Online-Zustandstracking mit datengetriebenen Prozessmodellen

Zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften

der Fakultät für Maschinenbau
Karlsruher Institut für Technologie (KIT)

genehmigte

Dissertation

von

Dipl.-Inform. Susanne Witt

Tag der mündlichen Prüfung: 26. April 2018

Hauptreferent: Prof. Dr.-Ing. habil. Volker Schulze
Korreferent: Prof. Dr. rer. nat. Norbert Link

Kurzfassung

Durch immer weiter steigende Rechnerleistung und dadurch verbundene Möglichkeiten zur Modellierung und Simulation komplexer Fertigungsprozesse steigt die Herausforderung, diese aufwendig berechneten Modelle in reduzierte online-fähige Modelle zu übertragen, sodass diese z.B. zur Zustandsregelung eingesetzt werden können. Im Allgemeinen gilt: Je präziser die Simulation, desto zeitaufwendiger die Berechnungen. Online Zustandsverfolgung, sprich die Verfolgung des Bauteilzustandes im laufenden Prozess, ist bedeutend für effiziente automatisierte Regelung in der Fertigung.

Diese Arbeit stellt sich der Herausforderung, reduzierte nichtlineare Prozessmodelle mittels Methoden der künstlichen Intelligenz, im Speziellen neuronaler Netze und genetischer Programmierung, zu erlernen. Hierfür werden Daten aus Finite-Elemente-Simulationen zur Datenerzeugung verwendet. Durch die erlernten datengetriebenen Prozessmodelle soll der nicht messbare Zustand vorhergesagt werden.

Das reduzierte nichtlineare Prozessmodell entsteht durch die Reduzierung der Zustandsvariablen auf wenige charakteristische Merkmale. Dies geschieht durch eine hier vorgestellte Architektur für neuronale Netze, die offline eine nichtlineare Dimensionsreduktion durchführt. Dabei repräsentiert ein extrahiertes Merkmal charakteristische Eigenschaften einer Menge von Zustandsvariablen zu einem bestimmten Zeitpunkt und bei bestimmtem Prozessparameter bei gegebenen Randbedingungen. Der Zustand wird somit vom Zustandsraum in einen wesentlich geringer dimensionierten Merkmalsraum transformiert.

Für die extrahierten Merkmale werden mittels symbolischer Regression unter Verwendung genetischer Programmierung mathematische Beschreibungen erlernt. Diese Beschreibungen ersetzen das theoretische Modell bei der Zustandsbeobachtung. Für die Zustandsbeobachtung ist zusätzlich ein Messmodell nötig. Ein Messmodell bildet den nicht messbaren vorhergesagten Zustand auf die Messgrößen ab, um diese mit den wahren Messwerten des Prozesses zu vergleichen, um diese Differenz dann als Korrekturwert für die Zustandsbeobachtung zu verwenden.

Es werden zwei Prozesse betrachtet: Ein Wärmeleitungsprozess, für den ein analytisches Modell zur Datengenerierung vorliegt, und ein Tiefziehprozess, für den ein zwei- und ein dreidimensionales Finite-Elemente-Modell vorliegt. Es kann gezeigt werden, dass reduzierte Surrogat-Modelle für die drei Anwendungsfälle erlernt werden konnten. Beim zweidimensionalen Modell wird z. B. die Dimension um den Faktor 133 reduziert. Das beschriebene Modell im reduzierten Raum kann 88% der Daten des Finite-Elemente-Modells rekonstruieren.

Abstract

With the rising availability of computing performance and the corresponding possibility to model and simulate complex manufacturing processes, the challenge of transferring these complex models into reduced online models is rising. These models can be used, for example, for state control. In general, the more precise the simulation, the more time-consuming the computations. Online state tracking, i. e., the tracking of the state of the workpiece during the running manufacturing process, is important for efficient automated control in production.

This work addresses the challenge of learning a reduced nonlinear process model using artificial intelligence methods, in particular neural networks and genetic programming. For this purpose, data from finite element simulations are used to generate data for training the neural network. The non-measurable state is to be predicted by these data-driven process models.

The reduced non-linear process model results from the reduction of the state variables to a few characteristic features. This is done by means of a neural network architecture presented here, which performs a non-linear dimensional reduction offline. An extracted feature represents characteristic properties of a set of state variables at a particular point in time and certain process parameters at given boundary conditions. The state is thus transformed from the state space into a considerably smaller dimensioned feature space.

For the extracted features, mathematical descriptions are learned with symbolic regression using genetic programming. These descriptions replace the theoretical model in state observation. A measuring model is also required for state observation. The measurement model maps the non-measurable predicted state to the observables in order to compare these with the true observables of the process. The difference is then used as a correction value for the state observation.

Two processes are considered: A heat conduction process for which an analytical model for data generation is available and a deep drawing process for which a two-dimensional and a three-dimensional finite element model is available. It can be shown that reduced surrogate models for the three applications could be learned. For example, for the two-dimensional model, the dimension is reduced by a factor of 133. The described model in the reduced space can reproduce 88% of the state variables computed by the finite element model.

Danksagung

Die vorliegende Dissertation entstand während meiner Zeit als wissenschaftliche Mitarbeiterin am Institut für Angewandte Forschung der Hochschule Karlsruhe – Technik und Wirtschaft. Im Rahmen des gemeinsamem Graduiertenkollegs 1483 des Karlsruher Institut für Technologie (KIT) und der Hochschule Karlsruhe wurde diese Arbeit durch die Deutsche Forschungsgemeinschaft finanziert.

Auf den Weg zur Fertigstellung dieser Arbeit haben mich Menschen begleitet, denen mein besonderen Dank gilt. Allen voran möchte ich mich bei Prof. Dr.-Ing. habil. Volker Schulze für die Übernahme des Hauptreferats und die wertvollen Gespräche, die immer wieder neue Impulse für meine Arbeit gegeben haben, bedanken. Ebenso bei Prof. Dr. rer. nat. Norbert Link für die fachliche Unterstützung und die Chance, in seiner Arbeitsgruppe mitzuwirken und das damit verbundene Vertrauen in mich und meine Arbeit.

Zur Bearbeitung dieses interdisziplinären Themas war die Gemeinschaft der Kollegiaten des Graduiertenkollegs 1483 und deren unterschiedlichen Fachbereiche eine große Bereicherung. Insbesondere danke ich hier Jan Pagenkopf für das Bereitstellen des dreidimensionalen Modells, Florian Rieger und Kollegen vom Institut für Technische Mechanik des KIT für die Einführung in die Finite-Elemente-Methode und Daniel Schneider für seine vielen Fragen.

Bei meinen ehemaligen Kollegen am Institut für Angewandte Forschung der Hochschule Karlsruhe möchte ich mich ebenso für die Unterstützung bedanken; insbesondere Melanie Senn, die mich in die interdisziplinäre Thematik eingeführt hat, Onno Hensgen für seine Vorarbeit während seiner Master-Thesis und Jürgen Pollack für das offene Ohr bei kniffligen Fragen. Mein Dank geht auch an Moustafa Elshaabiny, der mich durch das RISE Programms des DAAD unterstützen konnte.

Einige Lebensabschnitte lassen sich am besten gemeinsam meistern. Freunden und Familie kann man hier nicht genug danken für die Unterstützung und Geduld, die sie während der Zeit dieser Arbeit aufgebracht haben. Besonders möchte ich Yoo-Jin Azad danken, die seit so vielen Jahren schon immer die passendsten Worte für mich bereit hat und sich so oft mit meiner To-do-Liste auseinander gesetzt hat. Vor allem danke ich auch meinem Mann Sascha Witt, der mich nicht nur bei der Durchsicht dieser Arbeit unterstützt hat, sondern auch immer daran glaubt, dass ich alles schaffen kann, was ich mir vornehme.

Karlsruhe, im Oktober 2018 *Susanne Witt*

Für Mathilda und Leopold.

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Motivation 1
1.2 Frühere Arbeiten 2
1.3 Zielsetzung und Aufgabenanalyse 3
1.4 Kapitelübersicht 5

2 Bildung datengetriebener Prozessmodelle **7**
2.1 Datenerzeugung 7
2.1.1 Finite-Elemente-Methode zur Datenerzeugung 8
2.1.2 Multiskalenmodellierung 9
2.2 Grundlagen: Modellbildung 10
2.2.1 Modelleigenschaften 11
2.2.2 Überblick über Verfahren zur Modellbildung 13
2.3 Künstliche neuronale Netze 16
2.4 Datengetriebene Dimensionsreduktion 20
2.4.1 Hauptkomponentenanalyse 21
2.4.2 Autoencoder 23
2.5 Principal Function Approximator 24
2.6 Symbolische Regression 27
2.7 Zusammenfassung 28

3 Zustandsverfolgung nichtlinearer Prozesse **31**
3.1 Grundlagen: Lineare Zustandsbeobachter 32
3.2 Erweitertes Kalman-Filter 34
3.3 Unscented Kalman-Filter 36
3.4 Beobachter basierend auf Black-Box-Modell 37
3.5 Zusammenfassung 41

4 Lösungsansatz und Architektur **43**

5 Online-Zustandstracking mit datengetriebenen Prozessmodellen **47**
5.1 Reduktion der Zustandsvariablen 47
5.1.1 Datenvorverarbeitung 50
5.1.2 Sequenzielles bottleneck neuronales Netz 51
5.1.3 Sequenzielles bottleneck neuronales Netz für Zeitreihen 54
5.1.4 Konfiguration und Training des Netzes 56

5.2 Identifikation des reduzierten Prozessmodells . . . 62
5.2.1 Prozessmodell . . . 62
5.2.2 Messmodell . . . 64
5.3 Beobachtung des Zustandes . . . 65
5.3.1 Beobachtbarkeit nichtlinearer Systeme . . . 66
5.3.2 Beobachtung mit EKF und UKF . . . 67

6 Anwendung . . . 71
6.1 Wärmeleitung . . . 72
6.2 Tiefziehen . . . 74

7 Evaluierung . . . 87
7.1 Reduktion der Zustandsvariablen . . . 87
7.2 Identifikation des reduzierten Prozessmodells . . . 102
7.3 Beobachtung des Zustandes . . . 115
7.4 Zusammenfassung . . . 121

8 Schlussfolgerung und Ausblick . . . 123

A Grundlagen des Kalman-Filters . . . 125
A.1 Hintergrund und Aufbau . . . 125
A.2 Berechnung der Sigmapunkte und Gewichte beim Unscented Kalman-Filter . . . 126

B Mathematische Definitionen . . . 127
B.1 Jacobi-Matrix . . . 127
B.2 Lie-Ableitung . . . 127

C Ergebnisse Dimensionsreduktion . . . 129
C.1 Reduktion von Mises Spannung . . . 129
C.2 Reduktion plastische Dehnung . . . 129

Abkürzungen und Symbole . . . 135

Abbildungsverzeichnis . . . 143

Tabellenverzeichnis . . . 145

Literaturverzeichnis . . . 147

1 Einleitung

1.1 Motivation

Die Produktion ist ständig wachsenden Anforderung an Effizienz, Flexibilität und Zuverlässigkeit ausgesetzt. Um wettbewerbsfähig zu bleiben, ist es wichtig, hohe Qualität bei niedrigen Kosten zu produzieren. Simulationen in der Fertigungstechnik ermöglichen es, durch die Modellierung der Prozesse und deren Wechselwirkungen aufwendige experimentelle Untersuchungen zu reduzieren und die Produktentwicklung deutlich zu beschleunigen.

Im Graduiertenkolleg 1483 der Deutschen Forschungsgemeinschaft „Prozessketten in der Fertigung“ werden Simulationsmethoden zur Beschreibung, Bewertung und Optimierung von Bauteilzuständen entwickelt und durch experimentelle Untersuchungen verifiziert. Ein Themenschwerpunkt im Graduiertenkolleg ist die Prozesskette Walzen – Glühen – Tiefziehen mit dem Werkstoff Stahl.

Für die Regelung der einzelnen Prozesse bzw. der Prozesskette werden zeitnah konkrete Informationen über den aktuellen Zustand des Bauteils benötigt, um während des laufenden Prozesses Regeleingriffe vornehmen zu können, sodass der optimale Prozesspfad realisiert werden kann. Um ein bestmögliches Ergebnis zu erreichen, kommen mathematische Beschreibungen zum Einsatz, die das Verhalten von bekannten Eingangsgrößen (Regelgrößen) zu messbaren Ausgangsgrößen (Observablen) in Form einer Übertragungsfunktion modellieren. Eine tiefere Einsicht in die Regelung komplexer Systeme bieten Zustandsregler, da diese nicht nur das Eingangs-Ausgangsverhalten modellieren, sondern das System auch durch das Verfolgen von nicht direkt messbaren Zustandsgrößen beschreiben. Zustandsregler eignen sich besonders für Systeme höherer Ordnung, nichtlineare Systeme sowie Mehrgrößensysteme. Für den Prozess des Tiefziehens gilt es beispielsweise, die Niederhalterkraft so einzustellen, dass u. a. kein Materialversagen im Bauteil, weder im Prozess noch in der späteren Nutzung, auftritt. Kapitel 6.2 geht näher auf den Prozess des Tiefziehens und mögliche Versagensgrenzen ein.

Um den nicht messbaren Zustand für den Zustandsregler zu verfolgen, kommen Zustandsbeobachter zum Einsatz. Ein Zustandsbeobachter im klassischen Sinne rekonstruiert den Zustand anhand der Eingangsdaten und des dynamischen Prozessmodells, das die Zustandsentwicklung aus dem vorherigen Zeitschritt darstellt. Die Synchronisation von Prozess und Beobachter erfolgt durch die Angleichung der Prozessobservablen und der durch ein zusätzliches Messmodell vorhergesagten Observablen. Sind die Observablen fehlerhaft oder fehlen gänzlich, kann der Zustandsbeobachter darauf reagieren.

Die Herausforderung bei der Entwicklung eines Zustandsbeobachters ist die Modellierung des nichtlinearen Prozesses. Hierbei ist das Ziel, durch das Modell den Zustand so präzise wie nötig vorhersagen zu können. Finite-Elemente-Prozesssimulationen ermöglichen es, detaillierte Zustandsentwicklungen zu erfassen und somit Zustandsvariablen zu extrahieren, die durch eine

direkte Messung im laufenden Prozess nicht ermittelt werden können. Die Verwendung der FE-Modelle zur Zustandsregelung ist jedoch durch den hohen Berechnungsaufwand nicht möglich. Bei der Verwendung bzw. Modellierung datengetriebener Prozessmodelle können jedoch die Zustandsvariablen und Observablen aus numerischen Simulationen als Daten verwendet werden, um so präzisere nichtlineare onlinefähige Prozessmodelle und ggf. Messmodelle zu beschreiben.

1.2 Frühere Arbeiten

Die Modellierung des Systems kann empirisch, analytisch oder numerisch erfolgen. Im Sinne der *empirischen* Prozessmodellierung durch reale Experimente haben Hsu et al. [37] eine *Steuerung* für Umformprozesse entwickelt. Der optimale Prozesspfad wird danach durch eine Referenz-Stempelkraft-Trajektorie vorgegeben. Dabei wird versucht die Niederhalterkraft möglichst so einzustellen, dass die Differenz zwischen gemessener Stempelkraft und Referenz-Stempelkraft möglichst gering ist. Für einen *Regler* mit Rückkopplung des gemessenen Signals ersetzen Hsu et al. [38] ihre empirisch ermittelte Referenztrajektorie durch ein *analytisches* Modell, welches die Stempelkraft in Abhängigkeit von Niederhalterkraft und Zeit repräsentiert, wodurch die Stempelkraft durch Einstellung der Niederhalterkraft geregelt werden soll. Die Parameter des Modells werden abhängig von Blechgröße und -dicke, den Materialeigenschaften und der Werkzeugform experimentell bestimmt. Das Modell berücksichtigt zusätzlich Modellunsicherheiten und Störgrößen. Im Vergleich zur Steuerung ermöglicht die Regelung mit der Rückkopplung die Überwachung der Ausgangsgrößen und kann dadurch auch auf Störungen, die auf das System wirken, reagieren.

Neuere Arbeiten modellieren das Prozessmodell anhand von *numerischen* Simulationen. Endelt et al. [18] verwenden die *Finite-Elemente-Methode* (FEM), um die Beziehung zwischen Zustand und Prozesseingang (Niederhalterkraft) sowie die Referenz-Zustandstrajektorie zu ermitteln. Dabei verwenden sie als Zustand den Materialfluss. Die Senkung des Materialflusses führt zu Einschnürungen und die Erhöhung des Materialfluss führt zu Falten im Werkstück. Der Flanscheinzug, der hierbei als Indikator des Materialflusses gilt, kann online gemessen werden. Die Regelung versucht die Differenz zwischen Referenz-Trajektorie und gemessenem Systemausgang durch Einstellung der Niederhalterkraft zu minimieren.

Hsu et al. [38] sowie Endelt et al. [18] wählen messbare Werte als zu regelnde Größen. Bauteilzustände bzw. Materialverhalten das im Fall von Hsu et al. nicht durch die Stempelkraft und im Fall von Endelt et al. nicht durch den Flanscheinzug beschrieben werden können, kann auch nicht in die Regelung eingehen. Die Zustandsentwicklung des Blechbauteils ist mittels der Verfahren nicht abzulesen. Damit der Zustand optimal geführt werden kann, ist es sinnvoll, auch nicht direkt messbare Größen zur ganzheitlichen Betrachtung des Werkstückes mit einfließen zu lassen.

Senn [87] entwickelte auf Basis der Daten aus FE-Prozesssimulationen ein *Beobachtermodell*, um auch diese nicht online zugänglichen Zustandsvariablen parallel zum Prozess vorherzusagen. Dieses Beobachtermodell wird näher in Kapitel 3.4 beschrieben. FE-Simulationen ermöglichen die Wahl von Zustandsgrößen, die für die Prozessführung relevanter sind als Messgrößen (*Observablen*), wie z. B. Spannungen und Dehnungen, die durch die jeweiligen Elemente des FE-Modell im gesamten Werkstück beschrieben werden. Die Observablen, wie Kräfte und Verschiebungen,

sowie die Zustandsvariablen, wie Spannungen, werden dabei für jeden Simulationsschritt extrahiert und bilden die Datenbasis. Das Beobachtermodell wird dann offline mittels eines Regressionsverfahrens, das die Transformationsfunktion zwischen Observablen und Zustandsvariablen sucht, erlernt. Weiterhin verwendet Senn keine vorberechnete Trajektorie für die Prozessregelung, sondern ermittelt und verfolgt den optimalen Pfad zu jedem Zeitpunkt anhand des aktuell vorhergesagten Zustandes. Dies ermöglicht auch eine Regelung von Prozessen in Prozessketten, da nachfolgende Prozesse online an den Ausgangszustand der vorherigen Prozesse anknüpfen. Das so entstandene Black-Box-Modell – die innere Struktur des Systems wird nicht verwendet – zur Vorhersage des Zustandes deckt jedoch lediglich die durch zeitliche Diskretisierung abgedeckte Dynamik ab. Bereiche, die durch das Ermitteln der Transformationsfunktion, die auf den diskreten Daten beruhen, nicht abgedeckt wurden, fließen nicht mit in die Regelung ein. Da die Vorhersage des Zustandes allein auf den Observablen beruht, können zusätzlich fehlerhafte Messungen oder Messlatenzen zu falscher Vorhersage des Zustandes führen.

Es bleibt die Frage nach einem dynamischen, nichtlinearen Prozessmodell für einen klassischen Zustandsbeobachter, der parallel zum Prozess geschaltet ist und dabei Ungenauigkeiten, die aus der zeitlichen Diskretisierung entstehen, vermeidet.

1.3 Zielsetzung und Aufgabenanalyse

Ziel dieser Arbeit ist es, durch Daten aus numerischen Prozesssimulationen für den jeweiligen Prozess ein onlinefähiges, dynamisches, nichtlineares Prozessmodell zu ermitteln, welches den Prozess genügend genau darstellt, um dieses Surrogat-Modell zur Zustandsbeobachtung einsetzen zu können. Die numerischen Simulationen können den Prozesszustand sehr genau vorhersagen, sind aber sehr rechenintensiv und dadurch nicht zur online Verfolgung des Zustandes geeignet. Jedoch können die gewonnen Daten aus den Simulationen zur Erstellung eines reduzierten Surrogat-Modells verwendet werden, welches die Prozessdynamik des durch die Simulationen abgetasteten Bereich darstellt. Hierfür wird ein nichtlineares Reduktionsverfahren benötigt, welches die zeitabhängigen Zustandsvariablen auf eine wesentlich geringere Anzahl reduziert. Anhand der reduzierten Zustandsvariablen und deren zeitlicher Änderung soll ein nichtlineares dynamisches Prozessmodell erlernt werden. Dieses so ermittelte datengetriebene Prozessmodell soll einem Zustandsbeobachter sowie Zustandsregler zur Rekonstruktion des nicht messbaren Zustandes zur Verfügung stehen.

Die Zielsetzung teilt die Arbeit in folgende Schwerpunkte:

- Erzeugung von *Daten* aus numerischen Prozesssimulationen für die datengetriebene Prozessmodellierung. (Daten)
- *Reduktion* der gewonnenen Daten, die den hochdimensionalen Zustand repräsentieren, auf niedrigdimensionale Zustandsmerkmale. (Dimensionsreduktion)
- *Bildung* eines onlinefähigen dynamischen *Prozessmodells* als Komponente eines nichtlinearen Zustandsbeobachters zur Vorhersage des nicht messbaren (inneren) Zustandes. (Modellbildung)

- Gesamtsystem zur online *Beobachtung* des realen Zustandes durch rekonstruierte Zustandsvariablen mittels Prozessmodell und rekonstruierte Observablen mittels *Messmodell*. (Zustandsbeobachter)

Zustandsbeobachter: Zustandsbeobachter dienen zur Rekonstruktion des nicht messbaren Zustandes anhand der Eingangsgrößen (Prozessparameter) und des Anfangszustandes. Bedeutend ist die Information über den momentanen Zustand für einen Zustandsregler, der den Prozess so regelt, dass das Bauteil den gewünschten Zustand behält bzw. erreicht. Der Zustandsbeobachter muss folglich die Prozessdynamik mittels eines Prozessmodells genügend genau nachbilden, um den Zustand zu jedem Zeitpunkt rekonstruieren zu können. Um die Prozessdynamik vom Zustandsbeobachter anhand von Observablen korrigieren zu lassen, stellt der Zustandsbeobachter zusätzlich die Abbildung von rekonstruierten Zuständen auf Observablen nach. Für die in dieser Arbeit relevanten nichtlinearen Prozesse wird ein Zustandsbeobachter benötigt, der auch nichtlineare Prozessmodelle mit Hilfe von nichtlinearen Messmodellen korrigieren kann.

Modellbildung: Da nichtlineare Prozesse betrachtet werden, sollten das Prozessmodell sowie das Messmodell in der Lage sein, diese Nichtlinearität nachzubilden. Da der Zustand der dynamischen Prozesse in der Regel kontinuierlich beschrieben wird, ist die dynamische Funktion f gesucht, für die gilt:

$$\dot{\boldsymbol{x}}(t) = f\left(\boldsymbol{u}(t), \boldsymbol{x}(t), \nu(t)\right) \tag{1.1}$$

mit Zustandsvariablen $\boldsymbol{x}(t) \in \mathbb{R}^n$, Prozessparameter (bzw. Regelgröße) $\boldsymbol{u}(t) \in \mathbb{R}^m$ und Prozessrauschen $\nu(t)$. Da Messungen in zeitlich diskreten Abständen (Abtastungen) erfolgen, ergibt sich für das Messmodell folgende Darstellung:

$$\boldsymbol{y}_t = g(\boldsymbol{x}_t, \boldsymbol{\rho}_t) \tag{1.2}$$

mit Observablen $\boldsymbol{y}_t \in \mathbb{R}^r$ und Messrauschen ρ_t zum Zeitpunkt $t \in \{t_1, t_2, \ldots t_T\}$. Für die Modellbildung gilt, dass das Prozessmodell onlinefähig sein soll, um zeitnahe Lösungen für den nicht messbaren Zustand zu liefern. Je komplexer der Zustandsraum ist, desto aufwendiger ist die Berechnung der Entwicklung der Zustandsvariablen. Folglich wird ein Verfahren oder eine Herangehensweise gesucht, die es ermöglicht, onlinefähige Beschreibungen des Zustandsraums zu ermitteln, die somit weniger rechenintensiv sind als die numerischen Methoden, mit welchen physikalische Prozess- und Messmodelle gelöst werden. Die zusätzliche Herausforderung hierbei soll sein, dass für die Modellstruktur bzw. die Dynamik des Prozessverhaltens physikalisch motivierte Interpretationen getroffen werden können.

Dimensionsreduktion: Es wird angenommen, dass für einen bestimmten Prozess die Zustandsvariablen im Wesentlichen in einem nichtlinearen Unterraum des „universellen“ Zustandsraumes liegen, sodass die Zustandsvariablen reduziert werden können, ohne einen gravierenden Informationsverlust zu erleiden. Dieser Unterraum sollte ausreichend sein, um den gesamten Zustand eines betrachteten Prozesses zu einem bestimmten Zeitpunkt zu beschreiben. In [21] haben wir die Hauptanforderungen an das Dimensionsreduktionsverfahren wie folgt formuliert:

- *Merkmalextraktion:* Die Zustandsvariablen werden auf wenige orthogonale und nach Wertigkeit geordnete Merkmale reduziert, um Korrelationen und redundante Informationen in den Daten zu beseitigen.
- *Rekonstruktion:* Die ursprünglichen Zustandsvariablen können aus den extrahierten Merkmalen rekonstruiert werden, um das Verfahren zu validieren.
- *Nichtlinearität:* Es wird angenommen, dass ein nichtlineares Verfahren, welches nichtlineare Merkmale extrahiert, die Daten besser approximiert als lineare Methoden.

Daten: Zur Evaluierung des Verfahrens stehen folgende Prozesse unterschiedlicher Komplexität zur Verfügung: Für den Machbarkeitsbeweis ein einfacher Wärmeleitungsprozess, für den bereits ein analytisches Modell vorhanden ist, sowie, für die eigentliche Anwendung im Bereich des Graduiertenkolleg 1483, der Prozess des Tiefziehens. Für diesen Prozess liegt ein zweidimensionales sowie ein komplexeres dreidimensionales Finite-Elemente-Modell vor. Diese Modelle sollen zur Datengenerierung (Berechnung von Zustandsgrößen und Observablen) sowie zur Evaluierung des aus den Daten gewonnenen reduzierten Modells dienen.

Für das Finite-Elemente-Modell des Tiefziehens werden in dieser Arbeit folglich numerische Prozesssimulationen durchgeführt, um Zustandsgrößen wie Spannungen und Dehnungen sowie Observablen wie Kräfte und Verschiebungen zu ermitteln. Aus dieser Datenbasis, aus Zustandsvariablen und Observablen, werden ein reduziertes nichtlineares dynamisches Prozessmodell und ein Messmodell erlernt. Zum Einsatz kommen diese Modelle in einem nichtlinearen Zustandsbeobachter, der anhand der beiden Modelle den Zustand verfolgt und auf Ungenauigkeiten bzw. Prozess- und Messrauschen eingeht. Diese Herangehensweise ermöglicht es, das Wissen aus aufwendig berechneten numerischen Simulationen für die Prozessoptimierung in Form von nichtlinearen und dynamischen Prozessmodellen und ggf. nichtlinearen Messmodellen nutzbar zu machen. Dadurch wird es möglich, mit Hilfe dynamischer Programmierung eine Zustandsregelung auf Grundlage nicht messbarer Zustandsgrößen zu nutzen, welche ein optimales Ergebnis mit minimalem Aufwand entlang des Prozesspfades erzielt.

1.4 Kapitelübersicht

Die nachfolgenden vier Kapitel beschreiben die erarbeitete Lösungsmethode auf Grundlage relevanter Verfahren aus dem Stand der Forschung. Kapitel 2 behandelt dabei den Schwerpunkt Dimensionsreduktion und Modellidentifikation, zeigt die Relevanz der Datenerzeugung mittels Finite-Elemente-Methode und erläutert Grundlagen der datengetriebenen Modellbildung. Bei den Dimensionsreduktionsverfahren liegt der Schwerpunkt auf jenen Verfahren, die aus Daten repräsentative Merkmale extrahieren und aus diesen Merkmalen die ursprünglichen Daten wieder rekonstruieren können. Ein tieferer Einblick wird dabei in künstliche neuronale Netze und Varianten dieser gegeben, da diese sowohl zur Dimensionsreduktion wie auch zur Modellidentifikation verwendet werden können. Als weiteres Verfahren zur Modellidentifikation wird symbolische Regression aufgeführt und erläutert.

Zur Zustandsverfolgung nichtlinearer Prozesse werden in Kapitel 3 Verfahren betrachtet, die es ermöglichen, den Zustand nichtlinearer Prozesse zu schätzen bzw. zu beobachten. Dafür werden

zuerst Grundlagen zu linearen Zustandsbeobachtern behandelt, um darauf aufbauend den Ablauf nichtlinearer Zustandsbeobachter bzw. Zustandsschätzer zu betrachten.

Kapitel 4 diskutiert die Anforderung an das Gesamtsystem – der online Zustandsverfolgung mit datengetriebenen Prozessmodellen – anhand der Grundlagen aus Kapitel 2 und 3 sowie der Zielsetzung aus Abschnitt 1.3. Die Erkenntnisse aus aktueller Forschung und die Vor- und Nachteile der Verfahren werden einbezogen, um notwendige Komponenten zu ermitteln.

In Kapitel 5 wird die Methodik zur Entwicklung eines onlinefähigen Zustandstrackers (im Verlauf der Arbeit auch Zustandsverfolger genannt) vorgestellt. Zuerst wird die Vorverarbeitung der Daten aus analytischen und numerischen Prozesssimulationen erläutert. Als Nächstes wird das entwickelte Dimensionsreduktionsverfahren beschrieben, welches die zeitvarianten Prozesszustandsvariablen auf charakteristische Merkmale reduziert. Diese reduzierten Merkmale dienen im nächsten Schritt zur Ermittlung eines reduzierten dynamischen Prozessmodells, das sich durch die Verwendung symbolischer Regression durch Parameterschätzung und Modellstrukturschätzung auszeichnet, um die Beziehung zwischen Prozesszustandsänderungen und Prozesskontrollparametern abzubilden. Zuletzt wird die Eingliederung des reduzierten Prozessmodells im Kontext des Beobachters dargestellt, der durch die Beziehung zwischen beobachtbaren Prozessobservablen und Prozesszustandsvariablen eine Korrektur der Zustandsschätzung vornimmt.

Als Anwendungsbeispiele werden in Kapitel 6 ein analytisches Modell zur Wärmeleitung und zwei numerische Modelle zum Tiefziehen vorgestellt. Dabei werden im Speziellen die Problemstellung, die Entstehung des Modells und die Ergebnisse, die diese Modelle für unterschiedliche Prozessparameter erzeugen, betrachtet.

Anhand dieser Beispielprozesse wird in Kapitel 7 die Systemlösung validiert. Hierfür werden Fehlermaße festgelegt, um einheitliche Aussagen über die jeweiligen Approximationen treffen zu können. Das für die Aufgabe des Zustandsbeobachters entwickelte Verfahren zur Dimensionsreduktion wird mit Methoden zur Dimensionsreduktion aus Kapitel 2.4 verglichen. Die auf den reduzierten Daten basierenden Lösungen für das reduzierte Prozessmodell werden auf Robustheit und Performanz überprüft. Des Weiteren wird untersucht, ob die Lösungen für das Prozessmodell des jeweiligen Prozesses so interpretierbar sind, dass sie Rückschlüsse auf physikalisches Verhalten des Prozesses erlauben. Das erlernte Prozessmodell wird auf Beobachtbarkeit und Steuerbarkeit überprüft und anschließend in den Zustandsbeobachter integriert und evaluiert. Kapitel 8 fasst die Schlussfolgerungen der Arbeit zusammen und gibt einen Ausblick auf zukünftige Verwendung.

2 Bildung datengetriebener Prozessmodelle

Onlinefähige Prozessmodelle dienen zur Zustandsbeobachtung und Regelung, wenn der Zustand des Prozesses nicht direkt oder nur mit erheblichem Aufwand messbar ist. Das Modell ermöglicht die Vorhersage des Zustandes, der durch seine Zustandsvariablen beschrieben wird. Für Prozesse mit hoher Komplexität ermöglichen datengetriebene Prozessmodelle eine ganzheitliche online Betrachtung des Prozesses. Datengetriebene Prozessmodelle sind Modelle, die mittels Verfahren der *Computational Intelligence* oder des *Maschinellen Lernens* auf Grundlage von Daten generiert werden.

Genutzt werden können Daten aus Messungen des Prozesses, aus Experimenten oder auch aus bereits vorhandenen numerischen Simulationen. Der Einsatz von numerischen Simulationen zur Ermittlung von onlinefähigen Prozessmodellen bzw., im weiteren Sinne, Prozesszustandsbeobachtern (Kapitel 3), ermöglicht es, aufwendig berechnetes Wissen über den Prozesszustand unter gegebenen Randbedingungen mit in die Regelung des Prozesses einfließen zu lassen.

Dieses Kapitel gliedert sich in drei Teile. Zuerst wird die Datenerzeugung aus numerischen Simulation vorgestellt. Danach werden allgemeine Grundlagen zur Modellbildung im Zustandsraum behandelt, inklusive eines Überblicks über mögliche Verfahren zur Erstellung datengetriebener Prozessmodelle. Der dritte Teil beinhaltet mehrere Unterkapitel mit Grundlagen zu relevanten Verfahren, wie künstlichen neuronalen Netze, symbolischer Regression und dem Principal Function Approximator.

Die vorgestellten Verfahren eignen sich für nichtlineare Mehrgrößensysteme, das heißt Systeme, die mehrere Ein- und Ausgangsgrößen besitzen können. Die Anzahl der Zustandsvariablen gibt die Dimension des Zustandsraums, in dem das Modell beschrieben wird, vor. Diese Dimension kann sehr hoch sein und folglich ungeeignet für das Ziel eines onlinefähigen Modells. Deshalb werden in diesem Kapitel auch Dimensionsreduktionsverfahren vorgestellt, die entweder direkt ein reduziertes Modell erzeugen oder die Anzahl der Zustandsvariablen so reduzieren, dass darauf aufbauend ein Verfahren eingesetzt werden kann, dass eine geringere Dimension voraussetzt.

2.1 Datenerzeugung

Datengetriebene Methoden ermöglichen es, aus bereits aufwendig berechneten Daten oder Experimenten Modelle zu erlernen. Erlernen bedeutet in diesem Sinne, dass aus den gegebenen Daten mit Hilfe des maschinellen Lernens Modelle erzeugt werden, die die Beziehungen dieser Daten untereinander abbilden. Diese Modelle können für gegebene Eingangsdaten die möglichen zugehörigen Informationen als Ausgangswerte vorhersagen. Die Modellierung geschieht während des *Trainings*, wobei für gegebene Ein- und Ausgangsdaten die nötige Abbildungsfunktion erlernt wird.

Diese Herangehensweise benötigt eine Vielzahl von Daten, um eine möglichst allgemeine Darstellung der Abbildungsfunktion zu erlernen, damit auch Vorhersagen für unbekannte Daten im gegebenen Gültigkeitsbereich getroffen werden können. Der Gültigkeitsbereich wird vor der Datenerzeugung definiert, um geeignete *Stichproben* für das Training des Modells zu wählen. Stichproben sind in dieser Arbeit zeitlich-diskrete Werte für einen oder mehrere Parameter berechnet durch Prozesssimulationen.

Das Ziel datengetriebener Verfahren sind schnelle, robuste Modelle. Die Interpretierbarkeit der Parameter, z. B. mit physikalischen Größen, steht nicht im Vordergrund. Jedoch ist es von großer Bedeutung, eine ausreichend genaue Abbildung bzgl. der relevanten Daten zu finden.

Nachfolgend wird die Gewinnung von Daten aus theoretischen Modellen, die mittels der *Finite-Elemente-Methode* (FEM) erstellt werden, erläutert. Weiterhin wird in Abschnitt 2.1.2 die Relevanz der Multiskalenmodellierung generell und insbesondere im Rahmen des Graduiertenkollegs geschildert.

2.1.1 Finite-Elemente-Methode zur Datenerzeugung

Die Lösung physikalisch-theoretischer Modelle mittels der Finite-Elemente-Methode ist in der Regel rechenintensiv und eignet sich daher nicht zur online Zustandsverfolgung. Jedoch können sie zur Datenerzeugung dienen, indem Simulationen mittels FEM für unterschiedliche Prozessparameter durchlaufen werden und für diskrete Zeitpunkte die berechneten Werte aus den Integrationspunkten herausgezogen werden.

Die Finite-Elemente-Methode ist das aktuell am weitesten verbreitete numerische Verfahren um Umformprozesse zu simulieren [2]. Dabei werden die Geometrien von Bauteil und Werkzeug, sowie Materialgesetze wie das elastische und plastische Verhalten für das Bauteil und Randbedingungen festgelegt. Abbildung 2.1 zeigt ein Viertel eines tiefgezogenen Blechbauteils, aufgeteilt in eine endliche Anzahl an Elementen. Farblich kodiert ist die von Mises Spannung, die unter gegebenen Anfangs-, Rand- und Übergangsbedingungen sowie gegebener Ansatzfunktion für jedes Zeitinkrement berechnet wurde. Je feiner das Bauteil vernetzt ist, desto präziser kann die Näherungslösung werden, jedoch steigt dadurch die Anzahl zu lösender Gleichungen. Es sollte daher ein Maß gefunden werden, das die Berechnungszeit und gewünschte Genauigkeit in Relation setzt.

Für die Erstellung der Datenbasis zur Ermittlung eines onlinefähigen Prozessmodells werden diese berechneten Werte für jedes Zeitinkrement ausgelesen. Um eine Vielzahl von Stichproben zu erhalten, werden mehrere Berechnungen für unterschiedliche Prozessparameter durchgeführt. Für die Observablen kann ebenfalls die FE-Simulation eingesetzt werden, da auch die Krafteinwirkungen, die durch die Werkzeuge ausgeübt werden, modelliert werden. Es können z. B. für bestimmte Referenzpunkte, an Elementen des Bauteils und/oder der Werkzeuge, Krafteinwirkungen und Verschiebungen für jeden Simulationsschritt ausgelesen werden.

Die in dieser Arbeit verwendeten FE-Modelle zur Datenerzeugung werden in Kapitel 6 näher erläutert. Weiterhin gibt Banabic [2] einen tiefen Einblick in die Simulation von Metallumformungen mittels der Finite-Elemente-Methode.

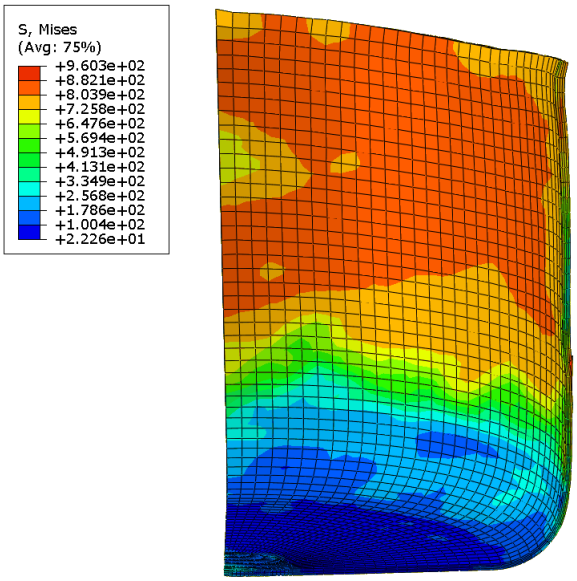


Abbildung 2.1: Finite-Elemente-Modell eines tiefgezogenen Blechbauteils mit Abaqus [89] erstellt. Farblich codiert ist die von Mises Spannung.

2.1.2 Multiskalenmodellierung

Multiskalenmodellierung im Allgemeinen befasst sich mit dem Modellieren eines Systems, das Auswirkung auf unterschiedlichen Skalen in Raum und/oder Zeit hat. Modelle unterschiedlicher Skalen beschreiben dann zusammen das System [101]. *Integrated Computational Materials Engineering* (ICME) im Speziellen hat zum Ziel, Materialeigenschaften und/oder Materialverhalten basierend auf Materialmodellen unterschiedlicher Skalierungen zu verknüpfen und zu evaluieren, um einen besseren Einblick in die Fähigkeiten eines Materials zu bekommen.

Im Rahmen des Graduiertenkollegs wird die Prozesskette Walzen-Glühen-Tiefziehen betrachtet. Hierbei werden mittels der Finiten-Elemente-Methode Mikrostrukturelementmodelle für das Walzen entworfen und simuliert. Die Ergebnisse fließen in die Simulation der Wärmebehandlung mittels Phasenfeldmodell ein [81]. Diese Ergebnisse dienen wiederum in Mikrostruktursimulationen zur Vorhersage makroskopischer mechanischer Kennwerte, die in die Werkstoffmodellierung für Umformsimulationen auf der Bauteilskala einfließen. Abbildung 2.2 zeigt diese einzelnen Prozesse mit der jeweiligen Skalierung.

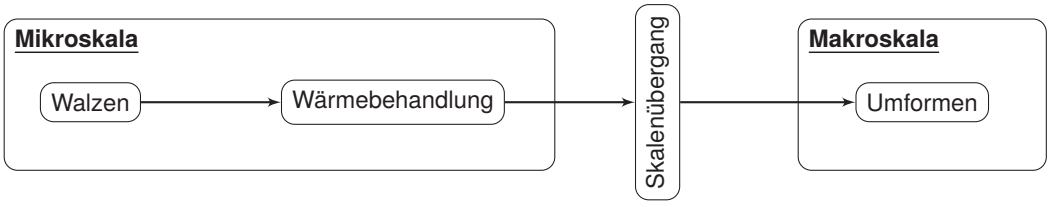


Abbildung 2.2: Betrachtete Prozesskette des Graduiertenkollegs 1483 mit unterschiedlichen Skalierungen der einzelnen Prozesse.

Modellen komplexer Systeme, wie des Tiefziehens bzw. der kompletten Prozesskette (Walzen – Glühen – Tiefziehen), alleine auf der Makroskalen-Ebene, fehlt es oft an Genauigkeit. Dem gegenüber, sind Modelle auf der Mikroskalen-Ebene oft nicht effizient genug oder beinhalten zu viele Informationen. Die Kopplung der Modelle unterschiedlicher Skalierungen soll somit die Vorteile hervorheben und die Nachteile reduzieren.

Durch eine skalenübergreifende numerische Simulation der Prozesskette werden makroskopische Material-, Bauteil- und Prozesseigenschaften mit Mechanismen auf der Mikroskala in Beziehung gesetzt. Dabei umfasst die Makroskalen-Ebene die mechanischen Eigenschaften und die Mikroskalen-Ebene die Anordnung der Atome in Gitterstrukturen. Der Einfluss der Mikroskala auf die Makroskala ist beim Tiefziehen durch texturbedingte Anisotropie zu erkennen. Hierbei hat das Gefüge (durch die Orientierung der Kristallite) Einfluss auf die mechanischen Eigenschaften des Bauteils, die sich beim Ziehen des Bleches durch die Blechdickenverteilung und Zipfelbildung (siehe Kapitel 6.2) ausprägen.

Durch die Kopplung von Modellen unterschiedlicher Skalen können somit mechanische Bauteileigenschaften, die ihren Ursprung in der Mikrostruktur haben, genauer vorhergesagt werden.

2.2 Grundlagen: Modellbildung

„*Ein Modell ist eine vereinfachte Nachbildung eines geplanten oder existierenden Systems mit seinen Prozessen in einem anderen begrifflichen oder gegenständlichen System. Es unterscheidet sich hinsichtlich der untersuchungsrelevanten Eigenschaften nur innerhalb eines vom Untersuchungsziel abhängigen Toleranzrahmens vom Vorbild.*“ - VDI-Richtlinie 3633 [97].

Die VDI-Richtlinie 3633 beschreibt folglich, dass ein Modell die physikalische Realität vereinfacht, aber für das Untersuchungsziel genügend genau nachbildet. In diesem Kapitel geht es um die Erstellung eines mathematischen Modells, das das Übertragungsverhalten – den Zusammenhang zwischen Ein- und Ausgangsgrößen eines Systems – im Zustandsraum beschreibt. Die Modellierung des Übertragungsverhalten kann auf zwei Arten erfolgen [56, 68]:

- **Prognose:** Das Modell ermittelt zukünftige Ausgangsgrößen anhand von Ein- und Ausgangsgrößen bis zum aktuellen Zeitpunkt (bspw. Wettervorhersagen, Modellprädiktive Regelung (MPC) [5, 29]).
- **Simulation:** Das Modell ermittelt die Ausgangsgrößen lediglich durch gegebene Eingangsgrößen. Dies benötigt in der Regel eine Rückführungskomponente für die Ausgangsgrößen innerhalb des Modells (bspw. Fehlerdiagnose).

Und findet in folgenden Bereichen Anwendung:

- **Optimierung:** Das Modell besteht aus Zielgrößen und Prognose- oder Simulationsmodell und dient z. B. zur Findung eines optimalen Verlaufs der Eingangsgrößen.
- **Analyse:** Das Modell hat zum Ziel, ein besseres Verständnis über den Prozess zu erlangen, z. B. durch Experimentieren mit Eingangsgrößen (bspw. Ordnungsreduktion).

- **Beobachtung:** Das Modell rekonstruiert den aktuellen Zustand und gegebenenfalls Störgrößen auf Basis der Ein- und Ausgangsgrößen (bspw. Zustandsregler).

Schwerpunkt dieser Arbeit ist die Ermittlung eines Modells zur Beobachtung des Zustands. Es ist zu untersuchen, ob sich hierfür ein Prognose- oder ein Simulationsmodell besser eignet. Kapitel 3.4 vergleicht Prognose- und Simulationsmodell im Kontext des Beobachters.

2.2.1 Modelleigenschaften

Mathematische Modelle weisen unterschiedliche Eigenschaften auf. Tabelle 2.1 zeigt diese Eigenschaften, die nachfolgend erläutert werden.

Tabelle 2.1: Ordnungsschema mathematischer Modelle anhand ihrer Eigenschaften.

Einteilung nach	**Eigenschaft**	
Modellbildung	theoretisch	experimentell
Modelldarstellung	parametrisch	nichtparametrisch
Modellverhalten	kontinuierlich	diskret
	dynamisch	statisch
Modellstruktur	linear	nichtlinear
Zeitverhalten	zeitinvariant	zeitvariant
Modellvariablen	deterministisch	stochastisch

Die theoretische Modellbildung setzt voraus, dass notwendige physikalische und phänomenologische Gleichungen verfügbar sind. Das Resultat der Modellgewinnung sind zeitkontinuierliche Integral-, Differenzial- oder algebraische Gleichungen mit Parametern, die als physikalische Größen interpretierbar sind [56]. Diese physikalisch basierte Modellgewinnung, mit Kenntnis über Modellstruktur und interpretierbaren Parametern, wird auch *White-Box-Modell* genannt. Für den Fall, dass das Problem nicht analytisch gelöst werden kann, kommen numerische Lösungsverfahren zum Einsatz. Wesentliche Vorteile der theoretischen Modellbildung sind die Interpretierbarkeit der ermittelten Zustandsgleichungen als reale Zustandsvariablen und die damit verbundene Erkenntnis über das Prozessverhalten. Die Zustandsgleichungen können für den Reglerentwurf eingesetzt werden. Weiterhin kann die theoretische Modellbildung bereits in der Planung- und Entwurfsphase des Prozesses eingesetzt werden. Ein wesentlicher Nachteil dieser Modellbildung ist jedoch, dass zwar eine hohe Approximationsgüte erreicht werden kann, dies jedoch mit einer sehr hohen Komplexität des Modells und einem damit verbundenen hohen Rechenaufwand einhergehen kann [61]. Durch Reduktion der Anzahl der Differenzialgleichungen (*Ordnungsreduktion*, Abschnitt 2.4), die zur Beschreibung des Systems notwendig sind, kann die Komplexität des Systems verringert werden.

Die experimentelle Modellbildung benötigt lediglich die Ein- und Ausgangsgrößen des Systems. Im Allgemeinen sind die dadurch ermittelten Differenzialgleichungen bzw. Übertragungsfunktionen von niedriger Ordnung. Da Messdaten der Ausgangsgrößen zeitdiskret vorliegen, werden typischerweise zeitdiskrete Modelle identifiziert. Die Parameterermittlung basiert auf Schätzung und Strukturanpassung. Da kein Wissen über die innere Struktur des System vorhanden ist sowie die Parameter nicht interpretiert werden können, spricht man auch von *Black-Box-Modell*. Vorteile der experimentellen Modellbildung sind, dass keine Kenntnisse über die physikalische Funktionsweise oder das Verhalten des Prozesses nötig sind, um ein Modell zu ermitteln sowie die bereits bei der Modellgewinnung resultierende niedrige Ordnung. Daraus ergibt sich jedoch auch der Nachteil der experimentellen Modellbildung: Es existieren im Allgemeinen keine Zustandsgleichungen mit realen physikalischen Variablen [56]. Deshalb muss zum Sammeln der Daten bereits ein experimentierfähiges theoretisches Modell oder eine reale Regelstrecke vorhanden sein. Dadurch kann das experimentelle Modell erst im Betrieb und nicht in der Entwurfsphase des Reglers eingesetzt werden.

Die erfahrungsbasierte Modellbildung führt Kroll [56] als dritte Art der Gewinnung mathematischer Modelle auf. Diese Art setzt ein Expertenmodell voraus, das heißt, es ist Wissen von Betriebspersonal über Ursache-Wirkungs-Verhalten eines Systems vorhanden. Dieses Wissen wird in einer Regel- und Datenbasis festgehalten. Eine Zugehörigkeitsfunktion legt fest, wie aus den Ein- die Ausgangsgrößen abgeleitet werden. In der vorliegenden Arbeit wird auf Grund fehlender Daten diese Methode nicht weiter betrachten. Eine Anwendung beschreibt Hassan et al. [30]; sie entwickelten ein Fuzzylogik-Modell zur Vorhersage von Fließspannung bei Umformprozessen. Als Daten- und Regelungsbasis dienen dabei Beobachtungen bei Umformversuchen (Material, Temperatur, Dehnung, Dehnungsrate), Hinweise von Blechbauteil-Herstellern und allgemeine Literatur zum Umformen (Spannungszustand, Bauschingereffekt). Die Arbeit zeigt, dass mittels des erstellten Fuzzylogik-Modells während eines Warmstauchen-Prozesses die vorhergesagte Fließspannung in guter Übereinstimmung mit experimentellen Daten steht.

Die Modelldarstellung kann beim experimentell ermittelten Modell *parametrisch* und *nichtparametrisch* erfolgen. Hierbei bedeutet „nichtparametrisch“, dass die Anzahl und die Art der Parameter zu Beginn nicht festgelegt sind. Ein einfaches Beispiel für nichtparametrische Modelle sind Modelle, die mit Wertetabellen oder Kurven gebildet werden. Im Bereich des maschinellen Lernens werden parametrische Modelle (Differenzialgleichungen, Übertragungsfunktionen) auch als symbolische Repräsentation und nichtparametrische Modelle als subsymbolische Repräsentation bezeichnet. Künstliche neuronale Netze (KNN) zählen zu den subsymbolischen Verfahren, da dem Black-Box-Modell entsprechend nur eine Abbildung von Eingangsgrößen auf Ausgangsgrößen ermittelt wird und dabei die Modellparameter weder in ihrer Anzahl feststehen noch eine physikalische Bedeutung haben. Eine Mischung aus White-Box und Black-Box-Modell ermöglicht es, bei der experimentellen Modellbildung physikalische Zusammenhänge einfließen zu lassen. Das aus dieser Mischung resultierende Modell wird auch *Grey-Box-Modell* genannt. Es ermittelt ein parametrisches Modell, bei dem durch physikalisch motivierte Annahmen über die Modellstruktur den Modellparameter Bedeutung zugeordnet werden kann, die jedoch nicht zwingend physikalischen Größen entsprechen.

Das Modellverhalten lässt sich kontinuierlich oder diskret sowie dynamisch oder statisch beschreiben. Ein dynamisches zeitkontinuierliches System lässt sich durch die Prozessmodellgleichung (2.1) und die Messmodellgleichung (2.2) beschreiben, mit $\boldsymbol{u}(t)$ als Vektor der Eingangsgrößen bzw. Regelgrößen, $\boldsymbol{y}(t)$ als Vektor der Ausgangsgrößen bzw. Observablen und $\boldsymbol{x}(t)$ als Vektor der Zustandsgrößen zum Zeitpunkt t, als kontinuierliche Zeit.

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \tag{2.1}$$

$$\boldsymbol{y}(t) = g(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \tag{2.2}$$

Nicht immer ist eine kontinuierliche Beschreibung möglich, z. B. falls die Modellbildung nur auf Messungen beruht oder numerische Methoden zur Modellbildung zum Einsatz kommen. Diese Daten liegen zu bestimmten Zeitpunkten vor, sodass der Zustand zwischen den zeitlichen Abtastpunkten nicht bekannt ist. Für ein dynamisches zeitdiskretes System ändert sich die Zustandsgleichung und Ausgangsgleichung wie folgt:

$$\boldsymbol{x}_{t+\Delta t} = f(\boldsymbol{x}_t, \boldsymbol{u}_t, t) \tag{2.3}$$

$$\boldsymbol{y}_t = g(\boldsymbol{x}_t, \boldsymbol{u}_t, t), \tag{2.4}$$

wobei Δt das Abtastintervall, d. h. der zeitliche Abstand zwischen zwei Abtastpunkten, ist. Eine statische Beschreibung des Systems bildet die aktuellen Eingangswerte auf die aktuellen Ausgangswerte ab, wobei keine explizite Zeitabhängigkeit auftritt.

Die Modellstruktur, d. h. die funktionalen Abhängigkeiten der Gleichungen (2.1) bis (2.4), kann linear oder nichtlinear sein. Methoden zur Gewinnung von Modellen für lineare Systeme sind weitestgehend erforscht. Jedoch sind Prozesse in Natur und Industrie in der Regel nichtlineare Systeme. Um lineare Methoden auch bei nichtlinearen Systemen anwenden zu können, wird Linearisierung eingesetzt. Falls mit der Linearisierung nicht die gewünschte Genauigkeit erreicht wird, werden nichtlineare Verfahren benötigt.

Das Zeitverhalten des Systems ist dann zeitvariant, wenn sich das Systemverhalten über die Zeit trotz gleicher Eingangswerte ändert. Dies kann z. B. durch Einflüsse von außen, wie Werkzeugverschleiß, hervorgerufen werden. Sind die Prozessmodellgleichung (2.1) bzw. (2.3) und Messmodellgleichung (2.2) bzw. (2.4) nicht explizit von der Zeit t abhängig, so handelt es sich um ein zeitinvariantes System.

Der Zusammenhang zwischen den Modellvariablen ist bei deterministischen Modellen vollständig reproduzierbar. Im Gegensatz dazu fließen bei stochastischen Modellen Zufallsgrößen und/oder Zufallsgesetze mit ein.

2.2.2 Überblick über Verfahren zur Modellbildung

In dieser Arbeit werden Verfahren zur theoretischen und experimentellen Modellbildung betrachtet, die zur Bestimmung der Form der Gleichungen (2.1) und (2.2) bzw. (2.3) und (2.4) für

einen bestimmten Prozess verwendet werden können. Durch die in der Regel hohe Komplexität der theoretischen Modelle und der damit verbundenen hohen Berechnungszeit werden sie hier nur zur Datenerzeugung verwendet (Kapitel 2.1). Diese Daten dienen als Grundlage zur Erstellung eines durch Daten definierten Surrogat-Modells mittels experimenteller Modellbildung, welches dann zur online Zustandsverfolgung des spezifischen Prozesses verwendet werden kann.

Abbildung 2.3 zeigt eine Aufteilung der Verfahren in theoretische und experimentelle Modellbildung. Die theoretische Modellbildung besteht in der Regel aus Aufstellen und Lösen von Differenzialgleichungen, analytisch oder numerisch. Für die experimentelle Modellbildung gibt es zahlreiche Verfahren. Abbildung 2.3 listet davon einige relevante und weitverbreitete Verfahren auf. Des Weiteren kann die Komplexität des theoretischen Modells auch durch Dimensionsreduktion verringert werden. Dies erfolgt mittels Ordnungsreduktion (Reduzierung der Anzahl der Differenzialgleichungen) oder Reduzierung der Daten. Verfahren zur Dimensionsreduktion werden in Kapitel 2.4 betrachtet.

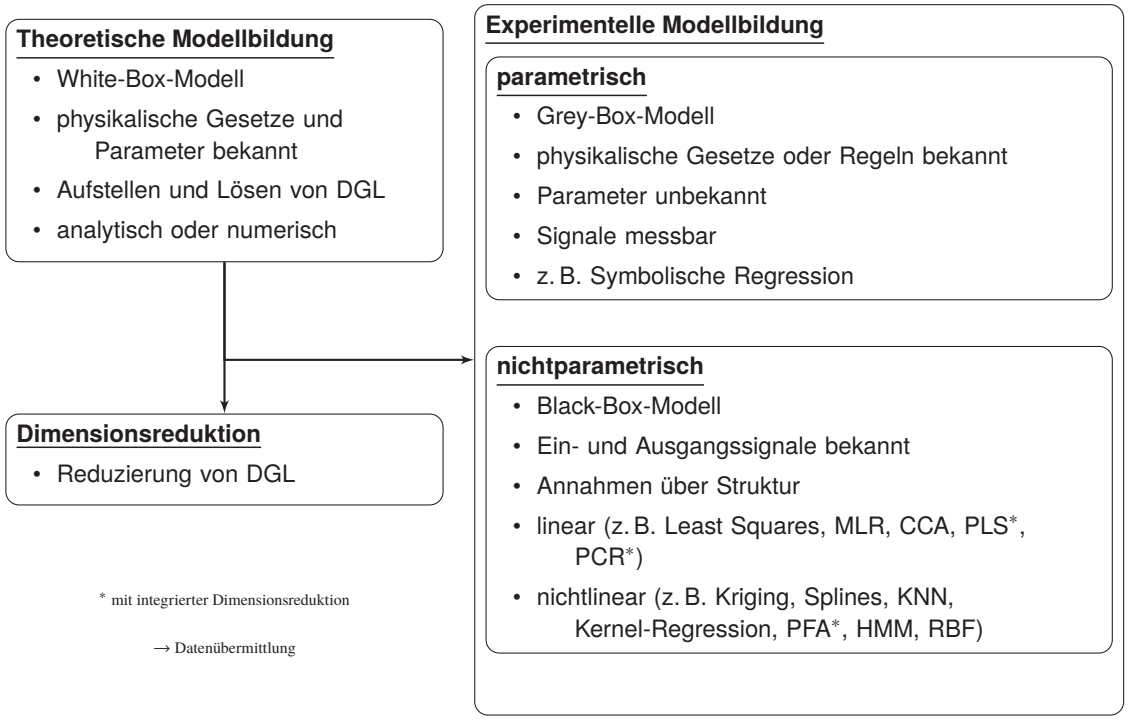


Abbildung 2.3: Übersicht über Eigenschaften und Verfahren zur Modellbildung.

Bei den Verfahren zur experimentellen Modellbildung handelt es sich um statistische Methoden zur Funktionsapproximation. Diese Verfahren versuchen, auf Basis von Daten aus realen Experimenten oder numerischen Simulationen eines experimentierfähigen Modells, Funktionen zu finden, die Zusammenhänge dieser Daten annähern. Die Modellierung der Beziehung zwischen einer abhängigen Variablen und einer oder mehreren unabhängigen Variablen wird *Regression* genannt; teilweise bei niedrigen Dimensionen auch *Curve-Fitting*.

Die in Abbildung 2.3 aufgelisteten linearen Verfahren Least Squares, Multiple Linear Regression (MLR) und Canonical Correlation Analysis (CCA) sind Verfahren zur linearen Regression [72]. Partial Least Squares Regression (PLS) [98] und Principal Component Regression (PCR) [42] führen zusätzlich zur linearen Regression auch eine lineare Dimensionsreduktion durch. Auf-

grund des nichtlinearen Verhaltens des zu repräsentierenden Prozesszustandes werden diese Verfahren hier nicht weiter betrachtet.

