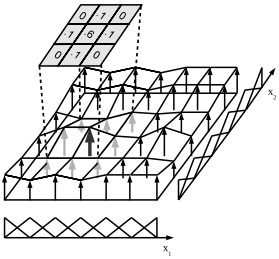


Figure 1: Schematic depiction of a SILKE template on a TS-0 approximator.

3 Concept for Controlled Inter- and Extrapolation

3.1 General Concept

TS-0 fuzzy systems have been shown to be universal local function approximators which are interpretable and easy to design thus meeting the requirements stated in chapter 1. To get a total localization of each rule of the system, linear triangular membership functions on a normalized rule base are chosen. To circumvent the need of minimizing a risk functional for each learning algorithm, the presented approach is based on regularization. Since local regularization already provides a framework for the formulation of local relationships between rule conclusions, this will also be used here. It is complemented by the *Anchoring* approach which is introduced because local regularization does not support the incorporation of absolute a priori knowledge.

In incremental learning a single training datum leads to the adaptation of the conclusions of the actively firing rules. After that, the regularization then alters only those active rule conclusions thus yielding only them to be consistent with the a priori knowledge, i.e. their templates. On the other hand, it does specifically not realize the consistency of the neighboring rule conclusions and their templates. To achieve this more global consistency, a local regularization step has to be applied on the neighboring rules as well as on their neighbors and so forth. This allows interpolation between rules or extrapolation of the knowledge gained by one training datum to more distant rules, especially in case of sparse data.

An efficient one step adaptation of the whole parameter vector is realized by the *GIESELA (Global Inter- and Extrapolation System for Efficient Learning of Approximators)* approach. To be able to distinguish between inter- and extrapolation which need different adaptations based on properties of the training data seen by a rule so far, *trust* parameters are introduced to provide a monitor for the local data density and further local meta-properties of the training data.

3.2 The Anchoring Approach

Anchoring means the gradual exclusion of certain approximator parameters from the adaptation by the learning rule. The difference to simply removing the parameter from the approximator is that by anchoring the respective rule will still be used by the learning rule and regularization. This way absolute a priori knowledge is also spread to neighboring rules. Therefore, each rule is given a pair of parameters $(\sigma_i^A, \aleph_i)$ with $\sigma_i^A \in [0;1]$ the local *stiffness* and $\aleph_i \in \mathbb{R}$ the local *anchor value*. Let $\omega^{t+1} = \omega^t + \Delta\omega^t$ be the parameter vector at time step $t+1$ and $\Delta\omega$ the desired difference between the old and new parameter vectors determined by the learning rule. Then for each adapted parameter ω_i^{t+1} the new value is determined by:

$$\omega_i^{t+1} = (1 - \sigma_i^A) \cdot (\omega_i^t + \Delta\omega_i^t) + \sigma_i^A \cdot \aleph_i \tag{2}$$

A stiffness of 0 means to ignore the anchor value for that parameter and to let the learning rule adapt the parameter normally. A stiffness of 1 means to exclude this parameter from adaptation and to always use the corresponding anchor value as the rule conclusion. Values in between 0 and 1 let the conclusion to be adapted by the learning rule to a certain degree. The anchor value then represents a fixed value the rule conclusion will be softly bonded to when being changed by the learning rule. This approach

alone therefore only has a local impact on the learning system and does not provide any generalization of the absolute a priori knowledge. This is then done by the GIESELA approach which is discussed in the following section.

3.3 The GIESELA Approach

The GIESELA approach is based on a priori knowledge formalized by local templates introduced by the SILKE approach [12]. It uses that knowledge to provide a global regularization of the learning problem. By applying that knowledge globally, it allows extrapolation from the conclusions actually adapted by training data to those parts of the input space where suitable a priori knowledge is available.

Let $\omega \in \mathbb{R}^n$ be the parameter vector of a local function approximator of TS-0 type with ω_i the i-th element of ω and n the number of parameters of the approximator, i.e. the number of rules. Let d be the dimensionality of the input space and $\xi_i \in \mathbb{R}^{3^d}$ the vector containing the conclusions of the neighboring rules of the rule corresponding to the i-th parameter and the conclusion of the i-th rule itself. $\xi_i^{\frac{3^d+1}{2}} = \omega_i$ is the central element of the vector ξ_i. Further let $s_i \in \mathbb{R}^{3^d}$ denote the SILKE template at the i-th rule. Then $S \in \mathbb{R}^{n \times n}$ is the matrix with row vectors containing the templates of each rule. Then $S \cdot \omega$ performs one SILKE adaptation step.

By extending the matrix S to $S \in \mathbb{R}^{(n+1)\times(n+1)}$ with the $(n+1)$st row the unit vector e^{n+1} lying in the $(n+1)$st dimension and the last row vector containing a constant offset for each template, it is possible to perform an affine transformation with a constant part for each rule.

Applying SILKE by multiplying the matrix S to the parameter vector does not lead to convergence after just one step. Convergence is reached after repetitive application of the local regularization. Hence this approach is not suited if global extrapolation based on the a priori knowledge is desired. To propagate a priori knowledge via local templates and local regularization to a rule far away from the position of the training datum, it is necessary to perform local regularizations on each rule lying on the way. Furthermore, after just one step the local regularization will typically not have converged at a rule so repetitive global application is needed to ensure that each conclusion has been adapted according to the training datum and the designer's a priori knowledge.

To overcome this obstacle the GIESELA approach formulates this problem as a linear equation system on the approximator's parameters, i.e. on the

rules defining the output function. *Evaluation templates* as known from the ODIL approach [13] are used for construction of the linear equation system. An evaluation template is a template that is used to measure the conformity of a rule conclusion with a corresponding SILKE template. It is obtained by subtracting 1 from the central element of a SILKE template [14].

A rule conclusion satisfies the designer's a priori knowledge given as a template if and only if the convolution with the corresponding evaluation template equals 0. This allows to formulate the regularization step as a linear equation system $S' \cdot \omega = \psi$ with S' being a matrix consisting of row vectors containing the evaluation templates for each rule and a vector ψ that describes the desired level of compliance of the rule conclusion and the a priori knowledge. If the vector ψ is chosen as the zero vector, the linear equation system will yield values for each rule conclusion that will satisfy all templates as long as exactly one solution for the linear equation system exists.

For example, a simple one-dimensional input space with 5 parameters the following linear equation system has to be solved:

$$\underbrace{\begin{pmatrix} s'_{1,2} & s'_{1,3} & 0 & 0 & 0 & s'_{1,4} \\ s'_{2,1} & s'_{2,2} & s'_{2,3} & 0 & 0 & s'_{2,4} \\ 0 & s'_{3,1} & s'_{3,2} & s'_{3,3} & 0 & s'_{3,4} \\ 0 & 0 & s'_{4,1} & s'_{4,2} & s'_{4,3} & s'_{4,4} \\ 0 & 0 & 0 & s'_{5,1} & s'_{5,2} & s'_{5,4} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}}_{S'} \cdot \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \\ \omega_5 \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \psi_4 \\ \psi_5 \\ 1 \end{pmatrix}}_{\psi} \tag{3}$$

$$s'_{i,j} = \begin{cases} s_{i,j} & \text{if } j \neq 2 \\ s_{i,j} - 1 & \text{else} \end{cases} \tag{4}$$

The desired level of compliance is to be specified by a local adjustment rate. These local adjustment rates $\alpha_i \in [0; 1]$ are equivalent to the local adjustment rates in [14]. In case of $\alpha_i = 0$ no regularization will be applied for that rule and in case of $\alpha_i = 1$ full regularization will be applied.

Thus combining local adjustment rates and soft anchoring into one step,

the matrix S' becomes:

$$S' = \begin{pmatrix} f_{\alpha_1,\sigma_1^A}(S'_1) \\ f_{\alpha_2,\sigma_2^A}(S'_2) \\ \vdots \\ f_{\alpha_n,\sigma_n^A}(S'_n) \\ e^{n+1} \end{pmatrix} \tag{5}$$

$$f_{\alpha_i,\sigma_i^A}(S'_i) = \begin{cases} (1-\sigma_i^A)\cdot\alpha_i\cdot S'_{i,j} & \text{if } j \neq \frac{3^d+1}{2} \\ (1-\sigma_i^A)\cdot(\alpha_i\cdot S'_{i,j}+(1-\alpha_i))+\sigma_i^A & \text{if } j = \frac{3^d+1}{2} \end{cases} \tag{6}$$

The vector ψ becomes:

$$\psi = \begin{pmatrix} (1-\sigma_1^A)(1-\alpha_1)\cdot\omega_1 \\ (1-\sigma_2^A)(1-\alpha_2)\cdot\omega_2 \\ \vdots \\ (1-\sigma_n^A)(1-\alpha_n)\cdot\omega_n \\ 1 \end{pmatrix} \tag{7}$$

Evaluation templates are also used to allow the incorporation of qualitative a priori knowledge if the fulfillment of the knowledge is binary. Templates that are used to induce certain properties such as monotony of the approximation are activated if those properties are violated. The designer therefore has to construct a conditional template that measures the contradiction of the approximation to the desired property and on which the activation of the property inducing template is based. These templates must yield a negative result when applied if the desired property is not met. In that case the adjustment template is activated. In case of positive monotony the designer designs a condition template that yields a negative result if the hypothesis does not fulfill that positive monotony. In that case a template is applied on the conclusion that enforces the desired property, e.g. by enforcing a constant function shape.

3.4 Incorporating the Learning History

The concept for the incorporation of a priori knowledge into the learning process introduced in section 3.3 requires the designer to choose local adjustment rates α_i for each rule in the input space. The optimal choice for the local adjustment rates depends on the learning history and thus on various factors like:

- data sparsity
- data conflict
- quality of the a priori knowledge

The less data are in a region of the input space, the less those data can be trusted to properly represent the underlying input-output-relationship. The same applies to conflicting data. The more conflicted the training data are, the less trustworthy is an approximation based on them. In the cases when the training data are not trustworthy it is desirable to follow the designer's a priori knowledge more than the training data. Typically the designer is not able to predict the sparsity and the conflict of the training data beforehand. Furthermore, in on-line scenarios data sparsity and data conflict are no constants but change over the course of the learning process. Thus it is desirable to adapt the local adjustment rates dynamically to those changing properties of the training data. However, the designer is usually unable to predict the dynamic development of the data sparsity and conflict beforehand, rendering a fixed a priori design of data-dependent adjustment rates for each rule impossible.

But vice versa, the optimal adjustment rates are also determined by the quality of the a priori knowledge. The more the a priori knowledge differs from the actual input-output-relationship, the less that knowledge should be followed thus the lower the adjustment rates should be chosen. While it might be possible for the designer to specify a certain trustworthiness for his a priori knowledge, the actual error of the a priori knowledge can't be known beforehand. Did the designer know what that error was, he would have been able to specify error-free templates.

Since all three kinds of uncertainties are hard or impossible to measure beforehand, it is desirable that the local adjustment rates of each rule automatically adjust locally according to the training data seen in the respective regions of the input space. In this paper *trust* is used to replace the user-defined local adjustment rates by *trust values* $\vartheta_i \in [0; 1]$ for each rule. A trust value ϑ_i measures the trustworthiness of a parameter ω_i with a trust value of 0 meaning the parameter can not be trusted at all and a trust value of 1 meaning the parameter can fully be trusted and treated normally.

The trust values are to measure the trustworthiness of a parameter based on the sparsity and conflict of the training data that lay in the region of the

input space where the respective rule is active according to:

$$\vartheta = \min\left(\vartheta^D, \frac{\vartheta^I + \vartheta^C}{2}\right) \tag{8}$$

This formula combines the *ignorance trust* ϑ^I, *conflict trust* ϑ^C and *direct trust* ϑ^D. The ignorance trust ϑ^I provides a measure for the sparsity of the training data while the conflict trust ϑ^C provides a measure for the conflict of the training data. The direct trust ϑ^D rates the trustworthiness of the adaptation of the current training datum to filter those data that do not match the previously seen data.

The different trust values are defined as follows:

$$\vartheta^I_{T+1} = \frac{\eta_I \sum_{t=0}^{T} \hat{\phi}(x_t)}{1 + \eta_I \sum_{t=0}^{T} \hat{\phi}(x_t)} \tag{9}$$

$$\vartheta^C_{T+1} = \frac{1}{1 + \eta_C \frac{1}{\sum_{t=0}^{T} \hat{\phi}(x_t)} \sum_{t=0}^{T} |\Delta\omega(x_t, y_t)|} \tag{10}$$

$$\vartheta^D_{T+1} = \frac{1}{1 + \eta_C |\Delta\omega(x_t, y_t)|} \cdot \left(1 - \frac{1 - \vartheta^D_T}{1 + \eta_I \hat{\phi}(x_T)}\right) \tag{11}$$

The function $\hat{\phi}(x_t)$ denotes the normalized activation of a rule for a training datum (x_t, y_t) and $\Delta\omega$ the adaptation of the parameter at the last adaptation step. It has been shown in [15] that the following heuristics for the choice of the parameters η_I and η_C perform well:

$$\eta_I = \frac{2}{\tau} \tag{12}$$

$$\eta_C = \frac{1}{\sigma^2 \cdot \sqrt{\frac{8}{\pi}}} \tag{13}$$

with the half-life period τ specifying the needed accumulated activation for each rule after which the ignorance trust should rise to 0.5 and the standard deviation σ specifying the standard deviation at which the conflict trust should rise to 0.5.

Combining all parts of the approach yields the following sequence when a new training datum is incorporated:

1. Determine the learning rules change $\Delta\omega_t$ of the parameter vector
2. Blend between the anchor value $\aleph$ and the learning rule's change $\omega_t + \Delta\omega_t$

3. Update local trust values $\vartheta_i^I, \vartheta_i^C, \vartheta_i^D$
4. Update local adjustment rates α_i based on the local trust ϑ_i
5. Regularize by GIESELA to obtain the updated parameter vector ω_{t+1}

4 Formal Analysis

4.1 Relationship to Tikhonov Regularization

Although the a priori knowledge is locally described, GIESELA provides a mechanism for a one-step adaptation of the whole approximator's parameter vector, i.e. it acts globally. Thus it shares similarities to the Tikhonov regularization which's matrix Γ is usually hard to design. The connection between GIESELA and the Tikhonov regularization can be seen when minimizing the Tikhonov functional given in equation (1) as shown in the following:

$$0 = \frac{\mathrm{d}}{\mathrm{d}\omega} \|\omega - \omega_t\| + \|\phi(x_t)\omega - y_t\| + \|\Gamma\omega\| \tag{14}$$

$$0 = \frac{\mathrm{d}}{\mathrm{d}\omega} \frac{1}{2}\|\omega - \omega_t\| + \frac{1}{2}\|\phi(x_t)\omega - y_t\| + \frac{1}{2}\|\Gamma\omega\| \tag{15}$$

$$0 = (\omega - \omega_t) + x_t(\hat{y} - y) + \Gamma\omega \tag{16}$$

$$-\omega - \Gamma\omega = -\omega_t + x_t(\hat{y} - y) \tag{17}$$

$$(-\mathbf{1} - \Gamma)\omega = -\omega_t + x_t(\hat{y} - y) \tag{18}$$

$$(\mathbf{1} + \Gamma)\omega = \omega_t - x_t(\hat{y} - y) \tag{19}$$

$$\omega = (\mathbf{1} + \Gamma)^{-1}(\omega_t - x_t(\hat{y} - y)) \tag{20}$$

$$\tag{21}$$

While at the same time a GIESELA step can be formulated as:

$$S' \cdot \omega = \omega_t - x_t(\hat{y} - y) \tag{22}$$

$$\omega = S'^{-1}(\omega_t - x_t(\hat{y} - y)) \tag{23}$$

Thus yielding:

$$\Gamma = S' - \mathbf{1} \tag{24}$$

So in fact the matrix S can be seen as the Tikhonov representation of the GIESELA matrix S' thus yielding three equivalent global regularization problems given by S, S' and Γ as long as the matrix S' is non-singular.

This now gives us a concrete rule for an easier and interpretable construction of a Tikhonov matrix Γ that performs the same adaptation of the approximator as the GIESELA approach itself but with certain well defined local characteristics.

4.2 Robustness

The robustness of the GIESELA approach can be measured by the condition number of the linear equation system defined by the GIESELA matrix S'. The condition number is a standard tool of linear algebra to determine the error propagation in linear equation systems. This allows the designer to know a priori how strongly errors in his knowledge will affect the outcome of the regularization step. The condition number $\kappa_{S'} \in \mathbb{R} > 1$ of the matrix S' is a measure for the amplification factor of the errors in the matrix S' and the vector ψ on the resulting parameter vector ω. The condition number $\kappa_{S'}$ is defined as

$$\kappa_{S'} = \|S'\| \cdot \|S'^{-1}\| \tag{25}$$

Due to the scaling invariance of linear equation system, there is an infinite number of templates representing the same a priori knowledge. However, some lead to a faster or slower convergence speed. The designer can use the condition number of the resulting linear equation system as a measure to decide between a fast convergence speed and a robust convergence with small error amplification based on how trustworthy he thinks his a priori knowledge is.

4.3 Stability

It is shown in [14] that the local regularization by the SILKE approach is stable for suitable templates, i.e. convergence under repetitive application of the SILKE convolution is guaranteed. Due to the equivalence of S' and the original local SILKE matrix S, the stability analysis from [14] also applies here. It is known that the application of SILKE converges if the spectral radius of the matrix S is under 1. In case the spectral radius is greater than 1 the regularization is not stable. In case the spectral radius is 1, which can be designed in case of most templates, and the assumption holds that all template entries are from the interval $[0; 1]$, the adaptation can be bounded by the infinity norm of the matrix S' due to $|\lambda| \leq ||S'||$.

$$\omega_{t+1} \leq \max_i(|\omega_t^i|) \cdot \mathbf{1} \tag{26}$$

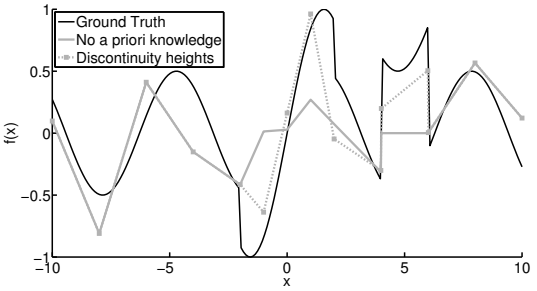


Figure 2: The function $f(x)$ and two approximator's hypothesis after 20 learning steps. One without any a priori knowledge and one with a priori knowledge about the discontinuity heights.

Thus the case of a spectral radius of 1 is bounded as well by the adjusted parameter vector consisting only of the largest element of the parameter vector before adjustment. Applying GIESELA is thus stable under the conditions given above.

5 Demonstrative Example

For a demonstration a simple one-dimensional problem is used. The following function f depicted in figure 2 as the ground truth is to be learned on the interval $[-10; 10]$:

$$f(x) = \begin{cases} \sin(x) & \text{if } x \in [-10; -2) \bigcup (2; 4) \bigcup (6; 10] \\ 2\sin(x) & \text{if } x \in [-2; 2] \\ \sin(x) + 1 & \text{if } x \in [4; 6] \end{cases} \tag{27}$$

This function combines different local monotonic behaviors and discontinuities. The performance metric chosen is the ground truth loss, i.e. the mean squared error of the approximator against the function f in each learning step. Choosing a measure for the deviation from the underlying function rather than the deviance from the training data makes sense here since the designer's a priori knowledge is knowledge about the function itself. The approximator is a TS-0 fuzzy system with normalized triangular membership functions at [-10; -8; -6; -4; -2; -1; 0; 1; 2; 3.99; 4.01; 5.99; 6.01; 8; 10] to allow an increased expressiveness at the points of discontinuity. The parameter vector is initialized as $\omega^{t=0} = \mathbf{0}$. The learning rule is a Passive-Aggressive-Learner with fixed adaptation rate $\lambda = 0.4$. 150

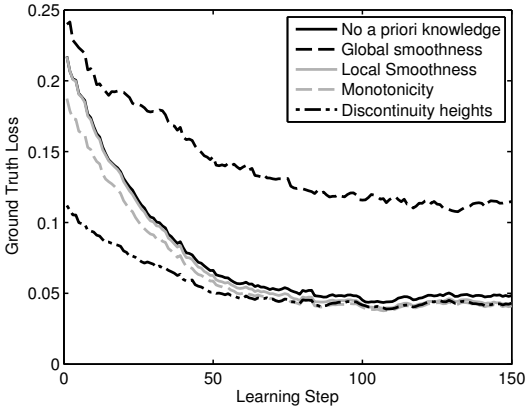


Figure 3: Comparison of the ground truth loss between different levels of a priori knowledge used when learning the function $f(x)$

samples are drawn independently and uniformly distributed over the input space with an uniformly distributed noise between 0% and 30%. Trust is used to automatically adapt the local adjustment rates according to section 3.4 and parametrized as $\eta_I = 1$ and $\eta_C = 0.4$. Figure 3 shows the ground truth loss for the incorporation of different levels of a priori knowledge. The black solid curve shows the loss when no regularization is applied at all and acts as the reference curve. The typical behavior of the ground truth loss can be seen. At the start of the learning process big adaptation steps quickly lead to a much better approximation thus reducing the loss quickly. Over the course of the learning the adaptation steps become smaller and the ground truth error converges finally at approximately 0.05.

The black dashed curve shows the effect of the ordinary SILKE approach that tries to smoothen the function during the learning process based only on local a priori knowledge and it's local application. Due to the discontinuities of the function, this leads to bad approximation results at the points of discontinuity which thus leads to a much larger remaining ground truth error of approximately 0.13. This is an example of how wrong a priori knowledge can actually hamper the learning. If the system had been given more time leading to larger trust values reducing the influence of GIESELA on the conclusions, the error would also have converged to 0.05 like in the case of no application of GIESELA at all. Still the convergence time has severely been worsened.

The solid gray curve shows the ground truth loss for the case of smoothing templates, i.e. relative a priori knowledge, taking the discontinuity points into account. At these points no smoothening is applied. It can be seen that the overall performance of the system has been slightly improved due to the propagation of the knowledge in the areas where smoothing is applicable.

The dashed gray curve shows the effect of the incorporation of a priori knowledge about the monotonicity, i.e. qualitative a priori knowledge, at the points of discontinuity. This allows the system to filter out training data that otherwise would have caused the approximation to violate that monotonicity. This leads especially at the beginning of the learning process when data are sparse to a much better behavior and faster reduction of the ground truth loss. Without any a priori knowledge at the beginning the approximator is more prone to follow noisy data hence leading to violations of monotony behaviors.

Finally, the black dash-dot line shows the effect of a priori knowledge that not only incorporates knowledge about the monotonicity at the points of discontinuity but also about the height of those discontinuities, i.e. qualitative and absolute a priori knowledge. This affects the learning process at the beginning even stronger since the adaptation of the approximation near the point of discontinuity can be used to adjust the conclusions at the other side of the discontinuity as well. This can be seen in figure 2. The gray solid line shows the approximator's hypothesis without the use of a priori knowledge after 20 learning steps, while the gray dashed and boxed curve shows the approximator's hypothesis incorporating that a priori knowledge. The improvements at the points of discontinuity are clearly visible.

In the end, all kinds of correct a priori knowledge lead to approximately the same remaining ground truth loss because then the limitations of the chosen approximator determine the precision of the approximation. But the more a priori knowledge is introduced and the more concrete that knowledge is, the better does the convergence speed get.

6 Discussion

The GIESELA approach presented in this paper allows the designer to introduce his absolute, relative and qualitative local a priori knowledge into the learning process to lead it to faster convergence on a global scale. To utilize that knowledge, the designer has to be able to localize and formulate his a priori knowledge in terms of local templates only. The construction

of those locally defined templates provides an interpretable way of incorporating the a priori knowledge which is relatively easy to handle. He can further judge the effects of wrong a priori knowledge by the condition number of the resulting GIESELA matrix S'. This gives the designer a tool to judge the robustness of the approach beforehand.

By a fixed local adjustment rate for a rule, the designer is able to control the dynamics of the learning process locally. This enables him to steer the approximation throughout the whole learning process beforehand, making the approach suitable even for safety-critical applications. Of course, if the designer doesn't have that knowledge or chooses wrong parameters, the convergence process can be slowed down instead of being accelerated. By choosing a fixed high GIESELA adjustment rate α_i, the designer can lead the approximator to faster convergence, even in cases of very sparse data.

Due to the dynamics of the learning process and the changing density and conflict of the training data, it is impossible to choose a suitable constant adjustment rate for each rule. The need to do that can be eliminated by utilizing local trust values for calculating the adjustment rates. In that case, the designer must choose the parameters η_I and η_C so that they reflect the expected local sparsity and local conflict of the training data. By these offline means, the designer is able to guide the learning process dynamically at run-time.

The GIESELA approach thus offers an easy way to incorporate absolute, relative and qualitative a priori knowledge into local grid-based function approximators. By combining all local a priori knowledge into one single matrix it is made possible to interpolate as well as extrapolate knowledge globally. It thus allows to easily meet the requirements for usage in safety-critical applications where fast convergence is imperative from the first training datum on.

Due to the design as a regularizer that changes the adaptations made by a learning rule, this approach is on the one hand applicable for every learning rule. But on the other hand mathematical properties, e.g. assurance of convergence, of the learning rule are lost. To restate those properties, each learning rule and it's interaction with GIESELA has to be formally analyzed.

Future work has to be done to further guide and support designers in the process of formulating their a priori knowledge as local templates. Due to possible unwanted interactions between local templates, the designer should be visually aided in the construction of the templates for an approximator. Since this concept requires the inversion of a matrix, the complex-

ity of the approach lies in $\mathcal{O}(n^3)$ with n the number of rules. Especially in higher dimensional cases this might hinder the application of GIESELA. Hence an approximation for those calculations is an approach to reduce the complexity of the algorithm. So a future point is the acceleration of the calculations for this algorithm.

7 Summary

In this paper the Anchoring approach and the GIESELA approach for the incorporation of a priori knowledge in local function approximators of TS-0 type have been introduced. The concept shares the advantages of a relatively easy formulation of absolute, relative and qualitative local a priori knowledge of local regularizers and the global application of that knowledge in the form of a global Tikhonov regularization. This enables interpolation between points of data as well as extrapolation into regions of the input space where no data has been seen. Formal properties and measures for the robustness and stability of the concept have been introduced. The use of different kinds of a priori knowledge has been demonstrated on a simple function with smooth regions and regions of discontinuities. It has been shown that the more concrete the a priori knowledge is, the more the algorithm is able to exploit this knowledge to improve the learning process. In consequence, the presented approach leads the approximator to faster convergence allowing for a more efficient use already in the early stages of the on-line learning and generally when data lie sparsely in the input space. This allows it's use even in a safety-critical application to guarantee a certain behavior to match the requirements of that application.

References

[1] Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: *Proceedings of the 1968 23rd ACM national conference*, S. 517–524. ACM. 1968.

[2] Barnhill, R. E.: Representation and approximation of surfaces. *Mathematical software* 3 (1977), S. 69–120.

[3] Kansa, E. J.: Multiquadrics - a scattered data approximation scheme with applications to computational fluid-dynamics - I surface approximations and partial derivative estimates. *Computers & Mathematics with applications* 19 (1990) 8, S. 127–145.

[4] Crammer, K.; Dekel, O.; Keshet, J.; Shalev-Shwartz, S.; Singer, Y.: Online passive-aggressive algorithms. *The Journal of Machine Learning Research* 7 (2006), S. 551–585.

[5] Blum, M.: Fixed memory least squares filters using recursion methods. *IRE Transactions on Information Theory* 3 (1957) 3, S. 178–182.

[6] Farrell, J. A.; Polycarpou, M. M.: *Adaptive approximation based control: Unifying neural, fuzzy and traditional adaptive approximation approaches*, Bd. 48. John Wiley & Sons. 2006.

[7] Crammer, K.; Lee, D. D.: Learning via gaussian herding. In: *Advances in neural information processing systems*, S. 451–459. 2010.

[8] Kivinen, J.; Smola, A. J.; Williamson, R. C.: Online learning with kernels. *Signal Processing, IEEE Transactions on* 52 (2004) 8, S. 2165–2176.

[9] Abu-Mostafa, Y. S.: Learning from hints. *Journal of Complexity* 10 (1994) 1, S. 165–178.

[10] Tikhonov, A. N.; Arsenin, V. Y.: Solutions of ill-posed problems. *Mathematics of Computation* (1977).

[11] Golub, G. H.; Hansen, P. C.; O'Leary, D. P.: Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications* 21 (1999) 1, S. 185–194.

[12] Rosemann, N.; Brockmann, W.: Concept for Controlled Self-optimization in Online Learning Neuro-fuzzy Systems. In: *KI 2007: Advances in Artificial Intelligence*, Bd. 4667, S. 498–501. Springer. 2007.

[13] Rosemann, N.; Brockmann, W.; Neumann, B.: Enforcing Local Properties in Online Learning First Order TS-fuzzy Systems by Incremental Regularization. In: *Proc. 2009 Int. Fuzzy Systems Assoc. World Congress / 2009 European Soc. For Fuzzy Logic and Technology Conf. - IFSA/EUSFLAT 2009*, S. 466–471. EUSFLAT. 2009.

[14] Rosemann, N.: *Beherrschbares Online-Lernen durch inkrementelle, lokale Regularisierung*. Dissertation, University of Osnabrück. 2012.

[15] Buschermöhle, A.: *Reliable On-Line Machine Learning for Regression Tasks in Presence of Uncertainties (submitted)*. Dissertation, University of Osnabrück. 2014.

Automatische Generierung von Bildoperationsketten mittels genetischer Programmierung und CMA-Evolutionsstrategie

R. Kalkreuth[1], G. Rudolph[1], J.Krone[2]

[1]Fakultät für Informatik,
Technische Universität Dortmund, Otto-Hahn-Str. 14, 44221 Dortmund
Tel. (0231) 755-7701 Fax (0231) 755-7740
E-Mail: {roman.kalkeuth, guenter.rudolph }@tu-dortmund.de

[2]Institut für Computer Science, Vision und Computational Intelligence,
Fachhochschule Südwestfalen, Frauenstuhlweg 31, 58644 Iserlohn
Tel. (02371) 566-303 Fax (02371) 566-209
E-Mail: krone.joerg@fh-swf.de

Abstract

Die in der digitalen Bildverarbeitung verwendeten Verfahren setzen sich häufig aus einer Aneinanderreihung von Bildoperationen (sog. Bildoperationsketten) zusammen. Möchte man genauere Informationen über die Operationen erhalten, welche auf ein Bild angewendet wurden, ist eine Rekonstruktion notwendig. Für eine schnelle und automatische Rekonstruktion empfiehlt sich der Einsatz eines Optimierungsverfahrens. Hierbei bietet sich die genetische Programmierung an, da eine Bildoperationskette als Chromosom im Sinne der genetischen Programmierung betrachtet werden kann. Durch den evolutionären Algorithmus kann durch die Rekonstruktion auch eine Optimierung erfolgen, in dem kürzere und damit weniger rechenintensive Wege gefunden werden. Neben der Rekonstruktion der Struktur einer Bildoperationskette ist auch das Auffinden der einzelnen Parameter von Bedeutung. Zum Auffinden der Parameter kann die *Kovarianz-Matrix-Adaptions* Evolutionsstrategie (CMA - ES) zum Einsatz kommen.

1 Einleitung

In der biomedizinischen Bildverarbeitung entsteht durch den wachsenden Einsatz von bildgebenden Verfahren eine hohe Dichte an Bildmaterial. Dieses Bildmaterial häufig im Rohzustand nicht direkt verwertbar und muss mithilfe verschiedener Verfahren aufgewertet werden.

Dies betrifft unter anderem die Verbesserung der Bildqualität sowie die Extraktion bestimmter Merkmale. Die Auswertung und Aufarbeitung des Bildmaterials wird häufig mithilfe von herkömmlichen Bildbearbeitungsprogrammen durchgeführt. Die hierbei verwendeten Verfahren setzen sich häufig durch eine Aneinanderreihung von Bildoperationen zusammen. Die sequenzielle Anordnung von Bildoperationen bezeichnet man auch als Bildoperationskette. In der biomedizinischen Bildverarbeitung werden häufig große Mengen an Bilddaten manuell über verschiedene Programme bearbeitet. Dies ist vor allem in der medizinischen Bildverarbeitung der Fall, wo viele Originalaufnahmen von Hand bearbeitet werden müssen, um entsprechende Ergebnisbilder zu erhalten. Für den Benutzer der Bildbearbeitungsprogramme ist die entstandene Kette mit ihren zugehörigen Parametern zwischen dem Original- und Ergebnisbild zu meist nicht genau einsehbar. Möchte man genauere Informationen über die Bildoperationskette erhalten, welche auf ein Bild angewendet wurde, ist eine Rekonstruktion der Kette notwendig. Die Rekonstruktion ermöglicht einen genauen Überblick, welche Bildoperationen und Parameter verwendet wurden, und kann zur Optimierung der Bildoperationskette dienen. Eine Optimierung umfasst das Auffinden von kürzeren und damit weniger rechenintensiven Ketten. Dies ist vor allem bei einer großen Menge an Bildmaterial hilfreich, da in der Summe Bearbeitungszeit für die Auswertung gespart werden kann. Über die Rekonstruktion kann beispielsweise die manuelle medizinische Auswertung von Bilddaten automatisiert werden, in dem man die gefundene Kette auf einen kompletten Bilddatensatz anwendet.

Die Rekonstruktion kann durch die Anwendung verschiedener Kombinationen aus einer festgelegten Auswahl an Bildoperationen geschehen. In den meisten Fällen ist eine manuelle Rekonstruktion sehr zeitaufwendig und ungenau. Auch das automatische Abarbeiten aller möglichen Kombinationen erfordert einen großen Rechenaufwand. Für eine genauere und schnellere Rekonstruktion empfiehlt sich der Einsatz eines Optimierungsverfahrens. Es konnte durch frühere wissenschaftliche Arbeit in [1] gezeigt werden, dass die genetische Programmierung eine gute Anpassungsmöglichkeit an die sich ergebene Problemstellung bietet. Eine Bildoperationskette kann als Chromosom im Sinne der genetischen Programmierung betrachtet werden. Durch eine gezielte Anpassung der genetischen Programmierung können kürzere Ketten gefunden werden. Das Verfahren, welches in [1] vorgestellt wurde, ist imstande eine Rekonstruktion sowie eine Optimierung der Struktur einer Bildoperationskette durchzuführen. Neben der Rekonstruktion der Struktur ist auch das Auffinden der einzelnen Parameter von Bedeutung. Viele Bildoperationen werden in der Bildverarbeitung mit unterschiedlichen Werten parametriert.

Die einzelnen Parameter können dabei in ihrem Wertebereich und Typ stark abweichen. Zum Auffinden der Parameter bietet die *Kovarianz-Matrix-Adaption Evolutionsstrategie* (CMA – ES) [2] als *State of the Art* Verfahren im Bereich der Evolutionsstrategien eine gute Kombinationsmöglichkeit mit der genetischen Programmierung. Die Parameter einzelner Operatoren können in einem Vektor zusammengefasst und der CMA-ES zur Optimierung übergeben werden. Die Kombination aus genetischer Programmierung und CMA-ES geschieht über einen evolutionären Algorithmus. Als Basis für die Entwicklung der Parameterrekonstruktion diente das Verfahren aus [1]. Der evolutionäre Algorithmus verbindet die Gebiete der evolutionären Optimierung und digitalen Bildverarbeitung. Man spricht hierbei auch von „evolutionärer Bildverarbeitung“.

2 Vorbetrachtung und Lösungskonzept

2.1 Analyse der Problemstellung

Die Problemstellung beinhaltet das Auffinden der Struktur der Bildoperationskette und dessen Parameter, welche vom Originalbild zum Zielbild führt (siehe Abb. 2.1). Das Zielbild ergibt sich häufig aus einer manuellen Bearbeitung des Original- bzw. Startbildes. Es sind keine genauen Informationen über die verwendeten Bildoperationen vorhanden. Meistens liefern die verwendeten Bildbearbeitungsprogramme nur grobe Informationen, welche Bildverarbeitungsverfahren oder Routinen hinter den einzelnen Funktionen des Programms stecken. Das Gebiet der verwendeten Bildoperationen kann daher lediglich eingegrenzt werden (z.B. Segmentierung oder Kantendetektion). Bevor die genetische Programmierung und die CMA-ES eingesetzt werden können, müssen im ersten Schritt Individuen definiert werden, welche die Problemstellung anhand der ihnen zugewiesenen Fähigkeiten bzw. Gene lösen können. Zur Einordnung der Ergebnisse der einzelnen Individuen muss zu dem eine Fitnessfunktion definiert werden, welche die Distanz zum Zielbild beschreibt. Damit sich nicht zu lange Ketten bilden, kann eine maximale Länge festgelegt werden. Für das Auffinden der Parameter können zudem Wertegrenzen festgelegt werden, um ergebnislose Operationen zu vermeiden.

Abb. 2.1. Problemstellung (links Startbild – rechts Zielbild)

2.2 Anpassung der Individuen

Für das Auffinden der Struktur muss ein Individuum definiert werden, welches eine Bildoperationskette repräsentiert und über eine Fitnessfunktion eingeordnet werden kann. Bei der genetischen Programmierung kann eine Bildoperationskette als Chromosom betrachtet werden, wobei jede einzelne Bildoperation ein Gen darstellt (siehe Abb. 2.2). Die Länge des Chromosoms kann von 1 bis zur maximalen Länge variieren. Jedes Individuum erhält damit eine Funktion, welche auf das Startbild angewendet werden kann. Das daraus resultierende persönliche Ergebnis kann zur Einordnung der Individuen verwendet werden. Für das Auffinden der Parameter muss das Individuum für den Einsatz der CMA-ES angepasst werden. Neben dem Chromosom müssen Vektoren für die einzelnen Parameter der Bildoperationen definiert werden. Für die CMA-ES sind zu dem noch Vektoren für Schrittweiten und Wertegrenzen notwendig (siehe Abb. 2.3). Die einzelnen Individuen können nach ihrer Anpassung in einer Population zusammengefasst werden.

Abb. 2.2 Chromosom aus drei Bildoperationen

$$\vec{X} = \begin{pmatrix} x1 \\ x2 \\ \dots \\ xn \end{pmatrix} \qquad \vec{\sigma} = \begin{pmatrix} \sigma 1 \\ \sigma 2 \\ \dots \\ \sigma n \end{pmatrix} \qquad \vec{L} = \begin{pmatrix} l1 \\ l2 \\ \dots \\ \ln \end{pmatrix} \qquad \vec{U} = \begin{pmatrix} u1 \\ u2 \\ \dots \\ un \end{pmatrix}$$

Abb. 2.3 Vektoren für Parameter, Schrittweiten und Wertegrenzen

2.3 Fitness und Selektion

Die Fitness der Individuen wird über die Distanz zwischen Ergebnisbild eines Individuums und dem Zielbild gebildet. Als genaues und simples Distanzmaß bietet sich die Summe der abweichenden Grauwerte an, welche *pixelweise* zwischen Bild A und Bild B gebildet wird (Manhattan-Distanz, siehe 2.3). Die Summe der abweichenden Grauwerte aus 2.3 kann als *Raw-Fitness* F_{raw} gewählt werden. Damit sich die Fitnesswerte zwischen 0 und 1 bewegen, wird aus der *Raw-Fitness* die *Adjusted-Fitness* F_{adj} nach [3] gebildet (2.4).

$$F_{raw} = \sum_{j=1}^{height} \sum_{i=1}^{width} \left| A[j,i] - B[j,i] \right| \qquad (2.3)$$

$$F_{adj} = \frac{1}{(1 + F_{Raw})} \qquad (2.4)$$

Ziel der Selektion ist es, gute Teilergebnisse für die Rekombination aufzufinden. Dies gilt zu gleich für die Strukturanpassung als auch für die Parameterentwicklung. Für die Selektion bietet sich zum einen die Turnierselektion als Selektionsstrategie an. Hierbei treten Individuen, welche zufällig aus der Population ausgewählt werden, in Turnieren gegeneinander an. Sieger eines Turniers ist das Individuum mit der höchsten Fitness. Der Vorteil dieser Selektionsstrategie liegt darin, dass sich Individuen mit guter Fitness durchsetzen. Der Selektionsdruck kann zu dem über die Turniergröße eingestellt werden. Es werden so viele Turniere durchgeführt, bis die erforderliche Anzahl an Eltern-Individuen für die Rekombination erreicht ist.

2.4 Genetische Anpassung der Chromosomen

Die genetische Anpassung der Chromosomen geschieht mittels Rekombination und Mutation im Sinne der genetischen Programmierung. Bei der Rekombination kann für kleinere Ketten (bis 10 Operationen) mit einem Kreuzungspunkt (*One-Point-Crossover*) gearbeitet werden. (siehe Abb. 2.4).

Von den zur Rekombination ausgewählten Eltern wird jeweils die Hälfte der Gene auf das neue Kind-Individuum übertragen. Für längere Ketten empfiehlt sich der Einsatz mehrerer Kreuzungspunkte (*Two-Point-Crossover, Uniform-Crossove*r) um eine vielfältigere Mischung der Gene zu erreichen.

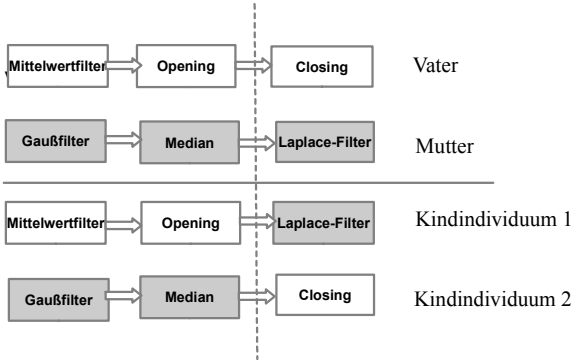


Abb. 2.4 Rekombination der Chromosomen mittels One-Point-Crossover

Enthalten die Eltern-Individuen gleiche Bildoperationen, so müssen deren Parameter und Schrittweiten rekombiniert werden. Dies wird für die Parameter $\vec{X}$ und Schrittweiten $\vec{\sigma}$ mittels arithmetischem Crossover (2.7) durchgeführt. Hierbei wird die Rekombination von zwei Parametern P1 und P2 vorgenommen, wobei α zufällig gewählt wird.

$$P_{cross} = \alpha P_1 + (1-\alpha) P_2 \qquad \alpha \in [0,1] \tag{2.7}$$

Neben der Rekombination wird auch eine Mutation der Chromosomen vorgenommen. In [1] wurde dies lediglich durch das Austauschen eines Gens durch ein zufälliges, neu gezogenes Gen durchgeführt. Um die Vielfältigkeit innerhalb der Population zu erhöhen, empfiehlt sich das Hinzufügen bzw. Entfernen, sowie das Vertauschen von Operationen als Mutation anzuwenden.

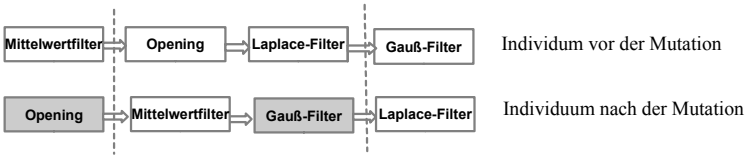


Abb. 2.5 Mutation durch Vertauschen mit 2 Vertauschungspunkten

2.5 Anpassung der Parameter

Nach der Strukturanpassung kann die Anpassung der Parameter erfolgen. Die Anpassung bzw. die Weiterentwicklung der Parameter für ein Individuum geschieht, wie schon erwähnt, über die CMA-ES. Gute Strukturergebnisse werden der CMA-ES zur Parameterentwicklung übergeben. Das heißt, dass nur Individuen mit entsprechender Fitness mittels CMA-ES weiterentwickelt werden. Es wird also nur eine gewisse Anzahl der besten Individuen aus der aktuellen Population verwendet. Die Struktur der Chromosomen bleibt bei der Anwendung der CMA-ES unverändert. Es werden lediglich die in Abb. 2.3. festgelegten Vektoren übergeben. Nach der Weiterentwicklung werden die Individuen wieder in die Population zurückgeführt. Durch den Einsatz von adaptiven Schrittweiten und „Suchrichtungen“ bzw. Lagewinkeln über die Kovarianzmatrix kann ein schneller Suchfortschritt beim Auffinden der Parameter erzielt werden. Bei der CMA-ES werden die in der Kovarianzmatrix enthaltenen Kovarianzen von Schrittweiten und Lagewinkeln in jeder Generation mit neuen Informationen aktualisiert, um diese nicht von Generation zu Generation neu berechnen zu müssen. Die Informationen basieren auf guten möglichen Lösungen, welche in der aktuellen Generation vorkommen. Ziel ist es, die Wahrscheinlichkeit von vormals guten Lösungsschritten zu erhöhen. Für das Update der Kovarianzmatrix stehen zwei Strategien zur Verfügung: das *Rank - μ Update* sorgt für eine gute Nutzung der Informationen für das Update aus einer Generation. Diese Update-Strategie empfiehlt sich bei großen Populationsgrößen. Das *Rank - one Update* hingegen nutzt Informationen für das Update generationsübergreifend und ist speziell wichtig bei kleineren Populationsgrößen. Abhängig von der Dimension eines Individuums bzw. seiner Parameter können *Rank- μ Update* oder *Rank-One Update* eingesetzt werden. Nach der Optimierung mittels CMA-ES kann die gewichtete Mittelwertlösung der besten Generation als beste Lösung verwendet werden. Diese wird letztlich in die aktuelle Population re-integriert.

2.5.1 Genetische Anpassung der Parameter

Die CMA-ES erzeugt in jeder Generation neue Lösungen über die Normalverteilung (2.9). Hierbei gibt g die aktuelle Generation an und $m^{(g)}$ den Mittelwert. Die globale Schrittweite $\sigma^{(g)}$ dient zur Skalierung der neuen Generation und $N(0,C^{(g)})$ „verzerrt“ die Normalverteilung gemäß der in der Kovarianzmatrix $C^{(g)}$ enthaltenen Kovarianzen von Schrittweiten und Lagewinkeln. Die genetische Anpassung wird für λ neue Lösungen durchgeführt

$$x_k^{(g+1)} = m^{(g)} + \sigma^{(g)} N\left(0, C^{(g)}\right) \text{für } k = 1,\ldots,\lambda \qquad (2.9)$$

2.6 Evolutionärer Algorithmus

Im evolutionären Algorithmus werden die zuvor besprochenen Selektions- und Rekombinationsstrategien iterativ miteinander verbunden. Der evolutionäre Algorithmus wird hierzu in zwei Teile aufgeteilt. Der äußere Teil übernimmt die Entwicklung der Struktur mittels genetischer Programmierung. Der innere Teil des evolutionären Algorithmus übernimmt die Entwicklung der Parameter mittels CMA-ES. In jeder Generation des evolutionären Algorithmus wird nach der Strukturanpassung die Parameterentwicklung vorgenommen. Alternativ kann die Parameterentwicklung erst gestartet werden, wenn die Individuen einen gewissen Grad an Entwicklung durch die Strukturanpassung erreicht haben. Vor dem Start des Algorithmus wird eine Anfangspopulation erzeugt, welche die erste Generation bildet. Mit jeder Iteration wird über die Rekombination eine neue Generation gebildet. Vor jedem Durchlauf wird geprüft, ob das Abbruchkriterium erfüllt ist. Dies kann die optimale Fitness eines Individuums oder die maximale Anzahl an Generationen sein. Die optimale Fitness wäre bezogen auf die Problemstellung eine gewünschte Übereinstimmung mit dem Zielbild.

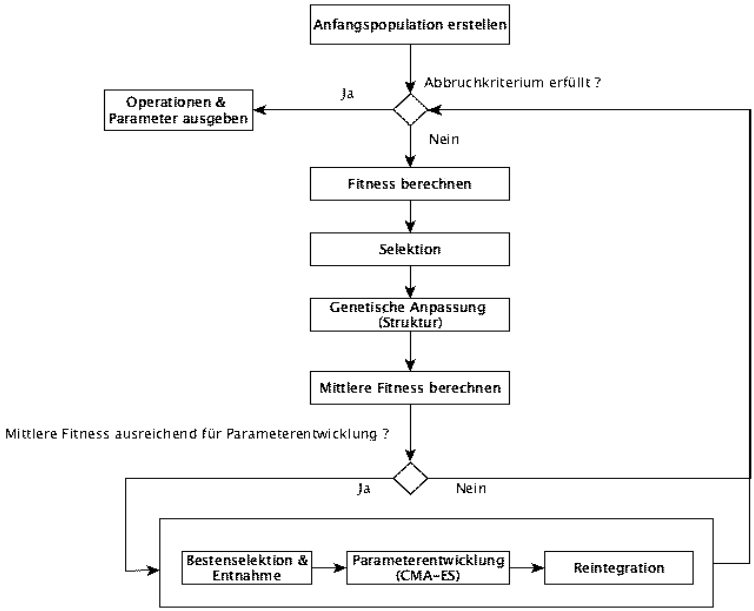


Abb. 2.6 Programmablaufplan evolutionärer Algorithmus

3 Praktische Umsetzung

Für die praktische Umsetzung des Lösungskonzepts wurde eine Java-Anwendung erstellt. Als Vorlage diente die Java-Anwendung aus [1].

3.1 Codierung und Implementierung der Bildoperationen

Die verschiedenen Bildoperationen werden mit Zahlen oder Buchstaben codiert. Auf diese Weise können die Bildoperationsketten bzw. Chromosomen als Permutationen dargestellt werden. Jede Bildoperation wird zu Beginn mit einer eindeutigen Zahl oder einem Buchstaben belegt.

```
Sobel-Operator          =>    4
Laplace-Filter          =>    7
Mittelwertfilter        =>    A
Gaussfilter             =>    9
Segmentierung           =>    5
```

Tabelle 3.1 Bildoperationen mit Bezeichner

Nach der Codierung werden entsprechende Permutationen gebildet. Die Länge der Permutationen variiert entsprechend von eins bis zur maximal festgelegten Kettenlänge. Für jene Bildoperationen, welche Parameter besitzen werden entsprechende Vektoren mit Zufallswerten gebildet.

```
Individuum 1 (Chromosom : 2691A)   x2=(3,10,0.3)
                                   xA=(5)

Individuum 2 (Chromosom : 560B)    x5 = (127)
                                   xB = (0.2)

Individuum 3 (Chromosom : 4203F)   x0 = (3, 0.5)
                                   xF = (7, 0,3)
```

Tabelle 3.2 Drei gebildete Individuen mit Chromosomen und Parametervektoren

Die Bildoperationen wurden mit OpenCV [4] implementiert. OpenCV zählt zu den größten Bildverarbeitungsbibliotheken und bietet eine große Auswahl an Bildoperationen aus verschiedenen Bereichen der Bildverarbeitung.

3.2 CMA-Evolutionsstrategie

Für den Einsatz der CMA-ES wurde die Java Implementierung von [7] verwendet.

3.3 Evolutionärer Algorithmus

Der evolutionäre Algorithmus wurde nach dem Programmablaufplan aus Abb. 2.6 umgesetzt. Der Algorithmus wurde in zwei Funktionen unterteilt, welche jeweils die Struktur- bzw. die Parameterentwicklung vornehmen. Notwendige Parameter wie die Anzahl der Generationen oder die Anzahl von Eltern- und Kindindividuen werden vor dem Start des Algorithmus separat initialisiert.

Das Abbruchkriterium stellt, wie schon beschrieben, eine Restdistanz zum Zielbild oder eine festgelegte maximale Anzahl an Generationen dar. Neben dem Ablauf aus Selektion, Rekombination und Mutation wird in jeder Generation die durchschnittliche Fitness der aktuellen Generation berechnet und geprüft, ob diese für die Parameterentwicklung ausreichend ist. Für die Parameterentwicklung werden n Individuen aus der Elite entnommen, welche dann mittels CMA-ES weiterentwickelt werden. Die Selektion aus der Elite kann nach zwei Kriterien geschehen : Selektion der besten Individuen oder per Zufall. Den mittels CMA-ES entwickelten Individuen wird die Mittelpunktlösung der CMA-ES zugewiesen bevor diese wieder in die aktuelle Generation re-integriert werden.

```
Evolutionärer Algorithmus (Abbruchkriterium, CMA-ES Startfitness, Genetische Parameter )
1   Elternpopulation ← erzeuge Anfangspopulation
2   while Abbruchkriterium nicht erfüllt
3   do berechne Fitness von Elternpopulation
4      Eltern ← Selektion(Elternpopulation)
5      Kindpopulation ← Rekombination(Eltern)
6      Mutaton(Kindpopulation)
7      Mittlere_Fitness← bereche mittlere Fitness (Kindpopulation)
8      if Mittlere_Fitness >= CMA-ES Startfitness
9      then Elite ← Bestenselektion(Kindpopulation)
10            CMA_Individuen ← Entnehme n Individuen aus Elite
11            Parameterentwicklung(CMA_Individuen, Genetische Parameter)
12            Integriere CMA_Individuen in Kindpopulation
13  Best ← Selektiere bestes Individuum
14  return Best
```

Listing 3.1 Evolutionärer Algorithmus

```
Parameterentwicklung (CMA_Individuen, Genetische Parameter)
1   for each Indviduum in CMA_Individuen
2   do CMA-Lösungen ← CMA-ES (Individuum)
3      Indvidiuum ← Mittelpunktbildung(CMA-Lösungen)
4   return Elite
```

Listing 3.2 Parameterentwicklung

4 Ergebnisse

Um das Verfahren zu testen, wurde der Benchmark Nr. 6 aus der biomedizinischen *Broad Bio Image Benchmark Collection* [5] verwendet. Der Benchmark enthält Problemstellungen im Bereich der Segmentierung und Zellerkennung, welche sich jeweils aus Originalaufnahmen und Referenzbildern *(Ground-Truth)* zusammensetzen.

Die kontrastarmen Aufnahmen enthalten menschliche U2OS-Zellen. Die Referenzbilder wurden vom *Broad-Institut* segmentiert und die Anzahl der Zellen festgehalten. Der Benchmark eignet sich sehr gut für eine praxisnahe Anwendung der evolutionären Bildverarbeitung. Über den evolutionären Algorithmus soll die Bildoperationskette zwischen den Rohbildern und den Referenzbildern (Abb. 4.1) rekonstruiert werden.

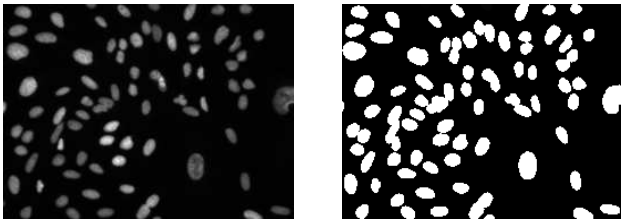


Abb. 4.1 Kontrastoptimierte Originalaufnahme (links) und Ground-Truth(rechts) aus dem Benchmark Nr. 6

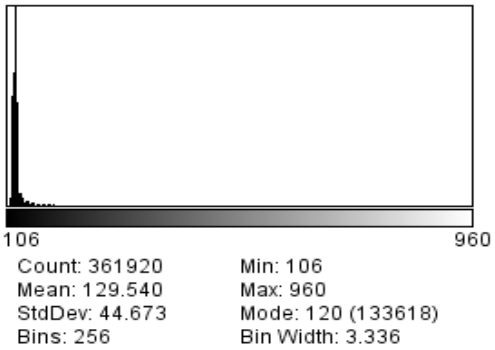


Abb. 4.2 Histogramm einer Originalaufnahme des Benchmarks

4.1 Auswertung des Benchmark

Aus dem Datensatz Nr. 16 (Aufnahmen im Fokus) mit 384 Zellbildern wurde eine Stichprobe von 32 Bildern entnommen. Der Benchmark wurde unter dem Gesichtspunkt der Segmentierung ausgewertet und die Referenzbilder entsprechend binarisiert. Dem evolutionären Verfahren lagen zur Rekonstruktion die Originalaufnahme und das Ground-Truth vor.

Um die Ergebnisse einordnen zu können, wurde als Vergleichsverfahren das sogenannte Otsu-Verfahren[8] ausgewählt. Das Otsu-Verfahren ist ein automatisches Segmentierungsverfahren, welches mit statistischen Mitteln arbeitet. Um die Anzahl der Zellen in den Ergebnisbildern beider Verfahren sowie der binarisierten Referenzbilder zu ermitteln, wurde die Objekterkennung des *Cell-Profilers*[6] verwendet. Für den Vergleich wurde die relative Abweichung F (4.1) zwischen Anzahl der Zellen im Ergebnisbild der beiden Verfahren n und der Anzahl der Zellen im Referenzbild n_{ref} gebildet. Neben der Anzahl der Zellen wurde auch die mittlere Größe des Zellbereiches *(mean cell area)* für die Auswertung der Ergebnisse einbezogen.

$$F = \frac{|n - n_{ref}|}{n_{ref}} \qquad (4.1)$$

Bei der Segmentierung können durch einen falschen Schwellwert, kleine Objekte entstehen, welche fälschlicherweise als Zellen erkannt werden. Um diese Objekte zu filtern, wurde auf das Ergebnisbild des Otsu-Verfahrens eine Opening-Operation jeweils mit 3x3 und 5x5 (S=3,S=5) angewendet. Der Suchraum des evolutionären Ansatzes setzte sich aus folgenden Operationen zusammen : globale und lokale Segmentierungsverfahren, Rauschunterdrückung, morphologische Operationen und Watershed-Verfahren. Tabelle 4.1 zeigt die wichtigsten Parameter, mit welchen der evolutionäre Algorithmus eingestellt wurde. Für die Problemstellung waren maximal 80 Individuen ausreichend. Die Anzahl der Individuen, welche mittels CMA-ES weiterentwickelt wurden (*CMA-Indviduen*), lag bei 5. Bei der Bearbeitung des Benchmarks variierte die Länge der Chromosomen zwischen 2 und 6 Operationen. Die Dimension Parameter lag abhängig vom Testfall zwischen 4 und 11 Parametern.

Genetische Programmierung	**CMA-ES**
Größe Elternpopulation : 60	$\mu = 10$, $\lambda = 20$
Größe Kindpopulation : 80	*Max. Durchläufe = 20*
Anzahl Elite : 10	*Anzahl CMA-Indviduen : 5*
Max. Anzahl Generationen : 10	
Mutationswahrscheinlichkeit : 0.2	
Selektionsstrategie : Turnier-Selektion	

Tabelle 4.1 Wichtige Parameter für den evolutionären Algorithmus

	GT		EA		Otsu S=3		Otsu S=5	
Images	n_{ref}	C_{mean}	$F(\%)$	C_{mean}	$F(\%)$	C_{mean}	$F(\%)$	C_{mean}
a01_s2	11	**1078**	18,0	**838**	100,0	**0**	100,0	**0**
a07_s2	40	**1580**	5,0	**1499**	27,5	**727**	25,0	**723**
b01_s1	44	**1393**	2,2	**1405**	27,3	**659**	27,3	**653**
b13_s1	85	**1403**	4,7	**1388**	21,1	**990**	17,6	**1010**
c02_s1	82	**1484**	0,0	**1479**	19,5	**1121**	19,5	**1113**
c11_s1	67	**1749**	5,9	**1646**	35,8	**1173**	35,8	**1162**
d03_s2	86	**1498**	0,0	**1486**	22,1	**1096**	22,1	**1087**
d18_s1	64	**1591**	4,6	**1516**	12,5	**1124**	10,9	**1130**
e06_s1	70	**1337**	2,8	**1326**	10,0	**1061**	7,1	**1075**
e21_s2	76	**1642**	6,5	**1792**	34,2	**1068**	30,2	**1092**
f01_s1	63	**1606**	1,5	**1544**	14,2	**1235**	14,2	**1223**
f15_s1	93	**1647**	1,1	**1679**	28,0	**1117**	28,0	**1117**
g04_s2	79	**1545**	3,7	**1598**	12,7	**1249**	14,1	**1208**
g23_s1	87	**1417**	5,7	**1499**	13,8	**1087**	14,9	**1070**
h01_s1	40	**1484**	0,0	**1499**	17,5	**762**	17,5	**752**
h19_s2	79	**1023**	10,6	**1300**	15,2	**1088**	11,4	**1094**
i02_s1	85	**1421**	7,1	**1322**	9,4	**1170**	8,2	**1170**
i11_s2	80	**1390**	7,5	**1300**	15,0	**1051**	15,0	**1045**
j07_s1	83	**1387**	6,0	**1320**	22,9	**1012**	18,0	**1034**
j18_s2	76	**1501**	3,9	**1443**	22,4	**1042**	19,7	**1051**
k09_s1	64	**1328**	0,0	**1330**	12,5	**1051**	12,5	**1044**
k12_s1	77	**1496**	3,8	**1466**	16,9	**1115**	15,6	**1118**
I03_s2	77	**1686**	7,7	**1835**	26,0	**1197**	24,7	**1200**
l20_s2	84	**1797**	0,0	**1799**	40,5	**1101**	44,0	**1061**
m08_s1	66	**1423**	1,5	**1442**	14,9	**1064**	18,2	**1022**
m24_s2	67	**1366**	5,9	**1288**	13,4	**1049**	13,4	**1039**
n10_s2	80	**1466**	5,0	**1388**	17,5	**1037**	13,9	**1031**
n14_s1	68	**2053**	2,9	**2017**	35,3	**1463**	36,8	**1438**
o05_s1	83	**1390**	1,2	**1387**	18,1	**1023**	15,7	**1033**
o19_s2	83	**1523**	2,4	**1475**	18,1	**1103**	18,1	**1093**
p02_s1	51	**1258**	9,8	**1356**	11,8	**620**	11,8	**612**
p13_s1	60	**1335**	6,6	**1280**	6,6	**1059**	6,6	**659**
μ	**70**	**1495**	**4,8**	**1467**	**22,3**	**1022**	**21,5**	**1005**

GT Ground Truth
EA Evolutionärer Ansatz
F(%) Relative Abweichung in %
Cmean Mean cell area

Tabelle 4.2 Ergebnisse Benchmark BBBC 6

Die Auswertung der Ergebnisse in Tabelle 4.3 zeigt, dass der evolutionäre Ansatz die Segmentierungsprobleme besser lösen kann, als das Otsu-Verfahren. Die prozentuale Abweichung F(%) fällt mit 4.8% geringer aus, als das Otsu-Verfahren mit > 20%. Auch der mittlere Zellbereich liegt näher an der Referenz als das Otsu-Verfahren. Der Vorteil des Otsu-Verfahrens liegt darin, dass keine Referenz zur Lösung eines Segmentierungsproblems benötigt wird. Der evolutionäre Ansatz benötigt zur Lösung zwar ein Referenzbild, ist aber im Stande alternative Ketten mit ihren entsprechenden Parametern zu finden. Abb. 4.1 zeigt den Vergleich eines Einzelergebnis zwischen Ground-Truth, evolutionärem Ansatz und Otsu-Verfahren. Man kann erkennen, dass mithilfe des evolutionären Ansatzes mehr Objekte segmentiert werden können, als mit dem Otsu-Verfahren. Allerdings weist auch das Ergebnisbild es evolutionären Ansatzes noch falsche Objekte auf. Für eine sehr genaue Zellerkennung wie sie im Ground Truth zu sehen ist, müssen noch komplexere Zellerkennungsalgorithmen im evolutionären Ansatz implementiert werden.

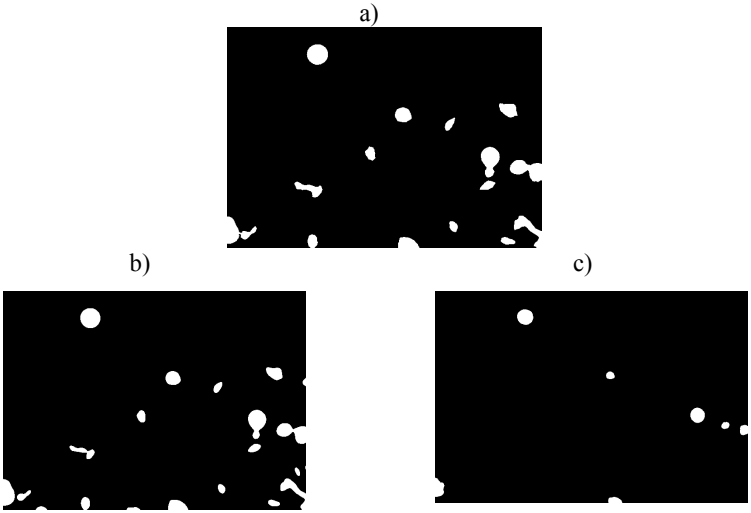


Abb. 4.1 Vergleich eines Einzelergebnis : a) Ground-Truth b) Ergebnis des evolutionären Ansatzes c) Ergebnis des Otsu-Verfahrens

5 Zusammenfassung und Ausblick

Es konnte gezeigt werden, dass mittels evolutionärer Optimierung reale Bildverarbeitungsprobleme aus der biomedizinischen Bildverarbeitung gelöst werden können. Die Kombination aus genetischer Programmierung und Evolutionsstrategie ist imstande, Struktur und Parameter einer Bildoperationskette zu rekonstruieren und zu optimieren. Wie schon beschrieben, werden in der biomedizinischen Bildverarbeitung große Mengen an Bilddaten von Hand ausgewertet. Der evolutionäre Algorithmus kann über die genaue Rekonstruktion der eingesetzten Bildoperationen zur Ableitung eines automatischen Verfahrens dienen. Als effektiv stellte sich beim Einsatz der CMA-ES die Re-Integration der Mittelpunktlösung heraus. Die Re-Integration der Mittelpunktlösung ist der Lösung des besten Individuums vorzuziehen, da diese einen größeren Lösungsraum repräsentiert als die beste Punktlösung. Ein Nachteil des Verfahrens stellt der hohe Rechenaufwand dar. Durch die Schachtelung der CMA-ES innerhalb der genetischen Programmierung müssen viele Fitnessberechnungen mittels Bildvergleich durchgeführt werden. Für ein schnelleres und effizienteres Auffinden einer Bildoperationskette muss die Fitnessberechnung bzw. das Distanzmaß optimiert werden. Hierbei kann zum einen die Berechnung der Fitnessfunktion parallelisiert werden oder der evolutionäre Algorithmus auf Teilalgorithmen aufgespalten werden. Des weiteren sollte der Einsatz von effizienten Verfahren zum Vergleichen von Bildern untersucht werden. Hierbei bietet sich u.a. der Einsatz von sogenannten *Keypoint-Verfahren* an, welche den Vergleich auf signifikante Merkmalsbereiche beschränken. Das Verfahren arbeitet derzeit nur mit linearer genetischer Programmierung. Der Einsatz von Verzweigungen bzw. Baumstrukturen stellt eine wichtige Erweiterung für die Rekonstruktion und Optimierung von komplexeren Bildverarbeitungsverfahren dar. In Zukunft sollen auf Basis von Baumbasierter genetischen Programmierung weitere Benchmarks mit industriellem und biomedizinischen Ursprung bearbeitet werden.

Literatur- und Quellangaben

[1] Kalkreuth, Roman; Krone, Jörg; Schneider, Michael (2012): „*Automatische Generierung von Bildoperationsketten mittels genetischer Programmierung*," Fachhochschule Südwestfalen, Institut für Computer Vision & Computational Intelligence, Erschienen in: *Proceedings 22. Workshop Computational Intelligence*, Seite(n) 325–340, 2012, ISBN 978-3-86644917-6.

[2] Hansen, Nikolaus (1998) : „*Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie : Eine Untersuchung zur entstochastisierten, koordinatensystemunabhängigen Adaptation der Mutationsverteilung*", Dissertation, Mensch & Buch Verlag, Berlin, ISBN 978-3933346292.

[3] Koza, J.R. (1992) : „*Genetic Programming : On the Programming of Computers by Means of Natural Selection",* The MIT Press, ISBN 978-0262111706.

[4] OpenCV : *Open Source Computer Vision*, URL : www.opencv.org

Abrufdatum : 10.08.2013

[5] BBBC : *Broad Bioimage Benchmark Collection,* Broad Institute, URL : www.broadinstitute.org/bbbc/

Abrufdatum : 10.08.2013

[6] CellProfiler : *cell image analysis software,* Broad Institute, URL : www.cellprofiler.org

Abrufdatum : 10.08.2013

[7] CMA-ES Source Code**,** URL : www.lri.fr/~hansen/cmaes_inmatlab.html

Abrufdatum : 10.08.2013

[8] Otsu, Nobuyuki (1979) : „*A threshold selection method from gray-level histograms",* Erschienen in : „Systems, Man and Cybernetics, IEEE Transactions on Volume:9; Issue: 1", Seite(n) : 62–66

Evolutionäre Strukturoptimierung für LOLIMOT

Jan Braun, Christoph Krimpmann, Frank Hoffmann, Torsten Bertram

Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund
Otto-Hahn-Str. 8, 44221 Dortmund
Tel.: (0231) 755-3592
Fax: (0231) 755-2752
E-Mail: jan.braun@tu-dortmund.de

Kurzfassung

Der datenbasierten Identifikation von Parametern dynamischer Systeme geht die Auswahl einer geeigneten Modellstruktur voraus. Der Beitrag stellt einen multikriteriellen evolutionären Algorithmus vor welcher, für LOLIMOT die pareto-optimalen Kompromisslösungen zwischen Modellgüte und Modellkomplexität ermittelt. Aus dieser Menge wählt der Anwender a posteriori das für ihn geeignete Modell aus, anstatt sich im vor hinein auf eine Modellstruktur festzulegen, oder in einem Trial-and-Error-Prozess Strukturauswahl und Parameteridentifikation einander anzupassen. Der iterative Identifikationsprozess von LOLIMOT erfordert eine Anpassung des evolutionären Algorithmus. Eine experimentelle Analyse vergleicht den neuen Ansatz mit etablierten Methoden der datenbasierten Modellierung hinsichtlich ihrer Generalisierungsfähigkeit.

1 Einführung

Die datenbasierte Modellierung von dynamischen Systemen ist eine bewährte Methode zum Zweck der Systemanalyse, Prädiktion, Simulation und Reglerentwicklung. In der Literatur finden sich eine Vielzahl von Verfahren für die nichtlineare Regression, u.a. polynomiale Modelle, Neuronale Netze, Fuzzy-Systeme und lokale Neuro-Fuzzy-Modelle. Ziel der Identifikation ist eine hohe Generalisierungsfähigkeit, also einen niedrigen Modellfehler auf ungesehenen Validierungsdaten, zu erzielen. Im Kontext der Reglerentwicklung legt der Nutzer Wert auf die Interpretierbarkeit und Analysierbarkeit des generierten Modells. Die Auswahl einer geeigneten

Modellstruktur lässt sich in drei Problemstellungen unterteilen: Die Auswahl des Modellierungsansatzes, der Konflikt zwischen Bias und Varianz, und die Abwägung zwischen Modellgüte und Interpretierbarkeit. Jedes Modell enthält inhärent Strukturannahmen, die mehr oder weniger für das zu modellierende System zutreffend sind. Dies begrenzt wie gut die Modellstruktur den Prozess abbilden kann. Ein lineares Modell kann beispielsweise ein nichtlineares System niemals perfekt modellieren, selbst wenn Trainingsdaten unbegrenzt zur Verfügung stehen und das Modell eine hohe Ordnung hat. Der Bias/Varianz-Konflikt ist mathematisch begründet [1]. Der Bias des Modells ist der Fehler, der selbst bei optimal eingestellten Parametern besteht. Bei Erhöhung der Parameteranzahl des Modells eröffnen sich zusätzliche Freiheitsgrade, die es erlauben den Modell-Bias zu reduzieren. Die Identifikation der Modell-Parameter erfolgt jedoch auf Basis eines in der Größe limitierten und meist rauschbehafteten Datensatzes. Die zur Verfügung stehende Information ist begrenzt. Eine Erhöhung der Parameteranzahl bedeutet zwangsweise auch eine Erhöhung des Varianz-Fehlers, da die Parameter weniger genau identifiziert werden können. So ergibt sich für jede Modellklasse ein Optimum zwischen Bias und Varianz, in der das Modell genug Freiheitsgrade hat, um sich den Daten anzupassen und genug Information zur Verfügung steht, um die Parameterwerte akkurat zu bestimmen. Wie gut das Modell in diesem Optimum ist, hängt zum einen davon ab, wie gut die Modellstruktur den Prozess abbilden kann. Zum anderen limitiert der Informationsgehalt des Datensatzes die erreichbare Güte. Entscheidend sind hier beispielsweise die Form des Anregungssignals, die Anzahl der Messpunkte und das Signal-Rausch-Verhältnis. Ermitteln lässt sich der Varianz-Fehler nur mit der Hilfe von Daten, die nicht für das Training verwendet werden, den Validierungs- und Testdaten. Geeignete Methoden zur Auswahl von Parametern sind modellabhängig. Üblich sind beispielsweise Wrapper-Methoden für die Hinzunahme von Parametern [2] (Vorwärtsselektion), oder der Rückwärtselimination [3]. Die Abwägung zwischen Modellgüte und Interpretierbarkeit des Modelles ist stark von den Nutzerpräferenzen und dem Anwendungszweck des Modells abhängig. Einige Modellklassen ermöglichen dem Anwender den Einsatz effizienter Analysemethoden, und Werkzeuge zum Beispiel für die Reglerentwicklung. Lineare Modelle bieten klar die besten Voraussetzungen. Vorteilhaft sind auch Modelle die lokal linear sind, wie Lolimot, oder lineare Komponenten enthalten wie Wiener-Hammerstein-Modelle. So eignen sich lokal lineare Modelle für die Reglersynthese mit TSK-Fuzzy-Reglern und ermöglichen mit LMI-Methoden den Nachweis globaler Stabilität. Auch die Bias/Varianz-Entscheidung muss nicht zwangsläufig zugunsten des Modells mit der besten Generalisierungsfähigkeit getroffen werden.

Steht die Interpretierbarkeit im Fokus, kann ein Modell mit weniger Parametern vorteilhaft sein, selbst wenn dafür ein höherer Bias-Fehler in Kauf genommen werden muss. Es existiert keine allgemeine Metrik zur Messung von Komplexität. Oft wird die Anzahl der einstellbaren Parameter eines Systems zur Charakterisierung der Komplexität verwendet. Dieses Maß ist gut geeignet, um zwischen Modellen eines Typs zu unterscheiden, verliert aber beim Vergleich unterschiedlicher Modellklassen an Aussagekraft.

Der Beitrag greift alle drei Fragestellungen der Strukturauswahl auf. Kapitel 2 stellt einen evolutionären Algorithmus vor der für LOLIMOT die Kompromisslösungen aus Modellgüte und Komplexität approximiert. Der Vergleich verschiedener Modellklassen ist subjektiv und stark von weiteren Randbedingungen bestimmt, als dass sich der Prozess automatisieren ließe. In Kapitel 4 werden Pareto-Fronten entwickelt, die dem Nutzer bei einer objektiven Entscheidungsfindung unterstützen.

1.1 LOLIMOT

Die datenbasierte Modellierung eines dynamischen Prozesses erfolgt auf Basis einer Messreihe von Eingangssignalen $\underline{u} \in \mathbb{R}^N$ und Ausgangssignalen $\underline{y} \in \mathbb{R}^N$. Unter der Annahme eines linearen Zusammenhangs ergibt sich das ARX (AutoRegressiv with eXogenous Inputs) Modell:

$$\begin{aligned}\hat{y}(t) + a_1\, y(t-1) + \ldots + a_{na}\, y(t-n_a) = b_1\, u(t-n_k) + \\ + b_2\, u(t-n_k-1) + \ldots + b_{nb}\, u(t-n_k-n_b+1) + e(t)\end{aligned} \quad (1)$$

Die Parameter $(a_1, a_2, \ldots, a_{na}, b_1, b_2, \ldots, b_{nb})$ werden numerisch effizient mit Hilfe der Least-Squares-Methode bestimmt. Die Modellordnung ergibt sich aus der Anzahl der Verzögerungen des Eingangs n_b und des Ausgangs n_a. Diese Strukturparameter sind a priori gewählt, ebenso wie die Anzahl der vorgeschalteten Verzögerungselemente n_k welche die Totzeit des Systems modellieren.

Für nichtlineare Prozesse stellt das ARX-Modell nur eine Approximation dar. Ein nichtlineares Modell ergibt sich aus der Unterteilung des Eingangsraums in M Bereiche, mit lokalen linearen Modellen (LLMs). Das Gesamtmodell wird aus der gewichteten Überlagerung der Teilmodelle mit der Hilfe der lokalen Gültigkeitsfunktionen $\Phi(\cdot)$ gebildet. Diese liefern abhängig vom Arbeitspunkt einen Wert zwischen 0 und 1, und definieren den Beitrag des lokalen Modells zur Ausgangsgröße.

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i(\underline{r})\Phi_i(\underline{r}) \tag{2}$$

Die Regressoren $\underline{r}$ dieser Modelle sind:

$$\underline{r} = [u(t-n_k), u(t-n_k+1), u(t-n_k+2), \ldots u(t-n_k-n_b+1), \\ y(t-1), y(t-2), \ldots, y(t-n_a), 1]^{\mathsf{T}}, \tag{3}$$

eine Kombination aus verzögerten Eingangswerten u und verzögerten Ausgangswerten y und einem konstanten Anteil um statische Abweichungen zu repräsentieren. LOLIMOT (LOcal LInear MOdel Tree) ist eine effiziente, iterative Methode zur Generierung eines Modellbaumes aus lokalen Teilmodellen. Ausgehend von einem globalen linearen Modell wird der Arbeitsraum $\underline{r}$ senkrecht entlang jeder Dimension geteilt und für die entstehenden Teilräume zwei LLMs trainiert. Diejenige Teilung, welche das Gesamtmodell mit dem kleinsten Fehler erzeugt wird beibehalten. Anschließend wird das LLM mit dem größten lokalen Fehler abermals unterteilt. So entsteht iterativ eine Baumstruktur aus lokalen, linearen Modellen. In [4] finden sich umfassende Erläuterungen zu LOLIMOT und verwandten Ansätzen. Die in dieser Arbeit verwendete Implementierung ist das LMNTOOL (Local Model Network Toolbox) [5].

Bei der Strukturauswahl für LOLIMOT werden die Regressoren iterativ in einer Vorwärtsselektion ausgewählt und hinzugefügt [6]. Aktuelle Ansätze integrieren die Strukturauswahl direkt in den Schritt zur Partitionierung des Eingangsraumes. Neben der Aufspaltung des schlechtesten Teilmodells wird die Erhöhung der Regressorenanzahl als alternative Option der Strukturanpassung geprüft [7]. Der hier vorgestellte Ansatz nähert sich dem Ziel der Strukturauswahl aus einer anderen Richtung indem er die Pareto-Front aus Modellfehler und Modellkomplexität mit Hilfe eines multikriteriellen evolutionären Algorithmus direkt approximiert.