Zu den nichtlinearen Verfahren gehört das *Kriging*, auch unter *Gaussprozess-Regression* [73] bekannt. Das Verfahren hat seinen Ursprung in der Geostatistik [55] und wurde von Matheron [64] weiterentwickelt. Es ist ein stochastisches Interpolationsverfahren, das einen nicht gemessenen Wert durch gewichtete Umgebungspunkte bestimmt. Die Varianz der geschätzten Kurve wird dabei durch die Gewichte (deren Summe gleich eins ist) minimiert. Durch seinen Ursprung findet Kriging überwiegend im Bereich der Geowissenschaften Anwendung. Es wird aber auch im Bereich von Fertigungsverfahren untersucht. Jakumeit et al. [40] optimieren Prozessparameter (u. a. zeitabhängige Niederhalterkraft) einer Tiefziehprozesssimulation mit Hilfe des Krigings. Sie konnten dadurch die Anzahl der Simulationen, die nötig sind, um ein optimales Ergebnis für die Prozessparameter zu ermitteln, deutlich reduzieren. Wang et al. [99] verwenden Kriging zur Optimierung von Ziehsicken beim Tiefziehen. Die Arbeiten machen deutlich, dass Kriging im Bereich des Umformens überwiegend zur statistischen Versuchsplanung verwendet wird. Weiterführende Literatur zum Algorithmus des Krigings ist in [70, 100] zu finden.

Eine weiteres Verfahren zu Curve-Fitting sind Splines. Dabei handelt es sich um ein deterministisches Interpolationsverfahren, das eine Funktion intervallweise aus Polynomen unter folgenden drei Bedingungen zusammensetzt [41]:

1. Die Datenpunkte $d_1, \ldots, d_n$ innerhalb eines Intervalls $]a, b[$, mit $a < d_1 < \ldots < d_n < b$, unterteilen das Intervall in $n + 1$ kleinere Intervalle
2. In jedem Teilintervall $[d_i, d_{i+1}]$ ist der Spline gegeben durch ein Polynom. Der Grad des Polynoms ist maximal der Grad des gesamten Splines.
3. Der Spline und seine Ableitung bis Ordnung $n - 1$ sind stetig im Intervall $]a,b[$.

Nachteil des Kriging und der Splines ist, dass abhängig von der Komplexität und Anzahl der Daten der Berechnungsaufwand signifikant steigt. Die Daten bzw. der Zustandsraum müssten folglich erst reduziert werden, um diese Verfahren anzuwenden.

Zu den nichtlinearen Regressionsverfahren, die sich auch für hohe Komplexität und große Datenmengen eignen, zählen unter anderem künstliche neuronale Netze und Kernel-Methoden. Künstliche neuronale Netze (KNN), zu denen auch radiale Basisfunktions-Netze (RBF-Netze) und Principal Function Approximators (PFA) zählen, finden bereits vielseitig Anwendung in der Regelungstechnik [67, 90] und in der Parameteroptimierung [1, 62, 103]. Das Universal Approximation Theorem nach [36] besagt, dass ein Feedforward-Netz mit einer verdeckten Schicht (siehe Kapitel 2.3) mit einer endlichen Anzahl an Neuronen beliebige kontinuierliche Zusammenhänge nachbilden kann. Künstliche neuronale Netze sind dadurch vielseitig anwendbar, insbesondere weil sie auch bei hochdimensionalen Daten angewendet werden können.

Wie auch das Kriging basieren radiale Basisfunktions-Netze (RBF) und die nichtlineare Erweiterung der Support Vector Machines [9] auf der Verwendung von Kernels (gewichtete Funktionen). RBF-Netze sind neuronale Netze, die radiale Basisfunktionen als Aktivierungsfunktion besitzen [27]. Kernel-Methoden haben den allgemeinen Nachteil, dass die Daten zuerst nichtlinear in einen höher-dimensionalen Raum transformiert werden, um das Problem dann linear zu lösen. Dabei ist die nichtlineare Transformation nicht explizit bekannt und somit das ermittelte Modell nicht mehr im ursprünglichen Zustandsraum darstellbar.

Mittels KNN ist es zwar möglich, beliebige Transformationen von Ein- zu Ausgangsgrößen zu ermitteln, jedoch ist die gefundene Transformationsfunktion nicht im Sinne eines Grey-Box-Modells interpretierbar. Zur Ermittlung einer interpretierbaren Lösung eignet sich die symbolische Regression. Hier ist die Modellstruktur nicht vorgegeben und muss zusätzlich bestimmt werden. Dieser hinzukommende Freiheitsgrad ermöglicht es, Expertenwissen über die Modellstruktur mit einfließen zu lassen und ein Modell zu erhalten, dessen Modellstruktur und -parameter mit physikalischem Verhalten und Größen in Relation stehen oder sogar konkret interpretiert werden können. In Kapitel 2.6 wird näher auf das Verfahren und seine Möglichkeiten eingegangen.

Weiterhin ist das Hidden Markov Model (HMM) [16] zu nennen. Es ist ein stochastisches Modell, das auf Markov-Ketten beruht, jedoch mit verdeckten (unbeobachteten) Zuständen. Markov-Ketten modellieren die Wahrscheinlichkeit eines zukünftigen Zustands anhand des aktuellen Zustands, vorherige Zustände fließen nicht mit ein. Da beim HMM die Zustände nicht sichtbar sind, sind nur die Beobachtungen, die vom aktuellen Zustand abhängen, ersichtlich. Auch hier steigt die Komplexität des Modells rapide mit der Komplexität des zu modellierenden Prozesses.

In dieser Arbeit werden KNNe im Hinblick auf die Modellbildung und nichtlineare Dimensionsreduktion genauer betrachtet, da diese in der Lage sind, auch das Verhalten komplexer Systeme mit einer hohen Anzahl an Zustandsvariablen zu approximieren. Kapitel 2.3 erläutert den grundlegenden Aufbau künstlicher neuronaler Netze. Wird das System durch Dimensionsreduktion (Kapitel 2.4) in einem reduzierten Zustandsraum betrachtet bietet sich zusätzlich symbolische Regression (Kapitel 2.6) an, um eine interpretierbare mathematische Beschreibung zu erhalten, die direkt für die Entwicklung eines klassischen Zustandsbeobachter (Kapitel 3) verwendet werden kann.

2.3 Künstliche neuronale Netze

Künstliche neuronale Netze (KNN) haben ihre Anfänge bereits in den 1940er Jahren. Einen kurzen geschichtlichen Überblick gibt Schmidhuber in [79]. Künstliche neuronale Netze haben natürliche neuronale Netze eines Nervensystems zum Vorbild. Einen ausführlichen Einblick in den Hintergrund gibt Rojas [76]. Zwei typische Aufgaben neuronaler Netze sind die Klassifikation und die Funktionsapproximation zugrundeliegender Daten. Zur Klassifikation wird das Netz mit zu klassifizierenden Daten als Eingabe und zu ermittelnden Klassen als Ausgabe des Netzes trainiert. Bei der Funktionsapproximation dienen die unabhängigen Variablen der zu erlernenden Funktion als Eingabe und die Funktionswerte als Ausgabe. In dieser Arbeit kommen künstliche neuronale Netze zur Funktionsapproximation zum Einsatz. Werden also Messwerte in Abhängigkeit des Zustandes (das Messmodell) gesucht $\boldsymbol{y} = g(\boldsymbol{x})$, dann erhält das Netz die Zustandsvariablen als Eingabe und die Messwerte als Ausgabe. Existiert die Umkehrfunktion $\boldsymbol{x} = g^{-1}(\boldsymbol{y})$, so können die Messwerte als Eingabe verwendet werden und die Zustandsvariablen als Ausgabe; dies kann nach dem Training zur Vorhersage bzw. Rekonstruktion des Zustandes anhand der Messwerte verwendet werden.

Aufbau: Ein einfaches neuronales Netz besteht aus einer Menge verbundener sogenannter Neuronen in einer oder mehreren Schichten. Die Verbindungen zwischen Neuronen erhalten

eine *Gewichtung*. Die Struktur des Netzes, sprich die Anzahl der Neuronen und Schichten sowie wie diese miteinander verbunden sind, wird als Topologie des Netzes bezeichnet. Abbildung 2.4 zeigt Beispiele der drei typischen Topologien, in Bezug auf die Verbindungen der Neuronen: das einschichtige und mehrschichtige feedforward-Netz sowie das rekurrente Netz. Feedforward-Netze haben nur nach vorne, zur Ausgabeschicht hin, gerichtete Verbindun-

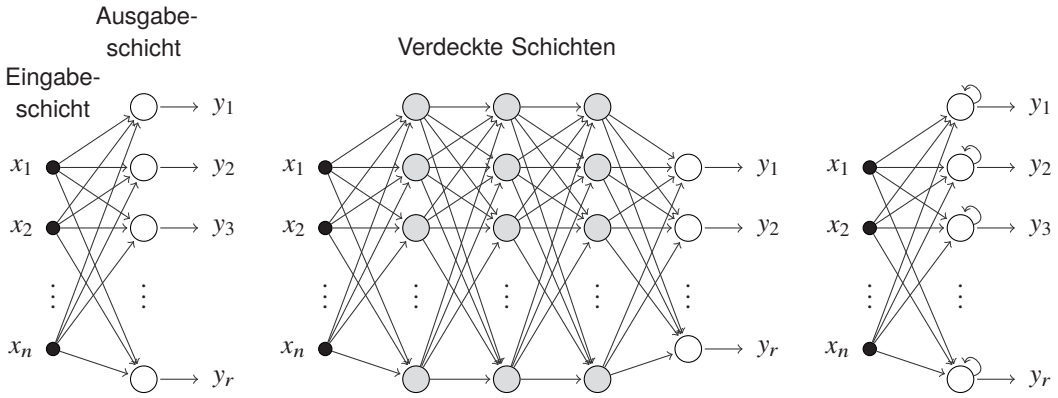


Abbildung 2.4: Links: Einschichtiges KNN mit n Eingaben und r Ausgaben. Mitte: Mehrschichtiges KNN mit drei verdeckten Schichten und Ausgabeschicht. Rechts: Einfaches einschichtiges rekurrentes KNN mit Rückführung des Aktivierungswerts der Ausgabeschicht.

gen. Das einschichtige feedforward-Netz besitzt nur die Ausgabeschicht. Das mehrschichtige feedforward-Netz hingegen besitzt weitere verdeckte Schichten, die außerhalb des Netzes nicht sichtbar sind. Diese verdeckten Schichten erhöhen die Abstraktion des Netzes, dass heißt vernachlässigbare Informationen, wie Rauschen, werden reduziert und gemeinsame Informationen (Merkmale) extrahiert. Diese Netze finden auch beim sogenannten *Deep Learning* Anwendung. Die Ausgabeschicht stellt immer eine sichtbare Schicht dar. Rekurrente Netze besitzen zudem auch rückwärts gerichtete Verbindungen, die ein dynamisches Verhalten nachbilden können. Tabelle 2.2 zeigt die wesentlichen Unterschiede zwischen feedforward-Netzen und rekurrenten Netzen.

Tabelle 2.2: Vergleich feedforward-Netz und rekurrentes Netz.

	Feedforward-Netz
Struktur	keine Zyklen
Input-output	keine Zeitreihen (statisch)
Verhalten	reaktiv
Mathematisch	statische Ein-/Ausgangs-Abbildung
	Rekurrentes Netz
Struktur	enthält gerichtete Zyklen
Input-output	Zeitreihen möglich (dynamisch)
Verhalten	mit Prädiktion über zukünftige Ereignisse
Mathematisch	dynamisches System

Besonders im Bereich der rekurrenten neuronalen Netze gibt es eine Vielzahl von unterschiedlichen Topologien und damit verbunden unterschiedlichen Lernverfahren. Zu nennende Topologien sind das Hopfield-Netz [35], das Elman-Netz [17], das Jordan-Netz [43] sowie das Long Short Term Memory (LSTM) [34].

Unabhängig davon ob feedforward oder rekurrent, jedes Neuron besitzt eine Aktivierungsfunktion, die bestimmt, wie die Neuroneneingabe verarbeitet wird. Abbildung 2.5 zeigt die Berechnung eines *Aktivierungswerts* (Ausgabe eines Neurons) durch die *Aktivierungsfunktion h* mit den aufsummierten gewichteten Eingabewerten bzw. gewichteten Aktivierungswerten der vorherigen Schicht als Funktionsparameter.

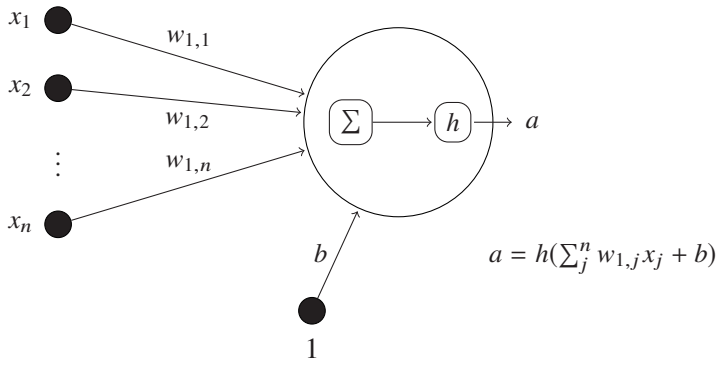


Abbildung 2.5: Berechnung des Aktivierungswert a durch ein Neuron, mit w_{ij} als Gewichtung der Verbindung zwischen Neuron j zu Neuron i, Eingabewerten $\boldsymbol{x}$ und Schwellwert b.

Zusätzlich kann ein Bias, ein Neuron mit Wert 1, eingesetzt werden, dessen Gewicht b als Schwellwert für die Aktivierungsfunktionen benutzt wird. In dieser Arbeit wird die Eingabe auch als Eingabeschicht bezeichnet. Diese Eingabeschicht beinhaltet jedoch keine Aktivierungsfunktion, sondern stellt lediglich die Eingabewerte zur Verfügung.

Lernen: Von der eingesetzten Topologie hängt auch das verwendete Lernverfahren ab. Die Grundidee der Lernverfahren besteht darin, durch Einstellung der Gewichte eine optimale Lösung des Problems zu ermitteln. Grundsätzlich lassen sich drei verschieden Lernverfahren unterscheiden: *Überwachtes Lernen*, *unüberwachtes Lernen* und *bestärkendes Lernen*. Das überwachte Lernen trainiert das Netz mit einer gegebenen Ein- und Ausgabe (Zielwerte), z. B. bei der Erkennung von Buchstaben. Hierbei ist für einen Eingabe-Datensatz bekannt, welcher Buchstabe zu jeder Eingabe gehört und somit als Ausgabe-Zielwert festgelegt werden kann. Beim unüberwachten Lernen liegen keine Zielwerte vor, sondern die Aufgabe liegt viel mehr darin, Strukturen in den Daten zu finden. Die Gewichte werden in Abhängigkeit der Eingabewerte bestimmt. Bei der Komprimierung von Daten mittels neuronaler Netze handelt es sich um unüberwachtes Lernen; die Eingabe ist durch den Datensatz gegeben, die komprimierte Darstellung der Daten muss ermittelt werden. Nähere Erläuterungen zu überwachtem und unüberwachtem Lernen geben Goodfellow et al. [25]. Die dritte Variante ist das bestärkende Lernen [91], bei dem durch „Belohnung“ (positiv oder negativ) während des Trainings eine definierte Zielfunktion approximiert wird.

Für die Funktionsapproximation, bei bekannter Ein- und Ausgabe, werden überwachte Lernverfahren verwendet. Backpropagation und Varianten dieses Verfahrens [39, 74, 95] sind die

am weitesten verbreiteten Trainingsverfahren für überwachtes Lernen neuronaler feedforward-Netze. Backpropagation basiert auf einem allgemeinen Gradientenverfahren, welches die optimale Lösung zur Einstellung der Gewichte entlang einer Kurve von Gewichtswerten sucht. Dabei wird die Eingabe vorwärts durch das Netz propagiert und die dadurch ermittelte Ausgabe mit der gewünschten Ausgabe (*Zielwerte*) verglichen. Der Fehler zwischen ermittelter Ausgabe $\hat{\boldsymbol{y}}$ und Zielausgabe $\boldsymbol{y}$ ist z. B. durch den mittleren quadratischen Fehler in Gleichung (2.5) über alle i Neuronen gegeben und dient als Fehlerfunktion des Gradientenverfahrens, die es zu optimieren gilt.

$$E = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.5}$$

Dieser Fehler zwischen der Ausgabe des Netzes und den Zielwerten wird nun durch das gesamte Netz zurück propagiert (siehe Abbildung 2.6). Die zusammengesetzte Gleichung

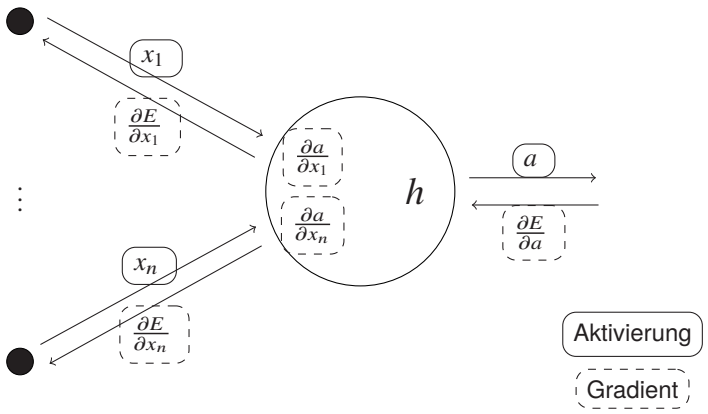


Abbildung 2.6: Backpropagation durch einen Knoten, mit $\frac{\partial E}{\partial x_1} = \frac{\partial E}{\partial a}\frac{\partial a}{\partial x_1}$ und $\frac{\partial E}{\partial x_n} = \frac{\partial E}{\partial a}\frac{\partial a}{\partial x_n}$.

$\hat{y} = w_{yz}\left(\ldots h\left(\sum_k^{n_k} w_{jk}\left(h(\sum_j^{n_j} w_{ij}x_j)\right)\right)\right)$ zur Berechnung der Ausgabewerte, mit w_{ij} für die Gewichte zwischen zwei Schichten. Um den Einfluss der Änderung der Gewichte w_{ij} auf den Fehler zu überprüfen, wird mittels Kettenregel der Gradient wie folgt bestimmt:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial a_i}\frac{\partial a_i}{\partial net_i}\frac{\partial net_i}{\partial w_{ij}}, \tag{2.6}$$

dabei ist w_{ij} die Gewichtung von Neuron j zu Neuron i, a_i der Aktivierungswert von Neuron i und net_i die gewichtete Summe über alle Eingabeneuronen für Neuron i:

$$net_i = \sum_{j=1}^{n} w_{ij}a_j. \tag{2.7}$$

Danach werden die Gewichte in Abhängigkeit von ihrem Einfluss auf den Fehler E aktualisiert:

$$w_{ij} = w_{ij} - \alpha\frac{\partial E}{\partial w_{ij}}. \tag{2.8}$$

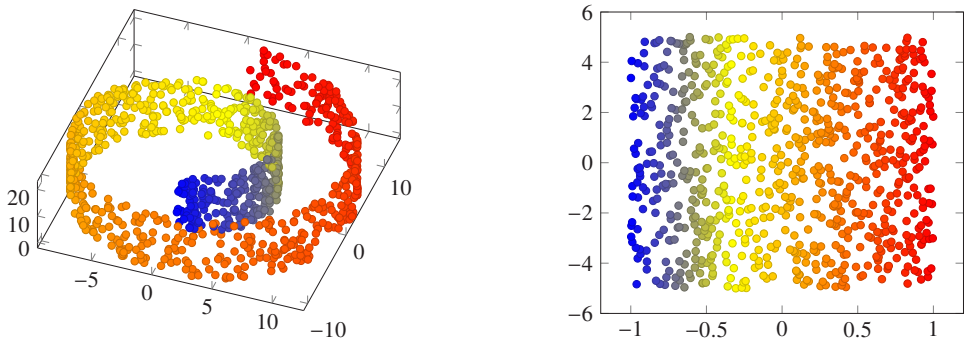


Abbildung 2.7: Beispiel eines zweidimensionalen Datensatzes (rechts), der in einem dreidimensionalen Raum liegt (links).

Diese Aktualisierung der Gewichte erfolgt für alle aufeinander folgenden Schichten. Dabei gilt für die Ausgabeneuronen $a = \hat{y}$:

$$\frac{\partial E}{\partial a} = \alpha(y - \hat{y}). \tag{2.9}$$

Die Wahl der Lernrate α bestimmt, wie schnell das Verfahren konvergiert. Eine zu kleine Lernrate kann dazu führen, dass das Netz sehr lange braucht, um eine optimale Lösung zu finden. Eine zu große Lernrate kann jedoch dazu führen, dass das Netz um die optimale Lösung herum oszilliert. Verfahren zur Steuerung der Lernrate und die Gewichtsänderung werden z. B. von [49], [78] und [15] eingeführt.

2.4 Datengetriebene Dimensionsreduktion

Um komplexe Systeme, die z. B. durch die räumliche Verteilung physikalischer Größen gekennzeichnet sind, möglichst genau darzustellen, wird eine Vielzahl von Zustandsvariablen benötigt. Die Anzahl der Zustandsvariablen entspricht der Dimension des Zustandsraums. Der hohe Rechenaufwand, der mit der hohen Komplexität des Modells einhergeht, ist jedoch für eine online-Zustandsverfolgung nicht geeignet. Hier kommen Reduktionsverfahren zum Einsatz, um redundante Informationen zu entfernen und somit das System, das im Zustandsraum beschrieben wird, in einen reduzierten Raum zu transformieren. In anderen Worten, es wird davon ausgegangen, dass das betrachtete System in einem niedrigdimensionalen Unterraum des Zustandsraumes liegt. Abbildung 2.7 zeigt ein einfaches bildliches Beispiel, die sogenannte „Swiss Roll“: Ein zweidimensionaler Datensatz, der in einem dreidimensionalen Raum liegt und der weitverbreitet als Benchmark für nichtlineare Dimensionsreduktionsverfahren dient [57].

Es wird somit bei der Dimensionsreduktion davon ausgegangen, dass der Prozess im Zustandsraum nur einen kleinen Raum einnimmt, der durch den sogenannten inneren Zustand beschrieben wird. Diesen sogenannten Merkmalsraum – die intrinsische Dimension – und die im Allgemeinen krummlinigen Koordinatenachsen gilt es zu identifizieren. Der Merkmalsraum wird von Merkmalen aufgespannt. Werden mittels der Dimensionsreduktion unkorrelierte Merkmale extrahiert, so ist hier von Merkmalsextraktion die Rede.

Im Wesentlichen können nichtlineare Dimensionsreduktionsverfahren in zwei Kategorien eingeteilt werden: topologieerhaltend und distanzerhaltend. Topologieerhaltende Verfahren versuchen, die Topologie im mathematischen Sinne zu erhalten, darunter zählen Self-Organizing Maps [50], Locally-linear Embedding [77] und Laplacian Eigenmaps [4]. Distanzerhaltende Verfahren berücksichtigen den Abstand oder Winkel zwischen den Datenpunkten, sodass die niedrigdimensionale Darstellung die wesentlichen geometrischen Eigenschaften der Daten enthält. Darunter fallen u. a. Multidimensional Scaling [12], Kernel-PCA [82] und Isomap [93].

Die genannten Verfahren neigen dazu, beliebige Merkmale zu finden, die eine passende niedrigdimensionale Darstellung liefern, was nicht zwingend zu unkorrelierten Merkmalen führt und dafür sorgt, dass die ermittelte Abbildung nicht reproduzierbar ist bzw. die Rückprojektion zum originalen Zustand nicht möglich ist. Nachvollziehbarkeit der Abbildung bzw. die Rekonstruktion des originalen Zustandes ist von Bedeutung, um die reduzierte Darstellung zu bewerten und wieder auf den originalen Zustand, der durch die Merkmale repräsentiert wird, für die weitere Verarbeitung zugreifen zu können. Eines der weitverbreitesten Dimensionsreduktionsverfahren, das sowohl zur Dimensionsreduktion wie auch zur Merkmalsextraktion eingesetzt werden kann, ist die Hauptkomponentenanalyse. Dieses lineare Verfahren wird häufig auch zur Ordnungsreduktion für nichtlineare Systeme verwendet und wird deswegen in Abschnitt 2.4.1 beschrieben. Durch Autoencoder kann die Hauptkomponentenanalyse mittels neuronaler Netze nachgebildet werden, jedoch gehen ohne zusätzliche Anpassungen die Eigenschaft der PCA, nach Varianz sortierte Komponenten zu generieren, verloren. Kapitel 2.4.2 zeigt die Funktionsweise der Autoencoder und eine Erweiterung, um nichtlineare Merkmale zu extrahieren.

2.4.1 Hauptkomponentenanalyse

Die Hauptkomponentenanalyse ist ein weit verbreitetes lineares Dimensionsreduktionsverfahren und ist im Bereich der Bildverarbeitung unter Principle Component Analysis (PCA) und im Maschinenbau auch unter Proper Orthogonal Decomposition (POD) bekannt. Sie wurde von Kosambi [52], Karhunen [48] und Loève [58] unabhängig voneinander eingeführt. Im Bereich des Maschinenbaus findet sie vor allem im Bereich der Fluiddynamik Einsatz [3, 11, 71]. Lumley [60] betrachtete erstmals die Verwendung der POD im Bereich der nichtlinearen Mechanik. Fritzen [23] untersuchte POD im Hinblick auf die Findung von Komponenten (auch als Modes bezeichnet), die die Informationen der charakteristischen Merkmale und der Wechselwirkungen heterogenen Materials beinhalten, um ein Verfahren zur nichtlinearen Homogenisierung zu entwickeln.

Die Hauptkomponentenanalyse startet mit der Annahme, dass die höchste Varianz in den zugrundeliegenden Daten (hier die Zustandsvariablen) dem höchsten Informationsgehalt entspricht und folglich kleine Varianzen vernachlässigbare Informationen oder Rauschen darstellen. Mathematisch betrachtet ist die Hauptkomponentenanalyse eine lineare Transformation, die die Daten in ein neues Koordinatensystem mit geringerer Dimension transformiert. Der Vektor in Richtung der höchsten Varianz wird zum ersten Basisvektor des reduzierten Raums (erste Hauptkomponente), dann wird weiter im zur ersten Hauptkomponente orthogonalen Unterraum die Richtung der nächsthöheren Varianz gesucht, die dann den zweiten Basisvektor (zweite Hauptkomponente) bildet und so weiter. Die erste Hauptkomponente ist somit die beste eindimensionale Approximation der Daten; der euklidische Abstand zwischen Datenpunkten und der

ersten Hauptkomponente ist geringer als der zwischen Datenpunkten und jeder anderen beliebigen Geraden. Die orthogonalen Hauptkomponenten bilden die Basis des reduzierten neuen Koordinatensystems. Dieses Verfahren ermöglicht es, durch Rückprojektion die ursprünglichen Daten wieder zu rekonstruieren. Zur Implementierung der Hauptkomponentenanalyse kann die Singulärwertzerlegung (SVD, engl. singular value decomposition) eingesetzt werden (siehe Implementierung 1).

Implementierung 1 Hauptkomponentenanalyse mittels Singulärwertzerlegung

Eingabe: Daten-Matrix $X \in \mathbb{R}^{s \times n}$, reduzierte Dimension $k < n$
Ausgabe: Reduzierte Daten $Z \in \mathbb{R}^{s \times k}$ im von $U_{reduced}$ aufgespannten Raum

1: $X \leftarrow X - mean(X)$ ▷ Zentrierung
2: $X \leftarrow \frac{X}{std(X)}$ ▷ Skalierung
3: $C \leftarrow \frac{1}{s} \cdot (X^\top X)$ ▷ Berechnung der Kovarianz
4: $[U,S] \leftarrow svd(C)$ ▷ Singulärwertzerlegung
5: $U_{reduced} \leftarrow U(1:n,1:k)$ ▷ Extraktion von k Komponenten
6: $Z \leftarrow X \cdot U_{reduced}$ ▷ Projektion auf die reduzierten Komponenten

Die Implementierung 1 erhält die Eingangsdaten $X \in \mathbb{R}^{s \times n}$ und die Dimension k, auf die reduziert wird. Die Eingangsdaten liegen für mehrere Stichproben s vor, z. B. die Zustandsvariablen bei unterschiedlichen Prozessparametern. Um statistisch unabhängige Komponenten zu finden, müssen die Daten normalverteilt sein. Unter dieser Annahme werden zuerst die Daten mit Hilfe des Mittelwerts der Zustandsvariablen über alle Stichproben zentriert. Für den Fall, dass die Zustandsvariablen nicht im gleichen Wertebereich vorliegen (z. B. Spannungswerte und Dehnungswerte), empfiehlt sich eine Skalierung der Daten. Dazu wird in Zeile 2 die Skalierung mit der Division durch die Standardabweichung durchgeführt. In Zeile 3 wird die Kovarianz der Matrix der nun zentrierten und skalierten Eingangsdaten (also mittelwertfrei und mit Einheitsvarianz) berechnet. Mittels Singulärwertzerlegung wird die Matrix U, die in ihren Spalten die Eigenvektoren γ_i enthält, und die Matrix S, die in ihren Diagonalelementen S_{ii} die dazugehörigen Singulärwerte besitzt, berechnet. Die reduzierte Dimension k in Zeile 5 kann im Voraus durch den Vergleich von totaler Varianz und der Varianz, die durch die reduzierten Komponenten erreicht wird, ermittelt werden. Hierfür können die Singulärwerte verwendet werden:

$$\varepsilon(k) = \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}}. \tag{2.10}$$

k wird nun so gewählt, dass $1 - \varepsilon(k) \leq \epsilon$ ist, mit $0 \leq \epsilon \leq 1$. Dabei entspricht $\epsilon = 1$ keiner Abdeckung und $\epsilon = 0$ einer vollständigen Abdeckung der Varianz der Daten durch k Komponenten. Zum Abschluss werden in Zeile 6 die Daten auf die Komponenten projiziert. Es ergibt sich durch die Linearkombination für jede Stichprobe:

$$\boldsymbol{z} = \sum_{i=1}^{k} \boldsymbol{u}_i \boldsymbol{x}^\top. \tag{2.11}$$

Die verringerte Anzahl an Eigenvektoren bildet die Hauptkomponenten und somit die Basis für den reduzierten Zustandsraum. Die Projektionswerte auf die ermittelten Hauptkomponenten bilden die Merkmale.

Wenn von der Proper Orthogonal Decomposition die Rede ist, dann liegen im Allgemeinen die Daten einer Zeitreihe $\{\boldsymbol{x}(t_i)\}_{i=1}^{T}$ (sogenannte *snapshots*) vor. Die snapshot-Matrix $X_{snap} = [\boldsymbol{x}(t_1) \dots \boldsymbol{x}(t_T)]$ entspricht dann der Matrix X in Zeile 1 der Implementierung 1. Anstelle von Hauptkomponenten spricht man im Zusammenhang mit der POD von Modes. Ziel ist es, die Ordnung komplexer physikalischer Systeme, die durch eine Menge von partiellen Differenzialgleichungen beschrieben werden, zu reduzieren [3]. In der Regel wird nur eine Größe (z. B. Spannungsfeld) betrachtet.

Die POD neigt zu Instabilität im Hinblick auf Parameteränderungen. Chinesta et al. [7] entwickelten basierend auf der Idee der POD eine neue Methode, die Proper Generalized Decomposition (PGD), um echtzeitfähige Simulationen für hochdimensionale Modelle zu ermöglichen. Unter Verwendung der PGD ist es möglich, eindeutige onlinefähige Modelle zu generieren, die alle Lösungen für alle Parameter enthalten. Es existieren jedoch noch Schwierigkeiten bei der Anwendung auf nichtlineare Modelle [8].

2.4.2 Autoencoder

Autoencoder (auch Autoassociative Neural Networks genannt) sind Bottleneck neuronale Netze, deren Ein- und Ausgabe für das überwachte Lernen die gleichen Werte zugewiesen werden. Die nichtlineare Hauptkomponentenanalyse (NLPCA, engl. Nonlinear Principal Component Analysis) basierend auf Autoencodern wurde von Kramer [54] eingeführt.

Abbildung 2.8 stellt die Schichten eines Autoencoders bzw. NLPCA da. Diese bestehen neben der Ein- und Ausgabeschicht aus drei verdeckten Schichten: Die Mapping-Schicht $X \rightarrow Z$, welches die Eingabedaten X auf die Bottleneck-Schicht abbildet; die Bottleneck-Schicht, deren Aktivierungswerte die reduzierten Daten repräsentieren; und die Demap-ping-Schicht ($Z \rightarrow X$), welche die Eingabedaten wieder rekonstruiert. Scholz und Vigário [83] erweiterten die NLPCA um eine hierarchische Ordnung der Merkmale, wie sie auch bei der linearen PCA durch die Eigenwerte gegeben ist. Folglich kann diese NLPCA zur Merkmalsextraktion verwendet werden, da nichtlineare Merkmale mit sichtlich unterschiedlichen Varianzen sortiert extrahiert werden können. Dies erfolgt während des Trainings durch eine hierarchische Fehlerfunktion E_h, die

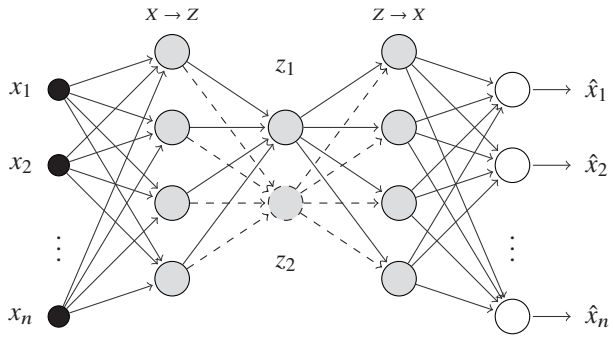


Abbildung 2.8: Topologie der NLPCA nach [83] mit zwei Merkmalen.

beim Training des Netzes erst den mittleren quadratischen Fehler E_1 zwischen Zielwerten X und Netz-Ausgabe $\hat{X}$ (rekonstruierte Zielwerte) unter Verwendung nur eines Bottleneck-Neurons (durchgezogene Linien in Abbildung 2.8) bestimmt. Dann wird gleichermaßen der Fehler $E_{1,2}$ berechnet, der zwei Neuronen in der Bottleneck-Schicht betrachtet (durchgezogene und gestrichelte Linien in Abbildung 2.8), usw. Dabei werden bei der Vorwärtspropagation die Aktivierungswerte bei den nicht betrachteten Bottleneck-Neuronen auf Null gesetzt. Es wird folglich für jede Betrachtung die Eingabe einmal durch das Netz propagiert und jeweils der entstehende Fehler ermittelt. Die Summe der einzelnen Fehler, $E_h = E_1 + E_{1,2} + \ldots + E_{1,\ldots,k}$, wobei k der Anzahl der Bottleneck-Neuronen entspricht, gilt es zu minimieren. Dabei wird bei der Rückwärtspropagation des Gesamtfehlers der Fehlergradient für die jeweils nicht betrachteten Bottleneck-Neuronen auf Null gesetzt. Danach werden die Gewichtsänderungen zu den aktuellen Gewichten aufaddiert und weiter trainiert. Diese Variante der nichtlinearen Hauptkomponentenanalyse wird im weiteren Verlauf mit NLPCA bezeichnet (nicht zu verwechseln mit der Kernel-PCA, die in mancher Literatur auch als nichtlineare PCA bezeichnet wird).

Die Aktivierungsfunktion (siehe auch Abbildung 2.5) eines Neurons x_i der aktuellen Schicht lautet:

$$x_i = h\left(\sum_{j=1}^{M} w_{ij} x_j\right), \tag{2.12}$$

wobei h eine lineare oder sigmoide-Funktion ist, x_j der Wert des j-ten Neuron der vorherigen Schicht, M die Anzahl der Neuronen der vorherigen Schicht und w_{ij} die Gewichte für die Verbindung von j-ten und i-ten Neuron. Die Ausgabe der Mapping-Schicht $X \rightarrow Z$ zusammen mit den Gewichten der Verbindungen von Mapping-Schicht zur Bottleneck-Schicht formen die Eingabe zur Bottleneck-Schicht. Simultan erlernt das Netz die inverse Abbildung $Z \rightarrow X$ über die Demapping-Schicht, um die ursprünglichen Daten wieder aus den reduzierten Daten zu rekonstruieren. Die Aktivierungswerte in der Bottleneck-Schicht entsprechen den reduzierten Daten. Die Bottleneck-Schicht und die Ausgabeschicht enthalten lineare Aktivierungsfunktionen. Die Wahl der Aktivierungsfunktionen in der Mapping- und Demapping-Schicht hängt von der beabsichtigten Dimensionsreduktion ab: linear oder nichtlinear. Lineare Aktivierungsfunktionen werden benutzt, um die Daten in einem linearen Unterraum abzubilden. Dadurch werden ähnliche Ergebnisse erzeugt wie durch die PCA. Um die Daten in einem nichtlinearen Unterraum zu erhalten, müssen in der Mapping- und Demapping-Schicht nichtlineare Aktivierungsfunktionen verwendet werden.

2.5 Principal Function Approximator

Der Principal Function Approximator (PFA) von Senn [87] führt eine nichtlineare Regression von Observablen zu Zustandsvariablen aus bei gleichzeitiger integrierter nichtlinearer Dimensionsreduktion. Dieses Verfahren dient dazu, anhand von Observablen die Zustandsvariablen vorherzusagen und simultan Zustandsmerkmale mit geringerer Dimension als der ursprüngliche Zustand zu extrahieren. Dadurch werden die Eingangsgrößen unter Berücksichtigung der Ausgangsgrößen reduziert, wie auch bei der Partial Least Squares Regression (PLS) [98] oder der Hauptkomponentenregression (PCR, engl. Principal Component Regression) [66], bei denen es sich um lineare Verfahren zur Regression mit simultaner Dimensionsreduktion handelt.

Die Topologie basiert, wie auch der Autoencoder in 2.4.2, auf Bottleneck neuronalen Netzen (mit fünf Schichten: Eingabe-, Mapping-, Bottleneck-, Demapping- und Ausgabeschicht). Anders als beim Autoencoder liegen jedoch in der Ein- und Ausgabeschicht nicht die gleichen Werte vor. Stattdessen erhält der PFA als Eingabe die unabhängigen Variablen und als Ausgabe die abhängigen Variablen der Regressionsanalyse. Im Fall der Zustandsvorhersage anhand der Observablen ist $\boldsymbol{x} = g^{-1}(\boldsymbol{y})$ gesucht und somit sind die Observablen $\boldsymbol{y}$ die Eingabe und die Zustandsvariablen $\boldsymbol{x}$ die Ausgabe für das Netz. Durch die Bottleneckschicht (Abbildung 2.9) werden die Zustandsmerkmale ermittelt und zwar so, dass sie eine reduzierte Darstellung der Observablen bilden und die Zustandsvariablen mit möglichst geringem Fehler rekonstruieren können. Somit soll sichergestellt werden, dass die Zustandsmerkmale die Daten möglichst genau repräsentieren und die Kovarianz zwischen Zustandsmerkmalen und Ein- bzw. Ausgabe maximiert ist.

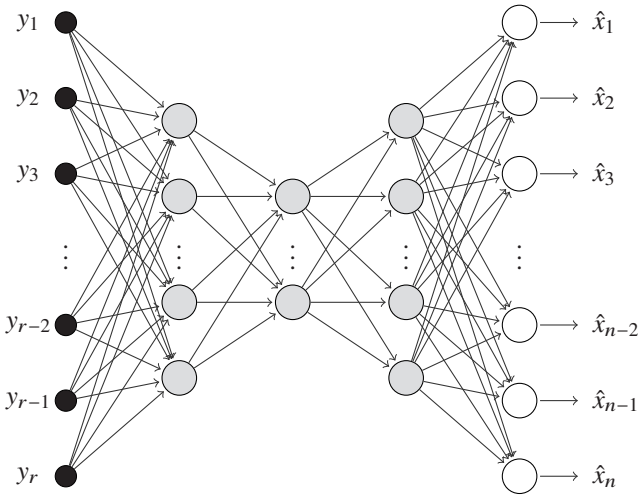


Abbildung 2.9: Topologie des Principle Function Approximators.

Um eine nichtlineare Regression und Dimensionsreduktion durchzuführen, bestehen die Aktivierungsfunktionen in den verdeckten Schichten aus sigmoiden Funktionen $h(x) = \frac{1}{1+e^{-x}}$. Die PFAs werden mit Resilient Backpropagation [74] trainiert. Der Nachteil an solchen gradientenbasierten Optimierungsverfahren ist die Tendenz zum „Steckenbleiben“ in lokalen Optima. Um eine möglichst gute Ausgangsbedingung für das Training zu schaffen, wird daher ein Pretraining angewendet, welches eine Initialisierung der Gewichte ermittelt. Das Pretraining von Senn basiert auf der Idee der inversen NLPCA [84], die nur die Bottleneck-, Demapping- und Ausgabeschichten mit einer Einheitsmatrix der Größe $s \times s$ (s Anzahl der Stichproben) trainiert. Senn erweitert dies durch ein anschließendes Training der Mapping-Schicht mit den soeben ermittelten Aktivierungswerten der Bottleneck-Schicht als Ausgabe. Die durch diese zwei Trainingsphasen ermittelten Gewichte werden als initiale Gewichte für das Training des gesamten Netzes verwendet. In der Mapping-Schicht werden doppelt so viele Neuronen wie in der Demapping-Schicht verwendet. Dadurch wird beabsichtigt, dass die Mapping-Schicht eine höhere Komplexität erhält und eine geringere Kompression aufweist als die Demapping-Schicht, mit dem Ziel einer ergebnisorientierten Dimensionsreduktion.

Die Zustandsmerkmale des PFAs sind nicht, wie bei PLS durch die erklärte Varianz, hierarchisch nach Relevanz geordnet. Die Zustandsmerkmale hängen somit stark von einander ab,

d. h. es ist nicht möglich, ein Zustandsmerkmal getrennt von den anderen zu betrachten bzw. ein Zustandsmerkmal zu entfernen. Hensgen [32] erweiterte die PFAs insofern, dass durch eine sequenzielle Anordnung der PFAs, mit jeweils nur einem Neuron in der Bottleneck-Schicht, tendenziell geordnete unabhängige Zustandsmerkmale extrahiert werden können. Abbildung 2.10 zeigt die Topologie der sequenziellen PFAs. Dabei ist ersichtlich, dass das erste Netz einem PFA mit einem Bottleneck-Neuron entspricht.

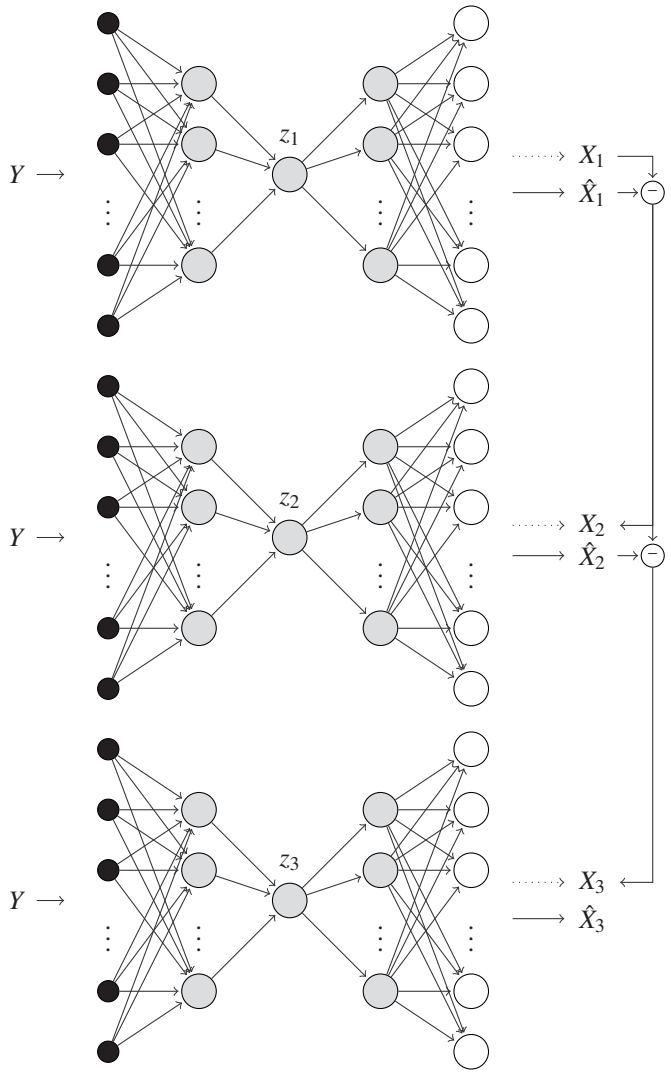


Abbildung 2.10: Topologie des sequenziellen Principle Function Approximators mit drei Zustandsmerkmalen.

Für alle weiteren Netze gilt für die Ausgabeschicht, dass die Zielwerte X_i des $i$ten Netzes sich aus der Differenz zwischen den Zielwerten X_{i-1} und den vorhergesagten Zielwerten $\hat{X}_{i-1}$ der Netzausgabe ergeben: $X_i = X_{i-1} - \hat{X}_{i-1}$ mit $i = 2,3,...,k$. Auf diese Weise wird versucht, mit jedem weiteren Netz die noch verbleibende Varianz durch das Zustandsmerkmal abzudecken. Damit ist auch der Vorteil verbunden, dass die Anzahl an Zustandsmerkmalen nicht vor dem Training bekannt sein muss, sondern jeweils ein weiteres Netz trainiert werden kann, wenn der erwünschte Regressionsfehler noch nicht erreicht ist.

2.6 Symbolische Regression

Symbolische Regression [53] verwendet genetische Algorithmen, um aus Daten nicht nur Modellparameter sondern auch Modellstruktur zu bestimmen. Ziel ist es, durch den – im Vergleich zur linearen und nichtlinearen Regression – hinzugewonnenen Freiheitsgrad der Modellstruktur eine exaktere Anpassung an die Daten zu erreichen. Abbildung 2.11 zeigt ein mögliches Ergebnis im Vergleich zur linearen und nichtlinearen Regression. Bereits vorhandenes Wissen über die Modellstruktur kann zur Ermittlung des Modells mit einfließen. Liegt zum Beispiel, wie bei den Daten in Abbildung 2.11, ein oszillierendes System zugrunde, können zur Suche der Modellstruktur Sinus- oder Cosinusfunktionen vorgegeben werden. Durch diese Möglichkeit kann ein wissenbasiertes Grey-Box-Modell erzeugt werden.

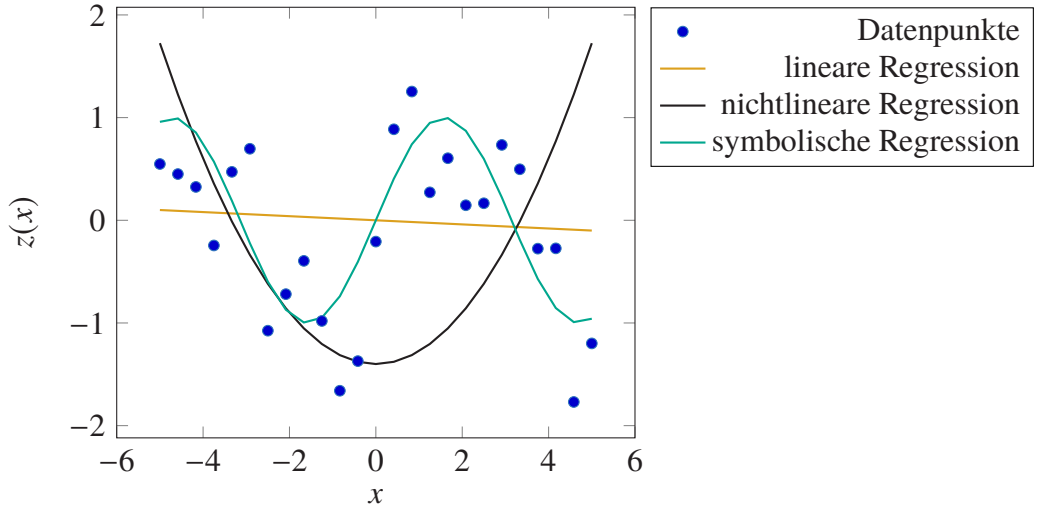


Abbildung 2.11: Beispielhafter Vergleich symbolischer Regression mit linearer und nichtlinearer Regression.

Im klassischen Fall der symbolischen Regression werden diskrete mathematische Gleichungen in Bäumen, wie in Abbildung 2.12, kodiert. Dabei befinden sich in den Knoten mathematische Operatoren, Konstanten oder Variablen. Die Summe zweier Variablen a und b lässt sich folglich aus einen Baum mit drei Knoten darstellen: ein Elternknoten mit der Operation + mit zwei Kinderknoten, jeweils für a und b. Der Vorteil der Baumstruktur besteht darin, dass jeder Elternknoten zusammen mit seinen Kindern für sich gesehen wiederum ein gültiger mathematischer Ausdruck ist. Dies ist für den nachfolgenden Algorithmus der symbolischen Regression von Bedeutung.

Algorithmus Abbildung 2.13 zeigt den Ablauf der symbolischen Regression. Sofern vorhanden werden über Expertenwissen mögliche Operatoren und mögliche Modellstrukturen festgelegt. Anhand dieser wird eine Anzahl zufälliger Bäume (*Individuen*) generiert, die den von Experten festgelegten Vorgaben entsprechen. Jedes einzelne Individuum der *Population* (Gesamtheit aller Individuen) wird durch eine *Zielfunktion* bewertet. Diese Bewertung der Zielfunktion, auch *Fitness* genannt, soll Aufschluss darüber geben, wie gut ein Individuum die gegebenen Daten anpasst. Für jedes Individuum einer *Generation* wird mit einer bestimmten

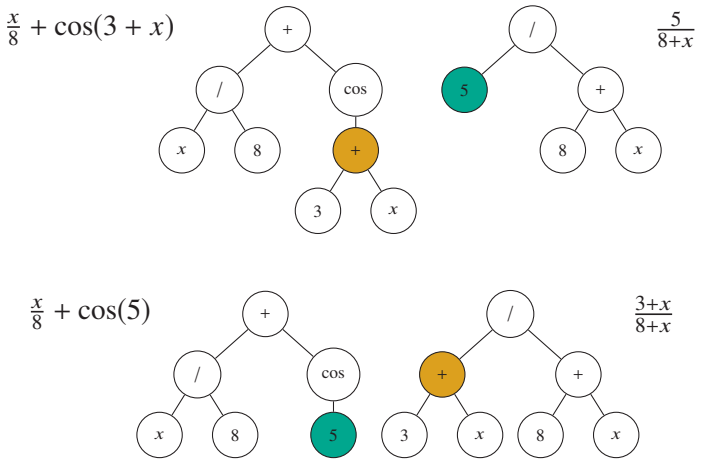


Abbildung 2.12: Baumstruktur der symbolischen Regression und die Operation Crossover.

Wahrscheinlichkeit beschlossen, wie es weiter verarbeitet wird. Mögliche Operationen sind *Selektion*, *Mutation* und *Crossover*. Bei der Selektion wird das Individuum basierend auf seiner Fitness selektiert oder nicht. Selektierte Individuen werden in die nächste Generation mit aufgenommen. Bei der Mutation werden zufällige Knoten im Baum verändert, dabei kann z. B. eine Addition zu einer Division werden. Das dadurch neu entstandene Individuum wird in die nächste Generation mit aufgenommen. Das Crossover ist in Abbildung 2.12 dargestellt. Hierbei werden zufällige Unterbäume zweier Individuen vertauscht. Auch hier werden die zwei neuen Individuen in die neue Generation aufgenommen. Danach werden alle Individuen der neuen Generation durch die Fitnessfunktion bewertet. Dies wird wiederholt bis ein Abbruchkriterium (z. B. maximale Anzahl an Generationen erreicht) erfüllt ist oder ein zufriedenstellendes Ergebnis erreicht wurde.

Ein Ergebnis ist optimal, wenn der Regressionsfehler minimal ist und die mathematische Gleichung minimale Komplexität hat. Die Komplexität ergibt sich aus der Art und der Anzahl der Knoten, wobei die Gewichtung durch Experten festgelegt werden kann. Dabei kann u. a. die Komplexität von Strukturelementen, die erwünscht sind, niedriger gewichtet werden als jene, die für das gesuchte Modell ungeeignet sind. Komplett unerwünschte Strukturelemente können von vornherein von der Suche ausgeschlossen werden. Ein geringer Regressionsfehler ist oft mit einer hohen Komplexität des Ergebnisses verbunden bzw. ein hoher Regressionsfehler mit einer geringen Komplexität. Die in Abbildung 2.14 eingezeichnete *Pareto-Front* zeigt beispielhaft die Menge aller optimalen Lösungen, d. h. für die jeweilige Komplexität den kleinsten Regressionsfehler, der erreicht wurde.

2.7 Zusammenfassung

Dieses Kapitel gab einen Überblick über Basismethoden zur Bildung datengetriebener Prozessmodelle. Als Ausgangspunkt der Modellbildung wurden hier theoretische Modelle zur Datenerzeugung verwendet, um daraus mittels experimenteller Modellbildung ein onlinefähi-

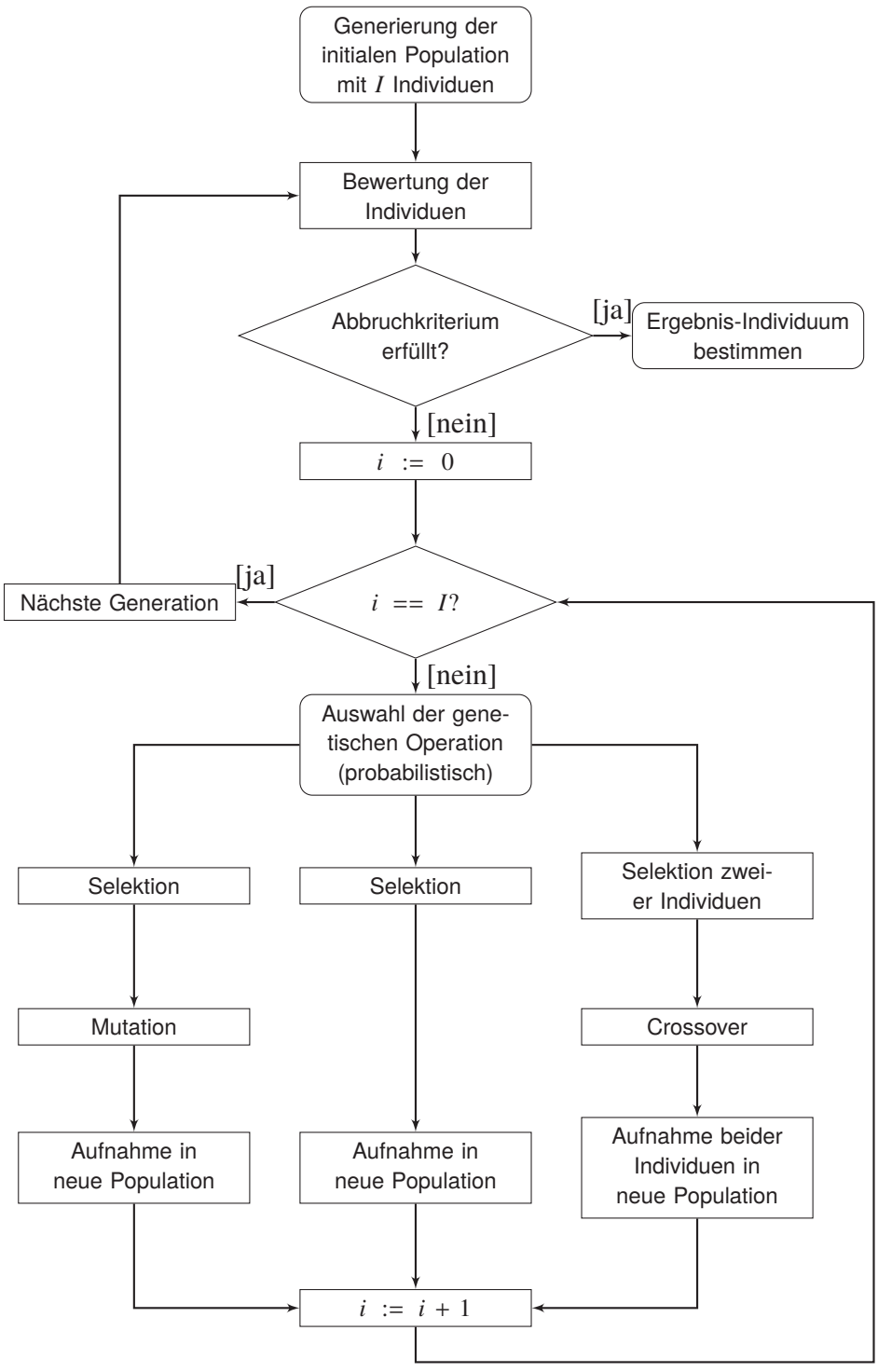


Abbildung 2.13: Ablauf der symbolischen Regression.

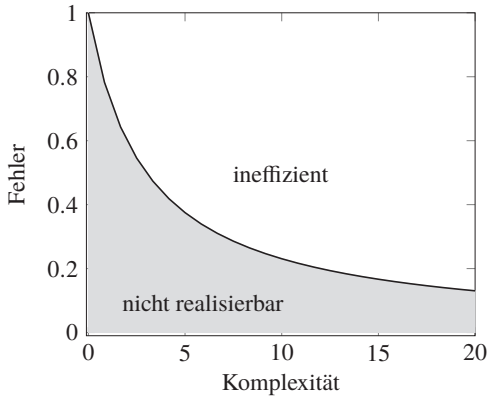


Abbildung 2.14: Prinzip der Pareto-Front.

ges, dynamisches und nichtlineares Prozessmodell zu ermitteln. Verfahren zur experimentellen Modellbildung wurden aufgeführt und ihre Vor- und Nachteile aufgelistet. Für große Datenmengen, die aus den Simulationen der theoretischen Modelle gezogen werden, eignen sich besonders künstliche neuronale Netze, da sie vielfältig konfiguriert werden können und zum Lernen von Transformationsfunktionen ebenso wie zur Dimensionsreduktion eingesetzt werden können. Weiterhin wurde die symbolische Regression näher betrachtet, da diese es, im Gegensatz zu künstlichen neuronalen Netzen, zusätzlich ermöglicht, Abbildungen zu erlernen, deren Parametern physikalische Bedeutung zugewiesen werden kann. Kapitel 4 betrachtet weiter die Anwendbarkeit von Dimensionsreduktion mittels KNNe und der Ermittlung eines nichtlinearen Prozessmodells im reduzierten Merkmalsraum mittels symbolischer Regression. Für die Verwendung des reduzierten, datengetriebenen, nichtlinearen Modells zur Zustandsbeobachtung werden im nachfolgenden Kapitel 3 Verfahren zur Zustandsverfolgung nichtlinearer Prozesse betrachtet.

3 Zustandsverfolgung nichtlinearer Prozesse

Die Verfolgung des Zustandes von nichtlinearen Prozessen im *Zustandsraum* kann durch Zustandsbeobachter erfolgen. Zustandsbeobachter dienen zur Ermittlung nicht messbarer *Zustandsvariablen*, die zur Systembeschreibung im Zustandsraum dienen. Die Anzahl der Zustandsvariablen gibt die *Dimension* des Zustandsraums vor. Die Zustandsdifferenzialgleichungen, die das Zustandsraummodell beschreiben, bestehen aus Differenzialgleichungen erster Ordnung. Diese können auch durch Umwandlung einer Systembeschreibung höherer Ordnung in gekoppelte Differenzialgleichungen erster Ordnung erstellt werden.

Die Zustandsdifferenzialgleichungen werden im weiteren Verlauf auch *Prozessmodell* genannt, da sie das Verhalten des Prozesses modellieren. Genauso werden die Ausgangsgleichungen aufgrund der Abbildung von Zustandsvariablen auf Ausgangsgrößen (hier Messgrößen) als *Messmodell* bezeichnet. Prozessmodell sowie Messmodell werden vom klassischen Zustandsbeobachter (wie Luenberger-Beobachter und Kalman-Filter) benötigt, um die Zustandsvariablen zu rekonstruieren.

Abbildung 3.1 zeigt, wie der aus dem Prozessmodell und den bekannten Eingangsgrößen $\boldsymbol{u}_t$ und Messgrößen $\boldsymbol{y}_t$ ermittelte Zustand $\hat{\boldsymbol{x}}_t$ zum Zeitpunkt t dem Zustandsregler übergeben wird. Die

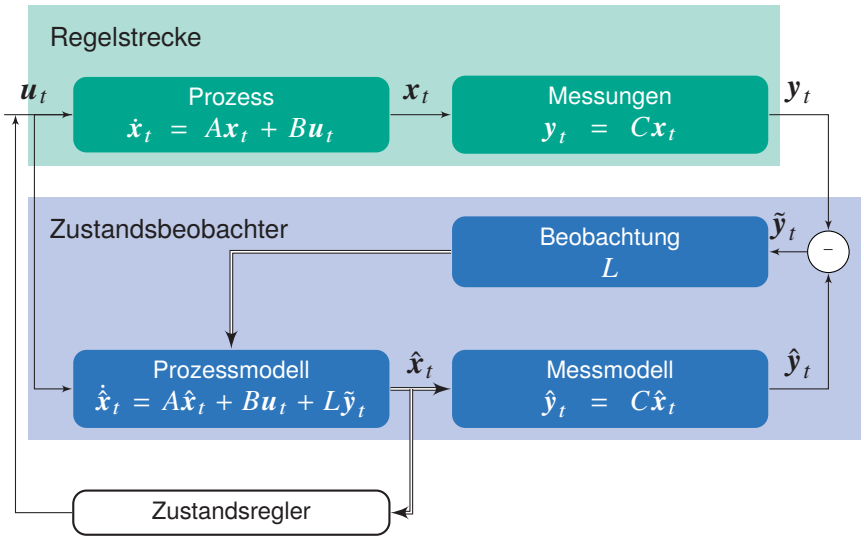


Abbildung 3.1: Luenberger-Beobachter parallel zur Regelstrecke.

Regelung mittels des rekonstruierten Zustands anstelle der Abweichung zwischen Messgrößen bzw. Stellgrößen v_t und Führungsgröße ω_t, wie bei einem *Standardregelkreis* in Abbildung 3.2, ermöglicht es, zeitlich schneller auf die Dynamik des Systems zu reagieren, da der Zustand früher zur Verfügung steht als die Messgrößen. Des Weiteren sind Zustandsbeobachter weniger empfindlich gegenüber dem Ausfall von Messgrößen oder Latenzen bei der Messung als ein

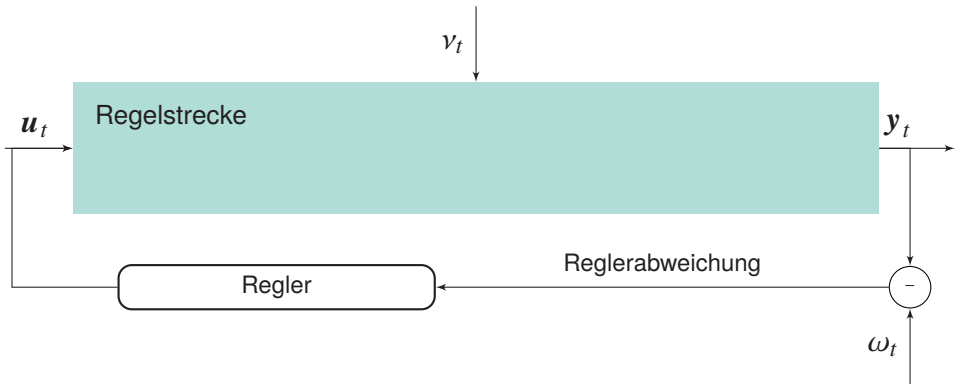


Abbildung 3.2: Aufbau eines Standardregelkreises.

Standardregelkreis mit Rückführung der Messgrößen. Zusätzlich kann mittels Zustandsbeobachter die zeitliche Entwicklung des Prozesszustandes direkt beobachtet werden.

Eine Zustandsregelung eignet sich besonders für zeitvariante, nichtlineare und Mehrgrößensysteme sowie Systeme höherer Ordnung. Eine Voraussetzung für die Zustandsregelung ist die Steuerbarkeit aller Zustandsvariablen. Des Weiteren muss die Regelstrecke beobachtbar sein, damit ein Zustandsbeobachter zum Einsatz kommen kann, was jedoch bei den meisten technischen Regelstrecken gegeben ist. Steuerbarkeit und Beobachtbarkeit nichtlinearer Systeme werden in Kapitel 5.3 thematisiert.

Für die Beschreibung, Analyse und Entwurf sowie Konzepte der Stabilität, Steuer- und Beobachtbarkeit von Beobachtern für *lineare Systeme* im Zeitbereich existieren bereits vollständig ausgearbeitete Theorien [22, 61]. In der Praxis handelt es sich jedoch meist um *nichtlineare Systeme*. Da nichtlineare Systeme wesentlich komplexer sind, gibt es keine einheitliche Lösung für den Beobachterentwurf nichtlinearer Systeme. Oft gelten die Verfahren nur für eine spezielle Klasse von nichtlinearen Systemen oder für bestimmte Anwendungsfälle.

In diesem Kapitel wird als Grundlage für die nichtlinearen Zustandsbeobachter zuerst ein kurzer Einblick in die linearen Zustandsbeobachter gegeben. Darauf aufbauend werden robuste nichtlineare Zustandsbeobachter basierend auf Prozess- und Messmodell dargestellt. Robuste Beobachter zeichnen sich durch ihre Robustheit gegenüber möglichen Modellunsicherheiten, unvollständig modellierter Systemdynamik und Parameteränderungen, sowie unbekannten Störungen, die auf den Prozess einwirken, aus. Anschließend werden Verfahren vorgestellt, die Prozess- und Messmodell in einem einzigen Black-Box-Modell realisieren.

3.1 Grundlagen: Lineare Zustandsbeobachter

Die weitverbreitesten Zustandsbeobachter bzw. Zustandschätzer sind der Luenberger-Beobachter und das Kalman-Filter. Beide basieren auf Matrix-Darstellung des Prozess- und Messmodells sowie der Parallelschaltung des Beobachters zur Regelstrecke. Beide dienen als Grundlage zur Erstellung eines nichtlinearen Beobachters und werden hier kurz beschrieben. Das zugrunde-

liegende System ist ein lineares dynamisches System, das im Zeitbereich diskretisiert wird: aus $\boldsymbol{x}(t)$ wird $\boldsymbol{x}_t$ mit $_t$ als aktuellem Zeitpunkt und $_{t-1}$ als vorherigem Zeitpunkt.

Luenberger-Beobachter Abbildung 3.1 zeigt den Luenberger-Beobachter, welcher dem klassischen Beobachter für lineare Systeme entspricht. Bereits 1964 hat Luenberger [59] die Theorie zur Parallelschaltung eines Beobachters zur Regelstrecke beschrieben. Er zeigt darin, dass der nicht messbare Zustand durch Beobachtung der Ein- und Ausgangsgrößen des Systems rekonstruiert werden kann. Dabei werden Beziehungen zwischen den Zustandsvariablen und den Ein- und Ausgangsgrößen in Matrizen und Vektoren dargestellt. Für das Prozessmodell und Messmodell, mit Systemmatrix A, Eingangsmatrix B und Ausgangsmatrix C, ergibt sich:

$$\begin{aligned} \dot{\boldsymbol{x}}_t &= A\boldsymbol{x}_t + B\boldsymbol{u}_t \\ \boldsymbol{y}_t &= C\boldsymbol{x}_t \end{aligned} \tag{3.1}$$

mit $\boldsymbol{x}_t \in \mathbb{R}^n$ für die Zustandsvariablen, $\boldsymbol{u}_t \in \mathbb{R}^m$ und $\boldsymbol{y}_t \in \mathbb{R}^r$ für die Eingangs- bzw. Ausgangsgrößen und dementsprechend $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ und $C \in \mathbb{R}^{r\times n}$. Es kann davon ausgegangen werden, dass $m \leq n$ ist, sprich die Anzahl der Eingangsgrößen ist nicht größer als die Anzahl an Zustandsvariablen. Für das Prozessmodell und Messmodell des Beobachters gilt dann:

$$\begin{aligned} \dot{\hat{\boldsymbol{x}}}_t &= A\hat{\boldsymbol{x}}_t + B\boldsymbol{u}_t + L(\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t) \\ \hat{\boldsymbol{y}}_t &= C\hat{\boldsymbol{x}}_t \end{aligned} \tag{3.2}$$

mit $\hat{\boldsymbol{x}}$ als rekonstruierten Zustandsvariablen und $\hat{\boldsymbol{y}}$ als rekonstruierten Ausgangsgrößen. Die Differenz zwischen den wahren Messwerten der Regelstrecke $\boldsymbol{y}$ und den rekonstruierten Messwerten des Beobachters $\hat{\boldsymbol{y}}$ wird durch die Beobachtungsmatrix L bewertet und zur Zustandsgleichung zurückgeführt. Durch $L(\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t) = LC(\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t)$ ergibt sich:

$$\dot{\hat{\boldsymbol{x}}}_t = (A - LC)\hat{\boldsymbol{x}}_t + B\boldsymbol{u}_t + L\boldsymbol{y}_t. \tag{3.3}$$

Der Beobachtungsfehler $\boldsymbol{e}_t = \boldsymbol{x}_t - \hat{\boldsymbol{x}}_t$ erfüllt die Gleichung $\dot{\boldsymbol{e}}_t = (A - LC)\boldsymbol{e}_t$. Die Matrix L gilt es so zu bestimmten, dass alle Eigenwerte der Matrix $(A - LC)$ negative Realteile besitzen, dann gilt $\lim_{t\to\infty} \|\boldsymbol{e}_t\| = \boldsymbol{0}$. Durch die Anwendung der Beobachtungsmatrix bei der Rückführung der Differenz zwischen wahren Messwerten und rekonstruierten Messwerten kann auf eigene Ungenauigkeiten des Beobachters regiert werden.

Kalman-Filter Das Kalman-Filter kann ebenfalls als linearer Beobachter eingesetzt werden. Er wurde von Swerling [92] und Kálmán [46] zur Filterung und Vorhersage in linearen Systemen mit zusätzlichen Störungen entwickelt. Diese Störungen wirken in Form von normalverteiltem Rauschen ν und ρ auf das Prozessmodell und Messmodell ein:

$$\begin{aligned} \dot{\boldsymbol{x}}_t &= A\boldsymbol{x}_t + B\boldsymbol{u}_t + \nu_t \\ \boldsymbol{y}_t &= C\boldsymbol{x}_t + \rho_t. \end{aligned} \tag{3.4}$$

Die Zustandsgleichung und Ausgangsgleichung des Kalman-Filters entsprechen denen des Luenberger-Beobachters in (3.2). Das Kalman-Filter und der Luenberger-Beobachter unterscheiden sich dann nur in der Bestimmung der Beobachtungsmatrix L. Beim Luenberger-Beobachter werden die Eigenwerte der Matrix $(A - LC)$ geeignet gewählt, um die Matrix L festzulegen. Das Kalman-Filter hingegen bewertet die Differenz zwischen Messung und rekonstruierter Messung basierend auf der Größe des Rauschens. Wenn großes Rauschen bzw. große Unsicherheiten beim vorherigen Zustand vorliegen, dann wird die rekonstruierte Messung stärker gewichtet. Liegen jedoch kleines Rauschen bzw. kleine Unsicherheiten beim vorherigen Zustand vor, dann wird die wahre Messung mehr gewichtet. Dadurch wird versucht, den Einfluss der Rauschgrößen ν und ρ auf den Beobachtungsfehler $\boldsymbol{e}$ minimal zu halten. Die Schätzung des aktuellen Zustandes wird durch den Mittelwert und die Kovarianz repräsentiert. Dabei ist die grundlegende Idee, dass die Schätzung des aktuellen Zustandes aus Mittelwert und Kovarianz des vorherigen Zeitschrittes sowie den Ausgangsgrößen und den Eingangsgrößen bestimmt wird. Der Algorithmus wird ausführlich in Anhang A.1 erläutert.

3.2 Erweitertes Kalman-Filter

Das erweiterte Kalman-Filter (EKF) ist eine nichtlineare Erweiterung des Kalman-Filters basierend auf Linearisierung. Im Gegensatz zur Matrizen-Darstellung des linearen Kalman-Filters für das Prozess- und Messmodell (3.4) werden diese beim erweiterten Kalman-Filter durch nichtlineare Funktionen f und g abgedeckt:

$$\begin{aligned} \dot{\boldsymbol{x}}_t &= f(\boldsymbol{u}_t, \boldsymbol{x}_{t-1}) + \nu \\ \boldsymbol{y}_t &= g(\boldsymbol{x}_t) + \rho \end{aligned} \tag{3.5}$$

mit normalverteiltem Prozess- und Messrauschen, ν und ρ. Der wesentliche Unterschied, der dadurch im Vergleich zum linearen Kalman-Filter entsteht, ist die Tatsache, dass die Annahme über $\boldsymbol{x}_t$ nur approximiert ist und nicht exakt. Es kann somit nicht der exakte Nachfolger, sondern nur dessen Mittelwert und die Kovarianz bestimmt werden. Dies führt jedoch in nichtlinearen Fällen zu besseren Ergebnissen als die Anwendung des linearen Kalman-Filters auf nichtlineare Probleme [94]. Für Zustand und Messwerte wird eine Normalverteilung angenommen.

Algorithmus Der Algorithmus basiert auf der Linearisierung der Funktionen f und g durch jeweils eine Tangente zu f bzw. g. Die Linearisierung erfolgt in der Regel durch Berechnung der Jacobi-Matrix (siehe Anhang B.1). Für das Prozessmodell wird die Jacobi-Matrix F_{t-1} ermittelt.

$$F_{t-1} = \frac{\partial f(\boldsymbol{u}_t, \boldsymbol{\mu}_{t-1})}{\partial \boldsymbol{\mu}_{t-1}} \tag{3.6}$$

mit $\boldsymbol{\mu}_{t-1}$ als Mittelwert des vorherigen Zeitschrittes als Erwartungswert für den Zustand. Für die Jacobi-Matrix des Messmodell gilt:

$$G_t = \frac{\partial g(\hat{\boldsymbol{\mu}}_t)}{\partial \hat{\boldsymbol{\mu}}_t}, \tag{3.7}$$

wobei $\hat{\boldsymbol{\mu}}_t = f(\boldsymbol{u}_t, \boldsymbol{\mu}_{t-1})$ entspricht.

Die meisten technischen Systeme werden durch ein kontinuierliches System mit diskreten Messungen beschrieben. Abbildung 3.3 zeigt für diesen Fall die Abfolge des erweiterten Kalman-Filters. Zuerst werden Mittelwert μ_0 und Kovarianz P_0 des Zustandes, die Kovarianz des Prozess-

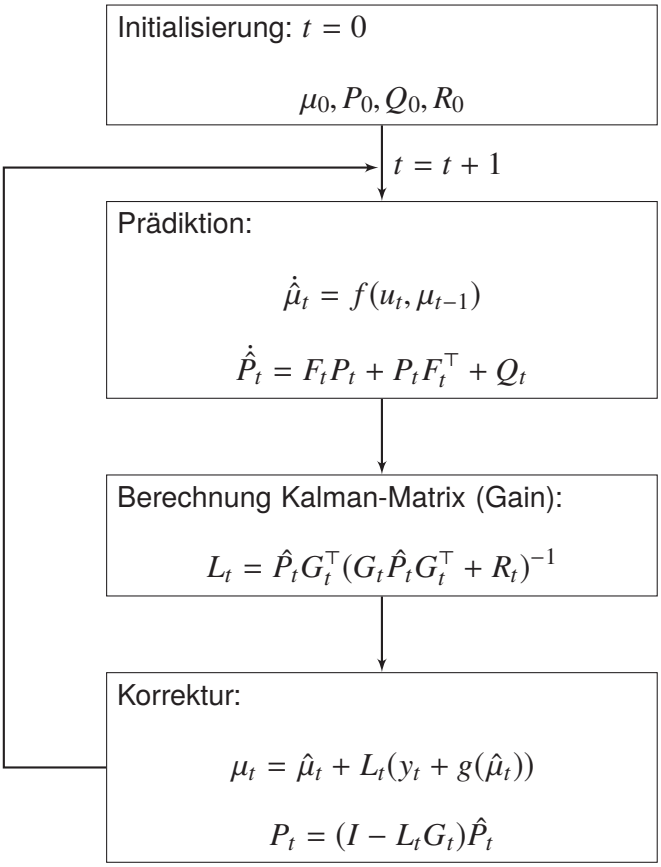


Abbildung 3.3: Ablauf des erweiterten Kalman-Filters.

rauschens Q sowie die Kovarianz des Messrauschens R initialisiert. Diese Initialisierung kann stark von der Problemstellung abhängen, wobei hier der initiale Mittelwert mit dem Anfangszustand, sofern bekannt, oder einem Einheitsvektor initialisiert wird und die initiale Kovarianz mit einer Einheitsmatrix. Zur Vorhersage des Mittelwertes und der Kovarianz muss jeweils $\dot{\hat{\boldsymbol{\mu}}}_t$ und $\dot{\hat{P}}_t$ gelöst werden, sodass $\hat{P}_t$ und $\hat{\boldsymbol{\mu}}_t$ für $t > 0$ zur Verfügung stehen. Dies kann u. a. mittels explizitem Euler-Verfahren erfolgen. Anhand der ermittelten Kovarianz $\hat{P}_t$, der Jacobi-Matrix des Messmodells G_t und der Kovarianz des Messrauschens R_t wird die *Beobachtungsmatrix* (auch *kalman gain*) L berechnet. Nun muss mittels dieser Beobachtungsmatrix die *Korrektur* des Mittelwerts und der Kovarianz vorgenommen werden. Der Mittelwert entspricht dem vorhergesagten Zustand für diesen Zeitschritt. Diese drei Schritte – Vorhersage, Berechnung der Beobachtungsmatrix und die Korrektur – werden für jeden Zeitschritt ausgeführt, um den Zustand zu jedem Zeitpunkt korrigiert vorherzusagen.

3.3 Unscented Kalman-Filter

Wie auch beim erweiterten Kalman-Filter wird beim unscented Kalman-Filter (UKF) die Verteilung der möglichen Zustandsvariablen und Messwerte durch Normalverteilungen dargestellt. Anders beim Unscented Kalman-Filter ist, dass es für die Parametrisierung von Mittelwert und Kovarianz diskrete Stichproben verwendet ohne, wie beim erweiterten Kalman-Filter, auf Linearisierung zurückzugreifen [44,45]. Hierbei werden aus der Dichtefunktion der Normalverteilung für die aktuelle Zustandsvorhersage $2n+1$ gewichtete Stichproben, sogenannte *Sigmapunkte*, gezogen, wobei n der Dimension der Normalverteilung entspricht. Abbildung 3.4 zeigt beispielhaft drei Sigmapunkte die für eine eindimensionale Normalverteilung $p(x)$ bestimmt wurden. Diese Werte werden, anstelle der Linearisierungen beim EKF, durch die nichtlinearen Funktionen des Prozessmodells propagiert. Durch das Ermitteln der Sigmapunkte kann der Mittelwert und die Kovarianz des vorhergesagten Zustandes exakter als beim erweiterten Kalman-Filter berechnet werden, bei gleichem Rechenaufwand.

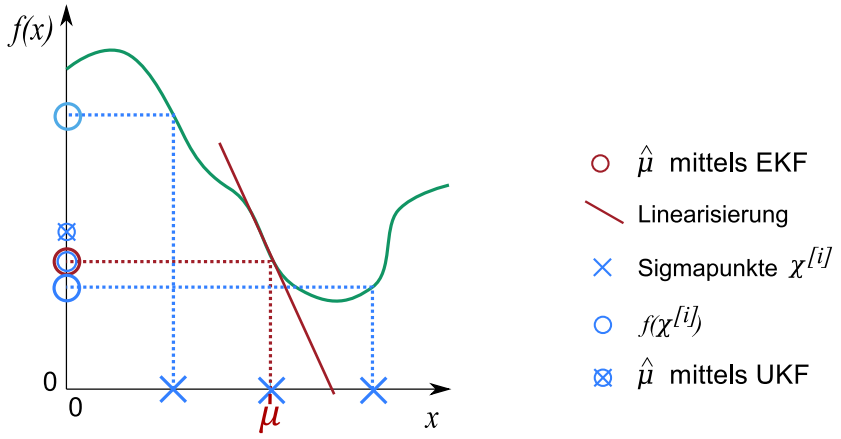


Abbildung 3.4: Vergleich des ermittelten Mittelwerts für die Zustandsvariable durch UKF mittels Sigmapunkte χ und EKF mittels Linearisierung.

Algorithmus Abbildung 3.5 zeigt den Ablauf des unscented Kalman-Filter. Wie beim EKF in Abschnitt 3.2 werden zuerst Mittelwert und Kovarianz des Anfangszustandes und die Kovarianzen des Rauschen initialisiert. Aus den initialen Werten werden die Sigmapunkte durch die sogenannte unscented Transformation erzeugt. In der Regel sind die Sigmapunkte am Mittelwert und symmetrisch entlang der Hauptachse der Kovarianz platziert (siehe Abbildung 3.4). Die Berechnung der Sigmapunkte wird in Anhang A.2 beschrieben.

Im Prädiktionsschritt für den Zustand werden die ermittelten Sigmapunkte χ_{t-1} anstelle des vorherigen Zustandes $\boldsymbol{x}_{t-1}$ durch das rauschfreie Prozessmodell $f(\boldsymbol{u}_t, \chi_{t-1})$ propagiert. Die Kovarianz und der Mittelwert werden durch die propagierten Sigmapunkte und das jeweilige Gewicht ω_c und ω_m, für die Berechnung der Kovarianz und des Mittelwerts, durch Linearkombination berechnet. Die Kovarianz für das Prozessrauschen Q wird auf die gewichtete Kovarianz $(\hat{\chi}_t^{[i]} - \hat{\mu}_t)(\hat{\chi}_t^{[i]} - \hat{\mu}_t)^\top$ addiert.

Anschließend werden neue Sigmapunkte aus der vorhergesagten Normalverteilung des Prozessmodells extrahiert. Diese umfassen jetzt die gesamte Unsicherheit nach der Prädiktion des Zustands.

Bei der Prädiktion der Observablen werden die neu berechneten Sigmapunkte durch das Messmodell propagiert und, wie bei der Vorhersage des Zustandes, die Messwerte durch den Mittelwert und die Kovarianz, die sich durch die gewichteten propagierten Sigmapunkte ergeben, berechnet. Zur ermittelten Kovarianz des Messmodells wird die Kovarianz der Messrauschens R_t addiert. Die Messkovarianz S_t entspricht $G_t \hat{P}_t G_t^\top + R_t$ beim EKF.

Die Kreuzkovarianz

$$\hat{P}_t^{x,y} = \sum_{i=0}^{2n} \omega_c^{[i]} (\hat{\chi}_t^{[i]} - \hat{\mu}_t)(\hat{\varsigma}_t^{[i]} - \hat{y}_t)^\top \tag{3.8}$$

zwischen Zustand und Messung wird verwendet, um die Beobachtermatrix L_t zu berechnen. Diese Kreuzkovarianz entspricht $\hat{P}_t G_t^\top$ beim EKF. Die Korrektur entspricht somit der des EKFs.

Vergleich mit anderen nichtlinearen Zustandsschätzern Im Vergleich zu EKF erzielt das UKF bessere Ergebnisse, da das UKF ohne Linearisierung auskommt. Bei linearen Systemen kann gezeigt werden, dass die Ergebnisse des UKF und des Kalman-Filters identisch sind, während bei nichtlinearen Systemen das UKF gleiche oder bessere Ergebnisse als das EKF zeigt [94]. Weiterere erwähnenswerte nichtlineare Zustandsschätzer sind Partikel-Filter, auch unter *Sampling Importance Resampling* [26] und *Sequenzielle Monte-Carlo-Methode* bekannt. Partikel-Filter verfolgen das zeitliche Verhalten eines Zustandes dadurch, dass die Wahrscheinlichkeitsdichte durch eine Menge von Stichproben, sogenannte Partikel, dargestellt wird. Dadurch können auch nicht normalverteilte Wahrscheinlichkeiten abgedeckt werden. Die Auswahl der Sigmapunkte beim UKF wird, anders als die Partikel beim Partikel-Filter, deterministisch bestimmt. Das Partikel-Filter wählt die Stichproben zufällig. Wenn man von einer normalverteilten Dichtefunktion ausgehen kann, dann ist das UKF effizienter als das Partikel-Filter.

3.4 Beobachter basierend auf Black-Box-Modell

Die Ermittlung eines Zustandsbeobachters auf Basis eines Black-Box-Modells steht im starken Zusammenhang mit der experimentellen Modellbildung. Dabei ist jedoch nicht ein Modell zu identifizieren, das den Zustand in Abhängigkeit von Eingangsgrößen (*Prozessparameter*) und Zeit beschreibt, sowie das Messmodell in Abhängigkeit vom aktuellen Zustand, sondern ein Modell, das den Zustand in Abhängigkeit von Prozessparametern und Messungen ermittelt.

Liegen keine Informationen bzw. kein mathematisches Modell des Systems vor, können Black-Box-Modelle zum Einsatz kommen. Diese statistischen Modelle werden anhand der zugrundeliegenden Daten aus Simulationen oder Experimenten ermittelt und bilden eine Transformationsfunktion von Eingangsgrößen auf Ausgangsgrößen. Wie in Kapitel 2.2 beschrieben kann ein nichtlineares Prozessmodell zur Prognose sowie zur Simulation der zukünftigen Ausgangsgrößen verwendet werden. Bei der Prognose handelt es sich um ein sogenanntes *seriell-paralleles* oder statisches Modell, bei dem anhand des Prozesseingangs $\boldsymbol{u}_t$ und des Prozessausgangs $\boldsymbol{y}_t$

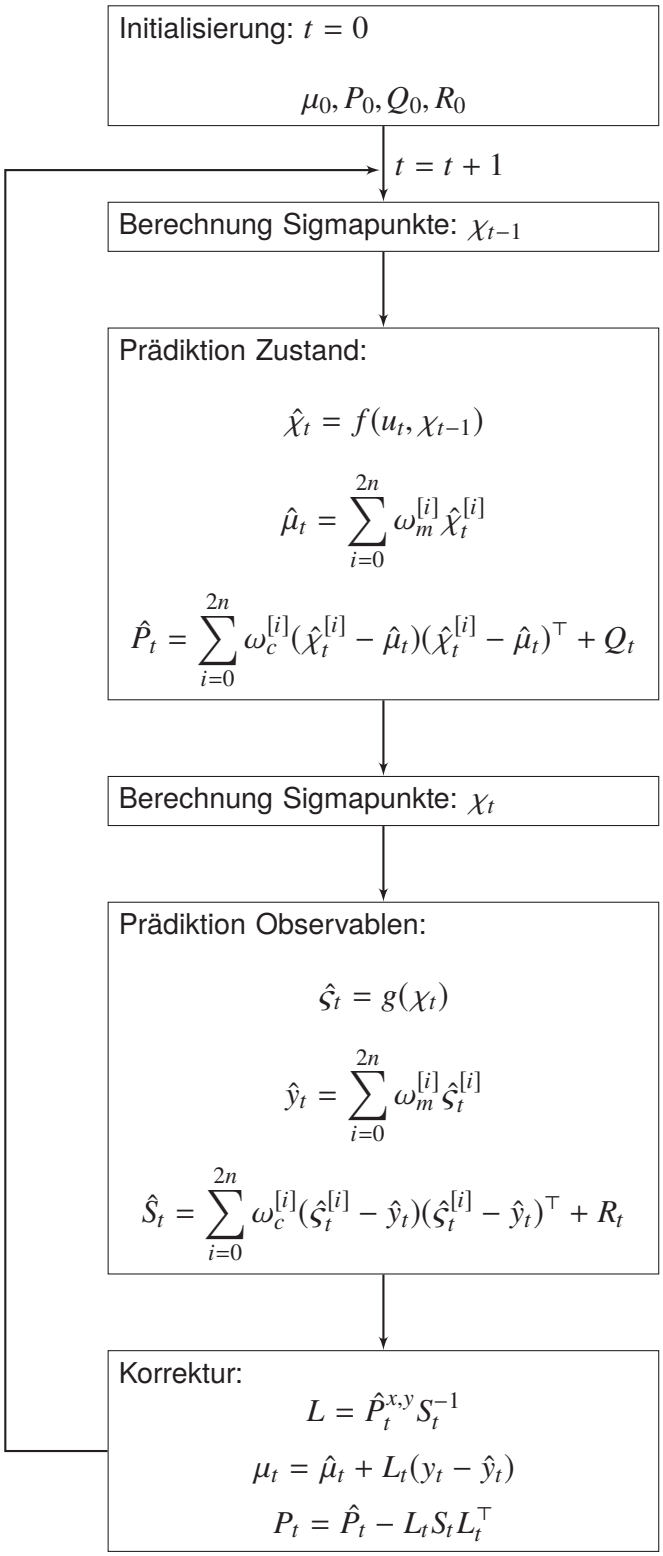


Abbildung 3.5: Ablauf des unscented Kalman-Filters.

ein oder mehrere Schritte in die Zukunft vorhergesagt werden. Voraussetzung dafür ist, dass die Ausgangsgrößen während des Prozesses messbar sind. Bei einem Modell zweiter Ordnung und der Vorhersage eines Schrittes gilt dann:

$$\hat{\boldsymbol{x}}_t = f(\boldsymbol{u}_{t-1}, \boldsymbol{u}_{t-2}, \boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-2}). \tag{3.9}$$

Dies erfordert die Messung der Ausgangsgrößen während der Ausführung. Die Simulation, auch paralleles oder dynamisches Modell genannt, erfordert hingegen die Messungen nicht zwingend, da der zukünftige Ausgang allein anhand des Prozesseingangs und Modellausgangs des vorherigen Zeitschrittes simuliert wird. Der vorherige Modellausgang (sozusagen das Simulationsergebnis) wird somit auf das Modell zurückgeführt. Dementsprechend ergibt sich für ein Modell zweiter Ordnung:

$$\hat{\boldsymbol{x}}_t = f(\boldsymbol{u}_{t-1}, \boldsymbol{u}_{t-2}, \hat{\boldsymbol{y}}_{t-1}, \hat{\boldsymbol{y}}_{t-2}). \tag{3.10}$$

Simulation wird benötigt, wenn der Ausgang nicht messbar ist. Das kann der Fall sein, wenn der Prozess simuliert wird, ohne direkt an den wahren Prozess gekoppelt zu sein, oder wenn Messausfälle oder Latenzen zu erwarten sind.

Zur Korrektur des Prozessmodells kann die Differenz $e_t^{(p)}$ zwischen wahren und simulierten Ausgangsgrößen bzw. die Differenz $e_t^{(sp)}$ zwischen wahren und vorhergesagten Ausgangsgrößen, zum Zeitpunkt t, wichtige Informationen liefern. Wird das Modell mittels Training ermittelt, wird anhand einer Kostenfunktion versucht, diese Differenz zu minimieren. Abbildung 3.6 zeigt eine Gegenüberstellung der beiden Arten. Deutlich zu sehen ist die jeweilige *Zurückführung* der Ausgangsgrößen beim statischen Modell bzw. der vorhergesagten Ausgangsgrößen beim dynamischen Modell.

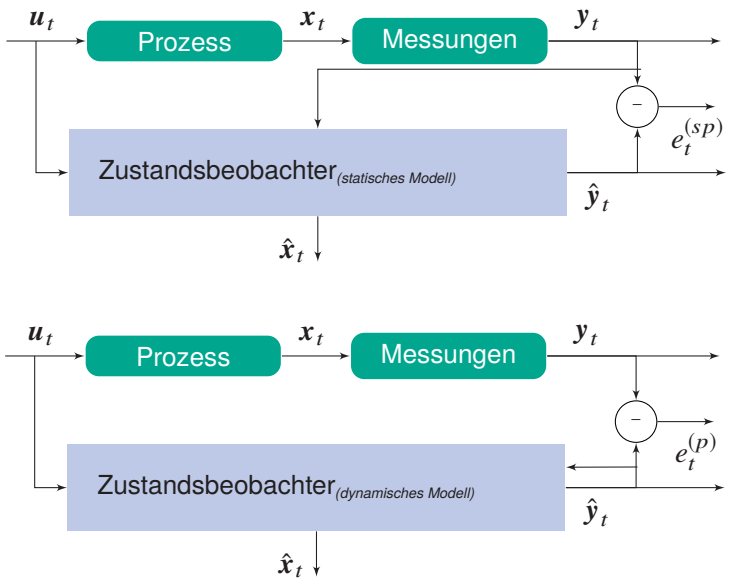


Abbildung 3.6: Oben: Statisches Modell zur Prädiktion zukünftiger Ausgangsgrößen. Unten: Dynamisches Modell zur Simulation zukünftiger Ausgangsgrößen.

Beispiel: Statischer Black-Box-Beobachter Von einem statischen Black-Box-Beobachter ist in dieser Arbeit die Rede, wenn ein Zustandsbeobachter aus einem statischen Black-Box-Modell besteht, welches den Zustand anhand der gegebenen Daten, wie Eingangsdaten und Observablen, rekonstruiert. Die typische Herangehensweise für einen statischen Beobachter nichtlinearer Systeme ist, lineare Prozessgleichungen zu verwenden und die auftretende Nichtlinearität zusätzlich zu approximieren. Dafür muss jedoch der lineare Prozessanteil bekannt und konstant sein.

Senn [87] stellt einen generischen Beobachter vor, der in Form eines statischen nichtlinearen Black-Box-Modells den nicht messbaren Zustand anhand der Ein- und Ausgangsgrößen vorhersagt. Hierbei werden mittels künstlicher neuronaler Netze die funktionalen Abhängigkeiten zwischen Messgrößen und Zustandsvariablen erlernt (Abbildung 3.7). Zuerst werden Daten, Zustandsvariablen und Messgrößen, über zeitliche Verläufe des Prozesses mittels numerischer Simulationen extrahiert und aus Experimenten gewonnen. Dies wird für unterschiedliche relevante Eingangsgrößen (Prozessparameter) oder, im Falle von zeitvarianten Eingangsgrößen, für unterschiedliche relevante Verläufe der Eingangsgröße durchgeführt. Die Topologie des neuronalen Netzes entspricht den in Kapitel 2.5 vorgestellten PFAs. Die Eingabe des neuronalen Netzes des PFAs entspricht den Messgrößen und die Ausgabe des Netzes entspricht den aus den numerischen Simulationen ermittelten Zustandsvariablen. Im Gegensatz zum klassischen Luenberger-Beobachter mit Prozessmodell, Messmodell und Rückführung der Messdifferenzen wird damit eine geeignete Transformationsfunktion erlernt, die die Messwerte auf den Zustand abbilden. Dadurch entsteht ein inverses Messmodell mit dem während des online Prozessab-

Abbildung 3.7: Zustandsvorhersage durch Observablen mittels Regression.

laufs anhand der Messwerte der Zustand vorhergesagt werden kann. Durch die PFAs wird somit die funktionale Abhängigkeit zwischen Messgrößen und Zustandsvariablen für einen konkreten Parameterbereich offline erlernt und kann dann als Black-Box-Modell zur online Zustandsbeobachtung eingesetzt werden.

Diese Methode zur Erstellung eines statischen Black-Box-Beobachters eignet sich vor allem dann, wenn ein analytisches Modell nicht vorhanden oder mit erheblichen Rechenaufwand verbunden ist und während des Ablaufes des Prozess sichergestellt ist, dass Messungen vorliegen. Für hochdimensionale Zustandsbeschreibungen extrahiert der PFA Zustandsmerkmale, die eine wesentlich geringere Dimension haben als der Zustand. Hierfür stellt Senn [87] zusätzlich einen Zustandsregler vor, der den Prozess auf Basis der Zustandsmerkmale regelt.

Dieser Beobachter wird bei Senn aus Simulationsdaten eines dreidimensionalen Tiefziehprozesses generiert. Abbildung 3.8 zeigt das Zusammenspiel von Prozess, Beobachter und Regler. Die Ergebnisse zeigen, dass sich dieser Beobachter mit seinen extrahierten niedrigdimensionalen Zustandsmerkmalen für die Regelung komplexer Prozesse eignet. Somit kann der Zustand komplexer Prozesse, deren Prozessmodell nicht vorliegt oder nur in einem hochdimensiona-

len Raum (z. B. in Form von numerischen Simulationen) online verfolgt werden. Nachteilig an dieser Methode ist, dass sie stark auf das Vorhandensein von Messgrößen und geringen Latenzen bei den Messungen angewiesen ist sowie für jede Beobachtungsfolge von c Beobachtungen c Schätzer trainiert werden, die der Beobachtungsfolge den aktuellen Zustand zuweisen: $\hat{\boldsymbol{x}}(t_c) = f_c(\{\boldsymbol{o}(t_1), \boldsymbol{o}(t_2), \ldots \boldsymbol{o}(t_c)\})$.

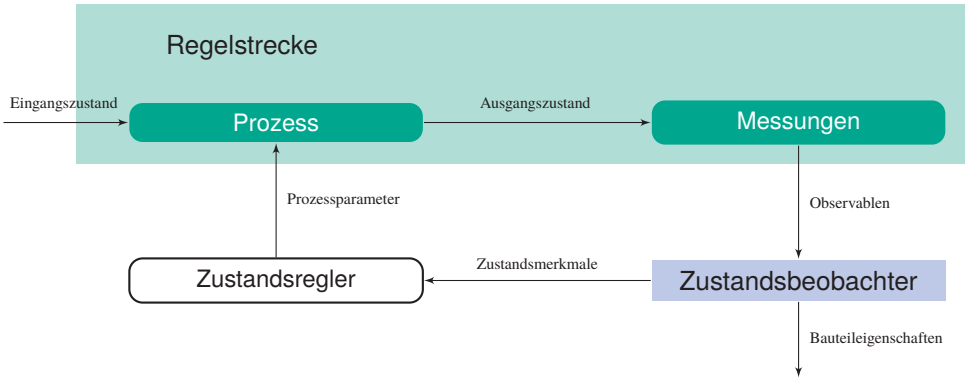


Abbildung 3.8: Aufbau statischer Black-Box-Beobachter und optimaler Regler nach Senn [87].

Beispiel: Dynamischer Black-Box-Beobachter Die Erstellung eines dynamischen Black-Box-Beobachters steht in enger Verbindung mit der Identifikation eines dynamischen Systems mittels experimenteller Modellbildung. Rekurrente neuronale Netze ermöglichen es, dynamische Systeme nachzubilden, jedoch sind rekurrente Lernverfahren sehr aufwendig und deren Stabilität allgemein nicht nachweisbar. Eine Alternative bietet z. B. das Nonlinear Autoregressive Exogenous Model (NARX), welches mit statischer Konfiguration trainiert wird, jedoch als dynamisches Modell verwendet werden kann [68]. Es ist ein feedforward neuronales Netz, welches eine verdeckte Schicht besitzt und, im Falle eines Prozessmodells, die Eingangsgrößen sowie den vorherigen Zustand auf den aktuellen Zustand abbildet. Dabei ist in der Regel der vorherige Zustand zum Zeitpunkt $t-1$ durch die Daten gegeben und kann dadurch zum Training verwendet werde. Die MATLAB® Implementierung des NARX [65] ermöglicht es, das statisch trainierte Netz bei der Anwendung als dynamisches Netz zu verwenden. Dabei wird das Netz mit den Modellausgangswerten zum Zeitpunkt $t-1$ als Eingabewerte zum Zeitpunkt t ausgeführt. Abbildung 3.9 zeigt die Rückführung der Ausgangswerte zum Eingang des Netzes.

3.5 Zusammenfassung

Dieses Kapitel gab eine kurze Einführung in lineare Zustandsbeobachter, um aufbauend darauf einen Blick auf die nichtlinearen Zustandsschätzer, erweitertes und unscented Kalman-Filter, zu werfen. Weiterhin zeigte dieses Kapitel die Verwendung eines Black-Box-Modells zur Zustandsbeobachtung. Kann auf die Interpretierbarkeit verzichtet werden, so können Modellbildung und Zustandsbeobachtung mittels rekurrenter neuronaler Netze und dem daraus entstehenden Black-Box-Modell in Betracht gezogen werden. Da diese Arbeit zum Ziel hat, einen Zustandstracker zu entwickeln, der nicht nur auf einen dynamischem Modell beruht, sondern auch ein mög-

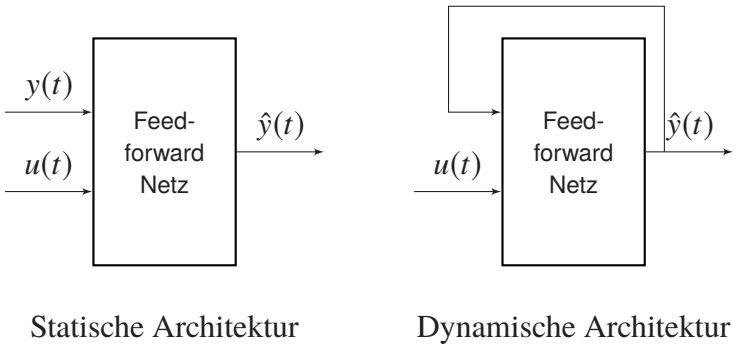


Abbildung 3.9: Statische und dynamische Konfiguration des NARX.

lichst interpretierbares Modell besitzt, wird für die Zustandsbeobachtung auf EKF und UKF zurückgegriffen.

4 Lösungsansatz und Architektur

Ausgehend von Zielsetzung und Anforderungen in Kapitel 1 und den ermittelten Grundlagen in Kapitel 2 und 3 gliedert sich die Architektur des Gesamtsystem in eine offline und in eine online Komponente. In der offline Komponente werden alle zeitaufwendigen Berechnungen durchgeführt. Darunter fallen:

- Durchführen von Simulationen zur Erstellung einer Datenbasis bestehend aus zeitlichen Verläufen von Zustandsvariablen und Observablen für unterschiedliche Prozessparameter.
- Reduzierung der Zustandsvariablen durch Merkmalsextraktion zur Ermittlung eines Unterraums, in dem der Prozesszustand mit einer geringeren Anzahl an Variablen beschrieben wird.
- Ermittlung eines Prozess- und Messmodells (zur Systemidentifikation), das die Änderung des Zustands im reduzierten Zustandsraum sowie die Abhängigkeit zwischen reduzierten Zustandsvariablen und Observablen beschreibt.

Die Datenbasis wird mit der in Kapitel 2.1 aufgeführten Herangehensweise aus den numerischen Simulationen erzeugt. Zur Erstellung der numerischen Simulation wird die Finite-Elemente-Software Abaqus [89] verwendet. Die hier betrachteten Simulationen, wie die Simulation des Tiefziehens, bestehen aus Modellen, deren finite Elemente gleichmäßig verteilt sind und während der Simulation nicht neu vernetzt werden. Durch das Extrahieren der Daten in jedem Integrationspunkt der finiten Elemente zu jedem Zeitschritt wird die Datenbasis aufgebaut. Dies wird für unterschiedliche Prozessparameter durchgeführt, sodass eine Vielzahl von Stichproben gesammelt wird. Eine Stichprobe entspricht einem Prozessparameter zu einem Zeitschritt, sodass die Gesamtanzahl bei einem zeit-invarianten Prozessparameter aus der Anzahl Zeitschritte mal der Anzahl unterschiedlicher Parameterwerte und bei einem zeit-varianten Prozessparameter aus der Anzahl Zeitschritte mal der Anzahl unterschiedlicher Werteverläufe entsteht.

Die extrahierten Zustandsvariablen und Observablen zu jedem Simulationschritt und Prozessparameter repräsentieren das A-priori-Prozesswissen. Zusätzlich stehen die Informationen zur Verfügung, die auch zur Erstellung des FE-Modells zur Verfügung standen, wie Materialeigenschaften, Tribologie, Werkzeug- und Bauteilgeometrie. Diese können durch reale Experimente und/oder Expertenwissen gesammelt werden. Da FE-Modelle zu jeden Zeitpunkt in jedem Integrationspunkt die Werte berechnen, ist der Rechenaufwand für online Berechnungen zu hoch, wenn der Zustand mit ausreichender räumlicher Auflösung und zeitlicher Abtastung betrachtet werden soll. Somit muss experimentelle Modellbildung oder eine vorherige Dimensionsreduktion für die Erstellung eines onlinefähigen Modells eingesetzt werden. Da diese Arbeit sich zusätzlich der Herausforderung stellt, eine interpretierbare Lösung in Form von Differenzialgleichungen zu finden, muss eine Dimensionsreduktion stattfinden, um anschließend Grey-Box-Modellierungsverfahren zur Auffindung der Differenzialgleichung auf Grundlage der dimensionsreduzierten Zustände anzuwenden.

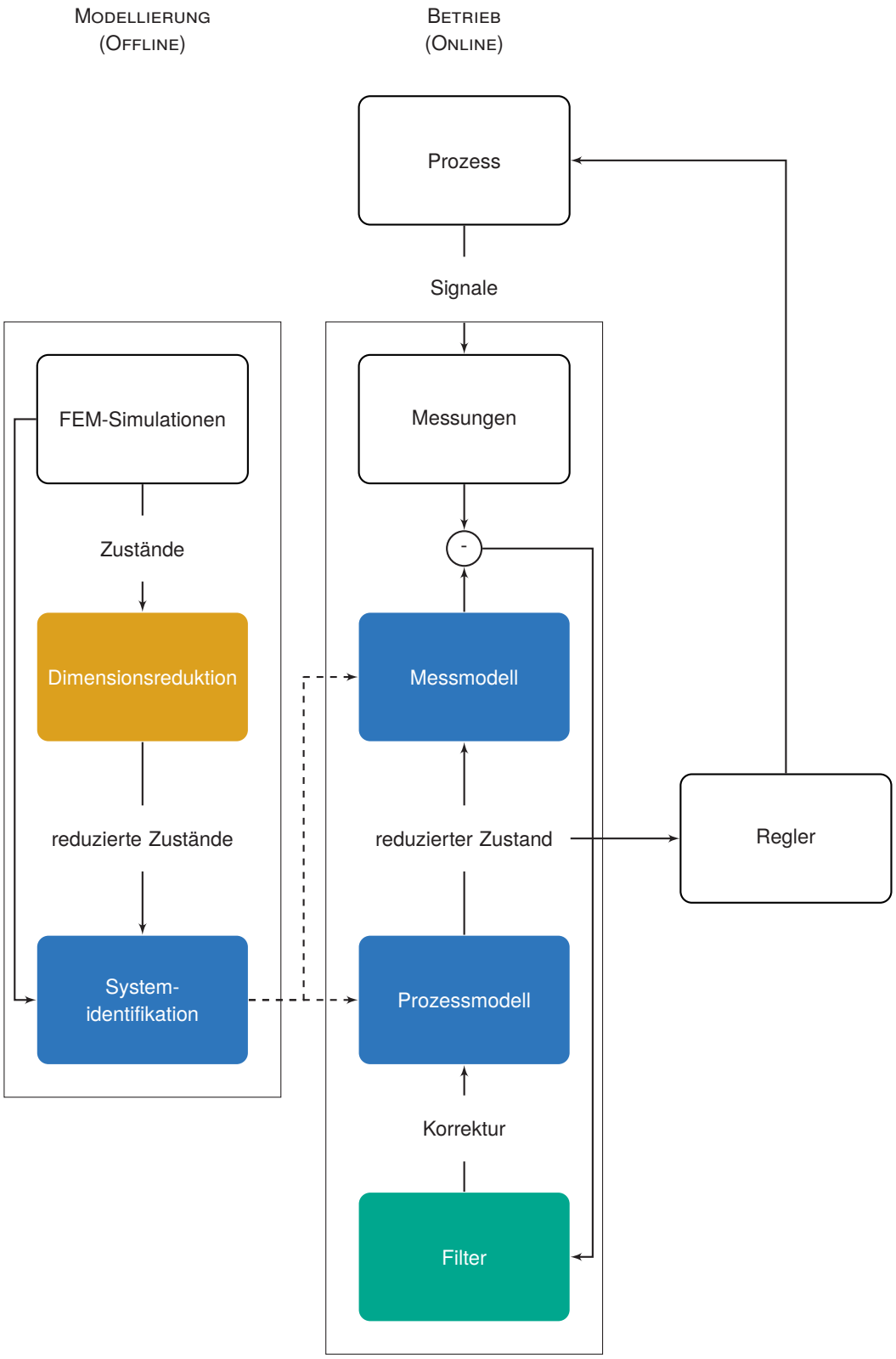


Abbildung 4.1: Systemarchitektur zur Erstellung eines onlinefähigen und datengetriebenen Zustandsbeobachter.

Die Dimensionsreduktion findet folglich in der offline Komponente vor der Modellbildung statt. Das Mengengerüst an Zustandsvariablen aus der Simulation variiert je nach Geometrie und verwendetem Materialmodell. Der in dieser Arbeit betrachtete Tiefziehprozess wird z. B. in einem zweidimensionalen FE-Modell mit 400 Elementen dargestellt, jedoch beschreibt dieses Modell lediglich isotropes Verfestigungsverhalten. Für die Darstellung des gleichen Prozesses bei anisotropem Material wird ein dreidimensionales Modell mit 8832 Elementen vorgestellt, da u. a. die durch Anisotropie verursachte Zipfelbildung nicht im zweidimensionalen Modell darstellbar ist. Das Prozesswissen wird somit durch die vorberechneten zeitlichen Verläufen des 8832-Elemente großen Zustandsvektors dargestellt.

Um nun das Wissen aus den offline berechneten Simulation auch für eine online Simulation zu verwenden, wird ein Dimensionsreduktionsverfahren benötigt, welches die Zustandsvariablen so reduziert, dass die gleiche Prozessdynamik wie in den offline Simulationen erhalten bleibt, allerdings mit weniger Zustandsvariablen bzw. weniger Gleichungen. Die Dimensionsreduktion kann unter anderem durch Entfernen von Redundanz und Korrelationen erfolgen, dabei sollen Informationen zur Nichtlinearität und Dynamik erhalten bleiben. Die Zustandsverläufe im Zustandsraum sollen sich wieder aus den Zustandsverläufen im Merkmalsraum rekonstruieren lassen. Hierfür wird im nachfolgenden Kapitel 5.1 das im Rahmen dieser Arbeit entwickelte Dimensionsreduktionsverfahren vorgestellt, das aus einer Vielzahl von Zustandsvariablen eine wesentlich geringere Anzahl an geordneten Zustandsmerkmalen ermittelt, die Grundlage für die Modellbildung sind und ebenso den ursprünglichen Zustand mit möglichst geringem Fehler rekonstruieren können. Die Ordnung der Merkmale soll durch die erklärte Varianz, die das jeweilige Merkmal abdeckt, erfolgen. Dies ermöglicht es, die Relevanz des jeweiligen Merkmals im Vergleich zu den anderen Merkmalen zu bestimmen.

Ziel der Modellbildung ist es, ein surrogates Modell für die Prozessdynamik der Zustandsvariablen zu ermitteln, welches sich für Regelungseingriffe weitestgehend so verhält wie das FE-simulierte System. Das surrogate Modell des gesamten Systems setzt sich aus Prozessmodell und Messmodell zusammen. Hierfür werden in der offline Komponente die durch die Dimensionsreduktion ermittelten Zustandsmerkmale (reduzierte Zustände) und die Observablen aus der FE-Simulation verwendet. Ein onlinefähiges Prozessmodell setzt eine geringere Rechenkomplexität voraus, welche durch einfache dynamische Gleichungen für die Zustandsmerkmale, die eine wesentliche geringer Dimension haben als die Zustandsvariablen, erzielt wird. Symbolische Regression ermöglicht es, Formelausdrücke aus gegebenen Daten zu erlernen und somit ein Grey-Box-Modell zu erlernen. Dadurch wird beabsichtigt, Gesetzmäßigkeiten für die ermittelten Zustandsmerkmale abzuleiten, die in Beziehung zum FE-simulierten Prozessverhalten stehen.

Abschnitt 5.2.1 beschreibt, wie symbolische Regression dazu genutzt werden kann, Differenzialgleichungen zur Systemidentifikation (Ermittlung eines Prozess- und Messmodells) zu erlernen. Da die Observablen in der Regel eine geringe Anzahl haben, müssen diese hierfür nicht reduziert werden. Es ist zu untersuchen, ob diese Herangehensweise zu einem plausiblen Ergebnis führt und inwieweit die Interpretierbarkeit nach der Reduktion der Zustandsvariablen gegeben ist. Hierfür wird in Kapitel 6 als Beispiel eine Anwendung der Wärmeleitung vorgestellt, für die bereits eine analytische Lösung existiert. Weiterhin muss das ermittelte Prozessmodell auf Beobachtbarkeit und Steuerbarkeit überprüft werden.

Ist ein für geeignet gehaltenes Prozess- und Messmodell gefunden, so kann dieses in der online Komponente eingesetzt werden. Die online Komponente realisiert einen nichtlinearen Zu-

standsbeobachter, welcher die gewonnenen Prozess- und Messmodelle nutzt. Das Ergebnis ist ein Zustandsbeobachter, der online die reduzierten Zustandsmerkmale verfolgt bzw. durch das Filter, anhand der Differenz zwischen vorhergesagten und beobachteten Messwerten, korrigiert.

Die online Komponente beinhaltet die Anwendung eines nichtlinearen Kalman-Filters zur Beobachtung des Zustandes anhand der in der offline Komponenten ermittelten Prozess- und Messmodelle. Zur Evaluierung werden die vorhergesagten Zustandsmerkmale für jeden Zeitpunkt in den ursprünglichen Zustandsraum zurücktransformiert. Der rekonstruierte Zustand kann dann mit dem Simulationsergebnis, das mittels FE-Methode ermittelt wurde, verglichen werden. Ziel dabei ist es, dass trotz des stark reduzierten Zustandsraums die Ergebnisse der online Verfolgung möglichst denen aus der FE-Simulation entsprechen. Die verwendeten nichtlinearen Verfahren zur Zustandsbeobachtung ermöglichen es, auf Prozessrauschen und Ungenauigkeiten von Prozess- und/oder Messmodell zu reagieren.

Abbildung 4.1 zeigt die Systemarchitektur mit den jeweiligen Subkomponenten in der online bzw. offline Komponente. Die farblich hinterlegten Subkomponenten werden nachfolgend in Kapitel 5 näher erläutert und in Kapitel 7 anhand der Anwendungen in Kapitel 6 evaluiert. Zur Evaluierung werden, statt der eigentlichen Messwerte aus dem realen Prozess, die Daten aus der FE-Simulation verwendet, um die Gültigkeit der reduzierten Merkmale bzw. das daraus erstellte dynamische Modell zu bewerten. Der reale Prozess kann dann mittels Zustandsregler geregelt werden. Ein Zustandsregler für reduzierte Zustände wird von Senn [87] vorgestellt und wird hier nicht weiter betrachtet.

5 Online-Zustandstracking mit datengetriebenen Prozessmodellen

Dieses Kapitel beschreibt die in Kapitel 4 bestimmten Komponenten, die zur Erstellung eines online Zustandstrackings mit datengetriebenem Prozessmodell benötigt werden. Abbildung 5.1 zeigt dabei die Reihenfolge, in der die einzelnen Komponenten erstellt werden. Ausgehend von den ermittelten Daten aus den numerischen Simulationen werden die einzelnen Komponenten in die drei Schwerpunkte „Reduktion der Zustandsvariablen“, „Identifikation des reduzierten Prozessmodells“ und „Beobachtung des Zustandes“ aufgeteilt. Abschnitt 5.1 beschreibt das Dimensionsreduktionsverfahren, das speziell für die gewählte Anwendung – die Ermittlung eines dynamischen nichtlinearen reduzierten Prozessmodells aus Simulationsdaten – entwickelt wurde, sowie die für die Dimensionsreduktion verwendete Vorverarbeitung der Daten. Darauf aufbauend beschreibt Abschnitt 5.2 die Identifikation des Prozessmodells sowie des Messmodells aus den reduzierten Daten des Abschnittes 5.1. Das identifizierte Prozess- und Messmodell dient zur Beschreibung des Systems im reduzierten Zustandsraum, der dem *Merkmalsraum* des Zustandes entspricht. Dabei wird gezeigt, wie symbolische Regression dafür eingesetzt werden kann, ein nichtlineares dynamisches Modell im Merkmalsraum zu ermitteln, um somit die nicht messbaren Zustandsgrößen bzw. Zustandsmerkmale vorherzusagen. Das ermittelte Prozess- und Messmodell wird anschließend für die Beobachtung des Zustandes mittels nichtlinearem Kalman-Filter in Abschnitt 5.3 verwendet. Um eine zuverlässige Beobachtung des Zustandes zu gewährleisten, ist ferner die Überprüfung auf Beobachtbarkeit notwendig.

Dieses Kapitel soll somit einen Überblick über die Erstellung eines nichtlinearen Zustandsbeobachters geben, der sich zusammensetzt aus einem aus Daten komplexer numerischer Prozesssimulationen erlernten, reduzierten und dynamischen Prozessmodells und einem ebenfalls erlernten Messmodells zur Bewertung des vorhergesagten Zustandes.

5.1 Reduktion der Zustandsvariablen

Dieser Abschnitt stellt das in dieser Arbeit entwickelte Dimensionsreduktionsverfahren vor, das auf Grundlage der Erkenntnisse aus der NLPCA (Kapitel 2.4.2), PFA sowie SPFA (Kapitel 2.5) ein künstliches neuronales Netz konstruiert, welches Zustandsvariablen auf wenige, nach Wertigkeit geordnete nichtlineare Zustandsmerkmale reduziert. Dabei ist hier die *Wertigkeit* dadurch definiert, dass das Merkmal mit der höchsten Wertigkeit die Richtung der höchsten Varianz in den Daten definiert und dabei den Abstand zwischen Datenpunkt x und Koordinatenwert z des orthogonal projizierten Datenpunkts minimiert: $d(x,z) = \|x - z\|_2$. Im *nichtlinearen* Fall kann die Richtung der maximalen Varianz durch eine Kurve beschrieben werden, sodass aus den k Komponenten ein gekrümmter Unterraum aufgespannt wird. Abbildung 5.2 zeigt beispielhaft die Projektion der Datenpunkte auf die nichtlineare Komponente. Die Koordinatenwerte entspre-

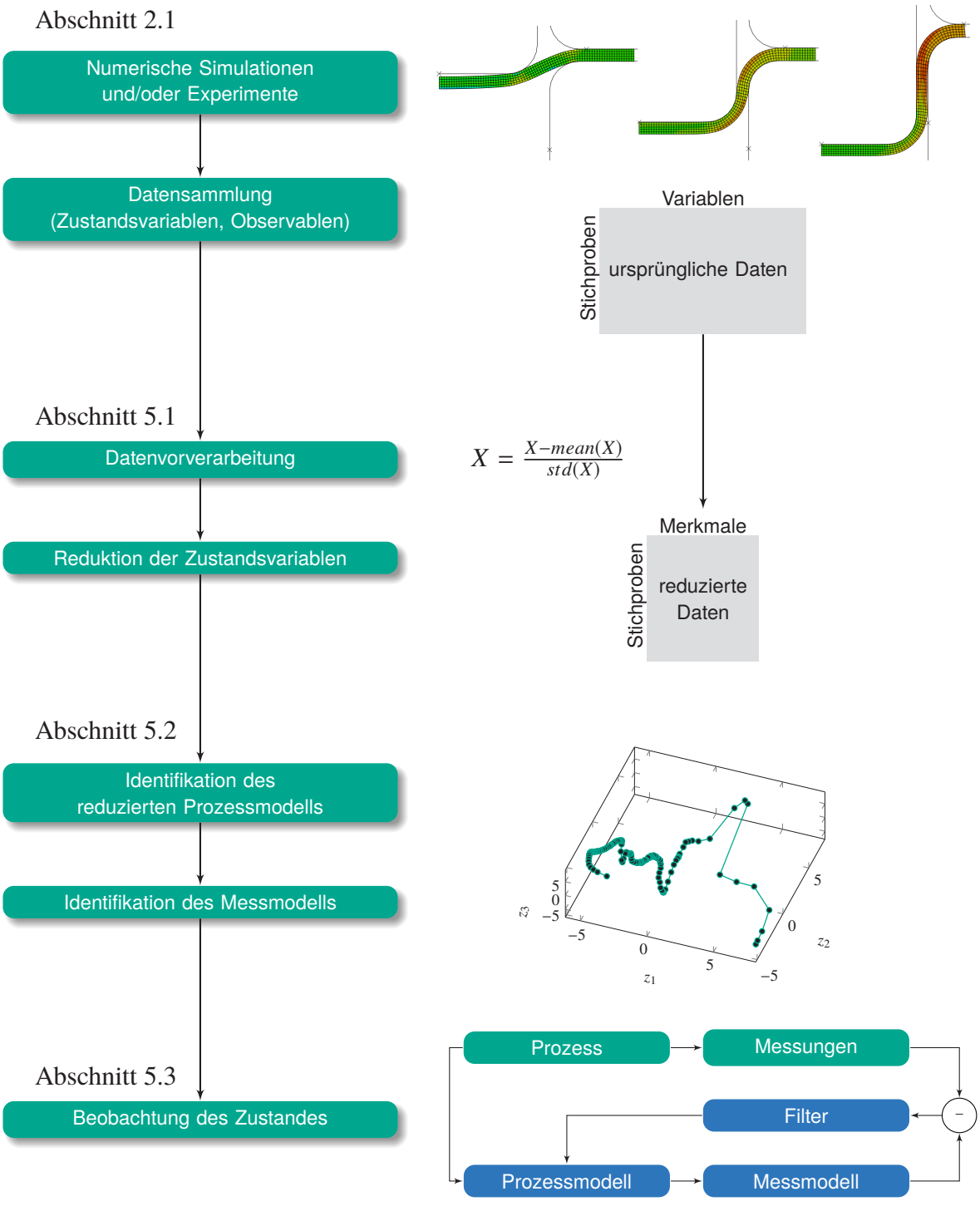


Abbildung 5.1: Kapitelaufbau und Vorgehensweise bei der Erstellung eines nichtlinearen dynamischen Zustandsbeobachters.

chen dem Zustandsmerkmal z. Die jeweilige Projektionslinie zwischen den Datenpunkten $\boldsymbol{x}$, die in diesem Beispiel jeweils durch zwei Zustandsgrößen beschrieben sind, und Zustandsmerkmal z ist durch eine gestrichelte Linie gekennzeichnet.