1.2 Parallele und serienparallele Struktur

Bei der Systemidentifikation wird zwischen zwei Modellstrukturen unterschieden, dem serienparallelen Prädiktionsmodell mit Verwendung des Gleichungsfehlers und dem parallelen Simulationsmodell auf Basis des Ausgangsfehlers. Der wesentliche Vorteil der serienparallelen Struktur ist, dass sich die Parameter des nichtlinearen Modells mit der Methode der

kleinsten Quadrate bestimmen lassen. Wird hingegen direkt für die Simulation der Ausgangsfehler optimiert, muss auf iterative Prozesse zurückgegriffen werden, was einen signifikant höheren Zeitaufwand für das Training bedeutet. Deshalb wird für die evolutionäre Struktursuche das serienparallele Modell trainiert, wodurch sich der Aufwand im Bereich einiger Minuten bis zu einer Stunde bewegt, während die Optimierung der parallelen Variante unter Umständen mehrere Stunden in Anspruch nehmen kann. Das Anwendungsszenario ist in diesem Beitrag immer die Simulation.

Die Identifikation direkt in der parallelen Struktur ist aufwändig, ermöglicht jedoch auch eine höhere Güte. Als Kompromiss aus Rechenaufwand und Modellgüte wird eine zweistufige Identifikation vorgeschlagen. In der ersten Stufe wird die Struktur anhand des serienparallelen Modells numerisch effizient bestimmt, anschließend wird die reduzierte Menge der pareto-optimalen Kandidatenstrukturen nochmals anhand des Ausgangsfehlers optimiert.

1.3 Abbruchbedingungen der Identifikation

Das LMNTOOL bietet eine Vielzahl von Abbruchedingungen für die Verfeinerung der Modellstruktur. Die einfachste Bedingung ist eine Maximalanzahl s an LLMs. Ein Abbruch ist auch möglich, falls keine weitere Reduzierung des Modellfehlers auf den Validierungsdaten zu erwarten ist. Dies ist der Fall bei einer Zunahme des Fehlers auf Validierungsdaten über mehr als n Schritte und eine Verschlechterung des AIC-Maßes [8] über m Iterationen hinweg. Dabei sind n und m einstellbare Parameter. Das AIC-Maß kombiniert den quadratischen Fehler, mit der Anzahl der vorhandenen Trainingsdaten und der Parameteranzahl, um ein Maß für die optimale Komplexität zu erstellen. Da in den verwendeten Benchmarks Validierungsdaten vorhanden sind, die eine direkte Überprüfung des Generalisierungsfehlers zulassen, wird auf diese Abbruchbedingungen verzichtet.

Beim Test des Generalisierungsfehlers der Simulation zeigt sich, dass bei dynamischen Daten keine generellen Aussagen über die Entwicklung der Modellgüte durch Hinzufügen lokaler LLMs gemacht werden kann. Abbildung 1 zeigt einen solchen Fall, bei welchem der Modell-Fit bei mehr als fünf lokalen Modellen zunächst abnimmt, obwohl komplexere Modellbäume mit 13 und 16 LLMs dieses Kriterium wiederum verbessern.

Um keine Modelle höherer Komplexität vorzeitig auszuschließen, wird nur eine maximalen Anzahl von LLMs als Abbruchbedingung angewendet.

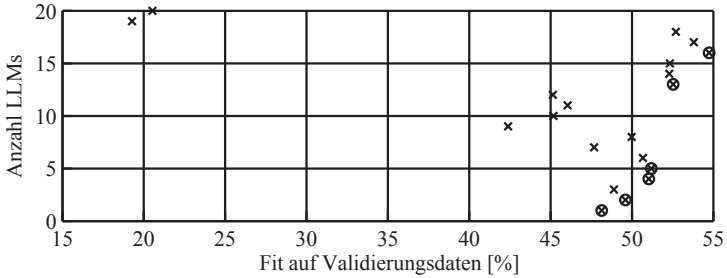


Bild 1: Ergebnisse eines einzelnen Parametersatzes. Verwendet wird der Datensatz *Robot Arm*. Nicht dominierte Lösungen sind durch Kreise markiert.

2 Evolutionärer Algorithmus

Der hier vorgestellte evolutionäre, multikriterielle Algorithmus basiert auf dem bekannten NSGAII [9], dieser wird jedoch in weitreichender Art und Weise für das Problem der Strukturidentifikation angepasst und ergänzt.

2.1 Ganzzahlige Parameter

Die zentralen Strukturparameter im Kontext von LOLIMOT für dynamische Systeme sind n_a und n_b, die Länge der Totzeit n_k und die Anzahl der LLMs. Der Ansatz Regressoren einzeln auszuwählen, wurde hier explizit nicht gewählt, da für die Analyse von Pol- und Nullstellen die Übertragungsfunktion in der faktorisierten Form benötigt wird. Somit haben nur die Regressoren mit der höchsten Ordnung Einfluss auf die Anzahl der Pol- und Nullstellen. Das Auslassen einzelner Regressoren in der polynominalen Darstellung bietet keine Vorteile.

Beispiele für die Optimierung von ganzzahligen Problemen mit Evolutionären Algorithmen sind in der Literatur selten. Klassisch arbeiten Genetische Algorithmen mit einer binären Codierung, während Evolutionsstrategien insbesondere für reellwertige Optimierungsprobleme konzipiert sind. Ganzzahlige nichtlineare Optimierungen werden häufig mit Heuristiken wie *Hill climbing*, *Simulated annealing* oder *Ant colony optimization* gelöst. Die Motivation für die Anwendung evolutionärer Algorithmen liegt in ihrer Fähigkeit zur multikriteriellen Optimierung und ihrer Robustheit gegenüber unstetigen Fitnesslandschaften. (Abbildung 3 zeigt eine teils deutliche Diskontinuität der Parameter, bei gleichzeitiger Nachbarschaft im Kriterienraum.)

Geeignete Operatoren für die evolutionäre Optimierung von ganzzahligen Parametern finden sich in [10]. Eine neuere Publikation [11] schlägt die Verwendung einer binären Parameterkodierung vor. Bei Tests haben sich eine uniforme binäre Rekombination und eine binäre Mutation als geeignet erwiesen.

2.2 Optimierungsablauf

Im Prinzip lassen sich die vier ganzzahligen Parameter n_a, n_b, n_k und die Anzahl der LLMs M direkt optimieren. Aufgrund des iterativen Identifikationsansatzes des LOLIMOT-Algorithmus ist diese Vorgehensweise jedoch ineffizient. Wird ein Modell mit einer gewählten Anzahl an Teilungen identifiziert, ergeben sich während dieses Prozesses automatisch alle Vorgängermodelle im Modellbaum. Dies legt nahe, M nicht als Optimierungsparameter einzubeziehen. Bild 2 zeigt die vorgeschlagene Abwandlung des Algorithmus. Begonnen wird jede Iteration mit einer Elternpopulation. Diese ist in der Abbildung oben mittig als eine Gruppe grau hinterlegter Parametervektoren dargestellt. Jedes Individuum entspricht einer Spalte. Die Abbildung stellt den Ablauf vereinfacht dar. In einer realen Optimierung werden typischerweise deutlich mehr als die vier dargestellten Lösungen verwendet.

Im ersten Schritt werden durch die Mutations- und Rekombinationsoperatoren neue Varianten erzeugt und in ihrer Güte bewertet. Im Kontext der LOLIMOT-Identifikation bedeutet dies, dass jede in der Population kodierte Modellkonfiguration, wenn möglich, bis zu einem vordefinierten Maximum s an Teilungen identifiziert wird. In speziellen Konfigurationen kann der Identifikationsprozess auch bei einer geringeren Anzahl von lokalen Modellen zum Abbruch gezwungen sein, beispielsweise für den Fall das ein Gültigkeitsbereich nicht genügend Datenpunkte enthält, um ein lokales Modell zu identifizieren.

Es entstehen basierend auf einem Parametersatz nicht eine, sondern eine Menge von bis zu s Lösungen. Jede weitere Teilung verbessert das Gesamtmodell, erhöht jedoch gleichzeitig die Anzahl an Parametern. Die Modelle, die aus einem Individuum hervorgehen, dominieren einander nicht, bezogen auf die Güten die mit den Trainingsdaten erzielt werden. Alle in der aktuellen Generation identifizierten Modelle werden in die *erweiterte Nachkommen-Population* gespeichert, die, sofern alle Teilungen durchgeführt werden können, s mal so viele Modelle wie die durch Rekombination und Mutation entstandene Population enthält.

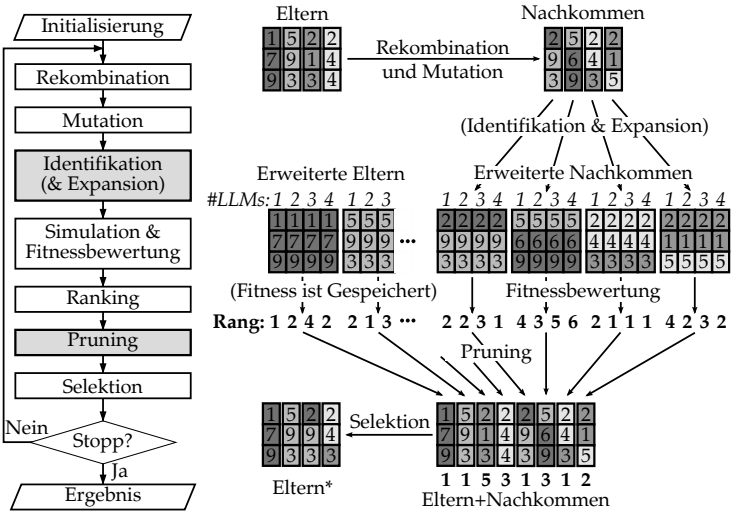


Bild 2: Ablauf des Evolutionären Algorithmus; Links: Flussdiagramm; Rechts: Darstellung der Populationsentwicklung an einem vereinfachten Beispiel mit 4+4-Strategie und $s = 4$.

Nach der Identifikation wird auf Basis der Verifikationsdaten für jedes Modell der erweiterten Nachkommen eine Simulation durchgeführt und der Modellausgang $\underline{\hat{y}}$ mit den gemessenen Daten $\underline{y}$ verglichen.

Das verwendete Kriterium ist der Modell-Fit nach Gleichung (4). Der Wert ist normalisiert, und gibt den Prozentsatz der vom Modell erklärten Standardabweichung der Ausgangsgröße an. 100 % bedeutet eine perfekte Übereinstimmung zwischen Simulation und Prozess, 0 % entspricht der Güte eines Modells das nur den statischen Mittelwert der Ausgangsdaten $\bar{y}$ vorhersagt. Diese Analyse ermöglicht eine intuitive Bewertung der Modellgüte.

$$\mathrm{Fit} \;=\; 100\,\% \left(1 - \frac{\left\|\underline{y} - \underline{\hat{y}}\right\|_2}{\left\|\underline{y} - \bar{y}\right\|_2}\right) \tag{4}$$

Die Definition der Komplexität einer Modellstruktur ist durchaus subjektiv. Üblicherweise dient, wie in diesem Beitrag, die Anzahl der frei einstellbaren Modellparameter als Komplexitätsmaß.

Die erweiterte Population wird mit der erweiterten Eltern-Population vereinigt (Plus-Strategie) und nach den bekannten Methoden multikriteriell

bewertet. Alle nicht-dominierten Modelle erhalten Rang eins und werden aus der Bewertungspopulation entfernt. Die nicht-dominierten Lösungen in der verbleibenden Untermenge erhalten Rang zwei und so weiter. Da in dieser Population Individuen s-fach auftreten können, wird diese nicht direkt als Selektions-Pool verwendet. Dies würde zu Rekombinationen von Lösungen mit denselben Parametern führen und im schlechtesten Fall den Algorithmus vorzeitig stagnieren lassen, da die Diversität der Eltern-Population verloren geht. Stattdessen werden die Populationen auf die ursprüngliche Größe reduziert. Als finaler Rang wird der jeweils beste auftretende Wert verwendet. In Abbildung 2 ist dieser als *Pruning* bezeichnete Schritt rechts unten dargestellt.

2.3 Modifikationen am Algorithmus

Die Gesamtanzahl von Modellidentifikationen ist durch die Rechenzeit limitiert. Zusätzlich ist der diskrete Suchraum relativ klein. Somit lässt sich eine Liste von bewerteten Parameterkombinationen führen. Bereits ausgewertete Lösungen werden keiner erneuten Fitnessbewertung unterzogen und aus der Nachkommenpopulation entfernt. Dieses Vorgehen ist gerechtfertigt, solange die Berechnungszeiten, die für das Anlegen und Abgleichen mit dieser Liste über den gesamten Optimierungslauf, geringer sind, als die redundante Mehrfachbewertung einzelner Parametersätze. Der Aufwand des Abgleichs nimmt aufgrund der wachsenden Liste über die Optimierung hinweg konstant zu. Allerdings steigt während der Optimierung auch die Quote an Duplikaten von Parametersätzen kontinuierlich an, da in der späten Phase stärker lokal gesucht wird.

Die Eliminierung bereits bewerteter Lösungen führt im Extremfall dazu, dass eine gesamte Nachkommenpopulation entfernt wird. Tritt dies über mehrere Generationen in Folge ein, wird der Algorithmus abgebrochen. Somit ergibt sich ein zusätzliches Stopp-Kriterium.

Wird eine Strategie verwendet, die keine Wieder-Auswertungen zulässt, sollten Strategievarianten genutzt werden, die den Verlust von Lösungen verhindern. In der Praxis betrifft dies zwei Aspekte. Benötigt wird ein Elite-Archiv aller nicht dominierten Lösungen. Durch die Parameteranzahl als Optimalitätskriterium ist die Größe dieses Archivs inhärent begrenzt. Die Selektion erfolgt nicht länger auf Basis einer paarweisen Turnier-Strategie, da hier bei einer ungünstigen Pärchen-Bildung nicht-dominierter Lösungen exkludiert werden, während dominierte Individuen selektiert werden. Eine Sortierung anhand des Ranges bildet das primäre Selektionskriterium. Solange die Anzahl der nicht dominierten Parametersätze die

Populationsgröße nicht überschreitet, bleiben bei der angewendeten Plus-Strategie alle Elite-Individuen erhalten.

Standardalgorithmen verwenden ein sekundäres Selektionskriterium, das zum Einsatz kommt, wenn zwischen Individuen gleichen Ranges diskriminiert werden muss. Dieses Kriterium steuert die Verteilung der Lösungen auf der Pareto-Front. Zum Beispiel die *crowding distance* beim NSGAII. Die Verteilung stellt bei dem hier vorgestellten Algorithmus kein Problem dar, da die Parameteranzahl als Kriterium die Front diskretisiert. Berücksichtigt wird stattdessen, wie viele Ausprägungen eines Individuums derzeit in der Front vertreten sind. Lösungen, die mehrfach auftreten, werden gegenüber selteneren oder einmalig auftretenden Lösungen bevorzugt.

3 Auswertung

Im Detail analysiert sind die Ergebnisse für den Datensatz *Heat Exchanger* aus [12], der auch Teil der DAISY-Benchmarksammlung [13] ist. In Abbildung 3 ist eine typische Menge finaler nicht dominierter Lösungen dargestellt. In den 20 Generationen der 20+20-Strategie werden 400 Modellstrukturen ausgewertet.

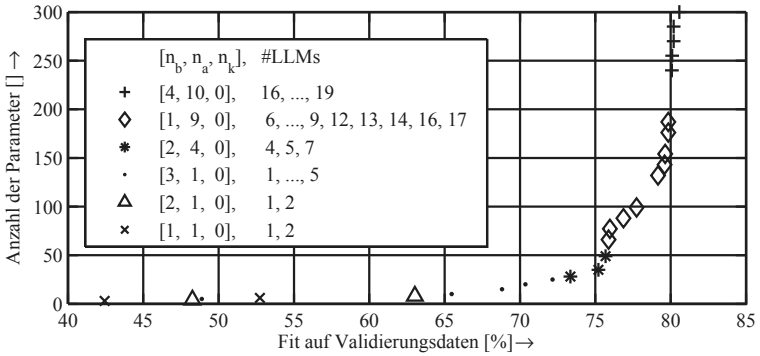


Bild 3: Finale Elite-Population einer LOLIMOT-Optimierung auf dem Datensatz *Heart Exchanger*.

Die finale Population enthält nur sechs verschiedene Parametersätze. Alle sind mehrfach mit verschiedenen LLM-Anzahlen in der Menge vertreten. Insgesamt enthält die erweiterte Elite-Population 26 Individuen. Unten links liegen die kompaktesten Modelle, die aufgrund ihrer geringen Parameteranzahl nicht dominiert sind. Die Ergebnisanalyse lässt Rückschlüs-

se auf den zugrundeliegenden Prozess zu. Ungeteilte Modelle mit ARX-Struktur erreichen maximal 65,5 %, ein Hinweis auf eine deutlich ausgeprägte Nichtlinearität. Mit zunehmender LLM-Anzahl steigt der Modell-Fit deutlich auf 76 %, dabei wird ein Modell mit sechs Regressoren verwendet. Die folgenden, komplexeren Modelle verbessern kontinuierlich die Güte. Der Parametersatz mit den meisten Regressoren und LLMs verbessert die Güte nur minimal. Die multikriterielle Optimierung erlaubt dem Nutzer eine qualifizierte Abwägung zwischen Komplexität und Fit-Wert. Im Kontext der Reglerentwicklung liegen die geeigneten Modelle um 75 %. Hier ist die Komplexität mit etwa sechs LLMs noch gut handhabbar und die Qualität angemessen. Ein gutes Simulationsmodell ist zum Beispiel das [1,9,0]-Modell mit 14 LLMs, da ab hier die Güte weitestgehend saturiert.

Durch die kontinuierliche Selektion der Modelle in der evolutionären Optimierung anhand der Validierungsdaten, wird der Datensatz exzessiv im Gesamtprozess verwendet. Es ist zu erwarten, dass sich somit im Laufe der Optimierung eine Überanpassung auf diesen Datensatz einstellt. Idealerweise steht nach der Strukturauswahl, auf Basis der Validierungsdaten ein dritter, ungesehener Datensatz zu Verfügung, der präzise Aussagen über den zu erwartenden Validierungsfehler zulässt.

Die Pfeile in Abbildung 4 geben die Veränderung der Modellgüte an. Erwartungsgemäß stellt sich im Schnitt eine Verschlechterung des Fits ein. Der Mittelwert des Fit-Rückgangs beträgt -1,48 %, ein moderater Wert. Auch der größte Güteverlust von -3,18 % schränkt die Funktionalität des Simulationsmodells nicht ein. Die Auswertung für die beiden anderen Modellklassen, die aus Gründen der Übersichtlichkeit nicht in der Abbildung eingetragen sind, zeigen qualitativ ähnliche Werte.

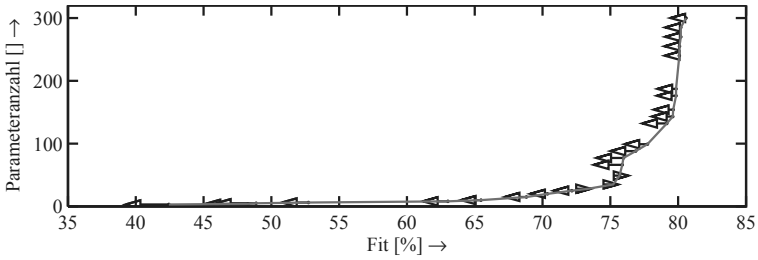


Bild 4: Überprüfung der Generalisierungsfähigkeit. Verglichen sind Validierungsdaten und ungesehenen Testdaten von LOLIMOT für den Datensatz *Heat Exchanger*.

4 Vergleich verschiedener Modelltypen

Um die erzielten Ergebnisse in einen Kontext zu setzen, werden die Optimierungsergebnisse mit denen anderer etablierter datenbasierter Modelle verglichen. Verwendet werden extern rekurrente MLP-Netze und Hammerstein-Wiener-Modelle. Wie die LOLIMOT Modelle werden sie in ihrer serienparellelen Form trainiert.

Der verwendete Trainingsalgorithmus für die Neuronalen Netze (NN) ist der Levenberg-Marquardt-Ansatz. Ein sinnvoller Vergleich zwischen der vorgestellten LOLIMOT-Strukturoptimierung und den Neuronalen-Netzen ist nur möglich, wenn auch hier eine nichtdominierte Population aus Modellgüte und Komplexität entwickelt wird.

Hierzu werden:

- die Anzahl der verzögerten Eingänge $n_b \in \{1, 2, \ldots, 32\}$,
- die Anzahl der verzögerten Ausgänge $n_a \in \{0, 1, \ldots, 31\}$,
- die Totzeit mit $n_k \in \{0, 1, \ldots, 31\}$ und
- die Anzahl der versteckten Neuronen optimiert.

Die so erzielte Regressor-Auswahl erfolgt analog zu der für LOLIMOT. Der Initialwert der iterativen Optimierung wird für alle Netze zufällig erzeugt, was zu einem nicht deterministischen Optimierungsverlauf führt. Die erzielten Ergebnisse schwanken bei wiederholten Parameteridentifikationen drastisch. Um den Effekt zufällig ungünstiger Initialisierungen abzumildern, wird das Training jeweils fünffach wiederholt und das beste Ergebnis verwendet. Die Anzahl der Neuronen wird nicht direkt optimiert, da die plausible Neuronenanzahl der verstecken Schicht von der Anzahl der Eingangsneuronen abhängig ist. Die Anzahl der Neuronen nimmt abhängig vom entsprechenden Optimierungsparameter das $0{,}5$ bis 2-fache der Eingangsanzahl ein. Ein Modell mit 10 Neuronen in der Eingangsschicht hat dementsprechend zwischen 5 und 20 Neuronen in der versteckten Schicht. Der Wertebereich des Optimierungsparameters $\{1, 2, \ldots, 16\}$ wird auf den Wertebereich der Neuronen abgebildet und gerundet.

Zum Vergleich wird auch eine Wiener-Hammerstein-Struktur (WH) verwendet. Diese besteht aus zwei stückweise linearen Kennlinien, die vor und hinter einem linearen Output-Error-Modell liegen. Optimiert werden,

analog zu den anderen beiden Modellstrukturen, die Modelleingänge sowie zusätzlich die Anzahl der Stützstellen der nicht linearen Kennlinien. Jede Stützstelle trägt zwei Parameter zum Gesamtmodell bei.

Da die einzelnen Fitnessbewertungen jeweils eine aufwändige Parameteridentifikation durchführen und der Suchraum relativ klein ist, wird mit einem sehr kleinen Budget an Fitnessbewertungen gearbeitet. Tabelle 1 stellt die drei zu vergleichenden Konfigurationen dar.

Tabelle 1: Konfiguration der Optimierungen

	LOLIMOT	NN	WH
Parameteranzahl	3	4	5
Strategie	$20 + 20$	$20 + 20$	$20 + 20$
Generationen	20	20	100
Identifizierte Modelle	bis zu $400 \cdot 20 = 8000$	$400 \cdot 5 = 2000$	2000

Das theoretische Limit s von 20 lokalen Teilmodellen wird dabei nicht immer ausgeschöpft. Insbesondere die Prozesse mit wenigen Daten, wie *Ball & Beam* und *Dryer*, sind nur begrenzt teilbar, da oft nicht genügend Datenpunkte zur Schätzung der LLMs verbleiben.

Angewendet werden die Verfahren auf die in [14] empfohlenen Benchmark-Daten aus der DAISY-Sammlung [13]. Verglichen werden die Ergebnisse für sechs SISO-Datensätze:

- MR-Damper (800 Datenpunkte für Training & 224 für Validierung),
- Flutter (1024 Datenpunkte für Training),
- Ball & Beam (500 Datenpunkte für Training & 500 für Validierung),
- Robot Arm (800 Datenpunkte für Training 224 & für Validierung),
- Heat Exchanger (2000 Datenpunkte für Training, 1000 für Validierung & 1000 als Testdatensatz) und
- Dryer (500 Datenpunkte für Training & 500 für Validierung).

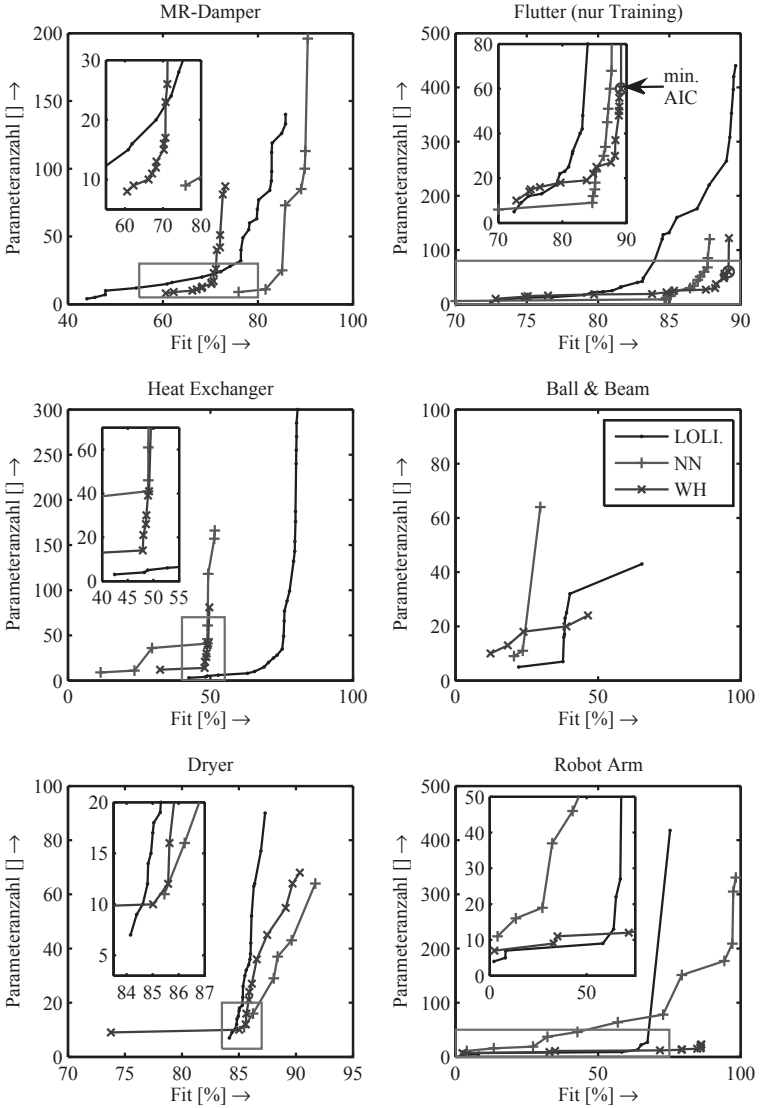


Bild 5: Vergleich der Mengen nicht-dominierter Lösungen für LOLIMOT, extern rekurrenten Neuronalen Netzen und Wiener-Hammerstein-Modellen. Dargestellt ist die Anzahl der freien Modellparameter über den Model-Fit auf Validierungsdaten.

Die ersten fünf Datensätze stammen aus der DAISY-Sammlung [13]. Der Datensatz *Flutter* einhält nur eine einzige Anregung des Systems, deshalb ist eine Aufteilung der Daten für Training und Validierung nicht möglich. Der sechste Datensatz, *Dryer*, wird von Ljung in mehreren Veröffentlichungen verwendet [15, 16] und ist Teil der Matlab System-Identification-Toolbox. Bei der Analyse der einzelnen Darstellungen wird deutlich, dass keine allgemein gültigen Aussagen möglich sind, sondern die Ergebnisse stark zwischen den Datensätzen variieren. LOLIMOT ist für die Datensätze *Heat Exhanger* und *Ball & Beam* besonders geeignet, und dominiert die anderen beiden Ansätze.

Die Modellkomplexität über die Metrik der Parameteranzahl zu messen, ist ein einfaches, nachvollziehbares und in der Literatur häufig verwendetes Prinzip. Es ist jedoch nur bedingt geeignet, um die vom Anwender „subjektive bewertete“ Komplexität abzubilden. Neben einer stark subjektiven Komponente sind die Modelle inhärent unterschiedlich gut interpretierbar. Die linearen Teilmodelle von LOLIMOT lassen sich beispielsweise mit vielen Hilfsmitteln der Systemtheorie untersuchen und erlauben standardisierte Methoden für die Reglersynthese. Aussagen über das Verhalten eines Neuronalen Netzes, selbst mit weniger Parametern, sind deutlich schwieriger. Somit wird möglicherweise ein Modell präferiert, obwohl es hinsichtlich der objektiven Kriterien von anderen, für die Anwendung jedoch ungeeigneten Modellklassen, dominiert wird.

5 Zusammenfassung

Der vorliegende Beitrag stellt einen multikriteriellen, evolutionären Algorithmus zur Optimierung von Strukturparametern für LOLIMOT Systeme vor.

Die multikriterielle Optimierung ermöglicht die Ermittlung pareto-optimaler Kompromisslösungen zwischen Modellgüte und der Parameteranzahl. Es sind zahlreiche Modifikationen am evolutionären Algorithmus implementiert. Durch die Baumstruktur von LOLIMOT entsteht aus einem Parametersatz nicht ein einzelnes Modell, sondern eine Vielzahl von Modellen, da jede Baumtiefe als eigenständig betrachtet werden kann. Die Kriterien werden in erweiterten Populationen gespeichert und multikriteriell bewertet. Die finale Menge an Lösungen ermöglicht dem Nutzer eine objektive Abwägung aller Modellvariationen. Verglichen wird der Ansatz mit extern verzögerten rekurrenten MLP-Netzen und Wiener-Hammerstein-Modellen. Auch diese Strukturen werden multikriteriell optimiert. Der Ver-

gleich anhand mehrerer Benchmark-Datensätze zeigt deutlich, dass die Wahl der Modellstruktur individuell für jedes System getroffen werden muss. Gerade hier ist die a posteriori Entscheidung auf Basis von Pareto-Fronten ein geeignetes Hilfsmittel.

Für die Datensätze *Heat Exhanger* und *Ball & Beam* ist LOLIMOT der im Vergleich beste Ansatz. Die Strukturoptimierung bezieht sich in diesem Beitrag speziell auf die Auswahl der Regressoren. Der Ansatz kann problemlos auf weitere (Meta-)Parameter der Modellstruktur ausgeweitet werden. Auch die Optimierung anderer Modellklassen wie nicht-lineare ARX-Modelle oder HILOMOT ist möglich.

Literatur

[1] Geman, S.; Bienenstock, E.; Doursat, R.: Neural networks and the bias/variance dilemma. *Neural computation* 4 (1992) 1, S. 1–58.

[2] Mendes, E. M.; Billings, S. A.: An alternative solution to the model structure selection problem. *IEEE Transactions on Systems Man and Cybernetics Part A: Systems and Humans* 31 (2001) 6, S. 597–608.

[3] Karnin, E. D.: A simple procedure for pruning back-propagation trained neural networks. *Neural Networks, IEEE Transactions on* 1 (1990) 2, S. 239–242.

[4] Nelles, O.: *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Springer. ISBN 9783642086748. 2010.

[5] Hartmann, B.; Ebert, T.; Fischer, T.; Belz, J.; Kampmann, G.; Nelles, O.: LMNTOOL-Toolbox zum automatische Trainieren lokaler Modellnetze. In: *Proceedings of the 22. Workshop Computational Intelligence (Hoffmann, F.; Hüllermeier, E., Hg.), S*, S. 341–355. 2012.

[6] Belz, J.; Nelles, O.: Lokale Modellnetze zur Selektion von Einflussgrößen. In: *Proceedings. 23. Workshop Computational Intelligence, Dortmund, 5.-6. Dezember 2013*, S. 265–280. KIT Scientific Publishing. 2013.

[7] Hartmann, B.; Nelles, O.: Identifikation mit achsenschrägen, lokal polynomialen Modellnetzen. *at-Automatisierungstechnik* 62 (2014) 6, S. 394–407.

[8] Akaike, H.: A new look at the statistical model identification. *Automatic Control, IEEE Transactions on* 19 (1974) 6, S. 716–723.

[9] Kalyanmoy Deb, Samir Agrawal, A. P. T. M.: A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. 2000.

[10] Rudolph, G.: An evolutionary algorithm for integer programming. In: *Parallel Problem Solving from Nature - PPSN III* (Davidor, Y.; Schwefel, H.-P.; Männer, R., Hg.), Bd. 866 von *Lecture Notes in Computer Science*, S. 139–148. Springer Berlin Heidelberg. ISBN 978-3-540-58484-1. 1994.

[11] Huy, P. N. A.; San, C. T. B.; Triantaphyllou, E.: Solving integer programming problems using genetic algorithms. In: *ICEIC: International Conference on Electronics, Informations and Commumications*, S. 400–404. 2004.

[12] Bittanti, S.; Piroddi, L.: Nonlinear identification and control of a heat exchanger: A neural network approach. *Journal of the Franklin Institute* 334 (1997) 1, S. 135 – 153.

[13] De Moor, B.; De Gersem, P.; De Schutter, B.; Favoreel, W.: DAISY: A database for identification of systems. *Journal A, Special Issue on CACSD (Computer Aided Control Systems Design)* 38 (1997) 3, S. 4–5.

[14] Hoffmann, F.; Mikut, R.; Kroll, A.; Reischl, M.; Nelles, O.; Schulte, H.; Bertram, T.: Computational Intelligence: State-of-the-Art Methoden und Benchmarkprobleme. In: *22th Workshop Computational Intelligence*, Bd. 45, S. 1–42. Universitätsverlag Karlsruhe. ISBN 978-3-86644-917-6. 2012.

[15] Ljung, L.: *System identification: theory for the user*. Prentice-Hall information and system sciences series. Prentice-Hall. ISBN 9780138816407. 1987.

[16] Ljung, L.: *System Identification Toolbox for Use with MATLAB: User's Guide*. MathWorks, Incorporated. 1991.

Instance-Based versus Rule-based Evolving Fuzzy Systems

Ammar Shaker, Eyke Hüllermeier

Department of Computer Science
University of Paderborn, Germany
{ammar.shaker,eyke}@upb.de

Abstract

In this paper, we a compare an instance-based and a fuzzy rule-based method for learning on data streams, specifically focusing on their ability to handle so-called concept drift. To this end, we make use of recovery analysis, an experimental procedure that we recently developed for exactly this purpose. Our results provide first evidence in favor of the conjecture that instance-based approaches to learning on data streams are not only competitive to model-based methods in terms of performance, but also advantageous with regard to the handling of concept drift.

1 Introduction

The idea of adaptive learning in dynamical environments has recently received increasing attention in different research communities, including the data mining community, in which it has been addressed under the notion of "learning from data streams" [5, 6], and the computational intelligence community, in which the notion of "evolving fuzzy systems" has been coined [2, 7, 1, 8]. Despite small differences regarding the basic assumptions and the emphasis of goals, the key motivation of these and related fields is the idea of a system that learns incrementally, and maybe even in real-time, on a continuous stream of data, and which is able to properly adapt itself to so-called *concept change*, that is, changes of environmental conditions or properties of the data-generating process.

Evolving fuzzy systems (EFS) are almost exclusively realized in the form of *rule-based* systems. More generally, most approaches to adaptive learning with evolving systems, also in other fields, are *model-based* in the sense that a (sometimes complex) model is learned from the streaming data

and adapted in the course of time. In fact, while a lot of effort has been invested in the development of such approaches, instance-based learning (IBL) on data streams has not received much attention so far, with only a few notable exceptions [3, 9]. This is arguably surprising, especially in light of some appealing properties of IBL. Obviously, IBL algorithms are inherently incremental, since adaptation basically comes down to editing the current case base (which essentially represents the model) by either adding or removing observed cases. Thus, incremental learning and model adaptation is simple and cheap in the case of IBL. As opposed to this, incremental learning is much more difficult to realize for most model-based approaches, since the incremental update of a model, such as a classification tree or a fuzzy system, is often quite complex and in many cases assumes the storage of a considerable amount of additional information.

The goal of this paper is to compare rule-based fuzzy systems with instance-based methods in a regression setting. To this end, we select a (state-of-the-art) representative for each of the approaches and make use of so-called *recovery analysis*, an experimental procedure that we recently developed [10]. Recovery analysis aims at assessing the ability of a learner to discover a concept change quickly, and to take appropriate measures to maintain the quality and generalization performance of the model. We briefly recall the basic ideas of recovery analysis in the next section, prior to presenting our experiments and results in Sections 3 and 4, respectively.

2 Recovery analysis

Recovery analysis works with three streams of data points $\boldsymbol{z} \in \mathbb{Z}$ in parallel, two "pure streams" and one "mixture". The pure streams

$$S_A = (\boldsymbol{z}_1^a, \boldsymbol{z}_2^a, \ldots, \boldsymbol{z}_T^a)$$
$$S_B = (\boldsymbol{z}_1^b, \boldsymbol{z}_2^b, \ldots, \boldsymbol{z}_T^b)$$

are supposed to be stationary and generated, respectively, according to probability distributions $\mathbf{P}_A$ and $\mathbf{P}_B$; in the case of real data, stationarity of a stream can be guaranteed, for example, by permuting the original

stream at random. These two streams must also be compatible in the sense of sharing a common data space $\mathbb{Z} = \mathbb{X} \times \mathbb{Y}$.

A third stream $S_C = (\boldsymbol{z}_1^c, \boldsymbol{z}_2^c, \ldots, \boldsymbol{z}_T^c)$, called mixture stream, is produced by randomly sampling from the two pure streams:

$$\boldsymbol{z}_t^c = \begin{cases} \boldsymbol{z}_t^a & \text{with probability } \lambda(t) \\ \boldsymbol{z}_t^b & \text{with probability } 1 - \lambda(t) \end{cases} \tag{1}$$

A concept drift can then be modeled, for example, by specifying the (time-dependent) sample probability $\lambda(t)$ as a sigmoidal function:

$$\lambda\left(t\right) = \left(1 + \exp\left(\frac{4(t - t_0)}{w}\right)\right)^{-1} . \tag{2}$$

This function has two parameters: t_0 is the mid point of the change process, while w controls the length of this process. Using this transition function, the stream S_C is obviously drifting "from S_A to S_B": In the beginning, it is essentially identical to S_A, in a certain time window around t_0, it moves away from S_A toward S_B, and in the end, it is essentially identical to S_B. Thus, a gradual concept drift is created, with a rate of change controlled by w.

Now, suppose the same learning algorithm $\mathcal{A}$ is applied to all three streams S_A, S_B and S_C. Since the first two streams are stationary, we expect to see a standard learning curve when plotting the generalization error (for example, root mean squared error) as a function of time. In the following, we denote the error curves for S_A and S_B by $\alpha(t)$ and $\beta(t)$, respectively. These curves are normally concave, showing a significant increase in the beginning before reaching a certain saturation level later on; see Figure 1 for an illustration. The corresponding saturation levels α^* and β^* provide important information, namely information about the best performance that can be expected by the learner $\mathcal{A}$ on the pure streams S_A and S_B, respectively.

Even more interesting, however, is the performance curve $\gamma(t)$ for the stream S_C, which exhibits concept drift. In the beginning, this curve will be effectively identical to the curve for S_A, so that the learner $\mathcal{A}$ should reach the level α^*. Then, upon the beginning of the concept drift, the performance

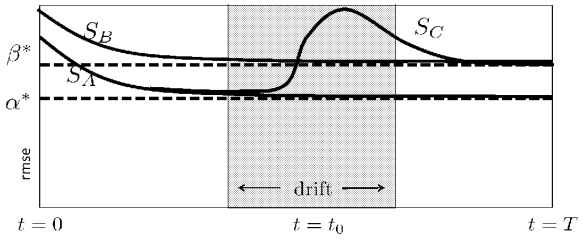


Figure 1: Schematic illustration of a recovery analysis: The three performance curves are produced by training models on the pure streams S_A and S_B, as well as on the mixed stream S_C, each time using the same learner $\mathcal{A}$. The region shaded in grey indicates the time window in which the concept drift (mainly) takes place. While the concept is drifting, the performance on S_C will typically drop to some extent. This can be seen by the increase of the prediction error.

is expected to drop, and this decrease is supposed to continue until the drift ends and the learner $\mathcal{A}$ starts to recover. Eventually, $\mathcal{A}$ may (or may not) reach the level β^*. This level is indeed a lower bound on the asymptotic error, since $\mathcal{A}$ cannot do better even when being trained on S_B from the very beginning. Thus, reaching this level indicates an optimal recovery.

Obviously, the performance curve for S_C provides important information about the ability of $\mathcal{A}$ to deal with concept drift. In particular, the minimum of this curve indicates how strongly $\mathcal{A}$ is affected by the concept drift. Moreover, the curve informs about how quickly the performance deteriorates (giving an idea of how sensitive $\mathcal{A}$ is), how much time $\mathcal{A}$ needs to recover, and whether or not it manages to recover optimally.

3 Experiments and results

The remainder of the paper is devoted to a case study, in which we compare the instance-based learner IBLStreams [9] with the fuzzy rule-based system FLEXFIS [7] with respect to their ability to handle concept drift. A main goal of this study is to see whether there are important differences between the methods, and whether recovery analysis helps uncover them.

We conducted experiments with three different drift settings. To this end, we varied the speed of change by modifying the width parameter w in the sigmoid function (2). More specifically, we control the angle θ of the tangent of this function at $t = t_0$, which is inversely proportional to w, i.e., $\tan(\theta) = \frac{1}{w}$. Figure 2 depicts the three drift velocities:

- $\theta = \frac{\pi}{75}$ for a slow concept drift,
- $\theta = \frac{\pi}{30}$ for concept drift with a modest speed,
- $\theta = \frac{\pi}{2}$ for a sudden concept change (concept shift).

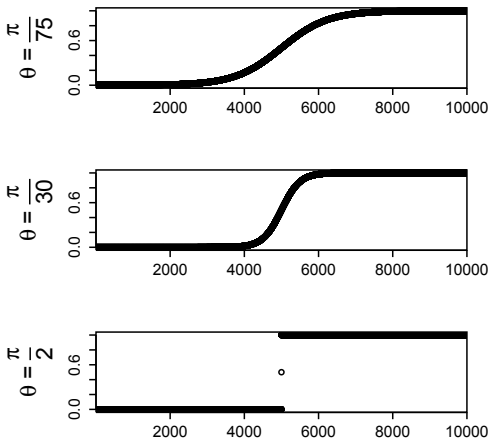


Figure 2: Sigmoid transition function modeling different types of concept drift: slow drift (top), moderate drift (middle), sudden drift (bottom).

In fact, since the data sets do not share the same size, we define the length of the window as $w = \frac{\ell}{100 \tan(\theta)}$, where ℓ is the size of the data

set. Thus, given the drift angle θ, the proportion of the entire stream that is subject to drift is the same for all data sets. The results are generated by plotting the average root mean square error on each data chunk. To this end, we use the test-then-train approach which works in an sample by sample way [4]. In the following, we present results for two data sets; further experiments on other data sets produced quite similar results.

3.1 HyperDistance

This is a synthetic data set that we produced by modifying the Hyperplane-Generator (for classification data) in the MOA tool [4] as follows: The output for an instance $\boldsymbol{x}$ is not determined by the sign of $\boldsymbol{w}^\top \boldsymbol{x}$, where $\boldsymbol{w}$ is the normal vector of the hyperplane, but by the absolute value $y = f_1(\boldsymbol{x}) = |\boldsymbol{w}^\top \boldsymbol{x}|$. In other words, the problem is to predict the distance of $\boldsymbol{x}$ to the hyperplane. As an alternative, we also used the cubic distance $y = f_3(\boldsymbol{x}) = (\boldsymbol{w}^\top \boldsymbol{x})^3$, which is arguably more difficult to learn than the absolute distance. In this experiment, we generated S_A using f_1 and S_B using f_3; thus, the drift is from the simpler to the more difficult problem. This stream contains 125k 4-dimensional data samples.

Figure 3 shows the recovery curves of the learning methods. As can be seen, both IBLStreams and FLEXFIS have a relatively small error on the first problem. During the drift, both suffer from a high drop in performance, which is visible as a bell-shaped peak. Nonetheless, IBLStreams recovers quite well, whereas FLEXFIS never recovers.

3.2 Census-House

This data was obtained from the DELVE[1] repository, a collection of data sets designed on the basis of the US census data collected in 1999. We use the House8L data, the purpose of which is to predict the median price of houses in different regions based on the demographic properties. Each house has 8 attributes, with a total number of 22784 houses. The second pure stream S_B is created by "inverting" the target values: For an example

[1] http://www.cs.utoronto.ca/~delve/data/datasets.html

$(\boldsymbol{x}, y)$, the original output y is replaced by $y_{max} + y_{min} - y$, where y_{min} and y_{max} are the smallest and largest values in the data set.

By observing the performance during the drifts in Figure 4, we notice that the more rapid the change is, the sharper the drop in the performance of FLEXFIS becomes, whereas IBLStreams shows smaller errors for quicker drifts. By the end of the stream, both approaches recover quite well.

4 Conclusion

We have presented an experimental study, in which we used the method of recovery analysis to compare an instance-based with a model-based (rule-based) learning algorithm on data streams. From the results obtained, we can at least extract some trends and draw some preliminary conclusions, which are summarized in the following observations:

- FLEXFIS and IBLStreams are quite strong in terms of absolute accuracy. Nevertheless, they tend to have pronounced peaks (maximum performance loss) in the area of drifts.
- FLEXFIS tends to have problems with adapting to increasingly difficult situations: When the second stream is more complex than the first one, it often fails to recover (see Figure 3).
- IBLStreams tends to be less affected by the suddenness of the drift; compared to FLEXFIS, it exhibits a very small drop in performance in such cases.
- In general, IBLStreams appears to be superior with regard to recovery. In fact, it recovers well in all experiments.

Overall, our results provide (at least preliminary) evidence in favor of our conjecture that instance-based approaches to learning on data streams are not only competitive to model-based approaches in terms of performance, but also advantageous with regard to the handling of concept drift. This is arguably due to their "lightweight" structure: Removing some outdated examples is all the more simpler than completely reconstructing a possibly complex model.

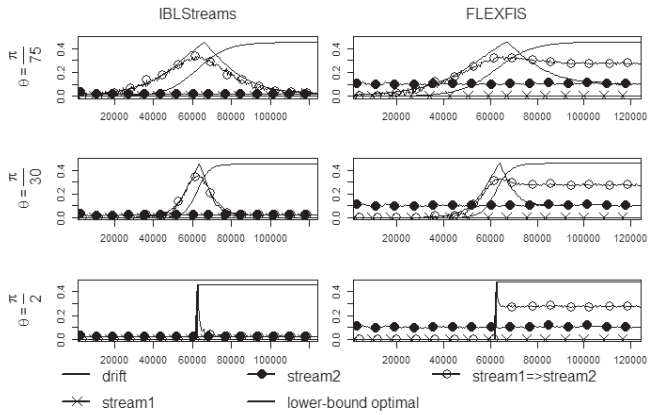


Figure 3: Performance curves (RMSE) on the HyperDistance data, with a drift from f_1 to f_3. The sigmoid curve indicates the range of the drift.

References

[1] Plamen P. Angelov, Dimitar P. Filev, and Nik Kasabov. *Evolving Intelligent Systems*. John Wiley and Sons, New York, 2010.

[2] Plamen P. Angelov, Edwin Lughofer, and Xiaowei Zhou. Evolving fuzzy classifiers using different model architectures. *Fuzzy Sets and Systems*, 159(23):3160–3182, 2008.

[3] Jürgen Beringer and Eyke Hüllermeier. Efficient instance-based learning on data streams. *Intelligent Data Analysis*, 11(6):627–650, 2007.

[4] Albert Bifet and Richard Kirkby. *Massive Online Analysis Manual*, August 2009.

[5] Mohamed Medhat Gaber, Arkady Zaslavsky, and Shonali Krishnaswamy. Mining data streams: A review. *ACM SIGMOD Record, ACM Special Interest Group on Management of Data*, 34(1), 2005.

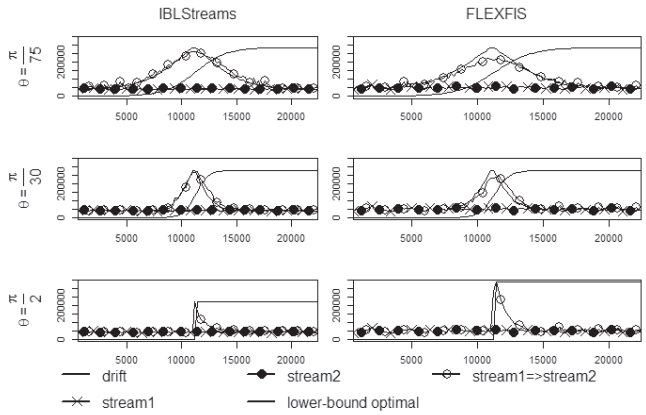


Figure 4: Performance curves (RMSE) on the House8L data. The sigmoid curve indicates the range of the drift.

[6] João Gama and Mohamed Medhat Gaber. *Learning from Data Streams*. Springer-Verlag, Berlin, New York, 2007.

[7] Edwin Lughofer. FLEXFIS: A robust incremental learning approach for evolving Takagi-Sugeno fuzzy models. *IEEE Transactions on Fuzzy Systems*, 16(6):1393–1410, 2008.

[8] Edwin Lughofer. *Evolving Fuzzy Systems: Methodologies, Advanced Concepts and Applications*. Springer-Verlag, Berlin, Heidelberg, 2011.

[9] A. Shaker and E. Hüllermeier. IBLStreams: A system for instance-based classification and regression on data streams. *Evolving Systems*, 3(4):235–249, 2012.

[10] A. Shaker and E. Hüllermeier. Recovery analysis for adaptive learning from non-stationary data streams. In R. Burduk, K. Jackowski,

M. Kurzynski, M. Wozniak, and A. Zolnierek, editors, *Proceedings CORES 2013, 8th International Conference on Computer Recognition Systems*, pages 289–298, Wroclaw, Poland, 2013. Springer-Verlag.

Einfluss der Realisierung von Takagi-Sugeno Modellen auf den Reglerentwurf mit einem Beispiel zur Regelung von Windenergieanlagen

H. Schulte, S. Georg

HTW Berlin, Hochschule für Technik und Wirtschaft,
Ingenieurwissenschaften I, Fachgebiet Regelungstechnik
Email: {schulte,georg}@htw-berlin.de

Kurzfassung

Die Ableitung und Realisierung von Takagi-Sugeno (TS) Modellen aus Vektordifferentialgleichungen ist nicht eindeutig. Fast man dies als zusätzlichen Freiheitsgrad beim Reglerentwurf auf, kann durch eine geeignete Wahl die Anzahl der linearen Matrixungleichungen (LMIs) reduziert und damit das zugehörige konvexe Optimierungsproblem relaxiert werden. Dabei hängt die Lösbarkeit und Grad der Relaxation sowohl von der Struktur und Anzahl der einzelnen linearen Teilmodelle und den gemeinsamen bzw. differierenden Einträgen in den System- und Eingangsmatrizen ab. In dieser Arbeit werden zunächst die verschiedenen Varianten einer TS Modell-Realisierung basierend auf einer gegebenen nichtlinearen Systembeschreibung vorgestellt. Anschließend werden am Beispiel des Entwurfs einer Leistungsregelung für Windenergieanlagen zwei verschiedene Realisierungen bewertet.

1 Einführung

Die Wahl eines realisierten Takagi-Sugeno (TS) Modells kann wie bei linearen-zeitinvarianten (LTI) Systemen als zusätzlicher Freiheitsgrad beim Reglerentwurf aufgefasst werden. Als Erweiterung des linearen Falls werden neben der Modellordnung die Zugehörigkeitsfunktionen und die Struktur der Teilmodelle festgelegt. Kriterien der geeigneten Realisierung sind u.a. die Beobachtbarkeit, Steuerbarkeit, Anzahl der Teilmodelle sowie die Messbarkeit der Prämissenvariablen.
Die vorgestellte Arbeit konzentriert sich auf TS Modelle die direkt aus nichtlinearen Vektordifferentialgleichungen und Ausgangsgleichungen der Form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \,, \qquad \mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u}) \tag{1}$$

abgeleitet werden mit $\mathbf{x} \in \mathbb{R}^n$ als Zustandsvektor, als Systemeingang $\mathbf{u} \in \mathbb{R}^m$ und Systemausgang $\mathbf{y} \in \mathbb{R}^p$. Diese werden entweder durch eine gewichtete Kombination lokal gültiger linearer Zustandsraummodelle gebildet, welche die physikalischen Differentialgleichungen *approximativ* durch eine gewichtete Kombination globaler linearer Zustandsraummodelle repräsentiert, welche das Originalsytem innerhalb von festgelegten Sektorgrenzen exakt beschreiben. Das zuletzt genannte Verfahren wird deshalb auch als Methode der Sektornichtlinearitäten bezeichnet [11]. Zur Unterscheidung der beiden Fälle wird im folgenden von *approximativen TS Modellen* oder von *exakten TS Modellen* gesprochen.

Ein Großteil der bisherigen Arbeiten, die die Ableitung von TS Modellen aus Vektordifferentialgleichungen behandeln, untersucht die Klasse der *approximativen TS Modelle*. Dabei werden Kriterien zur Anzahl und Lage der Punkte der Taylorreihenentwicklung angegeben sowie die Interpretierbarkeit der lokalen Modelle untersucht. Neben der Linearisierung des nichtlinearen Systems um stationäre Arbeitspunkte wird auch der Fall untersucht, bei dem die linearen Teilmodelle transiente Vorgänge außerhalb von stationären Arbeitspunken (off-equilibrium linearisation) beschreiben [4]. Als kontinuierliche Bewertungskriterien werden der zulässige Approximationsfehler [7] sowie die Stabilität, Steuerbarkeit und Beobachtbarkeit der lokalen Teilmodelle eingeführt [5].

Im Gegensatz dazu ist aufgrund der Methode des Sektornichtlinearitäten bei den *exakten TS Modellen* die Lage und Anzahl der linearen Teilmodelle festgelegt. Diese befinden sich immer auf den Sektorgrenzen der einzelnen Nichtlinearitäten, wobei die Anzahl der Teilmodelle N_r sich aus der Anzahl der Nichtlinearitäten N_l mit $N_r = 2^{N_l}$ ergibt. Die Variation der Ableitung folgt aus der Wahl der Prämissenvariablen und der Festlegung der Sektoreinträge in den Teilmodellen. Bisher sind diese Ableitungsaspekt nicht systematisch untersucht worden: Bei den systemtheoretischen Betrachtungen sind die TS Modelle als Fallbeispiele gegeben. Die Ableitung wird hierbei nicht betrachtet. Ebenso wenig bei den anwendungsorientierten Arbeiten, bei denen meist eine Realisierung ohne Angaben von Gründen vorgestellt wird.

Die Arbeit gliedert sich wie folgt: Im folgenden Abschnitt wird die zu untersuchende Klasse der TS Modelle vorgestellt und die Methode der Sektornichtlinearitäten beschrieben. In Abschnitt 3 werden die Variationsmöglichkeiten bei der Anwendung dieser Methode klassifiziert und qualitative Bewertungen und Designregeln zur Ableitung von TS Modellen für den Beobachter- und Reglerentwurf angegeben. Im vierten Abschnitt wird am Beispiel des Enwurfs einer Leistungsregelung für Windenergieanlagen die zuvor eingeführten Designregeln angewandt. Der letzte Abschnitt fast die Ergebnisse zusammen und stellt Erweiterungsmöglichkeiten vor.

2 Ableitung von Takagi-Sugeno Modellen

2.1 Modellbeschreibung

Betrachtet werden zeitkontinuierliche Takagi-Sugeno Modelle mit Teilsystemen in Zustandsraumdarstellung, die im Rahmen der Fuzzy Set Theorie als Fuzzy-Regel basierte Modelle erstmals in [10] eingeführt worden sind. Die ursprüngliche Verwendung von Fuzzy-Regeln mit einer linguistischen Beschreibung ist jedoch für die Ableitung mit Hilfe von Sektorfunktionen ungeeignet. Die Verwendung von Fuzzy-Regeln würde fälschlicherweise suggerieren, dass das TS Modell eine Approximation des Originalsystems ist, obwohl es sich um eine exakte Beschreibung innerhalb der Sektorgrenzen (Gültigkeitssektor) handelt. Deshalb wird in dieser Arbeit die äquivalente defuzzifizierte Form von TS Modellen verwendet:

$$\begin{aligned} \dot{\mathbf{x}} &= \sum_{i=1}^{N_r} h_i(\mathbf{z}) \left(\mathbf{A}_i \, \mathbf{x} + \mathbf{B}_i \, \mathbf{u} \right) , \\ \mathbf{y} &= \sum_{i=1}^{N_r} h_i(\mathbf{z}) \, \mathbf{C}_i \, \mathbf{x} \, , \end{aligned} \tag{2}$$

mit $\mathbf{x} \in \mathbb{R}^n$ als Zustandsvektor, $\mathbf{u} \in \mathbb{R}^m$ als Eingangsvektor und $\mathbf{y} \in \mathbb{R}^p$ als Ausgangsvektor. Die linearen Teilsysteme $i = 1, \ldots, N_r$ werden aus $\mathbf{A}_i \in \mathbb{R}^{n \times n}$, $\mathbf{B}_i \in \mathbb{R}^{n \times m}$ und $\mathbf{C}_i \in \mathbb{R}^{p \times n}$ gebildet. Die nichtlinearen Funktionen $h_i(\mathbf{z}) : \mathbb{R}^l \to \mathbb{R}, \mathbf{z} \mapsto h_i$ gewichten die linearen Teilsysteme im TS Modell zueinander und erfüllen die Konvexitätseigenschaft

$$\sum_{i=1}^{N_r} h_i(\mathbf{z}) = 1 \, , \quad h_i(\mathbf{z}) \geq 0 \quad \forall \, i \in \{1, \ldots, N_r\} \, . \tag{3}$$

Der Vektor $\mathbf{z}$, im Folgenden als Prämissenvektor bezeichnet, setzt sich im Allgemeinen aus Systemzuständen, Systemeingängen und externen Größen zusammen. Verwendet man zur Ableitung die Methode der Sektornichtlinearitäten sind es bezogen auf das Ausgangsmodell (1) die Variablen, die nichtlinear in das mathematische Modell eingehen.

2.2 Methode der Sektornichtlinearitäten (SL)

Bei der Methode der Sektornichtlinearitäten wird das nichtlineare System (1) zunächst in

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{z})\,\mathbf{x} + \mathbf{B}(\mathbf{z})\,\mathbf{u} \tag{4}$$

überführt. Dieser Schritt ist nicht immer eindeutig, wie das folgende mathematische Beispiel zeigt:

Beispiel 1: Gegeben ist das nichtlineare System

$$\begin{aligned} \dot{x}_1 &= -x_1 + 2\,x_2 + 2\,u \\ \dot{x}_2 &= x_1\,x_2 - 3\,x_2 + 3\,u \end{aligned} \tag{5}$$

mit zwei Zuständen $\mathbf{x} = [x_1\,,\,x_2]^T$, einem Eingang u und einem Ausgang y. Dieses kann nun in

$$\dot{\mathbf{x}} = \underbrace{\begin{bmatrix} -1 & 2 \\ z & -3 \end{bmatrix}}_{\mathbf{A}(z)} \mathbf{x} + \underbrace{\begin{bmatrix} 2 \\ 3 \end{bmatrix}}_{\mathbf{B}} u$$

mit $z := x_2$ oder in

$$\dot{\mathbf{x}} = \underbrace{\begin{bmatrix} -1 & 2 \\ 0 & (z-3) \end{bmatrix}}_{\mathbf{A}(z)} \mathbf{x} + \underbrace{\begin{bmatrix} 2 \\ 3 \end{bmatrix}}_{\mathbf{B}} u$$

mit $z := x_1$ überführt werden. Beide Realisierungen sind äquivalent.

Danach werden die nicht konstanten, von $\mathbf{z}$-abhängigen Einträge jeweils durch skalarwertige Funktionen ersetzt

$$f_j(\mathbf{z}) = \underbrace{\frac{f(\mathbf{z}) - \underline{f}}{\overline{f} - \underline{f}}}_{w_{j1}(\mathbf{z})}\ \overline{f} + \underbrace{\frac{\overline{f} - f(\mathbf{z})}{\overline{f} - \underline{f}}}_{w_{j2}(\mathbf{z})}\ \underline{f} \tag{6}$$

mit

$$\overline{f} := \max f(\mathbf{z})\,, \quad \underline{f} := \min f(\mathbf{z})\,.$$

Die Gewichtsfunktionen w_{j1} und w_{j2} und die h_i Funktion erfüllen die Konvexitätseigenschaften, d.h.

$$w_{j1}(\mathbf{z}) + w_{j2}(\mathbf{z}) = 1\,, \quad w_{j1}(\mathbf{z}) \geq 0\,,\ w_{j2}(\mathbf{z}) \geq 0 \tag{7}$$

Die Ableitung von TS Modellen wird nun am Beispiel des mathematischen Pendels veranschaulicht.

Beispiel 2 [2]: Die bekannte Bewegungsgleichung des mathematischen Pendels wird zunächst in die Form (4) überführt

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l}\frac{\sin x_1}{x_1} & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{1}{m\,l^2} \end{bmatrix} u \tag{8}$$

mit $x_1 := \varphi$, $x_2 := \dot{\varphi}$, der Pendellänge l, der konzentrierten Masse m an der Position l und der Gravitationskonstante g. Im nächsten Schritt wird der Systemmatrixeintrag $a_{21}(x_1) = -\frac{g}{l}\frac{\sin x_1}{x_1}$ ersetzt durch

$$f(x_1) = w_1(x_1)\,\overline{f} + w_2(x_1)\,\underline{f} = -\frac{g}{l}\frac{\sin x_1}{x_1} \tag{9}$$

mit

$$w_1(x_1) = \frac{f(x_1) - \underline{f}}{\overline{f} - \underline{f}}\,, \qquad w_2(x_1) = \frac{\overline{f} - f(x_1)}{\overline{f} - \underline{f}}\,. \tag{10}$$

Somit folgt aus (8) unter Ausnutzung der Konvexität $w_1(x_1) + w_2(x_1) = 1$

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} 0 & 1 \\ w_1(x_1)\,\overline{f} + w_2(x_1)\,\underline{f} & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{1}{m\,l^2} \end{bmatrix} u \\ &= \begin{bmatrix} 0 & w_1(x_1) + w_2(x_1) \\ w_1(x_1)\,\overline{f} + w_2(x_1)\,\underline{f} & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{1}{m\,l^2} \end{bmatrix} u \end{aligned}$$

mit den Zwischenschritten

$$\begin{aligned} \dot{\mathbf{x}} = w_1(x_1) \underbrace{\begin{bmatrix} 0 & 1 \\ \overline{f} & 0 \end{bmatrix}}_{=:A_1} \mathbf{x} + w_2(x_1) \underbrace{\begin{bmatrix} 0 & 1 \\ \underline{f} & 0 \end{bmatrix}}_{=:A_2} \mathbf{x} + \\ w_1(x_1) \begin{bmatrix} 0 \\ \frac{1}{m\,l^2} \end{bmatrix} u + w_2(x_1) \begin{bmatrix} 0 \\ \frac{1}{m\,l^2} \end{bmatrix} u \end{aligned}$$

und $z := x_1$, $h_1(z) = w_1(z)$ und $h_2(z) = w_2(z)$ die TS Formulierung (2) von (8)

$$\dot{\mathbf{x}} = \sum_{i=1}^{2} h_i(z) \left(\mathbf{A}_i\,\mathbf{x} + \mathbf{B}\,u \right)\,.$$

2.3 LMI-Reglerentwurf

Um die Vor- und Nachteile einer Realisierung bezogen auf einen modellgestützten Reglerentwurf bewerten zu können, wird kurz auf die Synthese mit linearen Matrixungleichungen (LMIs) für Zustandsregler der Form

$$\mathbf{u} = -\sum_{i=1}^{N_r} h_i(\mathbf{z})\, \mathbf{K}_i\, \mathbf{x} \tag{11}$$

eingegangen. Falls hierbei die Funktionen h_i zu dem Modell der Strecke (4) identisch sind handelt es sich um das von Wang et al. eingeführte PDC (Parallel Distributed Compensation) Schema. Der geschlossene Regelkreis lautet hierfür

$$\dot{\mathbf{x}} = \sum_{i=1}^{N_r} \sum_{j=1}^{N_r} h_i(\mathbf{z})\, h_j(\mathbf{z}) \left(\mathbf{A}_i - \mathbf{B}_i\, \mathbf{K}_j\right) \tag{12}$$

Dieser ist global asymptotisch stabil (GAS) falls gilt:

Theorem 2.1 *Das TS Modell* (2) *mit dem Regelgesetz* (11) *ist global asymptotisch stabil (GAS) falls* $\mathbf{X} > 0$ *und* $\mathbf{M}_i$ *für* $i = 1, \ldots, N_r$ *existieren, so dass mit*

$$\mathbf{\Gamma}_{ij} = \mathbf{X}\, \mathbf{A}_i^T + \mathbf{A}_i\, \mathbf{X} - \mathbf{M}_j^T\, \mathbf{B}_i^T - \mathbf{B}_i\, \mathbf{M}_j \tag{13}$$

die linearen Matrixungleichungen

$$\begin{aligned} \mathbf{\Gamma}_{ij} < 0 \quad & \textit{für} \quad i = 1, 2, \ldots, N_r \\ \mathbf{\Gamma}_{ij} + \mathbf{\Gamma}_{ji} < 0 \quad & \textit{für} \quad j = i+1, i+2, \ldots, N_r \end{aligned} \tag{14}$$

erfüllt sind .

Mit diesem Theorem wird nicht nur die Stabilität nachgewiesen, sondern es lassen sich auch direkt die Zustandsrückführungsmatrizen für (11) aus

$$\mathbf{K}_i = \mathbf{M}_i \mathbf{X}^{-1} \quad i = 1, \ldots, N_r \tag{15}$$

berechnen. Der Herleitung von (13) ist u.a. in [6] beschrieben. Die Relaxationsbeziehung (14) stammt aus [12]. Die Anzahl der linearen Matrixungleichung N_{LMI} is abhängig von der Anzahl der Teilmodelle und entspricht dem Aufwand für das Finden einer Lösung. Falls das TS Modell gemeinsame Eingangs- oder Systemmatrizen der Teilmodelle aufweißt ($\mathbf{A}_i = \mathbf{A}\ \forall i$ oder $\mathbf{B}_i = \mathbf{B}\ \forall i$) reduziert sich die Anzahl der LMIs auf

$$N_{LMI} = N_r + 1\ . \tag{16}$$

Es ist daher erstrebenswert im Vorfeld des Reglerentwurfs eine Realisierung zu ermitteln, bei dem das TS Modell (2) gemeinsame $\mathbf{A}$ oder $\mathbf{B}$ Matrizen hat[1].

2.4 Designregeln für die Ableitung von TS Modellen mit SL

Im Folgenden werden vier Regeln für die Ableitung von TS Modellen aus nichtlinearen Mehrgrößensystemen (1) angegeben. Bei den Beispielen beschränken wir uns auf die Vektordifferentialgleichung $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, die Ausgangsgleichung $\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u})$ kann analog dazu behandelt werden.

1. **Sicherstellen der vollständigen Steuerbarkeit der Teilmodelle**
   Für den Reglerentwurf mit LMIs muss gewährleistet sein, dass die Teilmodelle $\{\mathbf{A}_i, \mathbf{B}_i\}$ vollständig steuerbar sind. Selbst wenn das ursprüngliche System vollständig steuerbar ist, kann durch eine ungünstige Wahl der Sektorfunktionen der Steuereingriff entweder komplett in die Prämissenvariablen der h_i Funktion aufgehen oder die Kopplung zwischen den Eingängen und mindestens einem Zustand wird aufgehoben. Dies wird an dem folgenden Beispiel verdeutlicht.

   Beispiel 3: Gegeben ist das System

   $$\begin{aligned} \dot{x}_1 &= -x_1\, u \\ \dot{x}_2 &= -x_1 - x_2 \end{aligned} \tag{17}$$

   wobei x_1 und u beschränkt sind da $x_1 \in [-2\,,\, 2]$ und $u \in [-1\,,\, 1]$. Das System (17) wird zunächst überführt in eine TS Form

   $$\dot{\mathbf{x}} = h_1(u) \begin{bmatrix} -\overline{u} & 0 \\ -1 & -1 \end{bmatrix} \mathbf{x} + h_2(u) \begin{bmatrix} -\underline{u} & 0 \\ -1 & -1 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u\ .$$

   Wie man erkennt, geht der Steuereingriff nur noch in die Prämissenvariablen ein. Die Teilmodelle sind somit aufgrund der Nullmatrix als Eingangsmatrix nicht mehr steuerbar. Jedoch kann alternativ auch folgende Realisierung abgeleitet werden

   $$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 \\ -1 & -1 \end{bmatrix} \mathbf{x} + h_1(u) \begin{bmatrix} -\overline{x}_1 \\ 0 \end{bmatrix} u + h_2(u) \begin{bmatrix} -\underline{x}_1 \\ 0 \end{bmatrix} u\ .$$

   Hierbei sind die Teilmodelle $\{\mathbf{A}, \mathbf{B}_1\}$ und $\{\mathbf{A}, \mathbf{B}_2\}$ steuerbar.

[1] Beim Beobachterentwurf führt dagegen eine gemeinsame $\mathbf{C}$ Matrix auf eine niedrigere Anzahl von LMIs.

2. **Sicherstellen der vollständigen Beobachtbarkeit der Teilmodelle**
Für den Beobachterentwurf mit LMIs muss gewährleistet sein, dass die Teilmodelle $\{\mathbf{A}_i, \mathbf{C}_i\}$ vollständig beobachtbar sind. Analog zu der vollständigen Steuerbarkeit kann durch eine ungünstige Wahl der Struktur (4) und Sektorfunktionen die vollständige Beobachtbarkeit der Teilmodelle nivelliert werden.

3. **Reduktion der Anzahl der LMIs**
Durch eine passende Wahl der Position der Sektorfunktion in den Matrizen der Teilmodelle können gemeinsame $\mathbf{A}$, $\mathbf{B}$ oder $\mathbf{C}$ Matrizen entstehen, wodurch die Anzahl der LMIs beim Regler- und Beobachterentwurf deutlich reduziert wird. Dies zeigt das folgende Beispiel:

   Beispiel 6: Gegeben ist das System

$$\begin{aligned} \dot{x}_1 &= -x_1 + x_2\, u^2 \\ \dot{x}_2 &= x_1 - x_2^2 + u\,. \end{aligned} \tag{18}$$

   Dies kann überführt werden in

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} -1 & 0 \\ 1 & -x_2 \end{bmatrix} \mathbf{x} + \begin{bmatrix} x_2\, u \\ 1 \end{bmatrix} u \\ &= \sum_{i=1}^{4} h_i(x_2, u) \left(\mathbf{A}_i\, \mathbf{x} + \mathbf{B}\, u \right) \end{aligned} \tag{19}$$

   oder in

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} -1 & u^2 \\ 1 & -x_2 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ &= \sum_{i=1}^{4} h_i(x_2, u) \left(\tilde{\mathbf{A}}_i\, \mathbf{x} + \tilde{\mathbf{B}}\, u \right)\,. \end{aligned} \tag{20}$$

   Beide sind von der Gesamtdynamik äquivalent. Für den Reglerentwurf ist jedoch (20) wegen der konstanten $\tilde{\mathbf{B}}$ Matrix besser geeignet als (19), da anstatt $N_{LIN} = 11$ nur $N_{LIN} = N_r = 4$ LMIs bei der Optimierung berücksichtigt werden müssen.

4. **Vermeiden von nicht-messbaren Prämissenvariablen**
Bekanntermaßen können nicht-messbare Zustände in dem hier betrachteten Rahmen mit TS Beobachtern rekonstruiert werden. Falls jedoch rekonstruierte Zustände Teil des Prämissenvektors sind, führt eine für den Nachweis der Stabilität notwendige Abschätzung einer

oberen Fehlerschranke häufig auf konservative Ergebnisse bei der konvexen Optimierung bzw. es kann keine zulässige Lösung ermittelt werden. Es ist daher sinnvoll, schon bei der Realisierung darauf zu achten, dass nur Prämissenvariablen gewählt werden, die messbar sind. Ein reale Anwendung, bei der dies durch eine geeignete Wahl einer Realisierung erreicht wurde ist in [8] dokumentiert. Anhand des folgenden Fallbeispiels wird dies ebenfalls verdeutlicht.

Beispiel 7: Gegeben ist das System mit nur einem messbaren Zustand x_2:

$$\begin{aligned} \dot{x}_1 &= -x_1 + x_1\, x_2 \\ \dot{x}_2 &= -x_2 + u\,. \end{aligned} \tag{21}$$

Dieses kann überführt werden in das TS Modell

$$\dot{\mathbf{x}} = \begin{bmatrix} -1 & x_1 \\ 0 & -1 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = \sum_{i=1}^{2} h_i(x_1) \left(\mathbf{A}_i\, \mathbf{x} + \mathbf{B}\, u \right) \tag{22}$$

oder alternativ in

$$\dot{\mathbf{x}} = \begin{bmatrix} x_2 - 1 & 0 \\ 0 & -1 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = \sum_{i=1}^{2} h_i(x_2) \left(\tilde{\mathbf{A}}_i\, \mathbf{x} + \tilde{\mathbf{B}}\, u \right). \tag{23}$$

Nur bei der letzten Realisierungsvariante (23) ist die Prämissenvariable messbar.

3 TS Modelle für den Reglerentwurf und Beobachterentwurf bei Windenergieanlagen

Im Folgenden sollen die zuvor vorgestellten Designregeln auf eine komplexe Anwendung, der Ableitung von TS Modellen für den Reglerentwurf bei Windenergieanlagen (WEA), übertragen werden. Zunächst wird hierzu das physikalische Modell einer WEA aus [3] eingeführt. Anschließend werden zwei verschiedene Realisierungen angegeben. Die Vor- und Nachteile bezogen auf den Regler- wie Beobachterentwurf werden gegenübergestellt.

3.1 Regelungsorientiertes Modell

Das regelungsorientierte Modell beinhaltet die Bewegungsdifferentialgleichung

$$\mathbf{M}\,\ddot{\mathbf{q}} + \mathbf{D}\,\dot{\mathbf{q}} + \mathbf{K}\,\mathbf{q} = \mathbf{f} \tag{24}$$

mit den generalisierten Koordinaten

$$\mathbf{q} := [\; y_T, y_B, \theta_r, \theta_g]^{\mathrm{T}} \tag{25}$$

und den generalisierten Kräften

$$\mathbf{f} := [\; F_T, F_T, T_r, -T_g \;]^{\mathrm{T}}\,. \tag{26}$$

Die Massenmatrix $\mathbf{M}$, Dämpfungsmatrix $\mathbf{D}$ und Steifigkeitsmatrix $\mathbf{K}$ sind gegeben durch

$$\mathbf{M} = \begin{bmatrix} m_T + N\,m_B & N\,m_B & 0 & 0 \\ N\,m_B & N\,m_B & 0 & 0 \\ 0 & 0 & J_r & 0 \\ 0 & 0 & 0 & J_g \end{bmatrix},\quad \mathbf{D} = \begin{bmatrix} d_T & 0 & 0 & 0 \\ 0 & N\,d_B & 0 & 0 \\ 0 & 0 & d_S & -d_S \\ 0 & 0 & -d_S & d_S \end{bmatrix},$$

$$\mathbf{K} = \begin{bmatrix} k_T & 0 & 0 & 0 \\ 0 & N\,k_B & 0 & 0 \\ 0 & 0 & k_S & -k_S \\ 0 & 0 & -k_S & k_S \end{bmatrix}.$$

Die aerodynamische Kopplung zwischen der mittleren Windgeschwindigkeit v, der Rotordrehzahl $\omega_r = \dot{\theta}_r$ und dem Pitchwinkel β der Rotorblätter wird in der Schubkraft

$$F_T = \frac{1}{2}\rho\,\pi\,R^2\,C_T(\lambda,\beta)\,v^2\,, \tag{27}$$

und dem aerodynamischen Rotormoment

$$T_r = \frac{1}{2}\rho\,\pi\,R^3\,C_Q(\lambda,\beta)\,v^2\,, \tag{28}$$

zusammengefaßt. Die Gleichungen beinhalten die analytischen Näherungen der aerodynamischen Kennfelder $C_T(\lambda,\beta)$ und $C_Q(\lambda,\beta)$ aus [3], die Luftdichte ρ, den Rotorradius R, β als kollektiven Pitchwinkel und

$$\lambda = \frac{\omega_r\,R}{v} \tag{29}$$

Symbol	Beschreibung	Einheit
m_T	konzentrierte effektive Masse von Gondel und Turm	kg
m_B	konzentrierte Masse vom Blatt bezogen auf die Blattspitzenbewegung	kg
N	Anzahl der Rotorblätter	–
J_r	Massenträgheit des Rotors	kg m^2
J_g	Massenträgheit des Generators	kg m^2
d_T	effektive Turmdämpfung	$\frac{\text{Ns}}{\text{m}}$
d_B	Blattdämpfung in Schlagrichtung	$\frac{\text{Ns}}{\text{m}}$
d_S	Antriebsstrangdämpfung (Torsionsdämpfung)	Nm s
k_T	effektiver Turmsteifigkeitskoeffizient	$\frac{\text{N}}{\text{m}}$
k_B	effektiver Blattsteifigkeitskoeffizient	$\frac{\text{N}}{\text{m}}$
k_S	Antriebsstrangdämpfung (Torsionssteifigkeit)	Nm
ρ	Luftdichte	$\frac{\text{kg}}{\text{m}^3}$
R	Rotorradius	m

Tabelle 1: Parameter des reduzierten Windturbinenmodells

als die dimensionslose Schnelllaufzahl.

Zusätzlich muss für den Reglerentwurf die Dynamik der Stellglieder, in diesem Fall die Pitchantriebe und der Generator, berücksichtigt werden:

$$\dot{\beta} = -\frac{1}{\tau}\,\beta + \frac{1}{\tau}\,\beta_d\,, \qquad \dot{T_g} = -\frac{1}{\tau_g}\,T_g + \frac{1}{\tau_g}\,T_{g_d}\,, \tag{30}$$

mit den Verzögerungszeiten τ, τ_g, dem Sollpitchwinkel β_d, dem Generatormoment T_g und dem Sollgeneratormoment T_{g_d}. Die Pitchverstelldynamik berücksichtigt dabei nur eine kollektive Rotorblattverstellung, d.h. eine unabhängige Verstellung der Rotorblätter wird nicht modelliert. Die restlichen Modellparameter sind in der Tabelle 1 zusammengefaßt.