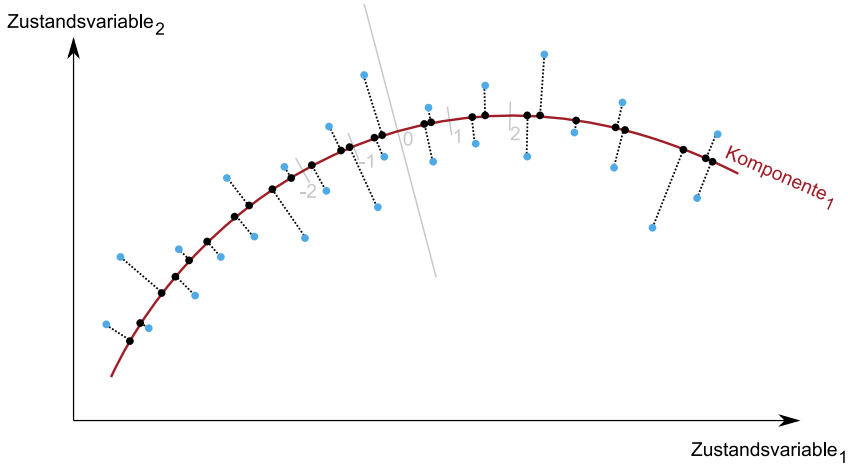


Abbildung 5.2: Exemplarische Reduktion von zwei Zustandsvariablen $\boldsymbol{x} \in \mathbb{R}^2$ (Datenpunkte: blau) auf ein Zustandsmerkmal $z \in \mathbb{R}$ (Koordinatenwerte: schwarz).

Ein weiterer wichtiger Punkt ist die *Unabhängigkeit* der Merkmale voneinander, d. h. die Eigenschaften eines Merkmals sollen unabhängig von den anderen Merkmalen sein und damit auch nicht in anderen Merkmalen wiederzufinden sein. Dies hat den Vorteil, dass die Merkmale einzeln betrachtet und ausgewertet werden können.

Die Anzahl der Merkmale, die nötig sind, um die Zustandsvariablen wieder genügend genau zu rekonstruieren, steht vor der Ermittlung der Merkmale nicht fest. Daher ist zusätzlich ein Verfahren nötig, dass Schritt für Schritt die Merkmale ermittelt und, abhängig von der Restvarianz, die mit der jeweiligen Anzahl Merkmale nicht abgedeckt werden konnte, feststellt, ob weitere Merkmale benötigt werden. Zu wenige Merkmale können zu einer nicht ausreichenden Beschreibung der Zustandsvariablen führen und eine zu hohe Anzahl wiederum zu einer zu komplexen Beschreibung, die die Vorteile der Reduktion – wesentlich geringere Anzahl an Zustandsmerkmalen und damit verbunden geringere Rechenzeit – wieder aufhebt. Folglich soll das Verfahren zusätzlich ermöglichen, während der Ermittlung der Merkmale, falls nötig, dynamisch weitere Merkmale zu ermitteln.

Sowohl zur Bewertung der verbleibenden Varianz als auch zum Zwecke der Zustandsregelung im ursprünglichen Zustandsraum müssen die Zustände aus den Zustandsmerkmalen *rekonstruiert* werden können. Die verbleibende Varianz ist durch den Rekonstruktionsfehler, der Differenz zwischen Zielwerten und vorhergesagten Werten des Zustandes, beschrieben. Somit wird anhand des Rekonstruktionsfehler darüber entschieden, ob die ermittelten Merkmale ausreichend sind.

Die Möglichkeit zur Rekonstruktion des Zustandes sowie die Möglichkeit, eine nichtlineare Dimensionsreduktion durchzuführen, sind bei NLPCA sowie PFA und SPFA durch eine Bottleneck-Schicht mit stark reduzierter Anzahl an Neuronen gegeben. Die Ermittlung unabhängiger und gleichzeitig geordneter Merkmale ist bei der NLPCA durch eine hierarchische Fehlerfunktion und beim SPFA durch sequenzielle Schaltung neuronaler Netze realisiert. Der PFA bietet diese

Möglichkeit nicht, da alle Merkmale gemeinsam trainiert werden und die Merkmale dadurch nicht einzeln betrachtet werden können. Abbildung 5.3 stellt die NLPCA und PFA schematisch dar; dabei ist zu beachten, dass der PFA die Observablen Y als Eingabe für das Netz verwendet und somit nicht nur eine Reduktion durchführt, sondern auch eine Abbildung der Observablen auf die Zustandsvariablen. Dabei werden beim PFA bei der Betrachtung von Zeitreihen die Daten zu jedem diskreten Zeitschritt als eigene Observable erachtet; somit multipliziert sich die Anzahl der Observablen mit der Anzahl vergangener Zeitschritte c, die betrachtet werden soll. Dies hat zusätzlich zur Folge, dass für jede neue Zeitreihe an Observablen, durch Hinzukommen neuer Zeitschritte, das Netz neu trainiert werden muss. Für den SPFA gilt das Gleiche wie für den PFA: Observablen als Eingabe für das Netz und für Zeitreihen erhöht sich die Eingabe um die Observablen vorheriger Zeitschritte; allerdings wird zusätzlich für jede Komponente ein Netz trainiert, statt alle Komponenten in einem Netz. Für die NLPCA gibt es keine festgelegte Betrachtungsweise für die Verwendung von Zeitreihen.

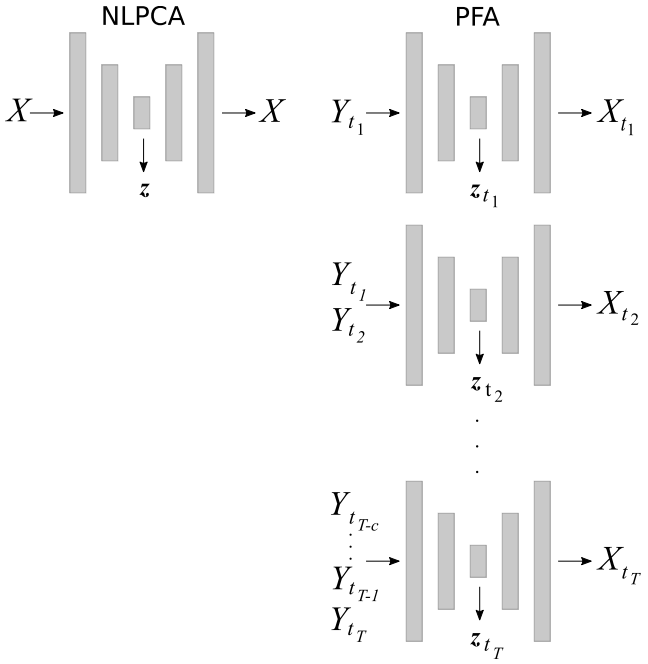


Abbildung 5.3: Vereinfachte Darstellung der Topologien von NLPCA und PFA mit Netzein- und ausgabe.

Nachfolgend werden zwei auf bottleneck neuronalen Netzen basierende Varianten vorgestellt, die die Anforderungen *geordnet*, *unabhängig*, *nichtlinear* und *rekonstruierbar* erfüllen.

5.1.1 Datenvorverarbeitung

Die zeitlichen Verläufe für Zustandsvariablen sowie Observablen werden jeweils in Datenmatrizen angeordnet. Dabei können die einzelnen der insgesamt T Zeitschritte der Zustandsvariablen entweder als einzelne Zustandsvariablen betrachtet werden oder, wie die Werte für die s unterschiedlichen Prozessparameter, als Stichproben. Die Datenmatrix $X \in \mathbb{R}^{s \times n}$, bestehend aus n Zustandsvariablen und s Stichproben, kann somit wie folgt erweitert werden:

- Zeitschritte als Zustandsvariablen: $X \in \mathbb{R}^{s\times(n\cdot T)}$, dies hat zur Folge, dass bei der Dimensionsreduktion die Korrelation zwischen Raum (Zustandsvariablen) und Zeit entfernt wird und die Varianz zwischen den Stichproben (Prozessparametern) bestehen bleibt.
- Zeitschritte als Stichproben: $X \in \mathbb{R}^{(s\cdot T)\times n}$, dabei werden bei der Dimensionsreduktion die Korrelationen zwischen den Zustandsvariablen reduziert und die Varianz über Zeitschritte und Prozessparameter bleibt bestehen.

Da in dieser Arbeit der Schwerpunkt auf der dynamischen Zustandsverfolgung durch ein reduziertes dynamisches Prozessmodell liegt, wird zur Einbringung des zeitlichen Aspekts die zweite Variante zum Aufbau der Datenmatrix gewählt. Das heißt, die Zustandsvariablen aller durchgeführten Simulationen werden durch die Matrix $X \in \mathbb{R}^{(s\cdot T)\times n}$ und die Observablen durch die Matrix $O \in \mathbb{R}^{(s\cdot T)\times r}$ repräsentiert.

Die Datenmatrizen für die Zustandsvariablen X sowie für die Observablen O werden jeweils durch das arithmetische Mittel ($mean(X)$), über alle Variablen, zentriert und durch die Standardabweichung $std(X)$ skaliert:

$$\tilde{X} = \frac{X - mean(X)}{std(X)}. \tag{5.1}$$

Durch die Zentrierung und Skalierung wird auch bei unterschiedlichen Größen verschiedener Wertebereiche gewährleistet, dass sich die Zustandsvariablen bzw. Observablen in vergleichbaren Wertebereich befinden und zur Evaluation der Verfahren der relative Approximationsfehler betrachtet werden kann.

5.1.2 Sequenzielles bottleneck neuronales Netz

Das hier vorgestellte sequenzielle bottleneck neuronale Netz ermöglicht es, durch die gewählte Topologie und Trainingsmethode Zustandsvariablen, beliebig komplexer Prozesse in einen nichtlinearen Unterraum von wesentlich geringerer Dimensionalität zu transformieren. Anders als bei PFA und SPFA wächst die Anzahl an Eingabegrößen für das Netz nicht mit der Anzahl der betrachteten Zeitschritte, sodass beliebig lange Zeitreihen betrachtet werden können.

In [21] beschreiben wir die Herleitung des Dimensionsreduktionsverfahren auf Basis des SPFA. Um eine nichtlineare Dimensionsreduktionsmethode zu erstellen, welche nach Varianz geordnete, vorzugsweise orthogonale Merkmale extrahiert, werden hier mehrere bottleneck neuronale Netze, mit jeweils einem Neuron in der Bottleneck-Schicht, sequenziell geschaltet. Abbildung 5.4 zeigt die grundlegende Architektur mit einem Neuron in der Bottleneck-Schicht jedes Netzes. Der Aktivierungswert der Bottleneck-Schicht entspricht dem jeweiligen Merkmal z_i. Die verdeckte Schicht vor dem Bottleneck-Knoten, die sogenannte Mapping-Schicht, ist für die Abbildung von Eingabewerten (den Zustandsvariablen) auf den Bottleneck-Knoten (das Merkmal z_i) zuständig. Matrix X besteht aus Zustandsvariablen $\boldsymbol{x} \in \mathbb{R}^n$ für alle Stichproben, für die das Netz trainiert werden soll. Das Merkmal z_i berechnet sich also für eine betrachtete Stichprobe wie folgt:

$$z_i = W^{(M)} sig\left(W^{(E)}\boldsymbol{x}\right), \tag{5.2}$$

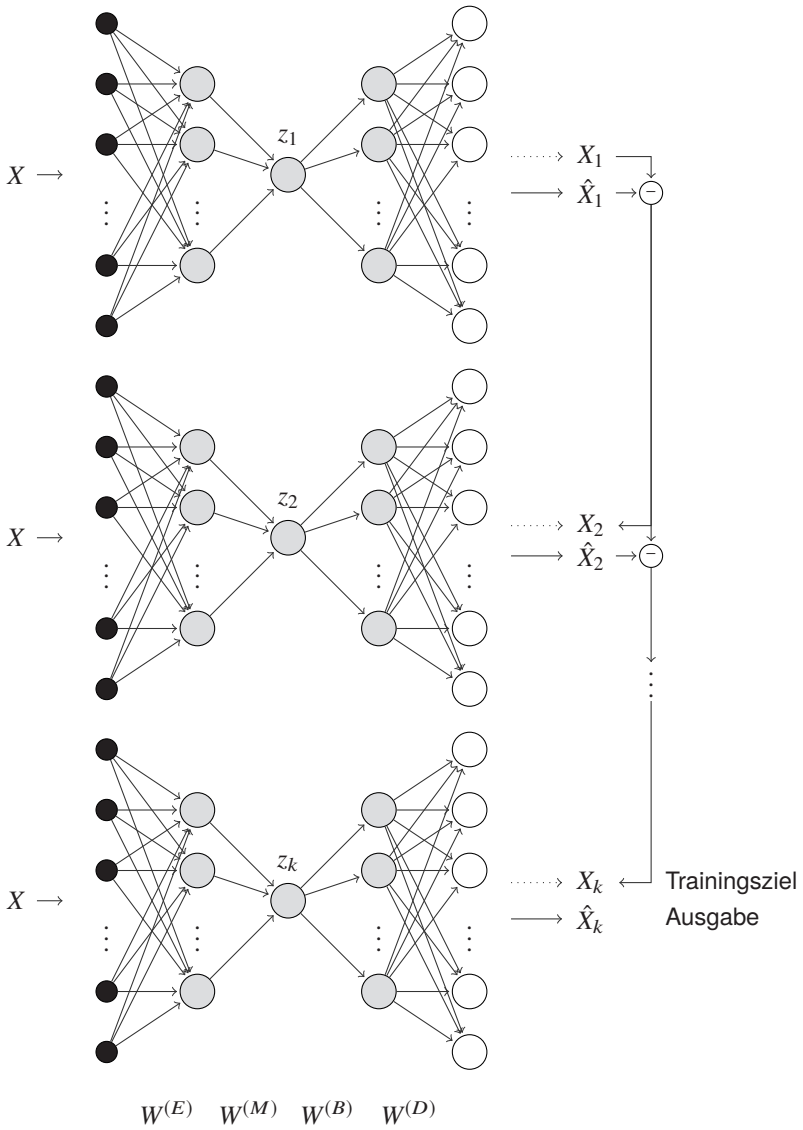


Abbildung 5.4: Topologie des sequenziellen Bottleneck neuronalen Netzes, welches zur Reduktion der Zustandsvariablen für alle Stichproben X verwendet wird.

wobei $W^{(E)}$ die Matrix mit den Gewichten für die Verbindungen zwischen Eingabe- und Mapping-Schicht und $W^{(M)}$ die Matrix für die Gewichte zwischen Mapping- und Bottleneck-Schicht ist. Die Eingabe für jedes Netz in der Architektur bleibt gleich, jedoch werden im Verlauf des Trainings für jedes Netz eigene Gewichte ermittelt.

Ebenso ist die Schicht nach der Bottleneck-Schicht, die sogenannte Demapping-Schicht, für die Abbildung von Merkmal auf die Zielwerte zuständig – sozusagen die Rekonstruktion der Zielwerte aus dem einzelnen Merkmal:

$$\hat{X}_i = W^{(D)} sig\left(W^{(B)} z_i\right), \tag{5.3}$$

wobei $\hat{X}_i$ die rekonstruierten Zielwerte des i-ten Merkmals ist und $W^{(B)}$ und $W^{(D)}$ die Gewichtsmatrizen für die Verbindungen zwischen Bottleneck-Schicht und Demapping-Schicht bzw. zwischen Demapping-Schicht und Ausgabeschicht sind.

Da hier eine nichtlineare Dimensionsreduktion betrachtet wird, wird für die Aktivierungsfunktion *sig* in Mapping- und Demapping-Schicht eine nichtlineare Funktion benötigt. Aufgrund der einfachen Differenzierbarkeit wird in dieser Arbeit der Tangens Hyperbolicus in folgender Form verwendet:

$$sig(x) = \frac{2}{1 + \exp^{(-2 \cdot x)}} - 1. \tag{5.4}$$

Diese Form entspricht der MATLAB® Implementierung der Aktivierungsfunktion *tansig* und ist in der Berechnung schneller als die MATLAB® Implementierung von *tanh*.

Ziel ist es, durch das Training die Gewichte der Verbindungen so anzupassen, dass die Zustandsvariablen in der Weise reduziert werden, dass die reduzierten Werte (Zustandsmerkmale) wiederum die Zustandsvariablen mit einem möglichst geringen mittleren quadratischen Fehler über alle betrachteten Stichproben rekonstruieren:

$$E_{\mathrm{MSE}} = \frac{1}{s} \sum_{i=1}^{s} \left(\hat{\boldsymbol{x}}^{(i)} - \boldsymbol{x}^{(i)}\right)^2 \tag{5.5}$$

mit s Anzahl der Stichproben. Während des Trainings dienen für das erste Netz, und somit das erste Merkmal, die Zustandsvariablen als Zielwerte. Für das zweite Netz bzw. alle weiteren Netze setzen sich die Zielwerte aus dem Residuum, also der Differenz zwischen Zielwerten und Ausgabe des vorangehenden Netzes, zusammen:

$$X_i = X_{i-1} - \hat{X}_{i-1}. \tag{5.6}$$

Die Summe der Ausgaben der einzelnen Netze entspricht den rekonstruierten Zustandsvariablen:

$$\hat{X} = \sum_{i=1}^{k} \hat{X}_i \tag{5.7}$$

mit k gleich der Anzahl an Zustandsmerkmalen bzw. der Anzahl der verwendeten Netze. Durch diese Vorgehensweise soll eine Rangfolge nach Wertigkeit der Merkmale konstruiert werden, sodass das erste Merkmal die Richtung der höchsten Varianz der Daten beschreibt, das zweite

Merkmal wiederum die Richtung der höchsten Varianz der nicht vom ersten Merkmal erfassten Daten, usw.. Dies wird fortgeführt, bis die gewünschte Anzahl an Merkmalen ermittelt wurde oder eine vordefinierte Varianz unterschritten ist. Jedes Netz stellt somit unabhängig von den anderen Netzen andere Bereiche der Daten dar. Die finale Anzahl der Netze entspricht der Anzahl Merkmale, auf die reduziert wurde.

Wie in Kapitel 5.1.1 beschrieben, werden die Zeitreihen der Zustandsvariablen für unterschiedliche Stichproben (unterschiedliche zeit-invariante Prozessparameter) hier so angeordnet, dass alle Zustandsvariablen $\boldsymbol{x}$ zu einem bestimmten Zeitpunkt und einem bestimmten Prozessparameter einer Stichprobe entsprechen:

$$X = \begin{matrix} x_1^{(1,1)} & x_2^{(1,1)} & \dots & x_n^{(1,1)} \\ x_1^{(2,1)} & x_2^{(2,1)} & \dots & x_n^{(2,1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(s,1)} & x_2^{(s,1)} & \dots & x_n^{(s,1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(1,T)} & x_2^{(1,T)} & \dots & x_n^{(1,T)} \\ x_1^{(2,T)} & x_2^{(2,T)} & \dots & x_n^{(2,T)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(s,T)} & x_2^{(s,T)} & \dots & x_n^{(s,T)} \end{matrix} \tag{5.8}$$

mit s Anzahl der unterschiedlichen zeit-invarianten Prozessparameter, n Anzahl der Zustandsvariablen und T Anzahl der Zeitschritte. Eine Zeile entspricht dementsprechend einer Stichprobe und somit einen Eingabevektor für das neuronale Netz.

Diese vorgestellte neuronale Netz-Architektur zur nichtlinearen Dimensionsreduktion mit simultaner Merkmalsextraktion greift die Idee des SPFA (Kapitel 2.5) sowie der von Kramer [54] auf, bottleneck neuronale Netze mit jeweils einem Neuron in der Bottleneck-Schicht sequenziell zu schalten. Im Gegensatz zum SPFA wird hier die zeitliche Abfolge der Zustandsvariablen – zusätzlich zur Variation der Prozessparameter – als Stichprobe verwendet und lediglich die Reduktion und nicht die Abbildung von Messgrößen auf Zustandsvariablen durchgeführt. Weiterhin unterscheidet sich die Konfiguration und das Training des Netzes, auf die in Abschnitt 5.1.4 weiter eingegangen wird. Der Unterschied zu Kramers sequenziellem Autoencoder liegt in der Verwendung der Eingabedaten und der Zielwerte. Während hier die Eingabe der Zustandsvariablen konstant bleiben, werden sie bei Kramer, wie auch die Zielwerte, durch das Residuum ersetzt. Dies führt dazu, dass jedes hintereinander geschaltene Netz ein Autoencoder bleibt, mit identischen Eingabedaten und Zielwerten. Da in dem hier vorgestellten Verfahren jedoch jedes Merkmal eine Reduktion der gesamten Zustandsvariablen sein soll, bleiben die Eingabedaten konstant.

5.1.3 Sequenzielles bottleneck neuronales Netz für Zeitreihen

Die neuronale Netz-Architektur aus 5.1.2 wird nun so erweitert, dass zusätzlich Informationen zum vorherigen Zeitschritt in das zu extrahierende Zustandsmerkmal mit einfließen, um so-

mit die Dynamik des Prozess im Zustandsmerkmal besser abbilden zu können. Üblicherweise werden für die Einbindung vorheriger Zeitschritte bzw. die Abbildung dynamischen Verhaltens rekurrente neuronale Netze verwendet, die sich durch Rückführung des aktuell ermittelten Zustandes auszeichnen (Abbildung 2.4). Die Rückführung kann über eine oder mehrere Schichten hinweg erfolgen. Diese Rückführung bewirkt, dass für die nächste Iteration die Informationen zum vorherigen vorhergesagten Zustand vorliegen. Rekurrente Netze sind jedoch wesentlich aufwendiger zu trainieren und eignen sich somit hauptsächlich dann, wenn keine Informationen zum tatsächlichen vorherigen Zeitschritt vorliegen, z. B. beim online Training eines dynamischen Modells.

Da durch die numerische Simulation Daten zum kompletten zeitlichen Verlauf des Prozesses vorliegen, kann weiterhin ein feedforward-Netz eingesetzt werden. Für dieses Netz werden als Eingabe nicht nur die Zustandsvariablen des aktuellen Zeitschrittes verwendet, welche einer Stichprobe entsprechen, sondern zusätzlich die Zustandsvariablen des vorherigen Zeitschrittes als Eingabe übermittelt. Abbildung 5.5 zeigt die Architektur eines Netzes. Wie in Abschnitt 5.1.2

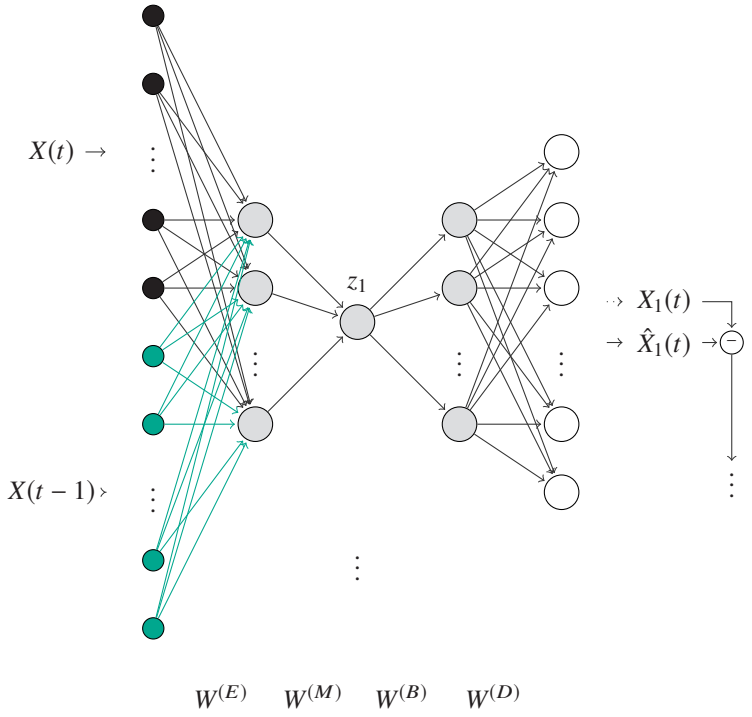


Abbildung 5.5: Topologie des bottleneck neuronalen Netzes für das erste Zustandsmerkmal z_1, mit Zustandsvariablen des aktuellen und vorherigen Zeitschrittes als Eingabe des Netzes.

besteht die komplette Netz-Architektur aus mehreren sequenziell geschaltenen Netzen. Die Eingabe bleibt auch hier konstant und die Zielwerte aller weiteren Netze ergeben sich wie in Abbildung 5.4 aus dem Residuum der Ausgabe und der Zielwerte gemäß Gleichung (5.6). Die Eingabedaten liegen folglich im folgenden Format für alle $t \geq 1$ vor:

$$\begin{bmatrix} X(t) \\ X(t-1) \end{bmatrix}^\top = \begin{bmatrix} \boldsymbol{x}^{(1,t)} & \boldsymbol{x}^{(2,t)} & \dots & \boldsymbol{x}^{(s,t)} \\ \boldsymbol{x}^{(1,t-1)} & \boldsymbol{x}^{(2,t-1)} & \dots & \boldsymbol{x}^{(s,t-1)} \end{bmatrix}^\top, \forall t \geq 1. \tag{5.9}$$

Wie aus der Eingabematrix 5.9 und Abbildung 5.5 hervor geht, enthält das extrahierte Merkmal z_i Information über die Zustandsvariablen zum aktuellen und vorherigen Zeitschritt:

$$z_i = W^{(M)} sig\left(W^{(E)} \begin{bmatrix} \boldsymbol{x}^{(s,t)} \\ \boldsymbol{x}^{(s,t-1)} \end{bmatrix}\right). \tag{5.10}$$

Eine Stichprobe s enthält nun Zustandsvariablen eines Zeitschrittes und des vorherigen Zeitschrittes für einen bestimmten zeit-invarianten Prozessparameter. Das Netz kann nach dem Training für jeden Zustand, für den Informationen zum aktuellen und vorherigen Zeitschritt vorliegen, ein Zustandsmerkmal extrahieren, welches wiederum den aktuellen Zustand rekonstruieren kann.

5.1.4 Konfiguration und Training des Netzes

Dieser Abschnitt umfasst das Ermitteln der Anzahl an Neuronen sowie die Initialisierung, das Training, den Test und die Validierung des Netzes. Diese Bereiche haben großen Einfluss auf die Performanz und Stabilität des Netzes und es gilt, die richtige Kombination aus allen zu finden, um bestmögliche Resultate zu erzielen.

Neuronen in den verdeckten Schichten Die Anzahl Neuronen in den verdeckten Schichten spielt eine wichtige Rolle im Hinblick auf die Stabilität des neuronalen Netzes. Es stellt sich die Frage: Erzielt das neuronale Netz für unbekannte Daten ein vergleichbar gutes Ergebnis wie für die Trainingsdaten? Zu viele Neuronen in den verdeckten Schichten können zur Überanpassung führen, das heißt das Netz erzielt zwar für die Trainingsdaten gute Ergebnisse, verliert aber die Fähigkeit der Generalisierung, d. h. gute Vorhersagen für Testdaten zu erzielen. Zu wenige Neuronen in den verdeckten Schichten können jedoch dazu führen, dass die Komplexität des Systems unzureichend erfasst wird (Unteranpassung). Abbildung 5.6 zeigt beispielhaft den Verlauf der Approximationsfunktion für Trainingsdaten und Testdaten mit jeweiligem Approximationsfehler. Es wird deutlich, dass ein geringer Fehler in den Trainingsdaten nicht zwingend zu einem geringen Fehler in den Testdaten führt.

Es existiert keine allgemeine Lösung, um die optimale Anzahl der benötigten Neuronen zu berechnen. Übliche Vorgehensweisen bestehen aus heuristischen Methoden verbunden mit Versuch und Irrtum. Sheela und Deepa [88] geben eine Übersicht über verbreitete Methoden, die zur Berechnung überwiegend die Anzahl der Neuronen in der Eingabe- und Ausgabeschicht und die Anzahl der Stichproben verwenden. Jedoch sind diese Methoden eher nur ein Richtwert; die optimale Anzahl Neuronen in einer verdeckten Schicht hängt weiterhin von folgenden Eigenschaften ab:

- Anzahl Neuronen in der Eingabe- und Ausgabeschicht
- Anzahl der Stichproben
- Rauschen in den Zieldaten
- Komplexität des Systems

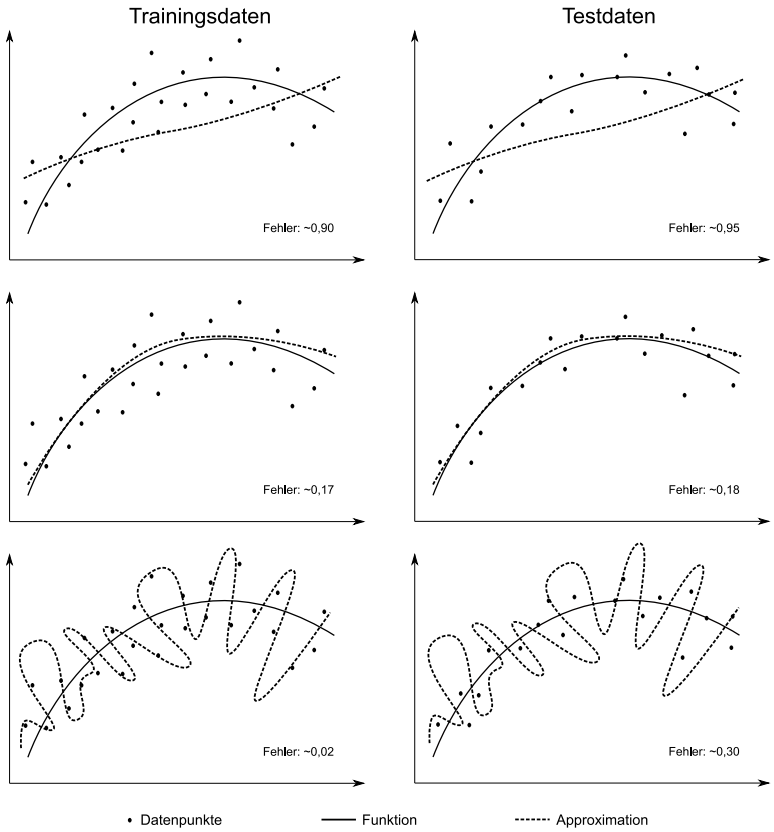


Abbildung 5.6: Unteranpassung (oben) und Überanpassung (unten): Approximation der Trainingsdaten (links) und Approximation der Testdaten (rechts) bei einer beispielhaften Regressionsanwendung.

- Topologie des Netzes
- Aktivierungsfunktionen
- Trainingsalgorithmus
- Regularisierung

Die *Anzahl der Neuronen in der Eingabe- und Ausgabeschicht* ist durch die Anzahl an Zustandsvariablen vorgegeben. Da die Anzahl der Zustandsvariablen von Prozess zu Prozess variieren kann, sollte die Anzahl der Neuronen in der Eingabe- und Ausgabeschicht mit in die Bestimmung der Anzahl der Neuronen in den verdeckten Schichten einfließen.

Eine hohe *Anzahl an Stichproben* ermöglicht es in der Regel, allgemeinere Modelle zu finden, welche für Testdaten eine genauere Approximation finden. Liegt nur eine geringe Anzahl an Stichproben vor und ist eine Erweiterung durch mehr Stichproben nicht möglich, sollte sich die Anzahl an Knoten von Schicht zu Schicht nicht zu stark reduzieren und es sollte in Erwägung gezogen werden, mehr als drei verdeckte Schichten zu verwenden. Da hier die Stichproben durch numerische Simulationen generiert werden, kann Einfluss auf die Anzahl der Strichproben genommen werden. Folglich werden bei nicht ausreichender Anzahl an Stichproben mehr Daten durch zusätzliche Simulationen generiert.

Rauschen in den Zieldaten spielt im Wesentlichen bei online Training eine Rolle, da beim online Training die gemessenen Daten als Zielwerte dienen und unterschiedlich starkem Rauschen unterliegen. In dieser Arbeit werden die aus den numerischen Simulationen ermittelten Zustandsvariablen direkt als Zielwerte gesetzt, ohne ein Rauschen hinzuzufügen.

Je höher die *Komplexität des Systems* ist, desto mehr Neuronen in der Mapping- und Demappingschicht werden benötigt, um diese Komplexität abbilden bzw. reduzieren zu können. Steigt die Komplexität wesentlich über die der Beispielanwendung in Kapitel 6, sollten zwei weitere verdeckte Schichten, jeweils vor und nach der Bottleneck-Schicht, in Betracht gezogen werden.

Methoden, die die *Topologie des Netzes* während des Trainings anpassen, sind z. B. das Pruningverfahren [24] und Cascade Correlation [20]. Das Pruningverfahren entfernt während des Trainings Gewichte, die keinen oder nur sehr geringen Einfluss auf die Verbesserung des Trainingsfehlers haben. Cascade Correlation fügt während des Trainings weitere Neuronen hinzu, bis eine effiziente Netztopologie gefunden ist. Für die Topologie der hier vorgestellten sequenziellen bottleneck neuronalen Netze ist die Anzahl der verdeckten Schichten und die Anzahl der Neuronen in der Bottleneck-Schicht fest vorgegeben. Zu bestimmen ist lediglich die Anzahl der Neuronen in den Mapping- und Demapping-Schichten. Die sequenzielle Schaltung der einzelnen Netze hat keinen Einfluss auf die Anzahl der Neuronen in den Schichten, da die Netze sich nur in ihren Werten für die Zieldaten unterscheiden. Bei der Bestimmung der Anzahl der Neuronen in der Mapping- und Demappingschicht ist somit hauptsächlich zu beachten, dass die Bottleneck-Schicht nur einen Knoten besitzt. Die Reduktion von der Anzahl der Neuronen in der Mapping-Schicht $|M|$ zu einem Merkmal sollte verhältnismäßig nicht stärker sein als die Reduktion von der Anzahl der Eingabeneuronen $|I|$ zur Mapping-Schicht, da die nichtlineare Reduktion durch die Mapping-Schicht realisiert wird. Gleichermaßen sollte die Demapping-Schicht D gleich viele oder weniger Neuronen besitzen als die Mapping-Schicht, in Abhängigkeit vom Verhältnis zwischen der Anzahl der Eingabe- und Ausgabeneuronen:

$$\frac{|I|}{|M|} \geq |M| \geq |D|. \tag{5.11}$$

Regularisierung im Sinne des maschinellen Lernens werden Verfahren genannt, die versuchen, den Lernalgorithmus so zu verändern, dass der Validierungsfehler reduziert wird, ohne Einfluss auf den Trainingsfehler zu nehmen [25]. Sie dienen folglich auch dazu, Überanpassung zu vermeiden. Dabei wird in die Berechnung der Bewertungsfunktion zusätzlich ein Term hinzugefügt, welcher die Gewichte und den Bias verkleinert und somit im Endresultat eine „glatteres“ Ergebnis erzeugt.

Aufbauend darauf, dass hier Regularisierung verwendet wird und die Netzarchitektur aus einem Neuron in der mittleren verdeckten Schicht besteht, wird hier zur Ermittlung der Anzahl Neuronen in der Mapping- und Demappingschicht eine PCA durchgeführt. Die PCA ermittelt hier auf dem gesamten Datensatz, wie viele Merkmale ausreichend sind, um 98% der Varianz der Daten abzudecken. Diese Anzahl an Merkmalen wird für die Anzahl der Neuronen in der Mapping- und Demappingschicht verwendet. Für die Variante der sequenziellen bottleneck neuronalen Netze für Zeitreihen, die doppelt so viele Neuronen in der Eingabeschicht besitzen, werden für die Mappingschicht jeweils doppelt so viele Neuronen verwendet wie für die Demappingschicht. Wird durch diese Ermittlung der Anzahl Neuronen in den verdeckten Schichten Gleichung 5.11

nicht erfüllt, so werden zwei weitere verdeckte Schichten – jeweils eine vor und hinter der Bottleneckschicht – hinzugefügt. Für die hinzugefügten Schichten muss Gleichung 5.11 ebenfalls gültig sein, $|I|$ wird dabei durch die Anzahl Neuronen in der vorherigen Schicht ersetzt.

Trainingsverfahren Für die sequenziellen bottleneck neuronalen Netze wird Resilient Backpropagation [74] verwendet, eine Erweiterung des in Abschnitt 2.3 beschriebenen Lernverfahrens. Resilient Backpropagation wird stabiler, indem es zusätzlich die Gewichtsänderung aus der vorherigen Iteration mit einbezieht.

Für die sequenzielle Schaltung der Netze wird jedes Netz einzeln nacheinander trainiert. Ist ein Netz trainiert, so wird der komplette Trainingsdatensatz X_i durch das i-te Netz geschickt. Die daraus resultierende Ausgabe $\hat{X}_i$ wird zur Ermittlung der Zielwerte für das darauf folgende Netz verwendet: $X_{i+1} = X_i - \hat{X}_i$. Nach dem Training wird die gesamte Netztopologie mit einem unabhängigen Testdatensatz durchlaufen und der Fehler, der durch den mittleren quadratischen Fehler (Gleichung 5.5) zwischen Zielwerten und Ausgabe definiert ist, bewertet.

Initialisierung der Gewichte Gradientenbasierte Trainingsverfahren neigen dazu, in lokalen Optima stecken zu bleiben und verhindern dadurch, gegebenenfalls bessere Merkmale zu erlernen. Das heißt, unterschiedliche initiale Gewichte können zu unterschiedlichen lokalen Minima der Fehlerfunktion im Optimierungsproblem führen. Abbildung 5.7 zeigt dies an einem einfachen Beispiel: Der linke Ausgangspunkt würde zum Gewicht $w = -2$ und der rechte Ausgungspunkt zu $w = 6$ führen. Für $w = 6$ wird jedoch ein geringerer Fehler erzielt und somit ist der rechte Ausgangspunkt zu bevorzugen. Jedoch ist die Fehlerkurve für die jeweiligen Gewichte

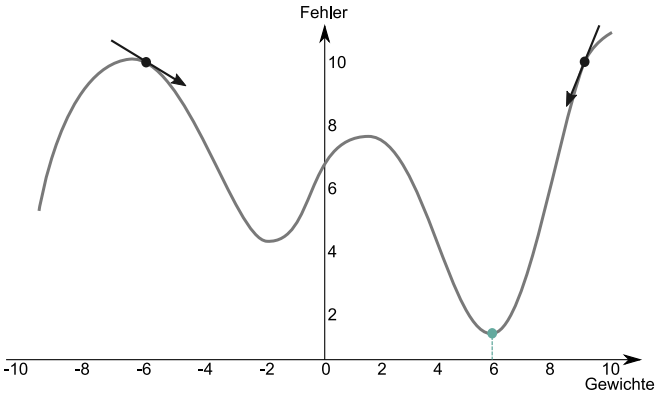


Abbildung 5.7: Gradientenabstieg für unterschiedliche Ausgangspunkte.

nicht bekannt. Um trotzdem einen möglichst optimalen Ausgangspunkt zu gewährleisten, ist die initiale Festlegung der Gewichte von Bedeutung.

Ein im Bereich des Deep Learnings weitverbreitetes Verfahren zur Initialisierung der Gewichte ist das sogenannte Pretraining anstelle von zufällig gewählten initialen Werten. Dabei sollen durch unüberwachtes Lernen der Gewichte beim Training von mindestens zwei aufeinander folgenden Schichten die Gewichte so festgelegt werden, dass sie sich bereits in einem Bereich befinden, der näher an der optimalen Lösung liegt als eine zufällige Initialisierung der Gewichte.

Nach der Initialisierung der Gewichte durch Pretraining müssen durch das überwachte Training des gesamten Netzes die initialen Gewichte nur noch verfeinert werden.

Hinton und Salakhutdinov [33] stellen ein Pretrainingverfahren vor, das jeweils zwei aufeinander folgende Schichten durch unüberwachtes Lernen trainiert. Erhan et al. [19] untersuchen dieses Verfahren im Hinblick auf unterschiedliche Netztiefe, sprich unterschiedliche Anzahl an Schichten, bei Klassifikation. Sie zeigen, dass ohne Pretraining mit steigender Anzahl Schichten die Wahrscheinlichkeit steigt, schlechte lokale Minima zu finden.

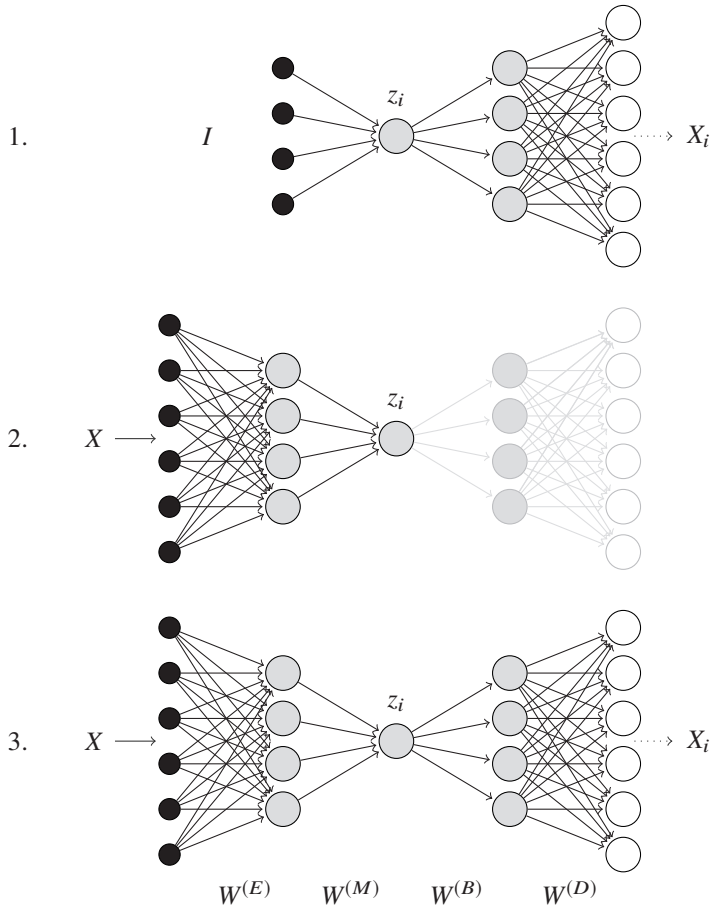


Abbildung 5.8: Pretraining des Netzes in Schritt 1 und 2 mit anschließendem Training des gesamten Netzes in Schritt 3.

Das hier verwendete Pretraining ist in Abbildung 5.8 dargestellt. Es basiert auf dem Pretraining von Scholz [84] für die NLPCA. Es trainiert im ersten Schritt den Bereich des Demappings mit einer Einheitsmatrix der Größe der Anzahl an Stichproben im Trainingset als Eingabe und den für das gesamte Netz vorgegebenen Zielwerten als Werte für die Ausgabenschicht. Die Einheitsmatrix ersetzt somit die noch unbekannte Eingabe in die Bottleneck-Schicht. Dabei wird durch die Struktur der Einheitsmatrix mit jeder Stichprobe ein Gewicht trainiert.

Im zweiten Schritt werden die Gewichte $W^{(M)}$ und $W^{(E)}$ trainiert, mit den Eingabewerten für das gesamte Netz und den im ersten Schritt ermittelten Aktivierungswerten für das Bottleneckneuron

als Zielwerte. Die in den zwei Schritten erlernten Gewichte dienen als initiale Gewichte für das Training des gesamten Netzes.

Kreuzvalidierung Mittels Kreuzvalidierung ist es möglich, die Vorhersageperformanz eines statistischen Modells zu bewerten. Es ist ein etabliertes Verfahren zur Validierung überwachter Lernalgorithmen, bei dem der Datensatz in zwei Teile aufgeteilt wird: Trainingsdaten und Testdaten. Da gute Ergebnisse beim Training des Netzes keine Aussagen über die Performanz für unbekannte Daten treffen, eignet sich der Trainingsfehler nicht zur Bewertung der Netze. Zur Bewertung der Performanz werden unabhängige Testdaten verwendet, die nicht für das Training verwendet wurden. Die Performanz des Netzes wird dann anhand des *Generalisierungsfehlers*, der beim Durchlaufen der Testdaten erzeugt wird, beurteilt. Bei der Kreuzvalidierung, im Speziellen, werden die Stichproben in k disjunkte Untermengen unterteilt. Eine Menge davon wird für das Testen benutzt, alle anderen zusammen zum Trainieren des Netzes. Das Training wird k-mal ausgeführt und jeweils eine andere Untermenge für das Testen und die restlichen Untermengen zum Training verwendet. Durch das mehrfache Ausführen des Trainings und Testens wird es auch k-fache Kreuzvalidierung genannt.

Der Datensatz kann auch in drei Teile unterteilt werden: Trainingsdaten, Validierungsdaten und Testdaten. Die Trainingsdaten werden weiterhin zum Trainieren des Netzes eingesetzt, sprich zur Minimierung des Fehlers zwischen Trainingsausgaben und Zielwerten. Die Validierungsdaten werden zur Bewertung der durch das Training erzeugten Hypothese verwendet. Für die ersten Trainingsdurchläufe sinkt neben dem Trainingfehler auch der Validierungsfehler. Verkleinert sich der Validierungsfehler während mehrerer Trainingsiterationen nicht mehr, kann das Training beendet werden (*early stopping*) und die Gewichte und Bias des Netzes mit dem geringsten Validierungsfehler werden als Ergebnis verwendet. Weiteres Training des Netzes würde nur zur Überanpassung der Daten führen. Die Generalisierung bzw. die Vorhersageperformanz kann nun mittels der Testdaten bewertet werden. Durch die Kreuzvalidierung wird somit auch sicher gestellt, dass für unbekannte Daten im Wertebereich, der Zustand mit vertretbarem Fehler rekonstruiert werden kann.

Da in dieser Arbeit die hintereinander geschalteten Netze unabhängig voneinander trainiert werden, wird die durch die k-fache Kreuzvalidierung ermittelte Trainingsmenge nochmals unterteilt. Dabei werden 65% der Trainingsmenge für das Training verwendet, 30% für die Validierung und 5% für das Testen. Die gesamte Netztopologie, bestehend aus den hintereinander geschalteten Netzen, wird dann mittels der durch die k-fache Kreuzvalidierung ermittelten Testmenge bewertet. Das Ergebnis ist die Netztopologie, die den geringsten Generalisierungsfehler erreicht hat.

Mit diesem hier eingeführten nichtlinearen Dimensionsreduktionsverfahren werden charakteristische Merkmale aus Simulationsdaten extrahiert, durch die die Zustände reduziert beschrieben werden können. Im nachfolgenden Abschnitt 5.2 werden diese reduzierten Zustandsbeschreibungen dazu genutzt, ein reduziertes, dynamisches, nichtlineares Prozessmodell zu identifizieren.

5.2 Identifikation des reduzierten Prozessmodells

Aufgabe der Identifikation des reduzierten Prozessmodells ist die Ermittlung der Abhängigkeiten zwischen Systemeingangsgrößen $\boldsymbol{u}$ und reduzierten Zustandsgrößen $\boldsymbol{z}$ durch die Modellierung eines dynamischen Modells anhand von Prozessdaten. In dieser Arbeit werden Prozesse betrachtet, für die zwar numerische Modelle bestehen, die aber zu rechenaufwendig für die online Zustandsverfolgung sind. Die hier vorgestellte Methode soll aus dem modellierten und berechneten Wissen aus den Finite-Elemente-Modellen, die durch die Methoden in Kapitel 5.1 reduziert wurden, ein reduziertes, nichtlineares, dynamisches Prozessmodell ermitteln.

Zur Identifikation des gesamten Systems, sozusagen der Ermittlung der quantitativen Abhängigkeit zwischen Aus- und Eingangsgrößen eines Systems, wird zusätzlich ein Messmodell benötigt. Die Identifikation des Messmodells umfasst die Ermittlung der Ausgangsgrößen in Abhängigkeit der reduzierten Zustandsvariablen. Beide, Prozess- und Messmodell, sind für die Beobachtung des Prozesses (Kapitel 5.3) von Bedeutung.

5.2.1 Prozessmodell

Verschiedene freie Software ist verfügbar, um symbolische Regression auf verschiedensten Datensätzen durchzuführen. Nennenswert sind dabei GPTIPS [85] (eine MATLAB® Toolbox), Eureqa® [69, 80] und HeuristicLab [51]. Als mächtigste und in den Ergebnissen zuverlässigste erwies sich Eureqa®, insbesondere im Hinblick auf das Erlernen von Differenzialgleichungen.

Abschnitt 2.6 beschreibt das Grundkonzept der symbolischen Regression – wie mittels genetischer Algorithmen Modellparameter und Modellstruktur so angepasst werden, dass sie zugrundeliegende Daten möglichst genau repräsentieren. Die Daten zur Ermittlung des reduzierten Prozessmodells sind die in Kapitel 5.1 ermittelten Zustandsmerkmale $\boldsymbol{z}$ zu jedem untersuchten Zeitpunkt t für jeden definierten Prozessparameter u.

Eine bestehende Herausforderung bei der symbolischen Regression ist es, Differenzialgleichungen zu erlernen, wenn keine konkreten Daten zur Zustandsänderung vorliegen. Schmidt und Lipson [80] zeigen am Beispiel eines Doppelpendels, dass es möglich ist, dass symbolische Regression allein aus Daten und ohne Vorwissen über physikalische oder geometrische Zusammenhänge, Gesetzmäßigkeiten zur Dynamik des Systems findet. Dabei erlernen sie Differenzialgleichungen anhand von experimentell erfassten Daten eines Doppelpendels. Sie schließen daraus, dass ihre Implementierung der symbolischen Regression in der Lage ist, bei gegebenen Daten dazugehörige Gesetzmäßigkeiten zu finden; liegen z. B. Daten zur Beschleunigung vor, können Bewegungsgleichungen gefunden werden. Diese Herangehensweise erfordert jedoch bestehendes Wissen über die Zustandsänderung. In der vorliegenden Arbeit stehen nur die Zustandsmerkmale zu diskreten Zeitpunkten zur Verfügung und es existiert kein Wissen darüber, inwiefern sie zu welchen möglichen physikalischen Größen in Beziehung stehen können. Dies erschwert das Auffinden bzw. das Erlernen der Differenzialgleichungen.

Die Vorgehensweise in dieser Arbeit: Zu ermitteln ist das reduzierte Prozessmodell $\dot{\boldsymbol{z}}(t) = f(u, \boldsymbol{z}(t))$ mit zeitinvariantem Prozessparameter u. Die Differenzialgleichungen können mittels

symbolischer Regression durch unterschiedliche Herangehensweise erzeugt werden. Wie in Kapitel 2.6 beschrieben, können zur Ermittlung von Abbildungen durch symbolische Regression, Operationen sowie Struktur der gesuchten Funktion vorgegeben werden. Da unterschiedliche Strukturvorgaben zu unterschiedlichen Ergebnissen führen können, sind auch einfache Umstellungen der Gleichungen als vorgegebene Struktur zu betrachten. Folgende Möglichkeiten wurden hier für die Struktur herausgearbeitet, wobei zu beachten ist, dass nicht zwingend äquidistante Zeitschritte vorliegen:

1. Das symbolische Regressions-Tool Eureqa® [80] gibt das Strukturelement $D(y, x)$ zum Erlernen von Differenzialgleichungen vor. Es berechnet die Ableitung von y in Bezug auf x als Funktion von $f(x, y)$. Für ein Zustandsmerkmal z_i und die gegebenen Daten gilt dann:
$$D(z_i, t) = f^{(E)}(t, u, \boldsymbol{z}(t)), \tag{5.12}$$
dabei entspricht $\boldsymbol{z}(t)$ allen verwendeten Zustandsmerkmalen $[z_1, z_2, \ldots, z_k]$.

2. Zentraler Differenzenquotient:
$$\frac{z_i(t_{j+1}) - z_i(t_{j-1})}{\Delta t} = f^{(Z)}(\Delta t, u, \boldsymbol{z}(t_j)), \tag{5.13}$$
mit $\Delta t = t_{j+1} - t_{j-1}$.

3. Vorwärtsdifferenzenquotient:
$$\frac{z_i(t_{j+1}) - z_i(t_j)}{\Delta t} = f^{(V)}(\Delta t, u, \boldsymbol{z}(t_j)), \tag{5.14}$$
mit $\Delta t = t_{j+1} - t_j$.

4. Umformung des Vorwärtsdifferenzenquotienten zur direkten Ermittlung der nachfolgenden Zustandsmerkmale $\boldsymbol{z}(t_{j+1})$:
$$z_i(t_{j+1}) = z_i(t_j) + \Delta t \cdot f^{(N)}(u, \boldsymbol{z}(t_j)), \tag{5.15}$$
mit $\Delta t = t_{j+1} - t_j$.

5. Differenz zwischen nachfolgendem und vorherigem Zeitschritt:
$$z_i(t_{j+1}) - z_i(t_{j-1}) = f^{(D)}(\Delta t, u, \boldsymbol{z}(t_j)), \tag{5.16}$$
mit $\Delta t = t_{j+1} - t_{j-1}$.

Für Gleichung (5.13), (5.14) sowie (5.16) wird die linke Seite der Gleichung jeweils vor dem Erlernen von f vorberechnet. Werden zeitvariante Prozessparameter betrachtet, so kann u durch $u(t)$ ersetzt werden. Die symbolischen Abbildungen f werden offline erlernt.

Konfiguration der symbolischen Regression: Wie auch beim Erlernen der reduzierten Zustandsvariablen werden hier die Daten in ein Trainings- und ein Testdatenset aufgeteilt. 75% der Daten sind Trainingsdaten und 25% Testdaten, dabei erfolgt die Zuordnung der Daten zum

jeweiligen Datenset zufällig. Die Optimierung des Prozessmodells erfolgt durch die Trainingsdaten und die Bewertung des gefundenen Prozessmodells durch die Testdaten. Die Struktur der Zielfunktion wird vor dem Durchführen der symbolischen Regression festgelegt. Dabei werden, abhängig von der Anwendung, benötigte Operationen, wie Addition, Division, Sinus, usw., ausgewählt und mit einer gewünschten Komplexität belegt. Als Abbruchkriterium wird ein Bestimmtheitsmaß von $R^2 = 0{,}95$ erwartet (siehe Approximationsgüte in Abschnitt 7). Wird diese absehbar nicht erreicht, wird das Verfahren manuell abgebrochen.

Nachteile: Zu beachten ist, dass jeweilige vorherige oder nachfolgende Zustandsmerkmale konkrete Werte aus den reduzierten Zustandsvariablen sind. Es liegen somit keine Daten zur Zustandsänderung vor. Die zeitliche Änderung der Zustands wird daher durch z. B. Differenzenqotienten nur approximiert. Die durch die symbolische Regression ermittelten Differenzialgleichungen können dadurch instabil werden, da durch online Anwendung der dynamischen Gleichungen nicht mehr die diskreten konkreten Zustandsmerkalswerte, die zur Erstellung der Gleichung gedient haben, verwendet werden, sondern die jeweils durch die Differenzialgleichungen berechneten Zustandsmerkmale. Es sollte somit auf einen möglichst geringen Fehler beim Erlernen der Differenzialgleichung geachtet werden, da kleine Differenzen zwischen gegebenem Zustandsmerkmal und berechnetem Zustandsmerkmal im Verlauf der Zeit zu unregulierbar großem Fehler führen können. Eine Alternative sind rekurrente neuronale Netze: Sie ermöglichen zwar keine interpretierbare Lösung, aber arbeiten schon beim Erlernen des dynamischen Systems mit den vorherigen ermittelten Werten und nicht mit den konkreten Werten aus dem Datensatz. Jedoch ist auch bei rekurrenten Netzen nicht garantiert, dass eine stabile Approximation gefunden werden kann. Sie kommen aktuell eher zur Klassifikation und weniger zur Funktionsapproximation zum Einsatz.

5.2.2 Messmodell

Das Messmodell soll anhand der vom Prozessmodell vorhergesagten Zustandsmerkmale die Observablen vorhersagen. Ziel des Messmodells ist es daher, die Schnittstelle zwischen nicht messbarem Zustand und Prozessobservablen zu bilden. Dafür wird offline ein Messmodell trainiert. Hierfür kann wieder symbolische Regression oder, für den Fall dass keine Interpretierbarkeit verlangt wird, ein künstliches neuronales Netz verwendet werden.

Künstliches neuronales Netz Senn und Link [86] zeigen, dass ein dreischichtiges neuronales Netz ausreicht, um Zustandsvariablen aus Observerablen mit vernachlässigbarem Fehler vorherzusagen. Es kann angenommen werden, dass, sofern die Umkehrfunktion existiert, vergleichbar gute Ergebnisse für die Umkehrfunktion erzielt werden, da auch das Universal Approximation Theorem nach [36] besagt, dass ein Feedforward-Netz mit einer verdeckten Schicht mit einer endlichen Anzahl an Neuronen beliebige kontinuierliche Transformationen nachbilden kann. Beim Training werden die vorhergesagten reduzierten Zustandsvariablen als Eingangsgrößen und die wahren Observablen aus den numerischen Simulationen als Zielwerte verwendet. Folglich ist die Anzahl der Ein- und Ausgangsneuronen mit der Anzahl der Zustandsmerkmale und

Observablen vordefiniert. Um die Anzahl der Neuronen in der einen verdeckten Schicht zu bestimmen, wird folgende Gleichung, basierend auf [6, 87], verwendet:

$$H = \frac{|O|(s-1)}{\beta(|I|+|O|)+1} \tag{5.17}$$

mit $|I|$ Anzahl der Eingangsneuronen, $|O|$ Anzahl der Ausgangsneuronen, s Anzahl der Stichproben und β als Problem-abhängiger anpassbarer Koeffizient. Als Aktivierungsfunktion in der verdeckten Schicht wird die sigmoide Funktion aus Gleichung 5.4 verwendet, um die nichtlineare Beziehung zwischen reduziertem Zustand und Observablen abzubilden. Für kleine Netze, wie diese, welche nur aus drei Schichten bestehen, eignet sich der Levenberg-Marquardt-Backpropagation-Algorithmus [28, 63] als Trainingsmethode. Die Levenberg-Marquardt-Backpropagation zeichnet sich durch ihre Schnelligkeit aus, benötigt jedoch mehr Speicher und eignet sich deswegen nur für kleine Netze. Der Algorithmus verwendet bei der Optimierung des Netzes die zweite Ableitung der Fehlerfunktion statt der ersten Ableitung, da bei der Fehlerfunktion die nichtlinearen Funktionen durch eine Linearisierung ersetzt werden. Näheres dazu ist in Hagan und Menhaj [28] zu finden.

Symbolische Regression Bei der Verwendung von symbolischer Regression zur Ermittlung des Messmodells wird eine Funktion mit folgenden Aufbau gesucht:

$$\boldsymbol{y}_t = g(\boldsymbol{z}_t, t). \tag{5.18}$$

Die Datenmatrix, aus der diese Funktion ermittelt wird, besteht für alle Stichproben aus den Observablen $\boldsymbol{y}_t$ und den Zustandsmerkmalen $\boldsymbol{z}_t$ zum jeweiligen Zeitpunkt t, für den die konkreten Werte auch vorliegen. Die Strukturelemente können abhängig von der Anwendung gewählt werden.

Es ist anhand der Anwendungen in Kapitel 6 zu bewerten, welches der beiden Verfahren für das Messmodell eher geeignet ist. Die Daten zum Erlernen des Messmodells werden in Trainings- und Testdaten aufgeteilt. Die Optimierung des Messmodells erfolgt durch die Trainingsdaten und die Bewertung des gefundenen Messmodells durch die Testdaten.

5.3 Beobachtung des Zustandes

Der Zustandsbeobachter schätzt den internen Zustand eines gegebenen realen Systems anhand der gemessenen Eingangs- und Ausgangsgrößen. Im hier vorgestellten online Zustandstracking mit datengetriebenen Prozessmodellen entspricht der interne Zustand den extrahierten Zustandsmerkmalen $\boldsymbol{z}$ aus Abschnitt 5.1, die Eingangsgröße u den Prozessparametern, wie Niederhalterkraft, und die Ausgangsgrößen $\boldsymbol{y}$ den Observablen, wie Kräfte und Verschiebungen. Die hier betrachteten Systeme sind nichtlineare Mehrgrößensysteme. Die konkreten Anwendungsfälle werden in Kapitel 6 beschrieben.

Abbildung 3.6 zeigt die Gegenüberstellung des statischen und dynamischen Beobachters. Es zeigt den wesentlichen Vorteil des dynamischen Beobachters gegenüber dem statischen: die Entkopplung des Beobachters von den Observablen. Dadurch sind die Informationen der Obser-

vablen keine Voraussetzung, um den aktuellen Zustand vorherzusagen. Um einen dynamischen Beobachter zu realisieren, kommt hier ein nichtlineares Kalman-Filter als Zustandsschätzer zur Anwendung. Dafür wird das ermittelte reduzierte Prozessmodell (aus Kapitel 5.2) zur Vorhersage des inneren Zustandes benötigt, sowie das Messmodell, welches die Observablen anhand des inneren Zustandes vorhersagt.

Dieses Kapitel beschreibt die spezielle Konfiguration des Zustandsbeobachters unter Verwendung der beiden Kalman-Filter, die in Kapitel 3 vorgestellt wurden (EKF und UKF). Der Vergleich der Performanz der beiden Filter wird in Kapitel 7.3 beschrieben. Zuerst zeigt dieser Abschnitt die angewendete Überprüfung auf Beobachtbarkeit der betrachteten nichtlinearen Systeme.

5.3.1 Beobachtbarkeit nichtlinearer Systeme

Damit ein Zustandsbeobachter zum Einsatz kommen kann, muss das betrachtete System durch seine Messgrößen beobachtbar sein. Die Frage, die sich somit für die Beobachtbarkeit stellt, ist: Kann der Zustand aus $\boldsymbol{u}(t)$ und $\boldsymbol{y}(t)$ bestimmt werden, wenn das System für einen endlichen Zeitraum beobachtet wird? Lineare Systeme sind durch die Erstellung einer Beobachtbarkeitsmatrix, z. B. mittels Kalman-Kriterium [47], und Überprüfung, ob der Rang der Beobachtbarkeitsmatrix gleich der Anzahl an Zustandsvariablen ist, auf Beobachtbarkeit überprüfbar. Der Nachweis der allgemeinen globalen Beobachtbarkeit bei nichtlinearen System ist nur sehr schwer möglich.

Für nichtlineare Mehrgrößensysteme kann die globale Beobachtbarkeit durch lokale Beobachtbarkeit angenähert nachgewiesen werden [31]. Dabei ist das System lokal beobachtbar, wenn in der Nachbarschaft zu $\boldsymbol{x}_0$ jeder Zustand $\boldsymbol{x} \neq \boldsymbol{x}_0$ von $\boldsymbol{x}_0$ abgeleitet werden kann. Die Herangehensweise zur Bestimmung der lokalen Bobachtbarkeit ist wie folgt:

1. Darstellung der Zustands- und Ausgangsgleichungen:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), u) &&= [f_1(\boldsymbol{x}(t), u), f_2(\boldsymbol{x}(t), u), \ldots, f_n(\boldsymbol{x}(t), u)]^\top \\ \boldsymbol{y}(t) &= \boldsymbol{g}(\boldsymbol{x}(t)) &&= [g_1(\boldsymbol{x}(t)), g_2(\boldsymbol{x}(t)), \ldots, g_r(\boldsymbol{x}(t))]^\top . \end{aligned} \tag{5.19}$$

   Das System ist bei $\boldsymbol{x}_0$ lokal beobachtbar, wenn gilt:

$$\boldsymbol{x}_0 \neq \boldsymbol{x}_1 \Rightarrow \boldsymbol{y}(\boldsymbol{x}_0) \neq \boldsymbol{y}(\boldsymbol{x}_1), \tag{5.20}$$

   d. h., sind die Messungen unterschiedlich, so sind auch die Zustände unterschiedlich.

2. Berechne Lie-Ableitungen:

$$l(x, u) \equiv \begin{bmatrix} L_f^0(g) \\ L_f^1(g) \\ \vdots \\ L_f^{n-1}(g) \end{bmatrix} \tag{5.21}$$

3. Erstelle Jacobi-Matrix der Lie-Ableitungen:

$$dG = \left.\frac{\partial l(x,u)}{\partial x}\right|_{x=x_0} \tag{5.22}$$

4. Ist $rang(dG) = n$, dann ist das System bei $\boldsymbol{x}_0$ lokal beobachtbar.

Es wird somit überprüft, ob für n Zustände auch n linear unabhängige Gleichungen zur Verfügung stehen. Die Lie-Ableitung $L_f^i(g)$ und die Jacobi-Matrix sind in Anhang B.2 näher definiert. Für die Überprüfung des reduzierten Prozessmodells und des Messmodells auf Beobachtbarkeit werden die Zustandsmerkmale $\boldsymbol{z}$ anstelle des Zustandes $\boldsymbol{x}$ betrachtet.

5.3.2 Beobachtung mit EKF und UKF

Der nicht messbare Zustand wird durch die Zustandsmerkmale $\boldsymbol{z}$ repräsentiert. Da es sich beim im Kapitel 5.2 bestimmten Prozessmodell zum Vorhersagen der Zustandsmerkmale anhand des Eingangsparameters u um eine Approximation handelt, unterliegt das Prozessmodell gewissen Modellunsicherheiten. Ebenso gilt dies für das Messmodell, dass die Zustandsmerkmale auf die Observablen abbildet. Zusätzlich kann im realen System Messrauschen vorliegen. Der Zustandsbeobachter soll den Einfluss von Modellunsicherheiten und Rauschen korrigieren. Die Zustandsschätzer EKF und UKF aus Abschnitt 3.2 bzw. 3.3 zur Prozessbeobachtung wurden hierfür in MATLAB® implementiert.

Die Initialisierung des Mittelwerts μ_0 für einen bestimmten Prozessparameter u erfolgt durch die Zustandsmerkmale zum Zeitpunkt t_0:

$$\mu_0 = \boldsymbol{z}(t_0, u). \tag{5.23}$$

Die Anfangskovarianz P_0, sozusagen die Unsicherheit für die Zustandsmerkmale zum Zeitpunkt t_0, wird mit einer Einheitsmatrix der Größe $\mathbb{R}^{k\times k}$ initialisiert:

$$P_0 = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & 1 \end{bmatrix}. \tag{5.24}$$

μ und P werden im Verlauf der Zustandsbeobachtung immer wieder durch das Filter aktualisiert. Sind die Zustandsmerkmale zum Zeitpunkt t_0 nicht bekannt oder sehr unsicher, sollte die Diagonale der Matrix P_0 mit sehr hohen Werten belegt werden, damit der Filter möglichst schnell konvergiert.

Die Kovarianz für das Prozessrauschen wird wie folgt initialisiert:

$$Q = \frac{1}{T}\left((Z - \tilde{Z})(Z - \tilde{Z})^\top\right), \tag{5.25}$$

dabei ist $Z \in \mathbb{R}^{T \times k}$ die zeitliche Abfolge der aus der Dimensionsreduktion ermittelten Zustandsmerkmale für einen konkreten Prozessparameter. $\tilde{Z} \in \mathbb{R}^{T \times k}$ umfasst die zeitliche Abfolge der durch die symbolische Regression ermittelten Zustandsmerkmale. Dabei ist zu beachten, dass hierfür das Filter noch nicht zur Anwendung kommt und somit Q im Wesentlichen den mittleren quadratischen Fehler zwischen den Ergebnissen der Dimensionsreduktion und der symbolischen Regression umfasst. Die Matrix teilt somit dem Filter mit, wie das Prozessmodell von den eigentlichen Zustandsmerkmalen abweicht.

In ähnlicher Weise wird die Kovarianzmatrix für das Messrauschen erzeugt:

$$R = \frac{1}{T}((Y - \tilde{Y})(Y - \tilde{Y})^{\top}) \tag{5.26}$$

mit $Y \in \mathbb{R}^{T \times r}$ und $\tilde{Y} \in \mathbb{R}^{T \times r}$ für die jeweiligen Observablen ermittelt durch die numerische Simulation bzw. durch das aus symbolischer Regression erstellte Messmodell. Da bekannt ist, dass die Observablen aus der numerischen Simulation ohne Rauschen vorliegen, kann für die Messunsicherheit auch eine Einheitsmatrix der Größe $\mathbb{R}^{r \times r}$ gewählt werden.

Der allgemeine Ablauf des Zustandstrackers ist in Implementierung 2 aufgeführt. Nach der Initialisierung der Kovarianzmatrizen und des Mittelwerts, durch **init**, wird für jeden Zeitschritt das verwendete Kalman-Filter **kf** ausgeführt. Dabei werden in jedem Schritt Mittelwert und Kovarianzmatrix zur Schätzung von $\hat{z}(t_i)$ aktualisiert.

Implementierung 2 Ablauf Zustandsbeobachter

1: $[Q, R, P_0, \mu_0] \leftarrow$ **init**$(u, \boldsymbol{t}, f, g)$ ▷ Initialisierung
2: **while** $i \leq T$ **do**
3: $\quad [\mu_i, P_i] \leftarrow$ **kf**$(f, g, \mu_{i-1}, P_{i-1}, t, Q, R, \boldsymbol{y}(t_i))$ ▷ EKF oder UKF ausführen
4: $\quad \hat{z}(t_i) \leftarrow \mu_i$

Implementierung 3 zeigt die Umsetzung des erweiterten Kalman-Filters. Dabei entspricht die Funktion **jac** der Berechnung der Jacobi-Matrix, die zur Linearisierung von Zustandsübergangsfunktion f und Messfunktion g benötigt wird. f und g bestehen dabei aus einer Anzahl an Funktionen entsprechend der Anzahl Zustandsmerkmale bzw. Observablen.

Implementierung 3 Extended Kalman-Filter

Eingabe: Symbolische Zustandsübergangsfunktion f und Messfunktion g, a priori Zustandsschätzung μ und a priori Zustandskovarianz P, Zeit t, Prozess- und Messrauschen Q bzw. R sowie aktuelle Observablen y

Ausgabe: A posteriori Zustandsschätzung $\hat{\mu}$ und a posteriori Zustandskovarianz $\hat{P}$

1: $[d\mu, F] \leftarrow \mathbf{jac}(f,\mu)$ ▷ Berechnung Jacobi-Matrix F
2: $\hat{\mu} \leftarrow d\mu \cdot t + \mu$ ▷ Zustandsvorhersage
3: $[\hat{y}, G] \leftarrow \mathbf{jac}(g, \hat{\mu})$ ▷ Berechnung Jacobi-Matrix G
4: $dP \leftarrow FP + PF^\top + Q$ ▷ Kovarianzänderung
5: $\hat{P} \leftarrow dP \cdot t + P$ ▷ Kovarianzvorhersage
6: $S \leftarrow G\hat{P}G^\top + R$ ▷ Verknüpfung Prozess- und Messmodell
7: $L \leftarrow \hat{P}G^\top S^{-1}$ ▷ Berechnung Kalman-Gain
8: $\hat{\mu} \leftarrow \hat{\mu} + L(y - \hat{y})$ ▷ Aktualisierung Zustandsschätzung
9: $\hat{P} \leftarrow (I - LG)\hat{P}$ ▷ Aktualisierung Zustandskovarianz

Implementierung 4 zeigt die Umsetzung des unscented Kalman-Filters. Zuerst werden die anwendungsabhängigen Parameter α und κ initialisiert, die zur Berechnung der Gewichte und Sigmapunkte benötigt werden. Die Sigmapunkte werden durch **sigmapkt** und die Gewichte durch **gewicht**, wie im Anhang A.2 beschrieben, berechnet. Die Bewertung der einzelnen Sigmapunkte zum Zeitpunkt $t - 1$ erfolgt durch die Zustandsübergangsgleichung f in Zeile 6. Die Zustandsvorhersage (durch Mittelwert) erfolgt dann durch die gewichtete Summe über alle Sigmapunkte (Zeile 7). Die dazugehörige Kovarianz wird dementsprechend in Zeile 8 ermittelt. Ebenso wie die Zustandsvorhersage erfolgt die Vorhersage der Observablen, jedoch mit neu ermittelten Sigmapunkten für den Zeitpunkt t und vorhergesagtem Zustand $\hat{\mu}$. Zur Berechnung der Korrektur-Matrix L wird die Kreuzkovarianz zwischen Zustandsmerkmal und Messungen benötigt (siehe Kapitel 3.3). Die anschließende Korrektur von $\hat{\mu}$ erfolgt gleicherweise wie beim EKF.

Implementierung 4 Unscented Kalman-Filter

Eingabe: Symbolische Zustandübergangsfunktion f und Messfunktion g, a priori Zustandsschätzung μ und a priori Zustandskovarianz P, Anzahl Zustandsmerkmale k, Zeit t, Prozess- und Messrauschen Q bzw. R sowie aktuelle Observablen y

Ausgabe: A posteriori Zustandsschätzung $\hat{\mu}$ und a posteriori Zustandskovarianz $\hat{P}$

1: $[\alpha, \kappa] \leftarrow$ **init**() ▷ Initialisierung
2: $\lambda \leftarrow \alpha^2(n + \kappa) - k$ ▷ Festlegung λ
3: $[w_m, w_c] \leftarrow$ **gewichte**(λ,k) ▷ Berechnung Gewichte für μ und P
4: $\chi_{t-1} \leftarrow$ **sigmapkt**($\mu, \sqrt{(k + \lambda)P}$) ▷ Berechnung Sigmapunkte
5: **for all** i **do** ▷ Für alle Sigmapunkte
6: $\quad \hat{\chi}_t^{(i)} \leftarrow f(\chi_{t-1}^{(i)})$
7: $\quad \hat{\mu} \leftarrow \hat{\mu} + w_m^{(i)} \cdot \hat{\chi}_t^{(i)}$
8: $\quad P \leftarrow P + w_c^{(i)} \cdot (\hat{\chi}_t^{(i)} - \hat{\mu})(\hat{\chi}_t^{(i)} - \hat{\mu})^\top + Q$
9: $\chi_t \leftarrow$ **sigmapkt**($\hat{\mu}, \sqrt{(k + \lambda)P}$) ▷ Berechnung Sigmapunkte
10: **for all** i **do** ▷ Für alle Sigmapunkte
11: $\quad \hat{\chi}_t^{(i)} \leftarrow g(\chi_t^{(i)})$
12: $\quad \hat{y} \leftarrow \hat{y} + w_m^{(i)} \cdot \hat{\chi}_t{}^{(i)}$
13: $\quad S \leftarrow S + w_c^{(i)} \cdot (\hat{\chi}_t^{(i)} - \hat{y})(\hat{\chi}_t^{(i)} - \hat{y})^\top + R$
14: $\hat{P}^{\mu,y} \leftarrow$ **cov**() ▷ Berechnung Kreuzkovarianz (siehe Gl. 3.8)
15: $L \leftarrow \hat{P}^{\mu,y} S^{-1}$ ▷ Berechnung Kalman-Gain
16: $\hat{\mu} \leftarrow \hat{\mu} + L(y - \hat{y})$ ▷ Aktualisierung Zustandsschätzung
17: $\hat{P} \leftarrow \hat{P} - LSL^\top$ ▷ Aktualisierung Zustandskovarianz

6 Anwendung

Um die in Kapitel 5 beschriebene Vorgehensweise zur Erstellung eines onlinefähigen Zustandstrackers zu untersuchen, werden hier zwei dynamische Prozesse vorgestellt, für die jeweils bereits experimentierbare Modelle vorliegen. Eine kurze Einführung in die Prozesse wird durch die Betrachtung folgender Punkte (angelehnt an VDI Richtlinie 3633 [97]) gegeben:

- *Problemstellung* (Beschreibung des zu simulierenden Prozesses)
- *Aufgabe und Ziel* (Zu untersuchendes Verhalten)
- *Simulationsdaten und Simulationsmodell* (Beschreibung des Modells)
- *Simulationsstudie* (Experimente mit dem Simulationsmodell)
- *Ergebnisse* (Auszüge aus den Resultaten der Simulationsstudie)

Zur Durchführung der Simulationsstudie werden die einzelne Bereiche aufgezeigt, in denen mit Prozessparametern experimentiert wird, um anschließend eine Auswertung der Simulationsstudie durchzuführen. Diese aus der Simulationstudie generierten Daten werden im nächsten Kapitel zu Erstellung und Evaluierung des onlinefähigen Prozess- und Messmodells bzw. des Zustandstrackers verwendet. Hierbei ist bei der Wahl des Prozessparameters zu beachten, dass dieser im gültigen und prozess-relevanten Bereich variiert wird, da die vorgestellten Verfahren zur Erstellung des onlinefähigen Zustandstrackers nur für diesen Bereich gültig sind.

Neben der Verwendung der Daten aus der Simulationsstudie zur Ermittlung des reduzierten Prozessmodells ist die Validierung der Ergebnisse des reduzierten Modells mit Hilfe der Ist-Daten aus den Simulationsmodellen ein Ziel. Sprich: Ist es möglich, dass das ermittelte reduzierte Modell ähnlich gute Ergebnisse erzeugen kann wie das zugrundeliegende komplexere System?

Für den Machbarkeitsbeweis wird ein dynamisches System der Wärmeleitung betrachtet, für das bereits ein analytisches Modell der Zustandsdynamik existiert. Für den Prozess des Tiefziehens aus dem Bereich des Graduiertenkollegs werden zwei Modelle unterschiedlicher Komplexität vorgestellt. Der Machbarkeitsbeweis, dass Finite-Elemente-Modelle als Datengrundlage für die Erstellung eines onlinefähigen Zustandstrackers verwendet werden können, wird hier an makroskopischen Zustandsbeschreibungen der von Mises Spannung und plastischen Dehnung während des Tiefziehens untersucht, da der Berechnungsaufwand für eine angemessene Anzahl an Stichproben wesentlich geringer ist als für Zustandsbeschreibungen auf Mikrostrukturgrößen.

6.1 Wärmeleitung

Für die erste Beispielanwendung liegt ein analytisches Modell vor, um bei der späteren Evaluierung zu überprüfen, ob aus den erlernten Differenzialgleichungen des reduzierten Modells, Rückschlüsse auf das Verhalten des analytischen Modells getroffen werden können.

Das gewählte dynamische System der Wärmeleitung mit analytischer Lösung steht durch die Exact Analytical Conduction Toolbox [96] zur Verfügung. Diese Toolbox stellt analytische Lösungen für Wärmeübertragung und -ausbreitung zur Verfügung. Die gegebenen analytischen Gleichungen werden in MATLAB® ausgeführt und Daten für Zustandsvariablen und Messwerte extrahiert. Diese werden in Kapitel 7 zur Erstellung und Evaluierung des reduzierten Prozessmodells sowie der Zustandsverfolgung verwendet. Ein bereits existierendes analytisches Modell wurde als Beispielanwendung mit aufgenommen, um zu untersuchen, ob die in Kapitel 5 beschriebene Methode zur Ermittlung reduzierter Prozessmodelle ein mit dem eigentlichen Modell in Relation stehendes, interpretierbares Modell findet.

Problemstellung Das theoretische Beispiel zur Wärmeleitung besteht aus einer Platte der Breite L, bei der auf einer Seite, an Position $p = 0$, die transiente Temperatur $\tilde{T}$ (d. h. deren Wechselanteil) cosinus-periodisch um eine konstante Temperatur variiert und auf der gegenüberliegenden Seite, an Position $p = L$, die transiente Temperatur gleich null ist und zur äußeren Umgebung hin isoliert ist. Abbildung 6.1 zeigt exemplarisch den Versuchsaufbau.

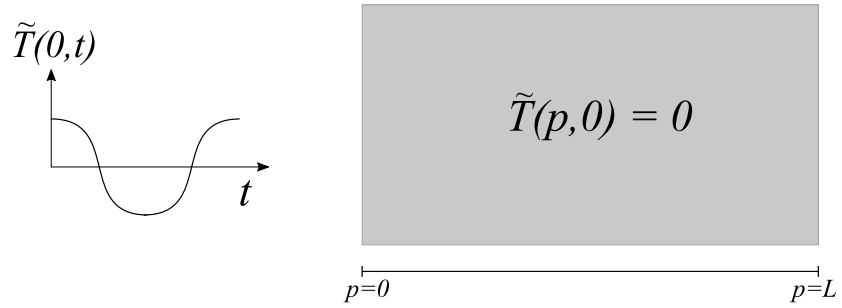


Abbildung 6.1: Versuchsaufbau Wärmeleitung nach [96].

Die Randbedingung für den Aufbau sind wie folgt:

$$\begin{aligned} \frac{\partial^2 \tilde{T}}{\partial p^2} &= \frac{1}{\alpha}\frac{\partial \tilde{T}}{\partial t}; \qquad 0 < p < L \\ \tilde{T}(0,t) &= T_0 \cos(\omega t) \\ \left(\frac{\partial \tilde{T}}{\partial p}\right)_{p=L} &= 0 \\ \tilde{T}(p,0) &= 0 \end{aligned} \tag{6.1}$$

mit Diffusionskoeffizient α und Frequenz ω. Die örtliche Temperaturänderung innerhalb der Platte entspricht der zeitlichen Änderung der Temperatur. Die anliegende Temperatur zum Zeitpunkt t an Position $p = 0$ entspricht der Umgebungstemperatur T_0 mit aufgeschaltetem frequenz-

und zeitabhängigen Cosinus. Die Anfangstemperatur im gesamten Bauteil ist konstant. Unterschiedliche Frequenzen verursachen unterschiedliche Temperaturausbreitungen im Bauteil.

Aufgabe und Ziel Zielsetzung für die Simulation ist es, für unterschiedliche Frequenzen den Temperaturverlauf im Bauteil zu ermitteln. Es ist somit zu untersuchen, welche Temperatur zu welchem Zeitpunkt an beliebiger Position im Bauteil vorherrscht. Dabei muss für die spätere Extraktion der Daten das Bauteil in eine endlich große Anzahl an Positionen aufgeteilt werden, für die die jeweilige Temperatur ermittelt wird.

Simulationsmodell Zur Erstellung eines Simulationsmodells werden üblicherweise zur Verfügung stehende Daten aus Experimenten und/oder Expertenwissen zum Aufbau der Datenbasis verwendet, um daraus das Simulationsmodell zu entwerfen. Für den hier vorgestellten Fall der Wärmeleitung wird auf das bereits bestehende analytische Modell von Cole und Brettmann [10] zurückgegriffen. Die Lösung ist durch folgende Reihe gegeben:

$$\tilde{T}(p,t) = T_0 \cdot 2 \sum_{m=1}^{\infty} \frac{\beta_m^2 \sin\left(\frac{\beta_m p}{L}\right)}{\beta_m^4 + \hat{\omega}^2} \left(\cos(\omega t) + \frac{\hat{\omega}}{\beta} \sin(\omega t) + e^{\frac{\beta_m^2 \alpha t}{L^2}}\right) \tag{6.2}$$

mit $\beta_m = \frac{(2m-1)\pi}{2}$.

Zur Normierung werden folgende dimensionslose Größen eingeführt:

$$\begin{aligned} \hat{T} &= \frac{\tilde{T}}{T_0} \\ \hat{\omega} &= \frac{\omega L^2}{\alpha} \\ \hat{\sigma} &= \frac{1+i}{\sqrt{2}}\sqrt{\omega} \\ \hat{t} &= \frac{\alpha t}{L^2} \\ \hat{p} &= \frac{p}{L} \end{aligned} \tag{6.3}$$

Die Variablen der Randbedingungen in (6.1) ändern sich dementsprechend. Für die Lösung bzw. das Simulationsmodell unter normierten Bedingungen ergibt sich dann:

$$\hat{T}(p,t) = Real\left[\frac{e^{-\hat{\sigma}\hat{p}} + e^{-\hat{\sigma}(2-\hat{p})}}{\left(1 + e^{-2\hat{\sigma}}\right)} e^{i\hat{\omega}\hat{t}}\right] + 2\sum_{m=1}^{\infty} \frac{\beta_m^2 \sin\left(\beta_m \hat{p}\right)}{\beta_m^4 + \hat{\omega}^2} e^{-\beta_m^2 \hat{t}}. \tag{6.4}$$

Simulationsstudie Die Durchführung der Simulationsstudie hat hier zum Ziel, das zeitliche Temperaturverhalten in der Platte für unterschiedliche Frequenzen zu analysieren und dabei Daten zu sammeln, die zur Erstellung eines reduzierten Modells verwendet werden können. Basierend auf diesem definierten Ziel wird die Frequenz im Bereich [1,12] variiert. Für jede Frequenz wird für 200 Zeitschritte an 100 Positionen entlang der $x-$Achse die Temperatur ermittelt und als Zustandsvariable extrahiert. Als Messwert wird ebenfalls die Temperatur verwendet, jedoch nur an zwei Positionen im Bauteil. Für die Anwendung der Methoden zur Ermittlung eines onlinefähigen Zustandsverfolgers stehen somit nach der Durchführung der Simulationsstudie 12 verschiedene zeitliche Temperaturverläufe für das Bauteil zur Verfügung. Für jede der

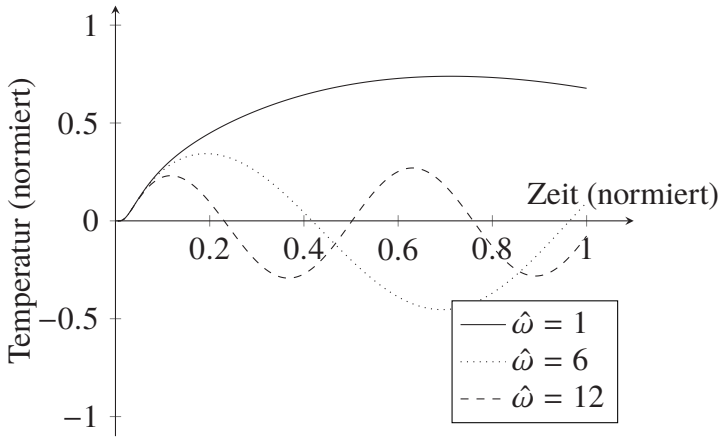


Abbildung 6.2: Temperaturverlauf für unterschiedliche Frequenzen $\tilde{\omega}$ an Position $\tilde{p} = 0{,}5$.

100 Positionen im Bauteil, an der die Zustandvariablen extrahiert werden, stehen folglich 2400 Werte zur Verfügung. Tabelle 6.1 fasst die wesentlichen Größen noch einmal zusammen.

Tabelle 6.1: Größen der Simulationsstudie.

Parameter	**Wert**
Länge Platte (normiert)	1
Dauer des Prozesses (normiert)	1
Anzahl Zeitschritte	200
Prozessparameter	$\tilde{\omega}$
Anzahl Variationen $\tilde{\omega} \in [1{,}12]$	12
Anzahl Zustandsvariablen	100
Anzahl Observablen	2

Ergebnisse Einige Ergebnisse der Simulationstudie sind in Abbildung 6.2 und 6.3 dargestellt. Sie zeigen zum einen den zeitlichen Verlauf der Temperatur in der Mitte der Platte für unterschiedliche Frequenzen, und zum anderen für die Frequenz $\tilde{\omega} = 6$ die Temperaturverteilung über das gesamte Bauteil für unterschiedliche Zeitschritte.

Abhängig von der anliegenden Frequenz verteilt sich die Temperatur im Bauteil. Wird das Bauteil im mittleren Element betrachtet, so zeigt sich, dass bei höheren Frequenzen die Temperatur über den zeitlichen Verlauf stärker alterniert. Bei Frequenz 1 in einem Zeitraum von 0 bis 1 ist die Temperatur überwiegend steigend. Bei Zeit $\tilde{t} = 0{,}005$ beträgt die transiente Temperatur im Großteil des Bauteils 0. Mit fortschreitender Zeit steigt und sinkt die Temperatur periodisch.

6.2 Tiefziehen

Für die Simulation des Tiefziehens stehen hier ein zweidimensionales und ein dreidimensionales Simulationsmodell basierend auf der Finiten-Element-Methode zur Verfügung. Wie in Kapi-

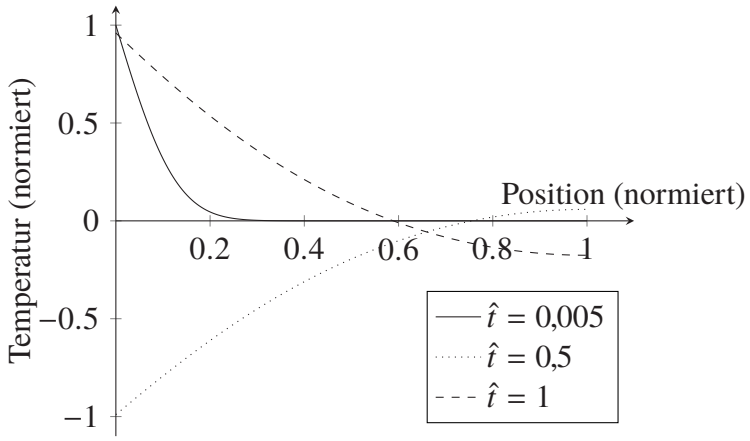


Abbildung 6.3: Temperatur im Bauteil für unterschiedliche Zeitschritte mit Frequenz $\tilde{\omega} = 6$.

tel 2.1.1 beschrieben, werden die berechneten Daten aus den Integrationspunkten der einzelnen Elemente für die Erstellung des reduzierten Prozessmodells benötigt. Um ein interpretierbares reduziertes Prozessmodell zu erlernen, ist es von Bedeutung, die Entstehung des numerischen Simulationsmodells nachzuvollziehen. Die *Problemstellung* sowie *Aufgabe und Ziel* des Simulationsmodells beschreiben zuerst kurz den Prozess des Tiefziehens bevor in *Simulationsmodell* auf die einzelnen Modelle mit ihren jeweiligen Geometrieeigenschaften, Materialmodellen und festgelegten Randbedingungen eingegangen wird.

Problemstellung Tiefziehen gehört zu den Zugdruckumformverfahren und ist ein bedeutsames Umformverfahren für Blechbauteile [13]. Hierbei wird das Ausgangshalbzeug, das durch Walzen hergestellte Blech, zwischen einem Ziehring (Matrize) und einem Niederhalter eingespannt (siehe Abbildung 6.4). Mittels eines Ziehstempels wird das Blech in die Aussparung der Matrize gezogen und wird somit zu einem Hohlkörper mit möglichst konstanter Blechdicke umgeformt.

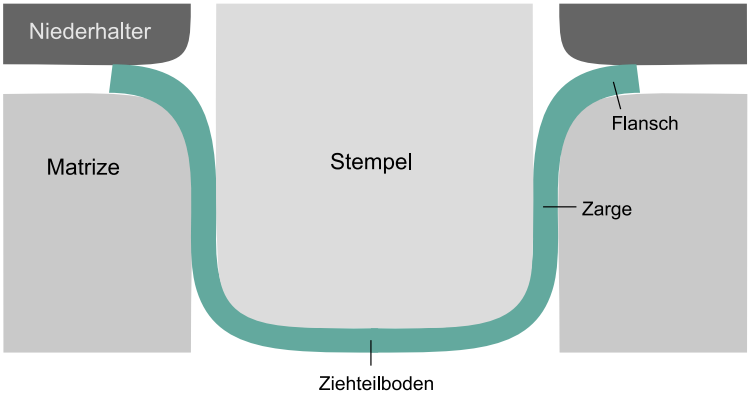


Abbildung 6.4: Schematischer Aufbau einer Tiefziehvorrichtung.

Das elastische und plastische Materialverhalten des Metalls wird durch Zugversuche ermittelt. Anhand der Messkurve der Versuchsreihe wird das probenform-unabhängige Spannungs-

Dehnungs-Diagramm beschrieben. Abbildung 6.5 zeigt exemplarisch ein solches Spannungs-Dehnungs-Diagramm. Darin ersichtlich ist der linear-elastische Bereich am Anfang, der durch

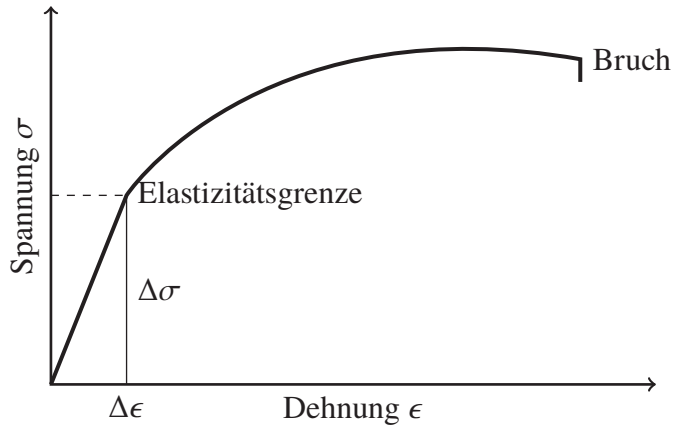


Abbildung 6.5: Schematische Darstellung des Spannungs-Dehnungs-Diagramm.

das Hookesche Gesetz (Dehnung ϵ proportional zur Spannung σ) das elastische Verhalten eines Festkörpers beschreibt und dadurch das *Elastizitätsmodul* (die Proportionalitätskonstante $E = \frac{\sigma}{\epsilon}$). Das irreversible plastische Verformen beginnt, wenn die Fließgrenze bzw. *Elastizitätsgrenze* überschritten wird. Die Plastizität wird durch eine nichtlineare Funktion beschrieben. Wenn die Zugfestigkeit überschritten wird, versagt der Werkstoff und es kommt zu Einschnürung, dass heißt, das Werkstück wird nicht mehr gleichmäßig gedehnt, was zum Bruch des Materials führt. Der Zugversuch liefert weiterhin den Kennwert zur senkrechten Anisotropie, *r-Wert*, der Aussage über die Tiefziehfähigkeit gibt, sowie den Verfestigungsexponent, *n-Wert*, der Aussage über die Streckziehfähigkeit gibt.

In dieser Arbeit betrachten wir rotationssymmetrische Bauteile. Abbildung 6.4 zeigt eine schematische Darstellung eines geformten Napfes in einer Tiefziehvorrichtung. Doege und Behrens [14] beschreiben zwei Phasen des Tiefziehens. Zuerst wird durch den Ziehstempel der Boden des Werkstückes geformt. Dabei fängt der Werkstoff an, aus der Blechdicke zu fließen. Die Radien der Kanten von Stempel sowie Matrize sind ausschlaggebend für die Blechdickenabnahme. Diese ist beim jeweiligen kleineren Radius am größten.

In der zweiten Phase des Umformvorgangs setzt das eigentliche Tiefziehen ein, dabei beginnt der Werkstoff aus dem Flanschbereich in die Zarge des Werkstückes zu fließen und sich plastisch zu verformen. Dabei treten folgende Belastungszustände in den drei Bereichen auf:

- Flansch: Zug-/Druckbeanspruchung
- Zarge: ebener Dehnungszustand durch direkte und indirekte einachsige Zugbeanspruchung
- Boden: Streckziehen durch Zug-/Zugbeanspruchung

Die im Flanschbereich herrschenden radialen Zugspannungen werden von tangentialen Druckspannungen überlagert, sodass sich die Blechdicke hier während des Tiefziehens nicht wesentlich verringert. Von Bedeutung ist hier auch, dass durch die Einstellung der Niederhalterkraft das Gleiten des Bleches zwischen Niederhalter und Matrize gewährleistet wird.

Die verschiedenen Belastungszustände können Versagensfälle verursachen und Einfluss auf die Formgenauigkeit haben. Bedeutende Versagensfälle sind Falten in Flansch (Falten 1. Art) und Zarge (Falten 2. Art), Reißer im Boden sowie – in Bezug auf Formgenauigkeit – das Rückfedern. Durch die tangentiale Druckspannung im Flanschbereich können sogenannte *Falten 1. Art* entstehen. Diese können durch Einstellen der Niederhalterkraft verhindert werden. Falten im Zargenbereich sind sogenannte *Falten 2. Art* und entstehen in Bereichen, in denen das Werkstück zeitweise oder durchweg keinen Kontakt mit den Werkzeugflächen hat. Falten 2. Art werden somit z. B. durch große Stempel- und Matrizenrundungen, einen großen Ziehspalt, einen kleinen n-Wert oder ein kleines Ziehverhältnis begünstigt. Risse im Bereich der Stempelkante im Boden des Werkstückes können durch die starke Belastung während des Umformens und durch eine zu starke Blechdickenabnahme (kleiner Stempelkantenradius) verursacht werden. Rückfederung tritt durch den elastischen Dehnungsanteil auf.

Ein weiterer ungewollter Effekt ist die Zipfelbildung. Die Zipfelbildung entsteht durch anisotropes Materialverhalten des Bleches. Dabei treten z. B. bei zylindrischen Werkstücken stärkere Dehnungen entlang und quer zur Walzrichtung des Bleches auf. Eine zusätzliche Nachbearbeitung ist notwendig, um die Zipfel zu entfernen. Es wird daher versucht, die Zipfelbildung so gering wie möglich zu halten.

Weitere Details zu Umformprozessen und speziell zum Tiefziehen finden sich in [2] und [14].

Aufgabe und Ziel Als Zustandsbeschreibung für das Bauteil soll die von Mises Vergleichsspannung, die den Spannungszustand in allen drei räumlichen Richtungen in eine eindimensionale fiktive Größe überführt, und die plastische Dehnung in Ziehrichtung auf das Verhalten bei unterschiedlichen Niederhalterkräften untersucht werden. Durch die von Mises Spannung und plastische Dehnungen als Zustandsvariablen soll das makroskopische Verhalten des Bleches während des Tiefziehens repräsentiert werden. Weiterhin werden für die Erstellung des Zustandstrackers Observablen benötigt; diese werden wie die Zustandsvariablen aus den Simulationen mit unterschiedlichen Niederhalterkräften extrahiert. Als Observablen sollen die Reaktionskraft des Stempels und der Matrizen sowie die Verschiebung des Stempels dienen. Aufgabe des Simulationsmodells und der Simulationsstudie ist somit die Ermittlung von Zustandsvariablen und Observablen in einem angemessenen Bereich unterschiedlicher Niederhalterkräfte, mit dem Ziel, einen Zustandstracker zu erstellen, der fähig ist, den Zustand des Bauteils für unterschiedliche Niederhalterkräfte zu jedem Zeitpunkt online zu bestimmen.

Simulationsmodell Mittels Finite-Elemente-Methode wird eine Näherungslösung des Systems partieller Differenzialgleichungen bestimmt. Dafür muss die Geometrie des Bauteils und der Werkzeuge modelliert werden. Zusätzlich müssen durch ein Materialmodell die Materialgesetze mit einfließen und Randbedingungen festgelegt werden.

Es stehen ein zweidimensionales und ein dreidimensionales FE-Modell zur Verfügung. Beide Modelle verwenden eine Blechronde und besitzen achsensymmetrische Randbedingungen. Für ein Material mit isotropem Verfestigungsverhalten wird ein weniger komplexes zweidimensionales Modell verwendet, das aus quadratischen Elementen besteht. Das zweite Modell verwendet ein Material mit anisotropem Verfestigungsverhalten und wird somit in einem dreidimensionalen Modell, bestehend aus einem Viertel der Blechronde mit Volumenelementen, simuliert.

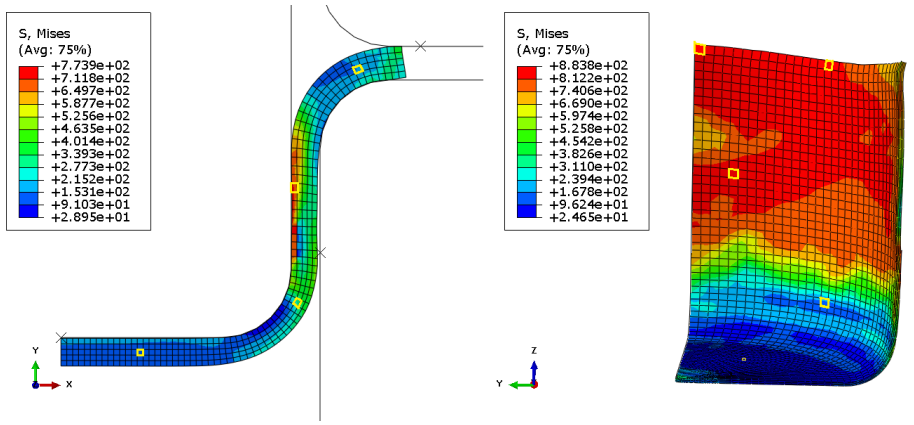


Abbildung 6.6: Zweidimensionales (links) und dreidimensionales (rechts) FE-Modell eines tiefgezogenen Napfes mit von Mises Spannung als Zustandsbeschreibung. Die gelb markierten Elemente werden für die Ergebnisse und Evaluation genauer betrachtet.

Tabelle 6.2 zeigt die Geometrieeigenschaften für das zweidimensionale und dreidimensionale Tiefziehmodell, die beide in Abbildung 6.6 dargestellt sind.

Tabelle 6.2: Geometrieeigenschaften der Tiefziehmodelle.

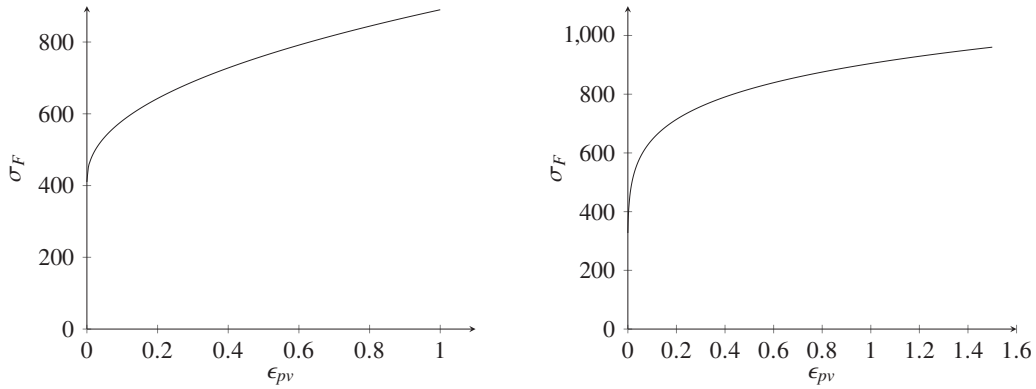
Parameter	**2D-Modell**	**3D-Modell**
Blechdicke	2,5 mm	1 mm
Rondendurchmesser	80 mm	200 mm
Stempeldurchmesser	40 mm	100 mm
Stempelkantenradius	6 mm	10 mm
Ziehverhältnis	2	2
Ziehspalt	2,5 mm	2 mm
Matritzenkantenradius	6 mm	10 mm

Das Ziehverhältnis eines Napfes ergibt sich aus dem Verhältnis zwischen dem Durchmesser der Ronde und dem Enddurchmesser des Napfbodens. Das zweidimensionale Tiefziehmodell stammt aus der Übung zur „Einführung in die FEM“ des Instituts für Technische Mechanik am Karlsruher Institut für Technologie aus dem Sommersemester 2013. Die Geometrie des Bauteils ist ein kreisrundes 2,5 mm dickes Blech mit 80 mm Durchmesser. Die Werkzeuge Stempel, Matrize und Niederhalter sind als Starrkörper modelliert. Die Materialkennwerte und -verhalten des Fe-28Mn-9Al-0,8C-Stahls sind Yoo et al. [102] entnommen.

Das isotrope Verfestigungsverhalten ist gegeben durch:

$$\sigma_F(\epsilon_{pv}) = \sigma_{F0} + \Theta\epsilon_{pv}^n \tag{6.5}$$

mit plastischer Vergleichsdehnung ϵ_{pv}, Anfangsfließspannung $\sigma_{F0} = 410$ MPa, Fließspannungszuwachs $\Theta = 480$ MPa (bei 100% akkumulierter plastischer Dehnung). Der Verfestigungsexponent n beträgt 0,45 und Reibungskoeffizient wird mit 0,03 festgelegt. Der Stempelvorschub ist 25 mm in 1 s. Durch das kreisrunde Bauteil mit isotropem Verfestigungsverhalten kann von ei-



(a) Material Fe-28Mn-9Al-0,8C-Stahl. (b) Material DP600.

Abbildung 6.7: Verfestigungsverhalten des Materials für das zweidimensionale (links) und dreidimensionale (rechts) Tiefziehmodell.

nem achsensymetrischen mechanischen Problem ausgegangen werden. Es kann dadurch mittels eines zweidimensionalen Modells abgebildet werden. Das Blechbauteil wird somit mit 5 quadratischen Elementen in der Höhe und 80 in der Breite mit linearer Ansatzfunktion gleichmäßig vernetzt.

Für das dreidimensionale Tiefziehmodell wurde ein Dualphasen-Stahl, DP600 der Firma Salzgitter Flachstahl GmbH, verwendet. Rieger [75] untersuchte im Rahmen des Graduiertenkollegs das Umformverhalten des DP600 ausführlich. Die Geometrie des Werkstückes in der vorliegenden Umformsimulation ist auch hier ein kreisrundes Blech, mit 200 mm Durchmesser und 1 mm Blechdicke.

Die Werkzeuge sind jeweils wieder als Starrkörper modelliert. Das Verfestigungsverhalten ist im experimentell ermittelten Spannungs-Dehnungs-Diagramm in Abbildung 6.7 rechts dargestellt. Der Verfestigungsexponent n beträgt 0,18, der mittlere r-Wert 0,9, die Anfangsfließspannung 327 MPa und das E-Modul 210 GPa. Der Reibungskoeffizient beträgt 0,12. Für die einzelnen experimentellen Untersuchungen, die mit dem Stahl DP600 durchgeführt wurden, sei auf [75] verwiesen.

Die experimentellen Untersuchungen zeigen, dass der DP600 ein nahezu planar-isotropes Materialverhalten aufweist. Das durch anisotropes Materialverhalten verursachte Zipfelprofil ist dadurch sehr gering. Abbildung 6.7 rechts zeigt das durch Streifenziehversuche ermittelte und auf das Simulationsmodell angewendete Verfestigungsverhalten des DP600.

Planung und Durchführung der Simulationsstudie Für die Ermittlung eines reduzierten Prozessmodells für die Zustandsbeobachtung nach Kapitel 5 wird der zeitliche Verlauf des Zustandes für unterschiedliche Niederhalterkräfte benötigt. Als Zustand soll die von Mises Spannung sowie die Dehnung in Ziehrichtung betrachtet werden. Der für die Simulationsexperimente zu variierende Prozessparameter ist die Niederhalterkraft. Ausgangspunkt der Simulation ist das eingespannte Blech mit belastetem Niederhalter. Die Simulation endet, wenn der Stempel bis zur vorgegebenen Ziehtiefe durchgefahren ist. Die dafür benötigte Zeit beträgt 1 s.

Für das zweidimensionale Tiefziehmodell wird eine Studie mit unterschiedlichen, konstanten Niederhalterkräften durchgeführt. Dabei wird die Niederhalterkraft F_{NK} 200 mal zwischen 70 kN und 100 kN variiert. Die maximale Anzahl an Zeitinkrementen, die für das vorgestellte zweidimensionale Modell benötigt werden, beträgt 129 Zeitschritte. Für die Ermittlung eines datengetriebenen Prozessmodells werden somit 400 Zustandsvariablen zu 129 Zeitschritten für 200 verschiedene Niederhalterkräfte extrahiert. Bei Simulationen mit weniger Zeitschritten, wird der Zustand für die fehlenden Zeitinkremente linear interpoliert.

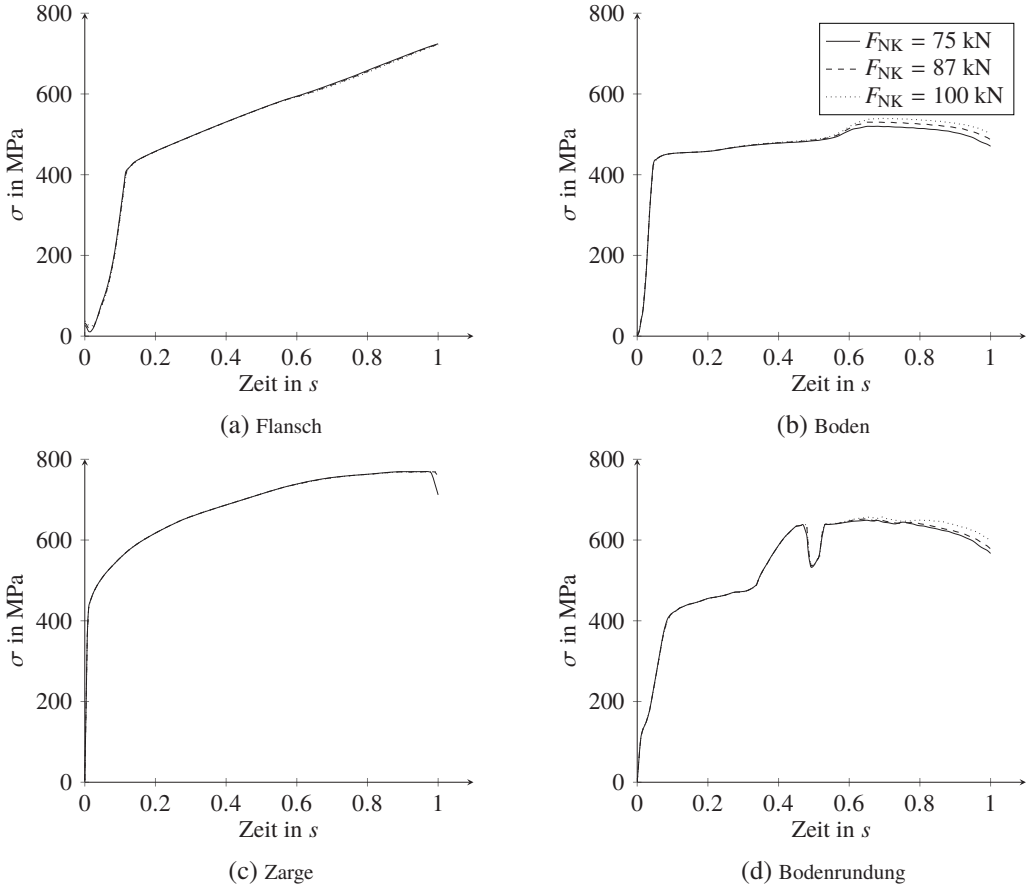
Für das dreidimensionale Modell wird als Zustand ebenfalls die von Mises Spannung sowie die plastische Dehnung in Ziehrichtung untersucht. Für die Tiefziehsimulation ist auch hier der Ausgangspunkt ein bereits belasteter Niederhalter und der Endpunkt der ausgefahrene Stempel. Die Zeit, die dafür benötigt werden soll, beträgt 2,8 s. Das dreidimensionale Modell mit seinen 8832 Integrationspunkten wird mit einer konstanten Niederhalterkraft belastet, die für die verschiedenen Simulationsdurchläufe 100 mal in äquidistanten Abständen zwischen 45 kN und 150 kN variiert wird. Es werden 41 Zeitschritte für die Simulation benötigt. Zum Erlernen eines onlinefähigen Prozessmodells steht nun ein Datensatz von $(41 \times 100) \times 8832$ Werten zur Verfügung.

Bei alle Simulationsstudien werden die Reaktionskräfte der Matrize und des Stempels sowie die Verschiebung des Stempels in Ziehrichtung als Observablen gemessen. Die Anzahl der Zeitschritte wird bestimmt durch die maximale Anzahl an Zeitschritten, die für eine Berechnung der numerischen Simulation benötigt wurde; für die Simulationen mit geringerer Anzahl an Zeitschritten werden die Werte für die fehlenden Zeitpunkte interpoliert. Es ist zu beachten, dass die Zeitinkremente nicht äquidistant sind.

Tabelle 6.3: Übersicht der Kennwerte der Simulationsstudien für das zwei- sowie dreidimensionale Tiefziehmodell.

Parameter	**2D-Modell**	**3D-Modell**
Anzahl Zustandsvariablen	400	8832
Anzahl Observablen	3	3
Anzahl Zeitinkremente	129	41
Anzahl Variationen F_{NK}	200	100
Variation F_{NK}	$[70, 100]$ kN	$[45, 150]$ kN

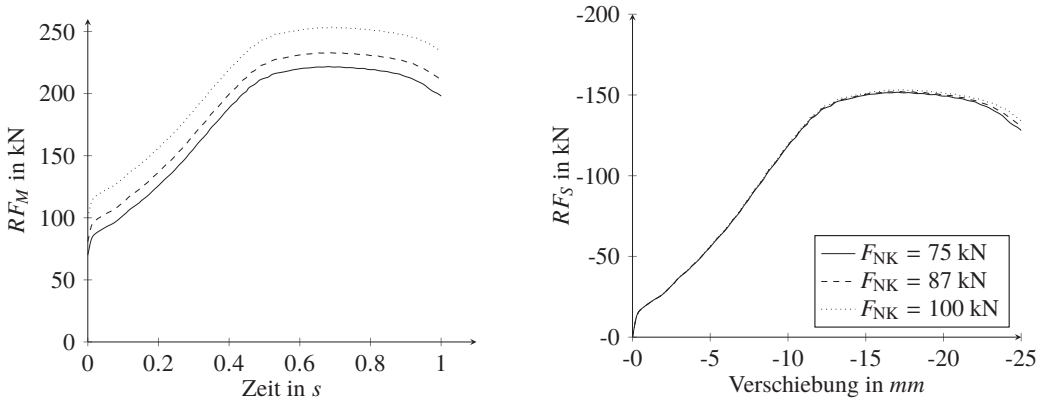
Ergebnisse Abbildung 6.8 zeigt die Simulationsergebnisse für den zeitlichen Verlauf der von Mises Spannung für unterschiedliche Niederhalterkräfte beim zweidimensionalen Modell: jeweils für ein Element im Flansch-, Zarge-, Boden- sowie Bodenrundungs-Bereich. In Abbildung 6.6 sind diese Elemente jeweils gelb umrandet. Für das Element im Flanschbereich ist der Einfluss der unterschiedlichen Niederhalterkräfte auf die von Mises Spannung am geringsten. Bei den Elementen der anderen Bereiche ist der Einfluss der Niederhalterkraft erst gegen Ende des Tiefziehens sichtbar. Die dazugehörige Matrizenkraft wird in Abbildung 6.9a dargestellt. Es ist zu sehen, dass die Reaktionskraft der Matrize bei steigender Niederhalterkraft ebenfalls höher ist. Wird die Stempelkraft in Abhängigkeit zur Stempelverschiebung in Abbildung 6.9b betrachtet, dann ist zu sehen, dass die Niederhalterkraft auch hier erst gegen Ende des Tiefziehens einen Einfluss zeigt. Der geringe Einfluss der Niederhalterkraft auf die von Mises Spannung in diesem



(a) Flansch (b) Boden

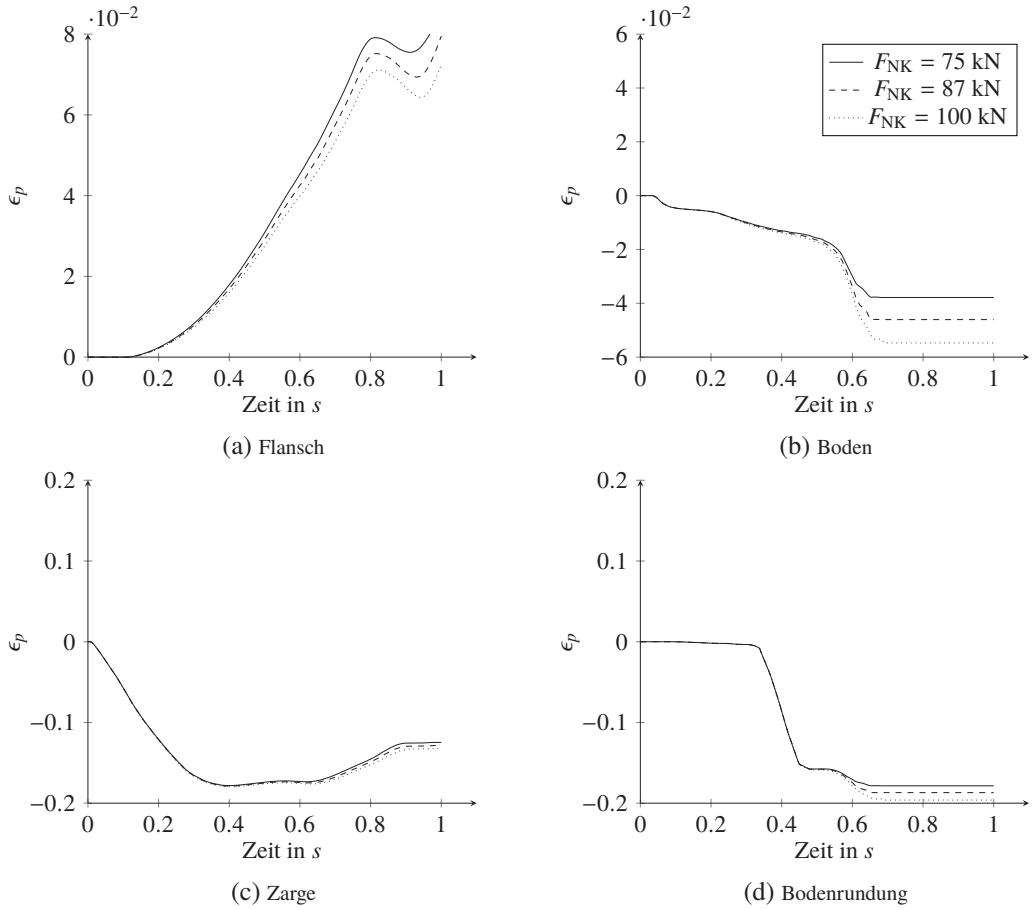
(c) Zarge (d) Bodenrundung

Abbildung 6.8: 2D-Tiefziehmodell: Zeitlicher Verlauf der von Mises Spannung in je einem Element im Flansch, Zarge, Boden und Bodenrundung des Werkstückes (vgl. Abbildung 6.6).



(a) Matrize. (b) Stempel.

Abbildung 6.9: 2D-Tiefziehmodell: Ergebnisse für unterschiedliche Niederhalterkräfte (F_{NK}). Links: Zeitlicher Verlauf der Matrizenkraft in Ziehrichtung. Rechts: Stempelkraft versus Stempelverschiebung.