3.2 TS Modell Realisierungen für den LMI-Entwurf

Unter Berücksichtigung der zuvor eingeführten Designregeln, können zwei Realisierungen basierend auf dem physikalischen Modell (24)-(30) angegeben werden. Zum einen das *TS Modell für den Reglerentwurf*

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}(\mathbf{x},\mathbf{u})\,\mathbf{x} + \mathbf{B}\,\mathbf{u} \\ &= \sum_{i=1}^{4} h_i(\mathbf{x},\mathbf{u})\,\big(\mathbf{A}_i\,\mathbf{x} + \mathbf{B}\,\mathbf{u}\big)\end{aligned} \tag{31}$$

mit

$$\begin{aligned}\mathbf{x} &= [\,y_T\,,\,y_B\,,\,\theta_r-\theta_g\,,\,\dot{y}_T\,,\,\dot{y}_B\,,\,\dot{\theta}_r\,,\,\dot{\theta}_g\,,\,\beta\,,\,T_g]^{\mathrm{T}}\,,\\ \mathbf{u} &= [\,\beta_d\,,\,T_{g_d}]^{\mathrm{T}}\end{aligned}$$

wobei

$$\mathbf{A}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ -\frac{k_T}{m_T} & \frac{Nk_B}{m_T} & 0 & -\frac{d_T}{m_T} & \frac{Nd_B}{m_T} & 0 & 0 & 0 & 0 \\ \frac{k_T}{m_T} & -\frac{k_B}{m_B} - \frac{Nk_B}{m_T} & 0 & \frac{d_T}{m_T} & -\frac{d_B}{m_B} - \frac{Nd_B}{m_T} & 0 & 0 & f_1(\mathbf{x},\mathbf{u}) & 0 \\ 0 & 0 & -\frac{k_S}{J_r} & 0 & 0 & -\frac{d_S}{J_r} & \frac{d_S}{J_r} & f_2(\mathbf{x},\mathbf{u}) & 0 \\ 0 & 0 & \frac{k_S}{J_g} & 0 & 0 & \frac{d_S}{J_g} & -\frac{d_S}{J_g} & 0 & -\frac{1}{J_g} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau_g} \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0}_{7\times 2} & \\ \frac{1}{\tau} & 0 \\ 0 & \frac{1}{\tau_g} \end{bmatrix}.$$

Mit der Erweiterung des Schubkraft- und Rotormomentenanteils um $\beta + \delta$ mit $\delta \ll 1$

$$f_1(\mathbf{x},\mathbf{u}) = \frac{1}{N\, m_B}\, F_T(\mathbf{x},\mathbf{u})\, \frac{1}{\beta+\delta}\,,$$
$$f_2(\mathbf{x},\mathbf{u}) = \frac{1}{J_r}\, T_r(\mathbf{x},\mathbf{u})\, \frac{1}{\beta+\delta}$$

ist die Steuerbarkeit der Teilmodelle aus $\mathbf{A}(\mathbf{x},\mathbf{u})$ sichergestellt.

Das *TS Modell für den Beobachterentwurf* unterscheidet sich in den Einträgen von $\mathbf{A}(\mathbf{x},\mathbf{u})$ und hat die Form

$$\mathbf{A}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ -\frac{k_T}{m_T} & \frac{Nk_B}{m_T} & 0 & -\frac{d_T}{m_T} & \frac{Nd_B}{m_T} & 0 & 0 & 0 & 0 \\ \frac{k_T}{m_T} & -\frac{k_B}{m_B} - \frac{Nk_B}{m_T} & 0 & \frac{d_T}{m_T} & -\frac{d_B}{m_B} - \frac{Nd_B}{m_T} & \tilde{f}_1(\mathbf{x},\mathbf{u}) & 0 & 0 & 0 \\ 0 & 0 & -\frac{k_S}{J_r} & 0 & 0 & -\frac{d_S}{J_r} + \tilde{f}_2(\mathbf{x},\mathbf{u}) & \frac{d_S}{J_r} & 0 & 0 \\ 0 & 0 & \frac{k_S}{J_g} & 0 & 0 & \frac{d_S}{J_g} & -\frac{d_S}{J_g} & 0 & -\frac{1}{J_g} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau_g} \end{bmatrix}$$

mit

$$\tilde{f}_1(\mathbf{x},\mathbf{u}) = \frac{1}{N\, m_B}\, F_T(\mathbf{x},\mathbf{u})\, \frac{1}{\dot{\theta}_r} \qquad \text{für} \quad \dot{\theta}_r > 0\,,$$
$$\tilde{f}_2(\mathbf{x},\mathbf{u}) = \frac{1}{J_r}\, T_r(\mathbf{x},\mathbf{u})\, \frac{1}{\dot{\theta}_r} \qquad \text{für} \quad \dot{\theta}_r > 0$$

wobei $\dot{\theta}_r > 0$ immer erfüllt ist, da die Regelung und der Beobachter erst nach dem Rotoranlauf aktiv ist. Das System ist beobachtbar mit der Ausgangsmatrix

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} .$$

Diese Realsierung ist jedoch nicht vollständig steuerbar, da der Pitchwinkel im Gegensatz zur vorhergehen Realisierung mit keinen der anderen Zustandsvariablen verkoppelt ist. Der Vorteil dieser Realisierung ist, dass die Näherung $\delta \ll 1$ nicht verwendet werden muss.

4 Zusammenfassung und Ausblick

Vorgestellt wurden Designregeln für eine LMI-entwurfsoptimierte Ableitung von TS Modellen aus nichtlinearen Systemen in Zustandsform. In zukünftigen Arbeiten soll der Ableitungsprozess durch die Berücksichtigung zusätzlicher Entwurfsziele, wie z.B. die Eingangs- und Ausgangsbeschränkung, weiter systematisiert werden.

Literatur

[1] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan. Linear Matrix Inequalities in System and Control Theory. Society for Industrial and Applied Mathematics, Philadelphia, 1994.

[2] S. Georg, M. Müller und H. Schulte, Wind Turbine Model and Observer in Takagi-Sugeno Model Structure. In EAWE Conference "The Science of Making Torque from Wind", European Academy of Wind Energy, preprint paper (http://arxiv.org/abs/1401.8199), Oldenburg, Germany, 2012.

[3] S. Georg, H. Schulte, H. Aschemann, Control-Oriented Modelling of Wind Turbines using a Takagi-Sugeno Model Structure, IEEE International Conference on Fuzzy Systems, Brisbane, Australien, 2012.

[4] T. A. Johansen, K. Hunt, P. Gawthrop, and H. Fritz. Off-equilibrium linearisation and design of gain-scheduled control with application to

vehicle speed control. Control Engineering Practice, 6(2), S. 167–180, 1998.

[5] A. Kroll, Zur regelungsorientierten Ableitung von Takagi-Sugeno-Modellen, Automatisierungstechnik 12/2011; 59, S. 705-720.

[6] Z. Lendek, T. M. Guerra, R. Babuska und B.D. Schutter, Stability Analysis and Nonlinear Observer Design Using Takagi-Sugeno Fuzzy Models, Springer-Verlag Berlin Heidelberg, 2010.

[7] H. Schulte, Approximative Modellierung, Systemidentifikation und Reglerentwurf mittels gewichteter Kombination lokaler Zustandsraummodelle am Beispiel fluidischer Antriebe, PhD thesis, Universität Kassel, 2006.

[8] H. Schulte, Robuster Beobachterentwurf für Takagi-Sugeno Fuzzy Systeme zur Überwachung hydrostatischer Fahrantriebe, In: Proc.18. Workshop Computational Intelligence, Dortmund, 3. Dezember - 5. Dezember, S. 90-104, 2008.

[9] H. Schulte und S. Georg, Nonlinear Control of Wind Turbines with Hydrostatic Transmission Based on Takagi-Sugeno Model, In: EAWE Conference 'The Science of Making Torque from Wind', European Academy of Wind Energy, Copenhagen, Denmark, June, 2014

[10] T. Takagi und M. Sugeno, Fuzzy identification of systems and its application to modelling in control, *IEEE Transactions on System, Man and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[11] K. Tanaka und H. O.Wang (2001). Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach. John Wiley & Sons, Inc.

[12] H. O. Wang, K. Tanaka, and M. F. Griffin. An Approach to Fuzzy Control of Nonlinear Systems: Stability and Design Issues. IEEE Transactions on Fuzzy Systems, 4(1), pp. 14–23, 1996.

Active Learning-Based Identification of Neuronal Assemblies in Parallel Spike Trains

Christian Braune, Rudolf Kruse

Institut für Wissens- und Sprachverarbeitung, Universität Magdeburg
Universitätsplatz 2, 39106 Magdeburg
Tel.: (0391) 67-52506
Fax: (0391) 67-12018
E-Mail: { christian.braune, rudolf.kruse}@ovgu.de

1 Abstract

For understanding how information is processed within the brain several different models have been proposed. They are either based on a common increase in neuronal firing activity or synchronous firing of several individual neurons. We present a novel method for detecting the so-called assemblies [4] of the latter model. Using parallel spike trains – recordings of neuronal activity – one use this information to answer quite well if an individual neuron belongs to (at least) one assembly or not. But detecting the underlying assembly structure remains a difficult task since normally neither the number of assemblies is known nor their respective size.

Using surrogate-based statistics [7] as an oracle we use active learning [15] to identify the underlying assembly structure. This approach not only uses the statistical information we calculate from the surrogates but the structural information we can obtain by defining a metric on the spike trains themselves. We show that for even a small number of coincidences relatively small assemblies can be detected without querying the oracle for every spike train.

2 Introduction

Understanding the way information is processed within the brain on a neuronal level is essential for understanding the way the brains works as a whole. Diseases such as Alzheimer's or epilepsy might be better understood if we could properly describe the expected behavior of the brain. On a single neuron level it is fairly well understood how the bio-electrical processes work and can be described with the help of several different models

such as the Hodgkin-Huxley model [9], a set of several differential equations, or simple Poisson processes that just model random spiking behavior. Using functional magnet resonance imaging (fMRI) the complete brain's activity can be observed. By measuring the amount of oxygen carried by the blood to different brain regions the general level of activity within each region can be estimated and conclusion may be drawn towards which regions are active when processing certain tasks [11].

On a multi-cellular level however a lot of discussion is still going on. Since Hebb's seminal work [4] it is commonly agreed that neuronal behavior is organized in groups of neurons working together. One focus in neural assembly research nowadays is how these assemblies exhibit themselves. Some favor that the hypothesis that neurons encode information and communicate with each other by (nearly) exact, single spikes. Others that the frequencies used within so-called bursts (spontaneous increases of the firing rate of a single neuron) is the true encoding scheme. Within this work we will consider the *spike time hypothesis* [14] which states that the nearly exact emission of single spikes is the carrier of information.

Within this hypothesis we still have to deal with several obstacles such as *temporal imprecision* and *selective participation*. The first describes imprecisions that might arise from the measurement or the biological process. Spikes that should be considered coincident might not appear at exactly the same time. The measurement devices sampling frequency might not allow such high precision or due to different distances between participating neurons to their nearest electrodes spikes exceed the detection threshold at slightly different times. On the other hand, the biological process producing the spikes cannot be expected to be such perfectly synchronized that we actually see two neurons emitting a spike at exactly the same time (on a nanoseconds scale). This problem can be coped with by not considering spikes as singular event but by rather treating them as intervals in which two spike may be considered synchronized [1].

The second obstacle is much more difficult to handle. Selective participation means that for every assembly activation not every neuron that belongs to this assembly takes part in the coincidence. The reason for this most certainly lies in the underlying biological process. Neurons need to accumulate a certain amount of neurotransmitters in their cell body to be able to emit a spike at all and so, after each activation they need some time before they are able to emit the next spike. If a neuron fired randomly just before the assembly activation, it might not yet be available for another activation and thus miss the coincidence. Seeing this effect and how strong

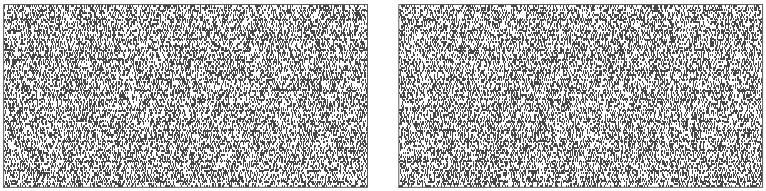


Figure 1: Two sets of spike trains, the left contains only random noise while the right contains an ensemble of 20 neurons. Upon visual inspection both are indistinguishable.

it influences the outcome of assembly detection will be one focus of our evaluation in Section 5.

With the current advancement in neural recording technology it is possible to gather the membrane potentials of many (in the order of hundred) channels in parallel at once. Each recorded channel undergoes a process called *spike sorting* during which single spikes are detected and assigned to individual neurons. Since a single electrode might be influenced by the membrane potentials of several neurons packed close together this step is necessary for the following analysis. The same holds true for one neuron possibly influencing more than one electrode at the same time but to a much lesser effect due to the spacing of the electrodes and the relative size of neuronal cells. After the *spike sorting* we have a set of neurons that ought to be present in the vicinity of the recording area, and each neuron can be associated with what is called a *spike train*. This is essentially an ordered set of time indices that identify individual spikes. A set of such spike trains originating from a recording of the same multi-electrode array, started at exactly the same time is called *parallel spike train*. See Figure 1 for an example of two sets of parallel spike trains – one containing only noise and the other a small ensemble.

Such a spike train T_i can be formally denoted as

$$T_i = \{t | \text{Neuron } i \text{ fires at time } t\}$$

and $\mathcal{T}$ is the set of all $T_i, 0 \leq i < n$.

An assembly $A \subset \mathcal{T}$ is a set of neurons (or spike trains) which exhibits more coincident spikes than what we would expect to see if they were independent of each other – according to the *spike time hypothesis*. Simply enumerating all subsets of $\mathcal{T}$ and testing them for the number of coincidences (e.g. with a χ^2 test) is a valid yet infeasible approach. With only 100 spike trains $2^{100} \approx 1.27 \cdot 10^{30}$ sets would have to be checked. An algorithm checking all possible combinations would roughly need three million

times longer than the age of our universe – and this only if it were able to check a million subsets within each second. For obvious reasons a more efficient way of finding assemblies is needed.

What we propose in this paper is the combination of an existing method for answering for a single spike train whether it belongs into an assembly or not [7] and semi-supervised support vector machines (S^3VM) [5], an extension of the well-known support vector machine [17]. By labeling the initially unlabeled data with the help of an oracle we will become able to train either an SVM or an S^3VM to classify the remaining, unlabeled data. Thus we are able to remove those spike trains that do not belong to any assembly and identify the assembly structure of the other spike trains.

The rest of the paper will be structured as follows. Section 3 gives a brief overview over existing assembly detection algorithms. Section 4 will describe our approach which will be evaluated in Section 5. In the final section we will discuss the results and present some options for future research.

3 Related Work

3.1 Neural Assembly Detection

Algorithms for detecting neural assemblies date back to at least [**?**] in which a first algorithmic way of coping with the combinatorial explosion problem is presented. Instead of testing all possible subsets for their assembly validity a greedy search scheme is developed where every pair of spike trains (properly binned into 1ms long bins) are tested with a χ^2 test for independence. The events used are retrieved from the spike train data such that a bin contains a 1 if there was at least one spike in that bin. If the result of the χ^2 test exceed a user defined significance level the spike trains involved are merged such that only those bins contain a 1 that contained a 1 in both spike trains (a bit-wise AND). The participating spike trains are then removed and replaced by the spike train resulting from the merge. Iteratively repeating this process leads to hopefully all assembly neurons being merged into a single spike train and all the assembly activations being the only 1 left.

A similar result can be obtained when viewing each time bin as a transaction and each neuron as an item. Then frequent item set mining based approaches can be used to identify the assembly structure. Here it has to be remarked that it is not sufficient to just report the maximal sets since a

single neuron might randomly be active while the assembly was active (an effect that is increased with increasing time bin width) which will lead to spurious assemblies.

All the above mentioned methods aim at directly identifying the assembly structure. This is probably one of the hardest tasks in assembly detection. The simplest one would be answering whether there is (at least) one assembly present at all. For a binned spike train this can be done by analyzing the distribution of activity patterns [12, 16], i.e. the distribution of how often a certain number of spike trains were active at the same time. Since the activation of a single neuron could be seen as a Bernoulli experiment we can estimate the distribution of these activations from the individual or the average firing rates under the assumption that all spike trains are independent of each other. If the deviation from the expected Bernoulli distribution is too high, we can assume that our assumption that all spike trains were independent of each other is wrong.

Building on that method we can define a relatively reliable and simple test for individual spike trains [2]. If we replace a single spike train by a new, randomly generated one with exactly the same number of spikes, the distribution should not change that much, if the replaced spike train was truly independent. If, however, the distribution changes significantly then we probably removed a spike train that belonged to an assembly.

This approach is similar to the set of methods implemented in the `NAss` library [6]. They selectively keep certain statistical properties of a spike train (like the number of spikes or the inter-spike interval distribution) while purposely destroying other properties. By doing so they can test if the newly generated spike train exhibits the same pattern as the original spike train. Whenever the same (or an even more extreme) pattern is observed it serves as an indicator for the randomness of the original pattern. The fraction of such patterns observed over the total number of trials can be seen as some kind of empirical p-value. With that in mind we can use this class of algorithm as an oracle in an active learning scenario.

4 Assembly Identification

In machine learning we can distinguish between unsupervised and supervised learning [13, 8]. While the first knows nothing about the data other than the data itself in supervised learning the data is labeled, i.e. there is a class label present which is often the target value that the learner should become able to predict for unlabeled data. However, obtaining labels for

data can be an expensive task which so that we do not have labels for all data points. The classical setting would be to only use the labeled instances and neglect any knowledge about the unlabeled data. The structure of the unlabeled instances might still give us some clues about the class label distribution. This can be accounted for with S^3VM [5]. They introduce another term into the objective function of the standard SVM and can use the information available from the location of the unlabeled instances as well. This sort of experiment is also called *optimal experimentaldesign*. On the other hand, simply querying more labels might be enough to train a normal SVM to solve the problem.

We therefore have to options to identity assembly neurons:

1. We use an SVM and neglect all knowledge about the structural information that might be available from the unlabeled points.
2. We use an S^3VM and use the structural knowledge to see if we benefit from it.

For the training process we use the NAss library[1] [7] as an oracle for obtaining labels in the transductive learning process. Since initially all spike trains are unlabeled (in contrast to the usual semi-supervised learning scenario), we randomly query spike trains until we have seen at least two different labels. All points that have been labeled so far are then used for training a support vector machine while the unlabeled points are omitted. The SVM uses an RBF kernel. After the initial training we query another ten additional points and re-train the SVM with the new set of labeled data (each time exactly one additional point). The point to be queried is now not chosen randomly anymore but instead we use a principle called *uncertainty sampling*. According to this the point which is supposed to be the most informative next point is the point for which the current prediction is the least certain, i.e. the point which lies closest to the decision boundary. For each unlabeled point we thus calculate the distance to the decision boundary and query the oracle for a label of that point.

Since spike trains themselves are not points in a metric space, directly calculating neither the distance between two trains nor the distance to a whatsoever natured decision boundary is simple in the first place. From [1] we already know that extending distance measures for binary vectors to spike trains can be a promising way to transform spike trains into points in a metric space. Such a transformation would be loss-less if we were to map

[1] Obtainable at `www.borgelt.net/nass.html`

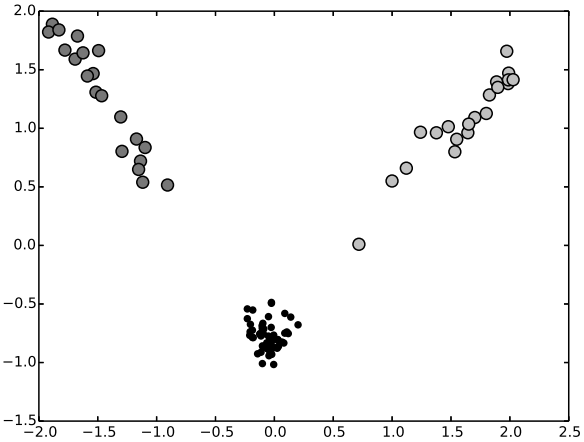


Figure 2: Metric representation of a set of 100 spike trains. Two assemblies are present (top left, top right corner) which can be clearly distinguished from the noise (bottom center).

the spike train from $\mathcal{T}, |\mathcal{T}| = n$ onto $\mathbb{R}^{n-1}$. However, since most of the spike trains might be considered noise anyways, a mapping onto a much lower-dimensional space is still suitable. In [1] two dimension are shown to be sufficient in most cases.

The first step for analyzing the spike trains is therefore calculating the distance matrix using $d_{Dice} = \frac{n_{10}+n_{n01}}{2n_{11}+n_{10}+n_{n01}}$ [3]. The resulting two-dimensional representation of spiek trains ideally looks like the on depicted in Figure 2. For this initially completely unlabeled data we query the oracle until we received two different class labels and with this data we can train either the SVM or the S^{3}VM. The results of this training can be seen in Figure 3. We can already see that the structural information available to the S^{3}VM helps it to initially get the right classification while the SVM misclassifies one point of this fairly easy data set.

Especially if more than one assembly is present in the data, it is likely that the learner distinguishes between one assembly and the remaining points. So we need more class labels. Querying the oracle another time and retraining the learners yields the results visible in Figures 5 and **??**. Please note that the order of queried points is different because different points lie closer to the decision boundary in each of the two training phases.

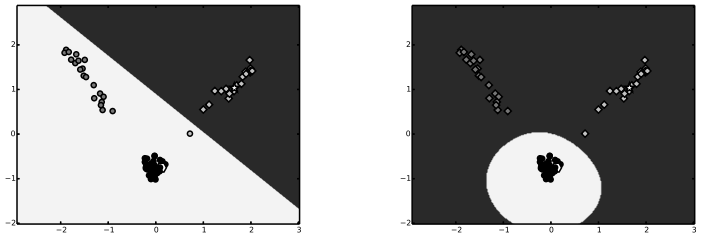


Figure 3: Decision boundary and classification results for SVM (left) and S^3VM (right) after querying only labels from two data points (star and hexagonal shapes). SVM misclassifies a complete group of points and a single point from the other group.

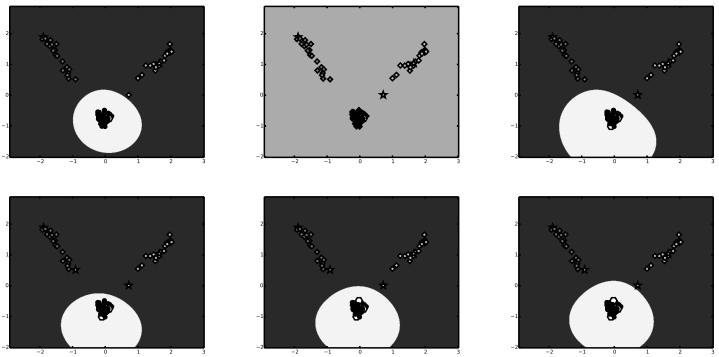


Figure 4: Training results of the SVM learner after querying additional labels. For each plot a single additional label has been queried.

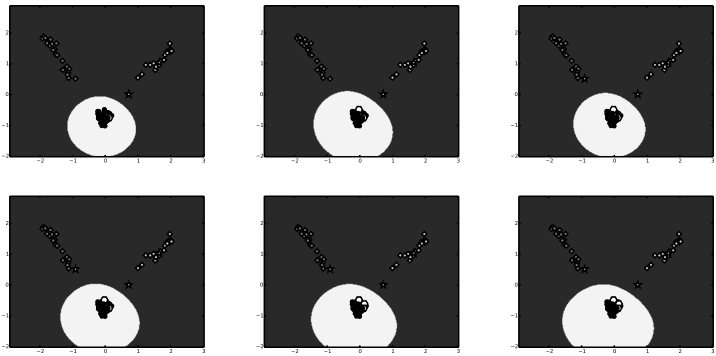


Figure 5: Training results of the S^3VM learner after querying additional labels. For each plot a single additional label has been queried.

As we can see, both methods are able to distinguish between the two classes (assembly and non-assembly) easily after the first extra label has been queried after the initial training. This is partially due to the extremely easy structure present here and might not always be the case.

In the next Section we will evaluate both strategies (SVM and S^3VM) in different problem setting (with varying number of assemblies, assembly neurons and different settings for selective participation).

5 Evaluation

We evaluate our proposed method on synthetically generated data. This way we can ensure that certain properties we are looking for are really contained within the data. Since there is still dispute on how neurons encode information and no one knows for sure if what we are looking for is actually contained in real spike trains, *in silico* analysis gives us some kind of control over the data. We can generate large amounts of data sets with various different settings and can therefor give some boundaries under which our method works well and to what degree the results can be trusted.

The model we chose for generating spike trains is a simple Poisson process. The inter-spike intervals are drawn from an exponential distribution with its only parameter being the firing frequency of the neuron to be modeled. Neurons modeled this way will exhibit solely background noise fi-

ring containing no information at all. Whether such neurons actually exist is debatable. It is more likely that due to the measurement some assemblies might not be covered completely and thus some neurons that belong to incompletely covered assemblies appear to show only random noise.

Neurons that form an assembly are modeled in a slightly different way. For such neurons we model the background firing with one Poisson process and the coincidences shared by the assembly by another (*mother*) process which is also a Poisson process. Combining both the individual background processes and the mother process into a single spike train still yields a Poisson process with increased firing rate. By carefully adjusting the background firing rate, we can model a neuron that belongs to any number of (overlapping) assemblies. For our tests here we ensure that each neuron belongs to at most one assembly. Points are copied into the background firing process with a certain probability (usually 1.0, 0.8 or 0.6) to model selective participation.

To obtain the initial set of labels, we query the oracle for random points until we obtained two different labels. After the training of the initial SVM we calculate the classification accuracy using the Adjusted Rand Index [10]. We do so for the initial SVM as well as for ten subsequent queries (the same for S^3VM).

To show that our method is capable of detecting the assembly neurons with high precision in difficult cases, we test it in different scenarios. All setups have in common that there are always 100 spike trains present and all neurons have a fixed firing rate of $20Hz$. The assembly activity will be specified by the number of shared coincidences – not by an actual firing rate. Thus fluctuations arising from the stochastic process generating the spike trains are minimized, i.e. there is no variation in the number of coincidences. For a large number of coincidences the distinguishing merely matters but for low coincidence rates such variations might lead to (significantly) different results. All spike trains have been simulated over a period of 3 seconds with 15 shared coincidences if not stated otherwise.

6 Conclusion and Future Work

In this paper we have presented how active learning in combination with (semi-) supervised learning can be used to identify neural assemblies. The results show that unsurprisingly the best results can be achieved when the assemblies present in the data are quite large and no selective participation has to be considered. What is more surprising is the fact, that the S^3VM

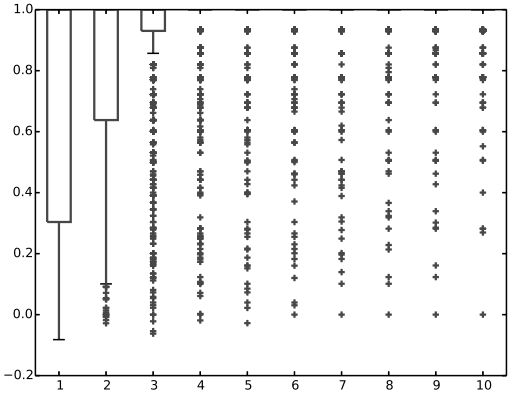


Figure 6: Classification accuracy measured using the Adjusted Rand Index for an S^3VM classifier for a single assembly of 10 neurons with copy probability 1.0.

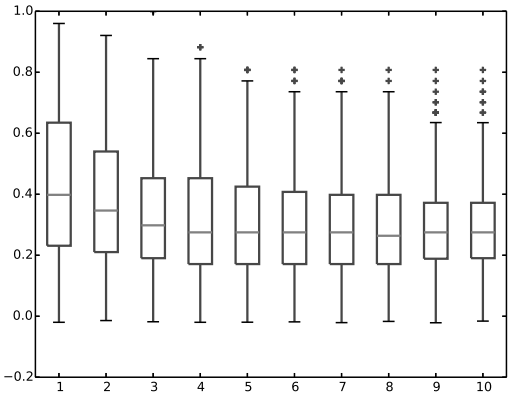


Figure 7: Classification accuracy measured using the Adjusted Rand Index for an S^3VM classifier for two assemblies of 20 neurons each with copy probability 0.6.

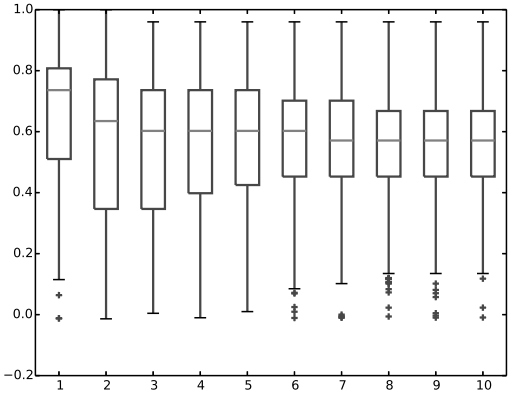


Figure 8: Classification accuracy measured using the Adjusted Rand Index for an S^{3}VM classifier for two assemblies of 20 neurons each with copy probability 0.8.

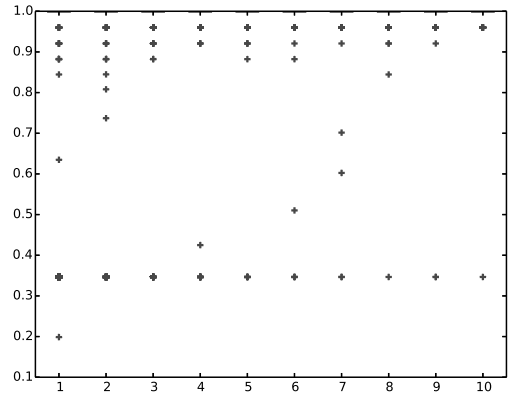


Figure 9: Classification accuracy measured using the Adjusted Rand Index for an S^{3}VM classifier for two assemblies of 20 neurons each with copy probability 1.0.

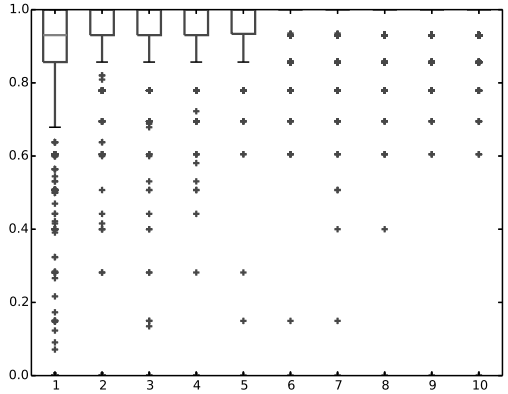


Figure 10: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for a single assembly of 10 neurons with copy probability 0.8.

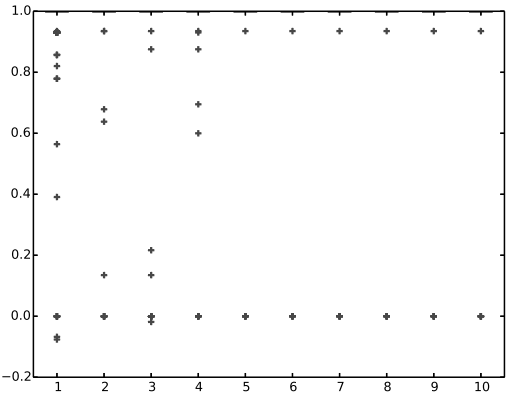


Figure 11: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for a single assembly of 10 neurons with copy probability 1.0.

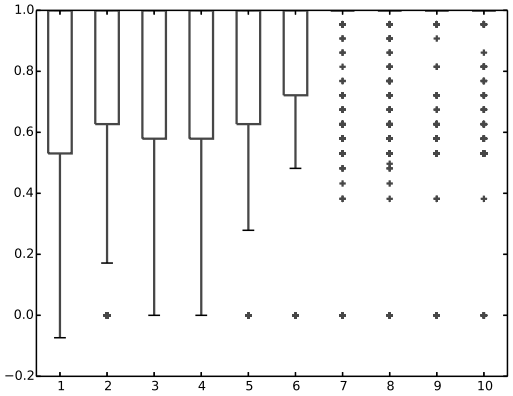


Figure 12: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for a two assemblies of 10 neurons each with copy probability 1.0 but only 10 coincidences shared.

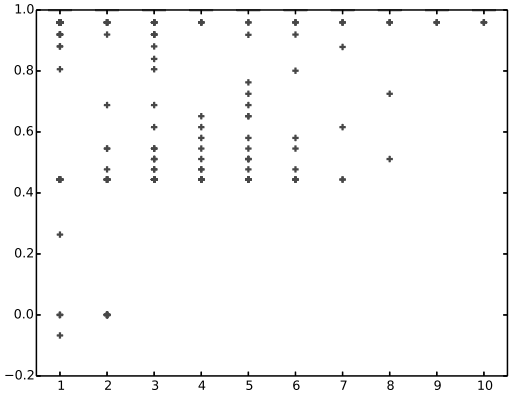


Figure 13: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for a two assemblies of 10 neurons each with copy probability 1.0 but only 10 coincidences shared.

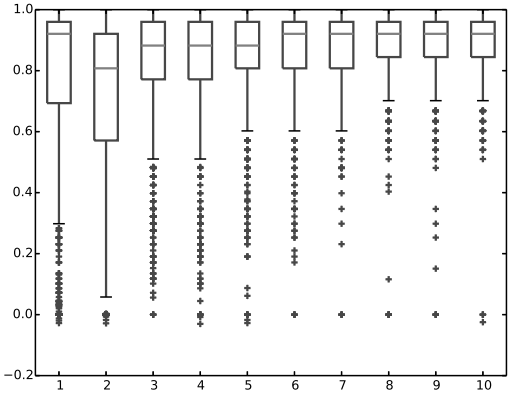


Figure 14: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for two assemblies of 20 neurons each with copy probability 0.6.

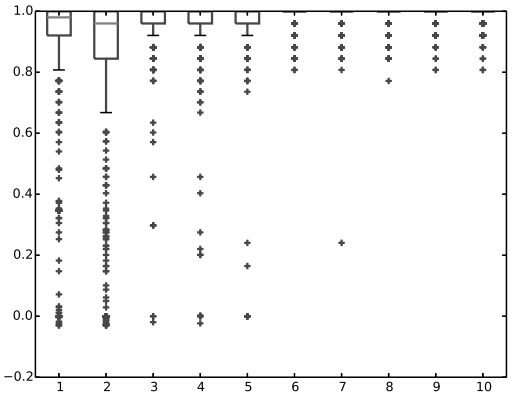


Figure 15: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for two assemblies of 20 neurons each with copy probability 0.8.

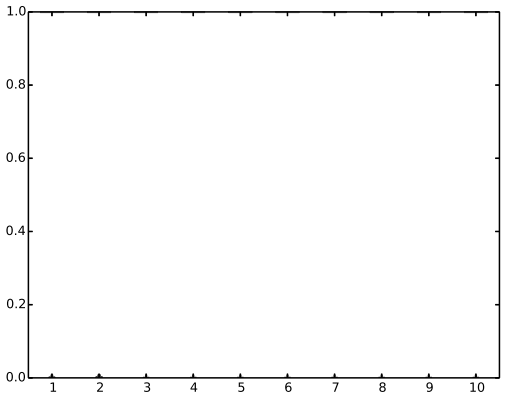


Figure 16: Classification accuracy measured using the Adjusted Rand Index for an SVM classifier for two assemblies of 20 neurons each with copy probability 1.0. Here you do not see any boxplots because for every step in the training phases the classification result was absolutely flawless and thus all scores are 1.0

has a lower accuracy when compared to a SVM. Especially with lower copy probabilities, the groups of points move closer together. The additional structural information paired with the relatively high similarity between noise and single assembly neurons may *confuse* the S^3VM. Maybe the querying strategy (uncertainty sampling) is not suitable for the training process in such cases since it will never try to validate existing, established boundaries. And finally the oracle used might be the culprit. Though the NAss library reports p-values for each neuron analyzed, we only accept a p-value of less than 0.0001 as indicator for an assembly class label. Especially when the number of coincidences becomes low (due to the low copy probability) the chance to observe similar patterns increases. Here some additional work might be needed to find out which p-values are still acceptable without getting too many false positive results.

Literatur

[1] Braune, Christian, Christian Borgelt, and Sonja Grün.: Assembly detection in continuous neural spike train data. Advances in Intelligent Data Analysis XI. Springer Berlin Heidelberg, 2012. 78-89.

[2] Braune, Christian, Stephan Besecke, and Rudolf Kruse.: Using Changes in Distribution to Identify Synchronized Point Processes. Strengthening Links Between Data Analysis and Soft Computing. Springer International Publishing, 2015. 241-248.

[3] Dice, Lee R.: Measures of the amount of ecologic association between species. Ecology 26.3 (1945): 297-302.

[4] Hebb, D.O.: The organization of behavior: a neuropsychological theory. Wiley (1949)

[5] Gieseke, Fabian, Airola, Antti, Pahikkala, Tapio and Kramer, Oliver.: Sparse Quasi-Newton Optimization for Semi-Supervised Support Vector Machines. In Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods (ICPRAM 2012). 2012, 45–54.

[6] Grün, Sonja, Denise Berger, and Christian Borgelt.: Identification of neurons participating in cell assemblies. Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on. IEEE, 2009.

[7] Berger, D., Borgelt, C., Louis, S., Morrison, A., Grün, S.: Efficient identification of assembly neurons within massively parallel spike trains. Computational intelligence and neuroscience 2010, 1 (2010)

[8] Kruse, Rudolf, Borgelt, C., Klawonn, F., Moewes, C., Steinbrecher, M., Held, P.: Computational Intelligence: A Methodological Introduction. Springer, 2013.

[9] Hodgkin, Alan L., and Andrew F. Huxley.: A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of physiology 117.4 (1952): 500.

[10] Hubert, Lawrence, and Phipps Arabie.: Comparing partitions. Journal of classification 2.1 (1985): 193-218.

[11] Huettel, Scott A., Allen W. Song, and Gregory McCarthy.: Functional magnetic resonance imaging. Vol. 1. Sunderland, MA: Sinauer Associates, 2004.

[12] Louis, Sebastien, Christian Borgelt, and Sonja Grün.: Complexity distribution as a measure for assembly size and temporal precision. Neural Networks 23.6 (2010): 705-712.

[13] Mitchell, Tom.: Machine learning. Burr Ridge, IL: McGraw Hill 45 (1997).

[14] Nádasdy, Zoltán: Spatio-Temporal Patterns in the Extracellular Recording of Hippocampal Pyramidal Cells: From Single Spikes to Spike Sequences. Rutgers, The State University of New Jersey, PhD Thesis, 1998

[15] Settles, B.: Active Learning Literature Survey, Computer Sciences Technical Report 1648. University of Wisconsin-Madison, (2009)

[16] Staude, Benjamin, and Stefan Rotter.: Higher-order correlations in non-stationary parallel spike trains: statistical modeling and inference. BMC Neuroscience 10.Suppl 1 (2009): P108.

[17] Cortes, Corinna, and Vladimir Vapnik.: Support-vector networks. Machine learning 20.3 (1995): 273-297.

Electro-Mechanical Throttle as a Benchmark Problem for Nonlinear System Identification with Friction

Salman Zaidi, Andreas Kroll

Fachgebiet Mess- und Regelungstechnik
Fachbereich Maschinenbau, Universität Kassel
Mönchebergstrasse 7, 34125 Kassel
Tel.: (0561) 804 2767
Fax: (0561) 804 2847
E-Mail: {salman.zaidi, andreas.kroll}@mrt.uni-kassel.de

Abstract

Using benchmark problems to demonstrate and compare novel methods to the work of others could be more widely adopted by the Computational Intelligence (CI) community. For this task, this article presents the electro-mechanical throttle as a benchmark problem in nonlinear system identification. Electro-mechanical throttles are standard components in Diesel and Otto combustion engines and are therefore widespread deployed. To obtain robust low-cost components, construction is kept simple. This results in significant friction and other nonlinear effects. This article describes the technical system and the modeling tasks. The two signals of the benchmarking data sets are introduced: a multisine and a "quasi" amplitude modulated pseudo random step signal for identification and model validation, respectively. Assessment criteria for model performance are defined. Results for two well know techniques of nonlinear system identification namely Piecewise Affine (PWA) modeling and Takagi-Sugeno (TS) fuzzy modeling are presented and compared.

1 Introduction

The use of bench mark problems allows engineers, researchers and scientists to compare the performance of their developed methods to the state-of-the-art methods. However, such comparisons are rarely made due to the lack of sufficient insight into methods other than the own research focus. This problem can be circumvented if benchmark problems are adopted more widely such that one can retrieve competing results from literature without having to become an expert in other methods. In fact, well-established benchmark problems are available for problems such as classification, control and modeling, to name a few. The objective of this article is to present an electro-mechanical throttle as a new benchmark

problem for system identification of nonlinear dynamic systems with friction and to promote a wider adoption for comparing alternative CI methods on throttles.

Continuous advancement of modern automobiles demands to develop efficient engine management systems to increase fuel efficiency and reduce emissions. Accurate models of associated actuators are thus required for operations like Hardware-in-the-Loop (HiL) simulation, model based control, or fault detection. Electro-mechanical throttles, one of the important mechatronic components of engine management systems, are used primarily for maintaining air-fuel ratio and combustion temperature at a prescribed level. However, the presence of friction, the nonlinear spring characteristic, and the mechanical hard stops hamper their accurate modeling. These nonlinearities mainly include the effects of dead zone, hysteresis and saturation.

Nonlinear models of throttles based on the first principles and semi-automated modeling techniques have been proposed in the literature, cf. [1],[2]. Modeling based on first principles alone cannot take into account the complex state dependent friction effects and manufacturing imperfections. Therefore, in order to obtain a mathematical model that best describes the underlying complex dynamical behavior, estimating a model based on the techniques of system identification can be more effective. This technique has the ability to provide a mathematical framework that can incorporate various uncertainties in the observed data sets for identification and test. In this connection, Vasal et al. [4] proposed a PieceWise AutoRegressive eXogenous (PWARX) model based on K-means Clustering and Multi-category Robust Linear Programming (MRLP). By taking into account of different operating regions of a throttle valve and using the techniques of classification and linear identification, Lebbal et al. [3] developed a model in which nonlinearities were considered as additive unknown inputs. Ren et al. [5] developed a computationally efficient PWA model by utilizing K-means clustering and Multi-category Support Vector Machine (M-SVM). The model was first trained for serial-parallel evaluation and was succeedingly optimized for parallel evaluation using the simplex method/algorithm.

Since, the accuracy of an estimated model depends heavily on the data, it is crucial to design efficient experiments and thus to obtain informative data sets. This article provides details of a proposed benchmark problem for nonlinear system identification. The used data sets are provided with free access on the web page of the German technical committee on Computational Intelligence [6]. As examples, this article contains two models

that are identified using the benchmark data: At first a TS fuzzy model and secondly a PWA model. The PWA model was developed by Ren et. el [5]. The rest of this article is organized as follows: Section 2 presents some typically used assessment criteria for the modeling performance. A technical description of electro-mechanical throttle is provided in Section 3 along with a brief task description. The proposed data set for benchmarking is presented in Section 4. The identification approach for TS modeling is discussed in Section 5. Section 6 provides experimental results obtained with TS and PWA models. Finally, a brief conclusion is drawn in the last section.

2 Assessment criteria for model performance

Most commonly, the approximation/prediction error is used as assessment criterion for model performance. Most significant is the result for validation/test rather than for the training data. Many different criteria are proposed as e.g. sometimes the worst case and sometimes the average deviation maybe more important. In case of benchmark problems, it is recommended to report a few widely accepted criteria such as the following ones, see [7] for a more complete overview. Given N data sets where $y(k)$ is the output of a system and $\hat{y}(k)$ the corresponding output of the model, this could be the *maximum absolute error* (MaxAE)

$$J_{\text{MaxAE}} = J_{\text{max}} = \max_{1 \le k \le N} |y(k) - \hat{y}(k)|, \tag{1}$$

and/or the *Root Mean Squared Error* (RMSE)

$$J_{\text{RMSE}} = \sqrt{\frac{1}{N} \sum_{k=1}^{N} \left(y(k) - \hat{y}(k)\right)^2}. \tag{2}$$

A relative measure is the *Normalized Mean Squared Error* (NMSE)

$$J_{\text{NMSE}} = \frac{\sum_{k=1}^{N} (y(k) - \hat{y}(k))^2}{\sum_{k=1}^{N} (y(k) - \bar{y})^2}, \tag{3}$$

It is important to differentiate between one-step-ahead and recursive model evaluation: In the first case measurements available until present time k are

used to predict the output $\hat{y}(k+1)$ one-step-ahead into the future

$$\hat{y}(k+1) = f(y(k), \ldots, y(k-n_y), u(k-\tau), \ldots, u(k-\tau-n_u)), \quad (4)$$

where n_y and n_u, respectively, is the number of lagged terms considered, τ a discrete dead-time, $y(k) \in \mathbb{R}$ and $u(k) \in \mathbb{R}$ represents the output and input of a Single-Input-Single-Output (SISO) system at the k-th instant and $\hat{y}(k+1)$ is the model prediction for the $(k+1)$-th instant. In a second case, lagged predictions are used as model inputs instead of measured data:

$$\hat{y}(k+1) = f(\hat{y}(k), \ldots, \hat{y}(k-n_y), u(k-\tau), \ldots, u(k-\tau-n_u)). \quad (5)$$

Good recursive model evaluation results are more difficult to achieve than good one-step-ahead predictions.

3 Electro-mechanical throttle

Figure 1 shows a technology scheme of a throttle. This mechatronic system consists of DC motor, gear box, return spring and throttle plate integrated in metal housing. A potentiometer is used to measure the plate's rotational position $\psi(k)$ that can take values between $10°$ (fully closed) and $90°$ (fully open), which is considered as the output $y(k)$. The motor supply is a pulse width modulated signal. Its duty cycle (in $\%$) is the system input $u(k)$. In theory, a physical model can easily be derived for this process. In practice however, the friction can e.g. be state-dependent, the spring may have nonlinear characteristics, and some parts may be made of plastics such that deformations may occur. In addition, physical properties such as inertias or spring characteristics are typically not known. A test stand that has been used for collecting data for identification and validation is illustrated in figure 2.

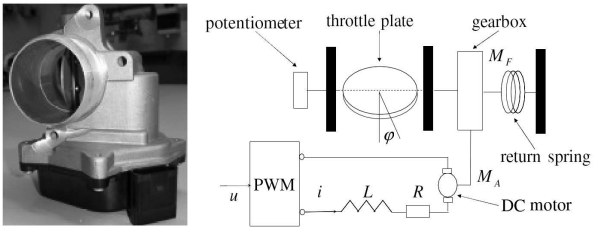


Figure 1: Electro-mechanical throttle

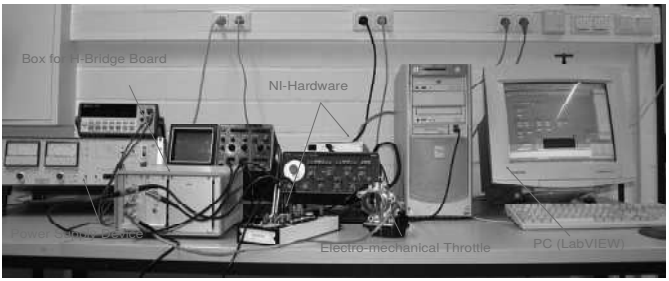


Figure 2: Test stand [2]

Filtered measurement data are recorded with $T_0 = 10$ ms from a standard automotive throttle in a laboratory setup without load (see figure 2). The data sets, which are publically available at [6] for benchmarking, consist of a multisine data set ($N_{\text{IDENT}} = 10000$) for identification and a "quasi" amplitude modulated pseudo random step (QAMPRS) data set ($N_{\text{TEST}} = 2500$) for validation. An anti-aliasing filter having cutoff frequency adjusted according to the Nyquist criterion was used to pre-process the data before sampling. Using this data, a parsimonious dynamical process model for simulation purposes is to be identified.

4 Proposed data sets for benchmarking

The multisine signal which is used for identification is displayed in figure 3. It was designed in such a way that it keeps the throttle moving within its operating range (between 10° and 90°) for as long as possible, without getting stuck at its hard mechanical stops at the opening and closing. The response of the throttle for the multisine input is shown in figure 4. A brief description of test signal design is discussed in the sequel.

A band-limited multisine signal with M harmonic components is given by

$$u(t) = \sum_{i=1}^{M} a_i \cos(\omega_i \cdot t + \phi_i) + u_{\text{offset}}, \tag{6}$$

where a_i, ω_i and ϕ_i represents the amplitude, angular frequency and phase shift of the i-th harmonic component, respectively, and u_{offset} is the offset. The duration of the multisine signal was chosen to be 100 s, as a compromise between the estimation quality of model and the computational

tractability. The system has an approximate bandwidth of 5 Hz (determined experimentally) and thus the upper frequency limit of the signal was set to be this value ($f_{\max} \approx 5$ Hz). The lower frequency was selected to be small enough to capture the friction effect in the low frequency range ($f_{\min} \approx 0.7$ Hz). In order to reduce the nonlinear distortions, only the prime harmonics were included in the frequency band [9]. The amplitudes and the offset were determined experimentally considering that they should be able to excite all important operational system characteristics as well as to avoid staying at the mechanical hard stop limits too often. In order to optimize the peak factor of input signal, initially the Schröder phases were used [8]. The Schröder phases for a multisine signal having d spectral components is given by

$$\phi_1 = 0, \phi_i = \phi_1 - i \cdot (i-1) \cdot \pi/d, 2 \leq i \leq d$$

These phases were succeedingly optimized by nonlinear optimization technique in order to minimize the peak factor. The rationale behind optimizing the peak factor of a multisine signal is that it allows to inject maximum power into the system for the given range of input amplitudes and thus increase the signal to noise ratio.

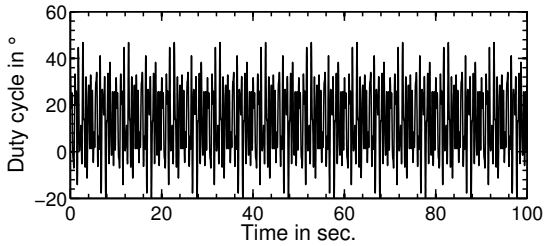


Figure 3: Multisine input signal for identification

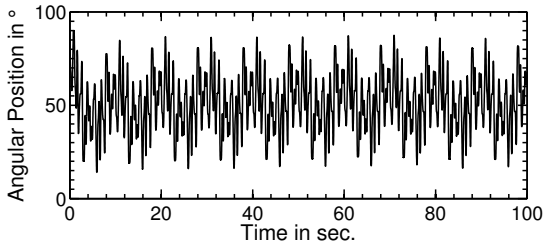


Figure 4: System output for identification

In order to test the quality of the model, a QAMPRS signal was used as shown in figure 5. This signal was designed to test the modeling performance of the throttle at various operating regions including the hard mechanical stops. It consists of a series of random pulses of different widths ranging from the input range of duty cycle from $0\ \%$ to $100\ \%$. The corresponding output for validation is shown in figure 6.

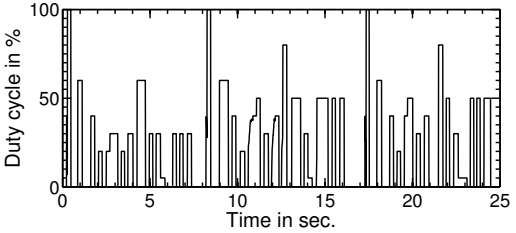


Figure 5: Quasi pseudo input random signal for validation

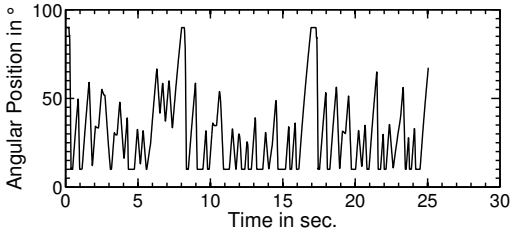


Figure 6: System output for validation

5 Identification Approach

PWA and TS models have been used for the identification of the throttle benchmark. For the details of PWA model see [5]. In the sequel, the description of TS identification is provided.

A TS fuzzy model with multidimensional reference fuzzy sets [11] and affine consequents were used in the proposed TS fuzzy modeling. Considering the SISO[1] case, the i-th fuzzy rule of the TS fuzzy model having c rules can be written as

$$R_i: \quad \text{IF } \mathbf{z} \text{ IS } \mathbf{v}_i \text{ THEN } \hat{y}_i = f_i(\mathbf{x}) \tag{7}$$

[1]Extension to MIMO case is straightforward., e.g. see [12]

with:

R_i: i-th fuzzy rule,
$\mathbf{z}$: antecedent or scheduling variable, $\mathbf{z} = [z_1, \ldots, z_{r_a}]^T \in \mathbb{R}^{r_a \times 1}$,
$\mathbf{v}_i$: i-th cluster prototype, $\mathbf{v}_i = [v_{1,i}, \ldots, v_{r_a,i}]^T \in \mathbb{R}^{r_a \times 1}$,
$\hat{y}_i$: crisp output of the i-th rule, $\hat{y}_i \in \mathbb{R}$,
f_i: affine conclusion function, $f_i(\mathbf{x}) = a_{0,i} + \sum_{j=1}^{r_c} a_{j,i} x_j$,
$\mathbf{x}$: consequent variable, $\mathbf{x} = [x_1, \ldots, x_{r_c}]^T \in \mathbb{R}^{r_c \times 1}$.

In case of NARX nonlinear dynamic systems, $\mathbf{z}$ and $\mathbf{x}$ are usually chosen as the vectors of lagged inputs and measured outputs, e.g., taking $\tau = 1$, the antecedent variable can be written as

$$\mathbf{x}(k) = [u(k-1), \ldots, u(k-n_{u_c}), y(k-1), \ldots, y(k-n_{y_c})]^T$$

with $r_c = n_{u_c} + n_{y_c}$, and $\hat{y}_i(k) = \hat{y}_i(\mathbf{x}(k))$. However, the input to the $\mathbf{z}$ and $\mathbf{x}$ can be any function of lagged inputs and outputs in general. For instance, in this research, the inputs to $\mathbf{z}$ are chosen to be $\mathbf{z}(k) = [u(k-1), y(k-1) - y(k-2)]^T$ with $r_a = 2$ as it will shown later. The degree of fulfillment for the i-th rule is determined by evaluating the i-th Membership Function (MF)

$$\mu_i(\mathbf{z}(k)) = \left[\sum_{j=1}^{c} \left(\frac{||\mathbf{z}(k) - \mathbf{v}_i||_2}{||\mathbf{z}(k) - \mathbf{v}_j||_2} \right)^{\frac{2}{\nu-1}} \right]^{-1}, \quad \nu > 1 \tag{8}$$

where ν is the fuzziness parameter, and $||.||_2$ is the Euclidean distance [2]. The final crisp output is given as the average of outputs of the c rules according to (7) weighted by their membership values as follows

$$\hat{y}(k) = \sum_{i=1}^{c} \mu_i(\mathbf{z}(k)) \hat{y}_i(\mathbf{x}(k)). \tag{9}$$

Note that the MFs defined by (8) are orthogonal, i.e. $\sum_{i=1}^{c} \mu_i(\mathbf{z}(k)) = 1$. The algorithm consists of the identification of:

1. Premise parameters, i.e. c cluster prototypes lumped into $\mathbf{v} \in \mathbb{R}^{cr_a \times 1}$, $\mathbf{v} := [\mathbf{v}_1^T, \ldots, \mathbf{v}_c^T]^T$.

2. c sets of consequent parameters of the local affine models lumped into $\mathbf{a} \in \mathbb{R}^{c(r_c+1) \times 1}$, $\mathbf{a} := [\mathbf{a}_1^T, \ldots, \mathbf{a}_c^T]^T$, where $\mathbf{a}_i = [a_{0,i}, \ldots, a_{r_c,i}]^T$.

[2] other possibilities include p-norm Minkowski (p=2, Euclidean) or Mahalanobis distance

The Fuzzy C-Means (FCM) [13] is used for the identification of the premise structure. The cluster prototypes ($\mathbf{v}_{\text{NARX}}$) are obtained by minimizing the objective function

$$\mathbf{v}_{\text{NARX}} := \mathbf{v}^* = \arg\min_{\mathbf{v}} \left(J_{\text{FCM}}(Z, P, \mathbf{v}) \right), \tag{10}$$

with

$$J_{\text{FCM}}(Z, P, \mathbf{v}) = \sum_{k=1}^{N} \sum_{i=1}^{c} \mu(\mathbf{z}(k))^{\nu} ||\mathbf{z}(k) - \mathbf{v}_i||_2^2, \tag{11}$$

where Z is the input matrix for the antecedent part,

$$Z := [\mathbf{z}(1), \ldots, \mathbf{z}(N)]^T \in \mathbb{R}^{N \times r_a},$$

and P is the partition matrix,

$$P := [\mu_i(\mathbf{z}(k))] \in \mathbb{R}^{c \times N}.$$

It is clear from the objective function that cluster prototypes are not adjusted to optimally estimate the input-output behavior of the system but to group the data.

The consequent parameters ($\mathbf{a}_{\text{NARX}} \in \mathbb{R}^{c(r_c+1)\times 1}$) are estimated globally by using the OLS method (NARX model). Denote $M_i \in \mathbb{R}^{N \times N}$, the diagonal matrix having membership grades $\mu_i(\mathbf{x}(k))$ as its k-th diagonal element with $1 \leq i \leq c$ and $1 \leq k \leq N$. Define a matrix

$$X_e := [X, \mathbf{1}] \in \mathbb{R}^{N \times (r_c+1)},$$

where X is the input matrix for the consequent part,

$$X := [\mathbf{x}(1), \ldots, \mathbf{x}(N)]^T \in \mathbb{R}^{N \times r_c},$$

and $\mathbf{1}$ is a unitary column vector in $\mathbb{R}^{N \times 1}$. Moreover, define

$$X' \in \mathbb{R}^{N \times c(r_c+1)}$$

as

$$X' := [M_1 X_e, \ldots, M_c X_e],$$

then $\mathbf{a}_{\text{NARX}}$ is calculated as

$$\mathbf{a}_{\text{NARX}} = [(X')^T X']^{-1} (X')^T \mathbf{y}.$$

The premise and consequent parameters can be lumped into $\boldsymbol{\theta}_{\text{NARX}}$ as follows

$$\boldsymbol{\theta}_{\text{NARX}} := [\mathbf{v}_{\text{NARX}}^T, \mathbf{a}_{\text{NARX}}^T]^T \in \mathbb{R}^{c(r_a+r_c+1)\times 1}. \tag{12}$$

Good evaluation properties of NOE or parallel models are important for simulation or for long-range predictions, e.g., in the context of model-based predictive control [14]. The matlab function `lsqnonlin` was used for determining optimal cluster prototypes and local model parameters for parallel mode evaluation. This function uses a trust-region-reflective algorithm based on the interior-reflective Newton method [15],[16]. Denoting the lumped parameter vector for NOE model as $\boldsymbol{\theta}_{\text{NOE}} \in \mathbb{R}^{c(r_a+r_c+1)\times 1}$, $\boldsymbol{\theta}_{\text{NOE}} := [\mathbf{v}_{\text{NOE}}^T, \mathbf{a}_{\text{NOE}}^T]^T$. It is obtained by the minimizing the Mean Squared Error (MSE) of NOE model as follows

$$\boldsymbol{\theta}_{\text{NOE}} := \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{k=1}^{N}(y(k) - \hat{y}_{\text{NOE}}(\boldsymbol{\theta}, k))^2. \tag{13}$$

The starting value of $\boldsymbol{\theta}$ is chosen to be equal to $\boldsymbol{\theta}_{\text{NARX}}$.

6 Experimental Results

A piecewise affine and a Takagi-Sugeno model were chosen to have the same general structure; both use $c = 8$ local models. In case of TS, the value of fuzziness parameter is chosen to be $\nu = 1.1$. For having a parsimonious model, the value of c was selected based on the knee point of J_{RMSE} of the NOE model, after which no considerable improvement in model performance was observed. The value of ν was selected based as per suggestion in [10]. The antecedent/scheduling variable for partitioning is chosen to be $\mathbf{z}(k) = [u(k-1), y(k-1) - y(k-2)]^T$, and the local model structure for NARX (NOE correspondingly) is:

$$\begin{aligned} \hat{y}_i(k) &= \mathbf{a}_i^T \cdot [1, \mathbf{x}(k)^T]^T, \\ \mathbf{x}(k) &= [u(k-1), y(k-1), y(k-2)]^T \end{aligned}$$

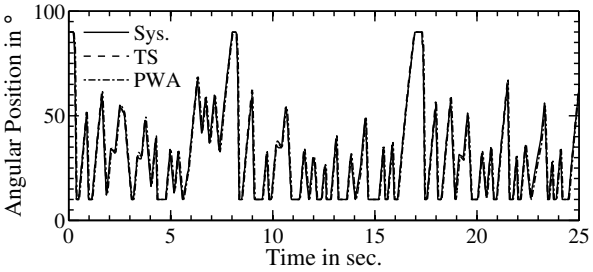


Figure 7: System and model outputs from TS and PWA

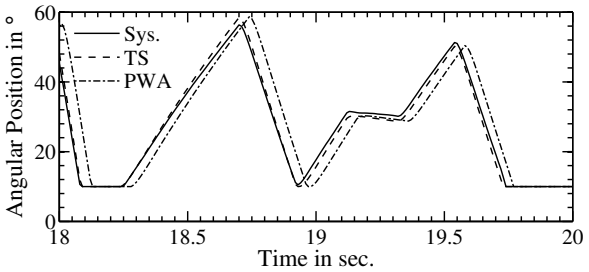


Figure 8: System and model outputs from TS and PWA (between 18 and 20 sec.)

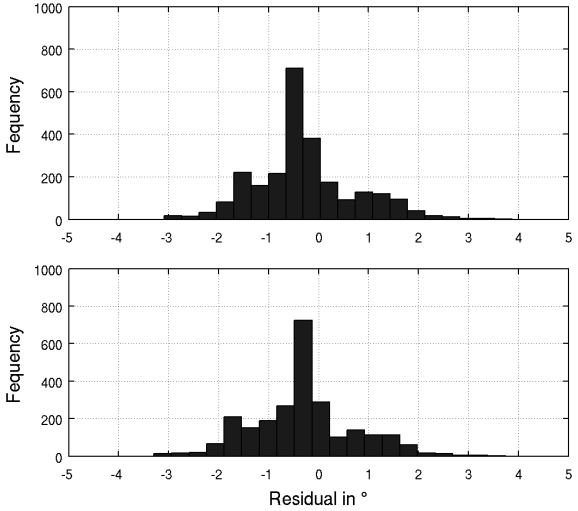


Figure 9: Frequency plot of residuals from recursive model evaluation for PWA (top) and TS model (bottom) for test data set

The major difference between PWA and TS is that the PWA model uses crisp polytope partitions, but the TS model uses a soft partitioning with the membership functions of fuzzy-c-means type. For the PWA model, the Hybrid Identification Toolbox (HIT) [17] is used . First clustering is performed in 2D space of $\mathbf{z}(k)$ for classification of data points, followed by the formation of 8 polytopes in 3D space of $\mathbf{x}(k)$ using the technique of Proximal Support Vector Classification (PSVC), such that they form the partition of the entire admissible space of $\mathbf{x}(k)$. The number of pairwise-adjacencies are found out to be 24 (out of total 39 polytopes). To have a fair comparison, these adjacencies were counted only once in calculating the number of partitioning parameters of the PWA model. The model performance obtained in recursive model evaluation is recorded in table 1. From the table, it is obvious that in the underlying modeling scenario, the TS model is well parsimonious in terms of number of partitioning parameters while providing the similar modeling performance. The responses of TS and PWA model for the test data set are illustrated in figure 7 and figure 8. The corresponding frequency plots of the residuals on the test data set are shown in figure 9.

Model	**Data set**	**Criteria**			**Number of parameters**	
		$\mathbf{J_{MaxAE}}$ **in** $^\circ$	$\mathbf{J_{RMSE}}$ **in** $^\circ$	$\mathbf{J_{NRMSE}}$ **in** $^\circ$	**Partitioning**	**Local Models**
PWA	Train	3.83	1.45	0.96	68	32
	Test	3.92	1.03	0.94		
TS	Train	4.47	1.42	0.91	16	32
	Test	3.74	1.06	0.95		

Table 1: Results for recursive model evaluation of PWA and TS throttle model for training and test data set

7 Conclusion

A wider engagement in testing and demonstrating novel methods on benchmark problems is a rewarding undertaking for the individual researcher and for the community: Well-defined identification can be solved with moderate efforts while permitting to compare own results with results from other subject matter experts. For this reason, a complete example of electromechanical throttle is presented in this article as a benchmark of a nonlinear systems with friction. Two test signals are proposed for identification

and validation and the results of modeling using PWA and TS fuzzy modeling are reported in this article.

References

[1] Scattolini, R., Siviero, C., Mazzucco, M., Ricci, S., Poggio, L., Rossi, C.: Modeling and identification of an electromechanical internal combustion engine throttle body. *Control Engineering Practice*. vol. 5(9), pp. 1253–1259. 1997.

[2] Ren, Z., Kroll, A., Sofsky, M., Laubenstein, F.: On methods for automated modeling of dynamic systems with friction and their application to electro-mechanical throttles. In: *49th IEEE Conference on Decision and Control*. pp. 7637–7642. Atlanta, GA, USA. 2010.

[3] Lebbal, M., Chafouk, H., Hoblos, G., Lefebvre, D.: Modeling and identification of non-linear systems by a multimodel approach : application to a throttle valve. *International Journal of Information and Systems Sciences*. vol. 3(1), pp. 67–87. 2006.

[4] Vasak, M., Mladenovic, L., Peric, N.: Clustering-based identification of a piecewise affine electronic throttle model. In: *31st Annual Conference of IEEE, IECON*. pp. 177–182. 2005.

[5] Ren, Z., Kroll, A., Sofsky, M., Laubenstein, F.: On identification of piecewise-affine models for systems with friction and its application to electro-mechanical throttles. In: *16th IFAC Symposium on System Identification Brussels, SysID*. pp. 1395–1400. Brussel, Belgium. 2012.

[6] GMA Benchmarking.: `http://www.rst.e-technik.tu-dortmund.de/cms/de/Veranstaltungen/GMA-Fachausschuss/Benchmark/index.html`. [Last visit September 05, 2014]. 2014.

[7] Kroll, A., Schulte, H.: Benchmark problems for nonlinear system identification and control using Soft Computing methods: Need and overview. *Applied Soft Computing Journal*. DOI=`http://dx.doi.org/10.1016/j.asoc.2014.08.034`. 2014.

[8] Schröder, M. R.: Synthesis of low-peak-factor signals and binary sequences with low autocorrelation. *IEEE Trans. Information Theory (Corresp.)*. vol. IT-16, pp. 85–89. 1970.

[9] Rees, D.: Automatic testing of dynamic systems using multifrequency signals and discrete Fourier transform. In: *Proc. IEE International Conf. New Developments in Automatic Testing*. pp. 24–27. London. 1977.

[10] Kroll, A.: On choosing the fuzziness parameter for identifying TS models with multidimensional membership functions. *Journal of Artificial Intelligence and Soft Computing Research*. vol. 1(4), pp. 283–300. 2011.

[11] Kroll, A.: Identification of functional fuzzy models using multidimensional reference fuzzy sets. *Fuzzy Sets and Systems*. vol. 80(2), pp.149–158. 1996.

[12] Babuska, R., Roubos, J.A, Verbruggen, H.: Identification of MIMO systems by input-output TS fuzzy models. In: *Proc. Fuzzy Systems. IEEE World Congress on Computational Intelligence*. vol. 1, pp. 657–662. 1998.

[13] Bezdek, J. C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press. 1981.

[14] Jelali, M., Kroll, A.: *Hydraulic Servo-systems: Modelling, Identification and Control*. Springer. 2002.

[15] Coleman, T. F., Li, Y.: On the Convergence of Reflective Newton Methods for Large-Scale Nonlinear Minimization Subject to Bounds. *Mathematical Programming*. vol. 67(2), pp. 189–224. 1994.

[16] Coleman, T. F., Li, Y.: An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds. *SIAM Journal on Optimization*. vol. 6, pp. 418–445. 1996.

[17] Hybrid Identification Toolbox: `http://sisdin.unipv.it/lab/personale/pers_hp/ferrari/HIT_toolbox.php`. [Last visit September 05, 2014].

Vergleich von Ansätzen zur approximativen T-S Modellierung abgetasteter nichtlinearer Systeme

Klaus Albert, Christian Dengler

Lehrstuhl für Regelungstechnik (Prof. Dr. B. Lohmann), TU München
Boltzmannstr. 15, 85748 Garching
Tel.: (089) 289-15681
Fax: (089) 289-15653
E-Mail: klaus.albert@tum.de

1 Problemstellung

Für die Regelung nichtlinearer Systeme werden heutzutage vorwiegend digitale Rechensysteme (z. B. Mikrocontroller) verwendet. Hierbei wird das zeitkontinuierliche System abgetastet und es entsteht ein zeitdiskretes, nichtlineares Verhalten.

Auf Basis zeitdiskreter Takagi-Sugeno (T-S) Modelle kann sowohl der Reglerentwurf als auch der Stabilitätsnachweis für abgetastete nichtlineare Systeme durchgeführt werden. Um eine Übertragbarkeit der Ergebnisse auf das Originalsystem gewährleisten zu können, muss das zeitdiskrete T-S Modell dessen Dynamik in ausreichender Güte approximieren [1].

Ausgehend von einem zeitkontinuierlichen, nichtlinearen Modell welches äquidistant abgetastet wird, gibt es in der Literatur verschiedene Methoden zur Erstellung eines zeitdiskreten T-S Modells [1, 2, 3]. Eine Auflistung und ein Vergleich der verschiedenen Methoden anhand eines Benchmarkbeispiels, mit denen ein zeitdiskretes T-S Modell für abgetastete nichtlineare Systeme erstellt werden kann, wurde nach dem Kenntnisstand der Autoren bisher noch nicht veröffentlicht.

In diesem Beitrag werden drei verschiedene Methoden vorgestellt und anhand des Benchmarkbeispiels *Inverses Pendel mit Wagen* in zwei Untersuchungen miteinander verglichen. Als Vergleichskriterium wird die Abweichung des berechneten Zustands der erzeugten zeitdiskreten T-S Modelle nach einem Abtastschritt zu dem berechneten Zustand des abgetasteten nichtlinearen Modells verwendet. Die erste vorgestellte Methode identifiziert die T-S Modelle mit Hilfe von Datensätzen, die durch die numerische Integration des nichtlinearen, zeitkontinuierlichen Modells über die Abtastzeit erzeugt wurden. Basierend auf einem zeitkontinuierlichen T-S Mo-

dell des nichtlinearen Systems wird in der zweiten Methode über die Zeitdiskretisierung der einzelnen T-S Teilmodelle ein zeitdiskretes T-S Modell erzeugt. In der dritten Methode wird zuerst das nichtlineare Modell zeitdiskretisiert und anschließend ein T-S Modell erzeugt. Nach der zeitdiskreten T-S Modellierung des Benchmarkbeispiels wird in den beiden Untersuchungen zum einen der Einfluss der Abtastzeit, zum anderen der Einfluss der Anzahl der T-S Teilmodelle auf den Zustandsfehler hin untersucht.

Ziel ist es, basierend auf dem Beispiel die Vor- und Nachteile der Methoden aufzuzeigen und somit Anhaltspunkte für die Wahl der Modellierungsmethode zu erleichtern.

2 Einleitung

Sind die physikalischen Zusammenhänge und Parameter der Strecke bekannt, kann eine Modellbildung erfolgen und aus den resultierenden nichtlinearen Differentialgleichungen ein T-S Modell formuliert werden. Bei unbekannten oder sehr komplexen physikalischen Zusammenhängen bietet sich die Erstellung des T-S Modells über eine Systemidentifikation, z. B. anhand von Messdaten, an.

Im Folgenden wird stets angenommen, dass die Beschreibung des Systems als zeitkontinuierliche, nichtlineare Differentialgleichung

$$\dot{\mathbf{x}}(t) = {}^{c}\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{1}$$

mit dem Zustandvektor $\mathbf{x}(t) \in \mathbb{R}^n$ und dem Eingangsvektor $\mathbf{u}(t) \in \mathbb{R}^m$ vorliegt und diese unendlich oft differenzierbar ist. Um später die Verwechslung von zeitkontinuierlichen und zeitdiskreten Funktionen bzw. bei T-S Modellen deren Matrizen zu vermeiden, indiziert jeweils ein hochgestelltes c vor dem Symbol die zeitkontinuierliche Formulierung und ein d entsprechend die zeitdiskrete Variante.

Unter der Annahme, dass das zeitkontinuierliche System äquidistant mit der Abtastzeit $T > 0$ abgetastet wird, erhält man ein zeitdiskretes, nichtlineares System

$$\mathbf{x}[k+1] = {}^{d}\mathbf{f}(\mathbf{x}[k], \mathbf{u}[k]), \tag{2}$$

wobei an den Abtastzeitpunkten kT mit $k \in \mathbb{Z}$

$$\mathbf{x}[k] := \mathbf{x}(kT) \tag{3}$$

gilt.

In technischen Systemen entspricht die Abtastung des zeitkontinuierlichen Systems oft dem Verhalten von Haltegliedern nullter Ordnung [4], sodass die Stellgrößen $\mathbf{u}(t)$ innerhalb der Abtastzeit T konstant sind und der Zusammenhang

$$\mathbf{u}[k] := \mathbf{u}(t), \quad \forall\, t \in [kT, (k+1)T[\tag{4}$$

gilt.

Ziel ist es nun, ein zeitdiskretes T-S Modell

$$\mathbf{x}[k+1] = \sum_{i=1}^{r} {}^{d}h_i(\mathbf{z}[k]) \left({}^{d}\mathbf{A}_i \mathbf{x}[k] + {}^{d}\mathbf{B}_i \mathbf{u}[k] \right) \tag{5}$$

mit den Matrizen ${}^{d}\mathbf{A}_i \in \mathbb{R}^{n\times n}$, ${}^{d}\mathbf{B}_i \in \mathbb{R}^{n\times m}$ der r linearen Teilsysteme bzw. ein zeitdiskretes T-S Modell

$$\mathbf{x}[k+1] = \sum_{i=1}^{r} {}^{d}h_i(\mathbf{z}[k]) \left({}^{d}\mathbf{A}_i \mathbf{x}[k] + {}^{d}\mathbf{B}_i \mathbf{u}[k] + {}^{d}\mathbf{a}_i \right) \tag{6}$$

mit dem zusätzlichen affinen Term ${}^{d}\mathbf{a}_i \in \mathbb{R}^n$ zu erstellen, welches das äquidistant abgetastete nichtlineare System (2) im interessierenden Bereich

$$\begin{aligned} \mathbf{x}[k] &\in \mathbf{X} \subset \mathbb{R}^n, \\ \mathbf{u}[k] &\in \mathbf{U} \subset \mathbb{R}^m \end{aligned} \tag{7}$$

in ausreichender Güte approximiert. Die einzelnen T-S Teilmodelle werden über die skalaren, nichtlinearen Funktionen ${}^{d}h_i(\mathbf{z}[k])$ mit dem Prämissenvektor $\mathbf{z}[k] \in \mathbb{R}^l$ im T-S Modell (5) bzw. (6) gewichtet und summiert [5]. Die Gewichtungsfunktionen ${}^{d}h_i$ (auch *Fuzzy-Basis-Funktionen* genannt [6]) müssen bei der Modellierung so gewählt werden, dass die konvexe Summeneigenschaft

$$\sum_{i=1}^{r} {}^{d}h_i(\mathbf{z}[k]) = 1, \quad {}^{d}h_i(\mathbf{z}[k]) \geq 0 \;\; \forall\, h_i \tag{8}$$

erfüllt ist. Der Prämissenvektor $\mathbf{z}[k]$ enthält hierbei die Zustands- und Eingangsgrößen, die zur Berechnung der Gewichtungsfunktionen ${}^{d}h_i$ benötigt werden.

Für die Erstellung eines zeitdiskreten T-S Modells sind verschiedene Methoden möglich, welche in Abschnitt 3 vorgestellt und in Abschnitt 4 anhand des Benchmarkbeispiels *Inverses Pendel mit Wagen* verglichen werden. Abschließend werden in Abschnitt 5 die wesentlichen Punkte des Beitrags zusammengefasst und ein Ausblick für weitere Arbeiten gegeben.

3 Modellierungsansätze für zeitdiskrete T-S Modelle

3.1 Zeitdiskrete T-S Modellierung über Identifikation (*T-S IDENT*)

Folgend wird nun eine Methode vorgestellt, welche mittels Daten ein zeitdiskretes T-S Modell (6) identifiziert. Die Methode wird folgend im Text als (*T-S IDENT*) bezeichnet. Der Ansatz ist hierbei, auf ein (analytisches) Zeitdiskretisierungsverfahren für das nichtlineare Modell bzw. die linearen T-S Teilmodelle zu verzichten und ein zeitdiskretes T-S Modell über ein erprobtes Identifizierungsverfahren zu bestimmen.

Die meisten Methoden, die mittels Daten ein T-S Modell identifizieren, gehen von einer stark beschränkten Anzahl an Datensätzen aus, die für die Identifizierung des zeitdiskreten T-S Modells zur Verfügung stehen [6]. Ist jedoch ein zeitkontinuierliches, nichtlineares Modell (2) der Regelstrecke vorhanden, kann eine beliebige Anzahl an Datensätzen zur Identifizierung durch numerische Integration des Modells (1) über die Abtastzeit T erzeugt werden. In der vorgestellten Methode wird der Vorteil eines vorhandenen nichtlinearen Modells (1) ausgenutzt, wodurch sich die Identifizierung deutlich vereinfacht und über die Parameterschätzung mittels der Summe der kleinsten Fehlerquadrate erfolgen kann. Folgend wird die Identifizierung von zeitdiskreten T-S Modellen (6) mit affinen Termen erläutert.

Zuerst wird hierfür die Anzahl r an T-S Teilmodellen in (6) sowie die Stützstellen $\mathbf{x}_{0,i}$,$\mathbf{u}_{0,i}$ der Teilmodelle festgelegt. Anschließend kann die Wahl der Gewichtungsfunktionen ${}^{d}h_i(\mathbf{z}[k])$ und des Prämissenvektors $\mathbf{z}[k]$ unter Berücksichtigung der konvexen Summeneigenschaft (8) erfolgen.