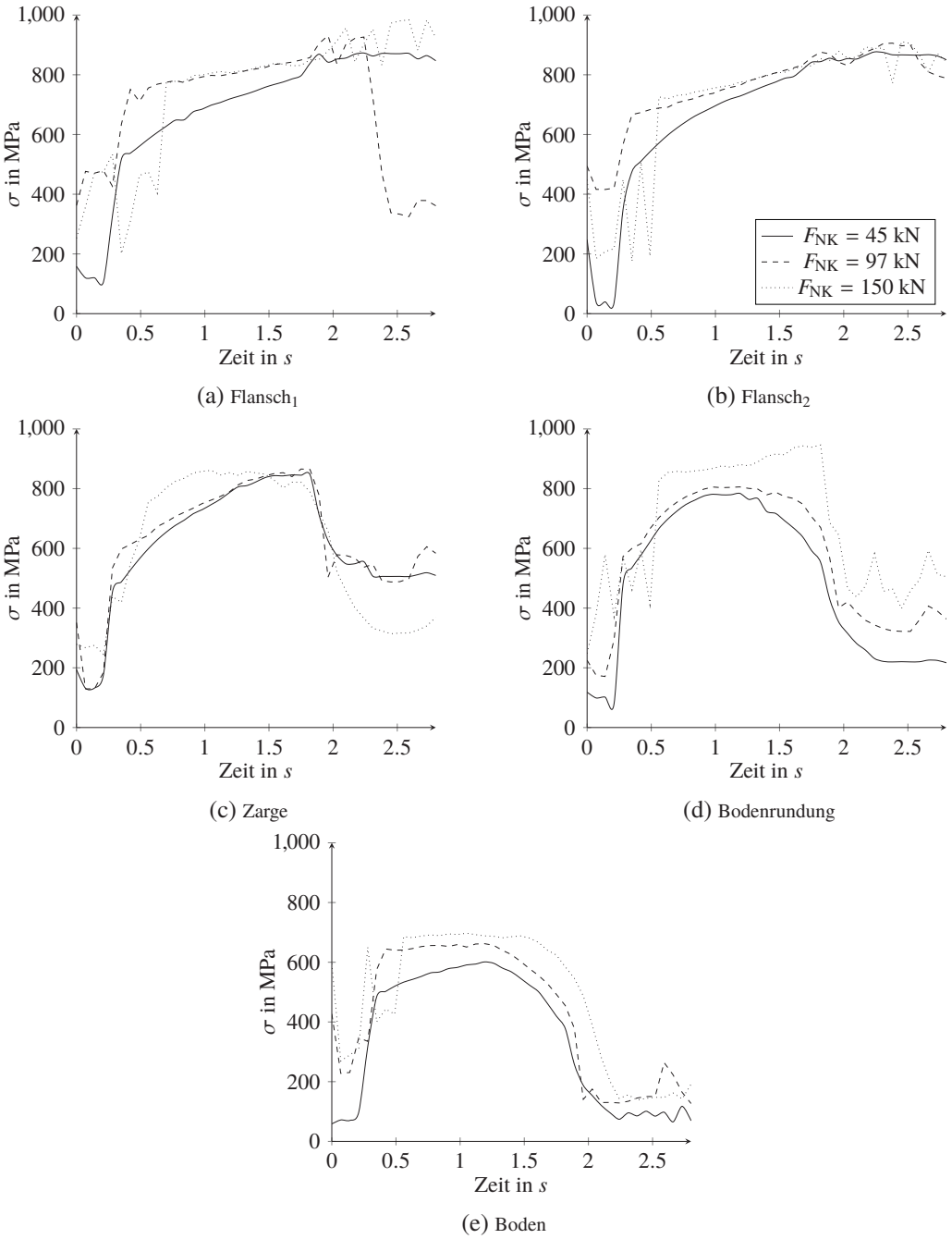
(a) Flansch (b) Boden

(c) Zarge (d) Bodenrundung

Abbildung 6.10: 2D-Tiefziehmodell: Zeitlicher Verlauf der Dehnung in Ziehrichtung in je einem Element in Flansch, Zarge, Boden und Bodenrundung des Werkstückes (vgl. Abbildung 6.6).

zweidimensionalen Modell erschwert es, die von Mises Spannung als Zustandsvariable für den Zustandstracker zu verwendet, da durch das zu erstellende reduzierte Prozessmodell Informationen verloren gehen. Es ist zu untersuchen, ob ein reduziertes Prozessmodell gefunden werden kann, welches den Zustand so exakt reproduziert, dass auch dieser geringe Einfluss sichtbar wird. Abbildung 6.10 zeigt die Ergebnisse für den zeitlichen Verlauf der plastischen Dehnung ϵ_p in Ziehrichtung in je einem Element in Flansch, Zarge, Boden sowie Bodenrundung. Deutlicher als bei der von Mises Spannung ist hier der Einfluss der Niederhalterkraft ab 0,6 Sekunden. Dabei wird das Element im Flansch im Wesentlichen gestreckt, während die gewählten Elemente in Zarge und Bodenbereich gestaucht werden.

Auch für das dreidimensionale Modell betrachten wir hier die von Mises Spannungen in einzelnen Elementen des Bauteils. Abbildung 6.11 zeigt diese zeitlichen Verläufe für drei unterschiedliche Niederhalterkräfte. Anders als im zweidimensionalen Modell ist hier der Einfluss der Niederhalterkraft deutlich sichtbar. Auch bei der Betrachtung der Observablen – des zeitlichen Verlaufs der Matrizenkraft sowie der Stempelkraft in Abhängigkeit von der Stempelverschiebung – ist der Einfluss der Niederhalterkraft in Abbildung 6.12 deutlich ersichtlich. Für den zeitlichen Verlauf der Dehnung als Zustandsvariable (Abbildung 6.13) ist vor allem im Bereich der Zarge und der Bodenrundung die Auswertung der Niederhalterkraft interessant, da die Niederhalterkraft zwischen Streckung und Stauchung des Elementes entscheidet.



(a) Flansch$_1$

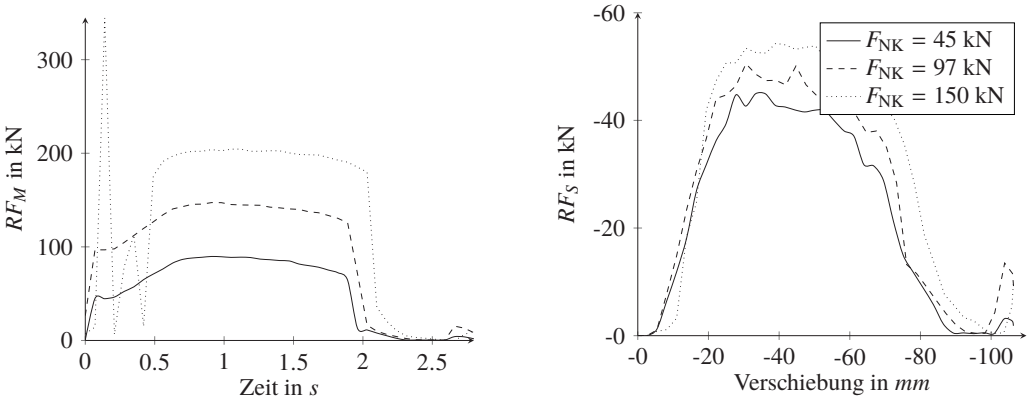
(b) Flansch$_2$

(c) Zarge

(d) Bodenrundung

(e) Boden

Abbildung 6.11: 3D-Tiefziehmodell: Zeitlicher Verlauf der von Mises Spannung in je einem Element in Flansch (Zipfel und Zipfeltal), Zarge, Bodenrundung und Boden des Werkstückes (vgl. Abbildung 6.6).



(a) Zeitlicher Verlauf der Matrizenkraft in Ziehrichtung.

(b) Stempelkraft versus Stempelverschiebung.

Abbildung 6.12: 3D-Tiefziehmodell: Ergebnisse für unterschiedliche Niederhalterkräfte (F_{NK}).

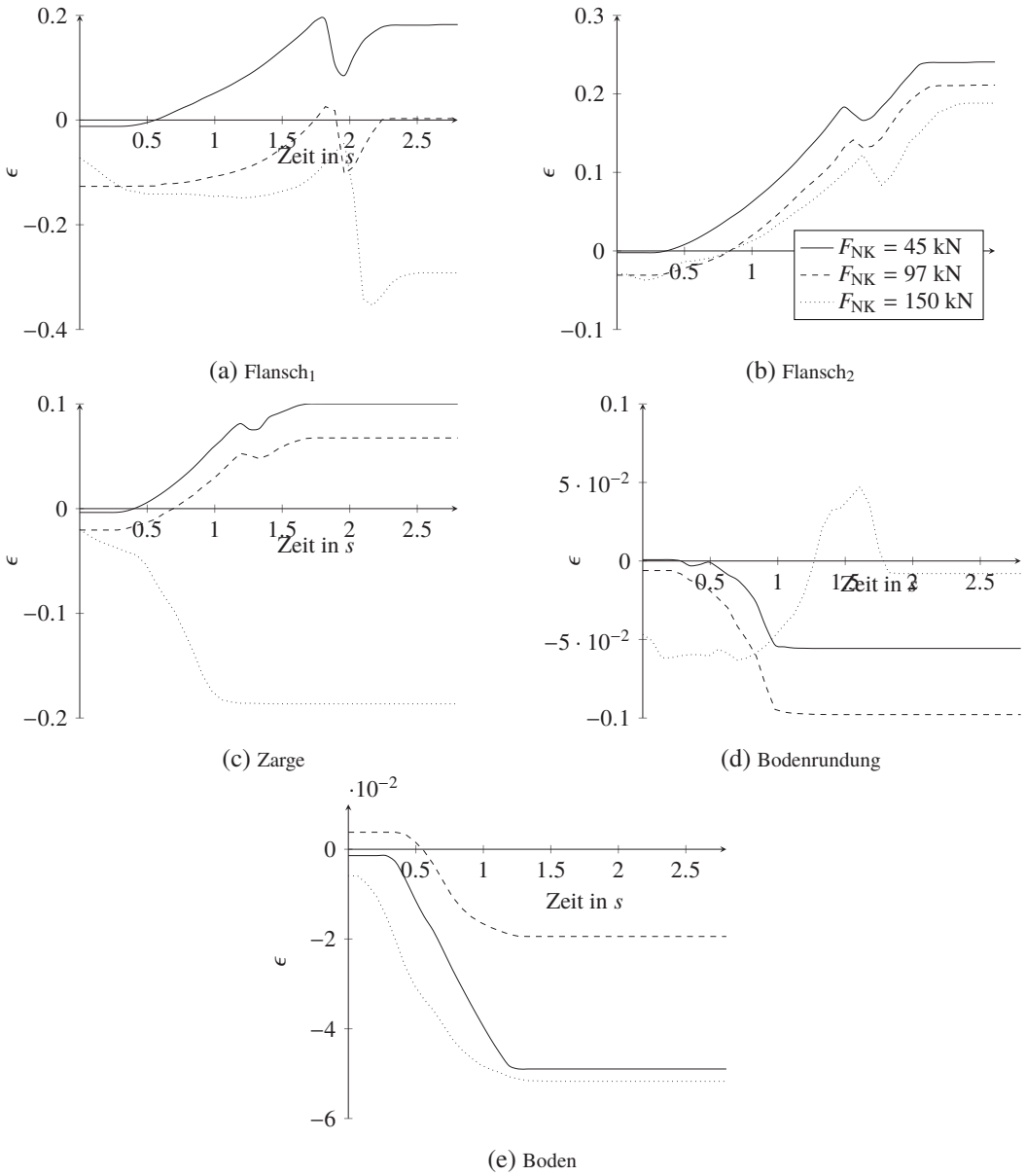


Abbildung 6.13: 3D-Tiefziehmodell: Zeitlicher Verlauf der Dehnung in Ziehrichtung in je einem Element in Flansch (Zipfel und Zipfeltal), Zarge, Bodenrundung und Boden des Werkstückes (vgl. Abbildung 6.6).

7 Evaluierung

Die Evaluierung der online Zustandsverfolgung mit datengetriebenen Prozessmodellen wird anhand der in Kapitel 6 vorgestellten Beispielprozesse durchgeführt. Dabei werden zuerst die Ergebnisse der Dimensionsreduktion für unterschiedliche Netzkonfigurationen betrachtet und mit der PCA und der NLPCA verglichen. Darauf aufbauend wird auf den Ergebnissen der Dimensionsreduktion mit dem geringsten Fehler die Identifikation des dynamischen Prozessmodells für unterschiedliche Formelausprägung durchgeführt und Performanz, Robustheit und Interpretierbarkeit genauer betrachtet. Für die Ergebnisse der vielversprechendsten erlernten Prozess- und Messmodelle wird die Zustandsbeobachtung angewendet. Es wird dabei die Beobachtbarkeit der Kombination aus Prozess- und Messmodell überprüft. Die Stabilität der gesamten Zustandsverfolgungsarchitektur wird untersucht, um Rückschlüsse auf die Verwendbarkeit der vorgestellten Dimensionsreduktion und Modellidentifikation zu treffen. Dabei wird der rekonstruierte Zustand für den Prozessverlauf mit den Ergebnissen der numerischen Simulation verglichen.

Approximationsgüte Für die drei Schwerpunkte Reduktion, Identifikation und Beobachtung werden Evaluationskriterien benötigt, die die jeweiligen Approximationen bewerten. Es wird zwischen *Fehler*, auch „wahrer Fehler“, der Abweichung des beobachten Wertes vom wahren Wert, und *Residuum*, der Abweichung des beobachteten Wertes vom rekonstruierten bzw. geschätzten Wert unterschieden. Wenn der wahre Wert nicht zur Verfügung steht, was in der Regel bei technischen Systemen der Fall ist, kann das Residuum zur Bewertung der Lösungen verwendet werden. In dieser Arbeit ist unter Fehler immer das Residuum zu verstehen, wenn nicht explizit auf „wahrer Fehler“ verwiesen wird.

Das Fehlermaß hat zwei Bedeutungen: Zum einen dient es zur Optimierung der Ergebnisse, sozusagen dem Auffinden einer Lösung, sodass der Fehler möglichst gering ist, zum anderen zum Vergleich der Lösungen unterschiedlicher Verfahren. Hierfür können unterschiedliche Fehlermaße eingesetzt werden, die für bestimmte Bewertungskriterien besser geeignet sind. In den nachfolgenden Unterkapiteln zur Reduktion, Identifikation und Beobachtung werden die Fehlermaße, die in diesen Bereichen Anwendung finden, mit jeweiliger Einsatzbegründung erläutert.

7.1 Reduktion der Zustandsvariablen

Zuerst wurden die sequenziellen bottleneck neuronalen Netze mit SBNN_{temp} und ohne SBNN Erweiterung für Zeitreihen mit den Ergebnissen für PCA und NLPCA verglichen. Vor der Dimensionsreduktion wurden die Zustandsvariablen durch Gleichung (5.1) normiert. Die lineare PCA wurde auf dem gesamten Datensatz berechnet und ist Anhaltspunkt dafür, wie der gesamte Datensatz in einem linearen Unterraum näherungsweise dargestellt werden kann – in anderen Worten:

die Transformation korrelierter Variablen in eine Menge von unkorrelierten Variablen. Sie dient dazu, umfangreiche Datensätze zu strukturieren, zu vereinfachen und zu veranschaulichen, indem eine Vielzahl statistischer Variablen durch eine geringere Zahl möglichst aussagekräftiger Linearkombinationen (die „Hauptkomponenten“) genähert werden. Die NLPCA und SBNN versuchen stattdessen nichtlineare Komponenten zu ermitteln. Wie in Kapitel 2.4.1 beschrieben bilden die Projektionswerte auf die ermittelten Komponenten die Merkmale – im nichtlinearen Fall, nichtlineare Merkmale. Diese sind für die NLPCA und SBNN nach Relevanz geordnet. Die Ergebnisse für NLPCA und SBNN wurden für einen unabhängigen zufälligen Testdatensatz, mit dem die Netze nicht trainiert wurden, ermittelt. Für beide Prozesse, Wärmeleitung und Tiefziehen, wird der zeitliche Verlauf der ersten drei Merkmale betrachtet. Anschließend wird der Einfluss der Anzahl Neuronen in den verdeckten Schichten, der Initialisierung der Gewichte und der Einfluss der Kreuzvalidierung am Beispiel des zweidimensionalen Tiefziehmodells mit den SBNN als Dimensionsreduktion untersucht. Dabei ist die Initialisierung der Gewichte durch die Variante mit (SBNN_{pre} bzw. $\text{SBNN}_{temp_{pre}}$) und ohne (SBNN bzw. SBNN_{temp}) Pretraining charakterisiert. Die Dimensionsreduktion wurde mit MATLAB® implementiert.

Approximationsgüte Bei der Reduktion werden die aus den Simulationen berechneten Werte mit den aus den Merkmalen rekonstruierten Daten verglichen. Um einen möglichst geringen Fehler zu erhalten, wird während des Trainings zum Auffinden der Merkmale der mittlere quadratische Fehler (MSE, engl. Mean Squared Error) minimiert. Die Differenz zwischen beobachteten Werten $x^{(i)}$ und geschätzten bzw. rekonstruierten Werten $\hat{x}^{(i)}$ wird quadriert und über die Anzahl der Stichproben s summiert und gemittelt:

$$E_{\text{MSE}} = \frac{1}{s} \sum_{i=1}^{s} \left(x^{(i)} - \hat{x}^{(i)} \right)^2 . \tag{7.1}$$

Der E_{MSE} bewirkt durch die Quadrierung eine stärkere Gewichtung großer Abweichungen. Ausreißer fallen dadurch stärker ins Gewicht und sorgen für einen größeren Fehler. Bei kleinem E_{MSE} ist die Varianz der Differenz zwischen ursprünglichen und rekonstruierten Daten somit klein.

Wärmeleitung Abbildung 7.1 zeigt den Vergleich des E_{MSE} für die unterschiedlichen Verfahren bei der Reduzierung der Temperatur als Zustandsvariable. Bei der Reduzierung mit der PCA waren zwei Merkmale ausreichend, um 98% der Varianz der Daten abzudecken. Somit wurde für den Vergleich der Verfahren festgelegt, zwei Merkmale zu extrahieren. Die NLPCA sowie die Varianten der SBNN erzielten mit einem Merkmal einen geringeren Fehler als die PCA. Die NLPCA erreichte dabei einen wesentlich geringeren Fehler mit einem Merkmal, erreichte aber für zwei Merkmale keinen geringeren Fehler als die PCA und $\text{SBNN}_{temp_{pre}}$-Variante. Für zwei Merkmale lagen die NLPCA und SBNN-Varianten knapp über der PCA mit einer maximalen Differenz von 0,004. Die SBNN_{temp}-Varianten für die Zeitreihen erreichten im Vergleich zur SBNN jeweils noch mal eine Verbesserung des Fehlers um ungefähr 0,002.

Abbildung 7.2 zeigt die Gegenüberstellung von ursprünglichem und aus zwei Merkmalen rekonstruiertem Zustand für den niedrigsten und höchsten Prozessparameter. Dabei ist das bessere Abschneiden der NLPCA und SBNN_{temp} gegenüber der PCA und SBNN ersichtlich.

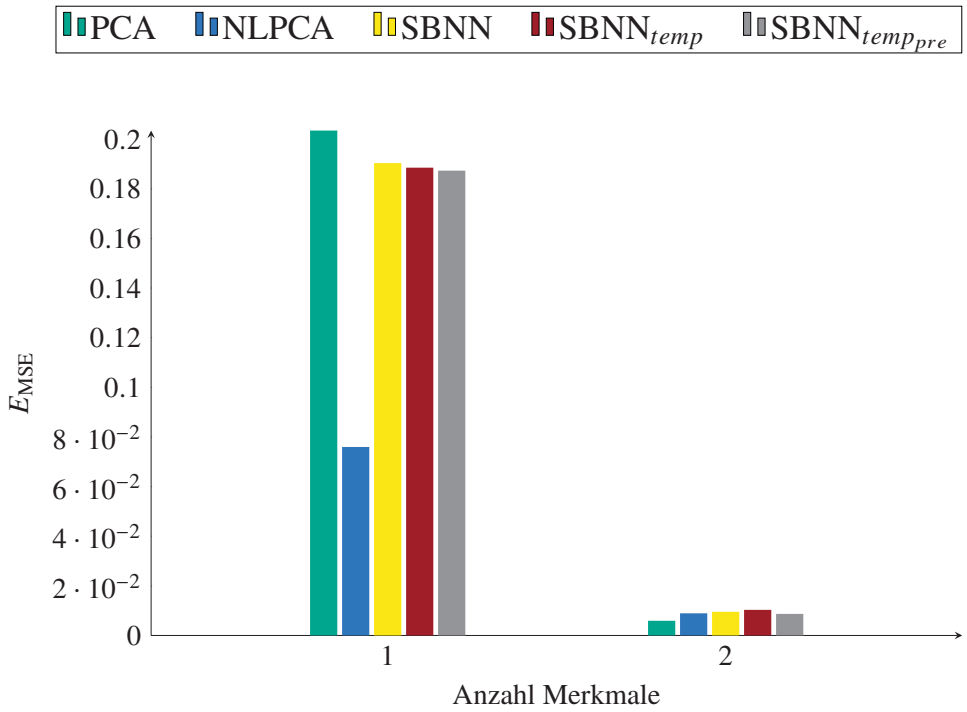
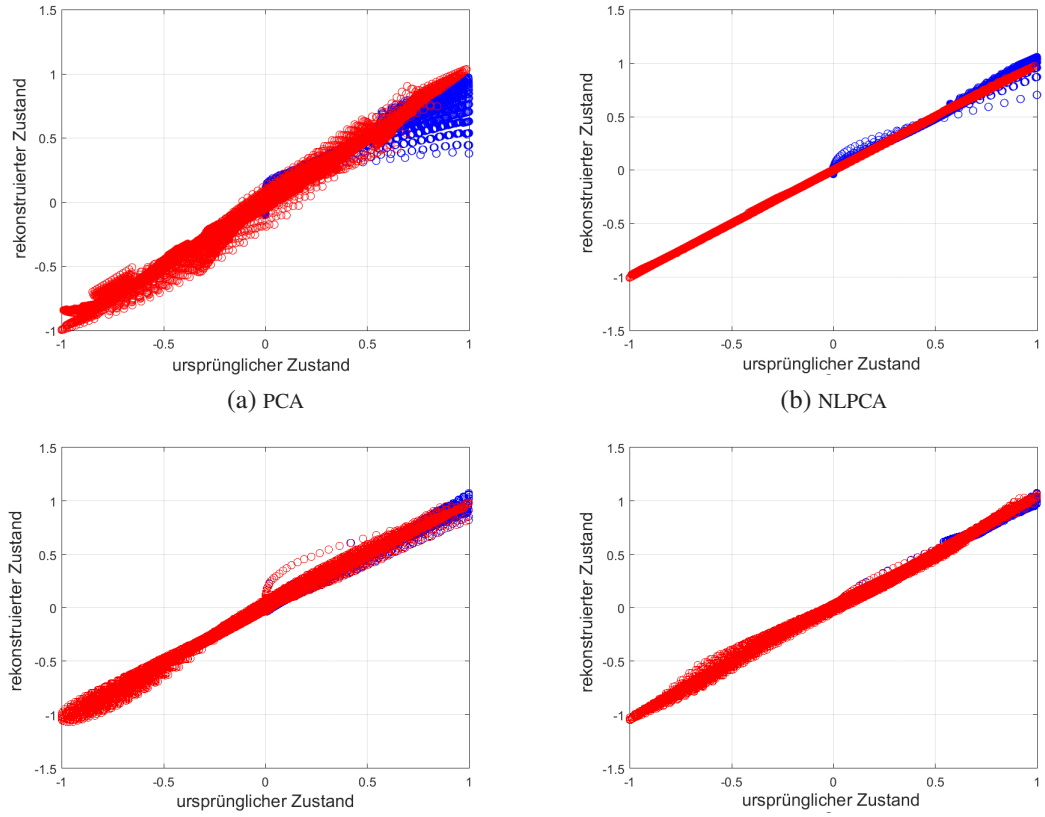


Abbildung 7.1: Wärmeleitung: Vergleich des Rekonstruktionsfehlers für die jeweilige Anzahl an Merkmalen für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{temp}$ und SBNN$_{temp_{pre}}$ auf normierten Daten.

In Abbildung 7.3 werden die Merkmale über die Zeit im Vergleich betrachtet. Dabei spiegelt sich der Vergleich des E_{MSE} darin wieder: Der Verlauf der zwei Merkmale für PCA und SBNN unterscheidet sich nicht wesentlich. Hierbei sei vermerkt, dass die Graphen an der x-Achse gespiegelt werden können, da die Daten vor der Anwendung der Dimensionsreduktion zentriert wurden und dadurch der Betrag des Komponentenwertes über die Abweichung zum Mittelwert der Daten bestimmt wird. Der charakteristische Temperaturverlauf der Ergebnisse der Wärmeleitung (in Abbildung 6.2) ist bereits im ersten Merkmal von PCA und SBNN deutlich erhalten geblieben. Im Gegensatz dazu zeigt die NLPCA zwar annähernd periodische Verläufe, weist jedoch unstetige Stellen auf.

Tiefziehen im 2D-Modell mit von Mises Spannung als Zustandsgröße. Für das zweidimensionale FE-Modell wurde die von Mises Spannung von 400 Werten auf sieben Werte pro Zeitschritt reduziert, da für die PCA sieben Merkmale ausreichend sind, um 98% Varianz der Daten abzudecken. Abbildung 7.4 zeigt, dass auch hier, wie auch beim Beispiel der Wärmeleitung, die NLPCA für das erste Merkmal einen wesentlich geringeren Fehler erreichen konnte als die PCA sowie auch einen geringeren Fehler als die SBNN. Ab dem sechsten Merkmal steigt der Fehler für die NLPCA jedoch wieder. Die extrahierten Merkmale, die durch die SBNN$_{temp}$ ermittelt wurden, haben für alle Merkmalsanzahlen den geringsten Fehler.

Abbildung 7.5 zeigt die Gegenüberstellung von ursprünglichem zu aus sieben Merkmalen rekonstruiertem Zustand für den niedrigsten und höchsten Prozessparameter. Dabei ist das bessere Abschneiden der SBNN$_{temp}$ gegenüber den anderen Methoden ersichtlich.



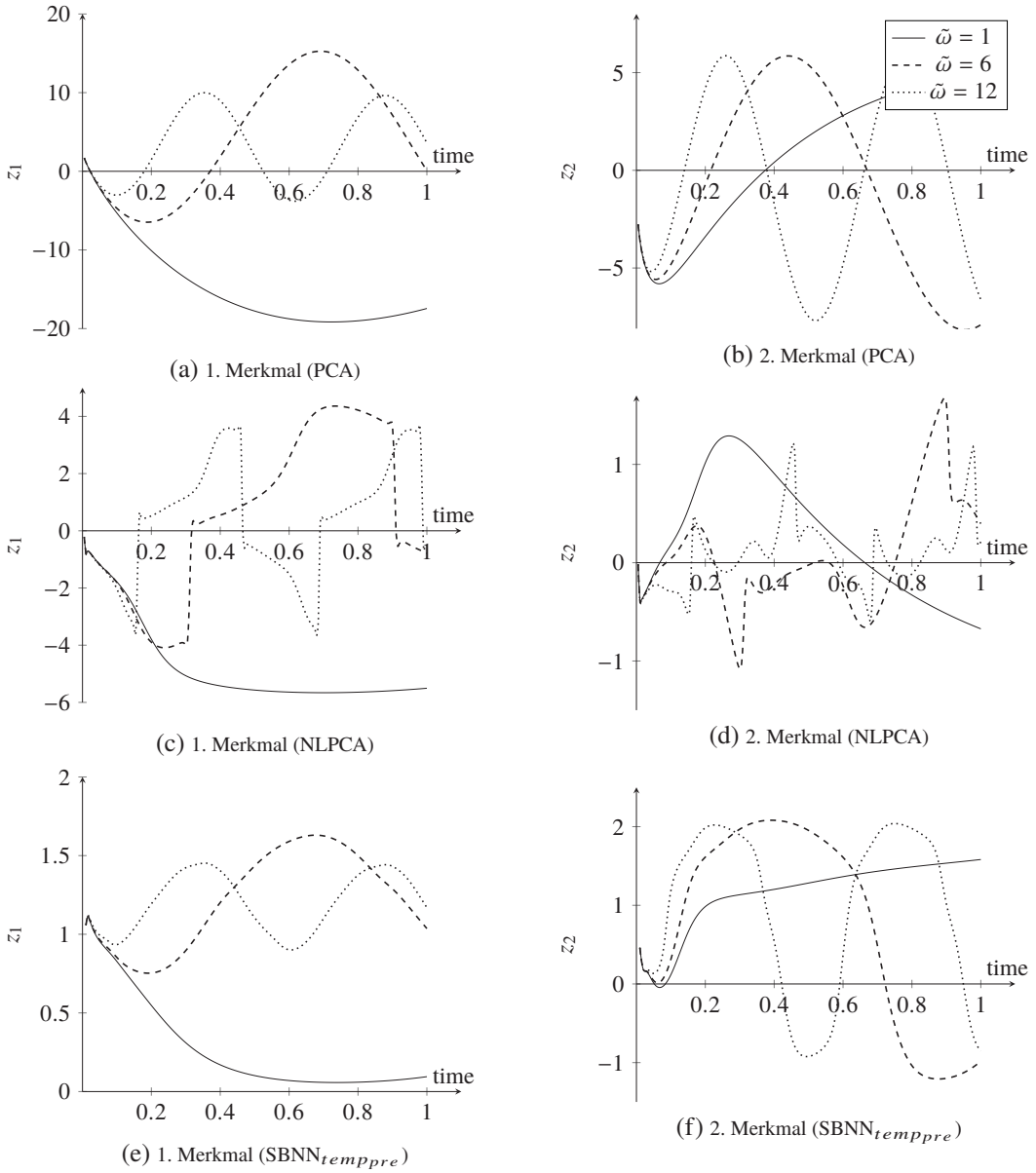
(a) PCA (b) NLPCA

(c) SBNN (d) SBNN$_{temp}$

Abbildung 7.2: Wärmeleitung: Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{temp}$ für Prozessparameter $\omega = 1$ (blau) und $\omega = 12$ (rot).

Abbildung 7.6 zeigt den Vergleich der zeitlichen Verläufe für die ersten zwei extrahierten Merkmale mittels der jeweiligen Verfahren. Für das erste Merkmal zeigen alle drei Verfahren ähnliche charakteristische Verläufe. Die SBNN$_{temp}$ erreichte dabei jedoch eine feinere Abstufung der Prozessparameter, das heißt, für unterschiedliche Prozessparameter können unterschiedliche Verläufe identifiziert werden. Der charakteristische Verlauf des ersten Merkmals gleicht dem mittleren Spannungs-Verlauf über alle Elemente. Anhang C.1 zeigt die Verläufe aller relevanten Merkmale der SBNN$_{temp}$ sowie mittlere Verläufe bestimmter Elemente des FE-Modells. Dabei ist zu erkennen, dass der mittlere Verlauf über alle Prozessparameter und alle finiten Elemente dem ersten Merkmal gleicht. Hier sei in Erinnerung gerufen, dass das erste Merkmal die Richtung der höchsten Varianz der Daten ermittelt, welche den mittleren Verlauf der von Mises Spannung im gesamten Bauteil entsprechen sollte. Das zweite Merkmal zeigt, wenn auch nur geringe, Gemeinsamkeiten zum mittleren Verlauf über alle Prozessparameter und die inneren Elemente im Boden des Werkstückes. Die restlichen Merkmalen zeigen keinen charakteristischen Verlauf, der mit dem Verlauf der von Mises Spannung in einem der Elemente des Werkstückes vergleichbar ist.

Tiefziehen im 2D-Modell mit plastischer Dehnung als Zustandsgröße. Die Dehnungswerte aus der FE-Simulation des Tiefziehens lassen sich durch die PCA auf sechs Merkmale reduzieren,



(a) 1. Merkmal (PCA)

(b) 2. Merkmal (PCA)

(c) 1. Merkmal (NLPCA)

(d) 2. Merkmal (NLPCA)

(e) 1. Merkmal (SBNN$_{temp_{pre}}$)

(f) 2. Merkmal (SBNN$_{temp_{pre}}$)

Abbildung 7.3: Wärmeleitung: Zeitliche Verläufe für die zwei Merkmale jeweils für Prozessparameter $\omega = 1$ (durchgezogene Linie), $\omega = 6$ (gestrichelte Linie) und $\omega = 12$ (gepunktete Linie), die durch PCA (**(a)**, **(b)**) NLPCA (**(c)**, **(d)**) und SBNN$_{temp_{pre}}$ (**(e)**, **(f)**) extrahiert wurden.

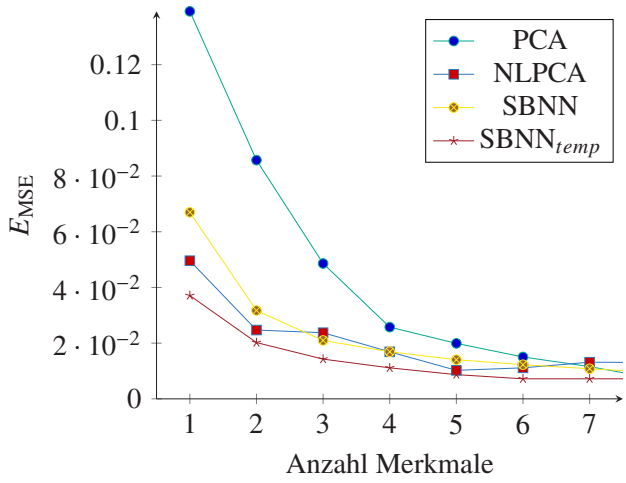
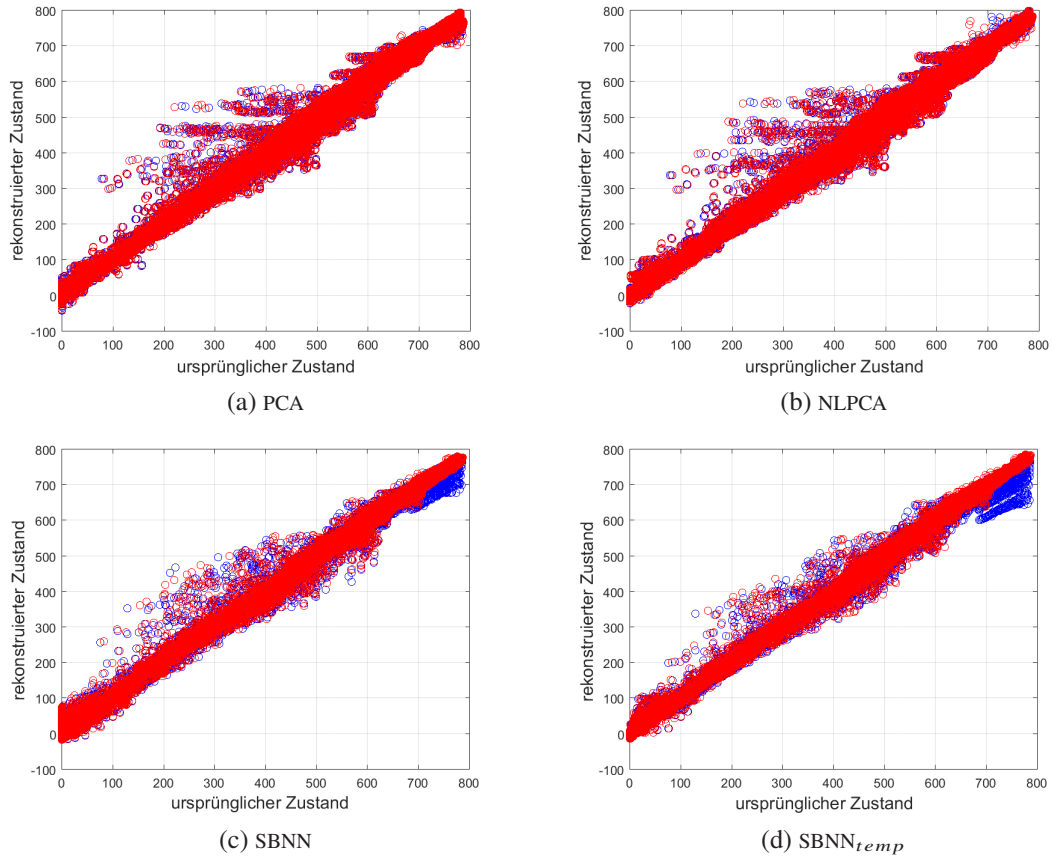


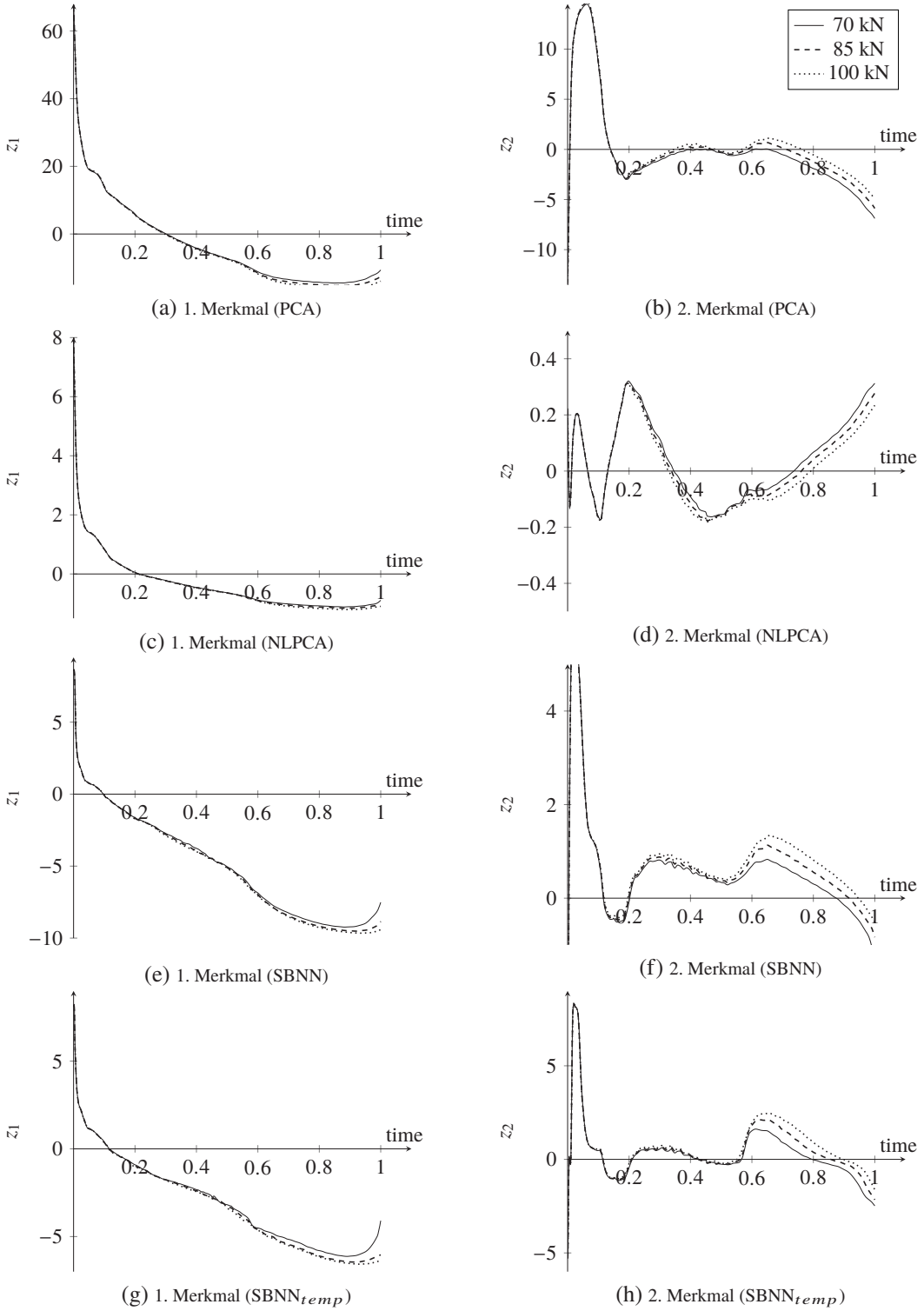
Abbildung 7.4: 2D-Tiefziehmodell (von Mises Spannung): Vergleich des Rekonstruktionsfehlers (MSE) pro Merkmal für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{temp}$.



(a) PCA (b) NLPCA

(c) SBNN (d) SBNN$_{temp}$

Abbildung 7.5: 2D-Tiefziehmodell (von Mises Spannung): Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{temp}$ für Niederhalterkraft 70 kN (blau) und 100 kN (rot).



(a) 1. Merkmal (PCA)

(b) 2. Merkmal (PCA)

(c) 1. Merkmal (NLPCA)

(d) 2. Merkmal (NLPCA)

(e) 1. Merkmal (SBNN)

(f) 2. Merkmal (SBNN)

(g) 1. Merkmal (SBNN$_{temp}$)

(h) 2. Merkmal (SBNN$_{temp}$)

Abbildung 7.6: 2D-Tiefziehmodell (von Mises Spannung): Zeitliche Verläufe der ersten zwei Merkmale für die Prozessparameter 70 kN, 85 kN und 100 kN mittels Berechnung nach PCA (**(a)**, **(b)**), NLPCA (**(c)**, **(d)**), SBNN (**(e)**, **(f)**) und SBNN$_{temp}$ (**(g)**, **(h)**).

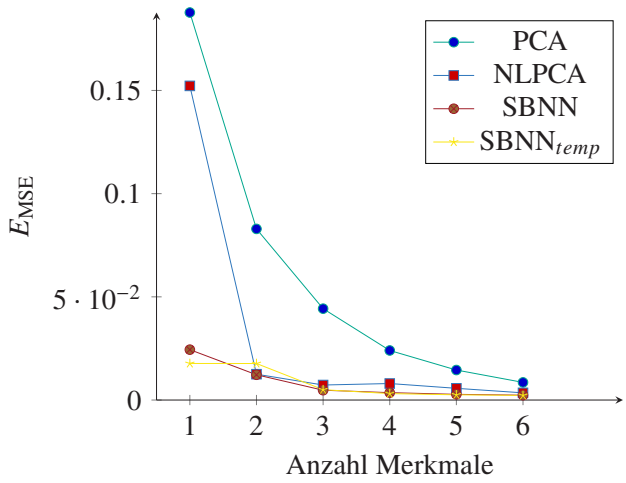


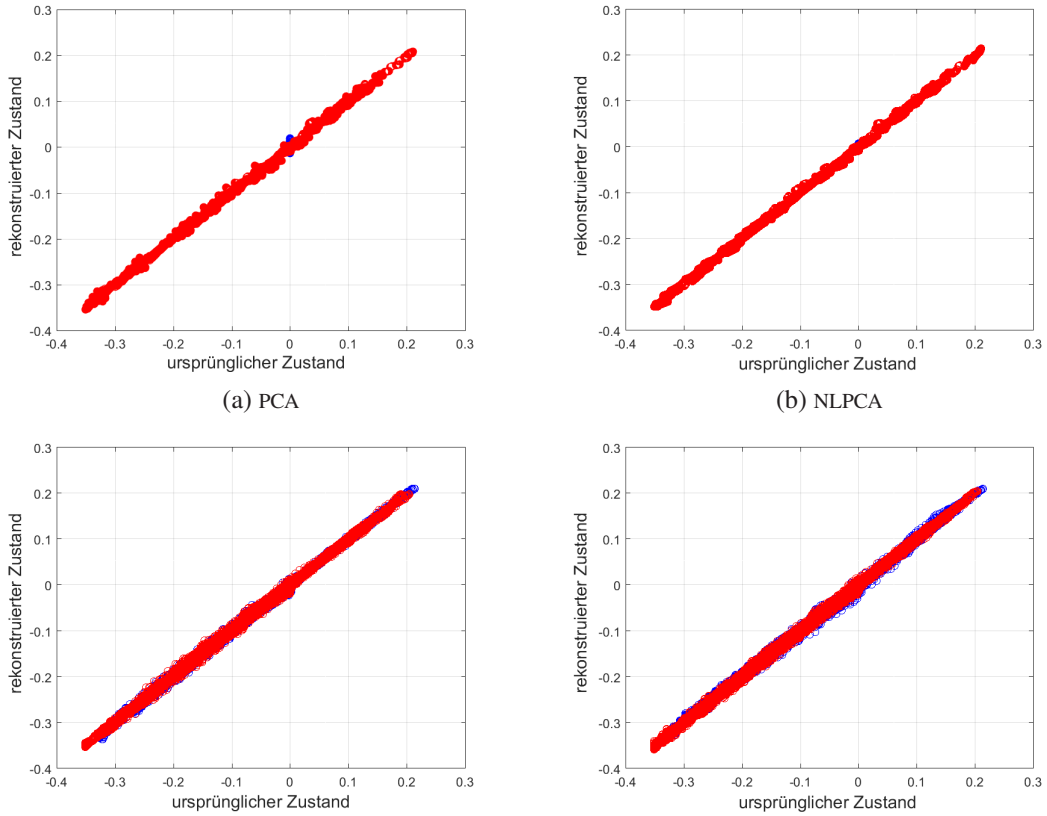
Abbildung 7.7: 2D-Tiefziehmodell (Dehnung): Vergleich des Rekonstruktionsfehlers (MSE) pro Merkmal für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{temp}$.

die immer noch 98% der Varianz der Daten abdecken. Für die Reduzierung der Werte für die plastische Dehnung in Ziehrichtung erreichten SBNN sowie SBNN$_{temp}$ vergleichbar gute Ergebnisse und ermitteln bereits für das erste Merkmal ein geringeren Fehler als die PCA und NLPCA (siehe Abbildung 7.7). Für die nachfolgenden Merkmale hat auch die NLPCA die Werte der SBNN Varianten zu vermerken, zeigt jedoch für das vierte Merkmal wieder einen Anstieg in den Fehlerwerten.

Abbildung 7.8 zeigt die Gegenüberstellung von ursprünglichem zu aus sechs Merkmalen rekonstruiertem Zustand für den niedrigsten und höchsten Prozessparameter. Die dicht beieinander liegenden Fehlerwerte aus Abbildung 7.7, nach der Rekonstruktion der Daten aus sechs Merkmalen, ist auch hier ersichtlich.

Abbildung 7.9 zeigt die zeitlichen Verläufe der Merkmale der reduzierten Dehnungswerte. Einen deutlichen Unterschied zwischen den verschiedenen Prozessparametern zeigt beim ersten Merkmal nur die PCA und beim zweiten Merkmal die NLPCA und SBNN. Das erste Merkmal zeigt für alle Verfahren ähnliches charakteristisches Verhalten und zeigt Ähnlichkeiten zum zeitlichen Verhalten der plastischen Dehnung im Bereich des Flansches. Die Verläufe für alle sechs Merkmale der SBNN$_{temp}$ sowie mittlere Verläufe der plastischen Dehnung sind in Anhang C.2 gelistet. Die SBNN$_{temp}$ erzeugt für das zweite Merkmal einen Verlauf, der Ähnlichkeiten zum Verlauf der Dehnung in Elementen in der Bodenrundung aufzeigt.

Tiefziehen im 3D-Modell mit von Mises Spannung als Zustandsgröße. Für das dreidimensionale Modell liegen 8832 Zustandsgrößen für jeden Zeitschritt vor. 41 Zeitinkremente sind ausreichend, um das Tiefziehen mittels des gegebenen Finite-Elemente-Modells zu simulieren. Da die Simulation für 100 verschiedene Niederhalterkräfte durchgeführt wurde, lagen 4100 Stichproben zur Verfügung. Um eine stabile und aussagekräftige Dimensionsreduktion durchzuführen, sollten mehr Stichproben als Zustandsgrößen vorliegen. Deshalb wurden anschließend nur 2223 Elemente auf der Oberseite des Bleches betrachtet, die sich nach dem Tiefziehen im Inneren des Napfes befinden. Bei der Reduktion der von Mises Spannungswerte durch die PCA

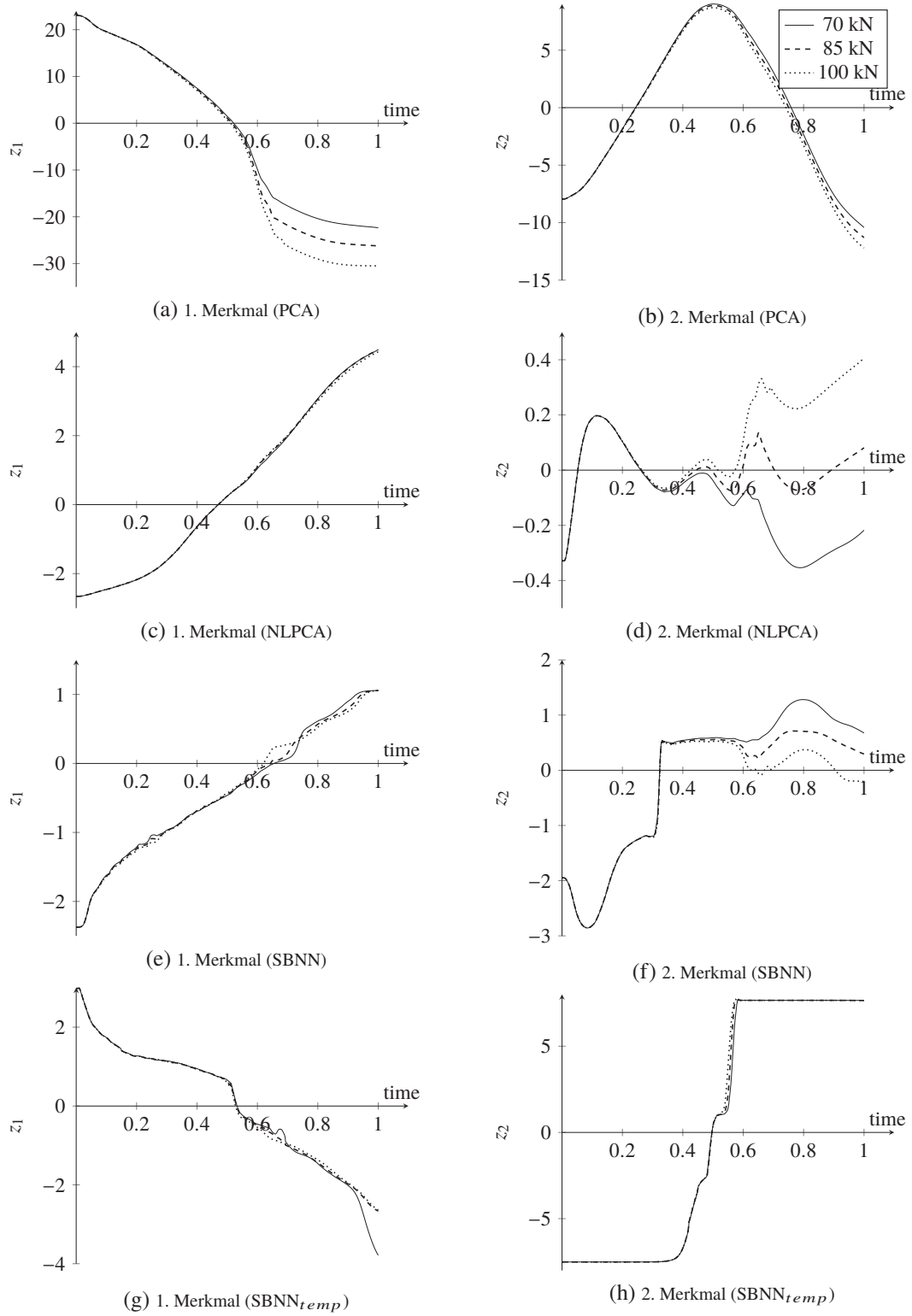


(a) PCA (b) NLPCA

(c) SBNN (d) SBNN$_{temp}$

Abbildung 7.8: 2D-Tiefziehmodell (Dehnung): Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{temp}$ für Niederhalterkraft 70 kN (blau) und 100 kN (rot).

können mit 8 Merkmalen 78% der Varianz der Daten rekonstruiert werden. Um 98% der Varianz der Daten rekonstruieren zu können, werden 393 Merkmale benötigt. Abbildung 7.10 zeigt den Rekonstruktionsfehler, der mit der PCA, NLPCA sowie SBNN ohne und mit Pretraining für 8 Merkmale erreicht wurde. Für die ersten drei Merkmale vermerkten die SBNN$_{pre}$ den geringsten Fehler. Bei allen weiteren Merkmalen erreichte die PCA den geringsten Fehler, jedoch sei hier in Erinnerung gerufen, dass die PCA auf dem kompletten Datensatz berechnet wurde und somit keine Aufteilung in Trainings- und Testdaten erfolgte, da die PCA hier lediglich als Richtwert gilt, wie gut die Daten linear approximiert werden können. Bei der NLPCA sind, wie bereits beim zweidimensionalen Modell, Anstiege des Fehlerwertes mit steigender Anzahl an Merkmalen zu beobachten. Dies deutet darauf hin, dass Merkmale mit deutlich unterschiedlicher Varianz extrahiert werden können, jedoch die NLPCA bei höherer Anzahl an Merkmalen, mit der geringeren Varianz in den Daten, keine eindeutigen Merkmale extrahieren kann. Datenvorverarbeitung, wie vorherige ausgeführte PCA, oder eine höhere Anzahl an Trainingsepochen würden die NLPCA bei der Findung eindeutiger Merkmale bei niedrigen Varianzen unterstützen, für den Vergleich der Verfahren wurden in dieser Arbeit jedoch keine weiteren Optimierungsschritte vorgenommen, da diese Verbesserungsschritte auch zu einer Verbesserung der Ergebnisse der SBNN führen würden.



(a) 1. Merkmal (PCA)

(b) 2. Merkmal (PCA)

(c) 1. Merkmal (NLPCA)

(d) 2. Merkmal (NLPCA)

(e) 1. Merkmal (SBNN)

(f) 2. Merkmal (SBNN)

(g) 1. Merkmal (SBNN$_{temp}$)

(h) 2. Merkmal (SBNN$_{temp}$)

Abbildung 7.9: 2D-Tiefziehmodell (Dehnung): Zeitliche Verläufe der ersten zwei Merkmale für die Prozessparameter 70 kN, 85 kN und 100 kN für jeweils Berechnung nach PCA ((**a**), (**b**)), NLPCA ((**c**), (**d**)), SBNN ((**e**), (**f**)) und SBNN$_{temp}$ ((**g**), (**h**)).

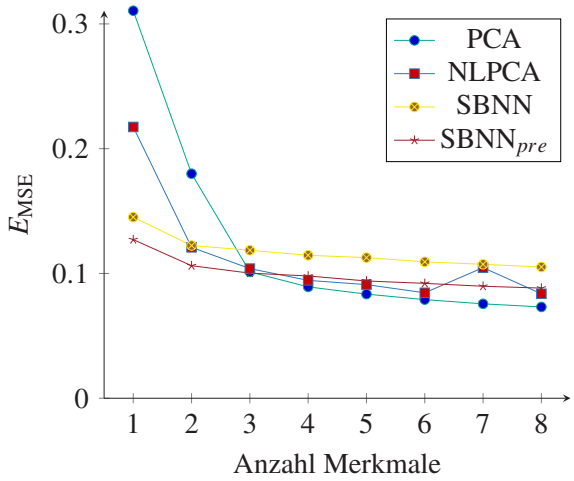
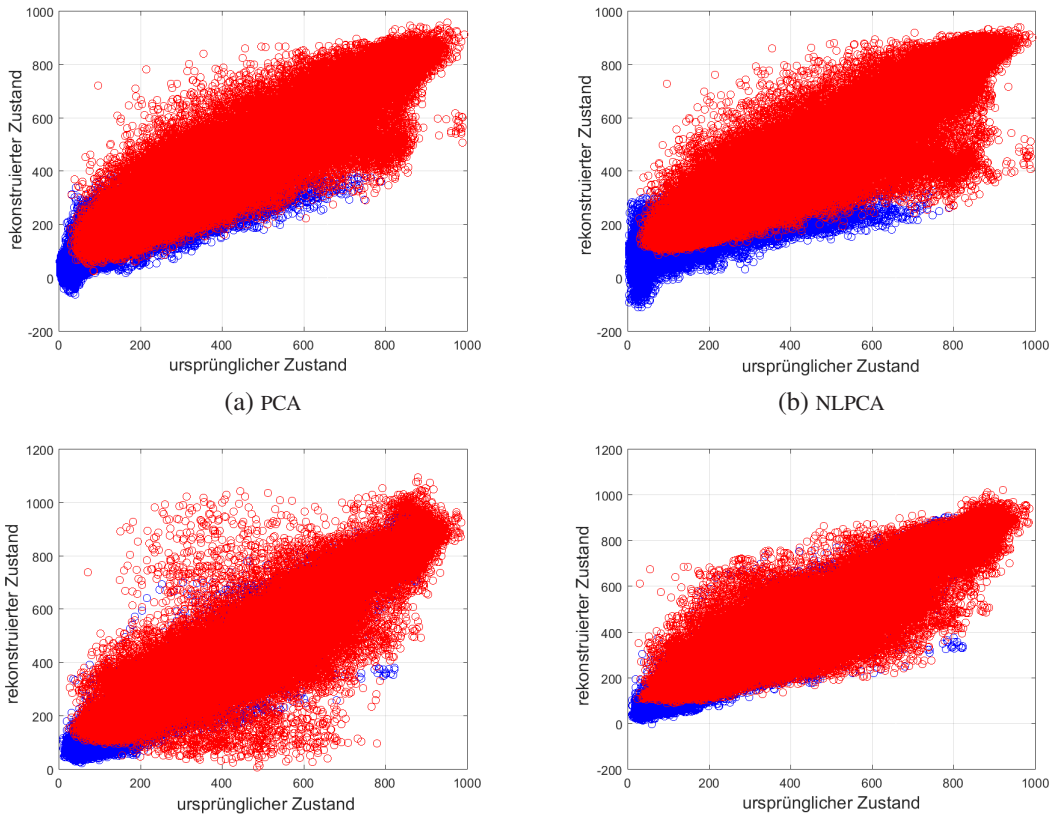


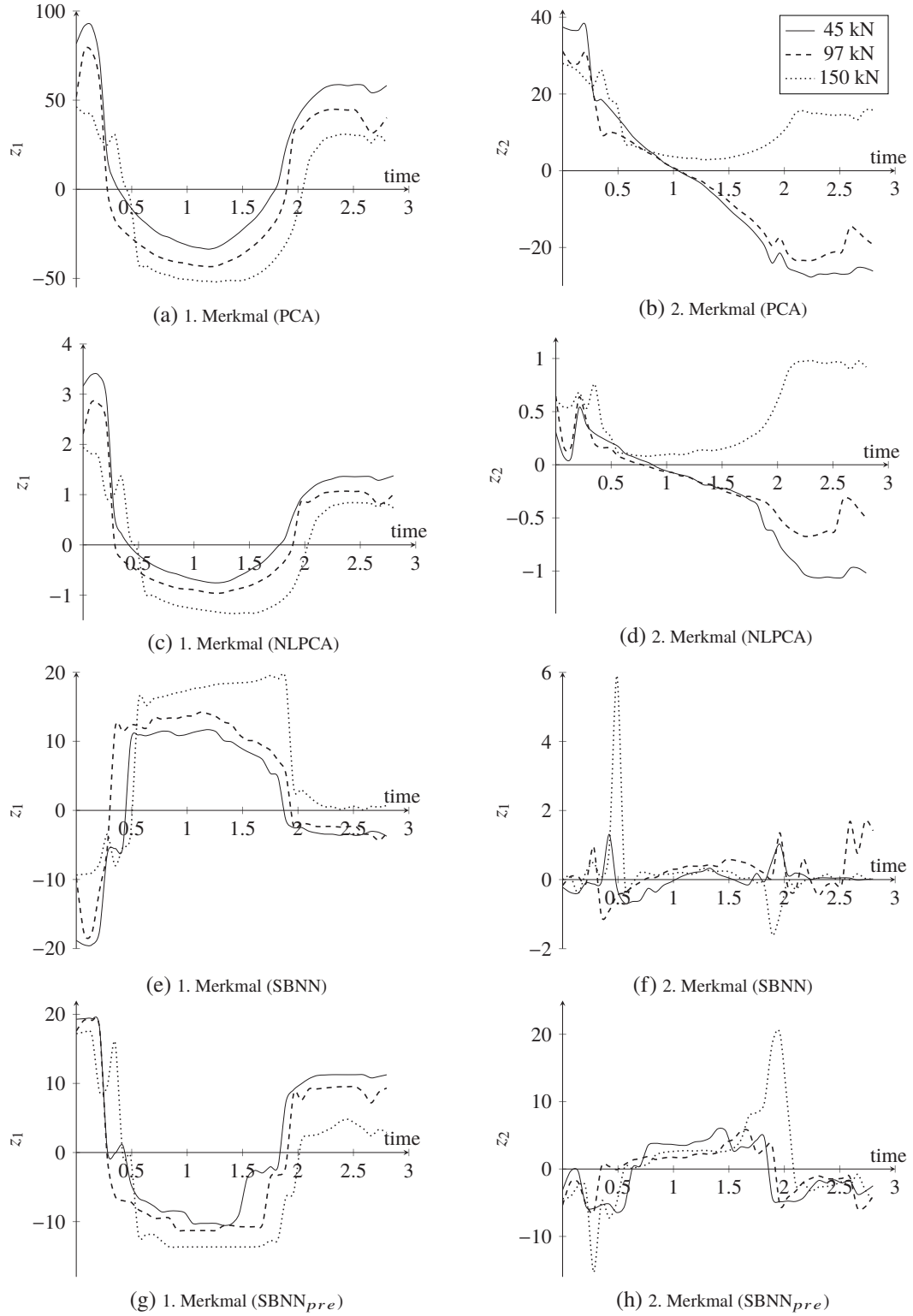
Abbildung 7.10: 3D-Tiefziehmodell (von Mises Spannung): Vergleich des Rekonstruktionsfehlers pro Merkmal für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{pre}$.