Um die für die Identifizierung notwendigen Daten zu berechnen, werden im Bereich (7) des T-S Modells gleichmäßig verteilte Zustände $\mathbf{x}_{0,j}$ und Eingangswerte $\mathbf{u}_{0,j}$ erzeugt. Die Anzahl p der generierten Datensätze

$$\begin{aligned} \mathbf{x}_j(kT) = \mathbf{x}_j[k] &= \mathbf{x}_{0,j} \quad j = 1 \dots p \\ \mathbf{u}_j(kT) = \mathbf{u}_j[k] &= \mathbf{u}_{0,j} \end{aligned} \tag{9}$$

sollte hierbei die Anzahl r der zu identifizierenden T-S Teilmodelle in (6) um Größenordnungen überschreiten. Anschließend wird mit Hilfe eines numerischen Integrationsverfahrens das nichtlineare, zeitkontinuierliche System (1) für jeden Datensatz j über die Abtastzeit T integriert, wodurch man den jeweils zugehörigen zeitdiskreten Zustand $\mathbf{x}_j[k+1]$ erhält. Die erzeugten Daten können nun zur Identifizierung des zeitdiskreten T-S Modells verwendet werden.

Aufgrund der einfachen Anwendung wird folgend die Methode der kleinsten Fehlerquadrate [6] zur Bestimmung der Matrizen ${}^{d}\mathbf{A}_i$, ${}^{d}\mathbf{B}_i$ bzw. des Vektors ${}^{d}\mathbf{a}_i$ der r T-S Teilmodelle angewandt: Zuerst wird hierfür das zeitdiskrete T-S Modell (6) umgeformt. Anschließend kann für jeden Datensatz j das T-S Modell mit

$$\mathbf{x}_j^T[k+1] = \sum_{i=1}^{r} {}^{c}h_i(\mathbf{z}_j[k]) \begin{bmatrix} \mathbf{x}_j^T[k] & \mathbf{u}_j^T[k] & 1 \end{bmatrix} \begin{bmatrix} {}^{d}\mathbf{A}_i^T \\ {}^{d}\mathbf{B}_i^T \\ {}^{d}\mathbf{a}_i^T \end{bmatrix} \tag{10}$$

dargestellt werden. Durch die Ausführung der Summierung wird (10) in die Form

$$\mathbf{x}_j^T[k+1] = \boldsymbol{\Upsilon}_j \boldsymbol{\Theta} + \mathbf{E}_j \tag{11}$$

gebracht, wobei die Matrix

$$\boldsymbol{\Theta} = \left[{}^{d}\mathbf{A}_1, \cdots, {}^{d}\mathbf{A}_r, {}^{d}\mathbf{B}_1, \cdots, {}^{d}\mathbf{B}_r, {}^{d}\mathbf{a}_1, \cdots, {}^{d}\mathbf{a}_r \right]^T \in \mathbb{R}^{((n+m+1)r)\times n} \tag{12}$$

die zu identifizierenden Matrizen ${}^{d}\mathbf{A}_i$, ${}^{d}\mathbf{B}_i$ und Vektoren ${}^{d}\mathbf{a}_i$ enthält. Die Matrix

$$\begin{aligned} \boldsymbol{\Upsilon}_j = \big[& {}^{d}h_1(\mathbf{z}_j[k])\mathbf{x}_j[k], \ldots, {}^{d}h_r(\mathbf{z}_j[k])\mathbf{x}_j[k], \\ & {}^{d}h_1(\mathbf{z}_j[k])\mathbf{u}_j[k], \ldots, {}^{d}h_r(\mathbf{z}_j[k])\mathbf{u}_j[k], {}^{d}h_1(\mathbf{z}_j[k]), \ldots, {}^{d}h_r(\mathbf{z}_j[k]) \big] \end{aligned} \tag{13}$$

der Dimension $\boldsymbol{\Upsilon}_j \in \mathbb{R}^{n\times 3r}$ besteht aus den mit ${}^{d}h_i(\mathbf{z}_j[k])$ gewichteten Zuständen $\mathbf{x}_j[k]$ und Eingangswerten $\mathbf{u}_j[k]$ für jedes der r T-S Teilmodelle. Der Fehlervektor $\mathbf{E}_j$ enthält die zu minimierende Abweichung $\mathbf{E}_j = \mathbf{x}_j^T[k+1] - \boldsymbol{\Upsilon}_j\boldsymbol{\Theta}$.

Für die p Datensätze kann nun die Gleichung

$$\begin{bmatrix} \mathbf{x}_1^T[k+1] \\ \vdots \\ \mathbf{x}_p^T[k+1] \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Upsilon}_1 \\ \vdots \\ \boldsymbol{\Upsilon}_p \end{bmatrix} \boldsymbol{\Theta} + \begin{bmatrix} \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_p \end{bmatrix} \tag{14}$$

aufgestellt und abschließend die Methode der kleinsten Fehlerquadrate angewandt werden [6]. Als Lösung erhält man die Matrix

$$\boldsymbol{\Theta}_{opt} = \arg\min_{\theta} \sum_{j=1}^{p} \mathbf{E}_j^T \mathbf{E}_j \tag{15}$$

aus der die Matrizen ${}^{d}\mathbf{A}_i$, ${}^{d}\mathbf{B}_i$ und Vektoren ${}^{d}\mathbf{a}_i$ der T-S Teilmodelle anschließend extrahiert werden können.

3.2 Zeitdiskrete T-S Modellierung über ein kontinuierliches T-S Modell (*T-S K2D*)

Die als *T-S K2D* bezeichnete Methode verwendet ein zeitkontinuierliches T-S Modell als Zwischenschritt, um anschließend ein zeitdiskretes T-S Modell zu erzeugen. Zuerst wird in diesem Abschnitt auf die Modellierung eines zeitkontinuierlichen T-S Modells eingegangen und darauf aufbauend dessen Zeitdiskretisierung erläutert.

Für nichtlineare Modelle (1) kann in vielen Fällen direkt ein T-S Modell

$$\dot{\mathbf{x}}(t) = \sum_{i=1}^{r} {}^{c}h_i(\mathbf{z}(t)) \left({}^{c}\mathbf{A}_i \mathbf{x}(t) + {}^{c}\mathbf{B}_i \mathbf{u}(t) \right) \tag{16}$$

mit dem *Sectornonlinearity*-Ansatz [5] erstellt werden. Die Anzahl der r linearen Teilmodelle des T-S Modells ergeben sich mit $r = 2^{l_N}$ direkt aus der Anzahl der l_N Nichtlinearitäten, die in das nichtlineare Modell (1) eingehen. Die einzelnen Teilmodelle werden über die skalaren, nichtlinearen Funktionen ${}^{c}h_i(\mathbf{z}(t))$ mit dem Prämissenvektor $\mathbf{z}(t) \in \mathbb{R}^l$ gewichtet und summiert. Die Funktionen ${}^{c}h_i(\mathbf{z}(t))$ folgen direkt aus der Umformung von (1) in (16) mit dem *Sectornonlinearity*-Ansatz. Innerhalb des bei der Erstellung gewählten Sektors entspricht das dynamische Verhalten des T-S Modells (16) exakt dem dynamischen Verhalten des originalen, nichtlinearen Modells (1) [7].

Bei komplexen Modellen mit vielen Nichtlinearitäten ist die Verwendung des *Sectornonlinearity*-Ansatz oft nicht möglich. Alternativ kann in diesen Fällen ein approximatives, zeitkontinuierliches T-S Modell

$$\dot{\mathbf{x}}(t) = \sum_{i=1}^{r} {}^{c}h_i(\mathbf{z}(t)) \left({}^{c}\mathbf{A}_i \mathbf{x}(t) + {}^{c}\mathbf{B}_i \mathbf{u}(t) + {}^{c}\mathbf{a}_i \right) \tag{17}$$

erstellt werden. Die Matrizen der affinen Teilmodelle werden durch die Linearisierung

$${}^{c}\mathbf{A}_i = \left. \frac{\partial {}^{c}\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))}{\partial \mathbf{x}} \right|_{\mathbf{x}_{0,i}, \mathbf{u}_{0,i}}, \; {}^{c}\mathbf{B}_i = \left. \frac{\partial {}^{c}\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))}{\partial \mathbf{u}} \right|_{\mathbf{x}_{0,i}, \mathbf{u}_{0,i}} \tag{18}$$

des zeitkontinuierlichen nichtlinearen Systems (1) an ausgewählten Punkten $\mathbf{x}_{0,i}, \mathbf{u}_{0,i}$ im Zustandsraum berechnet. Um die nichtlineare Systemdynamik zu approximieren, ist im Allgemeinen ein zusätzlicher affiner Term ${}^{c}\mathbf{a}_i$ mit

$${}^{c}\mathbf{a}_i = {}^{c}\mathbf{f}(\mathbf{x}_{0,i}, \mathbf{u}_{0,i}) - {}^{c}\mathbf{A}_i \mathbf{x}_{0,i} - {}^{c}\mathbf{B}_i \mathbf{u}_{0,i} \tag{19}$$

in den Teilmodellen zu berücksichtigen. Der Prämissenvektor $\mathbf{z}(t) \in \mathbb{R}^l$ enthält die l Zustands- und Eingangsgrößen, die für die Erstellung der affinen Teilmodelle variiert wurden und für die Berechnung der Interpolation zwischen den Teilmodellen benötigt werden. Die Interpolation erfolgt über die skalaren, nichtlinearen Gewichtungsfunktionen ${}^c h_i(\mathbf{z}(t))$, wobei diese die konvexe Summeneigenschaft

$$\sum_{i=1}^{r} {}^c h_i(\mathbf{z}(t)) = 1, \quad {}^c h_i(\mathbf{z}(\mathbf{t})) \geq 0 \ \ \forall \ h_i \tag{20}$$

erfüllen müssen.

Ausgehend von dem zeitkontinuierlichen T-S-Modell (16) bzw. (17) kann eine Zeitdiskretisierung des T-S-Modells erfolgen. Im Allgemeinen muss das gesamte zeitkontinuierliche T-S Modell zeitdiskretisiert werden, was jedoch auf eine nichtlineare, zeitdiskrete Funktion führt und nicht auf ein zeitdiskretes T-S Modell gemäß (5) bzw. (6).

Unter der Annahme, dass die Zeitableitungen der Gewichtungsfunktionen

$${}^c\dot{h}_i(\mathbf{z}(t)) \approx 0 \quad i = 1...r \tag{21}$$

vernachlässigbar klein sind, können die einzelnen linearen bzw. affinen Teilmodelle unabhängig voneinander zeitdiskretisiert werden. Die zeitdiskreten Matrizen lassen sich mit den aus der linearen Regelungstheorie bekannten Zusammenhängen

$$\begin{aligned} {}^d\mathbf{A}_i &= e^{{}^c\mathbf{A}_i T} && i = 1...r \\ {}^d\mathbf{B}_i &= (e^{{}^c\mathbf{A}_i T} - \mathbf{I})\ {}^c\mathbf{A}_i^{-1}\ {}^c\mathbf{B}_i \\ {}^d\mathbf{a}_i &= (e^{{}^c\mathbf{A}_i T} - \mathbf{I})\ {}^c\mathbf{A}_i^{-1}\ {}^c\mathbf{a}_i \end{aligned} \tag{22}$$

analytisch berechnen, sofern $\det({}^c\mathbf{A}_i) \neq 0$ ist [4]. Andernfalls kann durch eine Reihenentwicklung die Inversion der Matrix ${}^c\mathbf{A}_i$ vermieden und die zeitdiskreten Matrizen mit

$$\mathbf{S} = T \sum_{\nu=0}^{N} A^\nu \frac{T^\nu}{(\nu+1)!} \tag{23}$$

$$\begin{aligned} {}^d\mathbf{A}_i &= \mathbf{I} + \mathbf{S}\ {}^c\mathbf{A}_i \\ {}^d\mathbf{B}_i &= \mathbf{S}\ {}^c\mathbf{B}_i \\ {}^c\mathbf{a}_i &= \mathbf{S}\ {}^c\mathbf{a}_i \end{aligned} \tag{24}$$

angenähert werden [4]. Die Zahl N bestimmt hierbei den Abbruchfehler der Reihenentwicklung $\mathbf{S}$ und somit die Genauigkeit der Matrizen ${}^d\mathbf{A}_i$, ${}^d\mathbf{B}_i$ und Vektoren ${}^d\mathbf{a}_i$.

Abschließend werden die zeitkontinuierlichen Gewichtungsfunktionen durch zeitdiskrete ersetzt, sodass

$$^{d}h(\mathbf{z}[k]) = {}^{c}h(\mathbf{z}(kT)) \quad \forall t \in [kT, (k+1)T[\tag{25}$$

gilt.

Es wird abschließend nochmals darauf hingewiesen, dass der aufgezeigte Ansatz zur Erzeugung eines zeitdiskreten T-S Modells nur durchgeführt werden darf, wenn die Annahme zutrifft, dass die Zeitableitung $^{c}\dot{h}_i(\mathbf{z}(t))$ vernachlässigbar gering ist, was oftmals nur bei kleinen Abtastzeiten der Fall ist.

3.3 Zeitdiskrete T-S Modellierung über die Zeitdiskretisierung des nichtlinearen Modells (*T-S NL2D*)

Bei dem Ansatz *T-S K2D* wird zuerst aus dem zeitkontinuierlichen nichtlinearen Modell ein zeitkontinuierliches T-S Modell erstellt, welches anschließend zeitdiskretisiert wird. Alternativ zu der Methode *T-S K2D* kann jedoch auch zuerst das nichtlineare Modell (1) zeitdiskretisiert (siehe z.B. [3]) und aus der resultierenden zeitdiskreten nichtlinearen Funktion ein zeitdiskretes T-S Modell erstellt werden. Diese als *T-S NL2D* benannte Vorgehensweise, wird nun näher erläutert.

Zuerst erfolgt auch bei dieser Methode eine Rasterung des Zustandsraums, sodass die Zustände $\mathbf{x}_{0,i}$ und die Eingangswerte $\mathbf{u}_{0,i}$, an denen ein zeitdiskretes T-S Teilmodell berechnet werden soll, festgelegt sind.

Die anschließende Zeitdiskretisierung des nichtlinearen Modells (1) kann durch beliebige Zeitdiskretisierungsverfahren wie z. B. Euler, Heun oder eine Lie-Reihe [3, 8], erfolgen. Zur Bestimmung der Matrizen $^{d}\mathbf{A}_i$, $^{d}\mathbf{B}_i$ und Vektoren $^{d}\mathbf{a}_i$ sind die Jacobi-Matrizen der mit den Zeitdiskretisierungsverfahren berechneten Funktion notwendig. Neben Finite-Differenzenverfahren zur numerischen Approximation der Jacobi-Matrizen existieren eine Vielzahl weiterer Verfahren (siehe [8, 9]) mit denen die Jacobi-Matrizen mit hoher Genauigkeit berechnet werden können. Folgend wird beispielhaft die Berechnung der Jacobi-Matrizen und somit der Matrizen

der T-S Teilmodelle über die Lösung der Differentialgleichung

$$\frac{d}{d\tau}\begin{bmatrix}\boldsymbol{\xi}_{0,\mathbf{x}}(\tau)\\ \boldsymbol{\xi}_{0,\mathbf{u}}(\tau)\\ \boldsymbol{\xi}_1(\tau)\\ \boldsymbol{\xi}_2(\tau)\\ \vdots\\ \boldsymbol{\xi}_q(\tau)\end{bmatrix} = \begin{bmatrix} {}^c\mathbf{f}(\boldsymbol{\xi}_{0,\mathbf{x}}(\tau),\boldsymbol{\xi}_{0,\mathbf{u}}(\tau))\\ \mathbf{0}\\ \frac{\partial}{\partial(\mathbf{x}(\tau),\mathbf{u}(\tau))}\,{}^c\mathbf{f}(\boldsymbol{\xi}_{0,\mathbf{x}}(\tau),\boldsymbol{\xi}_{0,\mathbf{u}}(\tau))\;\boldsymbol{\xi}_1(\tau)\\ \frac{\partial}{\partial(\mathbf{x}(\tau),\mathbf{u}(\tau))}\,{}^c\mathbf{f}(\boldsymbol{\xi}_{0,\mathbf{x}}(\tau),\boldsymbol{\xi}_{0,\mathbf{u}}(\tau))\;\boldsymbol{\xi}_2(\tau)\\ \vdots\\ \frac{\partial}{\partial(\mathbf{x}(\tau),\mathbf{u}(\tau))}\,{}^c\mathbf{f}(\boldsymbol{\xi}_{0,\mathbf{x}}(\tau),\boldsymbol{\xi}_{0,\mathbf{u}}(\tau))\;\boldsymbol{\xi}_q(\tau)\end{bmatrix} \tag{26}$$

mittels eines numerischen Integrationsverfahrens [8] gezeigt. Auf eine ausführliche Erläuterung und Herleitung von (26) wird auf [8] verwiesen. Um neben ${}^d\mathbf{A}_i$ auch die Matrix ${}^d\mathbf{B}_i$ berechnen zu können, muss der Ansatz von [8] erweitert werden. Hierzu wird basierend auf (3) die Eingangsgröße $\mathbf{u}$ als Konstante über den Integrationszeitraum T mit $\frac{d}{d\tau}\mathbf{u} = 0$ in die Differentialgleichung aufgenommen. Der erste Spaltenvektor $\boldsymbol{\xi}_0$ mit

$$\boldsymbol{\xi}_{0,\mathbf{x}}(t) := \mathbf{x}(t), \quad \boldsymbol{\xi}_{0,\mathbf{u}}(t) := \mathbf{u}(t) \tag{27}$$

in der Differentialgleichung (26) beinhaltet hierbei den Zustandsvektor sowie die über den Integrationszeitraum konstanten Eingänge von (1). Für die Berechnung der Matrizen ${}^d\mathbf{A}_i$ und ${}^d\mathbf{B}_i$ werden $q = n + m$ Spaltenvektoren $\boldsymbol{\xi}_l(\tau) \in \mathbb{R}^{n+m}$ eingeführt und in der für die Abtastzeit T zu ermittelnden Jacobi-Matrix

$$\boldsymbol{\mathcal{X}}_t(x_{0,i}) = \left[\boldsymbol{\xi}_1(t)\;\boldsymbol{\xi}_2(t)\;\ldots\;\boldsymbol{\xi}_q(t)\right] \tag{28}$$

zusammengefasst. Die Anfangswerte für die numerische Zeitintegration müssen zu

$$\boldsymbol{\xi}_{0,\mathbf{x}}(t_0) := \mathbf{x}_{i,0},\; \boldsymbol{\xi}_{0,\mathbf{u}}(t_0) := \mathbf{u}_{i,0} \tag{29}$$

$$\boldsymbol{\xi}_1(t_0) = \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}, \boldsymbol{\xi}_2(t_0) = \begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix}, \boldsymbol{\xi}_q(t_0) = \begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix} \tag{30}$$

gewählt werden [8].

Das Ergebnis der Integration über die Abtastzeit T liefert neben dem numerischen Ergebnis der unbekannten Flussfunktion

$${}^d\mathbf{f}(\mathbf{x}_{i,0},\mathbf{u}_{i,0}) = \boldsymbol{\xi}_{0,x}(T) \tag{31}$$

auch die Jacobi-Matrix

$$\boldsymbol{\mathcal{X}}_T(\mathbf{x}_{i,0},\mathbf{u}_{i,0}) = \left[\boldsymbol{\xi}_1(T)\;\boldsymbol{\xi}_2(T)\;\ldots\;\boldsymbol{\xi}_q(T)\right] \tag{32}$$

für die Abtastzeit T, über die sich die Matrizen ${}^{d}\mathbf{A}_i$ und ${}^{d}\mathbf{B}_i$ für die T-S Teilmodelle gemäß

$$\begin{bmatrix} {}^{d}\mathbf{A}_i & {}^{d}\mathbf{B}_i \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \boldsymbol{\mathcal{X}}_T(\mathbf{x}_{i,0}, \mathbf{u}_{i,0}) \tag{33}$$

bestimmen lassen. Der affine Term für das T-S Teilmodell wird anschließend mit

$${}^{d}\mathbf{a}_i = \boldsymbol{\xi}_{0,x}(T) - {}^{d}\mathbf{A}_i\mathbf{x}_{i,0} - {}^{d}\mathbf{B}_i\mathbf{u}_{i,0} \tag{34}$$

berechnet.

Die Form der Gewichtungsfunktionen ${}^{d}h_i(\mathbf{z}[k])$ kann nun für die zeitdiskreten T-S Teilsysteme unter Berücksichtigung der konvexen Summeneigenschaft (8) frei gewählt werden.

Die erläuterte Methode bietet den Vorteil, dass keine Annahmen bzgl. der Zeitableitungen der Gewichtungsfunktionen getroffen werden müssen.

4 Vergleich der Modellierungsansätze

Die in den vorangegangen Abschnitten erläuterten Vorgehensweisen zur Erstellung eines zeitdiskreten T-S Modells sollen nun anhand des nichtlinearen Benchmarkbeispiels *Inverses Pendel mit Wagen* untersucht und verglichen werden. In zwei unterschiedlichen Vergleichsstudien wird zum einen bei einer fixen Anzahl an T-S Teilmodellen die Abtastzeit T variiert und zum anderen wird bei einer fixen Abtastzeit T die Anzahl der r Teilmodelle variiert. Untersucht wird jeweils der Einfluss auf den Approximationsfehler der zeitdiskreten T-S Modelle.

Als Approximationsfehler wird hierbei, ausgehend von gleichen Anfangswerten $\mathbf{x}[k] = \mathbf{x}(kT)$, die Abweichung des Zustands $\mathbf{x}[k+1]$ von dem des mittels numerischer Integration berechneten Zustands $\mathbf{x}((k+1)T)$ des nichtlinearen, zeitkontinuierlichen Modells betrachtet. Der Approximationsfehler setzt sich aus Fehlern, die bei der Zeitdiskretisierung und durch die T-S Modellierung selbst entstehen, zusammen.

Um die T-S Modelle in den jeweiligen Untersuchungen vergleichen zu können, werden $V = 500$ Zustände $\mathbf{x}_v[0] = \mathbf{x}_v(0)$ im Bereich (7) mit zufälliger Verteilung erzeugt. Für jeden Zustand wird anschließend das nichtlineare, zeitkontinuierliche Modell über die Abtastzeit T numerisch integriert. Der Approximationsfehler wird bestimmt durch die Differenz

zwischen dem Zustand $\mathbf{x}_v[1]$ der zeitdiskreten T-S Modelle und dem Zustand $\mathbf{x}_v(T)$ des nichtlinearen Modells. Über die Summation

$$E = \frac{1}{N} \sum_{v=1}^{V} |\mathbf{x}_v[1] - \mathbf{x}_v(T)|_2 \tag{35}$$

mit der 2-Norm des Approximationsfehlers wird anschließend für alle N Zustände eine Fehlersumme zur Beurteilung der Approximationsfehler der T-S Modelle berechnet. Zusätzlich wird jeweils auch das zeitkontinuierliche T-S Modell simuliert und mit in die Auswertung aufgenommen um unterscheiden zu können, welcher Anteil des Approximationsfehler aus der Approximation durch die T-S Modellierung an sich entsteht und welcher Anteil der Zeitdiskretisierung zuzuordnen ist.

Sämtliche Berechnungen werden mit *MATLAB* umgesetzt, als numerisches Integrationsverfahren wird der Solver *ODE45* von *MATLAB* verwendet.

4.1 Benchmarkmodell *Inverses Pendel mit Wagen*

Für den Vergleich der Methoden wird das Benchmarkbeispiel *Inverses Pendel mit Wagen* verwendet. Eine schematische Skizze des modellierten Aufbaus ist in Bild 1 zu sehen. Die Position und Geschwindigkeit des Wagens wird mit dem Symbol x und $\dot{x}$ angegeben und der Winkel sowie die Winkelgeschwindigkeit des Stabes mit φ bzw. $\dot{\varphi}$ gekennzeichnet. Der Winkel $\varphi = 0$ entspricht hierbei der oberen instabilen Ruhelage des Pendels. Es wird angenommen, dass sowohl das Pendel mit der Konstanten d_φ als auch der Wagen mit d_x viskos gedämpft sind. Die auf den Wagen wirkende Kraft wird mit $F_c(t) = k_m V_m$ berücksichtigt, wobei V_m der Klemmenspannung des Motors und k_m der Motorkonstanten entspricht. M_c und M_p sind die Massen des Wagens bzw. des Pendels. Das Trägheitsmoment des Pendels wird mit $4/3 M_p l_p^2$ als das eines dünnes Stabes um das Pendelgelenk angenommen. Die halbe Pendellänge wird mit l_p gekennzeichnet.

Nach der Modellierung mittels des Newton-Euler- oder Lagrange-II-Formalismus, erhält man die nichtlinearen, zeitkontinuierlichen Bewegungsgleichungen (36). Mit dem Zustandsvektor $\mathbf{x}(t) = [x,\ \dot{x},\ \varphi,\ \dot{\varphi}]^T$ und dem Eingang $u(t) = V_m$ lassen sich die Gleichungen in der Form (1) darstellen.

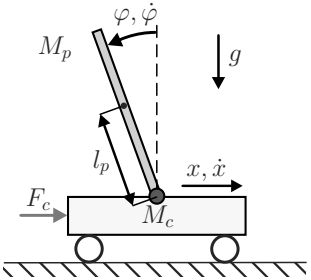


Bild 1: Inverses Pendel mit Wagen [7].

Tabelle 1: Modellparameter

Größe	Parameter	Einheit
Eingangskonstante	$k_m = 1.723$	$\frac{\mathrm{N}}{\mathrm{V}}$
Wagenmasse	$M_c = 9.4 \cdot 10^{-1}$	kg
Pendelmasse	$M_p = 2.30 \cdot 10^{-1}$	kg
halbe Pendellänge	$l_p = 3.302 \cdot 10^{-1}$	m
Viskose Dämpfung des Pendels	$d_\varphi = 2.4 \cdot 10^{-3}$	$\frac{\mathrm{N \cdot m \cdot s}}{\mathrm{rad}}$
Viskose Dämpfung des Wagens	$d_x = 7.724$	$\frac{\mathrm{N \cdot s}}{\mathrm{m}}$
Erdbeschleunigung	$g = 9.81$	$\frac{\mathrm{m}}{\mathrm{s}^2}$

$$\ddot{x} = \frac{4V_m k_m l_p - 4d_x l_p \dot{x} - 3d_\varphi \dot{\varphi}\ \cos(\varphi) - 4M_p l_p^2 \dot{\varphi}^2\ \sin(\varphi)}{l_p(4M_c + 4M_p - 3M_p\ \cos(\varphi)^2)} + \frac{3M_p g l_p\ \cos(\varphi)\ \sin(\varphi)}{l_p(4M_c + 4M_p - 3M_p\ \cos(\varphi)^2)}, \tag{36a}$$

$$\ddot{\varphi} = -\frac{3(M_c + M_p)d_\varphi \dot{\varphi} - 3V_m k_m M_p l_p\ \cos(\varphi) - 3M_p^2 g l_p\ \sin(\varphi)}{M_p l_p^2(4M_c + 4M_p - 3M_p\ \cos(\varphi)^2)} - \frac{3M_p^2 l_p^2 \dot{\varphi}^2\ \cos(\varphi)\ \sin(\varphi) - 3M_c M_p g l_p\ \sin(\varphi)}{M_p l_p^2(4M_c + 4M_p - 3M_p\ \cos(\varphi)^2))} \tag{36b}$$

Die verwendeten Modellparameter sind in Tabelle 1 zusammengefasst und entsprechen einem am Lehrstuhl vorhandenen Versuchsaufbau.

4.2 Untersuchung des Einflusses der Abtastzeit T

Zuerst wird der Einfluss der Abtastzeit auf den Approximationsfehler des jeweiligen T-S Modells hin untersucht. Für die Erstellung der T-S Modelle werden für die drei Methoden *T-S IDENT*, *T-S K2D* und *T-S NL2D* die gleiche Anzahl r an T-S Teilmodellen und die gleichen Stützstellen $\mathbf{x}_{0,i}$, $\mathbf{u}_{0,i}$ verwendet.

Da das zeitkontinuierliche Modell des inversen Pendeles mit Wagen lediglich Nichtlinearitäten in Abhängigkeit von φ und $\dot{\varphi}$ besitzt, werden die Stützstellen der T-S Teilmodelle mit je 9 Stück für den Winkel φ und 9 für die Winkelgeschwindigkeit $\dot{\varphi}$ festgelegt, sodass sich insgesamt $r = 81$ T-S Teilmodelle ergeben. Die Stützstellen werden hierbei äquidistant im Intervall

$$\begin{aligned} \varphi &\in [-60^\circ, +60^\circ] \\ \dot{\varphi} &\in [-200\tfrac{^\circ}{s}, 200\tfrac{^\circ}{s}] \end{aligned} \tag{37}$$

um die obere, instabile Ruhelage verteilt.

Für die Interpolation zwischen den T-S Teilmodellen werden Dreiecksfunktionen verwendet, sodass die Gewichtungsfunktionen ${}^d h(\mathbf{z}(kT))$ bekannt sind. Da der Einfluss der Abtastzeit T auf den jeweiligen Approximationsfehler untersucht werden soll, werden mit jeder der drei Methoden T-S Modelle mit verschiedenen Abtastzeiten T erzeugt. Folgend werden jeweils die Modellierungsschritte und Annahmen erläutert, die für die Anwendung der jeweiligen Methoden auf das Benchmarkmodell notwendig sind.

Die ersten zeitdiskreten T-S Modelle für verschiedene Abtastzeiten T werden mit der Methode *T-S IDENT* erzeugt. Für die Identifizierung werden $p = 20000$ Zustände $\mathbf{x}_{0,j}$ per Zufall in dem gewünschten Gültigkeitsbereich des zu erstellenden T-S Modells generiert. Anschließend erfolgt für jeden Zustand $\mathbf{x}_{0,j}$ als Anfangswert $\mathbf{x}_j(0)$ eine Integration des nichtlinearen, kontinuierlichen Modells mit einem numerischen Integrationsverfahren über die Abtastzeit T. Basierend auf den Integrationsergebnissen $\mathbf{x}_j(T)$ und den Anfangswerten $\mathbf{x}_j(0)$ erfolgt die Identifikation der $r = 81$ zeitdiskreten T-S Teilsysteme über die Minimierung der Summe der kleinsten Fehlerquadrate gemäß (14).

Für die Methode *T-S K2D* wird das nichtlineare, zeitkontinuierliche Modell an den Stützstellen der T-S Teilmodelle gemäß (18) linearisiert. Da die resultierenden Matrizen ${}^c\mathbf{A}_i$ in unserem Beispiel singulär sind, muss

für die Zeitdiskretisierung der T-S Teilmodelle mit den verschiedenen Abtastzeiten T auf die Reihenentwicklung (23) mit $N = 40$ Reihengliedern zurückgegriffen werden.

Abschließend werden die zeitdiskreten Modelle mit der Methode *T-S NL2D* erzeugt. Die Bestimmung der zeitdiskreten Matrizen ${}^d\mathbf{A}_i$ und ${}^d\mathbf{B}_i$ sowie dem Vektor ${}^d\mathbf{a}_i$ der T-S Teilmodelle erfolgt mit (26) mittels numerischer Integration. Als Anfangswerte (29) für die Integration werden jeweils die Stützstellen $\mathbf{x}_{0,i}$, $\mathbf{u}_{0,i}$ der Teilmodelle verwendet.

Im Bild 2 sind die berechneten Fehlersummen über die verschiedenen Abtastzeiten T dargestellt. Es ist zu erkennen, dass das identifizierte T-S Modell zu allen Abtastzeiten den geringsten Fehler aufweist und somit eine sehr gute Approximation des abgetasteten, nichtlinearen Modells im Bereich (37) bildet. Wie aus der Literatur [1, 6] bekannt ist, kann jedoch an einzelnen lokalen Punkten im Zustandsraum die Dynamik des T-S Modells deutlich von der Dynamik des nichtlinearen Systems abweichen, sodass z. B. Ruhelagen des nichtlinearen Modells nicht mit denen des identifizierten T-S Modells übereinstimmen. Somit eignet sich das identifizierte T-S Modell für den analytischen Reglerentwurf-/Beobachterentwurf meist nicht.

Die Fehlersumme des kontinuierlichen T-S Modells (17) als auch die Fehlersumme des zeitdiskreten T-S Modells, welches mittels *T-S NL2D* erstellt wurde, weisen für $T < 0.25s$ die gleiche Größenordnung auf. Hierdurch kann darauf geschlossen werden, dass der Fehler hauptsächlich durch die Approximation der nichtlinearen Funktion durch ein T-S Modell an sich entsteht und der durch die Zeitdiskretisierung entstehende Fehler eine untergeordnete Rolle spielt. Mit einer, um fast zwei Größenordnungen höherer Fehlersumme, liefert das Verfahren *T-S K2D* das schlechteste zeitdiskrete Modell. Es ist ersichtlich, dass nur bei sehr kleinen Abtastschritten dieser Modellierungsansatz angewandt werden sollte.

4.3 Untersuchung des Einflusses der Anzahl der T-S Teilmodelle

Neben der Wahl der Abtastzeit, welche oftmals auch durch die digitale Regelungselektronik vorgegeben ist, kann die Anzahl der T-S Teilmodelle variiert werden um den Approximationsfehler des zeitdiskreten T-S Modells zu verringern. Basierend auf einer fixen Abtastzeit von $T = 0.1s$ wird nun die Anzahl der Teilmodelle erhöht. Hierbei wird weiterhin die äquidistante Verteilung der Stützstellen der Teilmodelle beibehalten und die Verfeinerung gleichermaßen in beiden nichtlinearen Richtungen φ, $\dot{\varphi}$

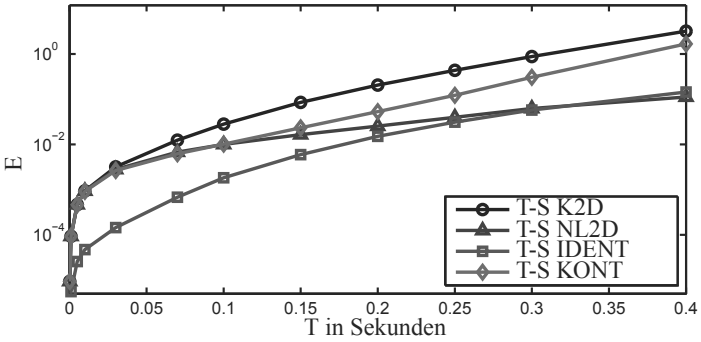


Bild 2: Fehlersumme E über der Abtastzeit T

vorgenommen. Bei 3 Stützstellen erhält man insgesamt $r = 9$ T-S Teilmodelle, bei 19 Stützstellen insgesamt $r = 361$ Teilmodelle. Die Erzeugung der T-S Modelle erfolgt analog zur vorherigen Untersuchung.

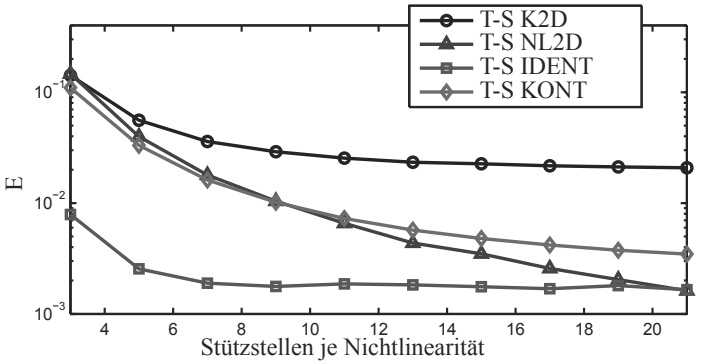


Bild 3: Fehlersumme E über der Anzahl an Teilmodellen für $T = 0.1s$

An den in Bild 3 gezeigten Ergebnissen ist auch in dieser Untersuchung zu erkennen, dass das zeitdiskrete Modell, welches mit der Methode *T-S K2D* erstellt wurde, den mit Abstand größten Fehler aufweist. Durch den Vergleich mit der Fehlersumme E des zeitkontinuierlichen T-S Modells lässt sich erkennen, dass bei einer Abtastzeit von $T = 0.1s$ der Approximationsfehler durch die Vernachlässigung der Zeitableitung der Gewichtungsfunktionen dominiert, sodass die Erhöhung der Anzahl an Teilmodellen keine nennenswerte Reduzierung des Approximationsfehlers bewirkt.

Die identifizierten T-S Modelle bieten bereits eine sehr gute Approximation bei einer geringen Anzahl an Teilmodellen und bei mehr als 9 Stützstellen je Nichtlinearität lässt sich nur noch eine geringe Senkung des Approximationsfehlers erreichen. Auch hier gilt, wie bei der Betrachtung des Einflusses der Abtastzeit erläutert, dass sich die per Identifizierung erzeugten Modelle nur bedingt für den Regler- und Beobachterentwurf eignen.

Der Approximationsfehler, des über den Ansatz *T-S NL2D* erzeugten T-S Modells, lässt sich sukzessive über die Erhöhung der Anzahl an Teilmodellen reduzieren. Dies zeigt zum einen, dass der durch die Zeitdiskretisierung der nichtlinearen Funktion entstehende Fehler eine untergeordnete Rolle spielt und zum anderen das T-S Modelle universelle Approximatoren [5] sind.

5 Zusammenfassung

In diesem Beitrag wurden drei verschiedene Methoden zur Erstellung von zeitdiskreten T-S Modellen vorgestellt und anhand des Benchmarkbeispiels *Inverses Pendel mit Wagen* in zwei Untersuchungen miteinander verglichen. Die erste vorgestellte Methode (*T-S IDENT*) identifiziert die T-S Modelle mit Hilfe von Datensätzen, die zweite Methode (*T-S K2D*) basiert auf der Zeitdiskretisierung eines zeitkontinuierlichen T-S Modells. Die zuletzt vorgestellte Methode (*T-S NL2D*) führt zuerst eine Zeitdiskretisierung des nichtlinearen Modells durch und erzeugt anschließend ein T-S Modell.

In dem betrachteten Beispiel bieten die identifizierten T-S Modelle bereits bei einer geringen Anzahl an Teilmodellen und auch bei großen Abtastzeiten eine sehr gute Approximation des abgetasteten nichtlinearen Modells. Nachteilig ist, dass sich diese T-S Modelle, wie aus der Literatur bekannt ist, nur bedingt für die analytische Auslegung von Reglern- und Beobachtern eignen. Zusätzlich konnte in dem gezeigten Benchmarkbeispiel der Approximationsfehler nicht sukzessive durch eine höhere Anzahl an T-S Teilmodellen verringert werden.

Die Erstellung eines zeitdiskreten T-S Modells mit der Methode *T-S K2D*, auf Basis eines zeitkontinuierlichen T-S Modells, hat sich als die Methode mit der größten Abweichung herausgestellt. Lediglich bei sehr kleinen Abtastzeiten bieten diese Modelle eine akzeptable Approximation.

Basierend auf der Methode *T-S NL2D* konnten T-S Modelle erstellt werden, welche zum einen einen geringen Approximationsfehler besitzen und sich zum anderen auch für den Regler- und Beobachterentwurf eignen. Es konnte anhand des Benchmarkbeispiels gezeigt werden, dass sich der Approximationsfehler sukzessive durch die Erhöhung der Anzahl an Teilmodellen verringern lässt und auch bei großen Abtastzeiten das zeitdiskrete Modell eine gute Approximation des abgetasteten nichtlinearen Modells bietet.

Aufbauend auf die am Benchmarkbeispiel gewonnenen Erkenntnisse wird für ähnliche nichtlineare Systeme somit die Methode *T-S NL2D* zur Ableitung eines zeitdiskreten T-S Modells für den zeitdiskreten Regler- und Beobachterentwurf empfohlen.

In zukünftigen Arbeit sollen noch andere Benchmarksysteme mit den vorgestellten Methoden untersucht werden. Ebenfalls ist angedacht, die Zeitdiskretisierung auf Basis eines *T-S K2D* Modells, welches mit dem *Sector-nonlinearity*-Ansatz erstellt wurde, mit in den Methodenvergleich aufzunehmen.

Literatur

[1] T. A. Johansen, R. Shorten, R. Murray-Smith: On the interpretation and identification of dynamic Takagi-Sugeno fuzzy models. *IEEE Transactions on Fuzzy Systems* Bd. 8 (2000) Nr. 3, S. 297–313.

[2] S. Lee, G. Yen: On the local interpretation of Takagi-Sugeno fuzzy models from a dynamical systems view. In: *Proceedings of the American Control Conference 2002*, Bd. 1, S. 519–524. 2002.

[3] S. Monaco, D. Normand-Cyrot, C. Califano: From Chronological Calculus to Exponential Representations of Continuous and Discrete-Time Dynamics: A Lie-Algebraic Approach. *Automatic Control, IEEE Transactions on* 52 (2007) 12, S. 2227–2241.

[4] H. Unbehauen: *Regelungstechnik II*. Vieweg. 2009.

[5] K. Tanaka, H. O. Wang: *Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach*. John Wiley & Sons. 2001.

[6] A. Kroll: *Computational Intelligence*. Oldenbourg. 2013.

[7] K. Diepold, K. Albert: Lokale Stabilitätsanalyse von Takagi-Sugeno Systemen unter Berücksichtigung ihres Gültigkeitsbereichs. *at - Automatisierungstechnik* (2014).

[8] N. Scherm: *Zum Entwurf nichtlinearer zeitdiskreter Beobachter mittels affiner Approximationen*. VDI-Verl. 1999.

[9] P. Philipp: *Centralized and distributed moving horizon strategies for state estimation of networked control systems*. Verl. Dr. Hut. 2014.

Does imputation work for improvement of domestic hot water usage prediction

Steffen Moritz, Thomas Bartz-Beielstein, Olaf Mersmann, Martin Zaefferer, Jörg Stork

Fakultät für Informatik und Ingenieurwissenschaften, FH Köln
Steinmüllerallee 1, 51643 Gummersbach
Tel.: +49 2261 8196-0
Fax: +49 2261 8196-15
E-Mail: steffen.moritz10@gmail.com

1 Introduction

The "Internet of Things" - the connection of everyday objects to the internet - is claimed to be one of the most important future trends. More and more domestic devices like refrigerators, stoves, smoke detectors, television or even lamps are meanwhile available with integrated internet connectivity. However the pure ability to connect to the internet is only one part. Customers expect some extra value like smart functions from these devices. Providing these smart functions often goes along with building predictive models on the data recorded by these devices. Because of the huge amount of accruing data and occurring problems like missing data this can be quite a challenging task. Especially missing data is a quite common phenomenon, because multiple possible reasons can lead to data gaps.

In this paper we take up this issue and have a look on how different imputation methods to replace missing values influence the outcome of afterwards applied predictive models. For our experiments we used recorded and anonymized data from about 600 connected heating systems. We apply different imputation methods like EM, k-NN and regression based algorithms in order to fill in missing values and afterwards apply a model (e.g. SVM, Neural Net, Random Forest based models) that does a prediction for the customers domestic hot water usage for the upcoming week. In our experiments we want to show the interdependencies between imputation algorithm and prediction model and we want to clarify the question, if imputation in this case is a possible way to improve prediction accuracy.

The remainder of the paper is structured as follows. Section 2 provides an insight into the specific problem we want to solve. The section is followed

by a short summary of the research questions and goals in Section 3. Section 4 gives an overview on previous research and methods. In Section 5 follows a description of the performed experiments, while in Section 6 the results of the experiments are analyzed. A . The paper closes with short summary and outlook in Section 7.

2 Problem Description

2.1 Motivation

As already mentioned in the introduction, the "Internet of Things" enables several new applications from just giving simple status information up to intelligent functions. The recorded data of the networked everyday objects are hereby often the enabler for new applications based on predictive algorithms. But with the network character and the placement at normal households problems with the data recording come along. In comparison to installations in labs or other managed surroundings there are plenty of uncontrollable influences, which can lead to gaps in the data logging. These logging gaps can become a problem later on, because the predictive models used for intelligent functions work best with clean and complete datasets. Missing data in the input of predictive algorithms very likely leads to poorer results or can even lead to no results at all. Thats why finding out how to handle missing data can be useful.

In our specific practical case we are looking at in this paper, we are using anonymized data of about 600 connected heating systems. All the heating installations of our dataset are placed at real customer households. In contrast, the data storage itself is not at the customers home, it is at a server back-end. Which means in our case: the data gets recorded at the customers heating system, is then sent wireless to a connected device, which transfers the data via the customers router and the internet to the server back-end. This is done this way, because of several practical reasons like more computing power on the back-end and too small storage in the heating system itself. But this also leads to more vulnerabilities for the possible data losses. Possible causes for missing data can occur on the whole chain towards the server back-end. This can be issues with the sensors of the heating system itself, problems with the wireless reception, disturbances of the customer's internet connection, the customer switching the router off or even a failure with the server back-end. Partially these problems get addressed with technical solutions. But all in all we can see from the data

we got, that missing data is quite a common phenomenon. So also for this specific real-life problem it is very interesting to address the problem of how to handle missing values and data gaps.

2.2 Dataset

We used anonymized data from about 600 connected heating systems.The timespan of the recorded data goes from December 2013 till end of July 2014. But not for every heating system installation the complete timespan of data is available. Most of the systems start and end logging at different times. So we do have different timespans of recorded data for different heating systems.

The original data that arrives at the server back-end consists out of more than fifty different attributes. These are technical variables which give information about the internal status of the heating system. These are for example informations about temperatures and flags about current working modes of components of the heating system. Imagine the data looking like in table 1 just with many more attributes.

Date Time	Temp. 1	Temp. 2	DHW Req.	further
02/02/2014 14:00:01	60.1	50.3		...
02/02/2014 14:00:05	60.1	50.3		...
02/02/2014 14:00:11	60.0	50.3		...
02/02/2014 14:00:15			1	...
02/02/2014 14:00:17	59.9	50.3		...
02/02/2014 14:00:20			0	...

Table 1: Example extract of the data from one heating installation

As you can see, there is data being recorded nearly every second. The timestamps are not regularly spaced, which means, the time distances between the rows may differ. There are quite a lot of empty rows in the table, but this mustn't be misunderstood as missing values. Empty values in this specific case also just can mean there is no difference to the last broadcasted value. To reduce the amount of data we extract just the data we need to train and built our predictive model out of the complete data. For our experiments we will only need two of the more than fifty attributes - the timestamp and a attribute called DHW Request. The attribute DHW Request (domestic hot water request) indicates when a customer uses domestic hot water. Every time there is a domestic hot water usage, this flags

turns to 1 and afterwards again to 0. To get the dataset we need, we calculate for every heating system the amount of daily DHW Requests and save this together with the timestamp. What we get are about 600 datasets (one for each gateway) looking like table 2. The table can be read like this: 'There where 21 usages of domestic hot water on the second of February 2014 for the shown installation.'

Date Time	Sum DHW Requests
02/02/2014	21
02/03/2014	29
02/04/2014	22
02/05/2014	5
02/06/2014	1
02/07/2014	5
02/07/2014	10
...	...

Table 2: Data in the format used for our experiments

Because we want to compare similar timespans, we define 04/01/2014 and 07/30/2014 as start and end point of our datasets. This means we have to spare all gateways with shorter logging timespans. We also have to spare gateways with missing data in this period. We want to compare imputation methods later on in our experiments and will therefore delete data in a controlled way to measure the performance. In case of missing data being already present, before we apply our missing data mechanism, this could distort the process and results. After taking out the heating systems with data that does not fit these criteria, there are 170 systems left, we can look at.

Shown in a graphical way, our created timeseries looks like in figure 1. Looking at figure 1, we can see there are, depending of the day, up to forty hot water usages a day. But there are also days, where there has been no hot water usage at all.

3 Questions and Goals

What is interesting at our datasets, is, that we have a interesting real-life problem to look at. We have a real-life scenario, where missing data might be a real issue and good ways to handle them are really useful.

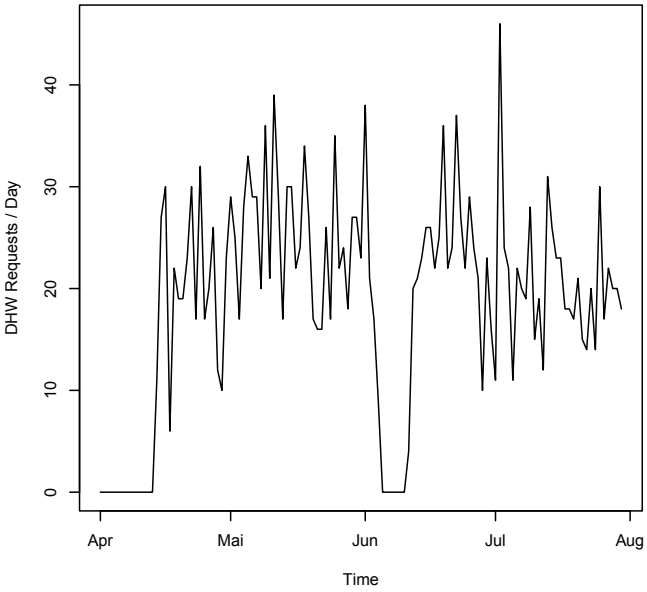


Figure 1: Timeseries of daily sums of DHW Requests for one heating system

What we want to achieve in this paper is, to look on the impact of missing data on predictive functions built on our specific data. To do this we use the preprocessed dataset as described in the 'Dataset' section of our paper. As an example for a predictive function we try to make a 14 day forecast of the amount of DHW requests. This predictive function is realized with different models like decision tree, multiple regression, support vector machines, exponential smoothing and a naive model.

Our first step then is to compare the results of the different models on complete datasets. Our second step will be to take out data from the datasets in a controlled way to look how this influences the performance of the models. In the third step we will also take out data, but afterwards impute them and again look how this influences the results.

The research questions we want to answer are:

1. What is the correlation between the amount of missing data and the performance of the predictive models

2. What general problems occur with missing data
3. Can imputation help to increase the performance of the predictive models
4. Are there best fits for prediction algorithm and imputation algorithm combinations

As performance metric for the comparison of our results we will use the standard metrics RMSE and MAE.

MAE The *mean absolute error* (MAE) between the predicted time series $\hat{y}$ and the respective true DHW Request time series y., i.e.,

$$\mathrm{MAE}(\hat{y}, y) := \frac{\sum_{t=1}^{n} |\hat{y}_t - y_t|}{n}$$

RMSE The *root mean square error* (RMSE) between the predicted time series $\hat{y}$ and the respective true DHW Request (test) time series y, i.e.,

$$\mathrm{RMSE}(\hat{y}, y) := \sqrt{\frac{\sum_{t=1}^{n} (\hat{y}_t - y_t)^2}{n}}$$

4 Previous Research and Methods

4.1 Imputation Methods

When missing data occurs, it may hinder the training of suitable prediction models. In some cases, it may be feasible to simply stop predicting new data once data-samples are missing. But often, especially in technical processes, a more continuously working system is needed.

Friese et al. [1] give an overview of several imputation methods. One important distinction, is whether multi-variate or uni-variate data is predicted. In the multi-variate case, missing data from one signal may be repaired by employing data from other, correlated signals. In the uni-variate case, methods can only rely on information and structure contained in the observed signal.

For uni-variate cases, the most simple method to replace missing data is Last Observation Carried Forward (LOCF). Here, a missing value is replaced by the last observed value. While being conceptually simple, this

has the advantage of being easy to calculate. As such, it will be used in this study. The herein described work will also use methods from the `mice` R-package[1] [2] that collects several methods for multivariate (as well as uni-variate) imputation.

We use:

mean Imputation of unconditional mean values.

norm.predict Imputation with linear regression models.

rf Random Forest is a tool for regression and classification that is based on ensembles of decision trees. It has been developed by Breiman [3]

Besides, any conceivable regression method can basically be employed for imputation. Research has also been applied to whether the imputation of several values for a single missing data-point may be a feasible [4].

4.2 Forecasting Methods

Various data driven methods can be used for prediction of time series data.

Most of the prediction methods used in this paper are taken from the `rminer` R-package. Only the ETS predictor is used from the `forecast` R-package. In the following, a short description of the employed methods is presented.

naive The naive prediction is simply the average of the observations. It is a simple and easy to implement predictor.

dt Decision Trees (DT) determine the outcome of the prediction (leaves of the tree) by several decisions made based on the presented data. The root of the tree (or first node) is the first decision to make, e.g., whether a value is larger or smaller than a certain threshold. Based on the given data, a branch leads to the next node (a new decision) or to a leave (predicted value). Their very simple structure makes decision trees easy to train. Furthermore, users may easily understand rules presented as decision trees.

[1] R is a programming language for statistical computing, see `http://www.r-project.org/` for further information and downloadable software.

svm Support Vector Machines (SVM) have originally been developed for classification by Cortes and Vapnik [5]. By mapping to a higher-dimensional feature space, SVMs can predict nonlinear data with linear hyperplanes. They have been extended to regression [6], which enables their application in this study.

mr In Multiple Regression (MR), linear effects of vector-valued predictor variables are learned.

ets While the above methods are all intended for regression problems in general, Exponential Smoothing State Space Models are more specifically developed towards predicting time-series data. The herein used methods are based on Work by Hyndman et al. [7].

5 Experiments

After we cleaned and preprocessed the initial data, as described in the section 'Dataset', we have 170 files with data to perform our experiments on. Each one of these files contains the daily domestic hot water requests from 04/01/2014 to 07/30/2014 for one specific heating system installation. In our experiments we want to compare different algorithms performing a 14 day forecast on this data. Further on we want to evaluate how missing values in the training data affect our forecast. Afterwards we check, if imputation is able to improve the results.

The complete procedure of our experiments looks like described in Algorithm 1. For the implementation we used the R programming language. For building forecasting models we relied on rminer and forecast R-packages. The used imputation algorithms were mainly taken out of the mice R-package.

Cluster	Size
Cluster 1	68
Cluster 2	27
Cluster 3	16
Cluster 4	1
Cluster 5	45

Cluster	Size
Cluster 6	3
Cluster 7	5
Cluster 8	3
Cluster 9	1
Cluster 10	1

Table 3: Size of clusters

The first step is a clustering of the 170 input files. The reason for this is, in our experiments we realized quite early, that there is a quite huge difference in behavior and results between the different heating system installations. Single installations had a MAE about ten times higher than others. In an overall evaluation with an averaged MAE these heating systems had a too huge impact compared to others. To fix this issue we are building clusters with similar heating systems. To do this we create a distance matrix, which is based on the euclidean distance between the data from the individual heating systems. From the distance matrix we create 10 clusters using the hclust() R-function. Afterwards heating systems with similar variations in daily domestic hot water usage are grouped together. The resulting clusters can be seen in table 3. As can be seen there are four clusters with more than ten heating installations and six clusters with just one to five installations. Our experiments we performed due to time issues just on cluster number 5.

Our experiment process mainly consists out of five loops, processing all the different combinations of results we want to get. The first loop iterates over the different gateway files, in our tested case this are the 45 files of cluster five. Loop two iterates over the different imputation methods we are testing. The third loop is for variation of the missing data rate. Furthermore we have a loop for choosing different random seeds. This one is to avoid random effects coming from the exponential distribution we use to create the missing data . The last loop iterates over the different models we use to make our predictions. In between the loops the necessary methods for our experiments get called. These are the functions for creating the train/test set, the function for creating the missing values, the function to do the imputation and the functions to train the models and to to the predictions. In the following subsection 'Predictions based on complete datasets' we will take a closer look how in general we make the predictions. In the subsection 'Predictions based on incomplete datasets without imputation' we explain our missing data mechanism. And in the final subsection 'Predictions based on incomplete datasets with imputation' we illustrate what it looks like when we do predictions on imputed datasets.

5.1 Predictions based on complete datasets

To be able to evaluate our predictions later on, we split our data in a training and a testing set. Because we are dealing with time-series, we do not create the test sets by random holdouts. Therefore what we do is, we cut the last 14 days of the time-series and define this as the test data. The rest of the

Algorithm 1: Program Structure

input : List of InputFiles $input$
input : List of Imputation Algorithms $sel.algoImp$
input : List of lamda values for exponential distribution $sel.rate$
input : List of random seeds $sel.seed$
input : List of Prediction Models $sel.model$
output: Result Vector $results$

```
files ← cluster(input)
for i.file in files do
    for i.algoImp in sel.algoImp do
        for i.rate in sel.rate do
            for i.seed in sel.seed do
                trainData ← createTrainData(i.file)
                testData ← createTestData(i.file)
                trainData ← missval(trainData, i.rate, i.seed)
                trainData ← impute(trainData, i.algoImp)
                for i.models in sel.model do
                    model ← fitModel(trainData, i.model)
                    prediction ← predict(model, testData)
                    results ← evaluate(prediction)
                end
            end
        end
    end
end
```

data becomes the training data. During our experiments we were trying about 9 different algorithms:

1. naive
2. naivebayes - naive bayes
3. lr - logistic regression (multinom R-package)
4. lda - linear discriminant analysis (MASS R-package)
5. dt - decision tree (rpart R-package)
6. mr - multiple regression (nnet R-package)
7. bruto - additive spline mode (mda R-package)

8. mars - multivariate adaptive regression splines (mda R-package)
9. knn – k-nearest neighbor (kknn R-package)
10. svm – support vector machine (ksvm R-package)
11. ets - exponential smoothing state space model (forecast R-package)

Most of this algorithms we called using the rminer R-package, which as meta package simplifies the usage of different algorithms. Actually later on in our 'Analysis' section we only used results of five different algorithms. This is because at some point we focused on naive, dt, svm, ets and mr, because these had the best results. The naive algorithm is a good indicator, if a algorithm is a fail for the task. Naive just outputs the mean value as a result and is very good in terms of computing time.

For evaluation of our results, we calculate for every heating installation the MAE and the RMSE for the 14 day predictions.

5.2 Predictions based on incomplete datasets without imputation

To do predictions on incomplete datasets, we have to take out a certain amount of data. We do achieve this by applying a missing data function on the training data. The test dataset of course remains the same. The important part here is the implementation of the missing data function.

To simulate the days to the next failure of logging and therefore loss of data, we use the exponential distribution. Using the exponential distribution we can also vary the failure rate given as λ. If there is a failure, we do not replace the values with *NA*, we completely delete this data. Replacing with *NA* would lead to errors in many of the prediction algorithms. To avoid outliers because of a lucky outcome of the exponential distribution for one run, we perform the same run with different random seeds. In our experiments we use the following failure rates: 0 (means we do not apply the missing data function at all), 0.4, 0.8, 1.2, 1.6. The rates we use are quite high, as can been seen in table 4. For example a rate of 1.6 means there are only 16 rows of the former 107 rows of the training set left.

After we minimized the test dataset we go on as we did with the normal data. We train our models and do the predictions. The only difference is the reduced training set, with which we train the models.

Rate	0	0.4	0.8	1.2	1.6
Rows	107	70	48	29	16

Table 4: Connection between rate and remaining rows

5.3 Predictions based on incomplete datasets with imputation

When doing imputation, we are using the function for creating missing data described in the section above and afterwards perform imputations on the reduced dataset. To be able to do this we don't completely remove the missing values, instead we replace them with *NA*. We are using the following algorithms for imputation.

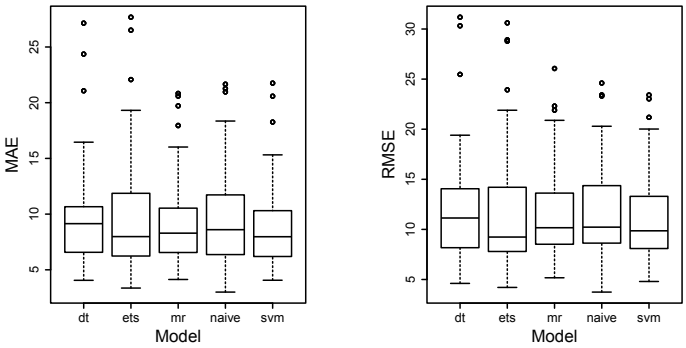
1. norm.predict - Linear regression
2. rf - Random forest imputations
3. mean - Unconditional mean imputation
4. locf - Last observation carried forward

The further steps stay the same as always, we take the reduced and afterwards by imputation filled up training data set to learn our models. Then we predict the following 14 days and calculate the MAE and RMSE.

6 Analysis

We ran our experiments as mentioned in the 'Experiments' section for 45 different heating system installations. Five different prediction algorithms and five different imputation methods were used. We also chose to run with three different random seeds and to take five different missing data rates. This means there were 16875 (45*5*5*3*5) models built and predictions made. Luckily the amount of the training data isn't too high, so that it was possible to perform this with a standard computer within one day.

The first surprise comes along as we look on the data for predictions without missing data in figure 2. None of the algorithms was able to perform significantly better than the 'naive' method. Also the performance of 'ets' who we thought to be especially good for time-series is not better. What also surprises is, that decision tree performs also quite ok compared to the



(a) MAE (b) RSME

Figure 2: MAE and RMSE for different models without missing data

other algorithms. We didn't expect decision tree to gain good results on a univariate time-series. What can't be seen here, we already had taken out some algorithms before our final run, to speed up processing. These were lda, knn, mars, bruto and bayes. Lda, bruto and bayes we took mostly out, because of their bad results that were significantly poorer than the naive algorithm. The rest of these algorithms performed approximately on the same level as the naive and the other algorithms here and were taken out just because of performance reasons. What also can be seen in this graphic, the variations in results for one algorithm are quite huge. There are huge MAE and RMSE outliers in the boxplots. Origin of these variations are not the different random seeds, as could be expected. The reason are the differences between the single gateways.

Interesting now, how the MAE values change if we increase the missing data rate in the training data. This can be seen in figure 3. To avoid confusion, the starting values of the algorithms, that are shown here are not shown in the boxplots of figure 2. In figure 3 the MAE values are the average MAE values for the specific prediction algorithm. The thick line in figure 2 is compared to this the median and not the average. First thing we see here in terms of average MAE, is that svm is the best overall algorithm. The next thing, that is really surprising is, except for ets none of the algorithms perform really bad for high missing data rates. The naive method is very constant over all missing data rates. This is actually not very that surprising, because it takes the average over all data as prediction. The other algorithms results alternate depending on the rate, getting even slightly better sometimes. This effect probably has to do with outliers

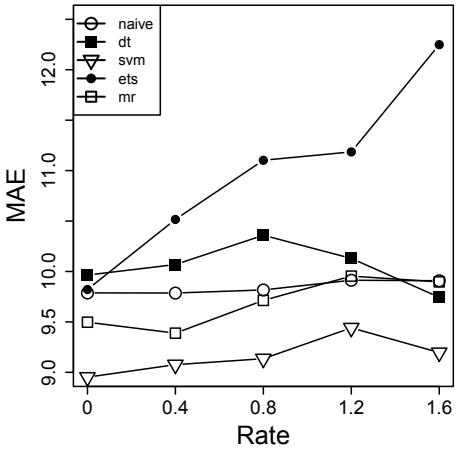


Figure 3: Algorithm performance for different missing data rates

being removed by the missing data function. Considering that a rate of 1.6 means, that just around 16 of 207 values of the training data remain, this the results are anyway quite unexpected.

Knowing, that even a huge missing data rate didn't have much impact on the results, it is already clear, that imputation won't bring a big gain. The question that remains for imputation is now rather: 'Will imputation lead to poorer results?'. This question is still interesting, because sometimes it is the case, that algorithms require a certain maximum of values to run without throwing an error. This was also the reason, we did not take rates higher than 1.6, because this then led to errors. So sometimes it can be useful to impute, just to have enough data. Furthermore it is interesting, weather imputation can bring ets on the level of the other algorithms. Figure 4 and figure 5 show results for different prediction/imputation combinations. The rate thereby stays constant - in figure 4 it is 0.4 and in figure 5 it is 1.2.

The results of the imputations are quite interesting. We have to differentiate between imputation at rate 0.4 and rate 1.2. For rate 0.4 as seen in figure 4 no imputation algorithm leads to significant poorer results than doing no imputation. The interesting thing is now, mean, rf and norm.predict

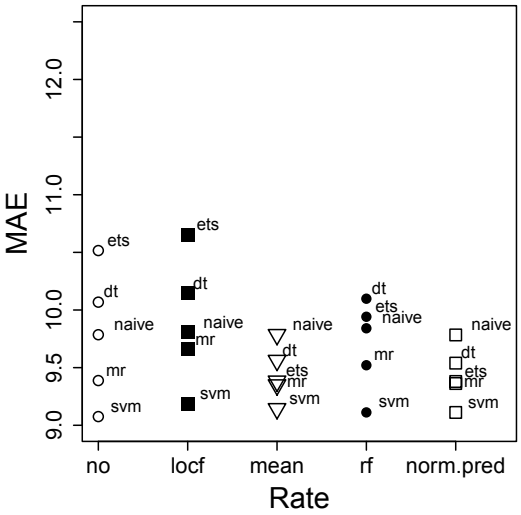


Figure 4: Prediction results for different imputation methods with rate 0.4

influence to prediction in that way, that also ets now reaches the same MAE results as the other algorithms. Last observation carried forward seems to be the worst imputation option. So overall there might be no big improvement, but mean, rf and norm.predict are solid options to use instead of no imputation.

Looking at the imputation with the higher missing rate of 1.2 in figure 5 it looks quite similar. Here it's is even clearer that locf is the worst option. Mean and norm.predict create solid results, being at least as good as the results with no imputation. Even ets has good results for these algorithms. But actually that is quite logical. We know from the naive prediction algorithm that forecasting the mean gives good results. We also do know, that at rate 1.2 already over 50 percent of data is missing and gets imputed. If now all this data gets imputed by mean, also ets will predict something very close to mean. So to sum up the imputation results: imputation does not lead to a new overall best result, but taking the right imputation method, it also does not worsen the results. On the contrary it helps improving results for algorithms that performed poorly before.

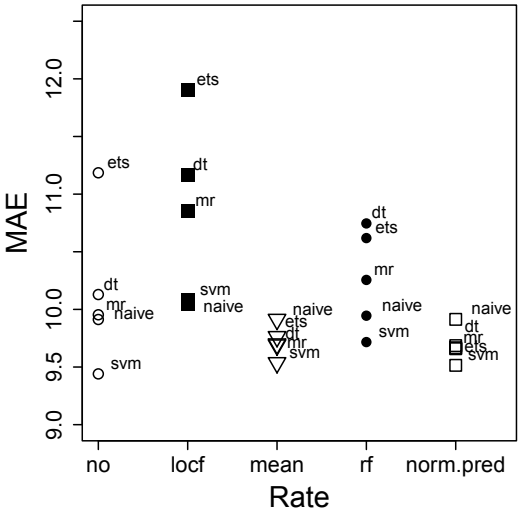


Figure 5: Prediction results for different imputation methods with rate 1.2

7 Summary and Outlook

One major insight we got out of our experiments is, that for this problem it is really hard to surpass the mean calculation of the naive algorithm. Another finding was, that all tested algorithms were extremely robust dealing with high rates of missing data. But we do not credit this to the capabilities of the algorithms - it has rather to do with the specific problem we are working on. With svm being slightly better than the naive algorithm, we figured out a favorite algorithm for this problem. But because of the proximity of the results the naive algorithm would also be a good solution. Our experiments with imputation showed, that most of the imputation methods led to similar results as doing no imputation. This can in some cases be of advantage, because sometimes there is a need of a certain amount of data to be present. Imputation also helped the prediction algorithms that performed poor before to improve their results.

As an outlook it would be interesting, if these results remain stable, even if we change something in our experimental settings. For example we plan to enhance the data set by adding additional attributes like 'day of week'. Another point we would like to check, is if our missing data function reflects reality. A possible way to check this would be to take data with real missing values and look if imputation on this data improves prediction results. Of course it wouldn't be possible to see what the results without missing data would be, but it could be seen weather the results are better with or without imputation. The clustering in front of the processing is also still an open point. In our Analysis we have seen, that there is still a huge variation in the results for different heating systems. Throwing the results of very different heating systems together and building an averaged MAE is not optimal. It would be good to find a method for clustering that better sticks similar heating systems together.

References

[1] Friese, M.; Stork, J.; Guerra, R. R.; Bartz-Beielstein, T.; Thaker, S.; Flasch, O.; Zaefferer, M.: UniFIeD Univariate Frequency-based Imputation for Time Series Data. Techn. Ber., Bibliothek der Fachhochschule Koeln Bibliothek, Betzdorfer Str. 2, 50679 Koeln. URL http://opus.bsz-bw.de/fhk/volltexte/2013/49. 2013.

[2] van Buuren, S.; Groothuis-Oudshoorn, K.: mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software* 45 (2011) 3, S. 1–67. URL http://www.jstatsoft.org/v45/i03.

[3] Breiman, L.: Random Forests. *Machine Learning* 45 (2001) 1, S. 5–32.

[4] Yuan, Y. C.: Multiple imputation for missing data: Concepts and new development (Version 9.0). *SAS Institute Inc, Rockville, MD* (2010).

[5] Cortes, C.; Vapnik, V.: Support-vector networks. *Machine Learning* 20 (1995) 3, S. 273–297. URL http://dx.doi.org/10.1007/BF00994018.

[6] Drucker, H.; Burges, C. J.; Kaufman, L.; Smola, A.; Vapnik, V.: Support vector regression machines. *Advances in neural information processing systems* 9 (1997), S. 155–161.

[7] Hyndman, R. J.; Akram, M.; Archibald, B. C.: The admissible parameter space for exponential smoothing models. *Annals of the Institute of Statistical Mathematics* 60 (2008) 2, S. 407–426.

Boosting Parameter-Tuning Efficiency with Adaptive Experimental Designs

Jörg Stork*, Andreas Fischbach*, Thomas Bartz-Beielstein*, Martin Zaefferer*, A.E. Eiben †

1 Introduction

Model-based tuning has proven to be a successful method for improving the performance of computational intelligence methods such as evolutionary algorithms or neural networks [1, 2, 3]. However, model-based tuning itself can be a demanding and time consuming task. One crucial step during the tuning process is the selection of an adequate experimental design as well as the limits of the algorithm parameter space to be explored, the so-called *region of interest* (ROI). In the current practice, the ROI is static, that is, chosen *a priori* and not changed during the tuning process.

In this paper we will investigate adaptive ROIs. In particular, we introduce mechanisms for appropriately locating and sizing the ROI on-the-fly. We will focus on the sizing aspects, because too large ROIs may slow down the tuning process resulting in worse results. This is due to a large search space leading to a lack of detail in the most critical regions. The meta model would not be able to represent these regions adequately. By adapting the size of the ROI during the tuning process (online) this issue can be dealt with.

To understand the working principles of adaptive ROIs, we implement special *moving* and *zooming* operations, where *zooming* can make the ROI smaller (*zooming in*) or larger (*zooming out*). Furthermore, we distinguish four algorithm variants, depending on whether the model building and the sampling steps utilize (or not) the adapted ROI settings, i.e., the zoomed, local information. Here by we obtain four main variants and our primary research question is: What is the effect of these policies on the tuning process?

Our approach to answer this question is experimental. Our adaptive ROIs are tested with simple benchmark functions and we analyze the effect of

*Dept. of Comp. Sci. and Eng. Sci., Cologne University of Applied Sciences, http://www.spotseven.de, E-Mail: firstname.lastname@fh-koeln.de

†Computational Intelligence Group, Dept. of Comp. Sci., Vrije Universiteit Amsterdam, E-Mail: a.e.eiben@vu.nl

the moving and zooming operations on tuner performance. The rest of this paper is organized as follows. Section 2 describes previous research in this field. The methods employed in this study are introduced in Sec. 3. This includes the employed tuning software as well as the ROI adaptation methodology. A detailed description of the experimental setup is given in Sec. 4. Afterwards, results are presented in Sec. 5. Finally, Sec. 6 concludes this work and gives a short outlook on promising future directions of research.

2 Previous Research

Adaptation of optimization bounds is not a totally new idea. It is closely related to concepts used in adaptive *evolution strategies* (ES). The most simple ES, the so called (1+1)-ES, has a step-size parameter which is multiplied with a preset factor in case of optimization failure, and divided by that factor in case of success [4]. Similar and more complex adaptations of step-sizes or mutation-rates can found in most optimization algorithms.

While model-based tuning algorithms such as *sequential parameter optimization* (SPO), see 3.1) do also employ such optimization algorithms when optimizing the model, they do not necessarily do so on the meta-level. Here, the step size may be best translated to the region in which the search is performed.

One similar approach is often employed with linear models, i.e., in the *response surface methodology* (RSM) [5]. Here, it is obvious that restricting a model to a certain region rather than exploiting knowledge of the whole search space makes sense. A linear model may not fit a complex function on its global scale. Also, depending on the specific model, a global optimization of it would often lead to placing new design points on outmost border of the search space. Hence, RSM adapts the search space sequentially. The question whether this does also make sense with other model-types is one motivation for this work.

A related approach has been introduced and further studied in [6, 7, 8]. The REVAC method is implicitly changing the ROI by continually adapting a distribution used to sample the parameter space. It has been shown to be effective in optimizing already highly developed evolutionary algorithms [9].

3 Methods

3.1 Sequential Parameter Optimization

The performance of most optimization algorithms is influenced by several parameters. Choosing those parameters in a meaningful and beneficial way may be difficult, especially considering that an ideal choice depends on the specific problem to be solved by the optimizer. The meta optimization of an algorithms parameters is often referred to as tuning. Tuning yields not only improved algorithms, but also allows for a fair comparison between competing algorithms.

One framework developed for parameter tuning is SPO [10] . It has been applied to numerous applications [11]. In its core, it is uses methods such as *design of experiments* (DoE), statistics, and machine learning.

A typical SPO run begins with the generation of an initial design. That is, several configurations of different parameter values are generated, e.g., with methods like Latin Hypercube Sampling. The design is generated in an predefined area of the search space, as defined by the ROI. The initial design is then evaluated with the tuned optimization algorithm to determine the quality of the chosen parameter values. With the gained knowledge about each configurations quality, a surrogate model is learned, which is supposed to represent how the parameters influence the quality.

Next, this surrogate model itself is subject to an optimization process. The best parameter configuration (according to the model) is determined. This again takes place in the ROI, and may be based on any algorithm including classical optimization algorithms, evolutionary algorithms or sampling methods (DoE). The optimal solution found may then be evaluated with the tuned algorithm. These steps of model building, model optimization, and algorithm evaluation are repeated in a sequential manner until some specified stopping criterion (e.g., number of steps, number of evaluations) is reached.

3.2 Adaptive ROI

The ROI defines lower and upper limits in each search space dimension and is usually set by an experienced user with more or less good knowledge or at least ideas of the interesting regions of the fitness landscape. These ROI limits are hard limits. The adaptation is performed in each sequential step. In case of the first iteration, it takes place directly after the evaluation of the

initial points, which are given by either a Latin hypercube sampling or a random uniform distributed sample. In the sequential iterations, adaptation takes place directly after evaluating the most recent candidate solution on the target function.

The adaptation can be controlled by two parameters. The first parameter (**MOVE**) controls whether the ROI is moved towards good candidate points or not. The second parameter (**z**) controls the zooming factor of the ROI. The ROI expansion will not go beyond the user defined ROI limits. Both operations, move (if enabled) and zooming are performed in each step. That means, the ROI will be adapted in each iteration by moving, extending, or shrinking the former ROI range. The range is hereby calculated as the distance between the upper an lower limit or the ROI. There are two cases in which different actions are performed by the adaptation (algorithm 1):

1. **Improvement.** If the new best point leads to an improvement, which means it is not equal to the last found point, and the move operation is enabled, than the center of the ROI is moved to this new point in the next iteration. Also, depending on factor **z**, the ROI is either shrinked ($\mathbf{z} > 1$) or extended ($\mathbf{z} < 1$). The first adaptation will be referred to as *zooming in*, whereas the latter is referred to as *zooming out*.

2. **Stagnation.** The center is not moved and the ROI range is either shrunk ($\mathbf{z} < 1$) or extended ($\mathbf{z} > 1$)

This procedure relies on the assumption that optima are situated in the neighborhood of good solutions. For **z** values larger than 1.0, the search is localized by shrinking the ROI if an improvement is achieved. And, if no improvement is made, the ROI is extended to be able to escape local optima.

The case of **z** smaller than 1.0 may yield a more localized, exploitative search in case of stagnation. This could help to focus more on the immediate neighborhood of the best solution. In contrast to the above case, one would not try to escape the local optimum, but rather to improve the best solution found so far in a more local sense. At the same time, improvement may yield a larger ROI, hence progress may be sped up by allowing an even larger step in the next SPO iteration.

Which strategy is better is supposed to be determined in the experiments.

Algorithm 1 Adaptation by Moving and Zooming

if $newCandidatePoint < bestFitness$ **then**
  $w \leftarrow 1/z$ ▷ Zooming(-in) ROI
else
  $w \leftarrow z$ ▷ Zooming(-out) ROI
end if
$l \leftarrow b - a$ ▷ Compute length of ROI
if MOVE **then**
  $p \leftarrow newCandidatePoint$ ▷ Move to new best x position
else
  $p \leftarrow a + l/2$
end if
$a \leftarrow p - w \times l/2$
$b \leftarrow p + w \times l/2$
if then$((b - a) < 1e - 20)$ ▷ If necessary reset ROI
  $a \leftarrow a0$
  $b \leftarrow b0$
end if

The above notions describe in detail when and how the adaptation is performed. One additional open question is how an adapted ROI should be used. That is, it is unclear whether the ROI should affect the boundaries of the search on the surrogate model, the selection of points for training the surrogate model, or even both.

4 Experimental Setup

4.1 Designed Experiments

Adaptation and moving is closely related to locality. A series of experiments was performed to answer the question how local or global model building affect the performance. Prediction of new points based on meta models is done in two steps: first, a (sub)set of points is chosen for the model building. Then, a second set of points has to be selected for the predictions. The correct method is of crucial importance for the search process. Therefore, four possible combinations of building and prediction are included in the design. The corresponding design parameters are explained in Sect. 4.3.

4.2 Objective Functions

The function f_1 is used to demonstrate positive aroi effects.

$$f_1(x) = \begin{cases} 1 & x < -1 \\ x^2 & \text{otherwise.} \end{cases} \tag{1}$$

The function f_2, also known as the *wild* function, is defined as follows.

$$f_2(x) = 10\sin(0.3x)\sin(1.3x^2) + 10^{-5}x^4 + 0.2x + 80. \tag{2}$$

The minimum is located at $x_{opt} = -15.81515$ with $f_2(x_{opt}) = 67.46773$.

Rastrigin's function, which is defined as

$$f_3(x) = 10 + x^2 - 10\cos(2\pi x) \tag{3}$$

was chosen as the third test function. The minimum is located at $x_{opt} = 0$ with $f_3(x_{opt}) = 0$.

All three functions are visualized in Fig. 1.

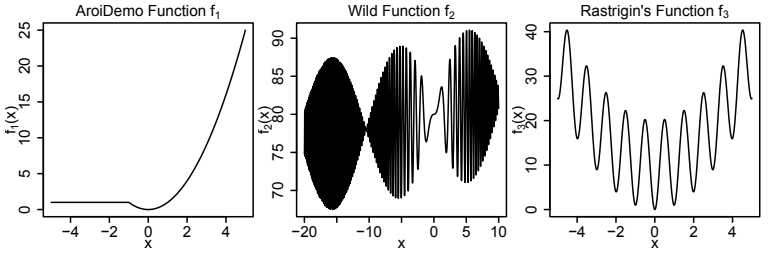


Figure 1: Visualization of the three objective functions.

4.3 Design

The initial design size, n, describes the number of initial sample points, which are evaluated to build the first meta model. Initial design sizes of five and ten were chosen for our experiments.

Regarding the meta modeling, there are four case to be considered.

Mdl-1 (global,global): meta model built on global data, predictions sampled in the region of the global data, i.e., $a(t) = a0$ and $b(t) = b0$ in every time step. Both zooming, z, and moving, $MOVE$, have no effect in this case.

Mdl-2 (global, local): meta model built on global data, predictions sampled in the region of the local data. Here, the samples are taken from the adapted ROI settings $a(t)$ and $b(t)$. The model is build with all evaluated design points at time step t.

Mdl-3 (local, global): meta model built on local data, predictions sampled in the region of the global data. Here, the model building is based on the x design points, which belong to the current ROI, i.e., $x(t) \in [a(t), b(t)]$.

Mdl-4 (local, local): meta model built on local data, predictions sampled in the region of the local data. Here, the model building is based on the x design points, which belong to the current ROI, i.e., $x(t) \in [a(t), b(t)]$. The samples are taken from the adapted ROI settings $a(t)$ and $b(t)$.

The number of repeats, i.e., the number of algorithm runs with unmodified parameter settings but different random seeds, is denoted as nRepeats.

4.4 Algorithm Parameter

Parameters of Algorithm 1 were chosen as follows: Number of points evaluated on the meta model $m = 1000$. To generate points on the meta model, the function `maximinLHS()` was used.

Experimental setups used in this study are shown in Tab. 1.

Table 1: Experimental Setup

	exp1	exprg05	rast01	exprg07	exprg08	exprg09	exprg10
nRepeats	5	50	10	100	100	100	100
zSeq	zSeq1	zSeq1	zSeq1	zSeq2	zSeq3	zSeq2	zSeq2
n	(5,10)	(5,10)	(5,10)	5	5	5	5
MOVE	(T,F)	(T,F)	(T,F)	(T,F)	(T,F)	(T,F)	(T,F)
iter	10	10	10	10	10	10	10
roiMDL		Mdl1-4		Mdl1-4	Mdl1-4	Mdl1-4	Mdl1-4
a0	-10	aSeq	-10	-1	-10	-20	-5.12
b0	10	10	10	10	1	10	15.12
fNum	01	03	01	01	02	03	

Settings for the zoom parameter z are chosen from the set $zSeq1$ = { 0.1, 0.9, 1,1.1, 2.0}, $zSeq2$ = { 0.1, 0.2, ...,1.9, 2.0}, $zSeq3$ = c(0.1, 0.2, ..., 4.9, 5.0). $aSeq$ = { -10, -9, ..., 0, 1}. The sample size is set to m =1000 and a LHD was chosen for each setting. Moving was controlled by the

parameter M with values from $\{true, false\}$. The parameter $iter$ denotes the number of function evaluations after the initial design was evaluated. The total number of function evaluations per run can be determined as $n + iter$.

5 Results

5.1 Effect of Moving and Shrinking

The first experiments are labeled *exp1*, *rast01*, and *wild01*. The initial ROI $a = -10, b = 10$ leads to the result shown in Fig 2. At the first sight, shrinking in case of stagnation leads to better results: the smallest function values were obtained with $z = 0.1$. Furthermore, moving of the ROI center point worsens the performance.

Note, this effect can be explained as follows. This set of experiments uses ROIs, which are symmetric to the center, the location of the optimum. Simply shrinking the interval, regardless of further considerations, obviously improves the function value. Since the ROI initially set by the user can not be exceeded, extending the ROI can never get past this bound. On the other hand, any shrinking of the ROI will naturally lead to improved optimization performance. In the most extreme case, the ROI would simply collapse to the location of the optimum itself, with $a0 = b0 = x_{opt}$.

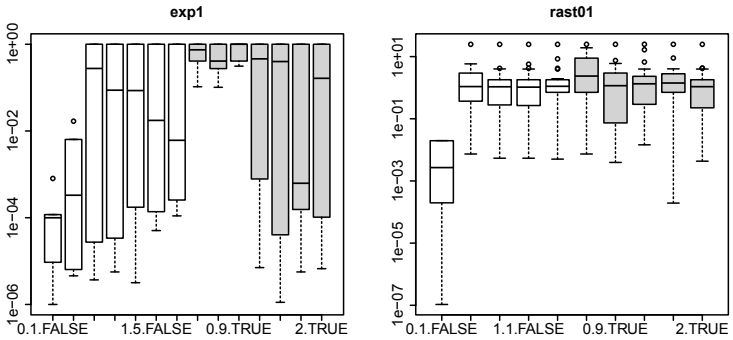


Figure 2: Function values (log.) plotted against $z * M$. Smaller values are better. *Left:* Results from exp1. Leftmost configuration uses M=FALSE, z=0.1. White = no move, gray = move. *Right:* Results from rast01.

To avoid this unwanted side-effect, i.e., optimization is driven by an adequate shrinking procedure and not by the optimization algorithms, a series of experiments were designed, which prevent this behavior. These experiments use asymmetric ROIs, so by simply shrinking the search interval, optimal values cannot be detected.

5.2 Asymmetric Search Interval and Locality in the Meta Model

Interesting effects occur if ROI symmetry is disturbed. In addition to the variations of the ROI locations, experiments which analyze the impact of the adaptation on the meta modeling (as described in Sect. 4.3) are performed. Since four meta modeling variants ($roiGlobal$) were implemented, their comparison might be of interest. The obtained results of the $\log(y)$ value are plotted as a function of the shrinking parameter z and the $roiGlobal$ factor. As can be seen in Fig. 3, local model result in performance improvements when moving is enabled.

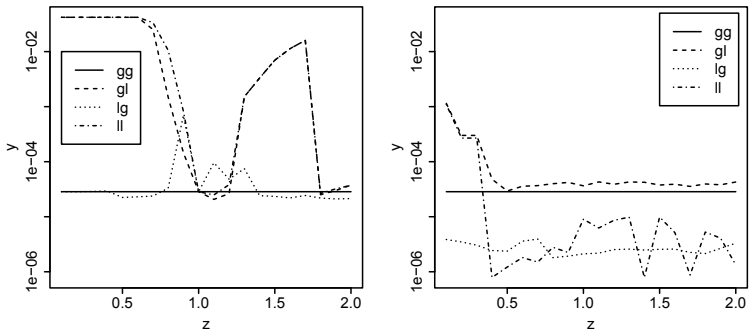


Figure 3: Function values plotted against z for the different modeling approaches; Results from exprg07. *Left:* without moving, a narrow valley can be seen around $z = 1$. *Right:* with moving, the ll and gl settings perform best.

Without Moving (left), the gg setting displays for most of the chosen settings for z an dominating behavior, but in the center a slightly better performance is achieved by the ll and gl model. Furthermore, as the best performance is obtained for values close to $z = 1$, zooming seems to have a negative effect. With moving (right) the local meta modeling performs

best, while zooming seems to have no significant effect. This behavior can be explained by a closer look at the function and the algorithm. Due to the asymmetric ROI (-1,10), the optimum is situated at the very left. If the center point cannot be moved, shrinking leads to an exclusion of the optimum. With moving, the center point of the ROI is moved to the left and the range becomes smaller because it is calculated as the difference between the upper and the here lower hard- limit of the ROI. Thus, even without zooming, the adapted ROI is shrinked and a localized search is performed in the region of the optimum.

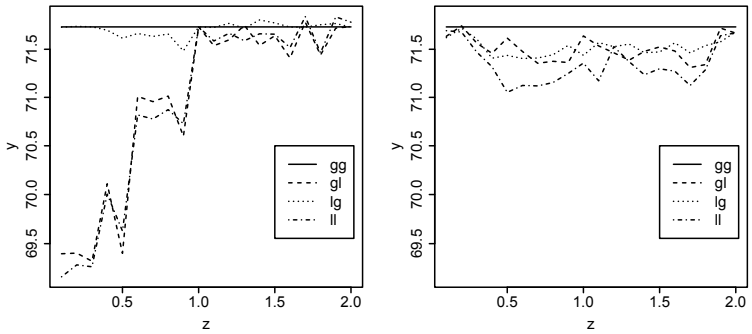


Figure 4: Function values plotted against z for the different modeling approaches for exprg09, the wild function. *Left:* without moving, the gl and ll models show a performance increase for small z values. *Right:* with moving, all modeling approaches perform better than gg where no moving or zooming is applied.

Results from exprg09 show similar results, with the difference of the performance without moving, as shown in Fig. 4 (left). The herein optimized wild function has many local optima. Hence, small values of z, where shrinking is applied if no better solution is found, seems to be beneficial for the local prediction modeling approaches gl and ll, but not with local meta modeling. This can be explained by understanding how the modeling is applied. A strongly localized meta model is, due to the small sample sizes, not able to model the overall behavior of the wild function. At the same time, a localized prediction on a global model leads to a good search direction.

Fig. 5 displays results from expgr08 (left) and expgr10 (right), both with moving. Exprg10 shows a performance similar to expgr07. For expgr08,

the z range was extended to see the behavior for large z values. Exprg08 has a difficult large plateau on the left side, where optimization algorithms usually fail to improve (and perform similar to a random search). Without moving (not shown) no benefit can be achieved with local modeling approaches. With moving, the optimum is often found for z values between 2 and 4, which imply a large shrinking if a good solution is found, together with local prediction models.

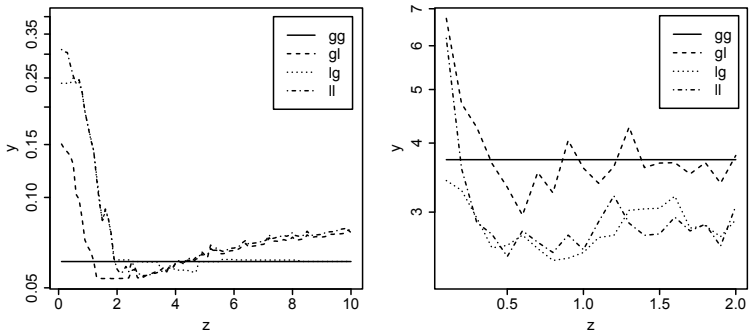


Figure 5: Function values plotted against z for the different modeling approaches. *Left:* Results from exprg08 with extended range of z, with moving. *Right:* Results from exprg10, with moving.

6 Conclusion

Regarding our main research question, how $moving$ and $zooming$ with different local model building affect the performance, the following insights were obtained:

- With $moving$ and $zooming$ a performance increase can be obtained
- The correct setting of the parameter z is important
- The four different model training/exploration approaches strongly affect the performance of the algorithm, depending on the underlying problem.

Our results have shown, that for each of our test problems there is at least one model building variant which outperforms the default without zooming or moving. The variant with lg with local meta modeling and a global prediction modeling with active moving seems to have the best generalization ability, as the performance is at least steady or improved on all test problems. For some problems, we could observe some interesting artifacts, which also might point in the direction of further improvement for our algorithm: if the ROI is centered around the optimum, zooming in without moving has a significant positive effect. We anticipated this, but it shows that the general idea is working and there is a potential for significant improvement. An also interesting effect takes place on the more complex wild function, where a localized search in case of no improvement of the found solution can lead to a significant boost in performance. So we still need to have a good problem knowledge for choosing the correct setting of z and the best model.

This study provides some interesting leads for future work on the topic of adaptive ROIs. We need to revise the moving and shrinking algorithm and try to find an adaptive solution for choosing z and the correct model building. The adaptive process is intended to improve the performance without interaction of the user. Further experiments should be able to give us more insight. We particularly need to perform experiments on higher dimensional cases as most tuning problems consist of a large set of parameters. Further, in future the algorithm shall be able to exceed the user predefined limits (to a still user-set absolute hard limit) so it is possible to enlarge the search space while keeping a initial local approach.

References

[1] Eiben, A. E.; Smith, J. E.: *Introduction to Evolutionary Computing*. Berlin, Heidelberg, New York: Springer. 2003.

[2] Bartz-Beielstein, T.; Branke, J.; Mehnen, J.; Mersmann, O.: Evolutionary Algorithms. *WIREs Data Mining and Knowledge Discovery* 4 (2014), S. 178–195.

[3] Stützle, T.; López-Ibáñez, M.: Automatic (Offline) Configuration of Algorithms. In: *GECCO '14: Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*. New York, NY, USA: ACM. 2014.

[4] Beyer, H.-G.; Schwefel, H.-P.: Evolution Strategies: A Comprehensive Introduction. *Natural Computing* 1 (2002) 1, S. 3–52.

[5] Box, G. E. P.; Draper, N. R.: *Empirical Model Building and Response Surfaces*. New York NY: Wiley. 1987.

[6] Nannen, V.; Eiben, A. E.: Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters. In: *Proc of IJCAI 2007* (Veloso, M., Hg.), S. 975–980. AAAI Press. 2007.

[7] Nannen, V.; Eiben, A. E.: Efficient Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters. In: *IEEE Congress on Evolutionary Computation*, S. 103–110. Singapore: IEEE Press, Piscataway, NJ. 2007.

[8] Nannen, V.; Smit, S. K.; Eiben, A. E.: Costs and Benefits of Tuning Parameters of Evolutionary Algorithms. In: *PPSN* (Rudolph *et al*, G., Hg.), Bd. 5199 von *Lecture Notes in Computer Science*, S. 528–538. Springer. ISBN 978-3-540-87699-1. 2008.

[9] Smit, S. K.; Eiben, A. E.: Beating the 'world champion' evolutionary algorithm via REVAC tuning. In: *IEEE Congress on Evolutionary Computation*, S. 1–8. IEEE Computational Intelligence Society, Barcelona, Spain: IEEE Press. URL `http://www.cs.vu.nl/~sksmit/bibs/CEC-2010-RobustParameters.pdf`. 2010.

[10] Bartz-Beielstein, T.; Lasarczyk, C.; Preuß, M.: Sequential Parameter Optimization. In: *Proceedings 2005 Congress on Evolutionary Computation (CEC'05), Edinburgh, Scotland* (McKay, B.; et al., Hg.), Bd. 1, S. 773–780. Piscataway NJ: IEEE Press. 2005.

[11] Bartz-Beielstein, T.: Sequential Parameter Optimization—An Annotated Bibliography. CIOP Technical Report 04/10, Research Center CIOP (Computational Intelligence, Optimization and Data Mining), Cologne University of Applied Science, Faculty of Computer Science and Engineering Science. 2010.

Constrained Optimization with a Limited Number of Function Evaluations

Patrick Koch[*], Samineh Bagheri[*], Christophe Foussette[⊕], Peter Krause[⊕], Thomas Bäck[⊕], Wolfgang Konen[*]

[*]Cologne University of Applied Sciences
Steinmüllerallee 1
51643 Gummersbach
E-Mail: {patrick.koch, wolfgang.konen}@fh-koeln.de

[⊕]divis intelligent solutions GmbH
Joseph-von-Fraunhofer-Str. 20
44227 Dortmund
E-Mail: {foussette, krause, baeck}@divis-gmbh.de

Abstract

In real-world optimization often constraints must be respected, restricting the number of feasible solutions. Therefore algorithms and strategies have been proposed to repair constraint-violating solutions or to avoid extensive search in infeasible regions. Such constraint handling methods are well-known from the literature, but most algorithms have the drawback that they require a large number of function evaluations. This can be especially problematic for real-world optimization tasks, which often incorporate expensive simulations. Up to now, only little work has been devoted to *efficient* constraint-based optimization (severely reduced number of function evaluations). A possible solution in that regard is to use *surrogate models* for the objective and constraint functions respectively. While the real function might be expensive to evaluate the surrogate functions are often much faster. Recently, as an example for this approach, the solver *COBRA* was proposed and outperforms most other algorithms in terms of required function evaluations on a large number of benchmark functions. In this paper we propose a new implementation of COBRA and compare it with other constraint-based optimization algorithms. We discuss the internal components of the algorithm and find that by adding new strategies, the algorithm can be significantly improved. We also report on negative results where COBRA still shows a bad behaviour and gives indications for possible improvements.

1 Introduction

Real-world optimization problems are often subject to constraints, restricting the feasible region to a smaller subset of the search space. It is the goal of most optimizers to avoid infeasible solutions and to stay in the feasible region, in order to converge in the optimum. However, the search in

constraint black-box optimization can be difficult, since knowledge about the size of the feasible region and the location of the optima is usually unknown. This problem even turns out to be much harder, when only a limited number of function evaluations is allowed for the search. However, this is often the case in real-world optimization, where good solutions are requested in very restricted timeframes. In this paper we present a state-of-the-art solver for constraint-based optimization and discuss advantages and common pifalls of the method.

1.1 Related work

For constrained problems several repair methods have been proposed [1, 2], that aim to repair infeasible solutions. Another common approach is to incorporate static or dynamic penalty terms to stay in the feasible region [3, 4, 5]. Other techniques handle constraints by optimizing objective function and constraint functions separately in a lexical order [6, 7]. [8] is another example of this approach with stochastic ranking. Also multi-objective optimization algorithms have been designed for constraint-based optimization, considering the constraint functions as additional objectives [8, 9]. Beyer and Finck [10] propose an extension of CMA-ES which allows to handle special types of constraints successfully.

In the field of model-assisted optimization algorithms for constrained problems, Support Vector Machines (SVMs) have been used by Poloczek and Kramer [11]. They make use of SVMs as a surrogate of the objective function, but achieve only slight improvements. Powell [12] proposes COBYLA, a direct search method which models the objective and the constraints by linear functions. Recently, Regis [13] developed COBRA, an efficient solver that makes use of Radial Basis Function (RBF) interpolation, and outperforms most algorithms in terms of required function evaluations on a large number of benchmark functions.

In this paper we analyze a new implementation of COBRA in `R`, which allows easy adaptation of certain components of the algorithm. In Sec. 2 we present the problem and the algorithm in more detail. In Sec. 3 we perform a thorough experimental study on analytical test functions and on a real-world benchmark function. The results are discussed with regard to specific parameter settings of the algorithm in Sec. 4 and we give conclusive remarks in Sec. 5.

2 Methods

2.1 Constrained-based optimization

A constrained optimization problem can be defined by the minimization of a real-valued objective function f subject to constraint functions $s_1, \ldots, s_m$:

$$\begin{aligned} &\text{Minimize} && f(\vec{x}), \vec{x} \in \mathbb{R}^d \\ &\text{subject to} && \\ & && s_i(\vec{x}) \leq 0, i = 1, 2, \ldots, m \end{aligned}$$

In this paper we always consider minimization problems. Maximization problems can be transformed to minimization problems without loss of generality.

2.2 Radial Basis Functions

The COBRA algorithm incorporates optimization on auxiliary functions, e.g., a regression model of the search space. Although numerous regression models are available therefore, we employ interpolating RBF [14, 15], since they outperformed other models. In this paper we use the same notation like Regis [16]. The RBF model requires a set of design points (a training set) as input: n points $\vec{u}^{(1)}, \ldots, \vec{u}^{(n)} \in \mathbb{R}^d$ are evaluated on the real function $f(\vec{u}^{(1)}), \ldots, f(\vec{u}^{(n)})$. We use an interpolating radial basis function as approximation:

$$\hat{f}(\vec{x}) = \sum_{i=1}^{n} \lambda_i \varphi(||\vec{x} - \vec{x}^{(i)}||) + p(\vec{x}), \quad \vec{x} \in \mathbb{R}^d \tag{1}$$

Here, $||\cdot||$ is the Euclidean norm, $\lambda_i \in \mathbb{R}$ for $i = 1, \ldots, n$, $p(\vec{x})$ is a linear polynomial in d variables, and φ is of cubic form $\varphi(r) = r^3$. Although other choices for φ are possible and have been tested in related work, cubic RBF have been shown to be superior in [17].

Fitting of the model can be done by defining a distance matrix $\Phi \in \mathbb{R}^n$: $\Phi_{i,j} = \varphi(||\vec{u}^{(i)} - \vec{u}^{j}||), i, j = 1, \ldots, n$. The cubic RBF model is obtained by solving the linear system of equations:

$$\begin{bmatrix} \Phi & P \\ P^T & 0_{(d+1)\times(d+1)} \end{bmatrix} = \begin{bmatrix} \lambda \\ \vec{c} \end{bmatrix} \times \begin{bmatrix} F \\ 0_{d+1} \end{bmatrix} \tag{2}$$

where $0_{(d+1)\times(d+1)} \in \mathbb{R}^{(d+1)\times(d+1)}$ is a zero matrix, $F = (f(\vec{u}^{(1)}), \ldots, f(\vec{u}^{(n)}))$, 0_{d+1} is a vector of zeros, $\lambda = (\lambda_1, \ldots, \lambda_n)^T \in \mathbb{R}^n$ and $\vec{c} = (c_1, \ldots, c_{d+1})^T \in \mathbb{R}^{d+1}$ are the coefficients of the linear polynomial $p(\vec{x})$. The matrix in Eq. (2) is invertible if it has full rank. This is usually the case, if $d + 1$ linearly independent points are provided. The matrix inversion can be efficiently calculated by using singular value decomposition (SVD) or similar algorithms.

RBF models are very fast to train, even in high dimensions. They often provide good approximation accuracy even when only few training points are given. This makes them ideally suited as surrogate models high-dimensional optimization problems with a large number of constraints.

2.3 Constrained Optimization by Radial Basis Function Approximation

Constrained Optimization by Radial Basis Function Approximation (COBRA) is an optimization algorithm proposed by Regis [13]. The main idea of this method is to use approximations of both the objective function and constraint functions, in order to save evaluations of the real function and constraints. Internally COBRA uses RBF interpolation for the modeling of the objective and constraints. Each iterate is a result of an optimization on a subproblem, which is defined by the RBF interpolation models of the objective and the constraint functions.

Fig. 1 presents a flowchart of the algorithm. In the beginning an initial population or design is generated to make it possible to create the first RBF model. This can be done by using various strategies which are described in more detail in Sec. 3.3. The resulting RBF models from the initial design are used to find the next iterate to be evaluated on the real function. Therefore, a sequential search is performed on the surrogate functions. The best point obtained from this search is referred to as *infill point* in the remainder of this paper. Note that the infill point is the only point that is also evaluated on the real function. This makes the algorithm efficient in terms of real function evaluations required. If the infill point is better than the current best solution, the best solution is replaced by the infill point. In any case the RBF models are updated using the new information. In the next round again a sequential search is performed until the number of function evaluations exceeds the maximum number of allowed evaluations given by the user (the budget for the optimization).

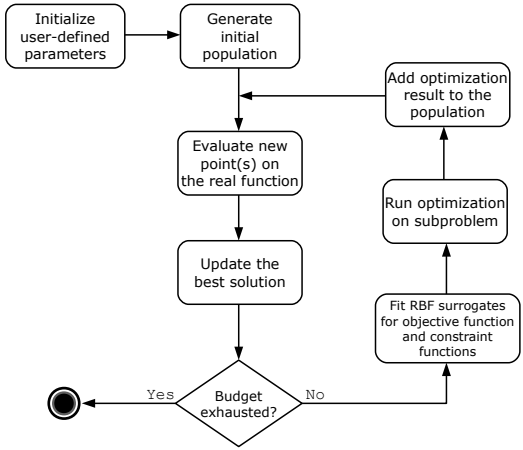


Figure 1: Flowchart of the COBRA algorithm.

Model-assisted optimization In each iteration COBRA performs an optimization on the RBF models. This can be done by either using a special constraint solver, or by using a method for unconstrained optimization with a penalty function to avoid infeasible solutions. In this paper we incorporate two constraint solvers, COBYLA [12] and ISRES [8], and also make use of unconstrained optimization methods in form of classical Hooke & Jeeves pattern search [18] and the simplex algorithm by Nelder & Mead [19]. The selection of the internal optimization strategy in COBRA is arbitrary and must be defined by the user. In Sec. 3.4 we compare different optimization strategies on a real-world problem.

Distance requirement cycle COBRA applies a distance requirement factor which determines how close the next solution $\vec{x}_{infill} \in \mathbb{R}^d$ is allowed to be to all previous ones. The idea is to avoid frequent updates in the neighbourhood of the actual best solution. The distance requirement can be passed by the user as external parameter vector $\Xi = \langle \xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(\kappa)} \rangle$ with $\xi^{(i)} \in \mathbb{R}^{\geq 0}$. In each iteration, COBRA selects the next element $\xi^{(i)}$ of Ξ and adds the constraints $||\vec{x}_{infill} - \vec{x}_j|| \geq \xi^{(i)}, \quad j = 1, ..., n$ to the set of constraints. This measures the distance between the proposed infill solution and all n previous infill points. The distance requirement cycle is

a clever idea, since small elements in Ξ lead to more exploitation of the search space, while larger elements lead to more exploration. If the last element of Ξ is reached, the selection starts with the first element again and so on. The size of the vector and the single components of the distance requirement vector can be arbitrarily chosen.

Uncertainty of constraint predictions COBRA aims at finding feasible solutions by extensive search on the surrogate functions. However, as the RBF models are probably not exact especially in the initial phase of the search, a factor ϵ is used to handle wrong predictions of the constraint surrogates. In the beginning we set $\epsilon_{init} = 0.005 \cdot l$, where l is the diameter of the search space. In each iteration n we only claim the point to be feasible if the following Eq. holds for all constraint surrogates $s_i^{(n)}$ with $i = 1, \ldots, m$:

$$s_i^{(n)} + \epsilon_i^{(n)} \leq 0 \tag{3}$$

That is, we tighten the constraints by adding the factor ϵ which is adapted during the search. The ϵ-adaptation is done by counting the feasible and infeasible infill points C_{feas} and C_{infeas} over the last iterations. When the number of these counters reaches the threshold for feasible or infeasible solutions, T_{feas} or T_{infeas}, respectively, we divide or double ϵ by 2 (up to a given maximum). When ϵ is decreased, solutions are allowed to move closer to the constraint boundaries (the imaginary boundary is relaxed), since the last T_{feas} infill points were feasible. Otherwise, when no feasible infill point is found for a while (T_{infeas}), the ϵ factor is increased in order to keep the points further away from the constraint boundary.

3 Experimental analysis

3.1 Benchmark functions

For evaluation we use popular benchmark functions, e.g., the G functions described in [20]. In Tab. 1 we present characteristics of these functions. It can be seen from the table that the functions differ in dimension, objective and number and type of constraints. Since these functions are often used in the scientific literature for analyzing constrained-based solvers, they provide a good analytical testbed for our algorithm.

Additionally we evaluate the algorithm on a high-dimensional real-world problem from the automotive industry: the MOPTA 2008 benchmark by

Table 1: Characteristics of G functions: d: dimension, ρ: percent feasible, R: range of objective function, LI: the number of linear inequalities, NI: number of nonlinear inequalities, NE: the number of nonlinear equalities

Fct.	d	type	ρ	R	LI	NI	NE
G01	13	quadratic	0.0003%	293.87	9	0	0
G02	20	nonlinear	99.9973%	0.69	1	1	0
G03	10	nonlinear	0.0026%	1	0	0	1
G04	5	quadratic	27.0079%	9725.83	0	6	0
G05	4	nonlinear	0.0000%	8850.43	2	0	3
G06	2	nonlinear	0.0057%	1247439.40	0	2	0
G07	10	quadratic	0.0001%	5660.62	3	5	0
G08	2	nonlinear	0.8581%	1691.26	0	2	0

Jones [21] is a 124-dimensional problem with 68 constraints. All input parameters have been normalized to $[0, 1]$. The constraint values are meaningfully scaled, e.g., if a constraint value s_i of 0.05 is returned, this means that the constraint boundary is violated by a percentage of 5%. The problem should be solved within $1860 = 15 \cdot d$ function evaluations which in reality refers to one month of computation time on a high-performance computer.

3.2 Results on benchmark functions

We compare our COBRA implementation in R[1] with the results from other research articles. Tab. 2 shows the results of 30 independent runs on the G functions introduced in Sec. 3.1. For comparison we present the results from COBRA by Regis [13], the stochastic ranking evolution strategy (ISRES) by Runarsson and Yao [8] and the Repair Genetic Algorithm (RGA) by Chootinan and Chen [2]. The results from COBRA by Regis [13], ISRES [8] and RGA [2] have been taken from the original articles of the authors, for COBYLA we ran experiments using the nloptr package in R.[2]

The results of our COBRA implementation can achieve accuracies similar or close to the results of the evolutionary algorithms (EA) ISRES and RGA. This already can be seen as a success, since it was not clear, if the surrogate models in COBRA are able to find very good approximations of the real function and constraints. However, with exception of function G02, where our COBRA implementation performed poorly, the results are

[1] http://cran.r-project.org/

[2] http://cran.r-project.org/web/packages/nloptr/nloptr.pdf

Table 2: Best (b), median (m) and worst (w) results and their standard deviation (sd) determined in 30 independent runs with different approaches.

Fct.	Optimum		COBRA-R	COBRA [13]	ISRES [8]	RGA 10% [2]	COBYLA [12]
G01	-15.00	b	-15.00	NA	-15.0	-15.0	-15.0
		m	-15.00	NA	-15.0	-15.0	-13.83
		w	-13.00	NA	-15.0	-15.0	-0.27
		sd	0.58	NA	5.8e-14	0.0	1.30
G02	-0.80355	b	-0.409403	NA	-0.803619	-0.801119	-0.272
		m	-0.346592	NA	-0.793082	-0.7857	-0.199
		w	-0.281917	NA	-0.723591	-0.745329	-0.164
		sd	0.028	NA	2.2e-02	-0.0137	0.023
G03	-1.0	b	-0.9899	-0.8965	-1.001	-0.9999	-1.0
		m	-0.9753	0.00	-1.001	-0.9999	-0.2289
		w	0.000	0.00	-1.001	-0.9997	0.0
		sd	0.19	NA	0.0	0.0	0.45
G04	-30665.539	b	-30665.5386	-30665.49	-30665.539	-30665.5386	-30665.539
		m	-30665.5386	-30665.15	-30665.539	-30665.5386	-30665.539
		w	-30665.5386	-30664.58	-30665.539	-30665.5386	-30665.539
		sd	7.5e-05	0.04	1.1e-11	0.0	8.3e-09
G05	5126.498	b	5126.4981	5126.5	5126.497	5126.498	5126.498
		m	5126.4981	5126.51	5126.497	5126.498	5126.498
		w	5126.4989	5126.53	5126.497	5126.498	5126.498
		sd	2.4e-04	0.0	7.2e-13	0.0	0.0
G06	-6961.8138	b	-6961.8134	-6944.54	-6961.81	-6961.81	-6961.81
		m	-6961.8116	-6795.6	-6961.81	-6961.81	-6961.81
		w	-6961.8044	-6460.53	-6961.81	-6961.81	91.05
		sd	1.5e-2	24.6	1.9e-12	0.0	1782.09
G07	24.306	b	24.306	24.48	24.306	24.329	24.306
		m	24.306	24.306	24.306	24.472	24.306
		w	24.309	29.33	24.306	24.835	1440.87
		sd	6.6e-04	0.15	6.3e-05	0.13	343.91
G08	-0.0958250	b	-0.0958250	-0.10	-0.0958	-0.0958	-0.0958
		m	-0.0957808	-0.09	-0.0958	-0.0958	-0.0272
		w	-0.0945741	-0.06	-0.0958	-0.0958	0.0
		sd	2e-04	0.0	2.7e-17	0.0	0.02

in line with the EA and also outperform the original COBRA algorithm in Matlab published by Regis [13] on some functions. While in general the implementations are similar to each other, the following differences can be made responsible for this:

- internal optimizer in COBRA (Regis uses `fmincon` in Matlab, we use COBYLA, ISRES, NMKB or HJKB in `R`)
- size of the initial design. Regis always uses $d+1$ points, whereas we also set higher values up to $3 \cdot d + 1$
- initial design type: Regis proposes random initial design, we allow LHS, optimized and biased designs (Sec. 3.3)
- usage of *repair infeasible* method (Sec. 3.5), for repairing slightly infeasible solutions

As Tab. 2 only reports the objective values, we want to indicate that COBRA has been designed for complex real-world optimization, and one of its main advantages is that optimization can be performed spending only very few function evaluations. Because COBRA makes use of internal surrogate models, it usually requires only a fraction of the function evaluations needed by other strategies such as the EA. In Tab. 3 we present the number of real function evaluations required by the methods to achieve the objective values in Tab. 2. It is clearly visible that both our COBRA implementation in `R` and the original COBRA implementation in Matlab need only very few function evaluations, while algorithms like ISRES or RGA sometimes require 1000 times more evaluations to yield similar results.

Comparing COBRA and COBYLA, it can be seen from the result tables that COBRA requires less function evaluations. One reason therefore might be that COBYLA does not incorporate any additional heuristics as the distance requirement and only uses linear models which might be worse for nonlinear functions. Another disadvantage of COBYLA can be the dependency of the delivered starting point. While COBRA uses a set of points, COBYLA is prone to early convergence in local optima when the problem is multimodal and the starting point is far from the optimum. As a consequence the final solutions of COBYLA are not as precise as the solutions delivered by COBRA and COBYLA needs substantially more function evaluations until convergence (cf. Tab. 3).

Table 3: Average number of function evaluations used to reach results shown in Tab. 2 for different algorithms.

	COBRA-R	COBRA [13]	ISRES	RGA 10%	COBYLA
G01	59	NA	350 000	95 512	986.4
G02	500	NA	350 000	331 972	5 000.0
G03	500	100	350 000	399 804	3 402.2
G04	100	100	350 000	26 981	441.3
G05	100	100	350 000	39 459	745.0
G06	100	100	350 000	13 577	276.6
G07	150	100	350 000	428 314	2 740.0
G08	150	100	350 000	6 217	430.0

3.3 Initial design

The initial design strategy in COBRA (Sec. 2.3) can be responsible for obtaining good or bad results. Especially for multimodal problems a good selection of the initial design can lead to meaninginful better results. In the `R` implementation of COBRA the user can select between three different initialization strategies:

LHS: Latin Hypercube Sampling of size n

Biased: The points are randomly sampled around the provided starting solution with given standard deviation sd.

Optimized: An initial optimization is performed without model-assisted optimization. In this paper we used Hooke & Jeeves [18] with a static penalty function for this.

In Fig. 2 we provide the results of 20 runs with COBRA on the G07 test function, first with a LHS random initialization and second with an optimized initialization. As can be seen from the plot the random LHS initialization leads to a premature convergence with no improvement after some early iterations. Instead the optimized initialization, where a Hooke & Jeeves search is performed using a simple penalty function, leads to much better progress and no stagnating behaviour. As a consequence, the selection of the initial design has a direct effect on the performance. A very bad selection of initial design points seems to completely deteriorate the algorithm's progress.

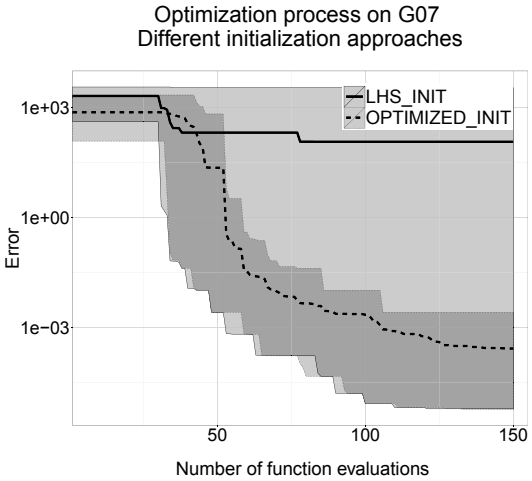


Figure 2: The plot shows the mean results of 30 independent runs on G07 for the LHS and optimized initialization strategies. The upper and lower ribbons depict best and worst solutions of the 30 runs.

3.4 Internal optimization strategy

We ran COBRA on the MOPTA benchmark problem with different optimizers for the optimization on the surrogate functions. E.g., in our COBRA `R` implementation we can select between the following optimization strategies:

- Hooke & Jeeves (HJKB) search [18]
- Nelder & Mead simplex algorithm (NMKB) [19],
- Constrained-based optimization by linear approximation (COBYLA) by Powell [12]

We assumed to get the best results with COBYLA, since it builds an internal model of the objective and constraints and usually outperforms the classical direct search strategies which are sometimes even not designed for high-dimensional problems (e.g., Nelder & Mead tends to get lost in large search spaces). In the left plot of Fig. 3 the mean out of ten runs of the best feasible solution visited over the optimization loop so far is depicted. In the initialization one feasible starting solution was given by

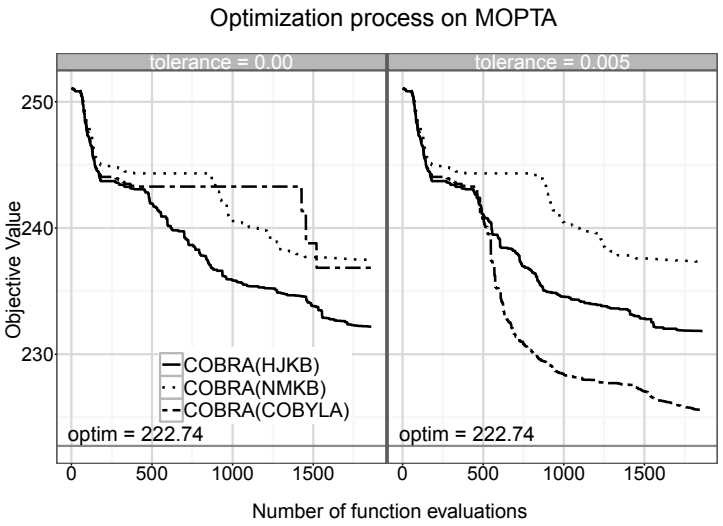


Figure 3: Average optimization progress for COBRA-R on the MOPTA 2008 benchmark. Left: best feasible solution up to the current iteration, right: best solution with 5% constraint violation (tolerance) allowed. The known optimum is shown as a straight grey line at 222.74.

Jones [21]. The rest of the 249 initial design points was generated by the *optimized* initialization strategy (cf. Sec. 3.3) using a size of 249 design points. It can be seen from the plot that surprisingly HJKB outperforms NMKB and COBYLA.

Interestingly, the situation changes if we allow for small constraint violations of up to 5%. Although infeasible solutions can be generated by this, it is possible that small constraint violations can be resolved afterwards, e.g., by applying a method similar to the *repair infeasible* algorithm we are presenting in Sec. 3.5.

In the right plot of Fig. 3 we again started the algorithms HJKB, NMKB and COBYLA on the MOPTA benchmark, but now allowing constraint violations of up to 5%. As can be seen from the plot the result of the COBYLA algorithm has been improved significantly. While COBYLA performed rather poor before with the hard constraint, it can now reach the optimum of the MOPTA benchmark in several runs. While the classical HJKB and NMKB did not yield any benefits from this small change, COBYLA could clearly improve its result and is close to the desired optimum.

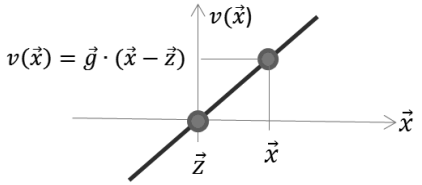


Figure 4: Repairing infeasible solutions with a gradient step. A slightly infeasible solution $\vec{x}$ is repaired with the help of the *surrogate function* $v(\vec{x})$ of a constraint (i. e. we do not need any expensive evaluations of the true function): We estimate the gradient $g(\vec{x})$ of the constraint surrogate. If the linear approximation near $\vec{x}$ holds, an appropriate step from $\vec{x}$ to a feasible solution $\vec{z}$ can be easily calculated.

3.5 Repairing infeasible solutions

Sometimes very small constraint violations occur for infill points, because the internal optimization method could not determine a feasible solution, or the solution returned by the optimizer was assumed to be feasible on the surrogates, but turns out to be infeasible after evaluation on the real function. As a consequence this can lead to a unwanted discarding of good but slightly infeasible solutions.

The *repair infeasible* method has been especially designed for internal optimization strategies inside COBRA like COBYLA. In most of our runs with COBYLA, it turned out that the infill points often have very small constraint violations. For COBYLA, it took a relatively long time to resolve such small constraint violations. To circumvent this issue we integrated a very simple gradient descent strategy for resolving small constraint violations. In Fig. 4 we describe the general sketch of the *repair infeasible* algorithm. The repair infeasible technique in our approach and [2] are both based on utilizing the gradient information from constraints. In [2], the repairing operation is embedded into a Genetic Algorithm and the infeasible results are repaired with a defined probability by deriving the gradient of the real constraint functions to direct the infeasible point towards the feasible region. Our approach only relies on constraint surrogates and does not impose any extra real function evaluations.

In Fig. 5 we show the performance of COBRA on function G06 with and without *repair infeasible*. It can be seen from the plot that after a first phase with similar progress of both variants the variant with activated repair infeasible can approximate the optimum much better. The progress is very fast with neat improvements during iterations 40 to 50. In contrast the variant without *repair infeasible* stagnates at an inferior level.

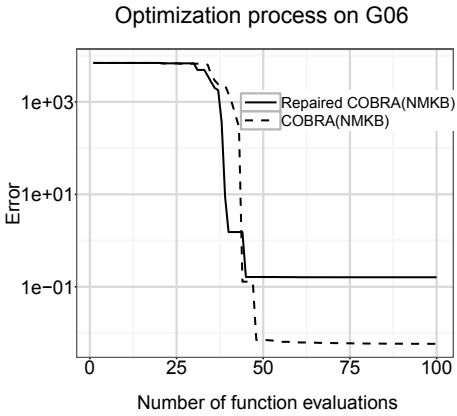


Figure 5: Impact of using the repair infeasible technique within COBRA(NMKB) on the G06 problem.

4 Discussion

In this section we discuss the components of the COBRA algorithm and want to draw the attention on common pitfalls and drawbacks of the method.

4.1 Initial design

The RBF models used internally in COBRA need at least $d+1$ points (where d is the dimension of the search space) to fit an interpolating model. In the COBRA algorithm an initial population is created as a first step (see Fig. 1). The method used for this can be designed by the user, but usually methods from statistics are chosen therefore. We have integrated the strategies described in Sec. 3.3, but depending on the problem also other designs are possible. Although with $d+1$ a lower bound exists for the initial design points and can be used by the algorithm, it can be advantageous to increase this number and to not rely on the minimum number of initial design points.

In our experimental study it was crucial to find a good strategy for the initial design generation and number of design points. We found these settings to be important for the later performance of the whole optimization run. Both settings can be very problem-dependent, but we showed several examples, where the selection of the right strategy makes a difference. A

general rule-of-thumb for the number of initial design points can be given with $3d$, where d is the dimension of the problem.

4.2 Internal optimization strategy

The selection of the internal optimization strategy in COBRA can be crucial to find good solutions for some problems. E.g., in highly multimodal landscapes, local optimization methods such as COBYLA, HJKB or NMKB are not well-suited. In fact these methods are designed for local optimization and will probably fail on multimodal landscapes. A possible solution to this problem can be to implement a restarting strategy, which from time to time makes random restarts and does not spent the whole budget for the optimization of the starting solution. Other solutions can be to select global optimization strategies like ISRES, which are also available in our `R` implementation of COBRA.

4.3 Distance requirement

In the literature a lot of work has been devoted to balancing exploration and exploitation of the search space. In COBRA, the user controls exploration and exploitation by setting the distance requirement parameter Ξ. Large values in Ξ lead to explorative steps, while smaller values enable closer approximations of the optima. However, too many large values can lead to uncontrolled jumping through the search space, whereas too many small values can lead to a stagnating behaviour in suboptimal regions of the search space. For this reason good settings of Ξ are necessary and must be defined anew for each test case. As a rule of thumb, Regis [13] proposes $\Xi_{local} = \langle 0.01, 0.001, 0.0005 \rangle$ for a locally-biased distance requirement, and $\Xi_{global} = \langle 0.1, 0.05, 0.01, 0.005, 0.001, 0.0005 \rangle$ for a more globally biased distance requirement. Note that these settings are only suggestions, specific properties of other problems might require to define other settings.

In our experiments we sometimes discovered, that a large element (i.e. $\xi_i = 0.03$) or a very small element ($\xi_i = 0.0$) in the set can have a beneficial effect. The large element can support the algorithm to make larger steps in the search space, which can be advantageous for highly multimodal functions. The small element instead can help to find better approximations of the target. Fig. 6 shows the infill solutions obtained in a run on function G06 with a small ξ_i of 0.0 included. Stepwise the points move closer to the

Table 4: Challenges of G-problems and MOPTA (MO) and their possible solutions in COBRA

	Challenge(s)	**Solution(s)**
G01	Small feasible region. Often solutions with slightly violated constraints.	Add 0.3 to DRC (exploration). Use repairInfeasible.
G02	Multimodal: Many local optima, especially in 20d	none!
G03	Nonlinear and non-separable objective $\rightarrow$ surrogate model not accurate. High dimension and large range R.	Logarithmic transform
G04	Fitness function and constraints with mixed terms $x_1x_2 \rightarrow$ difficult for constraint surrogates.	Use optimizer COBYLA (others fail: NMKB, ISRES)
G05	Extremely thin feasible region. Three nonlinear active constraints. Highly varying input ranges.	Add 0.0 to DRC, use optimizer COBYLA. Rescale inputs to $[0,1]^d$.
G06	Very thin feasible region, optimum at „tip of needle“. Steep objective function, large range R.	Add 0.0 to DRC (avoid „blocking“ the optimum)
G07	Constraints with mixed terms $x_1x_2 \rightarrow$ difficult for constraint surrogates. Very small ρ = 0.0001%.	Use optimizer COBYLA (others fail: NMKB, ISRES)
G08	Shallow optimum in feasible region is masked by high $(+/-)$ infeasible peaks	Use optimizer ISRES (others fail: NMKB, COBYLA)
MO	Very high d=124 and m=68. Often solutions with slightly violated constraints.	

DRC: distance requirement cycle

R: min-max spread of objective function over search space (see Table 1)

ρ: percentage of feasible volume in search space volume (see Table 1)

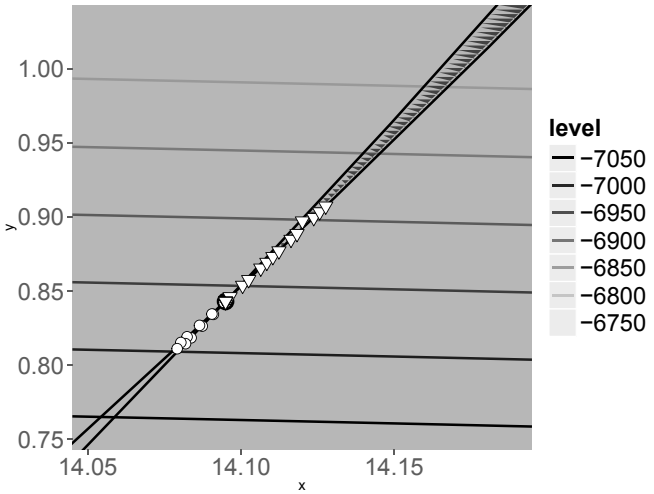


Figure 6: Optimization with COBRA-R on function G06 **with** $\xi_i = 0.0$ in the set. Triangles and circles are representing feasible and infeasible points. The big black circle is the true optimum, resp. The hatched area is the approximation of feasible region.

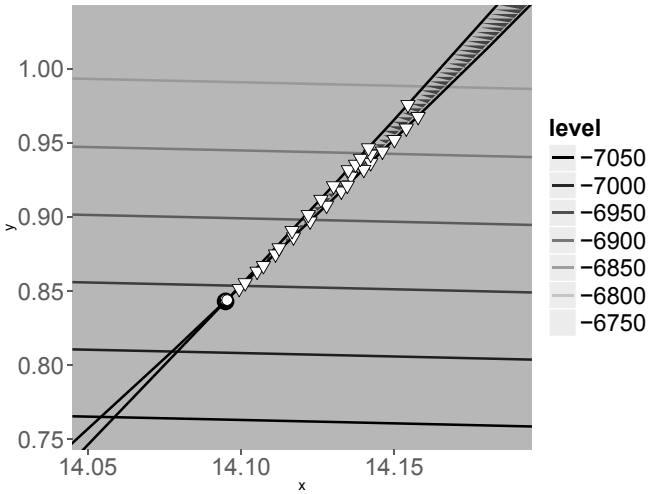


Figure 7: Optimization with COBRA-R on function G06 **without** $\xi_i = 0.0$ in the set.

optimum along the shaded area, which is located at the needle point of this area. Outside the shaded area the points are infeasible which makes this problem very hard for any optimizer, because the feasible region around the optimum becomes very small. With larger values for the Ξ parameter, it would not be possible to exploit solutions in the optimal region. By switching off the distance requirement method from time to time, the algorithm allows solutions in the direct neighbourhood of the known points and can move closer to the real optimum.

In Table 4 we summarize this discussion by showing the quite different challenges posed by each optimization problem and indicate possible solutions found in the framework of COBRA-R to cope with this challenges.

5 Conclusion and outlook

In this paper we presented a comparative study on constrained optimization problems under limited budgets. Therefore we developed a new implementation of the model-assisted algorithm COBRA in R. With our new implementation of COBRA we give users full flexibility for changing or adapting single components of the algorithm. In an experimental study on common benchmark functions we give evidence that our COBRA implementation can reach very good accuracies on most of the functions discussed in this paper (G01, G04, G05, G06, G07 and G08). The experiments showed, that the obtained results of COBRA-R are similar or close to the results of other constrained optimization algorithms including ISRES, RGA and COBYLA. We want to emphasize that the other strategies require a much higher number of function evaluations. Thus, as one of the main contributions of this paper, COBRA can achieve much faster convergence rates with almost similar accuracies on most functions. As only drawback we found negative results on the functions G02 and G03, presumably due to the dimensionality and complexity of these functions ($20d$ and multimodality in the case of G02, highly nonlinear in the case of G03).

Few questions remain open, e.g., the generation of a good initial design, the optimal settings for the distance requirement, or the choice of the best performing optimization strategy on the surrogate functions. In future work we want to elaborate on that and strengthen the advantages of model-assisted optimization under heavily restricted budgets. We are looking forward to improve our promising initial results on the MOPTA real-world problem by providing a deeper analysis of the hyperparameters of the algorithm.

References

[1] Oyman, A.; Deb, K.; Beyer, H.-G.: An alternative constraint handling method for evolution strategies. In: *Proceedings of Congress on Evolutionary Computation (CEC)*, S. 612–619. 1999.

[2] Chootinan, P.; Chen, A.: Constraint handling in genetic algorithms using a gradient-based repair method. *Computers & Operations Research* 33 (2006) 8, S. 2263–2281.

[3] Kramer, O.: A review of constraint-handling techniques for evolution strategies. *Applied Computational Intelligence and Soft Computing* 2010 (2010), S. 1–11.

[4] Mezura-Montes, E.; Coello Coello, C.: Constraint-handling in nature-inspired numerical optimization: past, present and future. *Swarm and Evolutionary Computation* 1 (2011) 4, S. 173–194.

[5] Coello Coello, C.: Constraint-handling techniques used with evolutionary algorithms. In: *Proceedings of the 14th International Conference on Genetic and Evolutionary Computation Conference (GECCO)*, S. 849–872. ACM. 2012.

[6] Takahama, T.; Sakai, S.: Constrained optimization by applying the alpha-constrained method to the nonlinear simplex method with mutations. *IEEE Trans. Evolutionary Computation* 9 (2005) 5, S. 437–451.

[7] Takahama, T.; Sakai, S.: Learning fuzzy control rules by alpha-constrained Simplex method. *Systems and Computers in Japan* 34 (2003) 6, S. 80–90.

[8] Runarsson, T.; Yao, X.: Search biases in constrained evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 35 (2005) 2, S. 233–243.

[9] Aguirre, A. H.; Rionda, S. B.; Coello Coello, C. A.; Lizárraga, G. L.; Montes, E. M.: Handling constraints using multiobjective optimization concepts. *International Journal for Numerical Methods in Engineering* 59 (2004) 15, S. 1989–2017.

[10] Beyer, H.-G.; Finck, S.: On the design of constraint covariance matrix self-adaptation evolution strategies including a cardinality constraint. *IEEE Transactions on Evolutionary Computation* 16 (2012) 4, S. 578–596.

[11] Poloczek, J.; Kramer, O.: Local SVM Constraint Surrogate Models for Self-adaptive Evolution Strategies. In: *KI 2013: Advances in Artificial Intelligence*, S. 164–175. Springer. 2013.

[12] Powell, M.: A Direct Search Optimization Method That Models The Objective And Constraint Functions By Linear Interpolation. In: *Advances In Optimization And Numerical Analysis*, S. 51–67. Springer. 1994.

[13] Regis, R.: Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. *Engineering Optimization* 46 (2013) 2, S. 218–243.

[14] Buhmann, M.: *Radial Basis Functions: Theory and Implementations*. Cambridge University Press. 2003.

[15] Powell, M.: The Theory of Radial Basis Function Approximation in 1990. *Advances In Numerical Analysis* 2 (1992), S. 105–210.

[16] Regis, R.: Particle swarm with radial basis function surrogates for expensive black-box optimization. *Journal of Computational Science* 5 (2014) 1, S. 12–23.

[17] Wild, S.; Shoemaker, C.: Global Convergence Of Radial Basis Function Trust-Region Derivative-Free Algorithms. *SIAM Journal on Optimization* 21 (2011) 3, S. 761–781.

[18] Hooke, R.; Jeeves, T.: "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the ACM (JACM)* 8 (1961) 2, S. 212–229.

[19] Nelder, J.; Mead, R.: A simplex method for function minimization. *The Computer Journal* 7 (1965) 4, S. 308–313.

[20] Michalewicz, Z.; Schoenauer, M.: Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation* 4 (1996) 1, S. 1–32.

[21] Jones, D.: Large-scale multi-disciplinary mass optimization in the auto industry. In: *Modeling And Optimization: Theory And Applications (MOPTA) 2008 Conference, Ontario, Canada*, S. 1–58. 2008.

Parallel Distributed Compensation for Piecewise Bilinear Models and Recurrent Fuzzy Systems Based on Piecewise Quadratic Lyapunov Functions

Stefan Gering, Jürgen Adamy
TU Darmstadt, Institut für Automatisierungstechnik
Landgraf-Georg-Str. 4
64283 Darmstadt
E-Mail:
sgering@rmr.tu-darmstadt.de,
jadamy@rmr.tu-darmstadt.de

Luka Eciolaza, Michio Sugeno
European Center for Soft Computing
Calle Gonzalo Gutiérrez Quirós S/N
33600 Mieres, Spain
E-Mail:
luka.eciolaza@softcomputing.es,
michio.sugeno@softcomputing.es

Abstract: Piecewise Bilinear Models and Recurrent Fuzzy Systems are universal approximators for any smooth nonlinear dynamics. One of their advantage is the efficient representation of the modeled system dynamics by means of rule-bases or look-up-tables. In this paper, it is shown how to obtain provably stabilizing controllers by means of piecewise quadratic Lyapunov functions. Interpolating controllers with affine local controllers are considered for interpolation, akin to the concept of parallel distributed compensation widely used for control of Takagi-Sugeno systems.

1 Introduction

In order to analyze and control nonlinear systems, it is well known that besides exact methods, approximate approaches may be utilized. In the context of fuzzy logic-based modeling and control of dynamic systems, Takagi-Sugeno (T-S) systems [1] are probably the most prominent models allowing for an approximate system representation. In addition, Piecewise Bilinear Models (PBM) [2], [3] have been investigated over the past decade, which can be seen as special case of T-S systems. As universal approximators,

they are also able to represent any smooth nonlinear dynamics, yet compared to T-S systems, their advantage is in the efficient representation of the system dynamics as look-up table. This makes PBM particularly interesting for industrial applications, where many times functional dependencies are given only approximately as look-up tables.

Independently of PBM, Recurrent Fuzzy Systems (RFS) [4] have evolved at the same time. Although the primary intention for this system class was to obtain a linguistically interpretable model based on fuzzy automata, due to recent developments they are utilized for control purposes as well. As found by the authors, both system classes are indeed similar and in special cases even identical, which makes it amendable to answer common questions related to both system classes in a common framework.

For both system classes, a number of literature exists dealing with the question of controller synthesis for the respected models. In the context of PBM, two main lines of research were followed up until now: The first is based on look-up table controllers which are derived for a given PBM by means of feedback-linearization [5] or a design by means of the vertex placement principle [6]. In a second approach, feedforward controllers are considered, which are obtained from feedback error learning [7].

In the case of RFS, fuzzy controllers [8], piecewise-polynomial controllers [9] and switching controllers [10] were discussed, whereas the latter approach focused especially on robust control.

In this paper, we now discuss the synthesis of provably stabilizing controllers for PBM and RFS, whereas the concept of parallel distributed compensation is used. The stability criterion is based on piecewise quadratic Lyapunov functions, which in the context of T-S systems were treated, e.g., in [11]. Based on the latter work, a similar discussion of stability analysis for PBM and RFS was carried out in detail in [12], which is now the starting point of the controller synthesis presented in this paper.

The remainder is organized as follows: Sec. 2 shortly reviews necessary definitions of PBM and RFS, whereas a review of the stability criterion under consideration is given in Sec. 3. The synthesis of state feedback is

then given in Sec. 4, whereas practical hints considering the implementation are given in Sec. 5. The method is applied to the inverted pendulum example in Sec. 6, and concluding remarks are given in Sec. 7.

2 Preliminaries

In this section, the basic definitions of PBM and RFS are reviewed as detailed, e.g., in [2] and [13]. It is thereby shown that both system classes follow the same idea of universal approximators for smooth nonlinear dynamics and in some cases and under mild constraints are even of identical structure.

2.1 Piecewise Bilinear Systems

In order to model nonlinear system dynamics by means of PBM, it is assumed beforehand that these are input-affine, i.e.,

$$\dot{\mathbf{x}} = \mathbf{f}_{\text{nl}}(\mathbf{x}) + \mathbf{G}_{\text{nl}}(\mathbf{x})\mathbf{u}. \tag{1}$$

By discretizing (1) at finitely many discrete vertices $\mathbf{v_j}$, such that

$$\mathbf{x} = \sum_{\mathbf{j}} \mathbf{v_j} \prod_{i=1}^{n} w_{j_i}(x_i), \tag{2}$$

$\sum_{\mathbf{j}} w_{j_i} = 1$, $w_{j_i}(x_i) \in [0,1]$, the samples $\dot{\mathbf{v}}_{\mathbf{j}} = \mathbf{f}_{\text{nl}}(\mathbf{v_j})$ and $\mathbf{G}_{\text{nl}}(\mathbf{v_j})$ thus obtained may be stored in a look-up table. Thus, PBM become particularly suitable for practical applications, and can at the same time be given as rules of the form

$$\begin{aligned} &\text{If } \mathbf{x} = \mathbf{v_j}, \\ &\text{then } \dot{\mathbf{x}} = \dot{\mathbf{v}}_{\mathbf{j}} + \mathbf{G}_{\text{nl}}(\mathbf{v_j})\mathbf{u}. \end{aligned} \tag{3}$$

Hence, by interpolating between rules (3) in every dimension of the state space using weights $w_{j_i}(x_i)$, the dynamics of the PBM

$$\dot{\mathbf{x}} = \sum_{\mathbf{j}} (\dot{\mathbf{v}}_{\mathbf{j}} + \mathbf{G}(\mathbf{v}_{\mathbf{j}})\mathbf{u}) \prod_{i=1}^{n} w_{j_i}(x_i) \tag{4}$$

are obtained, which are given here in *parameter form* [2]. Due to the sampling, a grid is introduced by the vertices $\mathbf{v}_{\mathbf{j}}$, whereas each rectangular region between $\mathbf{v}_{\mathbf{j}}$ and $\mathbf{v}_{\mathbf{j}+\mathbf{1}}$ is abbreviated $R_{\mathbf{j}} = \text{conv}\{\mathbf{v}_{\mathbf{j}}, \mathbf{v}_{\mathbf{j}+\mathbf{1}}\}$. Although these regions are hypersquares in general, the term *rectangle* is used for simplicity.

Similar to the dynamic function, the output function may be taken into consideration by means of the same sampling technique, which is omitted here due to space restrictions.

2.2 Recurrent Fuzzy Systems

In contrast to PBM, RFS allow for the modeling of any smooth nonlinear dynamics

$$\dot{\mathbf{x}} = \mathbf{f}_{\text{nl}}(\mathbf{x}, \mathbf{u}) \tag{5}$$

without assuming input-affinity. By sampling (5) on a grid at discrete vertex positions $(\mathbf{v}_{\mathbf{j}}, \mathbf{v}_{\mathbf{q}})$ akin to the PBM, gradients

$$\dot{\mathbf{v}}_{\mathbf{j},\mathbf{q}} = \mathbf{f}_{\text{nl}}(\mathbf{v}_{\mathbf{j}}, \mathbf{v}_{\mathbf{q}}) \tag{6}$$

are obtained. Furthermore, by introducing vectors of linguistic values $\mathbf{L}^{\mathbf{x}}_{\mathbf{j}}$, $\mathbf{L}^{\mathbf{u}}_{\mathbf{q}}$ and $\mathbf{L}^{\dot{\mathbf{x}}}_{\mathbf{j},\mathbf{q}}$ which are associated with crisp vertices $(\mathbf{v}_{\mathbf{j}}, \mathbf{v}_{\mathbf{q}})$ and gradients $\dot{\mathbf{v}}_{\mathbf{j},\mathbf{q}}$, the dynamics of the RFS

$$\begin{aligned} &\text{If } \mathbf{x} = \mathbf{L}^{\mathbf{x}}_{\mathbf{j}} \text{ and } \mathbf{u} = \mathbf{L}^{\mathbf{u}}_{\mathbf{q}}, \\ &\text{then } \dot{\mathbf{x}} = \mathbf{L}^{\dot{\mathbf{x}}}_{\mathbf{j},\mathbf{q}} \end{aligned} \tag{7}$$

are obtained for each vertex. In order to obtain again crisp gradient values from the RFS, membership functions $w_{j_i}(x_i), w_{q_p}(u_p) \in [0, 1]$ are introduced fuzzifying the linguistic variables. Following [13], the algebraic product is then chosen for aggregation and implication, such that the premise

weights

$$w_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u}) = w_{\mathbf{j}}(\mathbf{x}) w_{\mathbf{q}}(\mathbf{u}) = \prod_{i=1}^{n} w_{j_i}(x_i) \prod_{p=1}^{m} w_{q_p}(u_p) \tag{8}$$

are obtained. Evaluating the implication as

$$w^{\text{imp}}_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u}) = \dot{\mathbf{v}}_{\mathbf{j},\mathbf{q}} w_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u}) \tag{9}$$

and using the algebraic sum for accumulation,

$$w^{\text{acc}}_{\mathbf{j},\mathbf{q}} = \sum_{\mathbf{j},\mathbf{q}} w^{\text{imp}}_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u}) \tag{10}$$

is obained. Finally, by making use of the center of singleton defuzzification, crisp values

$$\dot{\mathbf{x}} = \frac{w^{\text{acc}}_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u})}{\sum_{\mathbf{j},\mathbf{q}} w_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u})} \tag{11}$$

are obtained. If furthermore $\sum_{\mathbf{j},\mathbf{q}} w_{\mathbf{j},\mathbf{q}}(\mathbf{x},\mathbf{u}) = 1$ holds, again a parametric form

$$\dot{\mathbf{x}} = \sum_{\mathbf{j},\mathbf{q}} \dot{\mathbf{v}}_{\mathbf{j},\mathbf{q}} \prod_{i=1}^{n} w_{j_i}(x_i) \prod_{p=1}^{m} w_{q_p}(u_p) \tag{12}$$

akin to (4) is obtained, showing the similarity between PBM and RFS.

2.3 Comparison

Comparing the system structure of PBM and RFS, it becomes obvious that their main difference is in the treatment of the input space. Whereas in the case of PBM, the class of system dynamics are restricted to input-affine systems, RFS allow for more general dynamics. Nevertheless, the restriction to input-affine systems is reasonable under a practical viewpoint, since this assumption holds for a variety of technical systems. In addition, the controller synthesis is considerably facilitated with this restriction.

From a structural point of view, it can be noted that for the special cases of unforced systems and single-input systems, PBM and RFS are fully equi-

valent. For $m > 1$, i.e., systems with multiple inputs, this equivalence does not hold in general. On the other hand, RFS can be rendered input-affine by means of a dynamic transformation, such that a conversion into PBM is again possible.

For both system classes, nothing was said until now about the type of membership function. In general, any kind of membership function is applicable fulfilling he properties $w_{j_i} \in [0, 1]$ and $\sum_{\mathbf{j}} w_{j_i} = 1$. On the other hand, the restriction to ramp- and triangular-shaped membership functions is considered due to several reasons: First, the interpolation by means of piecewise affine functions is easy to implement. Second, these membership functions are nonzero only in a local region, supporting the idea of a piecewise locally defined system. In addition, the simplicity of triangular membership functions allows for the derivation of system dynamics in a piecewise polynomial form, which can furthermore be expressed in a matrix form, being particularly suitable for implementation.

3 Stability Analysis

This section outlines the stability analysis of PBM and RFS by means of piecewise quadratic Lyapunov functions as discussed in [12]. It will form the basis for the controller synthesis, which is detailed in Sec. 4. Piecewise quadratic Lyapunov have already been utilized in the context of hybrid systems [14] or affine T-S fuzzy systems [11]. We closely follow these discussions, but apply the approach to open-loop PBM and RFS

$$\dot{\mathbf{x}} = \sum_{\mathbf{j}} \dot{\mathbf{v}}_{\mathbf{j}} \prod_{i=1}^{n} w_{j_i}(x_i). \tag{13}$$

In the following an equilibrium $(\mathbf{x}^*, \mathbf{u}^*)$ is assumed at the origin and incident with a vertex, which is without loss of generality, because each PBM and RFS may be augmented with a vertex at $(\mathbf{x}^*, \mathbf{u}^*)$ without changing the overall dynamics.

Starting with the well-known Lyapunov equations

$$V(\mathbf{x}) > 0, \quad \forall \mathbf{x} \neq \mathbf{0}, \tag{14a}$$

$$\dot{V}(\mathbf{x}) < 0, \quad \forall \mathbf{x} \neq \mathbf{0}, \tag{14b}$$

$$V(\mathbf{0}) = 0, \tag{14c}$$

a local approximation

$$V(\mathbf{x}) \approx r_\mathbf{j} + 2\mathbf{q}_\mathbf{j}^T\mathbf{x} + \mathbf{x}^T\mathbf{P}_\mathbf{j}\mathbf{x}, \quad \mathbf{x} \in R_\mathbf{j} \tag{15}$$

is imposed on the Lyapunov function. A reformulation in terms of matrix multiplications then yields

$$V(\mathbf{x}) \approx \begin{bmatrix}\mathbf{x}\\1\end{bmatrix}^T \begin{bmatrix}\mathbf{P}_\mathbf{j} & \mathbf{q}_\mathbf{j}\\ \mathbf{q}_\mathbf{j}^T & r_\mathbf{j}\end{bmatrix} \begin{bmatrix}\mathbf{x}\\1\end{bmatrix} = \overline{\mathbf{x}}^T\overline{\mathbf{P}}_\mathbf{j}\overline{\mathbf{x}}, \quad \mathbf{x} \in R_\mathbf{j}. \tag{16}$$

Furthermore, it is assumed that

$$r_\mathbf{j} = 0, \quad \mathbf{q}_\mathbf{j} = \mathbf{0}, \quad \forall R_\mathbf{j} \in \mathcal{R}_0, \tag{17}$$

in order for (14c) to hold. Therein, $\mathcal{R}_0$ denotes the set of rectangles containing the origin. (16) can then be considered as piecewise-defined function

$$\Rightarrow V(\mathbf{x}) = \begin{cases}\mathbf{x}^T\mathbf{P}_\mathbf{j}\mathbf{x}, & R_\mathbf{j} \in \mathcal{R}_0,\\ \overline{\mathbf{x}}^T\overline{\mathbf{P}}_\mathbf{j}\overline{\mathbf{x}}, & R_\mathbf{j} \notin \mathcal{R}_0.\end{cases} \tag{18}$$

It is crucial for (18) to be continuous across rectangle borders in order to guarantee stability. As shown in [11], this can be ensured by decomposing the matrices of Lyapunov function candidates according to

$$\mathbf{P}_\mathbf{j} = \mathbf{F}_\mathbf{j}^T\mathbf{T}\mathbf{F}_\mathbf{j}, \quad R_\mathbf{j} \in \mathcal{R}_0, \tag{19a}$$

$$\overline{\mathbf{P}}_\mathbf{j} = \overline{\mathbf{F}}_\mathbf{j}^T\mathbf{T}\overline{\mathbf{F}}_\mathbf{j}, \quad R_\mathbf{j} \notin \mathcal{R}_0, \tag{19b}$$

where $\overline{\mathbf{F}}_\mathbf{j} = \begin{bmatrix}\mathbf{F}_\mathbf{j} & \mathbf{f}_\mathbf{j}\end{bmatrix}$ and $\mathbf{f}_\mathbf{j} = \mathbf{0}$ for $R_\mathbf{j} \in \mathcal{R}_0$ are introduced, fulfilling the facet conditions

$$\overline{\mathbf{F}}_\mathbf{i}\overline{\mathbf{x}} = \overline{\mathbf{F}}_\mathbf{j}\overline{\mathbf{x}}, \quad \mathbf{x} \in R_\mathbf{i} \cap R_\mathbf{j}. \tag{20}$$

Then,

$$\dot{V}(\mathbf{x}) = \begin{cases} 2\mathbf{x}^T\mathbf{P}_\mathbf{j}\dot{\mathbf{x}}, & R_\mathbf{j} \in \mathcal{R}_0, \\ 2\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}^T \overline{\mathbf{P}}_\mathbf{j} \begin{bmatrix}\dot{\mathbf{x}}\\1\end{bmatrix}, & R_\mathbf{j} \notin \mathcal{R}_0 \end{cases} \tag{21}$$

follows from (14b). Herein, we now substitute the state and state derivative by (2) and (13), which are expressed as convex hull of vertices and gradients, i.e.,

$$\begin{aligned} \mathbf{x} &= \sum_{\mathbf{j}\backslash\mathbf{0}} \mathbf{v}_\mathbf{j} \cdot w_\mathbf{j}(\mathbf{x}) + \mathbf{0} \cdot w_\mathbf{0}(\mathbf{x}) \\ &= \sum_{\mathbf{j}\backslash\mathbf{0}} \mathbf{v}_\mathbf{j} \cdot w_\mathbf{j}(\mathbf{x}) = \operatorname*{conv}_{\mathbf{j}\backslash\mathbf{0}} \{\mathbf{v}_\mathbf{j}\}, \quad \mathbf{x} \in R_\mathbf{j} \end{aligned} \tag{22}$$

and

$$\begin{aligned} \dot{\mathbf{x}} &= \sum_{\mathbf{j}\backslash\mathbf{0}} \dot{\mathbf{v}}_\mathbf{j} \cdot w_\mathbf{j}(\mathbf{x}) + \underbrace{\dot{\mathbf{v}}_\mathbf{0}}_{=\mathbf{0}} \cdot w_\mathbf{0}(\mathbf{x}) \\ &= \sum_{\mathbf{j}\backslash\mathbf{0}} \dot{\mathbf{v}}_\mathbf{j} \cdot w_\mathbf{j}(\mathbf{x}) = \operatorname*{conv}_{\mathbf{j}\backslash\mathbf{0}} \{\dot{\mathbf{v}}_\mathbf{j}\}, \quad \mathbf{x} \in R_\mathbf{j}. \end{aligned} \tag{23}$$

For simplicity, the shorthand notation $\mathbf{j}\backslash\mathbf{0}$ was introduced meaning all vertices $\mathbf{j}$ not corresponding to the equilibrium.

Substituting (22) and (23) into (21) then yields

$$\begin{aligned} \dot{V}(\mathbf{x}) &\in \begin{cases} 2\operatorname*{conv}_{\mathbf{i}\backslash\mathbf{0}} \left\{\mathbf{v}_\mathbf{i}^T\right\} \mathbf{P}_\mathbf{k} \operatorname*{conv}_{\mathbf{j}\backslash\mathbf{0}} \{\dot{\mathbf{v}}_\mathbf{j}\}, & R_\mathbf{k} \in \mathcal{R}_0, \\ 2\operatorname*{conv}_{\mathbf{i}\backslash\mathbf{0}} \left\{ \begin{bmatrix}\mathbf{v}_\mathbf{i}\\1\end{bmatrix}^T \right\} \overline{\mathbf{P}}_\mathbf{k} \operatorname*{conv}_{\mathbf{j}\backslash\mathbf{0}} \left\{ \begin{bmatrix}\dot{\mathbf{v}}_\mathbf{j}\\1\end{bmatrix} \right\}, & R_\mathbf{k} \notin \mathcal{R}_0, \end{cases} \\ &= \begin{cases} 2\operatorname*{conv}_{\mathbf{i},\mathbf{j}\backslash\mathbf{0}} \left\{\mathbf{v}_\mathbf{i}^T\mathbf{P}_\mathbf{k}\dot{\mathbf{v}}_\mathbf{j}\right\}, & \mathbf{v}_\mathbf{i}, \mathbf{v}_\mathbf{j} \in R_\mathbf{k} \in \mathcal{R}_0, \\ 2\operatorname*{conv}_{\mathbf{i},\mathbf{j}\backslash\mathbf{0}} \left\{ \begin{bmatrix}\mathbf{v}_\mathbf{i}\\1\end{bmatrix}^T \overline{\mathbf{P}}_\mathbf{k} \begin{bmatrix}\dot{\mathbf{v}}_\mathbf{j}\\1\end{bmatrix} \right\}, & \mathbf{v}_\mathbf{i}, \mathbf{v}_\mathbf{j} \in R_\mathbf{k} \notin \mathcal{R}_0. \end{cases} \end{aligned} \tag{24}$$

Hence, (13) is asymptotically stable, if

$\forall\ R_\mathbf{k} \in \mathcal{R}_0$:

$$\mathbf{P}_\mathbf{k} \succ 0, \tag{25a}$$
$$\mathbf{v}_\mathbf{i}^T \mathbf{P}_\mathbf{k} \dot{\mathbf{v}}_\mathbf{j} < 0, \quad \mathbf{v}_\mathbf{i}, \mathbf{v}_\mathbf{j} \in R_\mathbf{k} \backslash \mathbf{0}, \tag{25b}$$

$\forall\ R_\mathbf{k} \notin \mathcal{R}_0$:

$$\overline{\mathbf{P}}_\mathbf{k} \succ 0, \tag{25c}$$
$$\begin{bmatrix} \mathbf{v}_\mathbf{i} \\ 1 \end{bmatrix}^T \overline{\mathbf{P}}_\mathbf{k} \begin{bmatrix} \dot{\mathbf{v}}_\mathbf{j} \\ 1 \end{bmatrix} < 0, \quad \mathbf{v}_\mathbf{i}, \mathbf{v}_\mathbf{j} \in R_\mathbf{k}. \tag{25d}$$

In order to account for the locality of these equations, the S-procedure (see, e.g., [15]) is utilized, such that inequalities (25) have to hold on $R_\mathbf{k}$ only instead of the entire state space. This relaxation comes at the cost that it renders the stability conditions sufficient only, because the S-procedure is itself in general only a sufficient conditions.

To apply the S-procedure for each rectangular regions $R_\mathbf{j}$, they are described by means of $2n$ bounding hyperplanes, such that

$$\overline{\mathbf{E}}_\mathbf{j} \overline{\mathbf{x}} \geq \mathbf{0}, \quad \mathbf{x} \in R_\mathbf{j}, \tag{26}$$

with $\overline{\mathbf{E}}_\mathbf{j} \in \mathbb{R}^{2n \times n+1}$ as detailed in [11]. Then, stability conditions for system (13) in relaxed form read

$\forall\ R_\mathbf{k} \in \mathcal{R}_0$:

$$\mathbf{P}_\mathbf{k} \succ 0, \tag{27a}$$
$$\mathbf{v}_\mathbf{i}^T \mathbf{P}_\mathbf{k} \dot{\mathbf{v}}_\mathbf{j} < 0, \quad \mathbf{v}_\mathbf{i}, \mathbf{v}_\mathbf{j} \in R_\mathbf{k} \backslash \mathbf{0}, \tag{27b}$$

$\forall\ R_\mathbf{k} \notin \mathcal{R}_0$:

$$\overline{\mathbf{P}}_\mathbf{k} - \overline{\mathbf{E}}_\mathbf{k}^T \mathbf{S}_\mathbf{k} \overline{\mathbf{E}}_\mathbf{k} \succ 0, \tag{27c}$$
$$\begin{bmatrix} \mathbf{v}_\mathbf{i} \\ 1 \end{bmatrix}^T \overline{\mathbf{P}}_\mathbf{k} \begin{bmatrix} \dot{\mathbf{v}}_\mathbf{j} \\ 1 \end{bmatrix} < 0, \quad \mathbf{v}_\mathbf{i}, \mathbf{v}_\mathbf{j} \in R_\mathbf{k}, \tag{27d}$$

with $\mathbf{S_k} \geq \mathbf{0}$ being component-wise non-negative. It has to be emphasized that only (27c) has to be relaxed by means of the S-procedure, whereas (27b) and (27d) are scalar inequalities and thus inherently local.

As additional constraints, the region of attraction $\mathcal{E} = \{\mathbf{x} \mid V(\mathbf{x}) \leq 1\}$ has to be contained in the set of rectangles $\mathcal{R}$. This was discussed, e.g., in [16] for T-S systems. For $\mathcal{R}$ being a symmetric polytope around the origin, this interpolation region can be rewritten as

$$\mathcal{R} = \{\mathbf{x} \,\big|\, |\mathbf{a}_i^T \mathbf{x}| \leq 1, \quad i = 1, \ldots, 2n\}. \tag{28}$$

Then, following [15], the additional constraint

$$\mathbf{a}_i^T \mathbf{P}_\mathbf{k}^{-1} \mathbf{a}_i \leq 1, \quad \forall\, \mathbf{a}_i \in R_\mathbf{k} \in \mathcal{R}_0, \tag{29a}$$

$$\begin{bmatrix} \mathbf{a}_i \\ 1 \end{bmatrix}^T \overline{\mathbf{P}}_\mathbf{k}^{-1} \begin{bmatrix} \mathbf{a}_i \\ 1 \end{bmatrix} \leq 1, \quad \forall\, \mathbf{a}_i \in R_\mathbf{k} \notin \mathcal{R}_0, \tag{29b}$$

for all $R_\mathbf{k} \cap \partial\mathcal{R} \neq \emptyset$ guarantees that $\mathcal{E} \subseteq \mathcal{R}$. Applying Schur complement, (29) can be rewritten as

$$\begin{bmatrix} \mathbf{P}_\mathbf{k} & \mathbf{a}_i \\ \mathbf{a}_i^T & 1 \end{bmatrix} \succ 0, \quad \forall\, \mathbf{a}_i \in R_\mathbf{k} \in \mathcal{R}_0, \tag{30a}$$

$$\begin{bmatrix} \overline{\mathbf{P}}_\mathbf{k} & \overline{\mathbf{a}}_i \\ \overline{\mathbf{a}}_i^T & 1 \end{bmatrix} \succ 0, \quad \forall\, \mathbf{a}_i \in R_\mathbf{k} \notin \mathcal{R}_0, \tag{30b}$$

with $\overline{\mathbf{a}}_i = \begin{bmatrix} \mathbf{a}_i^T & 1 \end{bmatrix}^T$. These conditions may lead to conservative results for sets $\mathcal{R}$ being non-symmetric around the origin. A way to circumvent this problem is proposed in [12].