(a) PCA (b) NLPCA

(c) SBNN (d) SBNN$_{pre}$

Abbildung 7.11: 3D-Tiefziehmodell (von Mises Spannung): Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{pre}$ für Niederhalterkraft 45 kN (blau) und 150 kN (rot).



(a) 1. Merkmal (PCA)

(b) 2. Merkmal (PCA)

(c) 1. Merkmal (NLPCA)

(d) 2. Merkmal (NLPCA)

(e) 1. Merkmal (SBNN)

(f) 2. Merkmal (SBNN)

(g) 1. Merkmal (SBNN$_{pre}$)

(h) 2. Merkmal (SBNN$_{pre}$)

Abbildung 7.12: 3D-Tiefziehmodell (von Mises Spannung): Zeitliche Verläufe der ersten zwei Merkmale für die Prozessparameter 45 kN, 97 kN und 150 kN für jeweils Berechnung nach PCA (**(a)**, **(b)**) NLPCA (**(c)**, **(d)**), SBNN (**(e)**, **(f)**) und SBNN$_{pre}$ (**(g)**, **(h)**).

Abbildung 7.11 zeigt die Gegenüberstellung von ursprünglichem zu aus acht Merkmalen rekonstruiertem Zustand für den niedrigsten und höchsten Prozessparameter. Die hohen Fehlerwerte bei der Rekonstruktion aus nur acht Merkmalen (Abbildung 7.10) ist hier durch die breite Streuung wiederzufinden.

Abbildung 7.12 zeigt die zeitlichen Verläufe der ersten zwei Merkmale für jeweils die Niederhalterkräfte 45 kN, 97 kN und 150 kN. Für das erste Merkmal konnten wieder ähnliche Verläufe extrahiert werden. Abbildung C.2 in Anhang C.1 zeigt zusätzlich Merkmale drei bis vier, sowie den mittleren Verlauf der von Mises Spannung über alle Parameter und alle Elemente. Diese mittlere Verlaufe ähneln stark dem ersten Merkmal der PCA. Das „alternierende“ Verhalten bis 0,5 Sekunden und ab 2 Sekunden im zweiten Merkmal für den Prozessparameter 150 kN ist ebenfalls im zeitlichen Verlauf der von Mises Spannung in den Elementen der Bodenrundung und Boden (Abbildung 6.13d und 6.13e) wiederzufinden.

Konfiguration des Netzes Die Performanz eines neuronalen Netzes ist stark von dessen Konfiguration abhängig. Kapitel 5.1.4 begründet die in dieser Arbeit verwendete Konfiguration. Dabei wird im Wesentlichen ein Blick auf die Anzahl Neuronen in den verdeckten Schichten, die Initialisierung der Gewichte sowie die Verwendung von Kreuzvalidierung geworfen.

Es wurde das zweidimensionale Tiefziehmodell mit der von Mises Spannung als Zustandsgröße untersucht. Dabei wurden die Stichproben für die Trainings- und Testmenge für die Bewertung der gesamten Netzarchitektur zufällig gewählt. Mittels der Hauptkomponentenanalyse konnte die Anzahl linearer Komponenten bestimmt werden, die benötigt werden, um z. B. 98% Varianz der Daten abzudecken. Diese Anzahl wurde jeweils als initiale Anzahl für die Anzahl Neuronen in der Mapping- bzw. Demapping-Schicht verwendet. Von dieser Anzahl ausgehend wurden weitere Netze trainiert, mit jeweils geringerer und höherer Anzahl an Neuronen in den verdeckten Schichten. Bei der Kreuzvalidierung wurden die Daten in k gleichgroße Mengen aufgeteilt. Eine Menge davon wurde für das Testen verwendet, der Rest für das Training der gesamten Netzarchitektur. Dies wurde k mal durchgeführt, jeweils mit einer anderen Testmenge. Das Ergebnis für die gesamte Netzarchitektur ergibt sich aus dem besten der k Durchläufe, d. h. dem Durchlauf mit dem geringsten Testfehler. Es wurden einfache, dreifache und fünffache Kreuzvalidierung verglichen. Zur Initialisierung der Gewichte wurde das Pretraining mit einer zufälligen Gewichtung der Verbindungen zwischen den Neuronen verglichen.

Abbildung 7.13 zeigt den E_{MSE} für alle Durchläufe bei einer dreifachen Kreuzvalidierung mit jeweils 6, 8, 10, 20 und 50 Neuronen in der Mapping- und Demapping-Schicht und ohne Pretraining. Zu beobachten ist, dass mit steigender Anzahl Neuronen in der Mapping- und Demapping-Schicht auch der Fehler des Testdatensatzes sinkt. Betrachtet man jedoch den zeitlichen Verlauf der extrahierten Merkmale, dann ist zu sehen, dass mit steigender Anzahl Neuronen in der Mapping- und Demapping-Schicht auch die Komplexität des Zeitverhaltens der Merkmale zunimmt. Dies wirkt sich beim zweidimensionalen Modell ab dem zweiten Merkmal deutlich aus. Abbildung 7.14 zeigt die Auswirkung am Beispiel des zweiten und dritten Merkmals bei konstantem Prozessparameter von 70 kN.

Abbildung 7.15 zeigt die Ergebnisse für einfache, zweifache und dreifache Kreuzvalidierung. Bei der einfachen Kreuzvalidierung wird der Datensatz halbiert in Training- und Testdaten und das Training und Testen jeweils nur einmal durchgeführt. Auffällig dabei ist, dass für die einfache

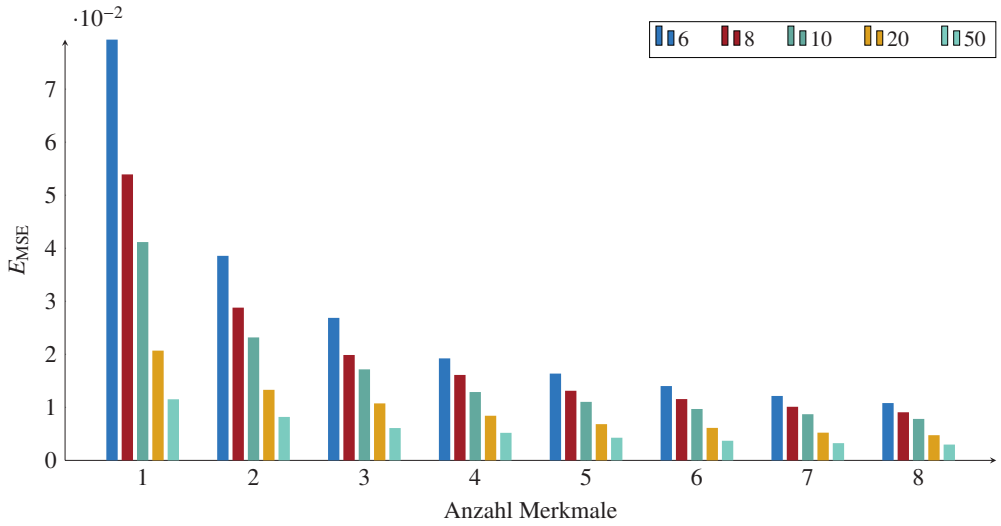
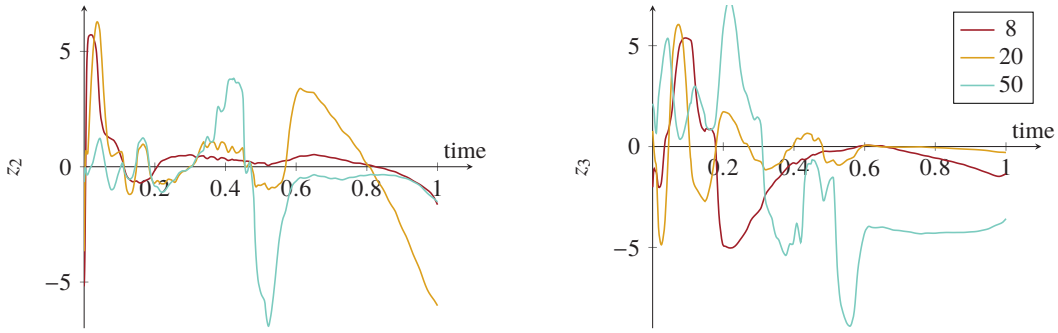


Abbildung 7.13: Ergebnisse Reduzierung von Mises Spannung für das zweidimensionale Modell für unterschiedliche Anzahl Neuronen in den verdeckten Schichten bei dreifacher Kreuzvalidierung ohne Pretraining.



(a) 2. Merkmal (b) 3. Merkmal

Abbildung 7.14: Vergleich zeitlicher Verlauf des zweiten und dritten Merkmals für unterschiedliche Anzahl Neuronen in den verdeckten Schichten bei dreifacher Kreuzvalidierung.

Kreuzvalidierung ab dem zweiten Merkmal für die Variante mit Pretraining ein wesentlich höherer Fehler auftritt als für die Variante mit der zufälligen Initialisierung der Gewichte. Die Ergebnisse der einfachen Kreuzvalidierung zeigen jedoch, dass eine k Kreuzvalidierung (in der Regel $k > 1$) zu stabileren Gesamtergebnissen führt. Zu beachten ist, dass die Trainingsdaten der gesamten Architektur für jedes einzelne Netz in der sequentiellen Architektur in Trainingsdaten, Validierungsdaten und Testdaten unterteilt wurden, um für jedes Netz unabhängig von den anderen Early Stopping (siehe Kapitel 5.1.4) durchzuführen.

Weiterhin zeigt die Abbildung, dass durch Pretraining für diese Netzarchitektur und Datensatz nur eine geringe Verbesserung bis zum dritten Merkmal erzielt wurde, für alle weiteren Merkmale sogar keine Verbesserung des Fehlerwertes. Vergleicht man den zeitlichen Verlauf der ersten vier Merkmale (Abbildung 7.16) mit und ohne Pretraining, so ist offensichtlich, dass erst ab den dritten Merkmal Unterschiede im Verlauf ersichtlich sind. Daraus lässt sich schließen, dass das

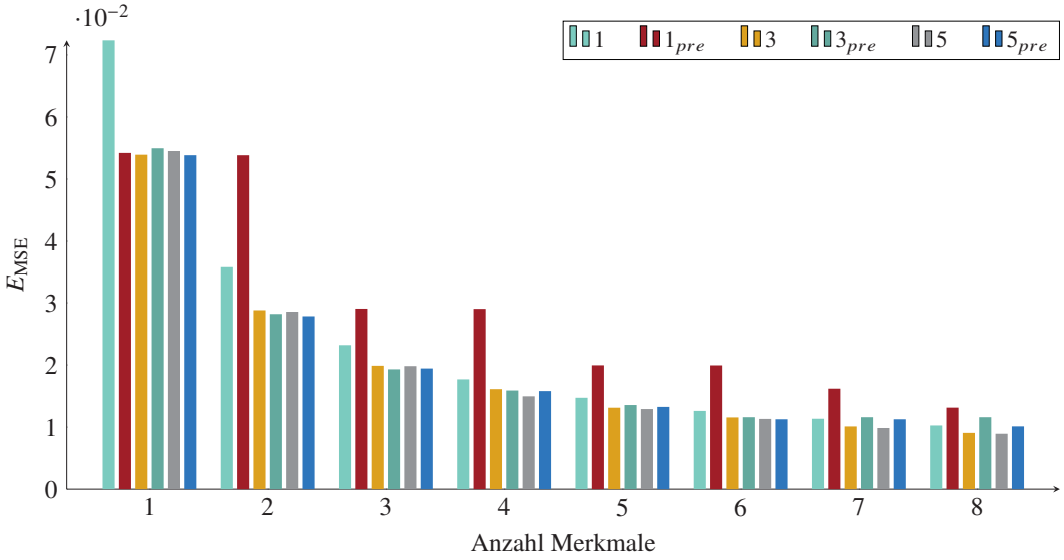


Abbildung 7.15: Ergebnisse Reduzierung von Mises Spannung für das zweidimensionale Modell für unterschiedliche Anzahl Kreuzvalidierungen bei 8 Merkmalen in den verdeckten Schichten mit und ohne Pretraining $_{pre}$.

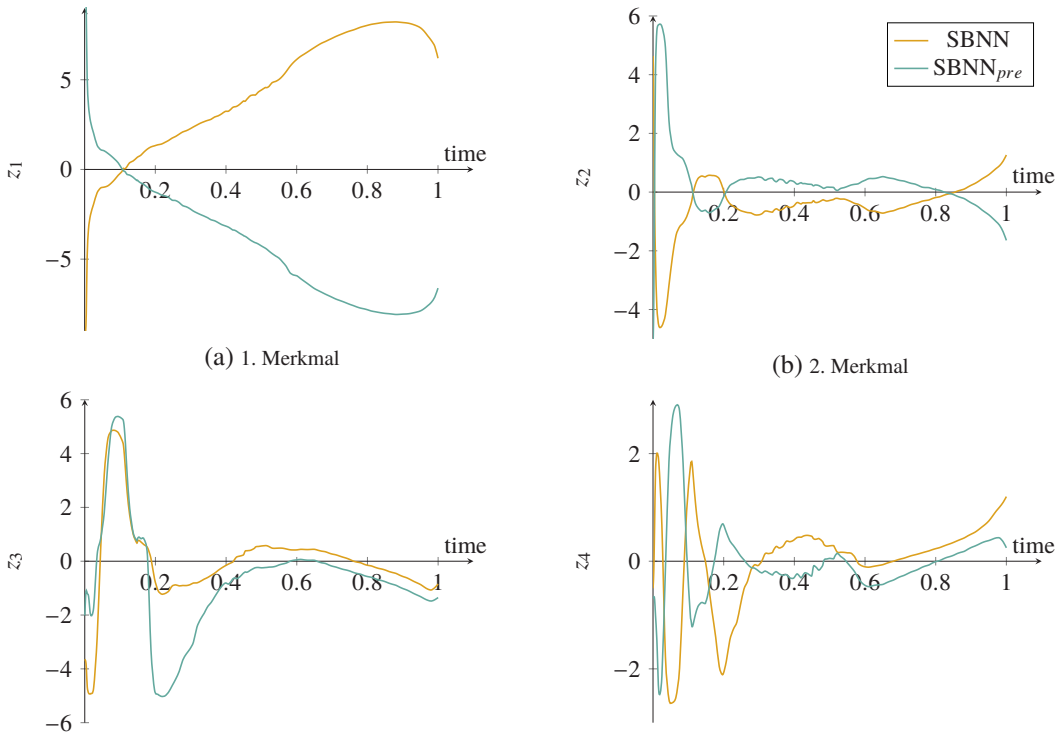


Abbildung 7.16: 2D-Tiefziehmodell (von Mises Spannung): Vergleich zeitlicher Verlauf der ersten vier Merkmale für Training mit und ohne Pretraining und dreifacher Kreuzvalidierung.

Pretraining für die Rekonstruktion, wie auch für den Verlauf der Merkmale, für diesen Datensatz keine wesentliche Verbesserung erzielte.

Der E_{MSE} für das dreidimensionale Modell in Abbildung 7.10 zeigt jedoch, dass dort durch Pretraining eine Verbesserung um durchschnittlich 0,0017 erreicht wurde. Dies kann auf die steigende Anzahl an Neuronen und die damit steigende Komplexität der Dimensionreduktion zurückzuführen sein.

Zusammenfassung Für die Anwendungsfälle Wärmeleitung und zweidimensionales Tiefziehmodell erreichten die vorgestellten sequentiellen Bottleneck Netze für Zeitreihen bessere Ergebnisse als die PCA und NLPCA. Für das dreidimensionale Tiefziehmodell wurden aufgrund der geringeren Anzahl an Stichproben im Vergleich zur Anzahl an Variablen nur das sequentielle Bottleneck Netz ohne die Zeitreihenerweiterung angewendet. Hier zeigt das SBNN mit Pretraining ein besseres Ergebnis als die NLPCA und für die ersten drei Merkmale auch ein besseres als die PCA. Es zeigt sich auch, dass es der NLPCA vor allem bei steigender Anzahl an Merkmalen schwer fällt, nach Varianz sortierte Merkmale zu ermitteln.

Das erste Merkmal der SBNN ähnelt im zeitlichen Verlauf in allen Anwendungsfälle der PCA. Daraus lässt sich schließen, dass das erste Merkmal (Richtung der höchsten Varianz der Daten) annähernd linear ist. Abschließend lässt sich sagen, dass ein geringerer Testfehler zwischen ursprünglichen Zustandsvariablen und rekonstruierten Zustandsvariablen nicht zwingend auf eine gute Lösung für die extrahierten Merkmale schließen lässt. Es hat sich gezeigt, dass die Verwendung der durch die PCA bestimmten Anzahl Neuronen in der verdeckten Schicht ein guter Anhaltspunkt ist, um Überanpassung in den Merkmalen zu vermeiden (d. h. die Komplexität gering zu halten). Damit die Merkmale eine allgemeine Darstellung der Zustandsvariablen liefern, sollte die Anzahl der Stichproben immer höher als die Anzahl an Zustandsvariablen sein.

7.2 Identifikation des reduzierten Prozessmodells

Bei der Identifikation des reduzierten Prozessmodells wurden für jede Beispielanwendung die extrahierten Merkmale verwendet, um durch symbolische Regression Differenzial- bzw. Differenzengleichungen und deren Parameter zu erlernen. Die Struktur der Gleichungen sind durch die Gleichungen 5.12 bis 5.16 vorgegeben. Da durch die Dimensionsreduktion möglichst unabhängige Merkmale ermittelt wurden, wurden bei der Durchführung der symbolischen Regression die Zustandsänderungen des jeweiligen Merkmals unabhängig von den anderen Merkmalen ermittelt. Für jede Strukturvorgabe wurde eine Vielzahl von Gleichungen unterschiedlicher Komplexität ermittelt. Zum Vergleich der Ergebnisse für die unterschiedlichen Strukturvorgaben wird jeweils ein Ergebnis ausgesucht, das möglichst ein Bestimmtheitsmaß von $R^2 \geq 0{,}90$ (Gleichung 7.2) hat, jedoch dabei eine möglichst geringe Komplexität aufweist. Die Komplexität der Gleichungen sollte so gering sein, dass die echtzeitfähige Ausführung möglich ist. Die zur Verfügung stehenden Operationen werden je nach Anwendung und Strukturvorgabe bestimmt. Die Merkmalsextraktion ist auf normierten Daten erfolgt; die resultierenden Merkmale sind dadurch nicht zwingend normiert, werden jedoch für die symbolische Regression nicht weiter normiert. Die Observablen für die Bestimmung des Messmodells sowie die Prozessparameter

Tabelle 7.1: Wärmeleitung: Fehler $(1-R^2)$ und Komplexität der gewählten Ergebnisse der symbolischen Regression für unterschiedliche Strukturvorgaben.

Struktur	**Fehler**	**Komplexität**
Merkmal z_1		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,0257	53
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,0900	88
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,3128	68
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0003	9
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,6951	33
Merkmal z_2		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,0235	37
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,1187	152
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,1380	188
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0012	15
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,2200	112

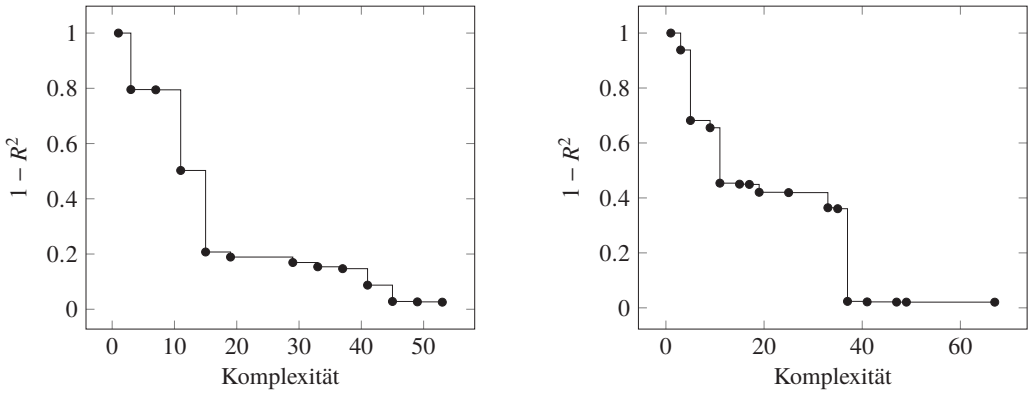
werden durch Gleichung (5.1) normiert. Die symbolische Regression wurde mit Eureqa® und auf einem 4-Kern Intel®Core™ i7-3820 Prozessor mit 3,60 GHz und 16 GB DDR3 RAM durchgeführt. Alle Ergebnisse sind aus Platzgründen auf mindestens zwei Nachkommastellen gerundet.

Approximationsgüte Beim Erlernen des reduzierten Prozessmodells und des Messmodells durch symbolische Regression wird zur Optimierung das Bestimmtheitsmaß R^2 verwendet:

$$R^2 = 1 - \frac{\sum_{i=1}^{s} \left(x^{(i)} - \hat{x}^{(i)}\right)^2}{\sum_{i=1}^{s} \left(x^{(i)} - \text{mean}(x)\right)^2}. \tag{7.2}$$

Der Quotient entspricht dem Verhältnis zwischen Variation des Fehlers und Variation der Daten. Das Bestimmtheitsmaß R^2 misst somit, welcher Anteil der totalen Varianz der Daten durch die geschätzten Daten abgedeckt wird. Ist der Fehler zwischen wahren Daten und geschätzten Daten 0, so ist $R^2 = 1$. Zu beachten ist, dass das Bestimmtheitsmaß für lineare Regression konzipiert ist. Die Wahl, welches gefundene Prozess- und Messmodell für das Zustandstracking verwendet wird, entscheidet sich jedoch nicht nur durch einen geringen Fehler bzw. möglichst hohes R^2, sondern auch die Komplexität der erlernten Funktionen sollte möglichst gering sein.

Wärmeleitung. Tabelle 7.1 zeigt die Komplexität und Fehlerwerte $(1 - R^2)$ der ausgewählten Ergebnisse für die jeweiligen Strukturvorgaben am Beispiel der extrahierten Merkmale der Temperatur. Für die direkte Ermittlung des nachfolgenden Zustandsmerkmals $z(t_{j+1})$ durch $f^{(N)}$ wurden die Operationen Addition, Subtraktion, Multiplikation sowie Sinus und Cosinus zugelassen. Für die restlichen Differenzialgleichungen zur Berechnung der Zustandsänderung



(a) 1. Merkmal. (b) 2. Merkmal.

Abbildung 7.17: Wärmeleitung: Pareto-Front für $f^{(E)}$.

wurden zusätzlich Exponentialterme hinzugefügt. Aus der Tabelle 7.1 ist ersichtlich, dass mit $f^{(V)}$ und $f^{(D)}$ weder für das erste noch für das zweite Merkmal eine Lösung gefunden wurde, die mehr als 90% der Varianz der Daten abdeckt, sprich einen Fehlerwert von unter 0,10 besitzt. $f^{(Z)}$ erreichte nur für das erste Merkmal das Ziel. Für die weitere Evaluation des Zustandstrackers wurden daher $f^{(E)}$ und $f^{(N)}$ weiter betrachtet. Abbildung 7.17 zeigt die Pareto-Front für die gesamten Ergebnisse von $f^{(E)}$ für die ersten beiden Merkmale nach einer Berechnung von 1.018.238 bzw. 1.252.802 Generationen und einer Berechnungszeit von 32 Stunden. Die für die weitere Betrachtung der Zustandsverfolgung gewählte Funktion für die Merkmalsänderung $D(z_i, t) = f^{(E)}(t, u, z_i(t, u))$ mit der Komplexität 53 ist:

$$D(z_1, t) = 21{,}01t^5 + 407{,}14t^4(\cos(t))^2 - 113{,}61(\cos(t))^2 - 404{,}18t^3\cos(t) + 111{,}88 \tag{7.3}$$

für das erste Merkmal. Die Merkmalsänderung für das zweite Merkmal mit einer Komplexität von 37 ist beschrieben durch:

$$D(z_2, t) = 727{,}51t^3 + 109{,}53t^2\sin(3{,}99t) - 1186{,}36t^2 + 484{,}93t - 72{,}50\sin(3{,}99t). \tag{7.4}$$

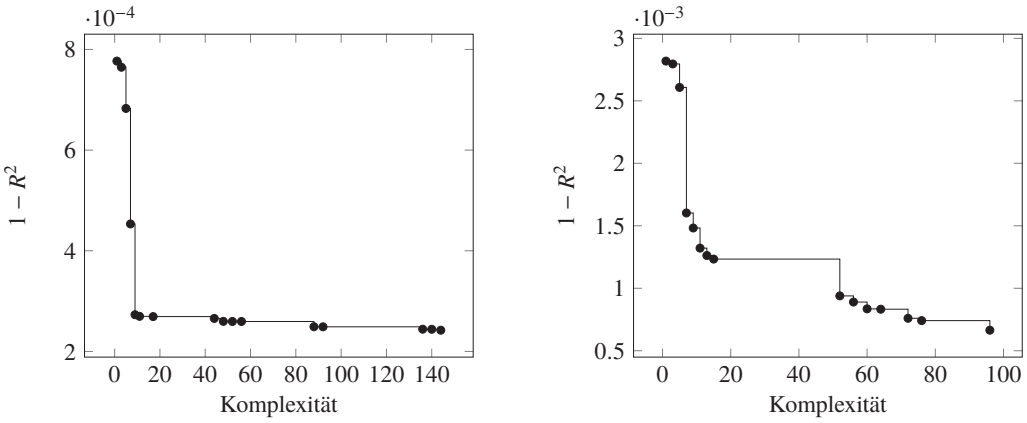
Dabei ist zu beachten, dass z_i bereits durch die Dimensionsreduktion in Abhängigkeit von Zeit t und Prozessparameter u steht. Die Variante mit Strukturvorgabe $f^{(N)}$ berechnet direkt das nachfolgende Zustandsmerkmal; für das erste Merkmal mit einer Komplexität von 9:

$$z_1(t_{j+1}, u) = z_1(t_j, u) - 3{,}08\Delta t\,(\cos(0{,}57 + t\,u)) \tag{7.5}$$

und für das zweite Merkmal mit einer Komplexität von 15:

$$z_2(t_{j+1}, u) = z_2(t_j, u) + 1{,}3\Delta t\,(u\sin(0{,}64t + t\,u) - 1{,}02)\,. \tag{7.6}$$

Abbildung 7.18 zeigt die Pareto-Front für die gesamten Ergebnisse von $f^{(N)}$ für die ersten beiden Merkmale nach einer Berechnung von 1.794.775 bzw. 2.809.546 Generationen und einer Berechnungszeit von 32 Stunden.



(a) 1. Merkmal.

(b) 2. Merkmal.

Abbildung 7.18: Wärmeleitung: Pareto-Front für $f^{(N)}$.

Für das Messmodell zur Vorhersage der Observable (Temperatur) an Position $p = 0$ ist folgende Gleichung ausgewählt wurden:

$$\hat{y}_1 = -0{,}42z_1^3 - 0{,}14z_2^3 + 0{,}20tz_2 - 0{,}34z_1 - 0{,}37z_2 - 0{,}37t + 1{,}95 \tag{7.7}$$

mit einem Fehler von 0,0104 und einer Komplexität von 37. Als weitere Observable wurde die Temperatur an Position $p = 0{,}6$ mit

$$\hat{y}_2 = 0{,}15z_2^3 - 0{,}15z_2^2 - 4{,}64z_1 + 0{,}20z_2 - 0{,}03t - 1{,}97\cos(1{,}63z_1) + 4{,}17 \tag{7.8}$$

vorhergesagt. Der Fehler der Vorhersage ist 0,0018 und die Komplexität 36. Für die Ermittlung des Messmodells wurden Addition, Subtraktion, Multiplikation, Sinus und Cosinus zugelassen.

Wie bereits bei der Reduktion der Temperatur als Zustandsgröße angemerkt, zeigt das erste Merkmal den mittleren charakteristischen zeitlichen Verlauf der Temperatur. Dabei wurden die 100 Positionen, an denen analytisch die Temperatur berechnet wurde, auf zwei Merkmale reduziert. Bei den Ergebnissen der symbolischen Regression für die Zustandsänderung des ersten und zweiten Merkmals ist die Summe trigonometrischer Ausdrücke wiederzufinden.

Tiefziehen im 2D-Modell mit von Mises Spannung als Zustandsgröße. Für das zweidimensionale Tiefziehmodell bzw. die 7 Merkmale der reduzierten von Mises Spannung wurde die symbolische Regression für die ersten drei Merkmale durchgeführt. Bei der Dimensionsreduktion wurde für drei Merkmale ein Testfehler von 0,0143 erreicht (MSE auf normierten Daten). Das jeweilige ausgewählte Ergebnis für jede Strukturvorgabe ist in Tabelle 7.2 gelistet. Die Berechnung wurde über 18 Stunden durchgeführt und mehr als 80.000 Generationen wurden entwickelt. Hier erwies sich $f^{(N)}$ als beste Lösung mit Hinblick auf die Fehlerwerte. $f^{(E)}$ erzielte den geringste Fehler für das zweite Merkmal, erreichte aber für das erste Merkmal nicht die gewünschte Varianzabdeckung von 90%. Als Zustandsänderung wird somit zusätzlich zu $f^{(N)}$ der zentrale Differenzenquotient $f^{(Z)}$ ausgewählt, da dieser nur für das dritte Merkmal, und somit das weniger relevante Merkmal, das Ziel nicht erreicht. Die für die weitere Betrachtung der Zustands-

Tabelle 7.2: 2D-Tiefziehmodell: Fehler ($1 - R^2$) und Komplexität der gewählten Ergebnisse der symbolischen Regression für die Merkmale der von Mises Spannung.

Struktur	**Fehler**	**Komplexität**
Merkmal z_1		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,2727	43
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,0147	17
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,0688	43
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0007	19
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,0496	45
Merkmal z_2		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,0155	39
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,0305	62
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,1263	37
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0511	45
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,0750	65
Merkmal z_3		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,1910	37
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,4160	46
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,5161	80
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0856	74
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,3468	39

verfolgung gewählten Funktionen für die Merkmalsänderung $\frac{z_i(t_{j+1}) - z_i(t_{j-1})}{\Delta t} = f^{(Z)}(\Delta t, u, \boldsymbol{z}(t_j))$ sind somit wie folgt definiert:

$$f^{(Z)}(\Delta t, u, z_1) = 4{,}20e^{z_1} - 17{,}28e^{0{,}83 z_1} \tag{7.9}$$

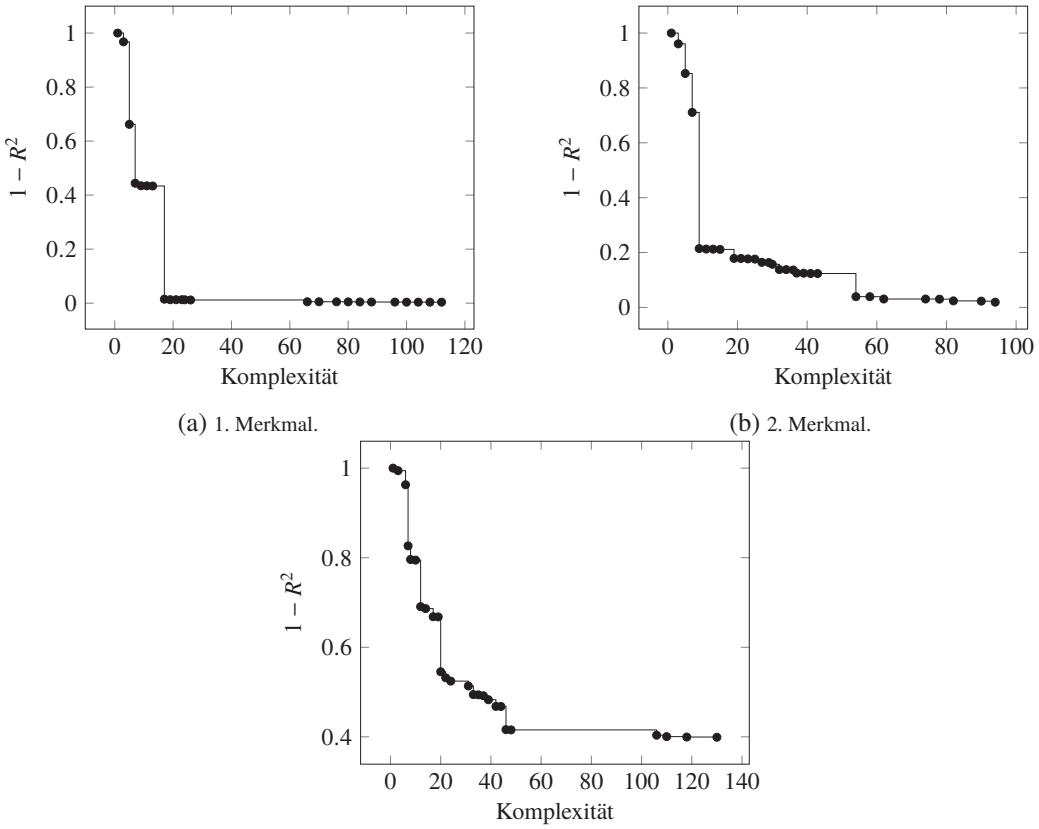
für das erste Merkmal mit einer Komplexität von 17 und

$$f^{(Z)}(\Delta t, u, z_2) = 408{,}36 + 1456{,}02e^{-362t} - 15{,}88 z_2 - 414{,}80 \cos(4340{,}04e^{-428{,}39t}) \tag{7.10}$$

für das zweite Merkmal mit einer Komplexität von 62, sowie für das dritte Merkmal mit einer Komplexität von 46

$$\begin{aligned} f^{(Z)}(\Delta t, u, z_3) = {} & 461{,}81e^{-190{,}51t} + 201{,}92t \sin(0{,}85 - 45{,}36t) \\ & -150{,}10 \sin(0{,}85 - 45{,}36t) - 6{,}68 z_3^2 \sin(-43{,}08t). \end{aligned} \tag{7.11}$$

z_i ist dabei bereits durch die Dimensionsreduktion von Zeit t und Prozessparameter u abhängig.



(a) 1. Merkmal.

(b) 2. Merkmal.

(c) 3. Merkmal.

Abbildung 7.19: 2D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(Z)}$.

Abbildung 7.19 zeigt die Pareto-Front für alle Ergebnisse für $f^{(Z)}$. Dabei ist zu sehen, dass für die ersten beiden Merkmale bereits bei einer Komplexität von 20 ein Fehlerwert von unter 0,2 erreicht werden konnte. Zur Verfeinerung der Ergebnisse steigt die Komplexität an.

Die Variante mit Strukturvorgabe $f^{(N)}$ berechnet direkt das nachfolgende Zustandsmerkmal. Für das erste Merkmal gilt:

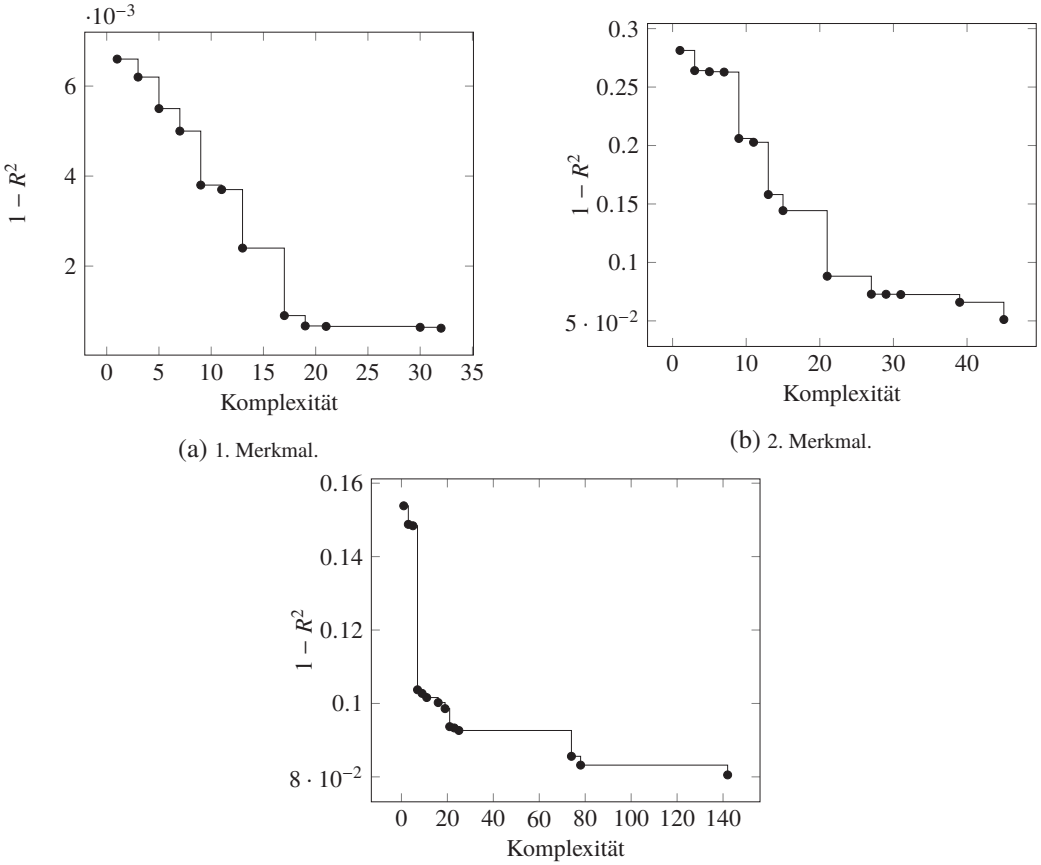
$$z_1(t_{j+1}, u) = z_1(t_j, u) + \Delta t \, (0{,}74 z_1 e^{z_1} - 6{,}16 e^{z_1} - 5{,}67) \tag{7.12}$$

mit einer Komplexität von 18 sowie für das zweite und dritte Merkmal mit einer Komplexität von 45 bzw. 74:

$$z_2(t_{j+1}, u) = z_2(t_j, u) + \Delta t \left(48{,}16 t z_2 + 65 \cdot 10^{10} t^5 e^{(41{,}59t - 65 \cdot 10^{10} t^7)} - 32{,}09 z_2\right) \tag{7.13}$$

$$\begin{aligned} z_3(t_{j+1}, u) = z_3(t_j, u) + \Delta t (2{,}07 \tan(142{,}40t) + 1{,}19 z_3 \tan(4{,}77 \\ + 320{,}92 \cos(\tan(142{,}40t))) - z_3 - 11{,}50. \end{aligned} \tag{7.14}$$

Abbildung 7.20 zeigt deutlich, dass für $f^{(N)}$ im Vergleich zu $f^{(Z)}$ in Abbildung 7.19 bereits bei geringer Komplexität auch ein geringer Fehler erzielt werden konnte.



(a) 1. Merkmal.

(b) 2. Merkmal.

(c) 3. Merkmal.

Abbildung 7.20: 2D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(N)}$.

Drei Observable wurden ausgesucht, anhand derer das System beobachtet werden soll. Dafür wurden jeweils Gleichungen gesucht, die die Zustandsmerkmale auf die Observablen abbilden. Für die Verschiebung des Stempels in y-Richtung (Ziehrichtung) als Observable in Abhängigkeit der Zustandsmerkmale wurde mit Hilfe der symbolischen Regression folgende Gleichung gefunden:

$$\hat{y}_1 = -0{,}002z_1 + 0{,}001z_2 - 3{,}43t + 1{,}57 \tag{7.15}$$

mit einem Fehler von 0,00009 und einer Komplexität von 13, dabei wurden Addition, Subtraktion, Multiplikation und Division als Operationen zugelassen. Die gleichen Operationen wurden für die Reaktionskraft des Stempels und der Matrize als Observable in Abhängigkeit der Zustandsmerkmale verwendet. Daraus resultierte folgende Gleichung für die Reaktionskraft des Stempels:

$$\hat{y}_2 = 0{,}38t\cos(z_1) - 17{,}78\sin(t) + 11{,}98t + 2{,}00, \tag{7.16}$$

mit einem Fehler von 0,0029 und einer Komplexität von 21. Als letzte Observable wurde die Reaktionskraft der Matrize gewählt:

$$\hat{y}_3 = 6{,}45t - 0{,}49tz_1 - 7{,}11t^2 - 0{,}26\sin(9{,}64t) - 1{,}82. \tag{7.17}$$

Tabelle 7.3: 2D-Tiefziehmodell: Fehler ($1 - R^2$) und Komplexität der gewählten Ergebnisse der sym. Regression für die Merkmale der Dehnung in Ziehrichtung.

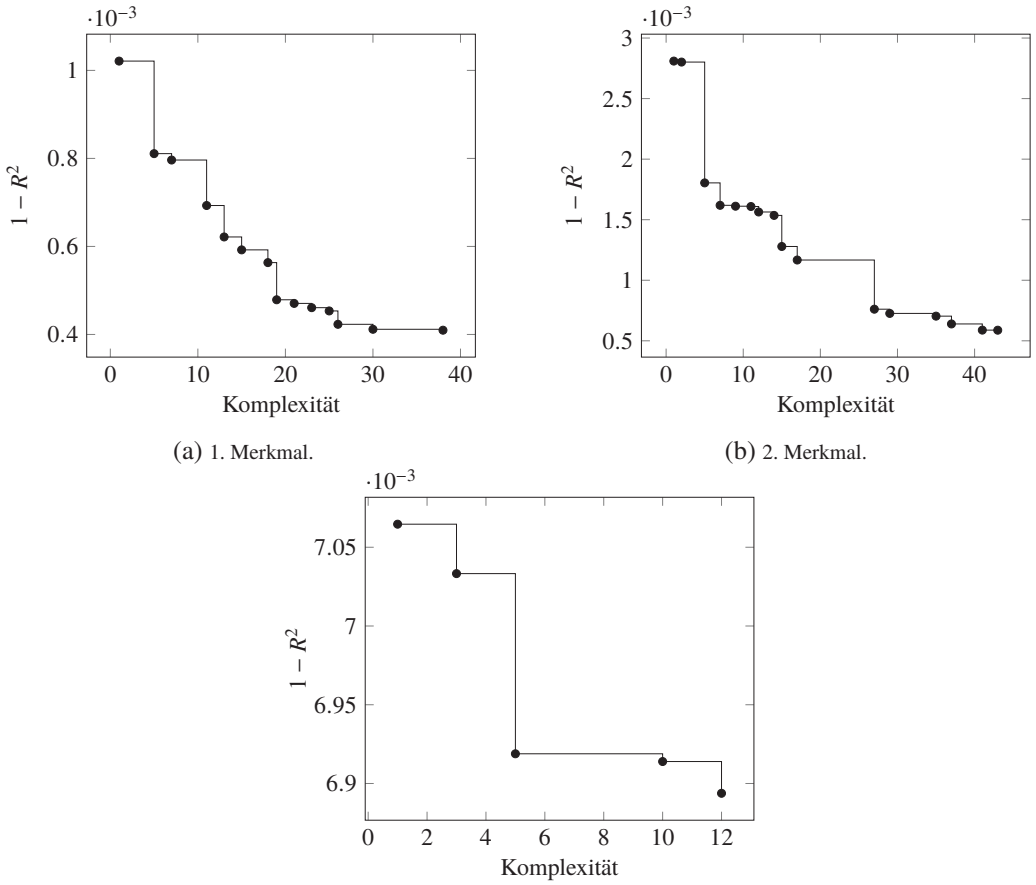
Struktur	**Fehler**	**Komplexität**
Merkmal z_1		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,2068	45
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,2560	94
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,4105	144
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0008	5
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,3500	39
Merkmal z_2		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,1350	65
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,1428	40
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,2358	49
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0018	5
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,0971	48
Merkmal z_3		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,2753	180
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,3920	57
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,4879	178
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0067	5
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,4331	84

Die Gleichung hat einen Fehler von 0,0277 und eine Komplexität von 26. Alle drei Observablen zeigten keine Abhängigkeit vom dritten Merkmal z_3.

Tiefziehen im 2D-Modell mit Dehnung als Zustandsgröße. Tabelle 7.3 zeigt die Ergebnisse für alle Strukturvorgaben, die beim Erlernen von Differenzialgleichungen mittels symbolischer Regression gewonnen werden konnten. Die erlernten Differenzialgleichungen, die die Merkmalsänderungen der reduzierten Dehnungswerte repräsentieren sollen, erreichen für das erste Merkmal keinen Fehler kleiner als 0,2. Das Lernverfahren wurde 158 Stunden durchgeführt bis es manuell unterbrochen wurde, da keine Verbesserung der Ergebnisse abzusehen war. Allein die Strukturvorgabe $f^{(N)}$, die direkt das nachfolgende Zustandsmerkmal durch Gleichung 5.15 ermittelt, konnte geringe Komplexität bei ebenfalls geringem Fehler erzielen.

Aus diesem Grund wurden für die weitere Betrachtung nur die Ergebnisse für $f^{(N)}$ einbezogen. Bei einer Komplexität von jeweils 5 wurden für die ersten drei Merkmale folgende Gleichungen erlernt:

$$\begin{aligned} z_1(t_{j+1},u) &= z_1(t_j,u) - \Delta t \left(z_1(t_j,u)^2 - 3{,}58\right) \\ z_2(t_{j+1},u) &= z_2(t_j,u) + \Delta t \left(-z_2(t_j,u)^2 + 64{,}61\right) \\ z_3(t_{j+1},u) &= z_3(t_j,u) + \Delta t \left(0{,}28u - 22{,}65\right). \end{aligned} \tag{7.18}$$



(a) 1. Merkmal.

(b) 2. Merkmal.

(c) 3. Merkmal.

Abbildung 7.21: 2D-Tiefziehmodell (Dehnung): Pareto-Front für $f^{(N)}$.

Die dazugehörige Pareto-Front ist in Abbildung 7.21 dargestellt. Sie zeigt, wie weit der Fehler bei steigender Komplexität noch verbessert werden kann.

Als Observable dienten ebenfalls die Verschiebung des Stempels in y-Richtung (Ziehrichtung), sowie die Reaktionskraft der Matrize und des Stempels. Für die Verschiebung des Stempels in Abhängigkeit der Zustandsmerkmale z ergab sich:

$$\hat{y}_1 = 1{,}61 - 0{,}02z_1 - 3{,}46t - 0{,}03t^2 \tag{7.19}$$

mit einem Fehler von 0,00008 und einer Komplexität von 15. Die erlernte Gleichung für die Reaktionskraft des Stempels lautet:

$$\hat{y}_2 = 0{,}38 + 0{,}37tz_2 - 0{,}21z_2 - 2{,}37t \tag{7.20}$$

mit einem Fehler von 0,0034 und einer Komplexität von 15; und die Reaktionskraft der Matrize lautet:

$$\hat{y}_3 = 4{,}82t + 0{,}08z_3 + 0{,}03z_2 - 1{,}67 - 0{,}37z_2t^3 \tag{7.21}$$

mit einem Fehler von 0,0211 und einer Komplexität von 23.

Tabelle 7.4: 3D-Tiefziehmodell: Fehler und Komplexität der gewählten Ergebnisse der symbolischen Regression der reduzierten von Mises Spannungswerte.

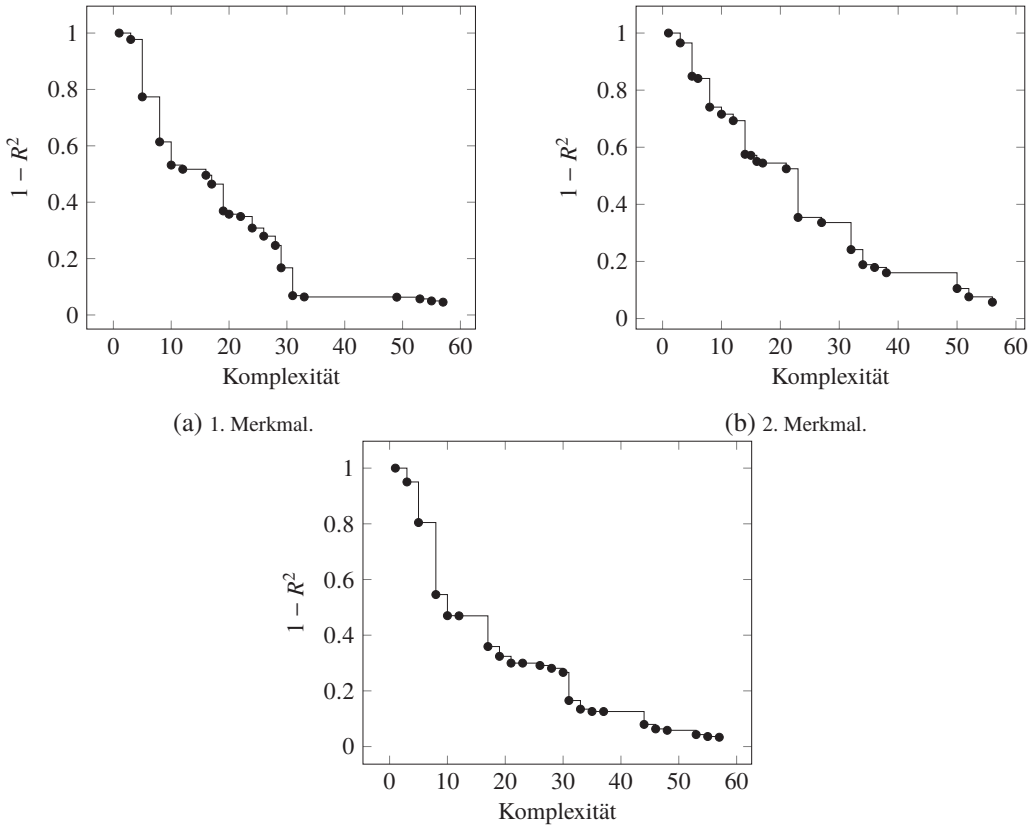
Struktur	**Fehler**	**Komplexität**
Merkmal z_1		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,0639	33
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,2400	68
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,3579	74
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0562	38
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,3853	43
Merkmal z_2		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,0574	56
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,5606	37
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,6789	45
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,1950	45
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,6335	58
Merkmal z_3		
$f^{(E)}$ (Eureqa® Ableitungsoperator)	0,0360	55
$f^{(Z)}$ (Zentraler Differenzenquotient)	0,5399	41
$f^{(V)}$ (Vorwärtsdifferenzenquotient)	0,6374	61
$f^{(N)}$ (Zustandsmerkmal bei t_{j+1})	0,0906	50
$f^{(D)}$ (Differenz Zustandsmerkmal bei t_{j-1} und t_{j+1})	0,6116	60

Tiefziehen im 3D-Modell mit von Mises Spannung als Zustandsgröße. Für die Merkmalsänderung der von Mises Spannung bzw. die Berechnung des Zustandsmerkmals zum Zeitpunkt t_{j+1} wurden folgende Operationen verwendet: Addition, Subtraktion, Multiplikation, Sinus und Cosinus, sowie Exponential. Die dabei entstandenen besten Ergebnisse werden in Tabelle 7.4 gelistet. Um diese Ergebnisse zu erreichen, wurde die symbolische Regression 112 Stunden durchgeführt und im Durchschnitt 3,4 Millionen Generationen berechnet. Es zeigt sich dabei, dass es für das komplexere dreidimensionale Modell wesentlich aufwendiger ist, ein reduziertes Modell mit geringem Fehler zu erlernen. Für die Ermittlung der Zustandsänderung erzielte der Eureqa® Ableitungsoperator die besten Ergebnisse (Strukturvorgabe $f^{(E)}$). Für das erste Merkmale mit einem Fehler 0,0639 und einer Komplexität 33:

$$D(z_1,t) = 75{,}46\cos(t)\cos(1{,}90t)\cos(0{,}18+9{,}75t) - 9{,}74 - 33{,}82\cos(1{,}70t). \tag{7.22}$$

Die Zustandsänderung für das zweite Merkmal mit einem Fehler von 0,0574 und einer Komplexität von 56 lautet:

$$\begin{aligned} D(z_2,t) = {} & 14{,}28 - 9{,}67t - 7{,}42t\sin(141468{,}43t) - 9{,}32\sin(5{,}68+141472t) \\ & -9{,}36\sin(5{,}46-165{,}68t) - 10.43\sin(141468{,}43t)\sin(5{,}68+141472t)t). \end{aligned} \tag{7.23}$$



(a) 1. Merkmal.

(b) 2. Merkmal.

(c) 3. Merkmal.

Abbildung 7.22: 3D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(E)}$.

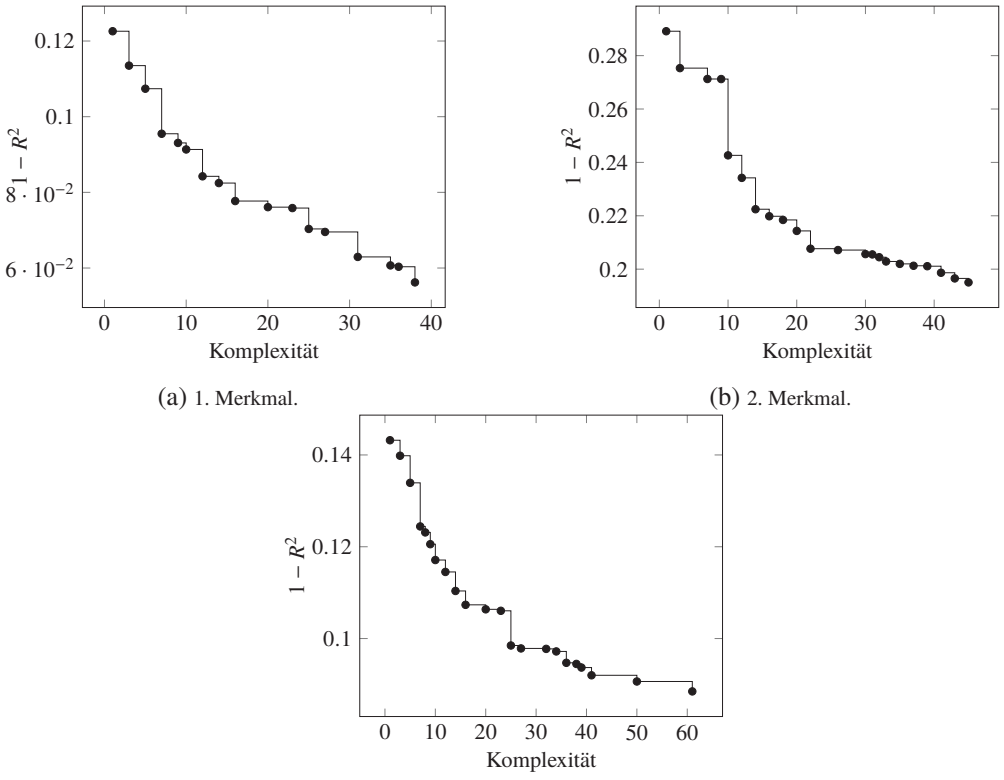
Die Zustandsänderung für das dritte Merkmal erreichte den geringsten Fehler mit 0,0360 und einer Komplexität von 55:

$$\begin{aligned} D(z_3,t) = 28{,}64 + 10{,}69t^2 + 59{,}76t\cos(8{,}53t) + 3{,}86t\cos(t^2) - 40{,}76t \\ -42{,}79\cos(8{,}53t) - 16{,}54t^2\cos(8{,}53t). \end{aligned} \tag{7.24}$$

Die Pareto-Fronten für alle drei Merkmalsänderungen sind in Abbildung 7.22 dargestellt. Sie zeigen, dass auch keine weiteren Ergebnisse mit höherer Komplexität, aber geringerem Fehler gefunden werden konnte. Ausnahme hierbei ist das dritte Merkmal, hier konnten zwar noch Lösungen mit einer höheren Komplexität als 36 gefunden werden, aber mit hohen Einbußungen in der Komplexität im Vergleich zur Fehlerreduzierung.

Auch in diesem Anwendungsbeispiel erzielte die direkte Berechnung des Zustandsmerkmals zum Zeitpunkt t_{j+1} gute Ergebnisse, nur das zweite Merkmal konnte das Ziel von 90% Varianzabdeckung nicht erreichen. Für das erste Merkmal ergibt sich:

$$\begin{aligned} z_1(t_{j+1},u) = s_1 + \Delta t(183{,}41 + 478{,}55t^2 + 26{,}05\cos(198375{,}06t) \\ -7{,}58s_1 - 647{,}86t - 0{,}31tu - 89{,}80t^3) \end{aligned} \tag{7.25}$$



(a) 1. Merkmal.

(b) 2. Merkmal.

(c) 3. Merkmal.

Abbildung 7.23: 3D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(N)}$.

mit einem Fehler von 0,0562 und einer Komplexität von 38. Dabei ist s_i definiert als $z_i(t_j,u)$. Das Ergebnis für das zweite Merkmal vermerkt einen Fehler von 0,1950 und eine Komplexität von 45:

$$\begin{aligned} z_2(t_{j+1},u) = s_2 + \Delta t(18t + ts_2^2\cos(0{,}88t^2 - t) - 5{,}08 - 4{,}26ts_2 \\ -8{,}23t^2 - 1{,}89s_2\cos(5{,}61 - s_2)). \end{aligned} \tag{7.26}$$

Für das dritte Merkmal wurde ein Fehler von 0,0906 und eine Komplexität 50 erreicht:

$$\begin{aligned} z_3(t_{j+1},u) = s_3 + \Delta t(97{,}69t + 142{,}82\cos(t) + 2{,}10s_3\cos(7{,}51t) - 147{,}71 \\ -2{,}86s_3 - 5{,}29\cos(7{,}51t) - 24{,}82\cos(t)\cos(7{,}51t)). \end{aligned} \tag{7.27}$$

Alle Ergebnisse von $f^{(N)}$ für das dreidimensionale Modell sind in der Pareto-Front in Abbildung 7.23 dargestellt. Auch hier wurden keine komplexeren Gleichungen für die drei Merkmale gefunden als die bereits ausgewählten.

Für die Observablen in Abhängigkeit der drei Merkmale wurde die Verschiebung des Stempels in Ziehrichtung sowie die Reaktionskraft von Stempel und Matrize ausgewählt. Dabei ergab sich für die Verschiebung des Stempels:

$$\hat{y}_1 = 1{,}69 - 0{,}0006z_2 - 0{,}0007z_1 - 1{,}21t \tag{7.28}$$

Tabelle 7.5: Vergleich E_{MSE} für Messmodell mittels symbolischer Regression (SR) und künstlichem neuronalen Netz (KNN).