4 Synthesis of Parallel Distributed Compensation

Based on the previously discussed stability analysis, this section is devoted to the synthesis of local affine controllers stabilizing a given PBM/RFS. Because the input space has to be considered for the controller synthesis, the discussion will be carried out for the more general RFS, which include PBM as special case.

We assume for each vertex $\mathbf{v_j} \in \mathcal{R} \subset \mathbb{R}^n$ an affine local controller $\mathbf{K_j x}+\mathbf{k_j}$, with $\mathbf{k_j} = \mathbf{0}$ for $\mathbf{v_j} = \mathbf{x}^*$. Then, by means of a linear interpolation, the control law

$$\mathbf{u} = \mathbf{k}(\mathbf{x}) = \sum_{\mathbf{j}} (\mathbf{K_j x} + \mathbf{k_j}) \prod_{i=1}^{n} w_{j_i}(x_i) \tag{31}$$

is obtained, where a notation similar to the parametric form for PBM/RFS was used. Note that local controllers were only chosen for each vertex in the state space, rather than the input-state space to avoid the controller from being an implicit function. For convenience, we will also denote (31) by

$$\mathbf{k}(\mathbf{x}) \in \underset{\mathbf{v_j}, \mathbf{x} \in R_\mathbf{k}}{\text{conv}} \{\mathbf{K_j x} + \mathbf{k_j}\}. \tag{32}$$

Starting from the dynamics (12), a partial matrix form

$$\dot{\mathbf{x}} = \sum_{\mathbf{j,q}} \dot{\mathbf{v}}_{\mathbf{j,q}} \cdot w_\mathbf{j}(\mathbf{x}) \cdot w_\mathbf{q}(\mathbf{u}) = \mathbf{f}_{\mathbf{j,q}}(\mathbf{x}, \mathbf{u}) \tag{33}$$

is derived, where for simplicity $\mathbf{f}_{\mathbf{j,q}}$ denotes the local dynamics being active in rectangle $R_{\mathbf{j,q}}$. Then, by means of a linearization

$$\mathbf{A}_{\mathbf{j,q}} = \frac{\partial}{\partial \mathbf{x}} \mathbf{f}_{\mathbf{j,q}}(\mathbf{x}, \mathbf{0}) \bigg|_{\mathbf{x}=\mathbf{v_j}}, \tag{34a}$$

$$\mathbf{a}_{\mathbf{j,q}} = \mathbf{f}_{\mathbf{j,q}}(\mathbf{v_j}, \mathbf{0}) - \mathbf{A}_{\mathbf{j,q}} \mathbf{v_j}, \tag{34b}$$

$$\mathbf{B}_{\mathbf{j,q}} = \frac{\partial}{\partial \mathbf{u}} \mathbf{f}_{\mathbf{j,q}}(\mathbf{v_j}, \mathbf{u}) \bigg|_{\mathbf{u}=\mathbf{0}} \tag{34c}$$

for each $\mathbf{v_j} \in R_\mathbf{k}$, (33) is restated as

$$\dot{\mathbf{x}} \in \underset{\mathbf{q}}{\text{conv}} \left\{ \underset{\mathbf{v_j}, \mathbf{x} \in R_\mathbf{k}}{\text{conv}} \{\mathbf{A}_{\mathbf{j,q}} \mathbf{x} + \mathbf{a}_{\mathbf{j,q}} + \mathbf{B}_{\mathbf{j,q}} \mathbf{u}\} \right\}, \tag{35}$$

where it is assumed that $\mathbf{a}_{\mathbf{j,q}} = \mathbf{0}$ for $\mathbf{v_j} \in \mathcal{R}_0$, i.e., the region of rectangles including the origin. Note that for $\mathbf{v}_{\mathbf{j,q}} = \mathbf{0}$ to be an equilibrium, this is also a necessary conditions.

By substituting (32) into (35), the dynamics of the closed-loop system

$$\begin{aligned}\dot{\mathbf{x}} &\in \operatorname*{conv}_{\mathbf{v_i},\mathbf{v_j},\mathbf{x}\in R_\mathbf{k};\mathbf{q}} \left\{\mathbf{A_{j,q}x} + \mathbf{a_{j,q}} + \mathbf{B_{j,q}}\left(\mathbf{K_i x} + \mathbf{k_i}\right)\right\} \\ &= \begin{cases} \operatorname*{conv}_{\mathbf{v_i},\mathbf{v_j},\mathbf{x}\in R_\mathbf{k};\mathbf{q}} \left\{\mathbf{A_{i,j,q}x}\right\}, & \mathbf{v_i},\mathbf{v_j} \in \mathcal{R}_0, \\ \operatorname*{conv}_{\mathbf{v_i},\mathbf{v_j},\mathbf{x}\in R_\mathbf{k};\mathbf{q}} \left\{\overline{\mathbf{A}}_\mathbf{i,j,q}\overline{\mathbf{x}}\right\}, & \mathbf{v_i},\mathbf{v_j} \notin \mathcal{R}_0, \end{cases} \end{aligned} \tag{36a}$$

are obtained as differential inclusion, with

$$\mathbf{A_{i,j,q}} = \mathbf{A_{j,q}} + \mathbf{B_{j,q}}\mathbf{K_i}, \qquad \mathbf{v_i},\mathbf{v_j} \in \mathcal{R}_0, \tag{36b}$$

$$\overline{\mathbf{A}}_\mathbf{i,j,q} = \begin{bmatrix} \mathbf{A_{j,q}} + \mathbf{B_{j,q}}\mathbf{K_i} & \mathbf{a_{j,q}} + \mathbf{k_i} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \qquad \mathbf{v_i},\mathbf{v_j} \notin \mathcal{R}_0. \tag{36c}$$

Then by means of a piecewise quadratic Lyapunov function (18),

$$\begin{aligned}\dot{V}(\mathbf{x}) &= \begin{cases} \operatorname{He}\left\{\mathbf{x}^T\mathbf{P_k}\dot{\mathbf{x}}\right\}, & R_\mathbf{k} \in \mathcal{R}_0, \\ \operatorname{He}\left\{\overline{\mathbf{x}}^T\overline{\mathbf{P}}_\mathbf{k}\begin{bmatrix}\dot{\mathbf{x}} \\ 1\end{bmatrix}\right\}, & R_\mathbf{k} \notin \mathcal{R}_0, \end{cases} \\ &\in \begin{cases} \mathbf{x}^T \operatorname*{conv}_{\mathbf{v_i},\mathbf{v_j},\mathbf{x}\in R_\mathbf{k};\mathbf{q}} \left\{\operatorname{He}\left\{\mathbf{P_k A_{i,j,q}}\right\}\right\}\mathbf{x}, & R_\mathbf{k} \in \mathcal{R}_0, \\ \overline{\mathbf{x}}^T \operatorname*{conv}_{\mathbf{v_i},\mathbf{v_j},\mathbf{x}\in R_\mathbf{k};\mathbf{q}} \left\{\operatorname{He}\left\{\overline{\mathbf{P}}_\mathbf{k}\overline{\mathbf{A}}_\mathbf{i,j,q}\right\}\right\}\overline{\mathbf{x}}, & R_\mathbf{k} \notin \mathcal{R}_0 \end{cases} \end{aligned} \tag{37}$$

follows from substituting (36a) into (21). Thus, $\dot{V}(\mathbf{x}) < 0$ is ensured, if

$$\operatorname{He}\left\{\mathbf{P_k A_{i,j,q}}\right\} \prec 0, \quad R_\mathbf{k} \in \mathcal{R}_0, \tag{38a}$$

$$\operatorname{He}\left\{\overline{\mathbf{P}}_\mathbf{k}\overline{\mathbf{A}}_\mathbf{i,j,q}\right\} \prec 0, \quad R_\mathbf{k} \notin \mathcal{R}_0 \tag{38b}$$

holds. In contrast to the gradient conditions (27b) and (27d) for the stability analysis, (37) can no longer be reduced to finitely many scalar inequalities at vertices. As a consequence, the S-procedure has to be utilized for (38) in order to take locality of the matrix inequalities into consideration. Thus, (38) holds if there exists symmetric matrices $\mathbf{S}_{\mathrm{a,k}}$ with non-negative components such that

$$\operatorname{He}\left\{\mathbf{P_k A_{i,j,q}}\right\} + \mathbf{E}_\mathbf{k}^T\mathbf{S}_{\mathrm{a,k}}\mathbf{E_k} \prec 0, \quad R_\mathbf{k} \in \mathcal{R}_0, \tag{39a}$$

$$\operatorname{He}\left\{\overline{\mathbf{P}}_\mathbf{k}\overline{\mathbf{A}}_\mathbf{i,j,q}\right\} + \overline{\mathbf{E}}_\mathbf{k}^T\mathbf{S}_{\mathrm{a,k}}\overline{\mathbf{E}}_\mathbf{k} \prec 0, \quad R_\mathbf{k} \notin \mathcal{R}_0. \tag{39b}$$

Akin to the conditions for stability analysis, the restriction of the region of attraction to the domain, i.e., $\mathcal{E} \subseteq \mathcal{R}$, is taken into consideration by means of (30). In order to maximize the size of the region of attraction, auxiliary ellipsoids $\varepsilon_i = \mathbf{x}^T \mathbf{M}_i \mathbf{x}$ for each sector $\mathcal{S}_i$ around $\mathbf{x}^*$ are inscribed into the region of attraction. With $\mathbf{n}_i^T$ being a normal vector pointing into the sector $\mathcal{S}_i$, the radius of each ellipse is maximized by $\min \mathbf{n}_i^T \mathbf{M}_i \mathbf{n}_i$. In addition, the constraint of auxiliary ellipses being contained in the region of attraction, i.e., $\cup_i \varepsilon_i \subseteq \mathcal{E}$, may be formulated as

$$\begin{cases} \mathbf{x}^T \mathbf{P}_\mathbf{k} \mathbf{x} \leq \mathbf{x}^T \mathbf{M}_i \mathbf{x}, & \mathbf{x} \in R_\mathbf{k} \in \mathcal{R}_0, \\ \overline{\mathbf{x}}^T \overline{\mathbf{P}}_\mathbf{k} \overline{\mathbf{x}} \leq \overline{\mathbf{x}}^T \begin{bmatrix} \mathbf{M}_i & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} \overline{\mathbf{x}}, & \mathbf{x} \in R_\mathbf{k} \notin \mathcal{R}_0 \end{cases} \tag{40}$$

for each $R_\mathbf{k} \cap \mathcal{S}_i$. Equivalently, by using the S-procedure, (40) can be written as LMI

$$\begin{cases} \mathbf{M}_i - \mathbf{P}_\mathbf{k} - \mathbf{E}_\mathbf{k}^T \mathbf{S}_{\mathrm{b},\mathbf{k}} \mathbf{E}_\mathbf{k} \succ 0, & R_\mathbf{k} \in \mathcal{R}_0, \\ \begin{bmatrix} \mathbf{M}_i & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} - \overline{\mathbf{P}}_\mathbf{k} - \overline{\mathbf{E}}_\mathbf{k}^T \mathbf{S}_{\mathrm{b},\mathbf{k}} \overline{\mathbf{E}}_\mathbf{k} \succ 0, & R_\mathbf{k} \notin \mathcal{R}_0. \end{cases} \tag{41}$$

Then, the controller synthesis problem for PBM and RFS is solved by means of the following theorem, summarizing the aforementioned conditions:

Theorem 1. *A PBM or RFS is piecewise quadratically stable, if there exist local controller matrices* $\mathbf{K_j}, \mathbf{k_j}$ *for each vertex* $\mathbf{v_j} \in \mathcal{R}$*, and symmetric matrices* $\mathbf{T}$*,* $\mathbf{S_k} > \mathbf{0}$*,* $\mathbf{S}_{\mathrm{a},\mathbf{k}} > \mathbf{0}$*,* $\mathbf{S}_{\mathrm{b},\mathbf{k}} > \mathbf{0}$*, such that*

$$\min \sum_i \mathbf{n}_i^T \mathbf{M}_i \mathbf{n}_i, \tag{42a}$$

$\forall\, R_\mathbf{k} \in \mathcal{R}_0$*:*

$$\mathbf{P}_\mathbf{k} - \mathbf{E}_\mathbf{k}^T \mathbf{S}_\mathbf{k} \mathbf{E}_\mathbf{k} \succ 0, \tag{42b}$$

$$\mathrm{He}\left\{\mathbf{P}_\mathbf{k} \mathbf{A}_{\mathbf{i},\mathbf{j},\mathbf{q}}\right\} + \mathbf{E}_\mathbf{k}^T \mathbf{S}_{\mathrm{a},\mathbf{k}} \mathbf{E}_\mathbf{k} \prec 0, \quad \mathbf{v_i}, \mathbf{v_j} \in R_\mathbf{k}, \tag{42c}$$

$$\begin{bmatrix} \mathbf{P}_\mathbf{k} & \mathbf{a}_i \\ \mathbf{a}_i^T & 1 \end{bmatrix} \succ 0, \quad R_\mathbf{k} \cap \partial\mathcal{R} \neq \emptyset, \tag{42d}$$

$$\mathbf{M}_i - \mathbf{P}_\mathbf{k} - \mathbf{E}_\mathbf{k}^T \mathbf{S}_{\mathrm{b},\mathbf{k}} \mathbf{E}_\mathbf{k} \succ 0, \tag{42e}$$

$\forall\, R_\mathbf{k} \notin \mathcal{R}_0$*:*

$$\overline{\mathbf{P}}_\mathbf{k} - \overline{\mathbf{E}}_\mathbf{k}^T \mathbf{S}_\mathbf{k} \overline{\mathbf{E}}_\mathbf{k} \succ 0, \tag{42f}$$

$$\mathrm{He}\left\{\overline{\mathbf{P}}_\mathbf{k} \overline{\mathbf{A}}_{\mathbf{i},\mathbf{j},\mathbf{q}}\right\} + \overline{\mathbf{E}}_\mathbf{k}^T \mathbf{S}_{\mathrm{a},\mathbf{k}} \overline{\mathbf{E}}_\mathbf{k} \prec 0, \quad \mathbf{v_i}, \mathbf{v_j} \in R_\mathbf{k} \tag{42g}$$

$$\begin{bmatrix} \overline{\mathbf{P}}_\mathbf{k} & \overline{\mathbf{a}}_i \\ \overline{\mathbf{a}}_i^T & 1 \end{bmatrix} \succ 0, \quad R_\mathbf{k} \cap \partial\mathcal{R} \neq \emptyset, \tag{42h}$$

$$\begin{bmatrix} \mathbf{M}_i & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} - \overline{\mathbf{P}}_\mathbf{k} - \overline{\mathbf{E}}_\mathbf{k}^T \mathbf{S}_{\mathrm{b},\mathbf{k}} \overline{\mathbf{E}}_\mathbf{k} \succ 0 \tag{42i}$$

holds for all $R_\mathbf{q} \in \mathcal{U}$. Then, the maximum region of attraction $\mathcal{E}$ is given by $V(\mathbf{x}) \leq 1$, and $\mathcal{E} \subseteq \mathcal{R}$.

If $\mathbf{K_j} = \mathbf{0}$ is chosen, then a controller is obtained, which is structurally equivalent to a Mamdani type fuzzy controller or look-up-table controller. Nevertheless, in general it is desirable to have more degrees of freedom by allowing non-zero $\mathbf{K_j}$, leading to an improved performance compared to pure static fuzzy controllers.

5 Implementational Aspects

Due to product terms in $\mathbf{P_k}$ and $\mathbf{K_j}$, $\mathbf{k_j}$, the optimization problem (42) contains bilinear matrix inequalities (BMI). Therefore, only suboptimal results can be obtained in contrast to the stability analysis. Because of the piecewise definition of the system, a linearizing transformation as in the case of linear systems with unique quadratic Lyapunov function appears to be impossible. Among the solution strategies proposed in the literature for solving BMI, we mention the path-following approach [17] linearizing the optimization problem and solving it in an iterative manner, as well as the rank minimization strategy [18], both being local solution algorithms. Global solution strategies have been proposed in [19] based on a branch-and-bound technique and in [20] based on the benders decomposition.

To limit the implementational burden, we make use of a V-K-iteration scheme as proposed in [21], which iteratively solves (42) by keeping either $\mathbf{P_j}$ or $\mathbf{K_j}, \mathbf{k_j}$ fixed while solving the LMI-problem for the remaining variables. Obviously, the solution thereby obtained relies heavily on the initial solution. Therefore, we propose to compute an initial controller heuristically akin to the initial solution proposed for the ILMI algorithm for T-S systems in [22]. The aim is to obtain for each vertex $\mathbf{v_j} \in \mathcal{R}$ a mean affine system

$$\dot{\mathbf{v}}_\mathbf{j} \approx \tilde{\mathbf{A}}_\mathbf{j}\mathbf{x} + \tilde{\mathbf{a}}_\mathbf{j} + \tilde{\mathbf{B}}_\mathbf{j}\mathbf{u}, \tag{43}$$

with $\tilde{\mathbf{A}}_\mathbf{j} = \frac{1}{N_q}\sum_\mathbf{q}\frac{1}{2^n}\sum_{\mathbf{k}:\mathbf{v_j}\in R_\mathbf{k}}\mathbf{A_{j,q}}$, $\tilde{\mathbf{a}}_\mathbf{j} = \frac{1}{N_q}\sum_\mathbf{q}\frac{1}{2^n}\sum_{\mathbf{k}:\mathbf{v_j}\in R_\mathbf{k}}\mathbf{a_{j,q}}$, $\tilde{\mathbf{B}}_\mathbf{j} = \frac{1}{N_q}\sum_\mathbf{q}\frac{1}{2^n}\sum_{\mathbf{k}:\mathbf{v_j}\in R_\mathbf{k}}\mathbf{B_{j,q}}$ where $\mathbf{A_{j,q}}, \mathbf{a_{j,q}}, \mathbf{B_{j,q}}$ are defined by (34a). Note that the necessity to average over linearizations around a vertex $\mathbf{v_j}$ for each of the 2^n adjacent rectangles arises, since an approximate gradient is to be obtained independently of a particular rectangle $R_\mathbf{k}$, and because the utilized triangular membership functions are not differentiable in $\mathbf{v_j}$.

For each local controller $\mathbf{K_j x} + \mathbf{k_j}$, the offset part is set according to

$$\mathbf{k_j} = -\tilde{\mathbf{B}}_\mathbf{j}^{+}\tilde{\mathbf{a}}_\mathbf{j} \tag{44}$$

with $\tilde{\mathbf{B}}_\mathbf{j}^{+} = \left(\tilde{\mathbf{B}}_\mathbf{j}^{T}\tilde{\mathbf{B}}_\mathbf{j}\right)^{-1}\tilde{\mathbf{B}}_\mathbf{j}$ denoting the Moore–Penrose pseudoinverse of $\tilde{\mathbf{B}}_\mathbf{j}$. For the remaining linear subsystem, $\mathbf{K_j}$ is then determined by means of pole placement. With these initial solutions for $\mathbf{K_j}$ and $\mathbf{k_j}$, (42) is solved for $\mathbf{P_j}$, then by fixing $\mathbf{P_j}$, new local controllers are computed. The procedure is repeated until no further improvements can be made.

As for the stability analysis, relaxing the S-procedure may lead to a reduced number of LMI variables. For non-symmetric regions, appropriate modifications of (42d), (42h) akin to (30) will prevent conservative solutions with regard to the stability region.

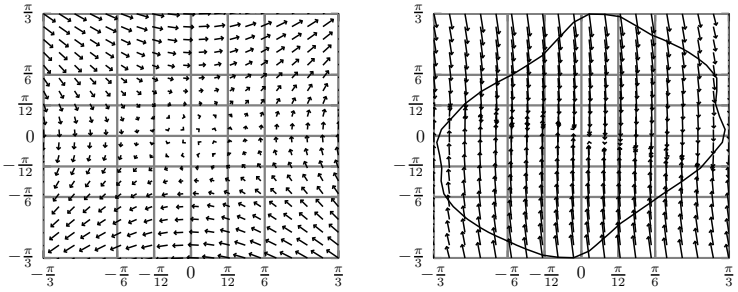


Figure 1: Phase portraits of (a) open-loop inverted pendulum modeled as PBM/RFS and (b) controlled inverted pendulum

6 Numerical Example

The controller synthesis algorithm outlined in Theorem 1 is applied to the well-known dynamics of a pendulum mounted on a cart

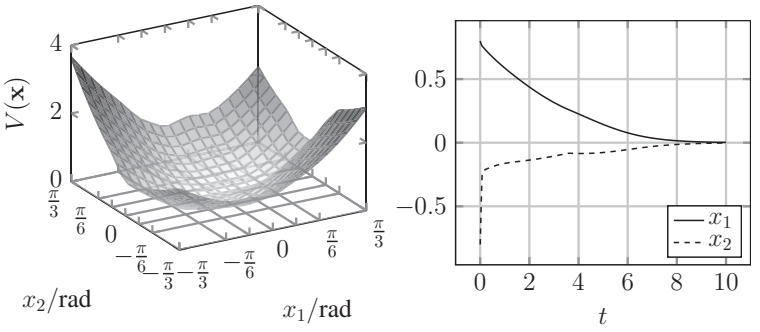
$$\dot{x}_1 = x_2, \tag{45a}$$

$$\dot{x}_2 = -\sin(x_1 + \pi) - \cos(x_1 + \pi)u, \tag{45b}$$

whereas the dynamics of the cart is neglected.

From these nonlinear dynamics, a PBM/RFS is derived by sampling (45) at vertices $(\mathbf{v_j}, \mathbf{v_q})$, whereas the vertices in each dimension are chosen to be $v_{j_1} = v_{j_2} = \{0, \pm\frac{\pi}{12}, \pm\frac{\pi}{6}, \pm\frac{\pi}{3}\}$ and $v_q = \{-10, 0, 10\}$. Since the system has only one input, it may be represented equivalently as PBM and RFS. Its open-loop dynamics are depicted in Fig. 1a, revealing the origin being an anti-stable equilibrium that is to be stabilized.

In order to solve the bilinear matrix inequality optimization problem (42), initial local controllers are first computed for the approximating affine local systems (43). The affine part of these initial controllers is given by (44), whereas for the remaining linear subsystems, the controller matrices $\mathbf{K_j}$ are chosen such that the poles of closed-loop local systems are at $s_i =$



(a) (b)

Figure 2: (a) Lyapunov function for controlled inverted pendulum and (b) state development of controlled inverted pendulum for initial condition of $\mathbf{x}_0 = [0.8,\ -0.8]^T$.

$\{-4, -5\}$. With this initial solution, (42) is then solved iteratively, resulting in a stable controlled system, whose phase portrait is documented in Fig. 1b. Therein, the region of attraction is shown as well, being fully included in the system domain. The final Lyapunov function is also shown in Fig. 2a.

By applying the resulting controller to the ground truth dynamics (45) with initial conditions $\mathbf{x}_0 = [0.8,\ -0.8]^T$, the development of the states as shown in Fig. 2b was obtained during simulation. As expected, the closed-loop dynamics reveal an asymptotically stable behavior.

7 Conclusion

A control method was proposed, which is applicable for PBM and RFS, leading to a provable stabilization of equilibria of the system. Since the concept of parallel distributed compensation was used, the dynamics of the controlled system will again be smooth, i.e., no chattering will occur, even though the system dynamics are defined piecewise. The stability analysis

was based on piecewise quadratic Lyapunov functions without introducing any other assumptions. Thus, with increasing precision of the PBM/RFS, the approximation of the Lyapunov function becomes increasingly precise as well. The piecewise definition on the other hand comes at the cost of the use of the S-Procedure, introducing a slight degree of conservatism, which renders the stability conditions only sufficient. Nevertheless, it was shown that the stability conditions can be extended towards controller synthesis in terms of bilinear matrix inequalities. To solve this non-convex problem in an iterative manner, a way for finding good initial solutions was proposed.

A thorough comparison with existing control concepts for general smooth nonlinear systems remains a topic for future research. Preliminary results indicate, that the framework of PBM/RFS as finite element approach is able to clearly outperform existing methods in nonlinear control, since no assumptions have to be made about the type of the Lyapunov function and dynamic function to be approximated. On the other hand, compared to other universal approximators, it is still possible to prove system properties such as stability.

Literatur

[1] Takagi, T.; Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man And Cybernetics* 15 (1985) 1, S. 116–132.

[2] Sugeno, M.: On stability of fuzzy systems expressed by fuzzy rules with singleton consequents. *IEEE Transactions on Fuzzy Systems* 7 (1999) 2, S. 201–224.

[3] Sugeno, M.; Taniguchi, T.: On improvement of stability conditions for continuous Mamdani-like fuzzy systems. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics* 34 (2004) 1, S. 120–131.

[4] Adamy, J.; Kempf, R.: Regularity and chaos in recurrent fuzzy systems. *Fuzzy Sets and Systems* 140 (2003) 2, S. 259–284.

[5] Taniguchi, T.; Sugeno, M.: Robust stabilization of nonlinear systems modeled with piecewise bilinear systems based on feedback linearization. In: *Advances on Computational Intelligence*, S. 111–120. Springer. 2012.

[6] Eciolaza, L.; Sugeno, M.: On-line design of LUT controllers based on desired closed loop plant: Vertex Placement Principle. In: *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, S. 1–8. IEEE. 2012.

[7] Eciolaza, L.; Taniguchi, T.; Sugeno, M.; Filev, D.; Wang, Y.: Piecewise Bilinear models for feedback error learning: On-line feedforward controller design. In: *IEEE International Conference on Fuzzy Systems*, S. 1–8. 2013.

[8] Gering, S.; Adamy, J.: Fuzzy control of continuous-time recurrent fuzzy systems. *Fuzzy Sets and Systems* (2014).

[9] Gering, S.; Schwung, A.; Gußner, T.; Adamy, J.: Sum of Squares Approaches for Control of Continuous-Time Recurrent Fuzzy Systems. In: *Proc. of the 21st IEEE Mediterranean Conference on Control and Automation*, S. 271–277. Chania, Greece. 2013.

[10] Gering, S.; Adamy, J.: Robust Stabilization of Recurrent Fuzzy Systems via Switching Control. In: *Proceedings of the IEEE Conference on Fuzzy Systems*. Beijing, China. 2014.

[11] Johansson, M.; Rantzer, A.; Arzen, K.-E.: Piecewise quadratic stability of fuzzy systems. *IEEE Transactions on Fuzzy Systems* 7 (1999) 6, S. 713–722.

[12] Gering, S.; Eciolaza, L.; Adamy, J.; Sugeno, M.: A Piecewise Approximation Approach to nonlinear Systems: Stability and Region of Attraction. *Transactions on Fuzzy Systems (Submited)* (2014).

[13] Adamy, J.; Flemming, A.: Equilibria of continuous-time recurrent fuzzy systems. *Fuzzy Sets and Systems* 157 (2006) 22, S. 2913–2933.

[14] Johansson, M.; Rantzer, A.: Computation of piecewise quadratic Lyapunov functions for hybrid systems. *IEEE Transactions on Automatic Control* 43 (1998) 4, S. 555–559.

[15] Boyd, S.; Ghaoui, L. E.; Feron, E.; Balakrishnan, V.: *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics Philadelphia. 1994.

[16] Diepold, K.; Pieczona, S.: Abschätzung des Einzugsbereiches für T-S Systeme: Beachtung des Gültigkeitssektors. In: *Proceedings 23. Workshop Computational Intelligence*. Dortmund. 2013.

[17] Hassibi, A.; How, J.; Boyd, S.: A path-following method for solving BMI problems in control. In: *Proceedings of the American Control Conference*, Bd. 2, S. 1385–1389. San Diego, California, USA. 1999.

[18] Ibaraki, S.; Tomizuka, M.: Rank minimization approach for solving BMI problems with random search. In: *Proceedings of the American Control Conference*, Bd. 3, S. 1870–1875. 2001.

[19] Goh, K.-C.; Safonov, M.; Papavassilopoulos, G.: A global optimization approach for the BMI problem. In: *Proceedings of the 33rd IEEE Conference on Decision and Control*, Bd. 3, S. 2009 –2014. 1994.

[20] Beran, E.; Vandenberghe, L.; Boyd, S.: A Global BMI Algorithm Based on the Generalized Benders Decomposition. 1997.

[21] El Ghaoui, L.; Balakrishnan, V.: Synthesis of fixed-structure controllers via numerical optimization. In: *Proceedings of the 33rd Conference on Decision and Control*, Bd. 3, S. 2678–2683. 1994.

[22] Kim, E.; Kim, S.: Stability analysis and synthesis for an affine fuzzy control system via LMI and ILMI: continuous case. *IEEE Transactions on Fuzzy Systems* 10 (2002) 3, S. 391–400.

Identification of Multi-Scale Motifs

Sahar Torkamani, Volker Lohweg

inIT – Institute Industrial IT
Ostwestfalen-Lippe University of Applied Sciences
Langenbruch 6, D-32657 Lemgo, Germany
Email: {sahar.torkamani, volker.lohweg}@hs-owl.de

Abstract

With the daily progress of industry, the tasks performed by electrical and mechanical machines are growing increasingly. Therefore, accurate machine diagnosis and condition monitoring for complex automated production systems become an important task. With recent advances in sensor and information fusion techniques, significant progresses have been made in this area by gathering useful information from the environment. Such knowledge is practical in order to predict the system behaviour, find patterns, anomalies, or *motifs*. The focus of this work is on time-series signals motif discovery. This work tackles the drawbacks of existing methods namely inability of finding shifted and multi-scale motifs of different size. The proposed approach is based on the invariant wavelet transform which is able to overcome the limitations of the existing algorithms.

1 Introduction

Machine diagnosis and condition monitoring for complex automated production systems has gained huge interest during recent years. To achieve the goal of the production system, namely to fabricate a high quality product, all parts of the system should work together properly. The behaviour diagnosis of such systems is a complex task, depending on many effects. For instance, collecting information from various sources as completely as possible without any artefacts, or discovering defective parts and machines, etc. [1–3].
Sensor and information fusion plays a major role in behaviour diagnosis of such systems. The typical character of multi-sensor systems is capturing the state of the environment and generating fused information using the obtained data and information fusion methods [1], [3]. Moreover, signals that are originated from physical sensors usually contain multiple events

or patterns that occur at different time scales. Such knowledge can be applied to predict the behaviour of a system or find patterns, anomalies, or *motifs*. The term *motif* was first triggered by Patel et al. [4], and means finding frequent unknown patterns in a signal without any prior information about their locations or shapes. The time-series motif of length m of a signal $T = (t_1, ..., t_n)$ is a pair of subsequences $(S_{i,m}, S_{j,m})$ of signal T for $1 \leq i < i + m \leq j \leq n - m + 1$ that $distance(S_i, S_j)$ is the smallest among all possible pairs [5].
Motifs can provide valuable insights about the problem under investigation to the user or expert. These unknown patterns can be used in tasks such as rule-discovery, and summarisation by providing useful information or in clustering and classification as prototypes for each class [6].
In spite of extensive research in time-series motif discovery, there is a lack of a method which is able to detect shifted and multi-scale motifs of different size. Such motifs have been stretched/squeezed or simply shifted in time or amplitude domain. Moreover, they can be rotated or have different phase. To able to handle such variances, this paper proposes an approach based on an invariant mapping method which is capable of handling stationary as well as non-stationary signals in the first step, followed by feature extraction in the second step and finally ended by similarity measures. Among many invariance mapping methods, explained in section 3, Wavelet Transforms are utilized in this work which provide time-scale resolution. Wavelet Transforms have been shown to be a valuable tool in many signal processing applications such dimensionality reduction, noise filtering and signal classification [7], [8], however to our knowledge these methods were not used in the time-series motif discovery task.
This article is organized as following: Section 2 reviews the related work regarding motif discovery. A brief background of invariance methods, notations and necessary definitions are given in section 3. Then the proposed approach is introduced in section 4. Following, experimental results based on real world data are explained in section 5. Finally, section 6 contains conclusions and directions for future work.

2 Related Work

As stated in [4], the definition of time-series motif is based on a user-defined range parameter R and a motif of length m. Two subsequences of length m match if their similarity determined by similarity measure methods is less than R.

Discovery of motifs in time-series usually faces one or more of the following problems: quadratic time-complexity, definition of static length of motifs, data streaming and finally, multi-scale motifs, depicted in Fig. 1.

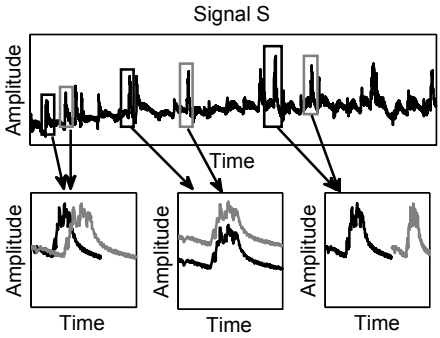


Fig. 1: Multi-Scale motifs

Many attempts have been undertaken to handle these problems, for example Mueen et. al. [9] applies the symmetry and triangular inequality properties of the Euclidean distance to improve time complexity. Lin and Li use a grammar-based algorithm to find approximate motifs in medical data [10], by continually generating a grammatical rule for each subsequence. In this algorithm, the subsequence with the most repeated rule is chosen as a motif. The advantage of this method is that there is no need to define motif lengths, however there are other parameters that must be defined. Additionally, this algorithm can handle data streaming.
Defining an optimal motif's length is one of the main problems of typical motif discovery algorithms which usually needs expert knowledge. Nunthanid et. al. [11] present Variable-Length Motif Discovery (VLMD), a non parametric method, which applies a sliding window of different lengths several times to find motifs from shortest to longest length, and later classifies the similar motifs into a group. Extreme time-execution especially in the case of huge data is the drawback of this method [12].
Many motif discovery algorithms find motifs only at a single resolution. Castro and Azevedo use indexable Symbolic Aggregate approXimation (iSAX) [13] to find motifs in different resolutions [14]. Recently, Vespier et al. [15] introduced a method to find motifs in multiple temporal scales by combining the scale-space theory and the Minimum Description Length principle (MDL) [16]. The scale-space method provides a complete description of the signal in different scales in terms of Gaussian smoothing [15]. Later, MDL tries to find the proper scales which can capture the ef-

fect of transient events in the signal, and represent those scales based on their complexity. Finally, each scale is transformed by a symbolic representation method, and motifs are identified by searching for repeating subsequences.
Despite the vast research in this area, the existing algorithms suffer from two drawbacks, explained in section 4 where the novel approach to tackle these problems is proposed as well.

3 Background and Notations

In this section, the basic theoretical concepts, some notations and useful definitions used in this paper are briefly recalled.
Time-series: An ordered set of n real-valued variables over time,
$T = (t_1, t_2, ..., t_n)$ with $t_i \in \mathbf{R}$, is called a time-series [14]. Time-series such as Space Shuttle Marotta Valve [17], or EEG signals of a patient over 10 years can be very long. In such data, the focus is more on the local properties rather than the global. Therefore, subsections of the time-series which are called subsequences are considered.
Time-Series Subsequence: If T is a time-series of length n, then a time-series subsequence $S = (s_i, ..., s_{i+k-1})$ is sampled at $k \leq n$ contiguous positions of T, such that $1 \leq i \leq n - k + 1$ [14]. The most common method to obtain such subsequences from a time-series is using a sliding window.
Sliding Window: Given a time-series T, and a subsequence or window W of length w, all possible subsequences S_p for $1 \leq p \leq n - w + 1$, are extracted by sliding W across T [18]. Most of time-series signals vary in time, amplitude, scale or phase. To compare such signals or find useful information from them, invariance techniques or methods should be applied.
Invariance methods: Various means and techniques are introduced in literature to tackle the problems of amplitude, time, phase or scale-variance of time-series signals. For example, two signals measured on different scales, say meter and feet, cannot be well matched even if they have similar shapes, since they have different amplitude scale [19]. *Amplitude invariance* techniques, such as z-normalization also known as "normalization to zero mean and unit of energy", partly overcome such a problem. Furthermore, time-series signals can have local misalignment. For instance, in order to compare and identify similar contents, small fluctuation of the tempo of the speakers should be allowed in recorded speech signals [20]. Such misalignments can be detected and measured by *warping invariance* methods such as Dynamic Time Warping (DTW) [21]. In case of periodic

time-series, *phase invariance* [19] is an important issue. To achieve phase invariance between two time-series, one time-series should be fixed and all circular shifts of the other time-series should be tested [22].
In some applications where a small subsequence of a time-series may be missing, *occlusion invariance* should be considered by selectively declining to match subsequences of a time-series [22].
As stated earlier, multiple patterns of different scales usually happen in time-series signals gathered from physical sensors. *Scale invariance* techniques like Uniform Scaling [20] or Wavelet Transforms [23] can be used to achieve scale invariance. However, most of real life problems need multiple invariances where most of the mentioned invariances can be achieved. Among many invariant methods, such as stated techniques, multi-scale histograms [24], or Short-Time Fourier Transform [25], this work focuses on the *Wavelet Transforms*. Wavelet Transforms are powerful tools used in different applications such as image processing, data compression, speech recognition, etc. during the last decades. They provide a flexible time-frequency window, which shrinks in observation of high frequency phenomena and widens in the case of low frequency behaviour. This property makes them suitable to find multi-scale patterns. Many different Wavelet Transforms are introduced in literature [23, 26, 27]. This work focuses on the *Dual Tree-Complex Wavelet Transform (DT-CWT)* [28]. The properties of the DT-CWT are multi-resolution, simple computation, perfect reconstruction, and efficient implementation. Moreover, it is shift, scale and rotation-invariant. The upcoming sections provide comprehensive information on the DT-CWT and the proposed approach.

4 Proposed Approach

In this section the proposed algorithm is described which is able to determine multi-scale motifs of variable lengths. As stated earlier, the existing methods of time-series motif discovery have mainly two drawbacks. First, inability of finding multi-scales motifs where the squeezed/stretched motifs are shifted in time/amplitude. Second, since the existing algorithms are only able to find motifs of similar lengths, lack of comparing and finding motifs of variable lengths is visible in these methods.
The proposal for the first problem is by means of an invariant method, which is able to achieve multiple invariances mentioned earlier. The aforementioned techniques have some limitations or can trivially achieve invariance mapping. For example, the z-normalization is not suitable if a constant signal over most of the time domain spans with minor noise at short

intervals. Or in uniform scaling, as the factor of scaling is not known in advance, all possibilities within a given range should be tested. Moreover, these methods consider only one type of the invariances, and therefore a multiple invariance mapping that consider most of the mentioned invariances is needed.
Such an invariant mapping applied in our proposed approach is the Dual Tree-Complex Wavelet Transform (DT-CWT) [28] which provides a scale and shift invariance transformation. The DT-CWT has been applied in many image processing applications [29–31], by examining a signal in different frequency scales.
To deal with the second problem, a brute force algorithm is proposed where all possible pairs of subsequences of all available lengths are obtained by sliding windows of various sizes. Moreover, in this procedure there is no need to define the motif's length in advance.
The proposed approach, in the first step segments the time-series into various subsequences of variable lengths. To obtain practical information from extracted subsequences, and to find multi-scale motifs, the DT-CWT is applied, which is shift and rotation-invariant and provides flexible time-scale resolution suitable for non-stationary time-series signals. Additionally, the DT-CWT has multi-resolution property that provides a comprehensive analysis of a signal into different frequency scales. To detect the best scale which provides the most applicable information, some criterion should be defined, explained in the following. After segmenting the time-series into subsequences of variable lengths and transforming them using the DT-CWT, feature extraction is applied in order to find multi-scale motifs of variable lengths. Feature extraction is a common method to compare patterns of various length and size, used in many pattern recognition applications. The applied features in this approach are stated in section 4.2.
Finally, to find motifs and their similarity/dissimilarity, distance similarity measures are used. This section starts by briefly explaining the DT-CWT, follows by feature extraction description and ends by explaining the applied similarity measures.

4.1 Dual Tree-Complex Wavelet Transform

Information regarding the Dual Tree-Complex Wavelet Transform (DT-CWT) [28] is provided in this section. The name dual tree comes from the fact that, the structure of the DT-CWT is based on two parallel Discrete Wavelet Transform (DWT) filterbank trees (cf. Fig. 2 (a)).
As known, the DWT is shift variant, since it uses downsampling in each

scale. Having two fully decimated trees equals to eliminating the down-sampling by 2, results in doubling the sampling rate, leads overcoming the shortcoming of the DWT by producing a shift invariance transform.

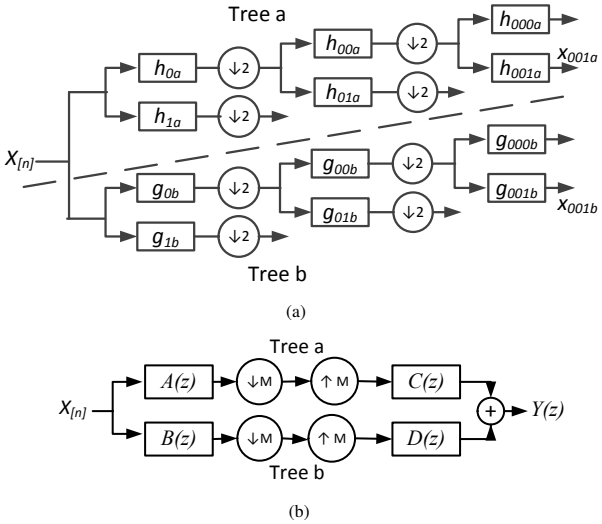


Fig. 2: (a) Dual Tree-Complex Wavelet Transform [28] (b) Analysis and Synthesis block of DT-CWT [28]

The DT-CWT decomposes a signal into real and imaginary parts, using an analytic representation of the signal. This representation provides non-negative frequency in the transform domain. As presented in Fig. 2 (a), "Tree a" represents the real part and "Tree b" provides the imaginary part. In the real "Tree a", h_{0a} and h_{1a} are low and high-pass filters respectively, and as well in the imaginary "Tree b", g_{0b} and g_{1b} are low and high-pass filters. Each tree uses a different set of filters. These two sets of filters are jointly designed so the transformation is analytic.
The transformation of DT-CWT is reconstructable. The reconstruction process is called "synthesis", where the signals at every level are upsampled and passed through the synthesis filters h'_{1a} and h'_{0a} in "Tree a", and g'_{1b} and g'_{0b} in "Tree b".
The wavelet functions in both trees are denoted by $\psi_a(t)$ and $\psi_b(t)$, which provide an analytic complex wavelet $\psi(t) = \psi_a(t) + j\ \psi_b(t)$ [28]. In this analytic representation, $\psi_b(t)$ is the Hilbert transform of $\psi_a(t)$ given by $\mathcal{H}\{\psi_a(t)\}$. This means $\psi_b(t)$ is 90° phase-shifted of the $\psi_a(t)$, and the Fourier transform of the complex wavelet function has no energy in the

negative frequency domain. The DT-CWT wavelet and its frequency spectrum is represented in Fig. 3.

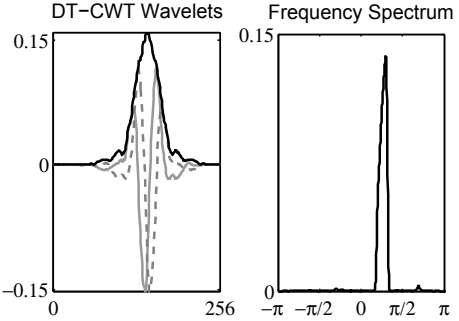


Fig. 3: (a) DT-CWT complex wavelet (black), Real part (grey), Imaginary part(dashed grey); (b) DT-CWT complex wavelet frequency spectrum

If $\psi(t) = \psi_a(t) + j\ \psi_b(t)$ donates the complex wavelet function and $\varphi(t) = \varphi_a(t) + j\ \varphi_b(t)$ gives the complex scaling function, then a signal $x(t)$ can be shown in terms of complex wavelets and scaling functions in DT-CWT via [28]

$$x(t) = \sum_{n=-\infty}^{\infty} c(n)\varphi(t-n) + \sum_{j=0}^{\infty}\sum_{n=-\infty}^{\infty} d(j,n)2^{j/2}\psi(2^j t - n), \qquad (1)$$

where the complex scaling coefficients $c(n)$ and wavelet coefficients $d(j,n)$ are obtained by the inner products

$$c(n) = \int_{-\infty}^{\infty} x(t)\varphi(t-n)dt, \qquad (2)$$

$$d(j,n) = 2^{j/2}\int_{-\infty}^{\infty} x(t)\psi(2^j t - n)dt. \qquad (3)$$

Additionally, the complex wavelet coefficients $d(j,n)$ are computed via $d(j,n) = d_a(j,n) + j\ d_b(j,n)$. The complex scaling coefficients $c(n)$ are determined similarly.

It should be noted that, although the DT-CWT is computed via two real DWTs, it requires a special design for the applied filters h and g. As stated, each DWT tree is designed to use a different filter set. To have wavelets that form an analytic representation, the low-pass filters in "Tree a" should have a half sample delay of low-pass filters in "Tree b" $g_b(n) = h_a(n - \frac{1}{2})$ [28]. However, the filters in the first level of the DT-CWT are different

from the rest of the levels. In the first stage, the filters have one-sample delay $g_b(n) = h_a(n-1)$ [28]. Therefore, at the first scale/level of the filter bank of DT-CWT, a unique low-pass filter pair is used which produces one pair of analytic wavelets at that scale, and subsequently, another distinct pair of low-pass filters is employed for the rest of the scales to obtain analytic wavelets in other scales.

To examine the shift-invariant property of the DT-CWT, assume retaining the coefficients of wavelet or scaling functions from only one level of the dual tree. For example, the wavelet coefficients from the third level x_{001a} and x_{001b} are retained and all other coefficients are set to zero. If the reconstructed signal from these coefficients is free of aliasing, then the transform for this level (level 3) is shift invariant [32]. Shown in Fig. 2 (b), the simplified analysis and synthesis parts of the DT-CWT are presented where coefficients of just one type and level are retained. In the example above, $M = 2^m = 8$ and $A(z) = H_{0a}(z)\ H_{00a}(z^2)H_{001a}(z^4)$, where $H_{0a}(z)$ and $H_{00a}(z^2)$ are the z-transformation of the low-pass filters in the first and second levels and $H_{001a}(z^4)$ is the z-transformation of the high-pass filter in the third level of "Tree-a". The transfer function $B(z)$, and the inverse functions $C(z)$ and $D(z)$ are defined in similar way [33]. Based on these definitions, Fig. 2 can be presented by [33]

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} X(W^k z)[A(W^k z)C(z) + B(W^k z)D(z)]. \qquad (4)$$

where $W = e^{j2\pi/M}$. In order to have shift invariance property, $B(W^k z)D(z)$ should cancel $A(W^k z)C(z)$ for $k \neq 0$. This design eliminates the overlap of the pass bands of the filters $C(z)$ or $D(z)$ with those of the shifted filters $A(W^k z)$ or $B(W^k z)$. To be able to cancel out the mentioned terms, two different strategies for low and high-pass filters should be considered [28]. For the low-pass filters, the filters designed such that [32]

$$B(z) = z^{\pm M/2}A(z) \ \text{ and } \ D(z) = z^{\mp M/2}C(z), \qquad (5)$$

so that $(-1)A(W^k z)C(z) = B(W^k z)D(z)$ for odd values of k. This reduces the overlap of the synthesis filters with the frequency-shifted analysis filters. For even values of k the overlap frequency is small since the frequency shift e. g. in the case of $k = 2$ is twice as great as when $k = 1$. In the case of high-pass filters, two prototype complex filters $P(z)$ and $Q(z)$, each having single passband $f_s/2M \rightarrow f_s/M$ and zero gain at all negative frequencies (f_s is the sampling frequency), are considered such

that [32]:

$$\begin{aligned} A(z) &= \Re[2P(z)] = P(z) + P^*(z), \\ B(z) &= \Im[2P(z)] = -j[P(z) - P^*(z)], \\ C(z) &= \Re[2Q(z)] = Q(z) + Q^*(z), \\ D(z) &= \Im[-2Q(z)] = j[Q(z) - Q^*(z)]. \end{aligned} \tag{6}$$

The conjugation form of $P(z)$ is given by $P^*(z) = \sum_r p_r^* z^{-r}$ which produces negative frequency pass bands. Including these filters in Eq. 4 results in [32]:

$$A(W^k z)C(z) + B(W^k z)D(z) = 2P(W^k z)Q(z) + 2P^*(W^k z)Q^*(z). \tag{7}$$

The design of the high-pass filters shows that the positive frequency of the complex filter $Q(z)$ does not overlap with shifted versions of the similar filter $P(z)$, results in a shift-invariant transformation.
Similar to other wavelet transforms, the DT-CWT decomposes a signal into different frequency scales using the filter bank structure. According to Parseval's theorem, energy of a signal after an orthonormal transformation is preserved. The DT-CWT filters are real, orthonormal, with length of 14 and 2 vanishing moments which are obtained using the design algorithm in [32]. Consequently, this transformation conserves energy. So, the energy of the signal is equal to the energy of the scaling and wavelet coefficients in each scale [28],

$$\sum_{j,n} \left(|c(n)|^2 + |d(j,n)|^2 \right) = \sum_n |x(n)|^2. \tag{8}$$

As explained earlier, wavelet transforms have an ability to analyse signals at different frequency scales. However, by increasing decomposition scales, the execution time increases considerably in large data sets. Therefore, the question is which scale provides better and useful information. The next section explains the criterion needed to select the suitable scale.

4.1.1 Information Content and Scale Selection Criterion

In most of the signal and image processing applications working with wavelet transforms, it is not necessary to consider all the frequency scales which the applied signal or image is decomposed into. Instead it is more common to chose the scale with the best spatial frequency solution and the largest information content [34]. Many methods such as [35], [36] and

[37] used energy of each frequency sub-band for this purpose. However, in the Dual Tree-Complex Wavelet Transforms the frequency bands in each scale does not have the same length. This makes the direct use of energy improper for choosing the best scale. For this reason, the energy density of the wavelet coefficients in each scale is computed. Based on this criterion, the scale with the highest amount of energy density is determined. After picking out the most appropriate scale, features are extracted from the scaling and wavelet coefficients of the nominated scale.

4.2 Feature Extraction

Features are individual measurable attributes which represent the data. In this work, statistical moments of wavelet coefficients found to be advantageous. These features are: mean value, variance, skewness and kurtosis. Applying such features are proved to be useful in most of the signal and image processing applications [8], [38], and [39]. In addition, energy of the wavelet coefficients is considered as another feature value. Since as the DT-CWT conserves energy even when the signal is shifted in time and amplitude. Moreover, extreme values of the scaling coefficients are included as feature values. Considering mirrored signals, the four mentioned statistical features and energy are similar in such signals. To be able to distinguish between this type of signals the order of the extreme values, local maximum/minimum, of the scaling coefficients are computed as well. For example, Fig. 4 (a) and (b) are graphical representations of two mirrored signal $S1$ and $S2$. Their scaling coefficients after low-pass filtering in scale

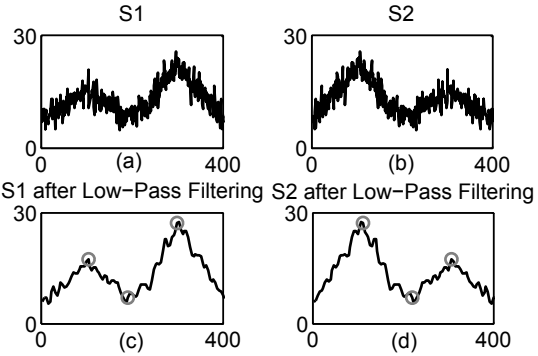


Fig. 4: (a), (b) Two mirrored signals; (c) local extreme of $S1$ equals [15,10,28], (d) local extreme of $S2$ equals [28,10,15]

two are presented in Fig. 4 (c) and (d). The grey circles in this picture depict the local maximum and minimum in two signals. Since their statistical features and energy are equal, the order of the local extreme, in this case $[15, 10, 28]$ for $S1$ and $[28, 10, 15]$ for $S2$, are used as features to differentiate these two signals. Moreover, Linear Discriminant Analysis (LDA) [40] is applied to determine the proficiency of the feature values. LDA is a linear classifier which classifies data into groups based on their feature values. Additionally, it determines which set of features can provide better classification results.

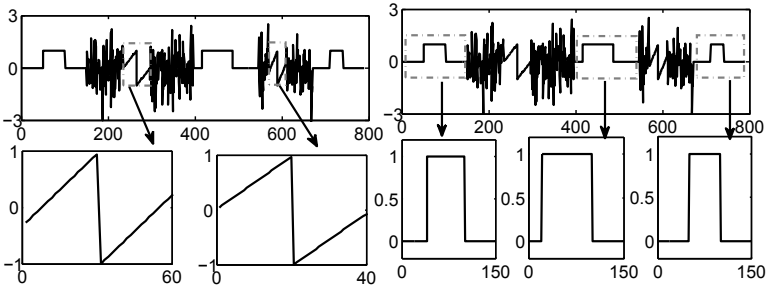
4.3 Similarity Measures

According to the definition of motifs, time-series motif is a pair of subsequences (S_i, S_j) of a time-series T that is the most similar. Most of the motif discovery algorithms use distance similarity measures [6]. Such a similarity measure $D(T_1, T_2)$, is a function taking two time-series T_1 and T_2 as inputs and returning the distance d between these series. This distance should be non-negative. Moreover, if this measure satisfies the additional symmetry property $D(T_1, T_2) = D(T_2, T_1)$ and sub-additivity $D(T_1, T_2) \leq D(T_1, U) + D(U, T_2)$, the distance considered as metric, such as Euclidean distance [6].
Most common similarity measures in time-series motif discovery methods are: Euclidean distance [41], Dynamic Time Warping [21], and Canberra distance [42]. Euclidean and Canberra distance have linear computational time. Moreover, since these measures are metric, the sub-additivity property helps to speed up the search of similar time-series motifs in the data set. Additionally, [43] showed that the Euclidean distance is surprisingly competitive with other more complex approaches, especially regarding large data sets. However, Euclidean distance suffers of being sensible to outliers and misalignments. On the other hand, Dynamic Time Warping is able to match various sections of a time-series by warping of the time axis. The proper alignment is obtained by the shortest warping path in a distance matrix. Time-complexity of this method is $O(n^2)$, although several measurements are introduced in literature to advance the computation [21].

5 Experimental Results

This section represents the actual results and findings. The proposed approach is tested by various data, however here the results of two data sets,

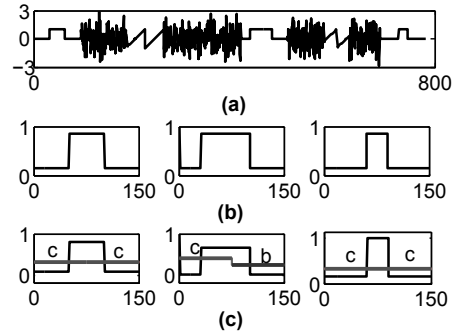
namely “Planted Motif” [9] and “AutoSense” [44], are explained subsequently.
Moreover, five common used motif discovery algorithms namely Mr. Motif [14], Smart Brute Force [9], Disk Aware Motif Enumeration (DAME) [5], Mueen-Keogh (MK) [45], and the grammar-based method [10] are applied as benchmarks to evaluate the novel approach. The test cases are tested by mentioned motif discovery algorithms to see if multi-scale motifs can be found by those algorithms.
The “Planted Motif” data is designed having two sets of patterns: a sawtooth and a square wave. Copies of these patterns with different scales are inserted into a sequence, depicted in Fig. 5. The novel approach can find all the inserted motifs, presented in Fig. 5(a), (b). Applying other



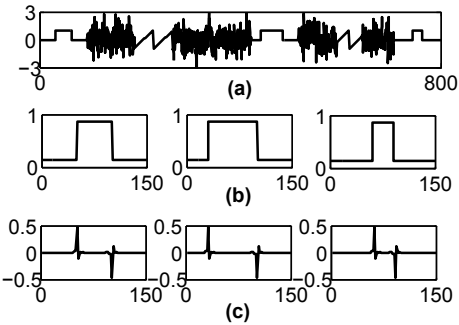
(a) (b)

Fig. 5: “Planted Motif” data set and the found motifs

algorithms as benchmark showed that these algorithms are not able to find motifs presented in Fig. 5(a), since these motifs have different length. On the other hand, Motifs presented in Fig. 5(b) found by Mr. Motif [14] and the grammar-based method [10]. Whereby, Smart Brute Force [9], Disk Aware Motif Enumeration (DAME) [5] and MK [45] were able to find small subsequences of these motifs.
The main reason for this is that most of these algorithms are based on the SAX [13] representation method. Fig. 6a, is a general graphical presentation of these methods, where first a time-series is segmented into subsequences of similar lengths, Fig. 6a(b). Next, each segment is presented by a word using the SAX [13] method. For example, in Fig. 6a(c) subsequences are presented by words *cc*, *cb*, and *cc*. These words in the next step are compared to each other to find similar subsequences.
On the other hand, Fig. 6b presents the first steps of the novel propose, where the segmented subsequences first transformed using the DT-CWT.



(a) First steps of Mr. Motif algorithm



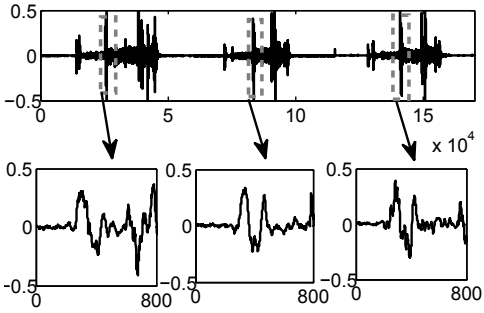
(b) First steps of the proposed aproach

Fig. 6: (a) First steps of Mr. Motif algorithm, (b) First steps of the proposed aproach

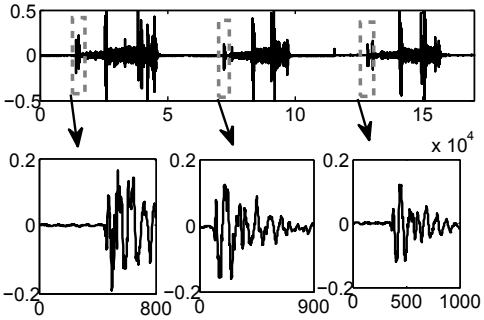
As shown in Fig. 6b(c), the DT-CWT is able to provide a similar representation for subsequences and their shifted versions having different distortion and variances in scale, phase, or amplitude. Therefore, the DT-CWT equips the novel approach with a multiple invariant transformation. Later, the similarity between these transformed subsequences are computed by the mentioned similarity measures after features extraction.

The second test case, “AutoSense” data is gathered from the running research project called “Adaptive energy self-sufficient sensor network for monitoring safety-critical self-service-systems" [44]. The focus of this project is monitoring security critical systems, e. g., the identification of criminal attacks on automated teller machines (ATMs). After identification of several relevant attacks on ATMs, these attacks are tested and the results of them are gathered from sensors connected to the system. This data consists of 24 signals with different lengths, gathered from 8 sensors

in 3 different experiments done on an ATM machine. After dividing this data into segments of variable lengths, our novel approach found the related motifs, depicted in Fig. 7a. It should be mentioned that the depicted motifs in Fig. 7a and 7b are not the only found motifs in this data set. The



(a) Motifs found in the “AutoSense” Data, with sliding windows of length 800



(b) Motifs found in the “AutoSense” Data, with sliding windows of variable lengths

Fig. 7: (a) Motifs found in the “AutoSense” Data by novel and tested Algorithms with sliding windows of length 800, (b) Motifs found in the “AutoSense” data by the proposed approach using sliding windows of different lengths

results obtained by other algorithms showed that motifs such as shown in Fig. 7a can be detected by these algorithms, since all have equal lengths. However, only our proposed approach is able to find motifs of variable lengths, displayed in Fig. 7b.

Moreover, to compare the performance of the proposed approach, each of these algorithms are ran 10 times on each of the mentioned data sets and the average of their execution time is calculated. Table 1 shows the execution time of our approach compared with others. As stated, the proposed

Table 1: Execution time (second) of five algorithms took to find motifs

Method/Data Set	Planted Motif	AutoSense
Mr. Motif [14]	0.11 s	0.16 s
Smart Brute Force [9]	0.12 s	5081.15 s
DAME [5]	0.19 s	5512.25 s
Crammer-Based [10]	0.17 s	3.07 s
MK [45]	0.49 s	4784.08 s
Proposed Approach	0.19 s	127.37 s

approach is based on a brute force algorithm where all the subsequences are compared with each other. In such algorithms the execution time is usually high, however comparing our approach with others showed that in case of small sized data such as "Planted Motif", our approach is as fast as others but when the size of the data increases Mr.Motif and the grammar-based methods can defeat the proposed approach.

6 Conclusion and Outlook

In this paper, a novel approach for multi-scale time-series motif discovery is proposed. The underlying approach employs a time-frequency scale-invariant transformation which is able to handle multiple distortion or variances in time-series. The Dual Tree-Complex Wavelet Transform (DT-CWT) is used in this approach, which overcomes the limitation of the standard DWT. Additionally, this transformation support multiple invariances. Like other Wavelet Transforms, the DT-CWT analyses signals in different frequency pass-bands. Spectral energy density is computed to define the scale with the maximum information content. After selecting the best scale, features are extracted from the wavelet coefficients of that scale. Finally, motifs are found by comparing the extracted features to each other using different distance similarity measures.
The practical results showed that the proposed approach is able to find squeezed/stretched motifs of variable length. The comparison of our results with most common algorithms proved that this novel approach tackles the drawbacks of existing methods.
The benefit of the DT-CWT in industry applications is its fast, simple and hardware efficient implementation. Additionally, it can be extended to higher dimensions. Considering higher dimensions is one of our further working steps.
In order to consider such data, one solution could be the *Quaternion Wavelet Transform (QWT)* [29]. The Quaternion Wavelet Transform is an exten-

sion of the DT-CWT, with improved phase and directionality properties [29]. Furthermore, in order to propose a multiple invariant motif discovery algorithm, the QWT can be applied in *scattering network* [46] hereafter. Scattering networks are invariant, stable and informative transformation which mostly used in signal classification applications [47].
Aforementioned, spectral energy density is considered as a criterion for DT-CWT scale selection. However, other measurements such as kurtosis, variance or entropy [34] should be considered as well. The next step is based on the application of Shannon-entropy [48], which provides a measurement for both energy and entropy. Furthermore, more investigations are necessary regarding similarity measures. A comprehensive introduction of different types of similarity measurements used in time-series data mining task are given in [6, 49]. Finally, semantics and meanings of the found motifs should be investigated later to be able to apply these motifs in broader tasks such as classification or anomaly detection. Therefore, other types of features should be considered in this work.

References

[1] Mönks, U.; Priesterjahn, S.; Lohweg, V.: Automated Fusion Attribute Generation for Condition Monitoring. In: *Proceedings. 23. Workshop Computational Intelligence, Dortmund.*, pp. 339–353. 2013.

[2] Thieullen, A.; Ouladsine, M.; Pinaton, J.: Application of Principal Components Analysis to improve fault detection and diagnosis on semiconductor manufacturing equipment. In: *European Control Conference*, pp. 1445–1500. 2013.

[3] Lohweg, V.; Voth, K.; Glock, S.: A Possibilistic Framework for Sensor Fusion with Monitoring of Sensor Reliability. *Sensor Fusion-Foundation and Applications, intechopen. com* (2011), pp. 191–226.

[4] Patel, P.; Keogh, E.; Lin, J.; Lonardi, S.: Mining motifs in massive time series databases. In: *Proceedings IEEE International Conference on Data Mining*, pp. 370–377. 2002.

[5] Mueen, A.; Keogh, E.; Bigdely-Shamlo, N.: Finding Time Series Motifs in Disk-Resident Data. In: *9th IEEE International Conference on Data Mining.*, pp. 367–376. 2009.

[6] Esling, P.; Agon, C.: Time-series data mining. vol. 45, pp. 1–34. ACM. 2012.

[7] Chaovalit, P.; Gangopadhyay, A.; Karabatis, G.; Chen, Z.: Discrete wavelet transform-based time series analysis and mining. vol. 43. ACM. 2011.

[8] Lohweg, V.; Hoffmann, J. L.; Dörksen, H.; Hildebrand, R.; Gillich, E.; Hofmann, J.; Schaede, J.: Banknote authentication with mobile devices. In: *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics. 2013.

[9] Mueen, A.: Enumeration of Time Series Motifs of All Lengths. In: *IEEE 13th International Conference on Data Mining*. 2013.

[10] Lin, J.; Li, Y.: Finding approximate frequent patterns in streaming medical data. In: *IEEE 23rd International Symposium on Computer-Based Medical Systems*, pp. 13–18. 2010.

[11] Nunthanid, P.; Niennattrakul, V.; Ratanamahatana, C.: Parameter-free motif discovery for time series data. In: *9th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, pp. 1–4. 2012.

[12] Mohammad, Y.; Ohmoto, Y.; Nishida, T.: G-SteX: greedy stem extension for free-length constrained motif discovery. In: *Advanced Research in Applied Artificial Intelligence*, pp. 417–426. Springer. 2012.

[13] Shieh, J.; Keogh, E.: iSAX: indexing and mining terabyte sized time series. In: *Proceedings of the 14th ACM international conference on Knowledge discovery and data mining*, pp. 623–631. ACM. 2008.

[14] Castro, N.; Azevedo, P. J.: Multiresolution Motif Discovery in Time Series. In: *Proceedings of the International Conference on Data Mining*, pp. 665–676. 2010.

[15] Vespier, U.; Nijssen, S.; Knobbe, A.: Mining Characteristic Multi-scale Motifs in Sensor-based Time Series. In: *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, pp. 2393–2398. ACM. 2013.

[16] Grunwald, P.: Tutorial on MDL. *Advances in Minimum Description Length: Theory and Applications* (2005).

[17] Ferrell, B.; Santuro, S.: NASA Shuttle Valve Data (2005). URL `http://www.cs.fit.edu/ pkc/nasa/data/`.

[18] Lin, J.; Keogh, E.; Lonardi, S.; Lankford, J. P.; Nystrom, D. M.: Visually mining and monitoring massive time series. In: *Proceedings of the 10th ACM international conference on Knowledge discovery and data mining*, pp. 460–469. ACM. 2004.

[19] Keogh, E.; Wei, L.; Xi, X.; Vlachos, M.; Lee, S.-H.; Protopapas, P.: Supporting exact indexing of arbitrarily rotated shapes and periodic time series under Euclidean and warping distance measures. *The VLDB Journal, The International Journal on Very Large Data Bases* 18 (2009) 3, pp. 611–630.

[20] Fu, A. W.-C.; Keogh, E.; Lau, L. Y.; Ratanamahatana, C. A.; Wong, R. C.-W.: Scaling and time warping in time series querying. *The International Journal on Very Large Data Bases* 17 (2008) 4, pp. 899–921.

[21] Keogh, E.: Exact indexing of dynamic time warping. In: *Proceedings of the 28th international conference on Very Large Data Bases*, pp. 406–417. 2005.

[22] Batista, G.; Keogh, E.; Tataw, O.; Souza, V.: CID: an efficient complexity-invariant distance for time series. pp. 1–36. Springer US. 2013.

[23] Meyer, Y.; Ryan, R.: *Wavelets: Algorithms and Applications*. Miscellaneous Bks. Society for Industrial and Applied Mathematics. 1993.

[24] Chen, L.; Ozsu, M. T.; Oria, V.: Using Multi-Scale Histograms to Answer Pattern Existence and Shape Match Queries over Time Series Data. In: *Proc. of 17th Int Conf. on Scientific and Statistical Database Management, CA, USA*, pp. 217–226. 2005.

[25] Dawood, H.; Dawood, H.; G., P.: Efficient Texture Classification Using Short-Time Fourier Transform with Spatial Pyramid Matching. In: *IEEE International Conference onSystems, Man, and Cybernetics*, pp. 2275–2279. 2013.

[26] Walker, J. S.: *A primer on wavelets and their scientific applications*. CRC press. 1999.

[27] Burrus, C.; Gopinath, R.; Guo, H.: *Introduction to Wavelets and Wavelet Transforms: A Primer*. Prentice Hall Series in Advanced. Prentice Hall. 1998.

[28] Selesnick, I.; Baraniuk, R.; Kingsbury, N.: The dual-tree complex wavelet transform. In: *IEEE Signal Processing Magazine*, vol. 22, pp. 123–151. 2005.

[29] Chan, W. L.; Choi, H.; Baraniuk, R. G.: Coherent multiscale image processing using dual-tree quaternion wavelets. vol. 17, pp. 1069–1082. IEEE. 2008.

[30] Roberts, T.; Kingsbury, N.; Holland, D.: Sparse recovery of complex phase-encoded velocity images using iterative thresholding. In: *20th IEEE International Conference on Image Processing*, pp. 350–354. 2013.

[31] Zhang, Y.; Kingsbury, N.: Restoration of images and 3D data to higher resolution by deconvolution with sparsity regularization. In: *17th IEEE International Conference on Image Processing*, pp. 1685–1688. 2010.

[32] Kingsbury, N.: Design of q-shift complex wavelets for image processing using frequency domain energy minimization. In: *Proceedings International Conference on Image Processing*. IEEE. 2003.

[33] Kingsbury, N.: Complex wavelets and shift invariance. In: *IEEE Seminar on Time-scale and Time-Frequency Analysis and Applications*. 2000.

[34] Glock, S.; Gillich, E.; Schaede, J.; Lohweg, V.: Feature extraction algorithm for banknote textures based on incomplete shift invariant wavelet packet transform. In: *Pattern Recognition*, pp. 422–431. Springer. 2009.

[35] Mothi, R.; Karthikeyan, M.: A wavelet packet and fuzzy based digital image watermarking. In: *IEEE International Conference on Computational Intelligence and Computing Research*, pp. 1–5. 2013.

[36] Zhenghua-Shu; Cuiqun, H.; Guodong-Liu; Zhong, R.; Zhijun, L.: Image Watermarking Based On Ridgelet Packet with best tree. In: *International Conference on Environmental Science and Information Application Technology*, vol. 2, pp. 116–120. 2010.

[37] Wang, Q.; Li, H.; Liu, J.: Subset selection using rough set in wavelet packet based texture classification. In: *International Conference on Wavelet Analysis and Pattern Recognition*, vol. 2, pp. 662–666. 2008.

[38] Delgado-Contreras, J.; Garcia-Vazquez, J.; Brena, R.: Classification of environmental audio signals using statistical time and frequency features. In: *International Conference on Electronics, Communications and Computers*, pp. 212–216. 2014.

[39] Lohweg, V.; Schaede, J.: Document Production and Verification by Optimization of Feature Platform Exploitation. *Optical Document Security-The Conference on Optical Security and Counterfeit Detection II* (2010), pp. 1–15.

[40] Alpaydin, E.: *Introduction to machine learning*. MIT press, second Aufl. 2010.

[41] Hui-min, L.; Pu, W.; Li-ying, F.; Jing-wei, L.: An algorithm based on time series similarity measurement for missing data filling. In: *24th Control and Decision Conference*, pp. 3933–3935. 2012.

[42] Liu, L.-W.; Ge, J.; Chen, X.-W.: Complex contourlet texture retrieval system using new combinational features. In: *IEEE 3rd International Conference on Communication Software and Networks*. 2011.

[43] Ding, H.; Trajcevski, G.; Scheuermann, P.; Wang, X.; Keogh, E.: Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment* 1 (2008) 2, pp. 1542–1552.

[44] *Adaptive energy self-sufficient sensor network for monitoring safty-critical self-service-systems*. Available at: http://www.hs-owl.de/init/research/projects/b/filteroff/253/single.html. Accessed: Sep, 2014.

[45] Mueen, A.; Keogh, E.; Zhu, Q.; Cash, S.; Westover, M. B.: Exact Discovery of Time Series Motifs. In: *Proceedings of the SIAM International Conference on Data Mining*, pp. 473–484. 2009.

[46] Oyallon, E.; Mallat, S.; Sifre, L.: Generic Deep Networks with Wavelet Scattering. *CoRR, arXiv preprint arXiv:1312.5940* (2013).

[47] *Scatteing Networks*. *http://www.di.ens.fr/data/scattering*. Accessed: Sep, 2014.

[48] Chen, J.; Li, G.: Tsallis Wavelet Entropy and Its Application in Power Signal Analysis. Multidisciplinary Digital Publishing Institute. 2014.

[49] Fu, T. C.: A review on time series data mining. vol. 24, pp. 164–181. Elsevier. 2011.

Analyse globaler Bildmerkmale zur Klassifikation von Verkehrsszenen

Malte Oeljeklaus, Felipe Posada, Frank Hoffmann, Torsten Bertram

Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund
Otto-Hahn-Str. 8, 44227 Dortmund
Tel.: (0231) 7554648
Fax: (0231) 7552752
E-Mail: malte.oeljeklaus@tu-dortmund.de

1 Einleitung

Auf dem Gebiet der bildbasierten Szenarienklassifikation wurden in der jüngsten Vergangenheit erhebliche Fortschritte erzielt. Aktuelle Verfahren sind in der Lage effektiv zwischen vielfältigen Kategorien von Szenarien zu diskriminieren, indem Modelle der Kategorien aus Bilddatensätzen gelernt werden [5] [2]. Diese Entwicklungen bieten Potential für Anwendungen im Automotive Bereich, da kommerziell verfügbare Fahrerassistenzsysteme typischerweise darauf ausgelegt sind, in speziellen, festgelegten Szenarien Aspekte der Fahraufgabe zu übernehmen. Methoden zur Szenarienklassifikation liefern Informationen über die Umgebung in welcher sich das Fahrzeug befindet, diese erlauben es Annahmen über die Umgebung zu adaptieren und so Systeme zu entwickeln, die in einem größeren Bereich von Umgebungen und Szenarien angewendet werden können. Aktuelle Satellitennavigationssysteme sind anfällig für Empfangsprobleme und basieren auf potentiell veralteten Karteninformationen, welche ihre Zuverlässigkeit begrenzen. Weiterhin können sich Umgebungsinformationen, wie beispielsweise zur Lage von Baustellen, dynamisch ändern, was einen Ansatz auf Grundlage von in Echtzeit erfassten Daten erforderlich macht. Da die visuelle Erscheinung eine Klassifikation der Szene vergleichsweise gut zulässt, bieten Kamerasensoren zu diesem Zweck die besten Voraussetzungen. Der Beitrag untersucht das Problem der Klassifikation von Verkehrsszenen in typische Kategorien auf Grundlage von Kamerabildern. Um dieses Problem allgemein zu lösen ist es erforderlich, dass der gewählte Ansatz in der Lage ist eine Vielzahl verschiedener Kategorien zu unterscheiden. Gleichzeitig kann die Erscheinung einzelner Verkehrsansichten, welche einer speziellen Kategorie zugeordnet sind, einer großen Variation unterliegen. Manuell konstruierte Methoden sind hierzu

nicht geeignet, weshalb dieser Beitrag untersucht, wie sich durch überwachtes Lernen datenbasiert Modelle für die einzelnen Kategorien von Verkehrsszenen bestimmen lassen. Der Beitrag folgt hierzu dem verbreiteten Ansatz, zunächst charakteristische globale Merkmale aus dem Kamerabild zu extrahieren und anschließend universelle Klassifikationsalgorithmen anzuwenden, um letztendlich die Kategorie einer Verkehrsszene zu bestimmen. Hierzu wird eine Reihe von Kombinationen aus globalen Bildmerkmalen und Klassifikationsalgorithmen in einem zweistufigen Ansatz untersucht. Dabei wird in einer ersten Stufe ein Ensemble aus mehreren Basisklassifikatoren gebildet. Anschließend werden in einer zweiten Stufe die Prädiktionen der Basisklassifikatoren von einem weiteren Klassifikator zur Ausgabe des Gesamtsystems fusioniert. Der Fokus des Beitrages liegt dabei auf der vergleichenden Analyse verschiedener Verfahren zur Extraktion globaler Bildmerkmale sowie verschiedener Klassifikationsalgorithmen hinsichtlich ihrer Eignung zur Klassifikation von Verkehrsszenen.

Der Beitrag gliedert sich wie folgt: zunächst gibt der folgende Abschnitt eine Übersicht der einschlägigen Literatur, anschließend wird im dritten Abschnitt der Datensatz beschrieben, welcher in diesem Beitrag verwendet wurde. Anschließend enthält der vierte Abschnitt eine Beschreibung der Systemarchitektur des untersuchten Lösungsansatzes. Der fünfte Abschnitt enthält eine Evaluation der Genauigkeit des Systems hinsichtlich der Klassifikation, abschließend folgt eine Schlussfolgerung.

2 Verwandte Arbeiten

Es existieren mehrere Veröffentlichungen, welche das Problem der Klassifikation von Kamerabildern in verschiedene Kategorien von Szenarien allgemein behandeln. Ein grundlegender Ansatz hierzu besteht darin, Histogramme von lokalen Bildmerkmalen zu bilden. Insbesondere das Bag of Words [10] Modell stellt dabei eine häufig angewendete Methode dar. Hierbei wird die Einteilung der Histogramme nicht äquidistant im Merkmalsraum gewählt, sondern stattdessen wird ein visuelles Wörterbuch durch eine Clusteranalyse im Merkmalsraum bestimmt. Aus der Häufigkeit, mit welcher die visuellen Wörter des Wörterbuchs auftreten, ergibt sich ein Histogramm, welches charakteristisch für verschiedene Kategorien von Szenarien ist.
[4] präsentiert PHOG Merkmale zur Szenarienklassifikation, welcher sich ergibt wenn die Methode der sogenannten „spatial pyramids“ [5] auf Merkmale aus Histogrammen orientierter Gradienten (HOG) [9] angewendet wird. Dabei wird das Bild in Teilbereiche abnehmender Größe unterteilt

und für jeden Teilbereich werden anschließend Merkmalshistogramme bestimmt. Im Gegensatz zum Bag of Words Modell ist das Schema der „spatial pyramids“ in der Lage, räumliche Korrespondenzen von Bildmerkmalen abzubilden.
Zur Szenarienklassifikation entwickelt [2] einen Ansatz, welcher unter der Bezeichnung GIST Merkmale bekannt ist und darauf basiert, Gabor Filter auf Kamerabilder anzuwenden. Der Ansatz wird unter anderem auf Aufnahmen im Außenbereich in urbanen Umgebungen, welche künstliche Objekte wie Straßen und Gebäude enthalten, evaluiert und erzielt dort gute Ergebnisse. Die Literatur enthält gegensätzliche Informationen über die Anwendbarkeit der GIST Merkmale für die Klassifikation von Verkehrsansichten [1] [3].

Nur wenige Arbeiten decken das spezielle Problem der Klassifikation von Verkehrsszenarien ab. Eine Ausnahme ist die Arbeit in [1], welche auf der Transformation von Bildern in den Frequenzbereich basiert. Das Konzept weißt dabei Ähnlichkeiten zu der „spatial pyramid“ Idee auf, da das Frequenzspektrum zur Merkmalsextraktion in ähnlicher Weise in Teilbereiche zerlegt wird. Eine Variante dieses Ansatzes untersucht [7]. Die Autoren berichten von guten Ergebnissen auf den untersuchten Datensätzen.
Eine weitere Arbeit, welche die Klassifikation von Verkehrsszenen betrachtet ist [3], welche einen zweistufigen Ansatz untersucht. Die erste Stufe führt eine Superpixel Segmentierung der Szene durch, anschließend verwendet die zweite Stufe die Superpixel Repräsentation um charakteristische Bildmerkmale zu berechnen. Zur Evaluation dieses Ansatzes wird die Klassifikation des Straßenverlaufs untersucht, dabei werden vergleichbare Ergebnisse zu denen der GIST Merkmale erzielt.
In Bezug auf die Anwendung dieser Methoden zur Unterstützung der Regelung mobiler Plattformen fokussiert sich der Großteil der vorhandenen Arbeiten auf mobile Service-Roboter. [6] präsentiert eine Systemarchitektur für mobile Roboter, welche Kontextinformationen über die Umgebung sammelt, um ein semantisches Verständnis der Umgebung zu erhalten. Hierdurch werden die autonomen Fähigkeiten des Roboters unterstützt, außerdem ermöglicht die semantische Repräsentation der Umgebung die Interaktion von Mensch und Roboter in natürlicher Sprache. Der Ansatz basiert auf globalen Bildmerkmalen und Histogramm Repräsentationen lokaler Bildmerkmale und führt anschließend eine Vorhersage der Umgebungskategorie (Raum, Korridor, Tür, Freifläche) durch Anwendung einer Support Vector Maschine (SVM) durch.

3 Datensatz

Bild 1: Beispielhafte Aufnahmen des Datensatzes aus den drei betrachteten Kategorien, oben links: Stadt (S), oben rechts: Landstraße (L), unten: Autobahn (A)

In dieser Arbeit wird das Problem der Klassifikation von Verkehrsansichten in die drei Kategorien Stadt, Landstraße und Autobahn untersucht. Der Datensatz besteht aus 330 Bildern mit einer Auflösung von jeweils 1280x556 Bildpunkten, welche von einer Kamera hinter der Windschutzscheibe aufgenommen wurden. Pro Kategorie sind somit 110 Aufnahmen vorhanden. Der Datensatz wird für eine fünffache Kreuzvalidierung vorbereitet, dies entspricht jeweils einer Aufteilung in 60% Trainings-, 20% Validierungs- und 20% Testdaten. Um unerwünschte, künstliche Korrelationen zwischen den einzelnen Aufnahmen zu vermeiden wurde ein minimaler zeitlicher Abstand von 15 s zwischen den Aufnahmen sichergestellt, wobei Aufnahmen längerer Standzeiten beispielsweise an Ampeln manuell aussortiert wurden. Insgesamt wurden Aufnahmen von vier verschiedenen Tagen ausgewertet. Der Datensatz ist anspruchsvoll im Vergleich zu anderen Datensätzen [14], da Verkehrsansichten in verschiedenen Umgebungen vergleichsweise viele ähnliche Bestandteile der Szene enthalten. Abbildung 1 zeigt beispielhafte Aufnahmen aus den drei Kategorien .

4 Systemarchitektur

Der untersuchte Ansatz verwendet Aufnahmen einer einzelnen Kamera als Eingang. Es werden dabei keine zeitlichen Korrelationen zwischen sequentiell aufgenommenen Bildern ausgewertet. Ebenso wird die Möglichkeit einer geometrische Rekonstruktion der Szene nicht betrachtet. Stattdessen

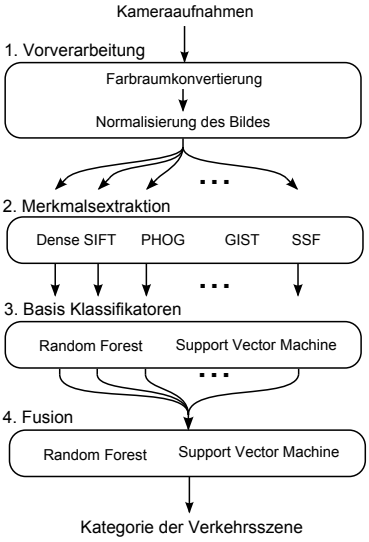


Bild 2: Systemarchitektur des untersuchten Ansatzes

basiert das System direkt auf der Extraktion charakteristischer Merkmale aus den Kamerabildern. Um die Gesamtleistung des Systems zu erhöhen, wird ein Ensemble aus mehreren Basis Klassifikatoren gebildet, welche jeweils auf unterschiedlichen Merkmalen trainiert werden. Die Fusion der Ausgänge der verschiedenen Basis Klassifikatoren erfolgt durch Anwendung der Stacking Methode, hierbei werden die Ausgänge der Basis Klassifikatoren als Eingänge eines zusätzlichen Klassifikators verwendet. Es handelt sich somit um einen „late fusion“ Ansatz. Die Systemarchitektur zeigt Abbildung 2. Im Folgenden werden die verschiedenen Verarbeitungsschritte detailliert beschrieben.

4.1 Vorverarbeitung

Die Wahl des Farbraumes kann signifikanten Einfluss auf die resultierende Genauigkeit des Klassifikators haben. Aus diesem Grund enthält der Vorverarbeitungsschritt eine Farbraumkonvertierung. Die endgültige Wahl des Farbraumes erfolgt in Abschnitt 5. Weiterhin wird, entsprechend der Ergenisse in [8], eine Bildnormalisierung durchgeführt. Dabei werden die Aufnahmen unabhängig voneinander mittelwertbefreit und auf eine Standardabweichung von 1 normalisiert.

4.2 Merkmalsextraktion

Mehrere verschiedene Merkmalsvektoren werden hinsichtlich ihrer Eignung für Verkehrsansichten untersucht, so wie GIST Merkmale [2], das Pyramiden Histogramm orientierter Gradienten (PHOG) [4], das Bag of Words Modell basierend auf dicht extrahierten SIFT Merkmalen [10] und der Merkmalsvektor aus [1], welcher im Folgenden als „spectral sampling features“ (SSF) bezeichnet wird. Nachfolgend wird eine detaillierte Beschreibung der verschiedenen Merkmalsvektoren aufgeführt.

GIST: Diese Merkmalsrepräsentation wurde explizit für den Zweck der Szenarienklassifikation entwickelt und hat in vorherigen Evaluationen gute Ergebnisse für Aufnahmen im Außenbereich erzielt [2]. Die Berechnung des GIST Merkmalsvektors basiert auf einer Gabor Filterbank mit 8 Orientierungen. Die Filterantworten werden jeweils auf Teilbereichen des Bildes auf einem 4x4 Raster ohne Überlappung berechnet, die einzelnen Filterantworten werden anschließend für jeden Teilbereich gemittelt. Für die vorliegende Arbeit wurde der Quellcode, welcher in [2] verfügbar ist, auf unterabgetasteten Aufnahmen mit 256x256 Bildpunkten angewendet.

SIFT (dicht extrahiert): Der „scale invariant feature transform“ (SIFT) Merkmalsvektor ist eine Standard Methode für eine Vielzahl von Anwendungen in der Bildverarbeitung. Dabei werden in lokalen Umgebungen bestimmter Schlüsselpunkte Orientierungshistogramme der Gradienten gebildet. Durch Aneinanderhängen der Werte der Histogramme entsteht ein Merkmalsvektor pro Schlüsselpunkt. Werden die Schlüsselpunkte auf einem regelmäßigen Raster gewählt, spricht man von der dicht extrahierten Variante der SIFT Merkmale. Zur Reduktion der Dimensionalität werden die Merkmalsvektoren auf ein visuelles Wörterbuch nach dem Bag of Words Modell [10] abgebildet. Das visuelle Wörterbuch wird dabei durch Einsatz des k-Means Algorithmus gebildet und besteht aus 200 Wörtern. Aus den Häufigkeiten, mit welchen die visuellen Wörter auftreten, wird ein Histogramm geformt welches anschließend als Merkmalsvektor für die Szenarienklassifikation verwendet wird. Hierzu wird der in [12] verfügbare Quellcode eingesetzt.

PHOG: Das Histogramm orientierter Gradienten (HOG) [9] wird gebildet, indem die Aufnahme nach Berechnung der Bildgradienten in regelmäßig gerasterte Teilbereiche zerlegt wird und anschließend Orientierungshistogramme auf für jeden Teilbereich berechnet werden. Das Pyramiden-Histogramm orientierter Gradienten (PHOG) [4] folgt der Idee der „spatial pyramid“, dabei wird die Aufnahme pyramidenartig in Teilbereiche abnehmender Größe zerlegt und anschließend der HOG Merkmalsvektor für

jeden Teilbereich auf jeder Ebene der Pyramide berechnet. Für die Implementierung wurde der in [4] verfügbare Quellcode verwendet.

SSF: Dieses Merkmal basiert auf der Repräsentation der Aufnahme im Frequenzbereich [1]. Die Berechnung erfolgt, indem zunächst ein Hochpassfilter auf das Bild angewendet wird. Anschließend wird die Aufnahme in Teilbereiche zerlegt und die diskrete Fouriertransformation auf jeden Teilbereich angewendet. Die Beträge der resultierenden Spektren werden abgetastet, wobei eine hohe Auflösung für niedrige Frequenzen und eine niedrige Auflösung für hohe Frequenzen verwendet wird. Hierzu werden Mittelwertfilter mit einer Filtermaske verwendet, welche sich aus einer quartären Baumstruktur (sog. „Quadtree“) ergibt [7]. Die resultierenden Werte bilden anschließend den Merkmalsvektor.

4.3 Basis Klassifikatoren

Der untersuchte Ansatz verwendet ein Ensemble aus Basis Klassifikatoren, welche jeweils unterschiedliche Merkmalsvektoren verarbeiten und auf dieser Grundlage Konfidenzwerte für jede untersuchte Szenenkategorie prädizieren. Grundsätzlich lässt sich jeder universelle Klassifikationsalgorithmus als Basis Klassifikator einsetzen, solange dieser als Ausgangsgröße nicht lediglich eine Klassifikation sondern auch Konfidenzwerte erzeugt. Die vorliegende Arbeit untersucht als Klassifikationsalgorithmen Support Vector Maschinen (SVM) und Random Forests (RF), welche im Folgenden beide genauer beschrieben werden.

SVM: Eine SVM löst im allgemeinen dass binäre Klassifikationsproblem, bei welchem zwischen zwei verschiedenen Klassen unterschieden wird. Sie wird aus einem Trainingsdatensatz gelernt, indem die zu den verschiedenen Klassen gehörigen Merkmalsvektoren durch lineare Hyperebenen voneinander separiert werden. Die Hyperebenen werden dabei so gewählt, dass der Abstand der Ebenen von den Merkmalsvektoren, welche die einzelnen Klassen repräsentieren, maximiert wird. Der Abstand eines Merkmalsvektors von der Hyperebene kann weiterhin als Konfidenzwert interpretiert werden. Wenn $\mathbf{w}$ der Einheitsnormalenvektor der Hyperebene und b der Abstand der Hyperebene vom Koordinatenursprung ist, ergibt sich somit die einem Merkmalsvektor $\mathbf{x}$ zugeordnete Klasse $\mathbf{y}$, sowie der zugeordnete Konfidenzwert p zu:

$$\mathbf{y} = \text{sign}(\mathbf{x} \cdot \mathbf{w} - b) \ , \ \ p = \mathbf{x} \cdot \mathbf{w} - b \tag{1}$$

Da das untersuchte Klassifikationsproblem nicht binärer Natur ist, wird die „one-vs-all“ Technik eingesetzt, bei welcher eine SVM pro Szenenkategorie trainiert wird. Auf diese Weise ergeben sich Konfidenzwerte jeweils für alle untersuchten Kategorien.
Eine lineare Entscheidungsgrenze im Merkmalsraum lässt häufig keine zuverlässige Unterscheidung der verschiedenen Klassen zu, aus diesem Grund verwenden SVM sogenannte Kernel Funktionen, welche die Merkmalsvektoren in einem höher dimensionalen Raum abbilden, in welchem lineare Entscheidungsgrenzen anwendbar sind. Die Kernel Funktionen definieren dabei Ähnlichkeitsmaße zwischen Paaren von Merkmalsvektoren, häufig verwendete Funktionen sind radiale Basis Funktionen (RBF), die χ^2-Funktion und die Histogram Intersection (HI) Funktion [14]:

$$k_{rbf}(\vec{x},\vec{y}) = \sum_{i=1}^{n} \mathrm{e}^{\frac{|x_i-y_i|^2}{2\mu}} \tag{2}$$

$$k_{\chi^2}(\vec{x},\vec{y}) = \sum_{i=1}^{n} \frac{(x_i-y_i)^2}{(x_i+y_i)} \tag{3}$$

$$k_{hi}(\vec{x},\vec{y}) = \sum_{i=1}^{N} \min(x_i,y_i) \tag{4}$$

Diese Arbeit verwendet den in [11] verfügbaren Quellcode zur Evaluation der SVM.

RF: Random Forests ergeben sich aus Mengen von Entscheidungsbäumen [13]. Die einzelnen Entscheidungsbäume ergeben sich durch Anwendung der Bootstrap Aggregation Methode, bei welcher jeweils eine zufällige Teilmenge des Trainingsdatensatzes für das Trainieren der einzelnen Entscheidungsbäume verwendet wird. Weiterhin wird für jeden Entscheidungsbaum nur eine zufälliger Teil des Merkmalsvektors verwendet. Jeder Entscheidungsbaum kann somit einer gegebenen Verkehrsansicht eine Szenenkategorie zuordnen. Die benötigten Konfidenzwerte ergeben sich aus dem Verhältnis der Anzahl der Entscheidungsbäume, welche jeweils bestimmte Kategorien ausgewählt haben und der Gesamtanzahl der Entscheidungsbäume.

4.4 Fusion

Die Ausgaben eines gegebenen Ensembles aus Basis Klassifikatoren müssen zu einer Gesamtausgabe fusioniert werden, welche die Kategorie der Verkehrsszene angibt. Ansätze, welche ausschließlich den SVM Algorith-

mus zur Klassifikation einsetzen, verwenden hierzu häufig die Methode der multiplen Kernel SVM wie in [14]. Sollen die Prädiktionen mehrerer verschiedener Klassifikationsalgorithmen fusioniert werden, besteht die einfachste Möglichkeit darin, eine Mehrheitsentscheidung (majority vote) durchzuführen. Nach [16] erzielt jedoch die ursprünglich von [15] eingeführte sogenannte „stacked generalization“ Methode im Allgemeinen bessere Ergebnisse, weshalb diese im Folgenden verwendet wird. Dabei werden die Konfidenzwerte, welche von den einzelnen Basisklassifikatoren $b_1 - b_n$ bestimmt werden, zu einem einzelnen Vektor aneinandergereiht und als Eingabe eines zusätzlichen Meta-Klassifikators verwendet wie in Abbildung 3 dargestellt. Dieser Meta-Klassifikator der zweiten Stufe ermittelt

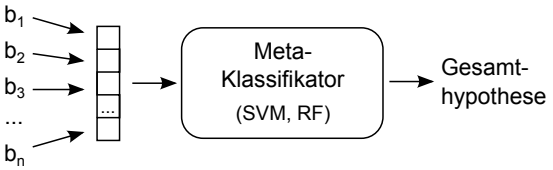


Bild 3: Schematische Darstellung der „stacked-generalization“ Methode

schließlich die Gesamthypothese, also die gegebene Szenenkategorie. Für den Klassifikator der zweiten Stufe sind dabei die selben Algorithmen anwendbar, wie für die Basis Klassifikatoren.

5 Ergebnisse

In diesem Abschnitt wird die Leistung des untersuchten Ansatzes evaluiert. Die Evaluation wird dabei entsprechend der Systemarchitektur in zwei Schritten durchgeführt, zunächst werden die Basis Klassifikatoren evaluiert und entsprechend ihrer Genauigkeit sortiert. Anschließend wird die Leistung des Gesamtsystems evaluiert, indem ein Ensemble aus den Basis Klassifikatoren mit der höchsten Genauigkeit gebildet wird. Die Auswertung wird dabei entsprechend einer fünffachen Kreuzvalidierung wiederholt durchgeführt. Die im Folgenden dargestellten Ergebnisse bilden jeweils die Mittelwerte aus den fünf durchgeführten Auswertungen ab.

5.1 Basis Klassifikatoren

Zunächst werden die Konfigurationsparameter der Basis Klassifikatoren unter Verwendung von Graustufenbildern optimiert.

SVM: Zur Auswahl der Kernel Funktion werden die Gleichungen (2)-(4) für alle betrachteten Merkmale ausgewertet. Die resultierende Genauigkeit der Klassifikation zeigt Abbildung 4.

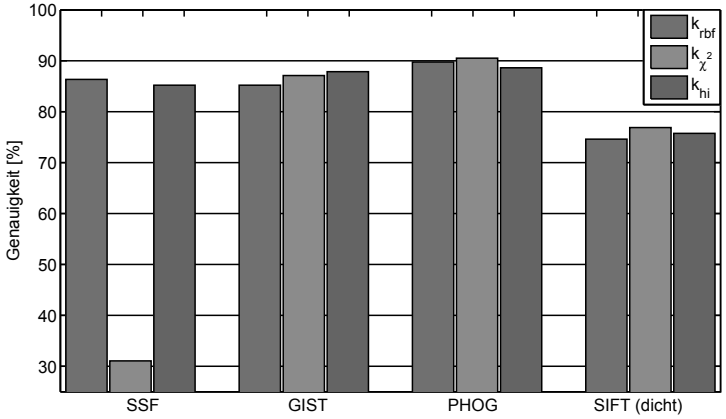


Bild 4: Genauigkeit der SVM Basis Klassifikatoren unter Verwendung von Graufstufenbildern für verschiedene Kernel Funktionen

Auffallend ist zunächst die schlechte Klassifikationsgenauigkeit bei Kombination der SSF Merkmale mit der k_{χ^2} Kernelfunktion. Da sich für die Trainingsdaten ähnlich schlechte Genauigkeiten ergeben ist davon auszugehen, dass eine lineare Separierbarkeit nach Anwendung der Kerneltransformation in diesem Fall nicht gegeben ist, die Art der Nichtlinearität für die gegebene Verteilung im Merkmalsraum also ungeeignet ist. Aus der Abbildung geht weiterhin hervor, dass sich die höchste Genauigkeit für die SSF Merkmalsvektoren unter Verwendung der k_{rbf} Kernel Funktion ergibt. Die k_{hi} zeigt unter Verwendung der GIST Merkmale die besten Ergebnisse und im Falle der PHOG Merkmale sowie der dicht abgetasteten SIFT Merkmale ist die k_{χ^2} Funktion überlegen. Daher wird die weitere Evaluation unter Verwendung dieser Kernel Funktionen durchgeführt. Der PHOG Merkmalsvektor führt zu der höchsten Genauigkeit von $90{,}5\,\%$, gefolgt von den GIST Merkmalen mit $87{,}9\,\%$, den SSF Merkmalen mit $86{,}4\,\%$ und schließlich den SIFT Merkmalen mit $76{,}9\,\%$.

RF: Um die Anzahl der Entscheidungsbäume der Random Forest Basis Klassifikatoren zu bestimmen, ist in Abbildung 5 die Genauigkeit über der Anzahl der Entscheidungsbäume aufgetragen.

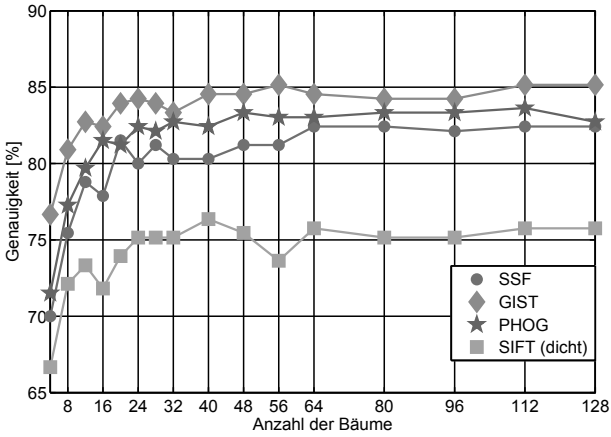


Bild 5: Genauigkeit der RF Basis Klassifikatoren unter Verwendung von Graustufenbildern

Bei Verwendung der GIST Merkmale ergibt sich eine maximale Genauigkeit von $85{,}15\,\%$ mit einer Anzahl von 56 Entscheidungsbäumen, gefolgt von den PHOG Merkmalen mit einer Genauigkeit von $83{,}64\,\%$, welche mit 112 Bäumen erreicht wird. Anschließend folgen die SSF Merkmale mit $82{,}42\,\%$ bei zuerst 64 Entscheidungsbäumen und den dicht abgetasteten SIFT Merkmalen mit $76{,}36\,\%$ und einer Anzahl von 40 Entscheidungsbäumen. Weiterhin geht aus der Abbildung hervor, dass die Genauigkeit der Klassifikation bei einer Anzahl von 64 Entscheidungsbäumen als saturiert angenommen werden kann, weshalb im Folgenden diese Anzahl beibehalten wird. Ebenso ist festzustellen, dass sich im Vergleich mit den SVM etwas geringere Genauigkeiten ergeben.

Die Evaluation der parametrierten Basis Klassifikatoren erfolgt sowohl unter Verwendung von Graustufenbildern als auch unter Verwendung aller Kanäle der RGB, HSV, CIE-XYZ und LAB Farbräume. Nachfolgend werden jeweils nur diejenigen Kombinationen aus Merkmalsvektor und Farbkanal genannt, welche zu den höchsten Genauigkeiten bei der Klassifikation führen. Die Ergebnisse zeigt Tabelle 1, hierin sind entsprechend der Kreuzvalidierung die gemittelten Klassifikationsgenauigkeiten aufgeführt.

Aus der Tabelle ist ersichtlich, dass sowohl der RF als auch der SVM Klassifikator für PHOG Merkmale auf dem A-Kanal des LAB Farbraums die

Tabelle 1: Auflistung der Basis Klassifikatoren

	Stacking-Klassifikator	Farb-raum	Merkmal	Genauigkeit in % S / L / A / **gesamt**
b_1	SVM	A	PHOG	93.6 / 93.6 / 92.7 / **93.3**
b_2	SVM	(rg)B	GIST	94.5 / 84.5 / 91.8 / **90.3**
b_3	SVM	V	PHOG	96.4 / 79.1 / 92.7 / **89.4**
b_4	SVM	S	GIST	90 / 82.7 / 94.5 / **89.1**
b_5	SVM	R	PHOG	96.4 / 79.1 / 90.9 / **88.8**
b_6	SVM	gray	PHOG	92.7 / 78.2 / 94.5 / **88.5**
b_7	RF	A	PHOG	91.3 / 88.5 / 84.7 / **88.2**
b_8	SVM	(rg)B	SSF	92.7 / 80.9 / 90 / **87.9**
b_9	SVM	V	GIST	90.9 / 83.6 / 89.1 / **87.9**
b_{10}	SVM	L	GIST	90.9 / 82.7 / 89.1 / **87.6**
b_{11}	SVM	A	SSF	85.5 / 86.4 / 90 / **87.3**
b_{12}	SVM	(la)B	SSF	89.1 / 86.4 / 85.5 / **87**
b_{13}	SVM	Y	GIST	87.3 / 79.1 / 92.7 / **86.4**
b_{14}	RF	V	GIST	91.3 / 75.8 / 90.2 / **85.8**
b_{15}	RF	(rg)B	PHOG	90 / 75.6 / 88.7 / **84.8**

besten Ergebnisse erzielen. Die maximale Genauigkeit des SVM Klassifikators ist dabei höher als die des RF Klassifikators, welcher lediglich dreimal in der Tabelle aufgeführt ist. Für die GIST Merkmale ergibt sich eine etwas verringerte Genauigkeit, gefolgt von den SSF Merkmalen. Die Ergebnisse der dicht abgetasteten SIFT Merkmale bleiben erneut deutlich hinter denen der PHOG, GIST und SSF Merkmale zurück und sind deshalb in der Tabelle nicht mehr enthalten.

5.2 Fusion

Die Evaluation des Gesamtsystems erfolgt sowohl für RF als auch für SVM als Klassifikator der zweiten Stufe. Für den RF Algorithmus wird dabei erneut eine Anzahl von 64 Entscheidungsbäumen verwendet, für die SVM wird die Kernel Funktion k_{rbf} eingesetzt, da diese zu den besten Ergebnissen führte. Es ist naheliegend anzunehmen, dass viele Basis Klassifikatoren zu einer hohen Genauigkeit des Gesamtsystems führen. Gleichzeitig führen viele Basis Klassifikatoren zu einer erhöhten Komplexität des Gesamtsystems. Um die Wahl der Ensemble Größe zu vereinfachen, zeigt Abbildung 6 die resultierende Genauigkeit, wenn die einzelnen Basis Klassifikatoren jeweils sukzessive zum Ensemble hinzugefügt werden,

beginnend mit einer Ensemblegröße von eins. Weiterhin ist der Basis Klassifikator mit der besten Genauigkeit durch b^* gekennzeichnet.

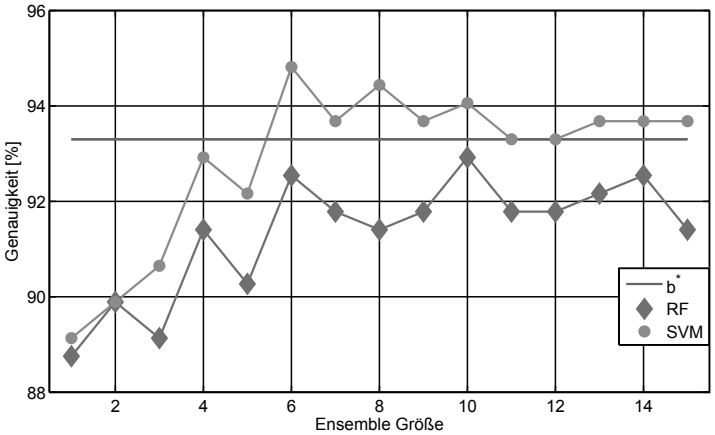


Bild 6: Genauigkeit des Gesamtsystems aufgetragen über der Anzahl der Basis Klassifikatoren

Die Variante mit der SVM als Stacking Klassifikator erreicht die maximale Genauigkeit früher mit einer Ensemble Größe von 6, während die Variante mit dem RF Klassifikator dies etwas später mit einer Ensemble Größe von 10 erreicht. Wie erwartet ist die Tendenz einer zunehmenden Genauigkeit mit zunehmender Ensemble Größe erkennbar. Die Variante mit der SVM erzielt eine maximale Genauigkeit von $94.82\,\%$ und übertrifft damit den besten Basis Klassifikator b^*, für den RF als Stacking Klassifikator ergibt sich eine maximale Genauigkeit von $92.92\,\%$, somit wird die Genauigkeit des besten Basis Klassifikators nicht erreicht. Insgesamt kann angenommen werden, dass die Genauigkeit des untersuchten Systems mit weniger als 15 Basis Klassifikatoren ihren Maximalwert erreicht.

Die Klassifikationsergebnisse werden nun noch einmal im Detail ausgewertet. Die Tabellen 2 und 3 zeigen die Konfusionsmatrizen für eine Ensemble Größe von 15, welche jeweils entsprechend der Kreuzvalidierung die kumulierten Werte enthalten.

Die Konfusionsmatrizen zeigen, dass das untersuchte System die Aufgabe der Klassifikation von Verkehrsansichten in die Kategorien Stadt, Landstraße und Autobahn mit einer hohen Genauigkeit löst. Die Konfusionsmatrix für die SVM Variante zeigt erneut etwas bessere Klassifikationsergebnisse als für die RF Variante. In beiden Fällen wurde für die Kategorie

Tabelle 2: RF Konfusionsmatrix

	S	L	A
S	103	6	1
L	6	90	14
A	0	8	102

Tabelle 3: SVM Konfusionsmatrix

	S	L	A
S	107	2	1
L	6	96	8
A	0	9	101

Stadt die höchste Genauigkeit erzielt, gefolgt von den Kategorien Autobahn und Landstraße.

6 Schlussfolgerung

Der Beitrag untersucht ein System zur Klassifikation von Verkehrsszenen auf Grundlage von Kamerabildern. Hierzu wird eine Anzahl verschiedener Bildmerkmale und generischer Klassifikationsalgorithmen ausgewertet. Das untersuchte System folgt dem Ensemble Ansatz, dabei wird eine Menge von Basis Klassifikatoren trainiert und in einem zweiten Schritt ein Stacking Klassifikator angewandt. Das System wird anhand der Kategorien Stadt, Landstraße und Autobahn auf einem eigens hierzu angelegten Datensatz evaluiert. Die Ergebnisse zeigen eine hohe Genauigkeit für alle untersuchten Kategorien. Unter Verwendung des SVM Algorithmus sowohl auf Ebene der Basis Klassifikatoren als auch auf der Ebene der Fusion ergeben sich dabei tendenziell bessere Genauigkeiten als unter Verwendung des RF Algorithmus.
Für zukünftige Arbeiten ist es geplant, die Genauigkeit des Ansatzes durch eine zeitliche Fusion der aufgezeichneten Bilder weiter zu verbessern. Ebenso soll der Ansatz für weitere Kategorien von Verkehrsansichten und anderen Datensätzen untersucht werden. Weiterhin sind Untersuchungen geplant, auf Grundlage der Kenntnis der Szenenkategorie aufbauende Systeme zu adaptieren und so deren Leistung zu steigern. Beispielsweise könnten Modelle zur Vorhersage des Straßenverlaufs speziell für Stadt, Landstraßen und Autobahnen entwickelt werden und zusammen potentiell eine höhere Genauigkeit der Vorhersage erzielen als nicht adaptive Modelle.

Danksagung

Diese Forschung wurde vom Land NRW im Rahmen des Ziel2 Projektes „Technologie- und Prüfplattform für ein Kompetenzzentrum für interoperable Elektromobilität, Infrastruktur und Netze“ (TIE-IN) finanziert.

Literatur

[1] R. Kastner, F. Schneider, T. Michalke, J. Fritsch and C. Goerick, Image-based classification of driving scenes by Hierarchical Principal Component Classification (HPCC), Proc. IEEE Intelligent Vehicles Symposium, 2009, pp.341-346

[2] A. Oliva and A. Torralba, Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope, International Journal of Computer Vision, Band 42, Ausgabe 3, 2001, pp.145-175

[3] A. Ess, T. Mueller, H. Grabner and L. J. Van Gool, Segmentation-Based Urban Traffic Scene Understanding, Proc. BMVA British Machine Vision Conf., 2009, pp.84.1-84.11

[4] A. Bosch, A. Zisserman and X. Munoz, Representing shape with a spatial pyramid kernel, Proc. ACM Conf. on Image and video retrieval, 2007, pp.401-408

[5] S. Lazebnik, C. Schmid and J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2006, pp.2169-2178

[6] L.F. Posada, K. K. Narayanan, F. Hoffmann and T. Bertram, Semantic classification of scenes and places with omnidirectional vision, Proc. IEEE European Conf. on Mobile Robots (ECMR), 2013, pp.113-118

[7] N. Bernini, M. Bertozzi, L. Devincenzi and L. Mazzei, Comparison of Three Approaches for Scenario Classification for the Automotive Field, Proc. IEEE Conf. on Image Analysis and Processing, 2013, pp.582-591

[8] Y. Alon, A. Ferencz, A. Shashua, Off-road Path Following using Region Classification and Geometric Projection Constraints, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2006, pp.689-696

[9] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2005, pp.886-893

[10] L. Fei-Fei and P. Perona, A Bayesian Hierarchical Model for Learning Natural Scene Categories, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2005, pp.524-531

[11] C. C. Chang and C. J. Lin, LIBSVM: A library for support vector machines, ACM Trans. on Intelligent Systems and Technology, Band 2, Ausgabe 3, 2011

[12] A. Vedaldi and B. Fulkerson, VLFeat: An Open and Portable Library of Computer Vision Algorithms, http://www.vlfeat.org, 2008

[13] L. Breiman, Random Forests, Machine Learning, Band 45, Ausgabe 1, 2001, pp.5-32

[14] J. Xiao, J. Hays, K. Ehinger, A. Oliva and A. Torralba, SUN Database: Large-scale Scene Recognition from Abbey to Zoo, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2010, pp.3485-3492

[15] D. H. Wolpert, Stacked generalization, Neural networks Band 5, Ausgabe 2, 1992, pp.241-259

[16] G. Sigletos, G. Paliouras, C. D. Spyropoulos, M. Hatzopoulos, Combining information extraction systems using voting and stacked generalization, Journal of Machine Learning Research Band 6, 2005, pp.1751-1782

Fuzzy Pattern Klassifikation von Datensätzen mit nichtkonvexen Objektmorphologien

Richard Neumann[1], **Alexander Dicks**[2], **Uwe Mönks**[2] **und Volker Lohweg**[2]

[1]HANNING ELEKTRO-WERKE GmbH & Co. KG
Holter Str. 90, D-33813 Oerlinghausen
Tel.: +49 (0)5202 707-548
Fax: +49 (0)5202 707-301
E-Mail: richard.neumann@hanning-hew.com

[2]inIT – Institut für industrielle Informationstechnik
Hochschule Ostwestfalen-Lippe
Langenbruch 6, D-32657 Lemgo
Tel.:+49 (0)5261 702-5803
Fax: +49 (0)5261 702-85803
E-Mail:{alexander.dicks,uwe.moenks,volker.lohweg}@hs-owl.de

Kurzfassung

Die Ermittlung der Zugehörigkeitswerte eines Objekts zu einer oder mehreren Klassen ist oft zweckmäßiger als die scharfe Klassenzuordnung. Ein Klassifikator, der neben einer Klassenzuordnung auch den Zugehörigkeitswert liefert, ist der auf der Fuzzy-Set-Theorie basierende Modified-Fuzzy-Pattern-Classifier (MFPC). Trotz seiner Vorteile hat er die Einschränkung, dass er in seiner ursprünglichen Formulierung multimodale Klassen und Klassen mit einer nichtkonvexen Objektmorphologie nicht adäquat modelliert. Die Klassifikationsleistung ist in solchen Fällen gering. In diesem Beitrag wird ein neues Verfahren vorgestellt, das auf der lokalen MFPC-Klassifikation beruht und aufgrund dessen die Klassifikationsleistung des MFPCs erheblich gesteigert werden kann, was mithilfe von Heuristiken gezeigt wird.

1 Einleitung

Bekannte Verfahren zur Klassifikation, Regression und Clusterung ermitteln die Zuordnung eines Objektes zu einer Klasse, jedoch nicht das Maß der

Zugehörigkeit zu dieser Klasse. Die Zugehörigkeit kann jedoch eine wichtige Information sein, besonders in Anwendungen, in denen die Ergebnisse verschiedener Klassifikatoren miteinander kombiniert werden müssen [1].

Es existieren Verfahren, die in der Lage sind, neben der Klassenzuordnung auch das Maß der Klassenzugehörigkeit zu ermitteln. Stellvertretend für jeweils eine Reihe von Verfahren können folgende beispielhaft genannt werden: das dichtebasierte Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [2] als Vertreter der unüberwachten Verfahren, die Relevance Vector Machine (RVM) [3] als eine Erweiterung der Support Vector Machine (SVM) [4] und der von Bocklisch und Priber entwickelte Fuzzy Pattern Classifier (FPC) [5, 6], als Vertreter der mehrwertigen (Fuzzy) Verfahren. Die von Lohweg [7, 8] vorgeschlagene Erweiterung des FPCs, der Modified-Fuzzy-Pattern-Classifier (MFPC) und dessen Einschränkung bestimmte Datensätze adäquat zu modellieren, bildet den Ausgangspunkt der in diesem Beitrag skizzieren Vorschlags.

Der MFPC weist entsprechend einer vorher definierten Zugehörigkeitsfunktion einem Objekt eine Zugehörigkeit zu jeder Klasse des Datensatzes zu. Dadurch gehört das Objekt gleichzeitig, aber im Idealfall unterschiedlich stark, zu verschiedenen Klassen. Das kommt dem menschlichen Verständnis von der Umwelt näher als scharfe Grenzen und Entscheidungen [9] und kann die Modellierung komplexer Systeme erleichtern. Weitere Eigenschaften des MFPC sind die Möglichkeit der effizienten Implementierung in Hardware und die anschauliche Wissensrepräsentation [10] im Vergleich zu anderen Verfahren, z. B. zu neuronalen Netzen.

Jedoch ist der MFPC in seiner ursprünglichen Formulierung nicht in der Lage, Objekte korrekt zu klassifizieren, falls der zugrundeliegende Datensatz starke Überlappungen der Zugehörigkeitsfunktionen in der Mehrzahl der Dimensionen aufweist (bspw. hervorgerufen durch eine sog. nichtkonvexe Objektmorphologie [11]). Die Einschränkung rührt daher, dass der MFPC zur Beschreibung des Datensatzes nur einen Schwerpunkt der Objekte pro Klasse und Dimension des Merkmalsraums annimmt und den Datensatz konvex modelliert. Diese Annahme entspricht jedoch nicht immer der Realität, wie man am Beispiel des Datensatzes, der bei der Untersuchung des XOR-Problems [12, 13] entsteht, erkennen kann (vgl. Abb. 3). Dieser Datensatz (nachfolgend XOR-Datensatz genannt) weist mehrere Schwerpunkte der Objekte pro Klasse auf und wird dadurch vom MFPC nicht angemessen modelliert.

Um diese Einschränkung aufzuheben, sind verschiedene Verfahren [11, 14] aufgezeigt und getestet worden. In diesem Beitrag werden drei Propositionen eines alternativen Ansatzes vorgestellt, die auf der Segmentierung des Merkmalraums mit einer anschließenden lokalen Klassifikation mittels MFPC basieren. Die Unterschiede zwischen den einzelnen Propositionen bestehen in der Art der Segmentierung des Merkmalraums und der Art der Klassenzuordnung.

Durch die Segmentierung des Merkmalraums wird statt eines globalen Schwerpunkts für jedes Segment ein lokaler Schwerpunkt berechnet. Dadurch können Überlappungen der Zugehörigkeitsfunktionen minimiert und die Klassifikationsleistung des MFPC deutlich verbessert werden. Die drei Propositionen werden mit den Klassifikatoren k-Nearest-Neighbor-Algorithmus (k-NN), Support Vector Machine (SVM), Random Forests, lineare Diskriminanzanalyse (LDA) und Decision trees [4, 15] anhand von Benchmarkdatensätzen verglichen. Dabei wird in Experimenten gezeigt, dass bei Anwendung des vorgestellten Ansatzes die Klassifikationsleistung des MFPC auf das Niveau der etablierten Verfahren angehoben werden kann, bei gleichzeitiger Beibehaltung seiner Vorteile.

Im nachfolgenden Abschnitt wird zunächst der MFPC eingeführt und dessen Einschränkungen aufgezeigt. Daraufhin werden im Abschnitt 2.2 weitere Arbeiten die sich der Verbesserung des MFPC widmen, aufgeführt. Im Abschnitt 3 wird der neue Ansatz beschrieben, dessen Ergebnisse im Abschnitt 4 zusammengefasst sind. Diese werden im Abschnitt 4.1 diskutiert. Eine Zusammenfassung des Beitrags und ein Ausblick werden zuletzt in Abschnitt 5 gegeben.

2 Modified-Fuzzy-Pattern-Classifier

Intuitiv gehören für einen Menschen Objekte umso mehr zu einer Klasse, je ähnlicher deren Eigenschaften sind. Im Merkmalsraum drückt sich diese Ähnlichkeit durch die Nähe eines Objekts zu anderen Objekten aus. So auch Bocklisch in [5]: „Es ist grundsätzlich vernünftig, mit steigender Distanz eine abnehmende Zugehörigkeit anzunehmen“. Eine scharfe Trennung der Klassen ist jedoch in vielen Fällen nicht sinnvoll oder möglich. Außerdem kommen vage formulierte Kriterien dem menschlichen Verständnis der Umwelt viel näher, als scharfe Abgrenzungen, wie Zadeh in [9] beschreibt. Um dem zu begegnen, können auf Grundlage der Fuzzy-Set-Theorie [9] Aussagen über den Grad der Zugehörigkeit eines Objekts zu einer Klasse getroffen werden. Ein Klassifikator, der auf diesem Prinzip beruht, ist der

MFPC, der zur Modellierung des Datensatzes unimodale Potentialfunktionen [16] verwendet. Derartige Potentialfunktionstypen, die zuerst von Aizerman vorgeschlagen wurden [16], sind bereits 1987 von Bocklisch in [5] zur unscharfen Prozessanalyse verwendet worden. Die erste Implementierung solcher Potentialfunktionen auf einem Field Programmable Gate Array (FPGA) erfolgte 2000 durch Eichhorn im Rahmen seiner Dissertation [17].

Bei dem MFPC wird die Zugehörigkeit $\mu(m_i)$ eines Merkmals m_i des Objektes $\mathbf{m} = \{m_1, ..., m_M\}$ (mit M = Anzahl der Merkmale) folgendermaßen ermittelt [1]:

$$\mu(m, \mathbf{p}) = A \cdot 2^{-d(m,\mathbf{p})} \,, \tag{1}$$

mit

$$d(m, \mathbf{p}) = \left(\frac{1}{B} - 1\right) \left(\frac{|m - m_0|}{C}\right)^D \tag{2}$$

Der Parametervektor $\mathbf{p} = (m_0, B, C, D)$ setzt sich aus dem Schwerpunkt ($m_0 = (\frac{m_{\max} - m_{\min}}{2} + m_{\min})$), dem Abstand der Randzugehörigkeit vom Schwerpunkt (C) und der Schärfe der Flanken der Potentialfunktionen (D) zusammen (vgl. Abb. 1). Während der Lernphase des Klassifikators wird der Parameter C mit

$$C = (1 + 2p_{C_\mathrm{E}}) \left(\frac{m_{\max} - m_{\min}}{2}\right) \tag{3}$$

aus der einstellbaren prozentualen Elementarunschärfe $p_{C_\mathrm{E}} \in [0, 1]$ berechnet, der zusammen mit B und A die einstellbaren Parameter bei der Auslegung des Klassifikators bilden. Der MFPC kann effizient implementiert werden, wenn die Parameter $B = 0,5$ und $A = 1$ gewählt werden, da sich dadurch die Gleichung (1) vereinfacht (vgl. dazu [8]). Die äußersten beiden Objekte des Lerndatensatzes sind mit $m_{\max}$ und $m_{\min}$ bezeichnet. Abb. 1 a zeigt exemplarisch die Potentialfunktion mit den Parametern B, C und D, Abb. 1 b,c den Einfluss der Parameter D und p_{C_E}.

Ein Objekt $\mathbf{m}$ gehört umso stärker zur entsprechenden Klasse, je näher es am Schwerpunkt dieser Klasse (m_0) ist. Die Gleichung (1) gilt für den eindimensionalen Fall. Bei mehrdimensionalen Objekten werden die einzelnen eindimensionalen Zugehörigkeitsfunktionen $\mu_1 ... \mu_M$ zu einer gemeinsamen

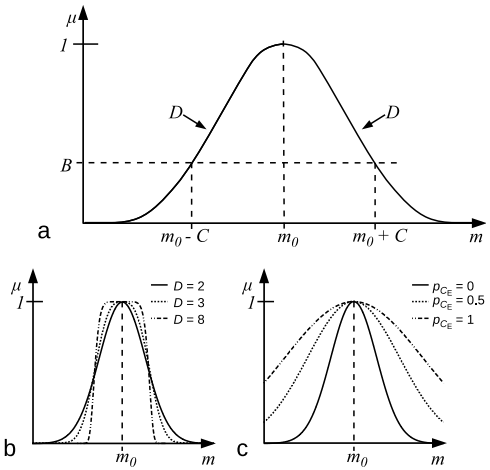


Abb. 1: a) Unimodale Potentialfunktion und b) Einfluss des Parameters D (mit $p_{C_\mathrm{E}} = 0$) sowie c) Einfluss des Parameters p_{C_E} (mit $D = 2$).

kombiniert. Der MFPC-Ansatz sieht dafür eine Aggregation mit Hilfe des geometrischen Mittels vor. Dies führt für M Dimensionen zu [1]:

$$\mu(\mathbf{m}, \mathbf{p}) = A \cdot 2^{-\frac{1}{M}\sum_{i=1}^{M} d_i(m_i, \mathbf{p_i})} \tag{4}$$

Es sei noch angemerkt, dass für die Konstruktion der Zugehörigkeitsfunktion auch andere Funktionen möglich sind. Diese können z. B. asymmetrisch sein, um eine bessere Anpassung des Modells an die Daten zu erreichen. Eine ausführliche Beschreibung und die Einflüsse der Parametervariation lassen sich in [18] finden. Ebenso sich andere Arten der Aggregation möglich, wie in z. B. in [19, 20] aufgezeigt wird.

2.1 Einschränkungen

Die Verteilung der Objekte des Lerndatensatzes im Merkmalsraum kann die Klassifikationsleistung des MFPC stark beeinflussen. Eine anspruchsvolle Aufgabenstellung (im Sinne der Klassifikation durch den MFPC) liegt nach Hempel [11] vor, wenn nichtkonvexe Objektmorphologien oder multimodale Klassen vorliegen. Abb. 2 zeigt Datensätze mit nichtkonvexen Objektmorphologien und Abb. 3 den XOR-Datensatz, bei dem zwei multimodale Klassen im Merkmalsraum vorhanden sind. Beide Fälle können

aus Sicht des MFPC als Abwandlungen des gleichen Problems angesehen werden, da sowohl die Ursache als auch die Auswirkung gleich sind. In beiden Fällen überlappen sich die vom MFPC modellierten Zugehörigkeitsfunktionen in vielen oder sogar allen Dimensionen (vgl. Abb. 3). Die Überlappungen entstehen, da das dem MFPC zugrundeliegende Modell nicht in der Lage ist, solche Klassen adäquat abzubilden.

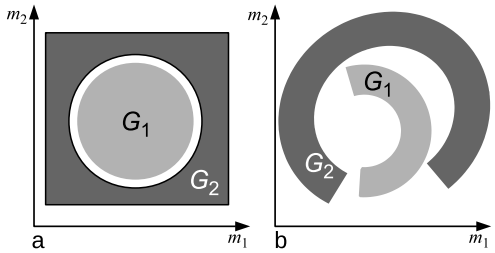


Abb. 2: Zwei mögliche Klassenanordnungen, die vom MFPC nicht adäquat modelliert werden. Mit a) Klasse G_1 vollständig von Klasse G_2 umschlossen und b) teilweise eingeschlossene Klassen.

Auch bei günstigen Verteilungen der Objekte kann der Schwerpunkt bereits durch einen einzelnen Ausreißer in dessen Richtung verschoben werden, was sich direkt auf die Zugehörigkeitswerte (vgl. dazu Gleichung (1)) auswirkt.

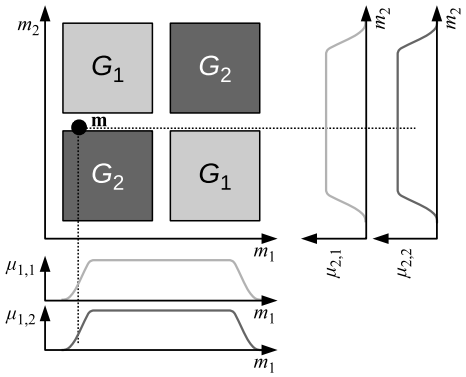


Abb. 3: XOR-Datensatz, als ein Beispiel für einen Datensatz mit multimodalen Klassen.

Die in beiden Fällen entstehenden Zugehörigkeitsfunktionen modellieren die tatsächlichen Verhältnisse oft unzureichend und führen zu nichtintuitiven Ergebnissen und einer geringen Klassifikationsleistung, wie in den

Versuchen, die im Rahmen dieses Beitrags durchgeführt worden sind, gezeigt wird (vgl. dazu Abschnitt 4).

Der Sachverhalt kann beispielhaft anhand des XOR-Datensatzes gezeigt werden, der in der Abb. 3 dargestellt ist. Darin sind dessen Merkmalsraum und die Zugehörigkeitsfunktionen, wie sie vom MFPC-Ansatz modelliert werden, abgebildet. Außerdem ist ein zu klassifizierendes Objekt $\mathbf{m}$ dargestellt, das intuitiv eine höhere Zugehörigkeit zur Klasse G_2 als zur Klasse G_1 haben müsste. Wie jedoch zu erkennen ist, sind die Zugehörigkeiten $\mu_{1,1}$, $\mu_{1,2}$ sowie $\mu_{2,1}$ und $\mu_{2,2}$ gleich groß. Entsprechend Gleichung (4) sind auch die aggregierten Zugehörigkeiten gleich groß, was zu Fehlklassifikationen führt.

2.2 Verwandte Arbeiten

Das Problem der Modellierung solcher Datensätzen wird von Hempel und Bocklisch in [14] aufgegriffen und verfolgt den Ansatz der Segmentierung des Merkmalsraums durch Clusterbildung. Das Verfahren stößt an seine Grenzen aufgrund der großen Anzahl von entstehenden Parametern, wie Hempel in [11] beschreibt: „Bei der Beschreibung nichtkonvexer Objektmengenmorphologien ist dieses Vorgehen je nach Komplexität der Morphologie mit einer großen Anzahl an Klassen und somit Unübersichtlichkeit behaftet - vgl. mit einer Kreis- bzw. Kugelschalengeometrie."

Ein weiterer Ansatz wird von Hempel in [11] vorgeschlagen und basiert auf der Idee von komplementären Fuzzy-Klassen. Unter der Annahme, dass die Klassen im Merkmalsraum komplementär zueinander sind, werden dabei hierarchische Strukturen von Klassifikatoren konstruiert.

Durch die in diesem Beitrag vorgestellte Erweiterung kann der MFPC auch Klassen mit nichtkonvexen Objektmorphologien berücksichtigen und dadurch einen weiteren Bereich an Anwendungsfällen abdecken, unter Beibehaltung seiner Vorteile (vgl. Abschnitt 1).

3 Ansatz

Nachfolgend werden drei unterschiedliche Propositionen vorgestellt und experimentell sowohl mit etablierten Klassifikationsverfahren als auch untereinander verglichen. Jede der Propositionen ist dazu geeignet, die Einschränkungen des MFPC zu reduzieren, wie die Experimente im Abschnitt

4 zeigen. Durch die Aufteilung auf drei Propositionen können unterschiedliche Aspekte bei der Segmentierung und deren Auswirkung auf die Klassifikation beleuchtet werden. Die dabei entstehenden Klassifikatoren werden als Modified-Local-Fuzzy-Pattern-Classifier (MLFPC) bezeichnet.

In allen drei Fällen ist das Ziel, einen Ausschnitt des Lerndatensatzes zu erhalten, dessen Zugehörigkeitsfunktionen sich zumindest in einer Dimension nicht stark überlappen und deshalb durch den MFPC angemessen modelliert werden können.

Proposition 1 (orthogonale Suche und Segmentierung): Angenommen, es existiert ein Datensatz $\mathbf{Z} = \{\mathbf{m}_1, ..., \mathbf{m}_\mathrm{P}\}, \mathbf{m}_p \in \mathbb{R}^M$ (mit $\mathbb{R}^M$ als dem M-dimensionalen Merkmalsraum), dessen Objekte $\mathbf{m}_p$ eine bekannte Klassenzuordnung haben. Die Klassen innerhalb des Datensatzes sind entweder multimodal oder besitzen einer nichtkonvexen Objektmorphologie. Ferner existiert ein zu klassifizierendes Objekt $\mathbf{m}$.

Die Klassifikation durch den MLFPC wird folgendermaßen durchgeführt:

Das Objekt $\mathbf{m}$ wird zunächst mit Hilfe eines MFPC klassifiziert. Der Klassifikator bestimmt die aggregierten Zugehörigkeiten $\mu_n(\mathbf{m}), n \in \{1, ..., N\}$ (mit N als Anzahl der Klassen im Datensatz) des Objekts $\mathbf{m}$. Die einzelnen Zugehörigkeiten können auch den gleichen Wert haben, im Extremfall gilt: $\mu_1 = \mu_2 = ...\mu_N$. Nach der Klassifikation wird die Differenz zwischen der höchsten Zugehörigkeit

$$\mu_{\max} = \max_{n=1}^{N} \mu_n(\mathbf{m})$$

und der zweithöchsten

$$\mu_{\max_2} = \max_{n=1}^{N} (\mu_n(\mathbf{m}) \backslash \mu_{\max})$$

gebildet. Falls die Differenz der beiden Zugehörigkeiten größer oder gleich als ein zuvor definierter Schwellwert $\beta \in [0, 1]$ ist, d.h.

$$\mu_{\max} - \mu_{\max_2} \geq \beta$$

wird die Klassenzuordnung des Objekts anhand der max. Zugehörigkeit durchgeführt, mit

$$G_{\max} = \arg\max_{n=1}^{N} (\mu_n(\mathbf{m})) \,,$$

entsprechend dem bisherigen Vorgehen beim MFPC. Dabei ist $G_{\max}$ die Klasse mit der höchsten Zugehörigkeit. Falls jedoch

$$\mu_{\max} - \mu_{\max_2} < \beta$$

ist, wird dies als ein Indiz für eine nichkonvexe Objektmorphologie gedeutet. Daraufhin wird der Merkmalraum mit Hilfe einer Umgebungssuche um das zu klassifizierende Objekt zerlegt (vgl. Abb. 4). Dabei wird in jeder Dimension des Merkmalraums eine (einstellbare) Menge k bekannter Objekte je Klasse, die in der Nähe zu $\mathbf{m}$ liegen, gesucht. Im zwei- und dreidimensionalen Fall ist der Suchraum ein Viereck bzw. Quader (vgl. Abb. 5).

Der Suchraum wird schrittweise erweitert, bis entweder mindestens für eine Klasse die zuvor definierte Anzahl k bekannter Objekte gefunden wurde (Bedingung B_1), oder der Suchraum gleich dem gesamten Merkmalsraum ist (Bedingung B_2). Die Schrittweite s bei jeder Ausweitung des Suchraums für jede Dimension i des Merkmalraums bestimmt sich mit

$$s = \frac{(m_{\max} - m_{\min})}{q}.$$

Dabei ist $q \in \mathbb{R}$ ein einstellbarer Parameter, der bei der Auslegung des Systems einen Freiheitsgrad bietet.

Sobald k Objekte gefunden wurden, wird eine erneute MFPC-Klassifikation des Objektes $\mathbf{m}$ anhand der k bekannten Objekte durchgeführt, die als Ergebnis die aggregierte, lokale Zugehörigkeiten $\mu_{n,\mathrm{L}}$ liefert. Da die k bekannten Objekte nur einen Ausschnitt des Datensatzes darstellen, wird dieser Schritt als lokale MFPC-Klassifikation bezeichnet. Die Klassenzuordnung wird in diesem Fall mit

$$G_{\max} = \arg\max_{n=1}^{N}(\mu_{\mathrm{n,L}}(\mathbf{m}))$$

aus der lokalen MFPC-Zugehörigkeit $\mu_{\mathrm{n,L}}$ mit dem maximalen Wert ermittelt. Falls die beiden höchsten Zugehörigkeiten gleich sind (Bedingung B_3) kann keine Klasse ermittelt werden und das Objekt wird der Rückweisungsklasse G_{r}, d.h. einer Klasse, für dessen Objekte keine Zuordnung zu einer der bekannten Klassen angegeben werden kann, zugeordnet.

Zusammenfassend kann geschrieben werden:

$$G_{\max} = \begin{cases} \arg\max_{n=1}^{N}(\mu_{\mathrm{n,L}}(\mathbf{m}), & \text{falls} \quad \mu_{\max} - \mu_{\max_2} < \beta \\ \arg\max_{n=1}^{N}(\mu_n(\mathbf{m})), & \text{falls} \quad \mu_{\max} - \mu_{\max_2} \geq \beta \\ G_{\mathrm{r}} & \text{falls} \quad B_2 \vee B_3 \end{cases}$$

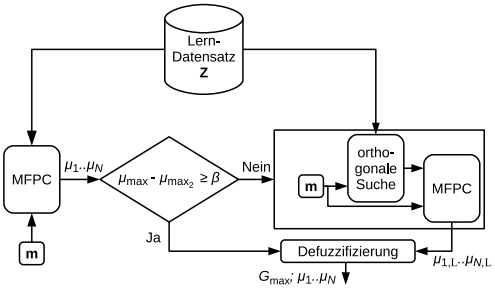


Abb. 4: Ablauf der Klassifikation bei der orthogonalen Suche.

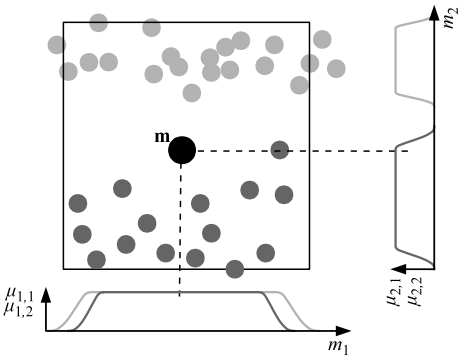


Abb. 5: Ausschnitt aus dem XOR-Datensatz mit möglichen Zugehörigkeitsfunktionen. Dadurch, dass lediglich ein Ausschnitt des Datensatzes betrachet wird, überlappen sich die Zugehörigkeitsfunktionen nicht in allen Dimensionen.

■

Proposition 2 (k-NN basierte Suche und Segmentierung): Es gelten die gleichen Bedingungen, wie in Proposition 1.

Die Vorgehensweise entspricht ebenfalls der in Proposition 1 erläuterten, jedoch mit dem Unterschied, dass die für die nachfolgende lokale MFPC-Klassifikation benötigten k Objekte durch eine k-NN-Suche [4] statt einer Umgebungssuche bestimmt werden (vgl. Abb. 6).

Um die Vergleichbarkeit zu gewährleisten, ist bei den nachfolgenden Experimenten (vgl. Abschnitt 4) der euklidische Abstand als Abstandmaß bei der k-NN-Suche festgelegt worden.

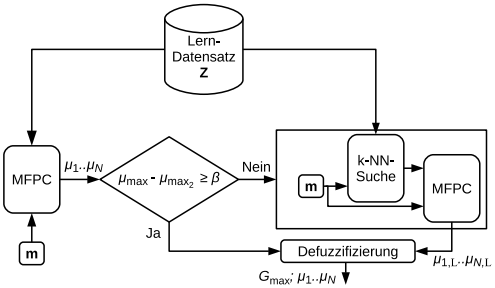


Abb. 6: Ablauf der Klassifikation bei der k-NN basierten Suche.

■

Proposition 3 (Kombination der MFPC-Zugehörigkeit mit einer k-NN-Klassenzuordnung): Es gelten die gleichen Bedingungen, wie in Proposition 1.

Der Unterschied zu den beiden vorhergehenden Propositionen besteht darin, dass in dem Fall, dass nach einer globalen MFPC-Klassifikation die Differenz der höchsten und der zweithöchsten Zugehörigkeiten

$$\mu_{\max} - \mu_{\max_2} < \beta$$

ist, die Klasse G_k des Objekts $\mathbf{m}$ mithilfe einer k-NN-Klassifikation bestimmt wird. Daraufhin wird die entsprechende globale Klassenzugehörigkeit $\mu_n(\mathbf{m})$ der Klasse G_k zugewiesen (vgl. Abb. 7). Die Zugehörigkeiten zu allen anderen Klassen werden verworfen.

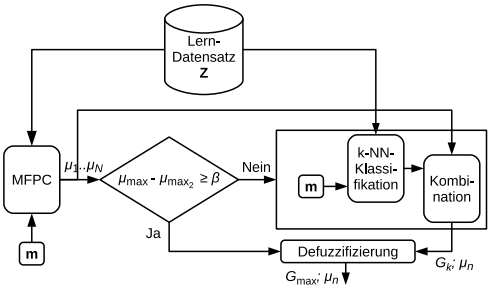


Abb. 7: Ablauf der Klassifikation bei der k-NN basierten Klassenzuordnung.

Um die Vergleichbarkeit zu gewährleisten, ist bei den nachfolgenden Experimenten (vgl. Abschnitt 4) der euklidische Abstand als Abstandmaß bei der k-NN-Klassifikation festgelegt worden.

■

4 Evaluierung

Nachfolgend wird der vorgeschlagene Ansatz mit anderen Klassifikatoren verglichen. Die Verfahren werden anhand der Vorhersagegenauigkeit $A = \frac{r_\mathrm{p}+r_\mathrm{n}}{n}$ und der Standardabweichung σ verglichen, die mit Hilfe einer 10-fachen Kreuzvalidierung mit nachfolgender Mittlung ermittelt wurden. Dabei ist r_p die Anzahl der richtig-positiv klassifizierten Objekte, r_n die Anzahl der richtig-negativ klassifizierten Objekte und n die Anzahl aller getesteten Objekte [21]. Laut Dietterich [22] ist die x-fache Kreuzvalidierung eins der am besten geeigneten Verfahren, um Klassifikatoren zu vergleichen. Durch die Aufteilung in zehn Abschnitte konnte sichergestellt werden, dass in den einzelnen Stichproben auch bei kleinen Datensätzen, z. B. dem Iris-Datensatz [23], noch ausreichend Objekte zur Verfügung standen.

Zum Vergleich wurden folgende Klassifikatoren verwendet:

Naïve Bayes, SVM, Random Forests, Decision Trees, k-NN und LDA [4, 15]. Es wurden jeweils die Implementierungen verwendet, die zum Programmumfang von MATLAB [24] gehören.

Die Parameter der Klassifikatoren wurden wie folgt festgelegt (nur die wesentlichen Parameter sind aufgelistet):

- Bei dem Bayes-Klassifikator wurden zwei Klassifikatoren verwendet, die unterschiedliche Annahmen bez. der Verteilung treffen. Der erste Klassifikator nimmt eine Normalverteilung an, der zweite verwendet einen Kerndichteschätzer (Gaußkern). Das jeweils bessere Ergebnis der beiden Klassifikatoren wurde gewählt.
- Beim MFPC- und MLFPC-Ansatz wurde $D = 8$ und $p_{C_\mathrm{E}} = 0.2$ gewählt.
- Die SVM-Klassifikation basiert auf der Wahl des besten Ergebnisses von vier SVMs mit einem linearen-, polynominellen- (3. Ordnung), quadratischen- und einem Radiale Basisfunktion-Kernel. Bei der Klassifikation von Datensätzen mit mehr als zwei Klassen wurde die „eine gegen alle“ Methode verwendet.

- Beim MLFPC mit k-NN-Klassifikator und dem MLFPC mit k-NN-Suche wurde die euklidische Distanz als Abstandsmaß und 5 Nachbarn ($k = 5$) festgelegt.
- Der Random-Forests-Klassifikator besteht aus 500 Entscheidungsbäumen.

Es ist selbstverständlich, dass durch eine Optimierung der Klassifikatorparameter bessere Ergebnisse auf den einzelnen Datensätzen erreicht werden können. Im Rahmen dieser Arbeit wurde darauf jedoch verzichtet, da die Optimierung unter Umständen bei jedem Datensatz individuell erfolgen müsste, was den Vergleich erschweren würde.

Tabelle 1 zeigt die Ergebnisse für die künstlich erzeugten Datensätze. Es wurden folgende Datensätze verwendet: ein Datensatz, der aus einem Kreis (Klasse G_1) in einem Viereck (Klasse G_2) (vgl. Abb. 2 a) besteht, der XOR-Datensatz und ein Datensatz, bei dem eine Klasse eine andere teilweise umschließt (vgl. Abb. 2 b). Der dritte Datensatz wird nachfolgend als Swiss-Roll-Datensatz bezeichnet und basierend auf einem Vorschlag aus [25]. Wie aus der Tabelle 1 ersichtlich, weist der MFPC bei den künstliche erzeugten Datensätzen eine der schlechtesten Klassifikationsleistungen auf. Demgegenüber zeigen die drei neuen Klassifikatoren (MLFPC-1 ... MLFPC-3) eine deutlich höhere Klassifikationsleistung.

Die künstliche erzeugten Datensätze sind zwar für die Untersuchung bestimmter Eigenschaften, in diesem Fall die Reaktion auf stark überlappende Zugehörigkeitsfunktionen, angebracht, spiegeln jedoch nicht die Realität wieder. Deshalb wurden die neuen Ansätze zusätzlich mit realen Datensätzen untersucht. Folgende Datensätze wurden verwendet: Iris, Wine, Wine Quality, Banknote, Haberman's Survival und Handwritten Digits der UCI Machine Learning Data Repository [23].

Aus den Tabellen 2 und 3 ist ersichtlich, dass auch für diese Datensätze die Klassifikationsrate auf dem Niveau der Benchmarkverfahren liegt.

Tabelle 1: Vorhersagegenauigkeit und Standardabweichung für künstlich erzeugte Datensätze.

	Kreis-Viereck		XOR		SwissRoll	
	A	σ	A	σ	A	σ
Bayes	0.96	0.015	0.507	0.04	0.99	0.008
SVM	0.99	0.01	1	0	0.97	0.07
Rand. Forest	0.97	0.015	1	0	0.99	0.008
Trees	0.96	0.02	1	0	0.99	0.01
k-NN-5	0.98	0.008	1	0	0.99	0.01
LDA	0.53	0.06	0.48	0.016	0.75	0.03
MFPC	0.5	0.003	0.48	0.019	0.5	0.005
MLFPC-1	0.98	0.011	1	0	0.99	0.007
MLFPC-2	0.98	0.014	0.99	0.001	0.99	0.01
MLFPC-3	0.98	0.008	1	0	0.99	0.008

Tabelle 2: Vorhersagegenauigkeit und Standardabweichung für natürliche Datensätze.

	Iris		Wine		Wine Q.	
	A	σ	A	σ	A	σ
Bayes	0.96	0.05	0.97	0.03	0.99	0.004
SVM	0.98	0.05	0.99	0.02	0.99	0.002
Rand. Forest	0.96	0.05	0.98	0.04	0.99	0.004
Trees	0.95	0.05	0.9	0.05	0.98	0.004
k-NN-5	0.96	0.05	0.68	0.1	0.94	0.08
LDA	0.58	0.03	0.98	0.03	0.99	0.002
MFPC	0.94	0.06	0.93	0.06	0.86	0.036
MLFPC-1	0.96	0.05	0.99	0.03	0.99	0.004
MLFPC-2	0.95	0.05	0.84	0.08	0.8	0.016
MLFPC-3	0.96	0.05	0.88	0.06	0.94	0.008

Tabelle 3: Vorhersagegenauigkeit und Standardabweichung für natürliche Datensätze.

	Bank.		Haber.		Digits	
	A	σ	A	σ	A	σ
Bayes	0.89	0.034	0.46	0.18	0.95	0.097
SVM	0.99	0.024	0.73	0.08	0.99	0.015
Rand. Forest	0.98	0.012	0.74	0.05	1	0.001
Trees	0.97	0.018	0.68	0.06	1	0
k-NN-5	0.97	0.016	0.73	0.06	0.99	0.003
LDA	0.92	0.02	0.75	0.043	0.85	0.007
MFPC	0.78	0.04	0.55	0.07	0.1	0
MLFPC-1	0.98	0.01	0.6	0.11	1	0
MLFPC-2	0.92	0.026	0.7	0.065	1	0
MLFPC-3	0.97	0.016	0.73	0.064	1	0

4.1 Diskussion der Ergebnisse

Trotz der erreichten Verbesserungen (sowohl der Klassifikation als auch des Modells) stellt sich die Frage, ob und in welchen Fällen die Segmentierung des Merkmalsraums zulässig ist.

Zur Beantwortung dieser Frage kann folgende Überlegung angestellt werden:

Es sei angenommen, Ähnlichkeit drückt sich durch Nähe von Objekten zu anderen Objekten aus und jedes Objekt des Datensatzes spannt einen lokalen Zugehörigkeitsraum um sich herum auf (sog. elementare Zugehörigkeitsfunktion). Zudem sei die Grundgesamtheit des Datensatzes unbekannt.

In den Bereichen des Merkmalsraums, an denen sich die elementaren Zugehörigkeitsfunktionen überschneiden oder zumindest berühren, kann eine gemeinsame Zugehörigkeitsfunktion konstruiert werden. Ferner kann davon ausgegangen werden, dass neue zu klassifizierende Objekte, die sich in diesem zusammenhängenden Bereich befinden, ähnlich den schon vorhandenen sind und zu deren Klasse gehören [5, 9].

Die Bereiche, in denen sich die lokalen Zugehörigkeitsfunktionen weder überschneiden noch berühren (nachfolgend Lücken genannt), dürfen jedoch nicht von dieser gemeinsamen Zugehörigkeitsfunktion überspannt werden, auch nicht in den Fällen, in denen die lokalen Zugehörigkeitsfunktionen um die Lücke herum angeordnet sind (vgl. Abb. 2). Solche Lücken werden jedoch vom bisherigen MFPC-Ansatz nicht berücksichtigt, da es zu allgemein bzw. nicht ausreichend spezifisch ist.

Folgender Punkt muss mitberücksichtigt werden: Das Modell, das dem MFPC zugrunde liegt, stützt sich bei der Berechnung der Lage des Schwerpunkts m_0 lediglich auf die zwei Objekte $m_{\max}$ und $m_{\min}$, wodurch ein einzelner Ausreißer den Schwerpunkt verschieben kann. Da die Grundgesamtheit nicht bekannt ist, kann jedoch nicht mit Sicherheit behauptet werden, dass es sich um einen Ausreißer handelt.

Die lokale MFPC-Klassifikation stellt nun eine Spezifizierung des Modells dar, indem es die elementaren Zugehörigkeiten bzw. die Nähe zu den in der Umgebung befindlichen Objekten stärker berücksichtigt, vergleichbar mit der Wahl eines Ausschnitts aus einem Bild, bei dem mehr Details sichtbar werden, jedoch der Gesamtüberblick verloren geht. Da der Gesamtüberblick aufgrund der Unkenntnis der Grundgesamtheit in den meisten Fällen ohnehin nicht sicher ist und zusätzlich verzerrt sein kann (aufgrund der Unfähigkeit des Modells, nichtkonvexe Objektmorphologien abzubilden), stellt der lokale Ansatz eine Verbesserung dar.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde eine Erweiterung des MFPCs vorgeschlagen, der auf der Segmentierung des Merkmalsraums mit nachfolgender Klassifikation durch den MFPC basiert. Durch die Segmentierung können Datensätze mit einer nichtkonvexen Objektmorphologie gegenüber FPC-basierten Ansätzen besser modelliert und in Folge dessen auch besser klassifiziert werden.

Es wurden drei Varianten des neuen Verfahrens untereinander und mit etablierten Klassifikationsverfahren anhand von künstlichen und natürlichen Datensätzen verglichen. Bei den verwendeten Datensätzen zeigten alle drei eine vergleichbare Klassifikationsleistung, die auf dem Niveau der etablierten Klassifikationsverfahren liegen. Demgegenüber weist der MFPC in der ursprünglichen Formulierung eine deutlich schlechtere Klassifikationsleistung auf.

Während der Arbeit an dem vorgestelltem Ansatz sind weitere interessante Aspekte aufgetreten. So ist die Eingliederung des vorgestellten Ansatzes in die Fuzzy-Set-Theorie interessant und muss untersucht werden. Ferner stellt sich die Frage nach der optimalen Größe des Ausschnittes und der Möglichkeit der Kombination mit dem von Hempel [11] entwickelten Verfahrens, was in zukünftigen Arbeiten untersucht werden sollte.

Literatur

[1] Lohweg, V.; Mönks, U.: Fuzzy-Pattern-Classifier based Sensor Fusion for Machine Conditioning. In: *Sensor Fusion and its Applications* (Thomas, C., Hg.). SCIYO. 2010.

[2] Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: *Proc. of 2nd International Conference on Knowledge Discovery and*, S. 226–231. 1996.

[3] Tipping, M. E.: Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research* 1 (2001), S. 211–244.

[4] Bishop, C. M.: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc. ISBN 0387310738. 2006.

[5] Bocklisch, S.: *Prozessanalyse mit unscharfen Verfahren*. Verlag Technik. 1987.

[6] Priber, U.: *Ein Verfahren zur Merkmalsreduktion für unscharfe Klassifikatoren*. 1989.

[7] Lohweg, V.: *On Effective Implementation of Adaptive Spectral Transforms in Application Specific Integrated Circuits*. Dissertation, Techn. Univ. Chemnitz. 2003.

[8] Lohweg, V.; Diederichs, C.; Müller, D.: Algorithms for Hardware-Based Pattern Recognition. *EURASIP J. Adv. Sig. Proc.* 2004 (2004) 12, S. 1912–1920.

[9] Zadeh, L. A.: Fuzzy Sets. *Information and Control* 8 (1965), S. 338–353.

[10] Li, R.; Lohweg, V.: Fuzzy pattern classification tuning by parameter learning based on fusion concept. In: *FUSION*. IEEE. ISBN 978-3-00-024883-2. 2008.

[11] Hempel, A.: *Netzorientierte Fuzzy-Pattern-Klassifikation nichtkonvexer Objektmengenmorphologien*. Universitätsverlag Chemnitz. ISBN 9783941003460. 2011.

[12] Minsky, M. L.; Papert, S.: *Perceptrons: An Introduction to Computational Geometry*. Cambridge Mass.: MIT Press, expanded ed. Aufl. ISBN 0262631113. 1988.

[13] Coetzee, F.; Stonick, V. L.: 488 Solutions to the XOR Problem. In: *NIPS* (Mozer, M.; Jordan, M. I.; Petsche, T., Hg.), S. 410–416. MIT Press. 1996.

[14] Hempel, A.; Bocklisch, S. F.: Hierarchical Modelling of Data Inherent Structures Using Networks of Fuzzy Classifiers. In: *UKSim* (Al-Dabass, D.; Orsoni, A.; Brentnall, A.; Abraham, A.; Zobel, R. N., Hg.), S. 230–235. IEEE. 2008.

[15] Breiman, L.; Friedman, J. H.; Olshen, R. A.; Stone, C. J.: *Classification and Regression Trees*. Wadsworth International Group. 1984.

[16] Aizerman, M. A.; Braverman, E. A.; Rozonoer, L.: Theoretical foundations of the potential function method in pattern recognition learning. In: *Automation and Remote Control,*, Nr. 25 in Automation and Remote Control,, S. 821–837. 1964.

[17] Eichhorn, K.: *Entwurf und Anwendung von ASICs für musterbasierte Fuzzy-Klassifikationsverfahren*. Dissertation, Techn. Univ. Chemnitz. 2000.

[18] Mönks, U.; Petker, D.; Lohweg, V.: Fuzzy-Pattern-Classifier Training with Small Data Sets. In: *IPMU (1)* (Hüllermeier, E.; Kruse, R.; Hoffmann, F., Hg.), Bd. 80 von *Communications in Computer and Information Science*, S. 426–435. Springer. 2010.

[19] Larsen, H. L.: Efficient Andness-Directed Importance Weighted Averaging Operators. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 11 (2003) Supplement-1, S. 67–82.

[20] Moenks, U.; Lohweg, V.; Larsen, H. L.: Aggregation Operator Based Fuzzy Pattern Classifier Design. *KI 2009 Workshop* (2009).

[21] Alpaydın, E.: *Maschinelles Lernen*. München: Oldenbourg. ISBN 9783486581140. 2008.

[22] Dietterich, T. G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation* 10 (1998), S. 1895–1923.

[23] Bache, K.; Lichman, M.: UCI Machine Learning Repository. URL `http://archive.ics.uci.edu/ml`. 2013.

[24] MATLAB: *version 8.3.0 (R2014a)*. Natick, Massachusetts: The MathWorks Inc. 2014.

[25] Tenenbaum, J. B.; de Silva, V.; Langford, J. C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290 (2000) 5500, S. 2319.

Einsatz künstlicher neuronaler Netze zur Modellierung des Raumklimas in der präventiven Konservierung unter expliziter Berücksichtigung auftretender Störgrößen

Simon Harasty, Steven Lambeck, Tarek Aissa

Hochschule Fulda, Fachbereich Elektrotechnik und Informationstechnik
Marquardstr. 35, 36039 Fulda
Tel.: (0661) 9640-557 und -570
Fax: (0661) 9640-559
Email: {Simon.Harasty; Steven.Lambeck, Tarek.Aissa}@et.hs-fulda.de

1 Einführung

In Anwendungen der präventiven Konservierung ist es notwendig, die klimatischen Bedingungen im direkten Umfeld von schützenswerten Kulturgütern zum Zweck ihres dauerhaften Erhalts zu regulieren. Hierzu sollen neben der Einhaltung konservatorisch zulässiger Bereiche von Temperatur und Luftfeuchtigkeit vor allem kurzfristige Schwankungen dieser Größen, die zu Feuchte- und Energietransporten in und aus den Exponaten führen, vermieden werden (siehe [1]).

Während bei zu geringer Luftfeuchtigkeit in Kombination mit zu hoher Temperatur eine Austrocknung der Objekte und bei zu hoher Luftfeuchtigkeit in Kombination mit zu niedriger Temperatur der Zerfall durch biologische Angriffe wie Schimmel- oder Schwammbildung droht, sind bereits durch vergleichsweise geringe stetige Änderungen in Temperatur oder Luftfeuchtigkeit durch ablaufende Sorptionseffekte physikalische Schädigungen an den Kulturgütern zu beobachten. Zum Erhalt des ausgestellten Kulturgutes innerhalb der Gebäude sollte also nicht nur ein an die jeweiligen Materialien angepasstes Klima vorherrschen, sondern vielmehr zusätzlich eine möglichst geringe Änderung der einzelnen klimatischen Größen angestrebt werden (siehe [2]).

Um neben der Einhaltung eines als akzeptabel geltenden Klimabereiches (siehe [3], Abbildung 1) Schwankungen innerhalb dieses Bereiches mit geeigneten regelungstechnischen Ansätzen gering zu halten, ist eine möglichst genaue Bestimmung des klimatischen Verhaltens des Raumes notwendig. Vor allem die verwendeten Bausubstanzen und die im Raum selbst befindlichen Materialien und deren Eigenschaften beeinflussen Energie- und Feuchtigkeitsaufnahme, Speicherverhalten und Abgabe. Das klimatische Verhalten eines Raumes hängt daher von einer Vielzahl von

Variablen ab, welche nicht ohne äußerst aufwendige Verfahren (Materialanalysen, Raummodellbildung) bestimmt werden können.

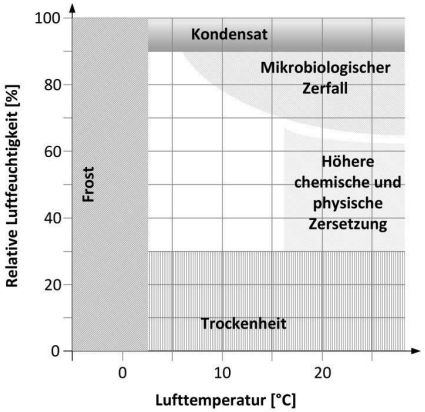


Abbildung 1:Bereich des akzeptablen Innenklimas

Als Applikationsobjekt wird das in verschiedenen Projekten der Hochschule Fulda bereits untersuchte Schloss Fasanerie der Kulturstiftung des Hauses Hessen (www.schloss-fasanerie.de) als typisches Objekt für Problemstellungen der präventiven Konservierung herangezogen. Die für das Training des Netzes verwendeten Daten wurden in Schloss Fasanerie aufgezeichnet. Die typischen Probleme der präventiven Konservierung sind hier in Form eines historischen Gebäudes mit Museumsbetrieb präsent. Für den gewünschten Erhalt des historischen Baubestandes sind vergleichsweise schlechte Grundlagen an Baubestand durch Jahrhunderte altes Sandsteinmauerwerk ohne hohe Isolationseffekte, dafür mit hohen Speicherpotentialen vorhanden. Durch den in den Sommermonaten stattfindenden Museumsbetrieb sind ständige zusätzliche Luftwechsel, Eintrag von Feuchtigkeit und geringe Abgabe von Wärme als zusätzliche Störgrößen vorhanden.

Das Verhalten von Temperatur und Luftfeuchtigkeit für einen Raum hängt von diversen, oft nicht einfach zu ermittelnden und teilweise unbekannten Größen ab (Baustoffe, im Raum befindliche Materialien, natürliche Luftwechselrate). In den vergangenen Jahren haben sich modellgestützte prädiktive Verfahren für die Regelung des Raumklimas etabliert (siehe [4]). Ein zusätzliches Problem der Modellbildung stellen nicht messbare Störgrößen (wie Besucherverkehr, willkürliche Lüftung durch öffnen von Fenstern und Türen) dar. Als Verfahren kann zur Modelbildung die Verwendung eines künstlichen neuronalen Netzes (KNN) verwendet werden, welches durch eine Teilrekursion die vorhandenen Speichereffekte

abbilden kann. Durch bereits vorhandene Messungen innerhalb des Gebäudes mit und ohne den Einfluss gezielten Lüftens durch Ventilatoren, ist eine nutzbare Datenbasis bereits vorhanden

Zur Umgehung der Problematik nicht messbarer Störgrößen, wird der Ansatz verfolgt, die Auswirkungen der Störgrößen auf das Prozessverhalten durch ein geändertes Modell abzubilden. Hierfür wird für jedes Prozessverhalten eine angepasste Gewichtung des ausgewählten KNNs verwendet (siehe Abbildung 2). In Anlehnung an das biologische Vorbild werden die Gewichtungen hier als Neurotransmitter, bzw. die Gewichtungen des gesamten KNN als Neurotransmitterstadium bezeichnet.

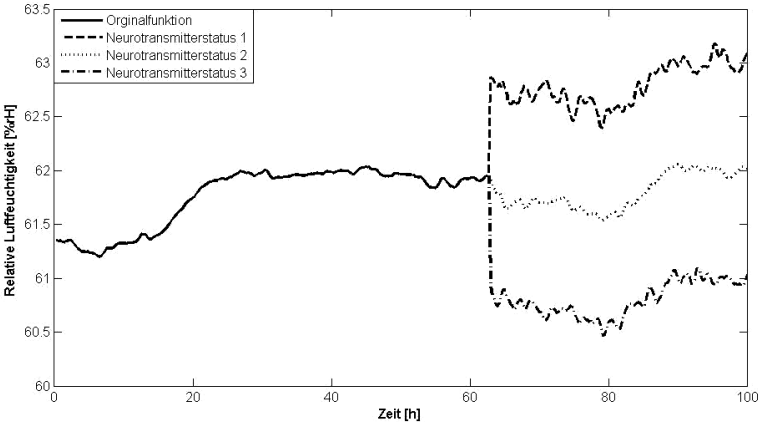


Abbildung 2:Prognose unterschiedlicher Neurotransmitterstadien

Um einen Initialisierungszustand des KNN festzulegen werden gesammelte Daten zu Zeitpunkten ohne Beeinflussung durch zusätzliche Lüftung und möglichst ohne Störgrößeneffekte genutzt und die Dimension und Größe des KNN experimentell zu bestimmen Das Verhalten innerhalb des Netzes wird durch die Anpassung der Gewichtungen und Aktivierungsfunktionen trainiert. Datensätze mit bereits bekannten Störgrößen, hier beispielsweise Tage mit hohem Besucherverkehr, werden daraufhin zur Festlegung einer neuen Gewichtung genutzt. Durch den Vergleich der Ergebnisse verschiedener Neurotransmitterstadien kann so das aktuell zu verwendende Stadium definiert werden. Neue Störgrößen können ebenfalls als neue Neurotransmitterstadien erkannt werden, indem ein maximaler Fehler des Modells als Grenze für das Training eines neuen Verhaltens genutzt wird (siehe Abbildung 3). Innerhalb dieses Beitrags soll der Einfluss dieser Neurotransmitterstadien zur Erkennung und Berücksichtigung von nicht messbaren Störgrößen untersucht werden.

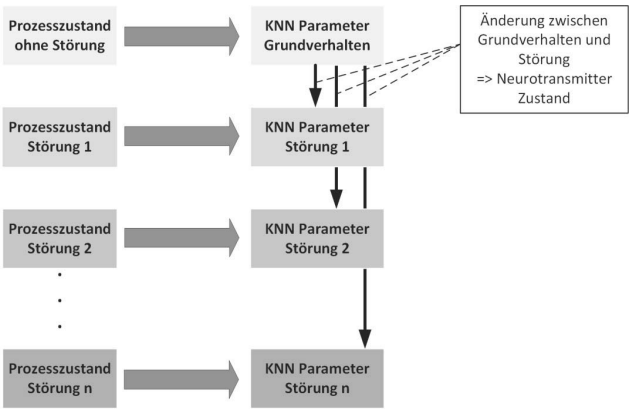


Abbildung 3: Erstellung von Neurotransmitterzuständen

2 Realisierung des KNN

Als Auswahl für das KNN wurde zuerst der darzustellende Prozess analysiert um eine Sicherstellung der nötigen Dynamik zu realisieren. Zur Wiedergabe der vorhandenen Speichereffekte erscheint die Auswahl eines teilrekursiven Netzes eines Multilayer Perceptrons (MLP) zur Darstellung des Systemverhaltens, insbesondere der Speichereffekte als sinnvoll, siehe [5].

2.1 Größe und Typ

Die sinnvoll zu wählende Größe (Anzahl Neuronen und Layer) ist von den Freiheitsgraden, sprich der Ordnung und dem Grad der darzustellenden Funktion abhängig. Die Definition der darzustellenden Funktion als keine paarweise orthogonale Funktion die stetig differenzierbar ist, ergibt eine Anforderung von mindestens drei Neuronenschichten (siehe [6]). Ausgehend von einem weitestgehend unbekannten Prozess, für den solch ein datengetriebener Ansatz sinnvoll erscheint, kann nach der Untersuchung der Dynamik durch den Test verschiedener Dimensionen die benötigte Größe festgelegt werden (siehe [7]).

Eher kritisch sollte allerdings wiederum der Datenbasis gegenüber gestanden werden, da eine Identifikation des Prozesses lediglich im Rahmen der Genauigkeit dieser Basis möglich ist. In dem hier betrachteten Fall der Klimaregelung stehen Daten, die in einem Zeitintervall von 15 min gesammelt wurden, zur Verfügung, was eine sichere Identifikation schnellerer Effekte, wie den kurzzeitigen Anstieg einer Störgröße, logisch ausschließt. Zur Regelung des Klimas innerhalb des betrachteten Raumes

ist eine Vorhersage von einer Stunde, sprich vier Datenwerten erwünscht, da diese genügend Zeit zur Einflussnahme, mit den zur Verfügung stehenden Aktoren bietet. Durch diese Betrachtung der relevanten Zeitbereiche und aus bekanntem Systemverhalten kann zwar die mindestens erforderliche Dimension des Netzes bestimmt werden, dennoch muss durch Tests die konkrete Anzahl an Neuronen und Schichten bestimmt werden (siehe [6, 7]).

Durch die Faktoren der zeitlich relevanten Bereiche und der Dynamik konnte das Netz mit 3 versteckten Schichten definiert und getestet werden. Zur Vorhersage des Prozessverhaltens inklusive der beschriebenen Speichereffekte wird hier eine Kombination aus MLP und rückgekoppeltem Netz verwendet (siehe Abbildung 4).

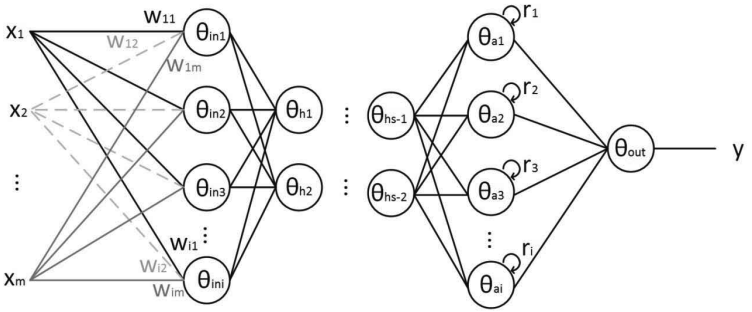


Abbildung 4: Teilrekursives MLP

Hierbei wurde eine angepasste Ausgabe durch eine erweiterte Aktivierungsfunktion (siehe Gl. 4) in der ebenfalls zur Rückkopplung verwendeten letzten Neuronenschicht, hier Adaptionsschicht, vorgesehen. Die Adaption bezieht sich hierbei auf die vorhandenen Speichereffekte des Prozesses.

Die Anwendung eines MLP läuft sukzessive durch Multiplikation der Eingangssignale im Vektor $X = [x_1, x_2, \dots, x_n]$ mit den Gewichtungsfaktoren W_i zu Vektor N_i. Dies erfolgt für jeden Neurotransmitterstatus λ.

$$\mathrm{W}_{i\lambda} = \begin{pmatrix} W_{11} & \cdots & W_{1n} \\ \vdots & \ddots & \vdots \\ W_{i1} & \cdots & W_{in} \end{pmatrix}_{\lambda} \tag{1}$$

$$N_{i\lambda} = X * W_{i\lambda} \tag{2}$$

Mit diesem wird über eine Sigmoidfunktion der Ausgabewert jedes Neurons berechnet. Hierfür wird folgende logistische Funktion verwendet (siehe [5]):

$$f_m(\theta_m, N_{i_m}) = \frac{1}{1 + e^{-(N_{i_m} - \theta_m)}} \tag{3}$$

Lediglich die Adaptionsschicht besitzt für ein breiteres Ausgabefeld eine bipolare sigmoide Funktion, realisiert durch einen tangens hyperbolicus und der Rückgekoppelten Funktion (siehe [5]):

$$f_{am}(\theta_{am}, N_{a_m}) = \frac{2}{1 + e^{-2(N_{a_m} - \theta_{am})}} - 1 + r_m * f_{am}(t-1) \tag{4}$$

2.2 Lernalgorithmus

Die Lernfunktion zum Training des hier vorgestellten Netzes ist eine leichte Anpassung des Backpropagation Algorithmus mit Berücksichtigung der Rückkopplung und der Adaption der Aktivierungsfunktion. Hierfür muss lediglich im ersten Schritt der letzte rückgeführte Wert mit neuer Adaption von sämtlichen Neuronen abgezogen werden und der jeweilige Eingang der Neuronen mittels Rückrechnung der aktualisierten Aktivierungsfunktion verrechnet werden.

1. Berechnung Fehler e
2. Adaptionsschicht Anpassen der Rückführung
   $r_{neu} = r_{alt} * \eta_r$
3. Ausgabefunktion mit neuer Rückkopplung
   $f_{au} = f_{au_{alt}} * (\eta_r - 1)$
4. Anpassung der Gewichte über Backpropagation unter Nutzung der neuen Rückführung als Ausgang

Sowohl für die Adationsschicht wie für die versteckten Neuronen gilt die aus dem Backpropagationalgotithmus genutzte partielle Ableitung zur Bestimmung der neuen Gewichtung für Neuron der Schicht u bei Lernrate η (siehe [5]), wobei e die Fehlerfunktion und δ_i der Fehler eines einzelnen Neurons ist.:

$$\Delta\overrightarrow{w_u} = -\frac{\eta}{2}\vec{\nabla}_{\vec{w}_u} e \tag{5}$$

Die Ableitung der Sigmoidfunktion kann hier für die der versteckten Neuronen nachfolgenden Schicht s verwendet werden um über die zuvor berechneten nachfolgenden Neuronen die Änderung der Gewichtung über die vorherige Ausgabefunktion des Neurons f_u zu bestimmen (siehe [5, 8]):

$$\Delta\overrightarrow{w_u} = \eta \left(\sum_{s=1}^{u_{k+1}} \delta_s \, w_s \right) f_u(1 - f_u) N_{iu} \tag{6}$$

Und für die Adaptionsschicht mit Tangens hyperbolicus:

$$\Delta\overrightarrow{w_u} = \eta \left(\sum_{s=1}^{u_{k+1}} \delta_s \, w_s \right) (1 + f_{au})(1 - f_{au}) N_{iu} \tag{7}$$

Um eine korrekte Adaption durchzuführen, ist durch die Rückführung immer der letzte Wert der Neuronen der Adaptionsschicht $f_{am}(t-1)$ zu berücksichtigen. Allerdings ist die Verwendung von Startwerten nicht einfach möglich, so das Zufallswerte verwendet werden.

Daraufhin kann der Standard Backpropagation Algorithmus durch die Änderung der Gewichte nach der Ableitung des Fehlers über die jeweilige Gewichtung angewandt werden.

2.3 Neurotransmitterstadien

Bei Störungen in komplexen Prozessen, die keinen messbaren Einfluss auf die Eingänge eines Netzes haben, entspricht das geänderte Prozessverhalten einer anderen Gewichtung des KNN. Wie am biologischen Vorbild zu sehen, werden hier analog zur Gewichtung die Weitergabe von Informationen zwischen Neuronen durch Neurotransmitter übermittelt (siehe [9]). Die Art und Anzahl der Neurotransmitter hat erheblichen Einfluss auf die Übertragung und deren Geschwindigkeit zwischen den einzelnen Neuronen. Die Auswirkung der wohl bekanntesten Vertreter Noradrenalin, Dopamin und Serotonin sind auch aktueller Inhalt vieler medizinischer Studien. Analog hierzu übernimmt in den künstlichen neuronalen Netzen die Gewichtungen an den einzelnen Neuronen diese Funktion.

Um nicht messbaren Störungen entgegen zu wirken, können verschiedene Gewichtungen angelernt und als neuer Neurotransmitterstatus abgelegt werden. In der praktischen Verwendung kann das angelernte Verhalten eines bereits trainierten Netzes als Standard Neurotransmitterstatus behandelt werden. Wenn ein festgelegtes Fehlermaß überschritten wird, hier 1% des Mittelwertes über eine Stunde bzw. vier Prognoseschritte, wird ein neuer Neurotransmitterstatus erstellt (siehe Abbildung 3). Für jede Störung wird so ihre eigene Gewichtsmatrix $W_{i\lambda}$ entsprechend eines Neurotransmitterstatus angelegt und über die in 2.2 beschriebenen Algorithmen auf Basis des bisher besten Neurotransmitterstatus trainiert.

Im Betrieb wird eine parallele Prädiktion der Prozessgrößen durch alle Neurotransmitterstadien durchgeführt und deren Ergebnisse durch Bildung

der kleinsten Fehlerquadrate (MSE) der letzten vier Prognosen miteinander verglichen.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{8}$$

Um einer zu hohen Anzahl an Neurotransmitterstadien entgegenzuwirken, sollte zuerst ein Standardverhalten trainiert werden, welches das Grundverhalten des Prozesses bereits gut wiedergibt. Des Weiteren darf das Fehlermaß nicht zu niedrig gesetzt werden. In den hier vorgestellten Prozessen ist ein vergleichsweise niedriges Fehlermaß eingesetzt worden, da bereits ein gut trainiertes Netz verwendet wurde. Desweiteren kann eine maximale Anzahl an Neurotransmittern festgelegt werden. Wird diese Anzahl überschritten wird der am wenigste genutzte Neurotransmitter wieder entfernt. In den hier vorgestellten Prozessen wurde allerdings niemals eine Anzahl von sechs Stadien überschritten, was mit der beschränkten nutzbaren Datenbasis zusammenhängt.

2.4 Störgrößenerkennung

Durch die Verwendung verschiedener parallel verarbeiteter Netzgewichte, bzw. Neurotransmitterstadien kann der Einfluss von zu erwartenden Störungen bereits prognostiziert werden, indem die bereits vorhandenen Stadien die möglichen Auswirkungen berechnen. Tritt nun eine bekannte Störung ein, kann über den Vergleich der Ausgaben aller Stadien das entsprechende Stadium ausgewählt und verwendet werden.

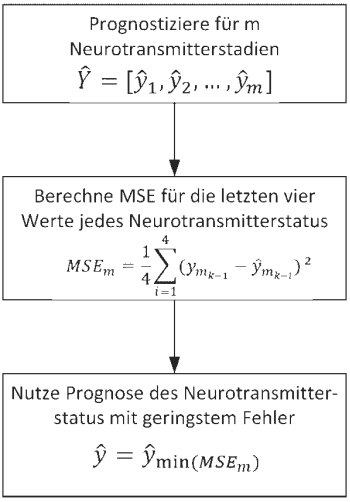


Abbildung 5: Auswahl Neurotransmitterstatus

Während der Verwendung wird der minimale Fehler der prognostizierten Werte mit den realen Werten verglichen. Ist ein Stadium als das aktuell genauste ermittelt worden, so wird dieses Stadium bis zur nächsten Änderung verwendet (siehe Abbildung 5).

Durch Definition eines Grundverhaltens und Beobachtung des Wechsels eines Neurotransmitterstadiums können die verantwortlichen Störgrößen ebenfalls detektiert werden, wie z.B. die erhöhte Luftwechselrate durch Besucherverkehr innerhalb des betrachteten Museumsraums.

3 Test und Validierung

Zum Test und zur Validierung der Funktion der Neurotransmitter wird hier zum einen mit in der Applikation gesammelten Daten in einem Zeitraum von nahezu einem Jahr gearbeitet und zum anderen eine Simulation verwandt. Zum Vergleich der Netzstrukturen für verschiedene gesteuerte Szenarien, wird die Simulation des Raumverhaltens genutzt. Um den Transport von Wärmeenergie und Feuchtigkeit in einen Raum oder aus einem Raum heraus zu simulieren, wird neben den Effekten des natürlichen und künstlichen Luftwechsels, der Transport durch Wände und Decken sowie die Speicherfähigkeit von Materialien innerhalb des Raumes betrachtet.

Zur Bestimmung der Fehlermaße in Simulation und anhand der Applikationsdaten wird der normalized mean square error (NMSE) nach [11] genutzt:

$$NMSE = \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

3.1 Raummodellbildung über Beukenmodell

Zur Darstellung des Wärmetransports in Abhängigkeit der Materialien eines Raumes bietet das Beukenmodell (siehe [11]) einen vereinfachten Ansatz. Hierbei wird der instationäre Transport durch Materialien betrachtet. Bei dieser Betrachtung wird die Speicherfähigkeit von Materialien nicht vernachlässigt, wodurch die Bilanz nicht nur von der Zeit sondern zusätzlich vom Ort abhängig ist, was wiederum eine partielle Differentialgleichung zur Folge hat. Durch Diskretisierung einer Anzahl von festen Schichtenbreiten werden im Beukenmodell nach Feist Zwischentemperaturen über Rückwärtsdifferenzquotienten gebildet. Wände werden durch eine sinnvoll festzulegende Anzahl (angelehnt an die unterschiedlichen Materialien und deren Stärke des entsprechenden Gebäudeteils) von Schichten mit jeweiligen Wärmeleitkoeffizienten dargestellt Die Transportvorgänge können durch ein elektrisches

Analogienetzwerk dargestellt werden (siehe Abbildung 6). Analog zu dem dargestellten elektrischen Netzwerk wird hierbei der Wärmewiderstand (entspricht el. Widerstand) und die Speicherfähigkeit (entspricht el. Kondensator) bestimmt. Ein analoges Modell kann für den Feuchtetransport verwendet werden, wobei die wirkliche Transportleistung hierbei zu vernachlässigen, allerdings die Speicherfähigkeit relevant ist.

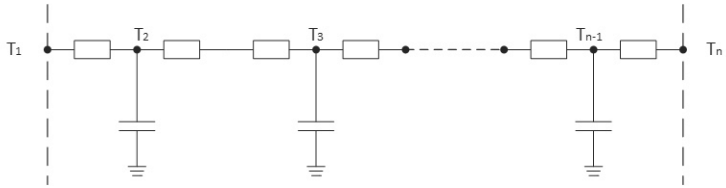


Abbildung 6: Elektrisches Analogienetzwerk für den Wärmetransport im Beukenmodell

3.2 Speichereffekte und Beeinflussung von Feuchtigkeit

Die Speichereffekte von Feuchtigkeit der im Raum befindlichen Materialien können vereinfacht wie in [12] vorgeschlagen als eine Pufferfunktion auf Basis eines Diffusions-Sorptions-Modells dargestellt werden. Mit Verwendung material- bzw. raumspezifischer konstanten R^F und T^F können so die relevanten Effekte dargestellt werden.

$$F(t) = \frac{1}{R^F} \sum_{i=0}^{\infty} e^{-t\frac{(2i+1)^2}{T^F}}$$

Analog den Wärmekoeffizienten müssen die genutzten material- und raumspezifischen Konstanten durch die Bestimmung der Materialien festgelegt werden. Ebenfalls müssen die der Luft ausgesetzten Flächen der jeweiligen Materialien bestimmt werden. Da es sich hierbei allerdings lediglich um Modellversuche handelt, werden zufällig gewählte Konstanten innerhalb realistischer Parameter ausgewählt.

3.3 Störgrösseneinfluss

Durch Verwendung des Beukenmodells und der Pufferfunktion konnte durch die Simulation einer zusätzlichen variablen Lüftung durch eine einfache Bilanzierungsgleichung ein nutzbares Modell mit typischen Raumverhalten geschaffen werden. Besonderes Augenmerk liegt hier in der Betrachtung von Störgrößen durch Änderung der Luftwechselrate. Zuerst wurde ohne weitere Störung das Netz mit einem Datensatz von 2500 Werten von Innen- und Außentemperatur sowie Innen- und Außenluftfeuchtigkeit trainiert. Das Abtastintervall betrug hierbei 15 min und es wurden bei der Applikation erfasste Werte für die Außendaten verwendet. Für die Prädiktion von Temperatur und Luftfeuchtigkeit wurden hierbei jeweils ein eigener Ansatz mit drei versteckten Schichten mit

jeweils drei Neuronen inklusive der zusätzlichen Adaptionsschicht (siehe 2.1) genutzt. Das ungestörte Systemverhalten konnte mit dieser Netzdimension ausreichend bestimmt werden. Vorhergesagt wurden lediglich die Änderungen vom aktuellen Zeitpunkt aus. Anschließend wurden mit um 25% und 50% geänderter Luftwechselrate sowie um 20% erhöhtem Speicherverhalten definierte neue Zustände als Störgrößenszenarien verwendet. Zum Test der Identifikation von geändertem Prozessverhalten wurde anschließend ein Profil mit sich laufend ändernden Zuständen und einem komplett neuen Fehler durch Kombination der Änderung der Luftwechselrate und Erhöhung des Speicherverhaltens erstellt. Ergebnisse der Simulation für ein erhöhtes Speicherverhalten (Störung 1) und erhöhte Luftwechselrate (Störung 2) sind in Abbildung 7 dargestellt.

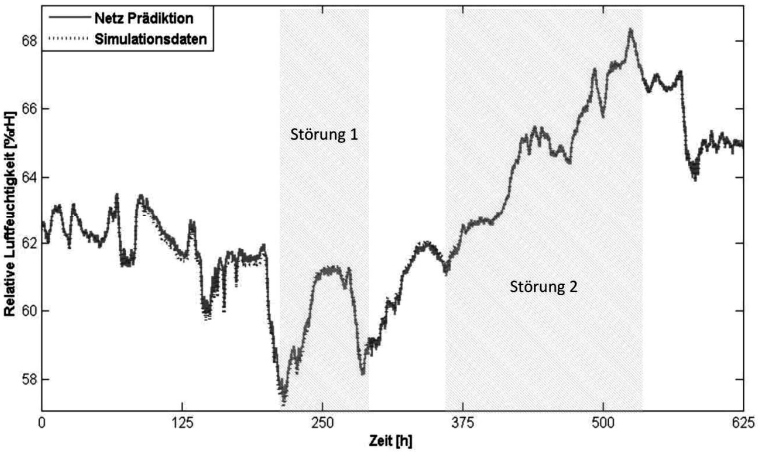


Abbildung 7: Prädiktion durch neuronales Netz mit Neurotransmitterstadien

Durch die Verwendung der einzelnen Neurotransmitterstadien können die Störungen zeitnah erkannt werden und durch einen Wechsel auf diesen Status ebenfalls prognostiziert werden. Abbildung 8 zeigt den Übergang zu einem Neurotransmitterstatus.

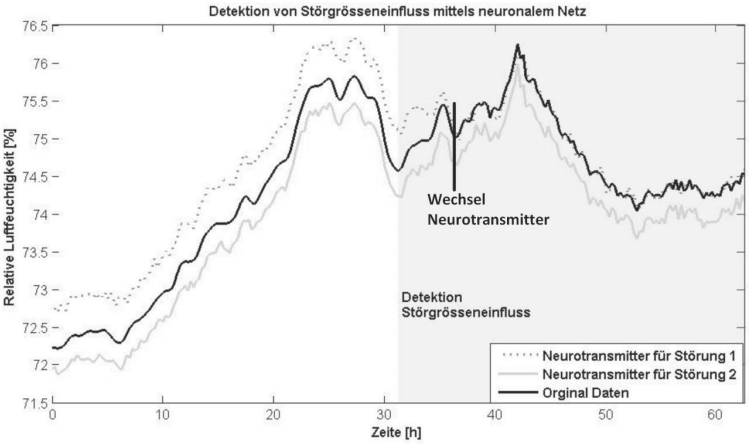


Abbildung 8: Vergleich zweier Störungsprädiktionen

Die prognostizierten Störungen 1 und 2 (siehe Abbildung 8) liegen bis zum Eintreten der Störung 1 außerhalb des akzeptablen Genauigkeitsbereichs und werden nicht verwendet. Ab Eintreten der Störung wird der Fehler des prognostizierten Ergebnisses für Störung 1 geringer und dieser Neurotransmitterstatus kann verwendet werden. Es konnte über die gesamte Simulation des beschriebenen Störungsprofil durch Wechsel der Neurotransmitterstadien ein NMSE (Normalized mean Square Error, siehe [11]) von 0.009 erreicht werden, was allerdings mit den nahezu perfekten Bedingungen einer solchen Simulation zusammenhängt.

3.4 Applikationsdaten

Zur Bestimmung der für die Klimatisierung relevanten Daten von Temperatur und Luftfeuchtigkeit in einem Prognosehorizont von einer Stunde werden hier nach ausführlichen Tests jeweils ein eigener Ansatz für jede Größe mit den drei Hidden Layern mit jeweils drei Neuronen genutzt. Es wurden Daten eines Zeitraums von etwas über 300 Tagen mit einer Intervallzeit von 15 Minuten genutzt. Dies entspricht circa 30.000 Datenpunkten. Hiervon wurden 50% zum Training des Netzes, 25% zum Validieren und zum Testen verwendet. Nach dem Training des Verhaltens ohne Nutzung weiterer Zustände erhält man bereits ohne Verwendung verschiedener Neurotransmitterstadien ein zufriedenstellendes Ergebnis für Bereiche ohne Störgrößeneinfluss (siehe Abbildung 9).

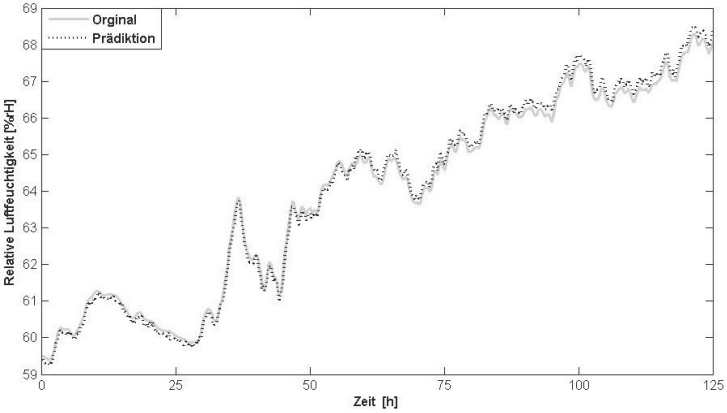


Abbildung 9: Prädiktion von Raumverhalten ohne Störgrößeneinfluss; NMSE 0,15

Sobald allerdings Störungen in den Datensätzen auftreten, konnten die Daten teilweise nicht hinreichend mit einer einzelnen Netzgewichtung prognostiziert werden. Hier wurde nun eine relative Abweichung von 2% als Grenze für die Errechnung eines neuen Neurotransmitterstatus eingeführt und der Wechsel über die kleinsten Fehlerquadrate der letzten Stunde verwendet. Den Vergleich des Netzes mit und ohne Wechsel des Neurotransmitterstatus in einem ausgewählten Bereich mit hohem Fehler ist in Abbildung 10 dargestellt. Der NMSE ohne Neurotransmitterstatuswechsel liegt in diesem Bereich bei 0,84 mit Wechsel bei 0,079. Die prädizierten Werte liegen wesentlich näher an den realen Werten.

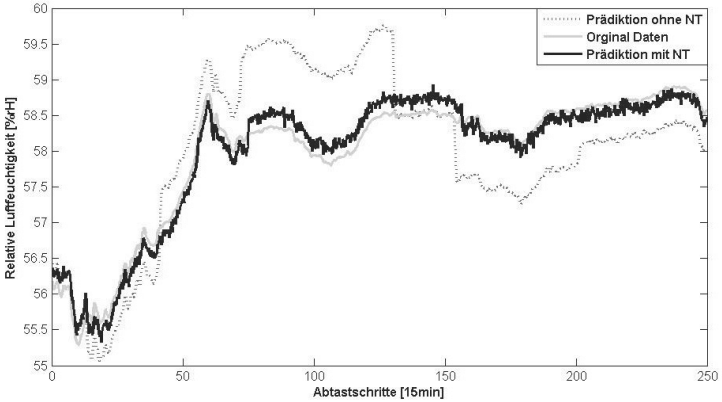


Abbildung 10: Vergleich der Feuchteprädiktion mit und ohne Neurotransmitterwechsel

4 Zusammenfassung, Ausblick und Diskussion

In diesem Beitrag wurde mit der Verwendung von Neurotransmitterstadien ein Verfahren zur Störgrößenerkennung vorgestellt. Besonders nicht messbare Störgrößen, welche wie ein geändertes Prozessverhalten wirken, können so einfach erkannt werden. Durch die Nutzung zusätzlicher Neurotransmitterstadien kann schnell auf Änderungen im Verhalten reagiert werden, sofern eine derartige Störung bereits auftrat und implementiert wurde. Erfolgt eine neue Störung kann diese adaptiert und in Zukunft schneller erkannt werden. Als eventuelle Kritik könnte die Verwendung von mehreren kleinen Netzen für spezielle Verhaltensweisen anstelle mehrerer übertrainierter Netze angebracht werden. Allerdings kann durch die offene Adaption von weiteren Störgrößen ein neues Verhalten ohne den Verlust des Grundverhaltens mit wesentlich geringerem Aufwand adaptiert werden. Der hierfür erhöhte Rechenaufwand stellt jedoch eine zusätzliche Belastung dar, und ist daher für zeitkritische Anwendungen nur bedingt einsetzbar.

Die Verwendung innerhalb eines Prozesses zur Regelung einer Klimatisierungsanlage soll im Rahmen eines nichtlinearen modellprädiktiven Reglers (NMPC) im kommenden Winter verwendet werden. Hierfür wird in Zukunft die Möglichkeit zur zusätzlichen Einflussnahme durch Aktoren genutzt, welche allerdings eine wesentlich höhere Komplexität in das System einbringen.

Ein weiterer Ansatz ist das Training der Umkehrfunktion eines Prozesses F_s^{-1}. Ist diesee als Modell vorhanden so kann durch die Vorgabe des gewünschten Sollwertes der hierfür nötige Stellgrad für die Aktoren ermittelt werden. Ebenfalls wäre die Erweiterung eines bereits vorhanden Simulationsmodells mit dem hier bereits vorgestellten Netztypen zur Adaption des Verhaltens denkbar (siehe Abbildung 11). Hierfür kann als Grundlage eine Gebäudesimulationssoftware (wie Beispielsweise TRNSYS oder WUFI) zur Modellierung des Verhalten eines Gebäude genutzt werden und lediglich eine Adaption von Störgrößen oder in der Simulation nicht beachteter Effekte durch das Netz erfolgen.

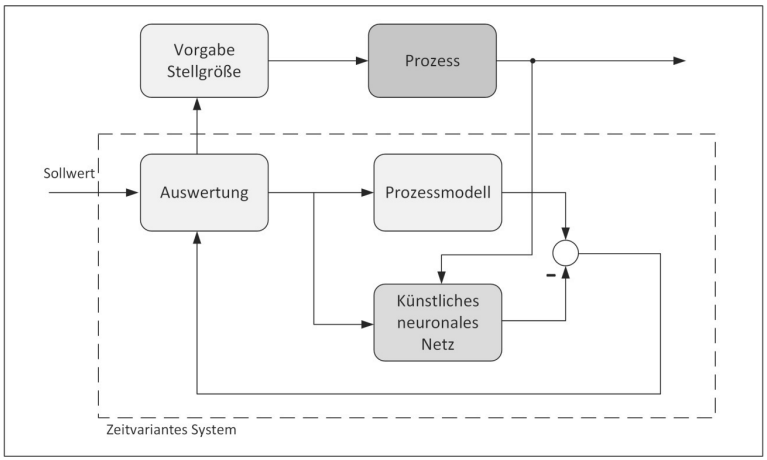


Abbildung 11: Durch KNN erweitertes Prozessmodell

Eine dynamische Anpassung des Lernfaktors könnte zu einer Verbesserung der Adaption neuer Zustände das teilweise lange Einlernen verkürzen. Ebenfalls wäre die Nutzung anderer Algorithmen, wie z.B. eines evolutionären Algorithmus möglich.

5 Literatur

[1] CEN: *Conservation of Cultural Property*. EN 15757 Standard. Bruxelles: European Committee for Standardization; 2008

[2] Garrecht H.: *Raumklimaoptimierung im Spannungsfeld von Denkmalpflege und Nutzung unter energetischen Aspekten, dargestellt an Praxisbeispielen.* Heringsdorf (Germany): 19. Hanseatische Sanierungstage – Bauphysik und Bausanierung; 2008

[3] Kilian R., Sedbauer K., Krus M.: *Klimaanforderungen für Kunstwerke und Ausstattung historischer Gebäude.* Stuttgart: IBP-Mitteilung; 2005

[4] Cigler, J., Siroky, J., Korda, M., Jones, C.: On the Selection of the Most Appropriate MPC Problem Formulation for Buildings.

[5] Kruse, R. et. al.: *Computational Intelligence*. Vieweg+Teubner Verlag; Wiesbaden; 2012, S.139-149 u. S.43-49

[6] Lenze, B.: Einführung in die Mathematik neuronaler Netze. Logos Verlag, Berlin; 1997 S.64 u. S128

[7] Zell, A.: Simulation neuronaler Netze. 4. Auflage. Oldenbourg Wissenschatsverlag, München; 2003

[7] Fausett, L.: Fundamentals of Neural Networks. Pearson, London; 1993, S.289-300

[8] Trepel, M.: *Neuroanatomie*. Urban & Fischer, München; 2008

[9] Hoffmann, F., Mikut, R., Kroll, A., Reischl, M., Nelles, O., Schulte, H., Bertram, T.:Computational Intelligence: State-of-the-Art Methoden und Benchmarkprobleme, KIT Scientific Publishing, Karlsruhe; 2012 S.16-21

[10] Feist, W.: *Thermische Gebäudesimulation - Kritische Prüfung unterschiedlicher Modellansätze*; Verlag C. F. Müller, Heidelberg; 1994

[11] Reick, M.; Setzer, M. J.: *Untersuchung des Sorptionsverhaltens wohn-raumumschließender Materialien*. DFG, Deutsche Forschungsgemeinschaft: DFG-Forschungsschwerpunktprogramm Bauphysik der Außen-wände- Schlussbericht; Fraunhofer IRB Verlag, Stuttgart; 2000, S. 363–374

Künstlich Neuronale Netze als Lösungsansatz zur Ermittlung komplexer Korrelationen in der Produktion zum Einsatz in Cyber Physical Systems

Tobias Fries, Roman Kalkreuth, Markus Hein

GNO Gesellschaft für Neuronale Organisationsintelligenz mbH
Hans-Böckler-Str. 14, 44787 Bochum
Tel.: (0234) 6870 98-11, E-Mails: tobias.fries@neurodialog.de;
roman.kalkreuth@tu-dortmund.de; markus.hein@neurodialog.de

Zusammenfassung

Künstlich Neuronale Netze sind schematische Abbildungen biologischer neuronaler Netze, deren Gemeinsamkeit die Selbstlernfähigkeit ist. Weitere Gemeinsamkeit ist ihre rekursive Selbstbeeinflussung, d.h. dass Änderungen an einer Stelle des Systems das zukünftige Systemverhalten verändern. Mehrere unterschiedliche zusammengeschaltete und interagierende Systeme in der Produktion werden als Cyber Physical Systems (CPS) bezeichnet. Der Aufsatz beschäftigt sich mit den für die Einrichtung eines (künstlich) intelligenten CPS erforderlichen Technologien. Er zeigt, wie in Künstlich Neuronalen Netzen unter Vermeidung linearer Steuerungstechnik durch Selbstlernprozesse auch komplexe Korrelationen erkannt und auf diese unter selbständiger Interaktion zwischen Subsystemen des CPS systembeeinflussend reagiert werden kann.

Stichworte: Cyber Physical Systems, CPS, Künstlich Neuronale Netze, Industrie 4.0, Design of Experiments, multiple Korrelationen, Multiparameter, Graph, künstliche Intelligenz, Produktion, Logistik.

1. Problemstellung

1.1. Definition von Industrie 4.0 und Cyber Physical Systems

Der Schritt in die Vernetzung unterschiedlicher Geräte und Maschinen wird mit dem Begriff *Industrie 4.0* oder *Cyber Physical Systems (CPS)* beschrieben. SENDLER definiert diese mit Bezug auf das DFKI als „Netzwerk miteinander agierender Elemente mit physikalischem In- und

Output“, in Abgrenzung zu reinen Netzwerken ohne physikalische Ein- und Ausgabe, aber ebenso Standalone-Geräten [12; S.7]. Im Zusammenhang einer solchen M2M- Vernetzung wirft er die Frage nach den zur Vernetzung verwendeten Standards und Schnittstellen, die für die Kommunikation der Maschinen und Endgeräte untereinander sorgen werden und betont im Folgenden die herausragende Bedeutung der verwendeten Software für Vernetzung und Prozessmanagement [12; S.12ff]. RUSSWURM nennt in einem Ausblick das Internet als Basis zum Informationsaustausch in Echtzeit, in denen Maschinen eigenständige Bewertungen und Entscheidungen vornehmen [11; S.31], was nur intelligente Systeme leisten können werden. Einen besonderen Stellenwert hat dabei die Einzelfertigung, in der keine traditionellen seriellen Prozesse anwendbar sind, HUBER nennt für diese als wesentlichen Erfolgsfaktor der Zukunft die Vernetzung auch über Unternehmensgrenzen hinweg zu Lieferanten und Kunden [5; S.121].

1.2. Integration unterschiedlicher Standards als Herausforderung

BAUM formuliert als Lösungsansatz für eine solche Software ein Domänenmodell mit inhaltlich konsistent beschriebenen strukturellen Beziehungen und Semantiken, das als Schnittstelle für die unterschiedlichen Systeme dient [2; S.47]. Als Herausforderung beschreibt er die Migration vorhandener Daten, welche sich durch zunehmende Datenmengen (Big Data) in Zukunft noch verstärkt, aber gleichzeitig große Potentiale für eben jenes Zusammenwachsen der Datenströme unterschiedlicher Geräte mit sich bringt. Besondere Möglichkeiten werden in der Auswertung und Korrelationsanalyse von Datenströmen verschiedenster Quellen unterschiedlicher Strukturen, Volumina und Fließgeschwindigkeiten gesehen, die jedoch mit traditioneller Informationstechnologie als nicht bearbeitbar eingeschätzt wird [2; S.47].

1.3. Komplexität

Der Wunsch nach miteinander sprechenden Maschinen, die (oder deren Schnittstellensoftware) eigenständige Entscheidungen aus großen Datenmengen unterschiedlicher Strukturen, Volumina und Fließgeschwindigkeiten trifft, lässt sich als komplex beschreiben. Diese Anforderung beinhaltet die Zusammenschaltung der Datenströme von unterschiedlichen Einzelsystemen zu einem übergeordneten System, welches aus dem Input von Daten des einen Systems Handlungen in einem anderen Teilsystem vornimmt, die sich wiederum auf das originäre oder andere Teilsysteme

auswirken. Kurz: Ein solches System ist selbstreferentiell und somit komplex in dem Sinne, dass es aufgrund der vielfältigen Wechselwirkungspotentiale weder exakt prognostizierbar, noch in seiner Kausalität retrospektiv erklärbar ist. In der Verwendung der Begrifflichkeit von Systemkomplexität sind die Ausarbeitungen von LUHMANN [7] zu Definition, Eigenschaften und Funktionsweise von Systemen weitgehend übereinstimmend mit dem hier verwendeten Verständnis, mitsamt ihren Möglichkeiten und Einschränkungen. Er selbst beschreibt die Themenwahl „Sozialer Systeme“ als Gegenstand seiner Forschung ausdrücklich nicht als Ausschluss einer Übertragbarkeit auf Maschinen, sondern als Weg der „Generalisierung und Respezifikation“, um den Besonderheiten spezifischer Systeme in der Adaption gerecht werden zu können [7; S.32].

Zwei wesentliche übereinstimmende Eigenschaften sind die Subjektivität und die permanente Dynamik von Systemen. Subjektiv ist ein komplexes Cyber Physical System deshalb, weil seine Geräte ebenfalls selbstreferentiell aufeinander wirken und sich so gegenseitig in einer nicht präzise prognostizierbaren Weise beeinflussen. Eine permanente Dynamik ist deshalb ebenfalls Eigenschaft von Cyber Physical Systems, weil durch die zahlreichen angeschlossenen Einzelsysteme mit ihrem jeweiligen individuellen Eigensystemstatus ein Abbild des Gesamtsystems niemals erzeugt werden kann, und durch den eigenständigen Betrieb jedes Teilsystems auch zu keinem Augenblick konstant stehen bleibt.

1.4. Problemdimensionen unter Komplexität

Vor einigen Dekaden konnte eine Maschine normalerweise noch durch die Erfahrung des Maschinenführers optimiert werden. Heute sind bereits groß dimensionierte Computer und Datenbanksysteme im Einsatz, Optimierungs- und Simulationssoftware ist umfangreich verfügbar und ausgereift für die Anwendung. Diese bieten ebenfalls Antworten auf stochastisches, also von Zufallskomponenten beeinflusstes Systemverhalten [exemplarisch hierzu: 10]. In der Praxis herrschen heute jedoch meist auf relationalen Datenbanken basierende Systeme vor, die Herausforderungen von Big Data nicht standhalten. Solche Herausforderungen sind etwa Korrelationsanalysen multipler Parameter, wie sie in zusammengeschalteten Cyber Physical Systems in den meisten Fällen, häufig jedoch bereits bei modernen Einzelproduktionsanlagen erforderlich sind.

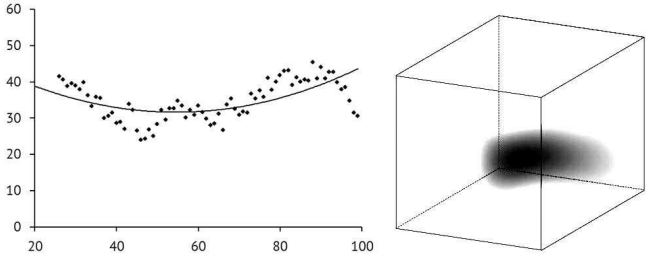


Abb 1: Fehlerpositionen im zwei- und dreidimensionalen Raum. *Eigene Darstellung.*

Die in Abbildung 1 gezeigten beispielhaften Fehlerdimensionen visualisieren den sich mit jedem zusätzlichen Parameter vergrößernden Möglichkeitenraum in der Lokalisierung eines Fehlers. Die Anzahl der Möglichkeiten folgt dabei den statistischen Grundsätzen der Kombinatorik und erreicht mit zunehmender Anzahl an Dimensionen schnell Größenordnungen, in denen herkömmliche Relationale Datenbanken an ihre Grenzen kommen. Bei lediglich 100 (diskreten) Einstellungsmöglichkeiten eines Parameters entstehen in der Auswertung von nur fünf Parametern bereits 10 Milliarden Kombinationsmöglichkeiten. Bisher werden zur Speicherung und Verwaltung von Datensätzen vorwiegend relationale Datenbanken verwendet. Dieses ist durch eine Ansammlung von Tabellen (Relationalen) charakterisiert. Jede Zeile in einer Tabelle bildet ein Tupel an Attributen welches einen Datensatz repräsentiert. Für die Verbindung der Datensätze unterschiedlicher Relationen können Beziehungen angelegt werden. Diese können über spezielle Attribute (Fremdschlüssel) erfolgen, welche die Referenzen bzw. die Schlüssel eines Datensatzes aus fremden Relationen enthalten und so die Verbindung zwischen den Datensätzen gewährleisten. Je nach Kardinalität wird eine spezielle Zuordnungstabelle zur Speicherung der Beziehungen verwendet. Ein Nachteil von relationalen Datenbanken ist, dass für ein hohes Maß an Vernetzung der Daten viele Beziehungen über Zuordnungstabellen realisiert werden müssen, um die einzelnen Bereiche untereinander zu verbinden. Eine Vernetzung dieser Art schafft Umwege, welche sich in der Summe addieren und somit für die Effizienz der Datenbank nachteilig sind. Um den Ansprüchen heutiger Datenansammlungen gerecht zu werden, muss die Möglichkeit Daten direkter miteinander zu verbinden gegeben sein. Hierbei bietet es sich an, die Vernetzung über ein Graphensystem zu realisieren, welches eine direktere und flexiblere Vernetzung der einzelnen Daten ermöglicht. Sogenannte Graphdatenbanken sind speziell für die Speicherung von vernetzen Daten und dessen Traversierung ausgelegt,

welche wie schon erwähnt mehr an Bedeutung gewinnen. Bei stark vernetzen und großen Datenmengen wirken sich diese Eigenschaften unter anderem durch deutlich kürzere Abfragezeiten von benötigten Informationen aus. Bei einer Graphdatenbank wird ein Datensatz als Knoten betrachtet, während die Beziehungen zwischen den Datensätzen über Kanten hergestellt werden. Bestimmte Eigenschaften können dabei sowohl einem Knoten als auch einer Kante zugewiesen werden. Zwar ist eine relationale Datenbank auch im Stande Graphen über Zuordnungstabellen abzubilden, stellt aber hingegen zur Graphdatenbank keine entsprechenden Möglichkeiten zur effizienten Durchsuchung und Manipulation der Graphstruktur bereit.

1.5. Heutige Antworten auf die Anforderungen von CPS

Die wesentliche sich stellende Frage ist nun die nach dem praktischen Umgang mit dieser Komplexität, nach ihrer Reduzierung auf ein handhabbares Niveau. BROY betont in der Anforderung an eine entsprechende Software ihre Interdisziplinarität, da eine solche ebenfalls an der Entwicklung der Produkte beteiligt ist [3; S.80]. Einen besonderen Stellenwert hat die softwareseitige Umsetzung und Abbildung in Datenbanken. Die Ansätze sind entsprechend geprägt von den jeweiligen meist linearen Denkweisen der beitragenden Disziplinen. Lineares Denken ist innerhalb der jeweiligen Disziplinen weder falsch, noch konnte sich in den meisten Disziplinen eine andere Art des Denkens in der praktischen Umsetzung bewähren. So folgen die meisten industriellen Fertigungsanlagen und -verfahren linearen Regeln aus „Wenn-Dann“-Bedingungen, ähnliches gilt für Simulationsmodelle und für die Organisation von Datenbeständen, worauf das folgende Kapitel näher eingeht. Die vorherrschende Sicht in der Literatur zum Thema Industrie 4.0 und Cyber Physical Systems ist ebenfalls von linearen Lösungsansätzen geprägt, so wie es sich für die einzelnen Disziplinen stets bewährt hat. Eine Antwort auf den Umgang mit Komplexität und Anforderungen von Cyber Physical Systems an eine interdisziplinäre Antwort greift BARABÁSI mit der Vorstellung der Netzwerkwissenschaften als datenbasierte, sich aktuell ausgesprochen dynamisch entwickelnde Perspektive auf [1]. Die interdisziplinare Bedeutung von Komplexität als Forschungsbereich unterstreicht auch er durch die Frage, ob sie nun der Physik, dem Ingenieurwesen, Biologie, Mathematik oder Computerwissenschaften zugeordnet werden könne, oder gar allen gemeinsam [1; S.14]. Ausdrücklich beschreibt er mit beispielhafter Referenz auf Finanzmarkt-, Mobilitätsdaten ganzer Länder, Import- und Export-Statistiken die Zugangsmöglichkeit zu nie gekannten Datenmengen, deren Bearbeitungswerkzeuge „vor unseren Augen in diesem

Moment erst geboren werden“ und deren Entwicklung im Rahmen der Netzwerkwissenschaften das Forschungsfeld der Komplexität neu definieren wird [1; S.15].

2. Stand der Technik

2.1. Vergleich relationaler mit graphenbasierten Datenbanksystemen

Bisher werden zur Speicherung und Verwaltung von Datensätzen vorwiegend relationale Datenbanken verwendet. Diese sind durch eine Ansammlung von Tabellen (Relationalen) charakterisiert. Jede Zeile in einer Tabelle bildet ein Tupel an Attributen, welches einen Datensatz repräsentiert. Für die Verbindung der Datensätze unterschiedlicher Relationen können Beziehungen angelegt werden. Diese können über spezielle Attribute (Fremdschlüssel), welche die Referenzen bzw. die Schlüssel eines Datensatzes aus fremden Relationen enthalten und so die Verbindung zwischen den Datensätzen gewährleisten. Je nach Kardinalität wird eine spezielle Zuordnungstabelle zur Speicherung der Beziehungen verwendet [8; S.28ff]. Ein Nachteil relationaler Datenbanken sind für ein hohes Maß an Vernetzung der Daten ihre viele Beziehungen über Zuordnungstabellen, welche die einzelnen Bereiche untereinander verbinden. Eine Vernetzung dieser Art schafft Umwege, welche sich in der Summe addieren und sich somit nachteilig auf die Effizienz der Datenbank auswirken. Um den Ansprüchen immer größer werdender Datensammlungen gerecht zu werden, müssen Daten direkter miteinander verbunden werden, die beschriebenen Umwege müssen soweit es geht eliminiert werden. Die Vernetzung über ein Graphensystem bietet eine direktere und flexiblere Vernetzung der einzelnen Datensätze. Sogenannte Graphdatenbanken sind speziell für die Speicherung von vernetzen Daten und deren Traversierung ausgelegt. Bei stark vernetzen und großen Datenmengen wirken sich diese Eigenschaften unter anderem durch deutlich verkürzte Schreib- und Lesezugriffe aus. Strukturell wird in einer Graphdatenbank ein Datensatz als ein Knoten betrachtet, während die Beziehungen zwischen den Datensätzen über Verbindungen hergestellt werden. Diese Begriffe stammen aus der Graphentheorie, und werden auch als Ecken (Knoten) und Kanten (Verbindungen) bezeichnet [9; S.236]. Sowohl Ecken als auch Kante können bestimmte Eigenschaften zugewiesen bekommen. Aus theoretischer Perspektive ist zwar auch in relationalen Datenbanken die Abbildung von Graphen über Zuordnungstabellen möglich, jedoch stehen –bedingt durch die oben beschriebene Organisationsstruktur in Form durch Relationen verbundener Tabellen– keine

Möglichkeiten zum effizienten Durchsuchen (traversieren) und zur Manipulation der Graphstruktur zur Verfügung.

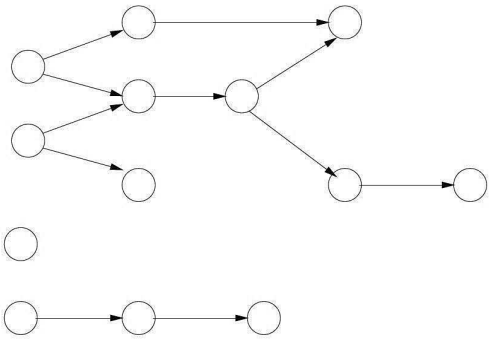


Abb. 2: Beispielhafter azyklischer Graph [13]

VICKNAIR (et. al) [13] verglichen die Leistung relationaler Datenbanken mit der von Graphdatenbanken in Bezug auf die Dauer traversierender Abfragen. Für den Vergleich wurden zwölf SQL-Datenbanken und zum Vergleich zwölf Neo4j Graphdatenbanken erstellt, die jeweils einen azyklischen Graphen enthielten, wie in Abbildung 2 dargestellt. Jedem Knoten des Graphen wurden zusätzlich unterschiedlich gestaffelte Mengen an Daten angefügt, jeweils in identischer Anzahl in den SQL- und den Neo4j Datenbanken. Die Staffelung der Knoten betrug 1.000, 5.000, 10.000 und 100.000 Knoten. Die Daten selber wurden zufällig für die Typen Integer, 8KB String und 32KB String generiert.

Das Testsystem verfügte über eine Intel Core 2 Duo CPU mit einer Taktfrequenz von 3GHZ, 4GB Arbeitsspeicher und das Betriebssystem Ubuntu Linux 9.10.

Der Versuchsaufbau beinhaltete für den Vergleich drei unterschiedliche Abfragen [13]:

S0: Finde alle Knoten, welche die Singleton-Eigenschaft besitzen, also über keine eingehende und ausgehenden Kanten verfügen.

S4: Traversiere den Graphen bis auf Ebene 4 und zähle die Nummer der erreichbaren Knoten.

S128: Traversiere den Graphen bis auf Ebene 128 und zähle die Nummer der erreichbaren Knoten.

Datenbank	MySQL S4	Neo4j S4	MySQL S128	Neo4J S128	MySQL S0	Neo4j S0
1000int	38,9	2,8	80,4	15,5	1,5	9,6
5000int	14,3	1,4	97,3	30,5	7,4	10,6
10000int	10,5	0,5	75,5	12,5	14,8	23,5
100000int	6,8	2,4	69,8	18	187,1	161,8
1000chars8K	1,1	0,1	21,4	1,3	1,1	1,1
5000chars8K	1	0,1	34,8	1,9	7,6	7,5
10000chars8K	1,1	0,6	37,4	4,3	14,9	14,6
100000chars8K	1,1	6,5	40,9	13,5	187,1	146,8
1000chars32K	1	0,1	12,5	0,5	1,3	1
5000chars32K	2,1	0,5	29	1,6	7,6	7,5
10000chars32K	1,1	0,8	38,1	2,5	15,1	15,5
100000chars32K	6,8	4,4	39,8	8,1	183,4	170

Tab. 1: Vergleich Abfragezeiten in Millisekunden zwischen MySQL und Neo4J [13]

Die Abfragezeiten aus Tabelle 1 zeigen die leistungsmäßige Überlegenheit stark vernetzter Daten, die in Graphen gespeichert werden, ebenso die Überlegenheit der Graphdatenbank beim Traversieren. Eine strukturelle Begründung liegt in der Datenablagestruktur von MySQL-Datenbanken in Form von Tabellen, die nicht für das Traversieren ausgelegt sind. Eine relationale Datenbank muss über sogenannte JOINS zwischen mehreren Tabellen arbeiten, welche das kartesische Produkt zwischen den Relationen bilden [8; S.68ff]. Einzelne Datensätze können daher nicht direkt miteinander in Beziehung gesetzt, was hingegen bei Graphdatenbanken möglich ist, und ihren wesentlichen Vorteil beim Traversieren begründet. Die relationale Datenbank ermittelt sämtliche mögliche Kombinationen zwischen den Relationen über das kartesische Produkt und filtert anschließend alle Datensätze heraus, die nicht zur Abfragebedingung passen. Diese Bearbeitung des gesamten Möglichkeitenraumes über zahlreiche JOINS kumuliert ein enormes Datenbearbeitungsvolumen, welches aufgrund der großen Menge bei gleicher Computersystemkonfiguration die Abfragezeiten deutlich mindert. Beziehen sich die Abfragen auf lediglich eine einzige, nicht vernetzte Tabelle, zeigen sich die Stärken der relationalen Datenbank, die für die Verarbeitung großer Mengen zusammengefasster Datensätze ausgelegt ist.

Diese Überlegenheit von Graphdatenbanken gegenüber relationalen Systemen prädestiniert sie für den Einsatz der Vernetzung von Cyber Physical Systems.

2.2. Künstlich neuronale Netze

Eine Graphdatenbank verwendet wie jeder mathematische Graph Knoten („Ecken“) und Verbindungen („Kanten“) [4; S.2]. Diese sind einerseits zur Modellierung von Netzwerken geeignet, andererseits können Sie in einer Graphdatenbank und mit entsprechenden Abfragen versehen biologische neuronale Netze simulieren. Hierauf geht HÜSKEN in seiner Definition Künstlich Neuronaler Netze ein. Er erklärt Künstlich Neuronale Netze als schwach angelehnt an natürliche neuronale Netze und beschreibt ihre Fähigkeit zur schematischen Übernahme der grundsätzlichen neuronalen Funktionsweisen von Dendriten, Axonen und Neuronen [6; S.6].

Zur Aktivierung der Selbstlerneigenschaften von KNN führt er eine mathematische Funktion zur Aktivierung eines künstlichen Neurons aj mit folgender Notation ein [6; S.6f]:

$$a_j = \sum_i w_{ji} z_i + w_{j0} \tag{2.1}$$

Die Ausgabe des Neurons über den Wert zj entsteht durch Aufruf der Aktivierung mittels normalerweise nicht linearen Aktivierungfunktion g: R→R

$$z_j = g(a_j) \tag{2.2}$$

In der Summation sind sämtliche eingehenden Verbindungen von Neuron i zu Neuron j inklusive unterschiedlicher Gewichte w als Parameter enthalten. Von den Gewichtungsfaktoren können auch mehrere enthalten sein. Das zusätzliche Neuron wj0 dient der konstanten Verschiebung, wobei in der Herleitung z(wj0) formal mit dem Wert 1 enthalten ist. Hierdurch enthält das Künstlich Neuronale Netz die Fähigkeit eigenständiger Dynamik, die ihm die Eigenschaften eines komplexen Systems verleiht.

3. Hypothesen zur Diskussion

Bezugnehmend auf die Anforderungen vernetzter Cyber Physical Systems und die oben hergeleitete Eignung Künstlich Neuronaler Netze zur Vernetzung ihrer Subsysteme, sollen folgende Hypothesen zur wissenschaftlichen Diskussion gestellt werden:

a. *KNN sind der Schlüssel zu Industrie 4.0 zur Vernetzung unterschiedlicher Geräte und Maschinen.*

b. *Innerhalb derartiger KNN können aus multiplen Parametern plattformübergreifende Korrelationen erkannt und somit bearbeitbar gemacht werden.*

c. *Durch die Selbstlern- und Selbstbeeinflussungsfähigkeit von KNN können mit ihrer Hilfe intelligente Cyber Physical Systems erschaffen werden.*

4. Herangehensweise

Als eine Voraussetzung zum effektiven Betrieb eines Cyber Physical Systems kann die Kommunikationsfähigkeit seiner Subsysteme gelten. Diese erfordern einen gemeinsamen Kommunikationsstandard und plattformübergreifende Schnittstellen.

Die Lösung dieses Problems ist sowohl auf Hardware- als auch auf Softwareseite zu suchen, wobei die Hardwaresicht für diesen Aufsatz ausgeklammert wird, da der Stand der Technik im Bereich Mikrocontroller und Schnittstellen als ausreichend angenommen wird.

Die größte Herausforderung dürfte in der Schaffung einer architektonisch sinnvollen Datenbankstruktur liegen, welche nicht nur neue eingelesene Daten verarbeiten, sondern auch Altdaten aus bestehenden relationalen Datenbanksystemen übernehmen kann. Ist dies erreicht können verschiedene Ereignisse perspektivisch zentriert und mittels Traversierung zu neuen Zusammenhängen zusammengeführt werden. Auf diese Art und Weise ist die Identifizierung auch von unspezifischen Kausalzusammenhängen möglich, die unter Umständen nur im Zusammenwirken zahlreicher Parameter überhaupt entstehen.

Jede Benutzung des Künstlich Neuronalen Netzes hinterlässt dabei Spuren, die wiederum das System selber beeinflussen und aus sich selbst zu neuen

Aktivitäten und weiteren Ausgaben führt. Dieser Verbindungsaufbau ist gleichzeitig

5. Anwendungsfälle und Ausblick

Der Einsatz von Cyber Physical Systems eignet sich besonders in komplexen Produktionsvorgängen, in denen mehrere Subsysteme und Umgebungsvariablen zusammentreffen. Als ein Beispiel könnte eine Reinraumfertigung angeführt werden, mit den exemplarischen Parametern Lufttemperatur, Luftfeuchtigkeit, Wassertemperatur und Wasserdruck, deren jeweiligen Werte jeweils protokolliert werden. Es wird unterstellt, dass die Wasser- und Luftaufbereitungssysteme getrennte Systeme sind, ebenso die Fertigung selber auch. Entstehen nun unter besonderen Bedingungen Fehler, so ist zur Behebung von Interesse zu erfahren, in welcher Parameterkonstellation dieser Fehler entstanden ist und ob diese Konstellation in unterschiedlichen Fehlersituationen statistisch korreliert. Die Antwort darauf vermag ein zu einem Künstlich Neuronalen Netz zusammengeschaltetes Cyber Physical System zu liefern, welche im Gegensatz zu linear-mathematischen Modellen vom Fehlerereignis selber ausgeht.

5.1. Künstliche Intelligenz

Künstliche Intelligenz beginnt dann, wenn innerhalb des Künstlich Neuronalen Netzes durch Etablierung neuer Verbindung und Abgleich mit vorhandenen Verbindungen Lerneffekte eintreten. Dieser Abgleich und Erstellung neuer Verbindungen entsteht durch die Aktivierung des künstlichen Neurons von alleine. Lerneffekte sind tatsächlich lediglich Systemänderungen durch Eigendynamik, die aufgrund von rekursiven Verknüpfungen in den folgenden Aktivierungen des gleichen Neurons zu anderen Ergebnissen führen können [umfassend hierzu: 7].

Eine solche rekursive Systembeeinflussung in der Praxis könnte in der Wassertemperaturregelung der beschriebenen Fertigungsanlage stattfinden. Das Wort Regelung ist vor dem Hintergrund eines Systems mit Künstlicher Intelligenz nicht ganz zutreffend, weil es eine lineare wenn-dann-Logik impliziert, die regelbasiert den Status des Systems ändert. Die charakterisierende Eigenschaft von selbstlernenden Systemen künstlicher Intelligenz ist aber eben das Fehlen von Regeln und der sich aufgrund des Systemzustands von selber einstellende Zustand. Dieser Unterschied ist von grundlegender Bedeutung, denn obgleich das Ergebnis häufig identisch

sein mag, ist das eine ein Resultat von außen dem System hinzugefügten Bedingungen (wenn Wassertemperatur >x, dann mehr Wasserdruck, o.Ä.), während das andere die Regeln aufgrund von Systemzuständen und früheren, also gelernten Erfahrungen innerhalb desselben Systems selber aufstellt. Die Lernerfahrung ist in diesem Fall intrinsisch und entstanden aus demselben System, in dem es erneut zur Anwendung kommt, nicht extrinsisch und von außen hinzugefügt, wie es bei linearer Steuerungstechnik der Fall ist.

Von echter künstlicher Intelligenz kann also gesprochen werden, wenn das System selber aufgrund von Lerneffekten in der Folge ein verändertes Systemverhalten zeigt, welches das System erneut beeinflusst. Sind Systemfehler also in einer bestimmten Konstellation besonders signifikant, so wird das System zur Behebung dieser Fehler eine Veränderung der Parameter vornehmen. Im Ausblick erscheint insbesondere die Forschung im Bereich von Design of Experiments noch vielversprechend, insbesondere welche Ergebnisse die Kombination mit Künstlicher Intelligenz aus KNN hervorbringen.

Literatur

[1] Barabási, A.-L. (2012): *The network takeover.* In: Nature Physics. Macmillan, Vol. 8, Jan. 2012, S. 14 - 16

[2] Baum, G. (2011). *Software: Die Zukunft der Industrie*. In: Sendler, U. (Hrsg.): Industrie 4.0 – Beherrschung der industriellen Komplexität mit SysLM (Systems Lifecycle Management). Springer Vieweg, Berlin Heidelberg, 2013

[3] Broy, M. (2011). *Modellbasiertes Software und Systems Engineering als Element eines durchgängigen Systems Lifecycle Managements (SysLM)*. In: Sendler, U. (Hrsg.): Industrie 4.0 – Beherrschung der industriellen Komplexität mit SysLM (Systems Lifecycle Management). Springer Vieweg, Berlin Heidelberg, 2013

[4] Diestel, R. (1996): *Graphentheorie.* Springer, Berlin, 4. Auflage 2010

[5] Huber, A.S. (2011*). Das Ziel Digital Enterprise: die professionelle digitale Abbildung von Produktentwicklung und Produktion.* In: Sendler, U. (Hrsg.): Industrie 4.0 – Beherrschung der industriellen Komplexität mit

SysLM (Systems Lifecycle Management). Springer Vieweg, Berlin Heidelberg, 2013

[6] Hüsken, M. (2003): *Evolution und Lernen zur Optimierung neuronaler Strukturen*. Dissertation, Fakultät für Physik und Astronomie, Ruhr-Universität Bochum, 2003

[7] Luhmann, N. (1984). *Soziale Systeme. Grundriß einer allgemeinen Theorie.* Suhrkamp, Frankfurt am Main, 1984, 2. Auflage

[8] Meier, A. (2010): *Relationale und postrelationale Datenbanken.* Springer, Heidelberg Dordrecht London New York, 7. Auflage, 2010

[9] Meinel, C.; Mundhenk, M. (2011): *Mathematische Grundlagen der Informatik. Mathematisches Denken und Beweisen. Eine Einführung.* Vieweg+Teubner, Springer Fachmedien, Wiesbaden, 2011

[10] Rose, O; März, L. (2011): *Simulation.* In: März (Hrsg.): Simulation und Optimierung in Produktion und Logistik. Praxisorientierter Leitfaden mit Fallbeispielen. Springer, Heidelberg Dordrecht London New York, 2011

[11] Russwurm, S. (2011). *Software: Die Zukunft der Industrie.* In: Sendler, U. (Hrsg.): Industrie 4.0 – Beherrschung der industriellen Komplexität mit SysLM (Systems Lifecycle Management). Springer Vieweg, Berlin Heidelberg, 2013

[12] Sendler, U. (2011). *Industrie 4.0 – Beherrschung der industriellen Komplexität mit SysLM (Systems Lifecycle Management).* Springer Vieweg, Berlin Heidelberg, 2013

[13] Vicknair, C.; Macias, M.; Zhao, Z.; Nan, X., Chen, Y.; Wilkins, D. (2010): *A Comparison of a Graph Database and a Relational Database. A Data Provenance Perspective.* Department of Computer and Information Science University of Mississippi. Downloaded 25.08.2014 via http://www.researchgate.com

Bereits veröffentlicht wurden in der Schriftenreihe des Instituts für Angewandte Informatik / Automatisierungstechnik bei KIT Scientific Publishing

1 BECK, S.
Ein Konzept zur automatischen Lösung von Entscheidungsproblemen bei Unsicherheit mittels der Theorie der unscharfen Mengen und der Evidenztheorie, 2005

2 MARTIN, J.
Ein Beitrag zur Integration von Sensoren in eine anthropomorphe künstliche Hand mit flexiblen Fluidaktoren, 2004

3 TRAICHEL, A.
Neue Verfahren zur Modellierung nichtlinearer thermodynamischer Prozesse in einem Druckbehälter mit siedendem Wasser-Dampf Gemisch bei negativen Drucktransienten, 2005

4 LOOSE, T.
Konzept für eine modellgestützte Diagnostik mittels Data Mining am Beispiel der Bewegungsanalyse, 2004

5 MATTHES, J.
Eine neue Methode zur Quellenlokalisierung auf der Basis räumlich verteilter, punktweiser Konzentrationsmessungen, 2004

6 MIKUT, R.; Reischl, M. (HRSG.)
Proceedings – 14. Workshop Fuzzy-Systeme und Computational Intelligence
Dortmund, 10. - 12. November 2004

7 ZIPSER, S.
Beitrag zur modellbasierten Regelung von Verbrennungsprozessen, 2004

8 STADLER, A.
Ein Beitrag zur Ableitung regelbasierter Modelle aus Zeitreihen, 2005

9 MIKUT, R.; REISCHL, M. (HRSG.)
Proceedings – 15. Workshop Computational Intelligence
Dortmund, 16. - 18. November 2005

10 BÄR, M.
µFEMOS – Mikro-Fertigungstechniken für hybride mikrooptische Sensoren, 2005

11 SCHAUDEL, F.
Entropie- und Störungssensitivität als neues Kriterium zum Vergleich verschiedener Entscheidungskalküle, 2006

12 SCHABLOWSKI-TRAUTMANN, M.
Konzept zur Analyse der Lokomotion auf dem Laufband bei inkompletter Querschnittlähmung mit Verfahren der nichtlinearen Dynamik, 2006

13 REISCHL, M.
Ein Verfahren zum automatischen Entwurf von Mensch-Maschine-Schnittstellen am Beispiel myoelektrischer Handprothesen, 2006

14 KOKER, T.
Konzeption und Realisierung einer neuen Prozesskette zur Integration von Kohlenstoff-Nanoröhren über Handhabung in technische Anwendungen, 2007

15 MIKUT, R.; REISCHL, M. (HRSG.)
Proceedings – 16. Workshop Computational Intelligence
Dortmund, 29. November - 1. Dezember 2006

16 LI, S.
Entwicklung eines Verfahrens zur Automatisierung der CAD/CAM-Kette in der Einzelfertigung am Beispiel von Mauerwerksteinen, 2007

17 BERGEMANN, M.
Neues mechatronisches System für die Wiederherstellung der Akkommodationsfähigkeit des menschlichen Auges, 2007

18 HEINTZ, R.
Neues Verfahren zur invarianten Objekterkennung und -lokalisierung auf der Basis lokaler Merkmale, 2007

19 RUCHTER, M.
A New Concept for Mobile Environmental Education, 2007

20 MIKUT, R.; Reischl, M. (HRSG.)
Proceedings – 17. Workshop Computational Intelligence
Dortmund, 5. - 7. Dezember 2007

21 LEHMANN, A.
Neues Konzept zur Planung, Ausführung und Überwachung von Roboteraufgaben mit hierarchischen Petri-Netzen, 2008

22 MIKUT, R.
Data Mining in der Medizin und Medizintechnik, 2008

23 KLINK, S.
Neues System zur Erfassung des Akkommodationsbedarfs im menschlichen Auge, 2008

24 MIKUT, R.; REISCHL, M. (HRSG.)
Proceedings – 18. Workshop Computational Intelligence
Dortmund, 3. - 5. Dezember 2008

25 WANG, L.
Virtual environments for grid computing, 2009

26 BURMEISTER, O.
Entwicklung von Klassifikatoren zur Analyse und Interpretation zeitvarianter Signale und deren Anwendung auf Biosignale, 2009

27 DICKERHOF, M.
Ein neues Konzept für das bedarfsgerechte Informations- und Wissensmanagement in Unternehmenskooperationen der Multimaterial-Mikrosystemtechnik, 2009

28 MACK, G.
Eine neue Methodik zur modellbasierten Bestimmung dynamischer Betriebslasten im mechatronischen Fahrwerkentwicklungsprozess, 2009

29 HOFFMANN, F.; HÜLLERMEIER, E. (HRSG.)
Proceedings – 19. Workshop Computational Intelligence Dortmund, 2. - 4. Dezember 2009

30 GRAUER, M.
Neue Methodik zur Planung globaler Produktionsverbünde unter BerücKsichtigung der Einflussgrößen Produktdesign, Prozessgestaltung und Standortentscheidung, 2009

31 SCHINDLER, A.
Neue Konzeption und erstmalige Realisierung eines aktiven Fahrwerks mit Preview-Strategie, 2009

32 BLUME, C.; JAKOB, W.
GLEAN. General Learning Evolutionary Algorithm and Method
Ein Evolutionärer Algorithmus und seine Anwendungen, 2009

33 HOFFMANN, F.; HÜLLERMEIER, E. (HRSG.)
Proceedings – 20. Workshop Computational Intelligence Dortmund, 1. - 3. Dezember 2010

34 WERLING, M.
Ein neues Konzept für die Trajektoriengenerierung und -stabilisierung in zeitkritischen Verkehrsszenarien, 2011

35 KÖVARI, L.
Konzeption und Realisierung eines neuen Systems zur produktbegleitenden virtuellen Inbetriebnahme komplexer Förderanlagen, 2011

36 GSPANN, T. S.
Ein neues Konzept für die Anwendung von einwandigen Kohlenstoff-nanoröhren für die pH-Sensorik, 2011

37 LUTZ, R.
Neues Konzept zur 2D- und 3D-Visualisierung kontinuierlicher, multidimensionaler, meteorologischer Satellitendaten, 2011

38 BOLL, M.-T.
Ein neues Konzept zur automatisierten Bewertung von Fertigkeiten in der minimal invasiven Chirurgie für Virtual Reality Simulatoren in Grid-Umgebungen, 2011

39 GRUBE, M.
Ein neues Konzept zur Diagnose elektrochemischer Sensoren am Beispiel von pH-Glaselektroden, 2011

40 HOFFMANN, F.; Hüllermeier, E. (HRSG.)
Proceedings – 21. Workshop Computational Intelligence Dortmund, 1. - 2. Dezember 2011

41 KAUFMANN, M.
Ein Beitrag zur Informationsverarbeitung in mechatronischen Systemen, 2012

42 NAGEL, J.
Neues Konzept für die bedarfsgerechte Energieversorgung
des Künstlichen Akkommodationssystems, 2012

43 RHEINSCHMITT, L.
Erstmaliger Gesamtentwurf und Realisierung der Systemintegration
für das Künstliche Akkommodationssystem, 2012

44 BRÜCKNER, B. W.
Neue Methodik zur Modellierung und zum Entwurf keramischer Aktorelemente, 2012

45 HOFFMANN, F.; Hüllermeier, E. (HRSG.)
Proceedings – 22. Workshop Computational
Intelligence Dortmund, 6. - 7. Dezember 2012

46 HOFFMANN, F.; Hüllermeier, E. (HRSG.)
Proceedings – 23. Workshop Computational
Intelligence Dortmund, 5. - 6. Dezember 2013

47 SCHILL, O.
Konzept zur automatisierten Anpassung der neuronalen Schnittstellen
bei nichtinvasiven Neuroprothesen, 2014

48 BAUER, C.
Neues Konzept zur Bewegungsanalyse und -synthese für Humanoide
Roboter basierend auf Vorbildern aus der Biologie, 2014

49 WAIBEL, P.
Konzeption von Verfahren zur kamerabasierten Analyse und Optimierung
von Drehrohrofenprozessen, 2014

50 HOFFMANN, F.; HÜLLERMEIER, E. (HRSG.)
Proceedings. 24. Workshop Computational Intelligence,
Dortmund, 27. - 28. November 2014

Die Schriften sind als PDF frei verfügbar, eine Nachbestellung der Printversion ist möglich.
Nähere Informationen unter www.ksp.kit.edu.

Dieser Tagungsband enthält die Beiträge des 24. Workshops „Computational Intelligence“ des Fachausschusses 5.14 der VDI/VDE-Gesellschaft für Mess- und Automatisierungstechnik (GMA) und der Fachgruppe „Fuzzy-Systeme und Soft-Computing“ der Gesellschaft für Informatik (GI), der vom 27. – 28. November 2014 in Dortmund stattfindet.

Der GMA-Fachausschuss 5.14 „Computational Intelligence“ entstand 2005 aus den bisherigen Fachausschüssen „Neuronale Netze und Evolutionäre Algorithmen“ (FA 5.21) sowie „Fuzzy Control“ (FA 5.22). Der Workshop steht in der Tradition der bisherigen Fuzzy-Workshops, hat aber seinen Fokus in den letzten Jahren schrittweise erweitert.

Die Schwerpunkte sind Methoden, Anwendungen und Tools für
- Fuzzy-Systeme,
- Künstliche Neuronale Netze,
- Evolutionäre Algorithmen und
- Data-Mining-Verfahren

sowie der Methodenvergleich anhand von industriellen und Benchmark-Problemen.

Die Ergebnisse werden von Teilnehmern aus Hochschulen, Forschungseinrichtungen und der Industrie in einer offenen Atmosphäre intensiv diskutiert. Dabei ist es gute Tradition, auch neue Ansätze und Ideen bereits in einem frühen Entwicklungsstadium vorzustellen, in dem sie noch nicht vollständig ausgereift sind.

ISSN 1614-5267 | ISBN 978-3-7315-0275-3