Observable	**SR**	**KNN**
Wärmeleitung		
o_1	0,0964	0,0367
o_2	0,1065	0,0385
2D von Mises Spannung		
o_1	0,0210	0,0029
o_2	0,0656	0,0020
o_3	0,0591	0,0228
2D plastische Dehnung		
o_1	0,0180	0,0011
o_2	0,0317	0,0014
o_3	0,0628	0,0217

mit einem Fehler von 0,0002 und einer Komplexität von 13. Das Ergebnis für die Reaktionskraft des Stempels ist von den ersten beiden Merkmalen abhängig:

$$\begin{aligned}\hat{y}_2 = {} & 0{,}06 + 0{,}06 z_1 + 0{,}31 \sin(0{,}68 + 903{,}12 t) \\ & + \cos(900{,}23 t) \cos(0{,}09 z_3) \cos(0{,}66 + 0{,}05 z_1)\end{aligned} \tag{7.29}$$

mit einem Fehler von 0,0155 und einer Komplexität von 39. Das Ergebnis für die Reaktionskraft in Abhängigkeit der Merkmale lautet:

$$\begin{aligned}\hat{y}_3 = {} & 0{,}0004 e^{-0{,}12 z_1} + 0{,}00001 z_2 t^2 - 0{,}06 - 0{,}0002 t \\ & - 0{,}0002 t \cos(6{,}10 + 0{,}11 z_1 + t^2)\end{aligned} \tag{7.30}$$

mit einem Fehler von 0,1362 und einer Komplexität von 40.

Symbolische Regression und künstliches neuronales Netz für das Messmodell im Vergleich. Ist keine Interpretierbarkeit der Struktur und Parameter des Messmodells nötig, so kann auch ein neuronales Netz für die Abbildung der Zustandsmerkmale auf die Observablen zum Einsatz kommen. Hierfür reicht ein dreischichtiges Netz mit den Zustandsmerkmalen als Eingabe, den Observablen als Ausgabe und einer verdeckten Schicht. Kapitel 5.2.2 erläutert den konkreten Aufbau dieses Netzes. Tabelle 7.5 zeigt den Vergleich des Rekonstruktionsfehlers E_{MSE} bei den Testdaten für alle Anwendungsfälle, die durch symbolische Regression bzw. durch das neuronale Netz erreicht wurden. Das künstliche neuronale Netz erzielt für jede Observable bessere Ergebnisse als die symbolische Regression. Die Überprüfung auf Beobachtbarkeit des Prozessmodells mittels des Messmodells, wie in Kapitel 5.3.1 beschrieben, ist jedoch mit einem Messmodell basierend auf KNNs nicht möglich. Weiterhin war es für die gegebenen Anwendungsfälle nicht möglich, Messmodelle ohne explizite Zeitabhängigkeiten zu ermitteln;

ein Messmodell mittels eines KNN würde dies ermöglichen. Da die nachfolgend untersuchten nichtlinearen Zustandschätzer auf einem Messmodell basieren, dass durch symbolische Gleichungen beschrieben wird, wird der Ansatz, das Messmodell mittels KNNs zu beschreiben, hier nicht weiter verfolgt.

7.3 Beobachtung des Zustandes

Zur Beobachtung des Zustandes anhand des erlernten Prozess- und Messmodells kommen das erweiterte Kalman-Filter (EKF) und das unscented Kalman-Filter (UKF), aus Kapitel 3.2 und 3.3, zum Einsatz. Für das EKF wurden zwei Varianten implementiert: Zum einen wird von einem dynamischen Prozessmodell und einem diskreten Messmodell ausgegangen, und zum anderen von jeweils diskretem Prozess- und Messmodell. Das heißt, es kommt das Prozessmodell der Form $f^{(E)}$ bzw. $f^{(Z)}$ und das der Form $f^{(N)}$ zum Einsatz. Die Implementierung des UKFs geht von einem Prozessmodell der Form $f^{(N)}$ aus.

Bevor der Zustandstracker ausgeführt wurde, wurde einmalig überprüft, ob das Prozessmodell bzw. die Zustandsmerkmale $\boldsymbol{z}$ anhand des Messmodells bzw. den Observablen $\boldsymbol{y}$ beobachtbar sind. Bei der Ausführung des Zustandstrackers wurde die Zeit gemessen. Die Observablen lagen für die Evaluation des Zustandstrackers durch die numerische Simulation rauschfrei vor. Die vorhergesagten Zustandsmerkmale wurden für die Evaluation bei jeden Zeitschritt gespeichert, um sie anschließend in den ursprünglichen Zustandsraum zurück zu transformieren. Diese Werte wurden dann mit den Werten aus der Finite-Elemente-Simulation verglichen, um die Frage zu beantworten, ob es möglich ist, mit einem erlernten reduzierten Zustandstracker den Zustand des Bauteils annähernd so präzise wie mit numerischer Simulation vorherzusagen. Die Prüfung der Beobachtbarkeit sowie die Zustandsverfolgung wurde mit MATLAB® implementiert und die Berechnungen wurden mit einem 4-Kern Intel®Core™i7-3820 Prozessor bei 3,60 GHz durchgeführt.

Approximationsgüte. Für die Bewertung des gesamten Verfahrens wurden nach Ablauf des Zustandstrackings des Prozesses die vorhergesagten Zustandsmerkmale in den ursprünglichen Zustandsraum zurücktransformiert, da zu Abweichungen im Merkmalsraum keine Aussagen getroffen werden können. Weiterhin ist zu beachten, dass der Mean Squared Error (MSE) sich zwar zur Fehleroptimierung, jedoch nicht zur Bewertung des gesamten Verfahrens eignet, da er keinen Aufschluss darüber gibt, ob der vorhergesagte Zustand in einem „grünen Bereich“ um die Referenzlösung (d. h. die der Simulation) liegt. Das liegt darin begründet, dass gleiche Fehlerwerte in unterschiedlichen Elementen des FE-Modells unterschiedliche Gewichtung haben, z. B. wird bei Elementen mit hoher Varianz über den zeitlichen Verlauf in der Regel ein größerer Fehler ermittelt als bei Elementen mit geringerer Varianz über die Zeit. Das heißt, es wird ein Fehlermaß benötigt, das angibt, in wie vielen Elementen die Zustandsgröße (z. B. von Mises Spannung) um wie viel verfehlt wurde. Hierfür wird ein globaler Fehler, der akzeptabel ist, durch Experten vorgegeben und dann für jedes finite Element und jeden Zeitschritt über die ganze Simulation hinweg gemessen, wie oft dieser Fehler überschritten wurde. So wird jeder Bereich des Bauteils gesondert betrachtet, um eine allgemeinere Aussage darüber zu treffen, wie gut das online Zustandstracking mit den datengetriebenen Modellen den numerisch simulierten

Prozess reproduziert. Als Vergleich zu diesem Fehlermaß wird für das gesamte Bauteil sowie alle Zeitverläufe und jede Parametervariation der Root Mean Squared Error (RMSE) berechnet:

$$E_{\mathrm{RMSE}} = \sqrt{E_{\mathrm{MSE}}} = \sqrt{\frac{1}{s}\sum_{i=1}^{s}\left(\boldsymbol{x}^{(i)} - \hat{\boldsymbol{x}}^{(i)}\right)^2}. \tag{7.31}$$

Da die Zustandsvariablen in der gleichen Maßeinheit vorliegen, kann der RMSE auch in dieser Maßeinheit betrachtet werden.

Ergebnisse. Tabelle 7.6 listet die Ergebnisse aller Anwendungsbeispiele unter Verwendung des jeweiligen Filters; dabei ist „S.wert$_{\mathrm{max}}$“ die maximale aufgetretene Überschreitung des Fehlerschwellwertes und „S.wert$_{\%}$“ beschreibt, welcher Prozentsatz der gesamten Daten den Schwellwert überschritten hat. Der Schwellwert beträgt hier jeweils weniger als 10% des gesamten Wertebereichs. Der Zustandsverfolger wurde im ursprünglichen Zustandsraum evaluiert, d. h. alle Werte sind in der Einheit der Zustandsgröße zu betrachten. Zu beachten ist, dass der RMSE nur innerhalb einer Anwendung verglichen werden kann, da er die gleiche Maßeinheit wie die Zustandsvariablen der Anwendung hat.

Wärmeleitung. Die Dimensionsreduktion erreichte mit den zwei Merkmalen einen E_{MSE} von 0,0087 und eine Varianzabdeckung von 99%. Darauf aufbauend erzielte die symbolische Regression einen mittleren $1 - R^2$ Fehler von 0,0246 für das dynamische Modell und 0,0008 für das diskrete. Das dynamische Modell kam beim EKF$_{dyn}$ zum Einsatz und das diskrete beim EKF und UKF. Tabelle 7.6 zeigt, dass für das EKF die geringsten Schwellwertüberschreitungen aufgetreten sind. Die benötigte Zeit zur Berechnung der Zustandsverfolgung beträgt nur ein Drittel der tatsächlichen Prozessdauer. Abbildung 7.24 zeigt die Fehlerüberschreitung für Parameter $\omega = 6$ zu den Zeitpunkten 0,005, 0,5 und 1. Zu sehen ist, dass für den ersten Zeitschritt für mehr als die Hälfte der Elemente der Fehlerschwellwert 0,1 überschritten wurde. Es handelt sich dabei um die 55 Elemente auf der Seite der Wärmequelle. Für den Zeitpunkt 0,5 sind dann keine Fehlerüberschreitungen mehr festzustellen. Erst am Ende des Prozesses sind wieder auf Seite der Wärmequelle Fehlerüberschreitungen in 18 Elementen zu beobachten. Dass die Fehlerüberschreitungen auf Seiten der Wärmequelle auftauchen, liegt darin begründet, dass dort die höchste Varianz in den Temperaturwerten vorliegt.

Tiefziehen im 2D-Modell mit von Mises Spannung als Zustandsgröße. Das Prozessmodell für den zweidimensionalen Tiefziehprozess mit der von Mises Spannung als Zustandsgröße ist durch sein Messmodell (Gleichungen (7.15) bis (7.17)) nicht beobachtbar. Grund dafür ist, das keine der Gleichungen des Messmodells (mittels symbolischer Regression generiert) eine Abhängigkeit vom dritten Merkmal hat. Durch Entfernen des dritten Merkmals wird das Prozessmodell anhand der Observablen beobachtbar. Die Dimensionsreduktion erreichte mit zwei Merkmalen einen E_{MSE} von 0,0202 und eine Varianzabdeckung von 98%. Darauf aufbauend erzielte die symbolische Regression einen mittleren $1 - R^2$ Fehler von 0,0226 für das dynamische Modell und 0,0259 für das diskrete. Wenn der Zustandstracker nicht in der Lage war, einen relativen Fehler von weniger als 50% zwischen vorhergesagten Zustandsmerkmalen und

Tabelle 7.6: Ergebnisse der nichtlinearen Kalman Filter als Zustandstracker.

Wärmeleitung				
Wertebereich				[0; 1]
Schwellwert				0,1
Prozessdauer				1,0 s
	S.wert$_{max}$	**S.wert$_{\%}$**	**RMSE**	**Zeit**
EKF	0,3408	5,63%	0,0283	0,0339
EKF$_{dyn}$	0,6113	24,20%	0,0530	0,0405
UKF	0,8720	76,52%	0,2214	0,0344
$2D_{Mises}$				
Wertebereich				[0; 800]
Schwellwert				50MPa
Prozessdauer				1,0 s
	S.wert$_{max}$	**S.wert$_{\%}$**	**RMSE**	**Zeit**
EKF	631	29,67%	90,0098	0,0280
EKF$_{dyn}$	631	6,64%	77,4962	0,0276
UKF	631	25,62%	87,3171	0,0420
$2D_{Dehnung}$				
Wertebereich				[−0,3257; 0,2142]
Schwellwert				0,05
Prozessdauer				1,0 s
	S.wert$_{max}$	**S.wert$_{\%}$**	**RMSE**	**Zeit**
EKF	0,2535	22,09%	0,0403	0,0372
$3D_{Mises}$				
Wertebereich				[0,900]
Schwellwert				50MPa
Prozessdauer				2,8 s
	S.wert$_{max}$	**S.wert$_{\%}$**	**RMSE**	**Zeit**
EKF	509,56	47,00%	80,1797	0,0093
EKF$_{dyn}$	509,56	41,89%	78,2092	0,0075
UKF	509,56	34,61%	75,1438	0,0078

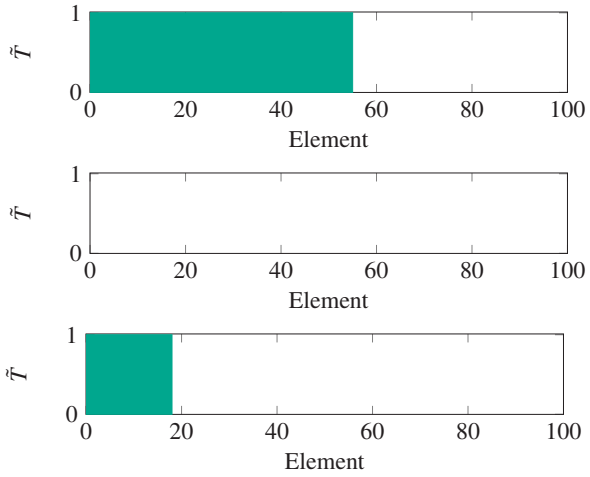


Abbildung 7.24: Wärmeleitung: Grün markiert die Überschreitungen des Fehlerschwellwertes für $\omega = 6$ zu den Zeitpunkten 0,005, 0,5 und 1.

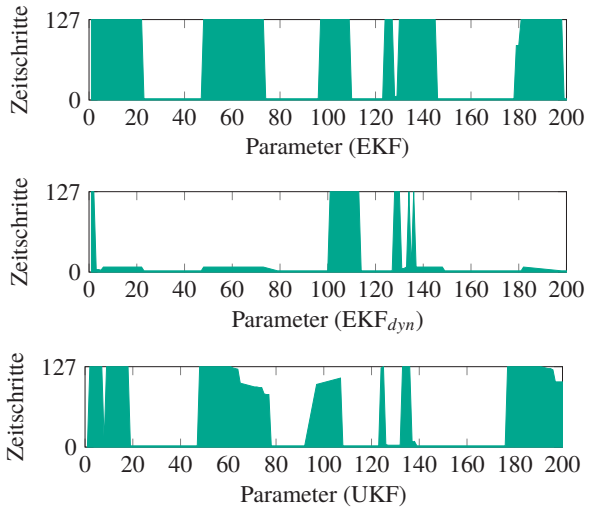


Abbildung 7.25: 2D-Tiefziehmodell (von Mises Spannung): Anzahl der Zeitschritte, die je Parameter durchlaufen wurden. Die Niederhalterkraft als Parameter variiert 200 mal äquidistant zwischen 70 und 100 kN.

ursprünglichen Zustandsmerkmalen (pro Zeitschritt) zu erreichen, wurde der Zustandstracker abgebrochen. Abbildung 7.25 zeigt im grünen Bereich die Anzahl Zeitschritte, die je Prozessparameter durchlaufen wurden. Es ist ersichtlich, dass beim dynamischen Modell nur für eine geringe Anzahl Parameter der Zustandstracker vollständig durchlaufen wurde. Abbildung 7.26 zeigt den Vergleich zwischen FE-Simulation und Zustandsverfolgung mittels dem erlernten reduzierten Modell und EKF für den Endzustand bei einer Niederhalterkraft von 70 kN. Tabelle 7.6 zeigt, dass EKF und UKF vergleichbar gute Ergebnisse erreichen konnten. Jedoch konnte das EKF für mehr Prozessparameter vollständig durchlaufen werden. Für fast 30% aller durchlaufenden Zeitschritte, Prozessparameter und FE-Elemente konnten Fehlerüberschreitungen registriert werden. Abbildung 7.27 zeigt die Bereiche der Fehlerüberschreitungen für den Endzustand.

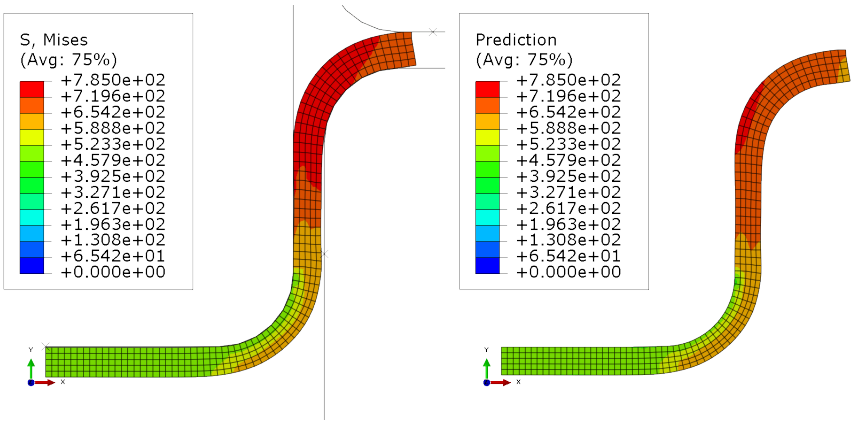


Abbildung 7.26: 2D-Tiefziehmodell (von Mises Spannung): Vergleich Endzustand in FE-Simulation (links) und nach Zustandsverfolgung mittels reduziertem Modell und EKF (rechts) für Niederhalterkraft 70 kN.

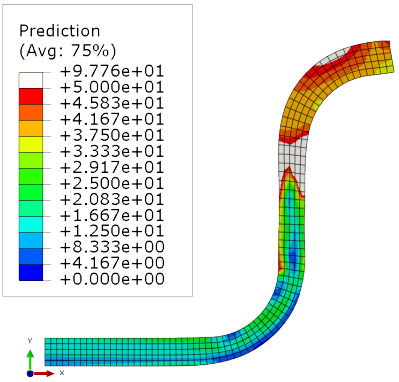


Abbildung 7.27: 2D-Tiefziehmodell (von Mises Spannung): Bereiche der Fehlerschwellwertüberschreitungen von mehr als 50 MPa (weiß) nach Zustandsverfolgung mittels reduziertem Modell und EKF für Niederhalterkraft 70 kN.

Tiefziehen im 2D-Modell mit plastischer Dehnung als Zustandsgröße. Die Untersuchungen auf Beobachtbarkeit haben gezeigt, dass die drei Zustandsmerkmale, die die plastische Dehnung des zweidimensionalen Modells repräsentieren, durch die drei Observablen beobachtbar sind. Die Dimensionsreduktion erreichte mit drei Merkmalen einen E_{MSE} von 0,0048 und eine Varianzabdeckung von 99%. Darauf aufbauend erzielte die symbolischen Regression einen mittleren $1 - R^2$ Fehler von 0,0031 für das diskrete Modell. Tabelle 7.6 zeigt die Ergebnisse, die erreicht werden konnten. Für das dynamische Modell, unter Verwendung des EKF$_{dyn}$, wurden mittels symbolischer Regression keine geeigneten Gleichungen gefunden. Für die Zustandsverfolgung mittels UKF konnten für keinen Parameter alle Zeitschritte durchlaufen werden, da Zustandsmerkmale und vorhergesagte Zustandsmerkmale zu weit divergieren. Abbildung 7.28 zeigt die Anzahl durchlaufener Zeitschritte für jeden Prozessparameter, die mit dem EKF erreicht wurde. Mittels dem EKF wurde für 22,09% aller durchlaufenden Zeitschritte, Elemente im FE-Modell und Parameter der Fehlerschwellwert überschritten. Abbildung 7.29 zeigt die Gegenüberstellung des Endzustandes der FE-Simulation und des Endzustandes, der durch den Zustandstracker mittels erlerntem Prozessmodell, welches lediglich drei Zustandsmerkmale be-

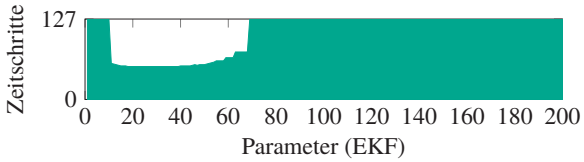


Abbildung 7.28: 2D-Tiefziehmodell (Dehnung): Anzahl der Zeitschritte, die je Parameter durchlaufen wurden. Die Niederhalterkraft als Parameter variiert 200 mal zwischen 70 und 100 kN.

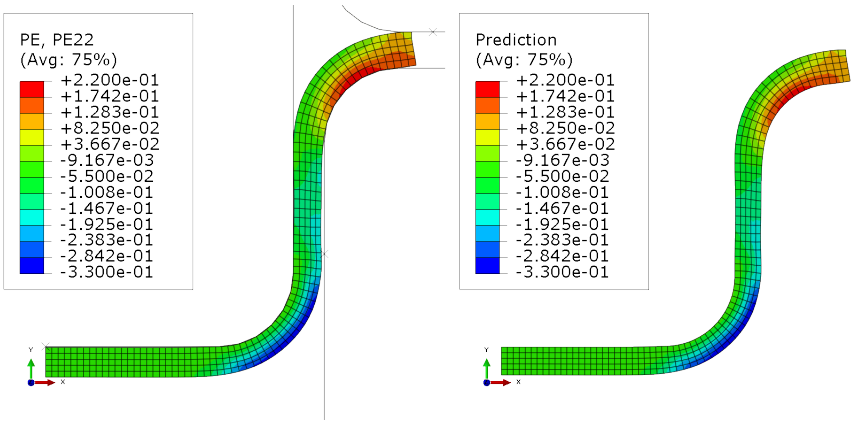


Abbildung 7.29: 2D-Tiefziehmodell (Dehnung): Vergleich Endzustand in FE-Simulation (links) und nach Zustandsverfolgung mittels reduziertem Modell und EKF (rechts) für Niederhalterkraft 70 kN.

sitzt anstelle von 400 Zustandsvariablen, erzielt wurde. Abbildung 7.30 zeigt, dass in keinem Bereich des Bauteils im Endzustand der Fehlerschwellwert überschritten wurde.

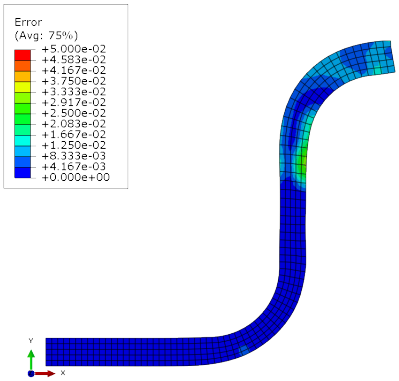


Abbildung 7.30: 2D-Tiefziehmodell (Dehnung): Fehlerwerte nach Zustandsverfolgung mittels reduziertem Modell und EKF für Niederhalterkraft 70 kN. Der Fehlerwert liegt für jedes Element unter dem Schwellwert von 0,05.

Tiefziehen im 3D-Modell mit von Mises Spannung als Zustandsgröße. Das reduzierte Prozessmodell für den dreidimensionalen Tiefziehprozess ist nicht mittels des Messmodells beobachtbar. Auch hier ist keine der drei Observablen in Gleichung (7.28) bis (7.30) abhängig vom dritten Merkmal. Durch die Entfernung der Gleichung für das dritte Merkmal aus dem Prozessmodell wird es beobachtbar. Die Dimensionsreduktion erreichte mit zwei Merkmalen einen

E_{MSE} von 0,1062 und eine Varianzabdeckung von 89%. Darauf aufbauend erzielte die symbolische Regression einen mittleren $1 - R^2$ Fehler von 0,1256 für das diskrete Modell und 0,0607 für das dynamische Modell. Tabelle 7.6 zeigt, dass im Wertebereich der von Mises Spannung von 0 bis 900 MPa mit dem EKF bei 47% aller Elemente und durchlaufenden Zeitschritte der Fehlerschwellwert von 50 MPa überschritten wurde. Für das EKF mit dynamischem Modell liegt die Fehlerüberschreitung bei 41,89% und beim UKF bei 34,61%. Wobei für den UKF für weniger Prozessparameter alle Zeitschritte durchlaufen werden konnten. Somit kann der Zustandstracker mit dem erlernten reduzierten Prozess- und Messmodell, bestehend aus zwei Gleichungen zur Beschreibung der zwei Zustandsmerkmale und drei Observablen zur Korrektur der zwei Zustandsmerkmale, 53% der 2223 Zustandsmerkmale und 100 Prozessparametern bis zu einem Fehler von 50 MPa rekonstruieren. Auch hier waren, ebenso wie beim zweidimensionalen Tiefziehprozess, nicht alle Zeitschritte für alle Parameter durch die nichtlinearen Kalman-Filter durchlaufbar, da an diesen Stellen wahre und geschätzte Zustandsmerkmale zu stark divergierten. Abbildung 7.31 zeigt für jeden Parameter, wie viele Zeitschritte für das jeweilige Kalman-Filter durchlaufen wurden.

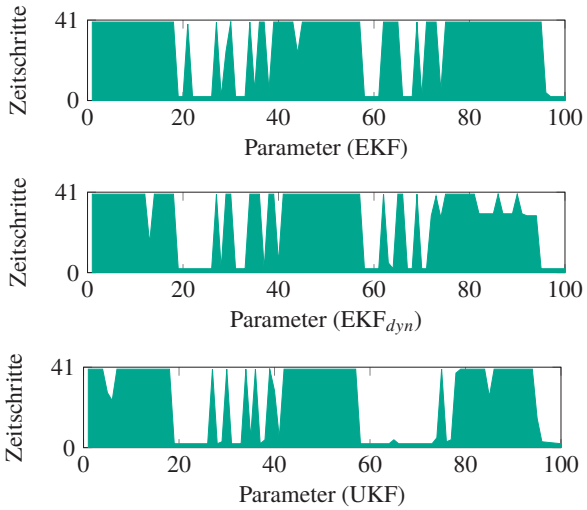


Abbildung 7.31: 3D-Tiefziehmodell (von Mises Spannung): Anzahl der Zeitschritte, die je Parameter durchlaufen wurden. Die Niederhalterkraft als Parameter variiert 100 mal zwischen 45 und 150 kN.

7.4 Zusammenfassung

Zielsetzung der Arbeit war es, eine Methode zur Zustandsverfolgung zu entwickeln, die aus aufwendig berechneten Zustandsvariablen (Daten) aus numerischen Prozesssimulationen ein onlinefähiges, dynamisches, nichtlineares Prozessmodell erlernt. Dieses Surrogat-Modell sollte den Zustand des Bauteils während des laufenden Prozesses vorhersagen. Ein ebenso aus Daten erlerntes Messmodell sollte dabei zur Korrektur von Mess- und Modellunsicherheiten dienen. Die für diese Aufgabe untersuchten Methoden der künstlichen neuronalen Netze und der symbolischen Regression haben dazu beigetragen, ein Gesamtkonzept zu entwickeln, dass aus Daten ein reduziertes sowie nichtlineares Prozess- und Messmodell erlernt, welches wiederum zum Einsatz bei bekannten nichtlinearen Zustandsschätzern wie dem erweiterten und unscented

Kalman-Filter kommen können. Die Ergebnisse zeigen, dass weniger komplexe Prozesse, die durch 400 Zustandsvariablen beschrieben werden, bei einer entsprechenden Anzahl an Stichproben auf weniger als sieben nichtlineare Merkmale reduziert werden können und dabei weiterhin 98% der Varianz der Daten repräsentieren. Bei komplexeren Prozessen, wie dem dreidimensionalen Tiefziehmodell, welches durch 8832 Zustandsvariablen dargestellt wird, benötigt es eine wesentlich höhere Anzahl an Stichproben, um auf eine ebenfalls so geringe Anzahl an Merkmalen wie beim zweidimensionalen Tiefziehmodell reduzieren zu können. Da der Aufwand zur Erstellung der Stichproben zu hoch ist, empfiehlt es sich, nur bestimmte Bereiche des Bauteils zu betrachten und diese zu reduzieren. Die symbolische Regression zum Erlernen von Differenzial- bzw. Differenzengleichungen hat gezeigt, dass ein zuverlässiges Ergebnis mit geringem Fehler nur mit der Strukturvorgabe $z_i(t_{j+1}) = z_i(t_j) + \Delta t \cdot f^{(N)}(u, \boldsymbol{z}(t_j))$ erreicht wurde. Die Performanz der anderen in Kapitel 5.2.1 vorgestellten Strukturvorgaben hing stark vom Anwendungsfall ab. Zur Überprüfung der Machbarkeit wurden für die Zustandsbeobachtung nur die ersten zwei bzw. drei Merkmale aus der Dimensionsreduktion durch symbolische Gleichungen, die mittels der symbolischen Regression erlernt wurden, verwendet. Die Observablen wurden dann mittels dieser drei Merkmale anhand des Messmodells beschrieben. Nach Prüfung, ob die Zustandsmerkmale anhand der gegebenen Observablen beobachtbar sind, wurden gegebenenfalls die nicht beobachtbaren Zustandsmerkmale aus dem Prozessmodell entfernt. Nach Anwendung der nichtlinearen Kalman-Filter für alle Prozessparameter über den gesamten Prozessablauf, konnte festgestellt werden, dass für einige Prozessparameter wahre und geschätzte Zustandsmerkmale zu stark divergieren. Dies kann Folge daraus sein, dass für das Erlernen des Prozessmodells mittels symbolischer Regression kein rekurrentes Lernverfahren existiert. Weiterhin könnte das Verwenden zusätzlicher Zustandsmerkmale, die durch gegebenenfalls zusätzliche Observablen beobachtbar sind, Abhilfe schaffen. Für die meisten Prozessparameter konnten die nichtlinearen Kalman-Filter jedoch gute Ergebnisse erzeugen und den reduzierten Zustand so verfolgen, dass der in den ursprünglichen Zustandsraum zurücktransformierte Zustand bis auf einen vorgegebenen Fehlerwert (abhängig von der Anwendung) den ursprünglichen Zustand widerspiegelt.

8 Schlussfolgerung und Ausblick

Diese Arbeit führte eine Methodik ein, die es ermöglicht, Simulationsergebnisse eines FE-Modells für die Erstellung eines onlinefähigen Zustandstrackers mittels machinellen Lernens zu verwenden. Hierfür wurden mit künstlichen neuronalen Netzen und symbolischer Regression ein reduziertes, dynamisches und nichtlineares Prozessmodell sowie ein nichtlineares Messmodell erlernt. Das Vorhandensein eines onlinefähigen Prozessmodells zur Vorhersage des inneren Zustandes des Bauteils und des Messmodells zur Vorhersage der Observablen in Abhängigkeit des inneren Zustandes ermöglichte die Verwendung von bereits etablierten Zustandsbeobachtern. In dieser Arbeit wurden dafür zwei nichtlineare Erweiterungen des Kalman-Filters untersucht: Erweitertes Kalman-Filter und unscented Kalman-Filter.

Die eingeführte Herangehensweise hat grundlegend keine Einschränkung, was die Komplexität des Systems, die Anzahl der Ein- und Ausgangsgrößen sowie Nichtlinearität angeht. Für den Algorithmus stehen lediglich die aus der FE-Simulation erzeugten Daten für Zustandsgrößen und Observablen zur Verfügung. Es wird lediglich ein bereits experimentierfähige numerische Modell des Prozesses zur Datengenerierung benötigt. Die Daten ermöglichen es, ein reduziertes Prozessmodell zu erlernen, welches möglichst genau den realen Prozess widerspiegelt. Dadurch können nicht messbare Zustandsgrößen vorhergesagt werden. Zusätzlich kann aus den Daten ein Messmodell erlernt werden, welches die ermittelten Zustandsmerkmale auf die extrahierten Observablen abbildet. Die erlernten Prozess- und Messmodelle dienen zum Einsatz eines nichtlinearen Kalman-Filters zur Zustandsverfolgung. Die vorgestellte Herangehensweise ermöglicht es somit, für komplexe, nichtlineare Mehrgrößensysteme den inneren Zustand vorherzusagen und somit z. B. Zustandsregler einzusetzen, um Prozesse und Prozessketten zu optimieren und zu regeln.

Durch die Dimensionsreduktion mittels neuronaler Netze und symbolische Regression konnten reduzierte Modelle erstellt werden, die z. B. die Information von 400 Zustandsvariablen durch drei Zustandsmerkmale repräsentieren. Der Zustandstracker verwendete dann lediglich die symbolische Beschreibung dieser drei Zustandsmerkmale, um den Prozess zu verfolgen. Die Ergebnisse zeigen, dass dabei bis zu 88% der Genauigkeit der numerischen Simulation rekonstruiert werden konnte. Für die Zustandsverfolgung erwies sich für die unterschiedlichen Anwendungsfälle das EKF am robustesten. Die Berechnungszeit der Zustandsmerkmale mittels des reduzierten Modells liegt weit unter der tatsächlichen Ausführungszeit des Prozesses. Beispielsweise wird für die vorliegenden Prozessbeispiele und den kompletten Durchlauf des Zustandstrackers nur maximal ein Fünftel der Zeit benötigt. Dadurch könnte auch in Betracht gezogen werden, komplexere Lösungen der symbolischen Regression für das Prozessmodell zu verwenden oder mehr Zustandsmerkmale zum Prozessmodell hinzuzufügen, mit dem Ziel noch näher an die Lösung der numerischen Simulation heranzukommen.

Bezüglich der Verwendbarkeit für unterschiedliche Prozesse, sind weitere Untersuchungen im Hinblick auf Prozesse bzw. FE-Simulationen, z. B. mit Neuvernetzung der Elemente während

der Simulation, erforderlich. Eine weitere Betrachtung von unterschiedlichen Gewichtungen spezieller Bereiche des Bauteils würde im Hinblick auf auftretende Belastungszustände wertvolle Informationen liefern.

Um den Approximationsfehler bei der symbolischen Regression weiter zu verringern, wäre die Konzipierung eines rekurrenten Lernverfahrens für symbolische Regression erstrebenswert. Des Weiteren wäre eine automatische Anpassung der einstellbaren Parameter des UKFs von Vorteil, um die bestmöglichen Ergebnisse zu erzielen. Weiterhin bleibt zu klären, warum bei den nichtlinearen Kalman-Filtern und den gegebenen Anwendungsbeispielen teilweise für bestimmte Prozessparameter die wahren Zustandsmerkmale und vorhergesagte Zustandmerkmale zu stark divergieren. Hierfür könnte auch eine Abwandlung der nichtlinearen Kalman-Filter, sodass diese statt symbolischer Gleichungen für das Messmodell auch künstlichen neuronale Netze akzeptieren, beitragen.

A Grundlagen des Kalman-Filters

A.1 Hintergrund und Aufbau

Ausgangspunkt für das Kalman-Filter ist ein kontinuierliches dynamisches System im Zustandsraum. Die Systembeschreibung liegt dabei im Zeitbereich diskretisiert vor:

$$\dot{\boldsymbol{x}}_t = A\boldsymbol{x}_t + B\boldsymbol{u}_t + \nu_t \tag{A.1}$$

Weiterhin ist ein Messmodell vorhanden, das den Zustand auf die Observablen abbildet:

$$\boldsymbol{y}_t = C\boldsymbol{x}_t + \rho_t, \tag{A.2}$$

mit $\boldsymbol{x}_t \in \mathbb{R}^n$ für die Zustandsvariablen, $\boldsymbol{u}_t \in \mathbb{R}^m$ und $\boldsymbol{y}_t \in \mathbb{R}^r$ für die Eingangs- bzw. Ausgangsgrößen und dementsprechend Zustandübergangsmatrix $A \in \mathbb{R}^{n\times n}$, Eingangsmatrix $B \in \mathbb{R}^{n\times m}$ und Ausgangsmatrix $C \in \mathbb{R}^{r\times n}$, sowie das normalverteilte Prozess- und Messrauschen $\nu_t \in \mathbb{R}^n$ bzw. $\rho_t \in \mathbb{R}^r$. Im Folgenden wird die Kovarianzmatrix für ν_t als Q_t und für ρ_t als R_t bezeichnet.

Zum Zeitpunkt t wird die Annahme über den Zustand x_t durch den Mittelwert μ_t und die Kovarianz P_t repräsentiert. Für Mittelwert μ_{t-1} und Kovarianz P_{t-1} sowie gemessene Observablen $\boldsymbol{y}_t$ und gegebenen Eingangsgrößen $\boldsymbol{u}_t$ berechnet sich neuer Mittelwert μ_t, Kovarianz P_t, sowie Kalman-Gain L_t wie folgt in zwei Schritten,
Vorhersage:

$$\begin{aligned} \hat{\mu}_t &= A\mu_{t-1} + B\boldsymbol{u}_t \\ \hat{P}_t &= AP_{t-1}A^\top + Q_t \end{aligned} \tag{A.3}$$

und Korrektur bzw. Aktualisierung:

$$\begin{aligned} L_t &= \hat{P}_t C^\top (C\hat{P}_t C^\top + R_t)^{-1} \\ \mu_t &= \hat{\mu}_t + L_t(\boldsymbol{y}_t - C\hat{\mu}_t) \\ P_t &= (I - L_t C)\hat{P}_t, \end{aligned} \tag{A.4}$$

mit Einheitsmatrix I.

A.2 Berechnung der Sigmapunkte und Gewichte beim Unscented Kalman-Filter

Die Sigmapunkte, die aus der Gaußverteilung extrahiert werden, befinden sich symmetrisch um den Mittelwert verteilt. Für eine n-dimensionale Gaußverteilung berechnen sich die Sigmapunkte wie folgt:

$$\begin{aligned} \chi^{(0)} &= \mu \\ \chi^{(i)} &= \mu + \left(\sqrt{(n+\lambda)P}\right)_i \quad \text{für } i = 1,\ldots,n \\ \chi^{(i)} &= \mu - \left(\sqrt{(n+\lambda)P}\right)_{i-n} \quad \text{für } i = n+1,\ldots,2n \end{aligned} \tag{A.5}$$

mit $\lambda = \alpha^2(n+\kappa) - n$, wobei α und κ skalierbare Parameter sind, die darüber bestimmen, wie weit die Sigmapunkte sich um den Mittelwert verteilen [94].

Jeder Sigmapunkte hat ein Gewicht ω_m und ein Gewicht ω_c, um den Mittelwert bzw. die Kovarianz der Gaußverteilung zu bestimmen. Nach [45] gilt für den Sigmapunkt am Mittelwert:

$$\omega_m^{(0)} = \omega_c^{(0)} = \frac{\lambda}{n+\lambda} \tag{A.6}$$

und für alle weiteren Sigmapunkte:

$$\omega_m^{(i)} = \omega_c^{(i)} = \frac{1}{2(n+\lambda)} \text{ für } i = 1,\ldots,2n. \tag{A.7}$$

B Mathematische Definitionen

B.1 Jacobi-Matrix

Die Jacobi-Matrix J von $\boldsymbol{f}$ bezüglich $\boldsymbol{x}$ berechnet sich wie folgt:

$$J = \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{B.1}$$

B.2 Lie-Ableitung

Die Lie-Ableitung von der Funktion g bezüglich des Vektorfelds $\boldsymbol{f}$ ist gegeben durch:

$$L_f^1(g) = \frac{\partial g(x)}{\partial \boldsymbol{x}} . \boldsymbol{f} = \sum_{i=1}^{n} \frac{\partial g(x)}{\partial x_i} f_i \tag{B.2}$$

mit

$$\boldsymbol{f} := \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{bmatrix} \tag{B.3}$$

Zusätzlich gilt hier $L_f^0(g) = g(x)$ und für Lie-Ableitungen höherer Ordnung:

$$L_f^j(g) = \frac{\partial}{\partial x} \left[L_f^{j-1}(g) \right] . \boldsymbol{f} \tag{B.4}$$

C Ergebnisse Dimensionsreduktion

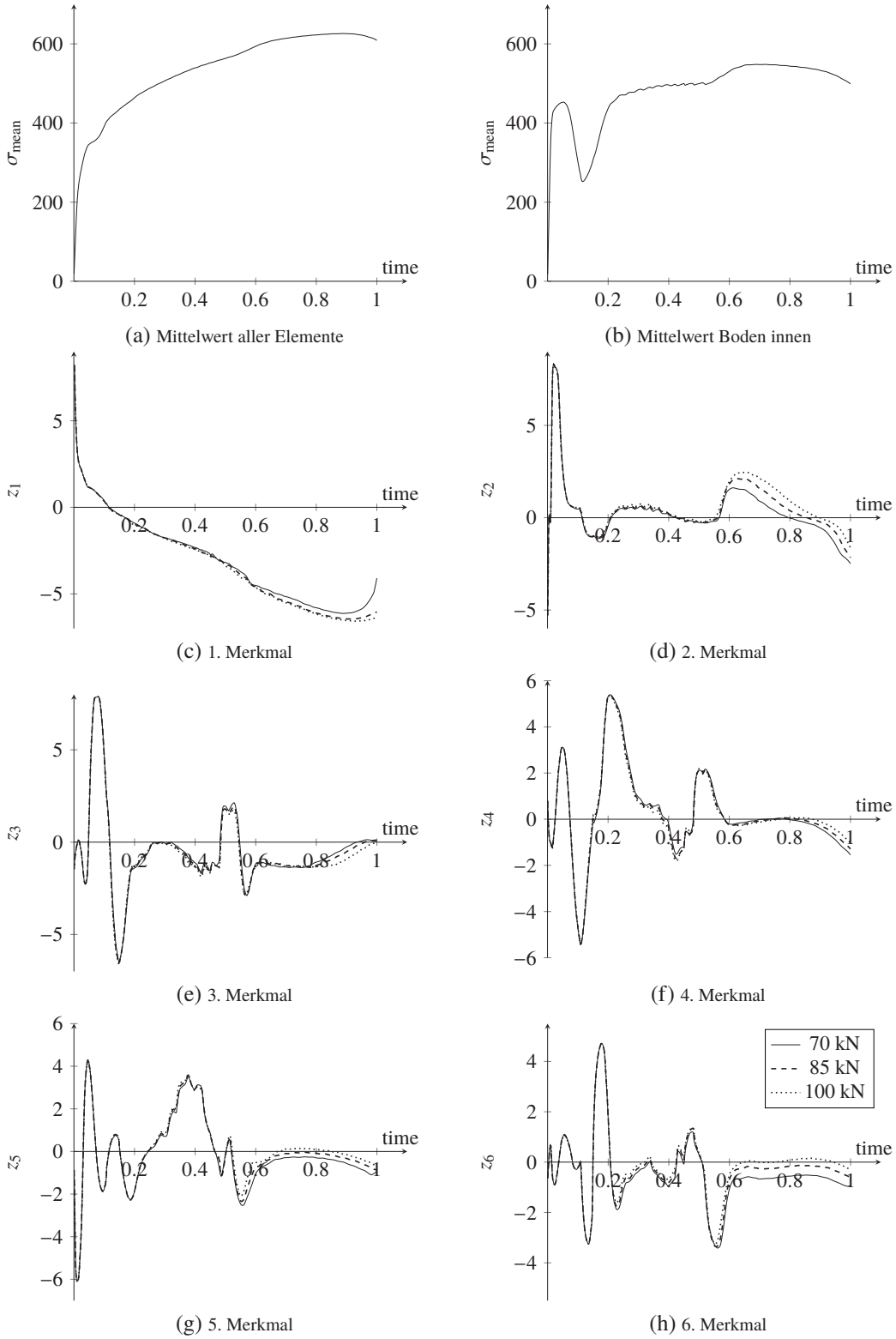
C.1 Reduktion von Mises Spannung

Abbildung C.1 zeigt die zeitlichen Verläufe der mittels SBNN_{temp} ermittelten Zustandsmerkmale für das zweidimensionale Tiefziehmodell und den Mittelwert über alle Elemente und alle Prozessparameter im gesamten Bauteil sowie der inneren Elemente im Bodenbereich.

Abbildung C.2 zeigt die zeitlichen Verläufe der mittels SBNN_{pre} ermittelten Zustandsmerkmale für das dreidimensionale Tiefziehmodell und den Mittelwert über alle Elemente und alle Prozessparameter im gesamten Bauteil sowie der inneren Elemente im Bodenbereich.

C.2 Reduktion plastische Dehnung

Abbildung C.3 und C.4 zeigt die zeitlichen Verläufe der mittels SBNN_{temp} ermittelten Zustandsmerkmale für das zweidimensionale Tiefziehmodell und den Mittelwert über alle Elemente und alle Prozessparameter sowie der inneren Elemente des Bodenbereichs.



(a) Mittelwert aller Elemente

(b) Mittelwert Boden innen

(c) 1. Merkmal

(d) 2. Merkmal

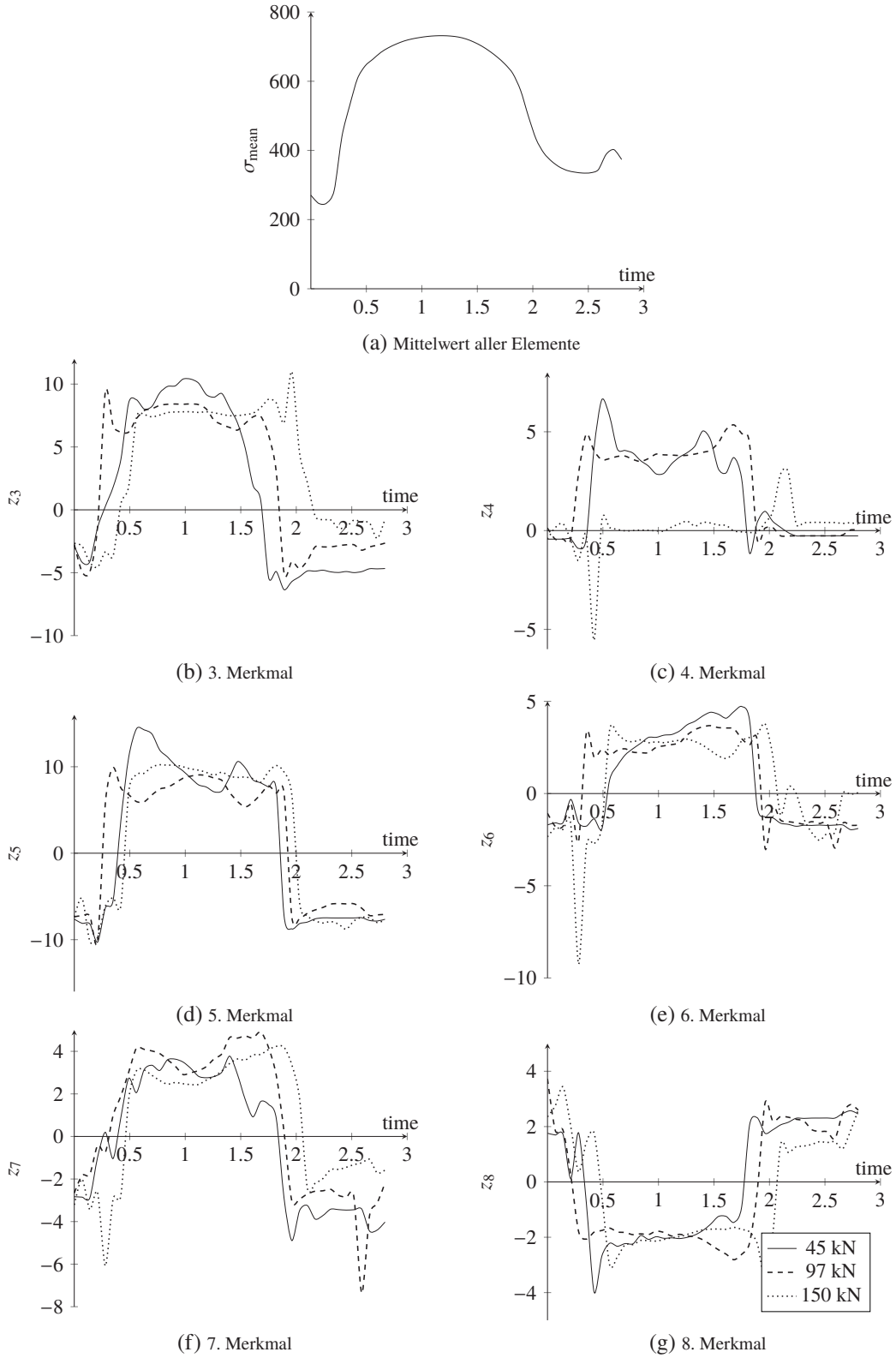
(e) 3. Merkmal

(f) 4. Merkmal

(g) 5. Merkmal

(h) 6. Merkmal

Abbildung C.1: 2D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $70kN$, $85kN$ und $100kN$ im Vergleich zu gemittelten von Mises Spannung in Elementen im Bauteil.



(a) Mittelwert aller Elemente

(b) 3. Merkmal

(c) 4. Merkmal

(d) 5. Merkmal

(e) 6. Merkmal

(f) 7. Merkmal

(g) 8. Merkmal

Abbildung C.2: 3D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $45kN$, $97kN$ und $150kN$ im Vergleich zu gemittelten von Mises Spannung in Elementen im Bauteil.

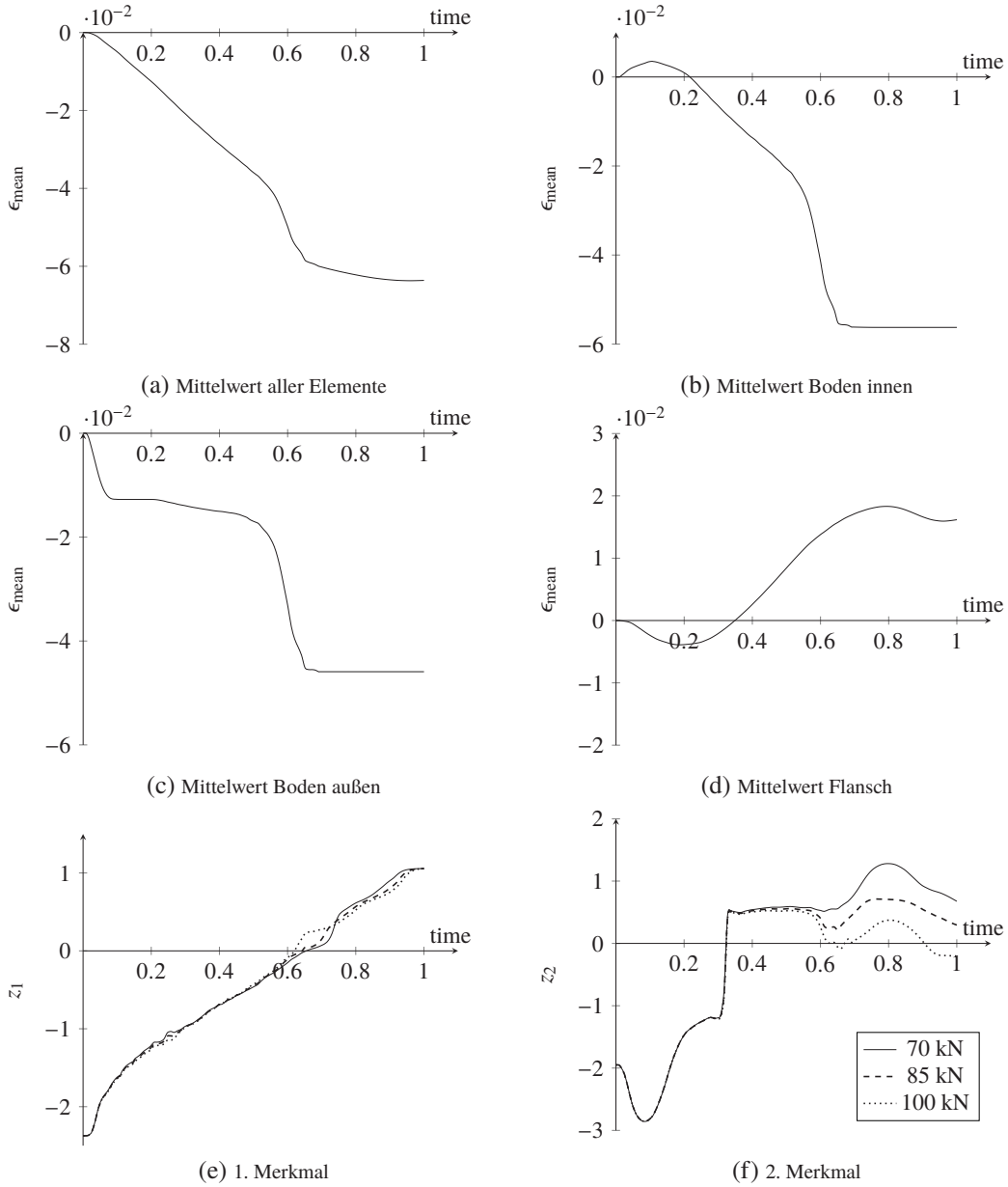
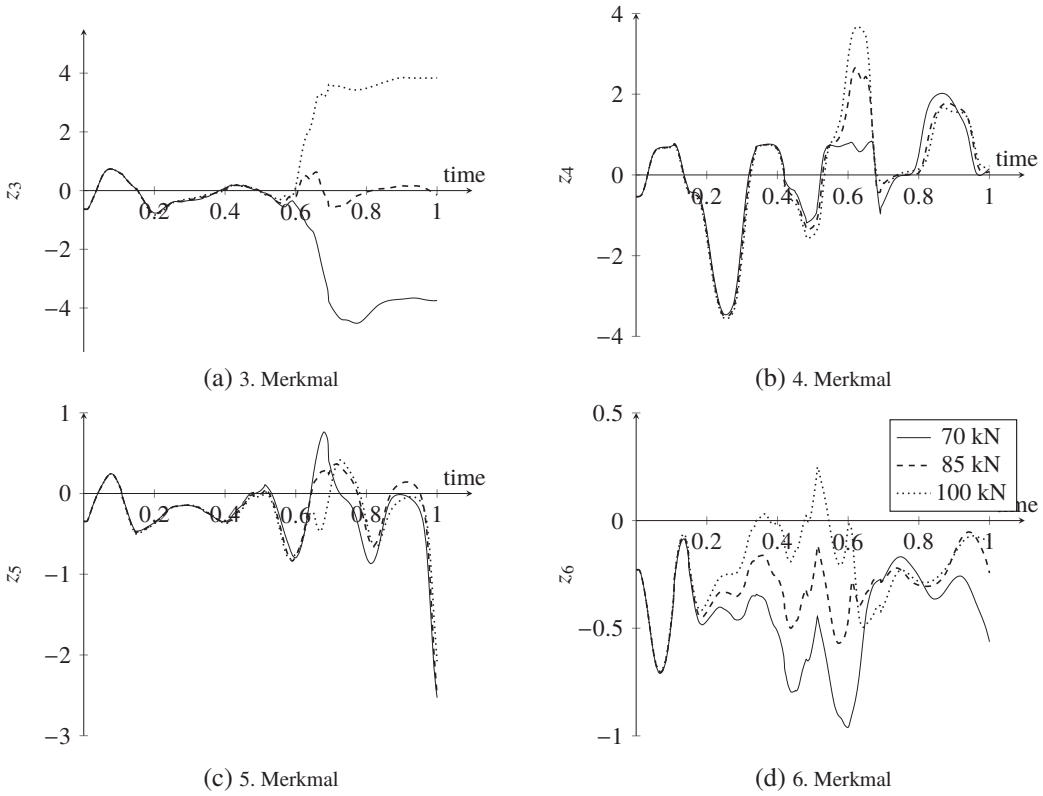


Abbildung C.3: 2D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $70kN$, $85kN$ und $100kN$ im Vergleich zu gemittelten plastischen Dehnung in Elementen im Bauteil (Teil 1).



(a) 3. Merkmal

(b) 4. Merkmal

(c) 5. Merkmal

(d) 6. Merkmal

Abbildung C.4: 2D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $70kN$, $85kN$ und $100kN$ im Vergleich zu gemittelten plastischen Dehnung in Elementen im Bauteil (Teil 2).

Abkürzungen und Symbole

Abkürzungen

KNN	Künstliches neuronales Netz
LSTM	Long Short Term Memory
MPC	Model Predictive Control
NARX	Nonlinear Autoregressive Exogenous Model
NLPCA	Nonlinear Principal Component Analysis
PCA	Principal Component Analysis (Hauptkomponentenanalyse)
PCR	Principal Component Regression
PFA	Principal Function Approximator
PGD	Proper Generalized Decomposition
PLS	Partial Least Squares
SPFA	Sequential Principal Function Approximator
SVD	Singular Value Decomposition (Singulärwertzerlegung)

Lateinische Symbole und Variablen

Kleinbuchstaben

$\boldsymbol{e}$	Beobachtungsfehler
f	Zustandsübergangsgleichung
g	Ausgangsgleichung
k	Anzahl Zustandsmerkmale
m	Anzahl Eingangsgrößen/Prozessparameter
n	Anzahl Zustandsvariablen
r	Anzahl Ausgangsgrößen/Observablen

s	Anzahl Stichproben
t	Zeit
$\boldsymbol{u}$	Eingangsgrößen/Prozessparameter/Regelgrößen
w_{ij}	Gewicht der Verbindung von Neuron j zu Neuron i
$\boldsymbol{x}$	Zustandsgrößen
$\hat{\boldsymbol{x}}$	Rekonstruierte Zustandsgrößen
$\boldsymbol{y}$	Ausgangsgrößen/Messgrößen/Observablen
$\hat{\boldsymbol{y}}$	Rekonstruierte Messgrößen
$\boldsymbol{z}$	Zustandsmerkmale

Großbuchstaben

A	Systemmatrix
B	Eingangsmatrix
C	Ausgangsmatrix
D	Anzahl Neuronen in der Demapping-Schicht
I	Anzahl Neuronen in der Eingabeschicht
L	Beobachtungsmatrix
M	Anzahl Neuronen in der Mapping-Schicht
Q	Kovarianz des Prozessrauschens
R	Kovarianz des Messrauschen
T	Anzahl Zeitschritte

Griechische Symbole und Variablen

α	Lernrate
Δt	Abtastintervall
ν	Prozessrauschen
ρ	Messrauschen
σ_n	Normalspannungen
σ_r	Radialspannungen
σ_t	Tangentialspannungen

ω_t	Führungsgröße

Bezeichnungsgrundsätze

$t_1, t_2, \ldots t_T$	Diskrete Zeitschritte
$\boldsymbol{x}$	Vektorielle Größe
$x_n^{(s)}$	n-te Zustandsvariable für s-te Stichprobe
$x_n^{(s,T)}$	n-te Zustandsvariable für s-te Stichprobe und T-ten Zeitschritt

Abbildungsverzeichnis

2.1 Finite-Elemente-Modell eines tiefgezogenen Blechbauteils mit Abaqus [89] erstellt. Farblich codiert ist die von Mises Spannung. 9
2.2 Betrachtete Prozesskette des Graduiertenkollegs 1483 mit unterschiedlichen Skalierungen der einzelnen Prozesse. 9
2.3 Übersicht über Eigenschaften und Verfahren zur Modellbildung. 14
2.4 Links: Einschichtiges KNN mit n Eingaben und r Ausgaben. Mitte: Mehrschichtiges KNN mit drei verdeckten Schichten und Ausgabeschicht. Rechts: Einfaches einschichtiges rekurrentes KNN mit Rückführung des Aktivierungswerts der Ausgabeschicht. 17
2.5 Berechnung des Aktivierungswert a durch ein Neuron, mit w_{ij} als Gewichtung der Verbindung zwischen Neuron j zu Neuron i, Eingabewerten $\boldsymbol{x}$ und Schwellwert b. 18
2.6 Backpropagation durch einen Knoten, mit $\frac{\partial E}{\partial x_1} = \frac{\partial E}{\partial a}\frac{\partial a}{\partial x_1}$ und $\frac{\partial E}{\partial x_n} = \frac{\partial E}{\partial a}\frac{\partial a}{\partial x_n}$. 19
2.7 Beispiel eines zweidimensionalen Datensatzes (rechts), der in einem dreidimensionalen Raum liegt (links). 20
2.8 Topologie der NLPCA nach [83] mit zwei Merkmalen. 23
2.9 Topologie des Principle Function Approximators. 25
2.10 Topologie des sequenziellen Principle Function Approximators mit drei Zustandsmerkmalen. 26
2.11 Beispielhafter Vergleich symbolischer Regression mit linearer und nichtlinearer Regression. 27
2.12 Baumstruktur der symbolischen Regression und die Operation Crossover. 28
2.13 Ablauf der symbolischen Regression. 29
2.14 Prinzip der Pareto-Front. 30

3.1 Luenberger-Beobachter parallel zur Regelstrecke. 31
3.2 Aufbau eines Standardregelkreises. 32
3.3 Ablauf des erweiterten Kalman-Filters. 35
3.4 Vergleich des ermittelten Mittelwerts für die Zustandsvariable durch UKF mittels Sigmapunkte χ und EKF mittels Linearisierung. 36
3.5 Ablauf des unscented Kalman-Filters. 38
3.6 Oben: Statisches Modell zur Prädiktion zukünftiger Ausgangsgrößen. Unten: Dynamisches Modell zur Simulation zukünftiger Ausgangsgrößen. 39
3.7 Zustandsvorhersage durch Observablen mittels Regression. 40
3.8 Aufbau statischer Black-Box-Beobachter und optimaler Regler nach Senn [87]. . 41
3.9 Statische und dynamische Konfiguration des NARX. 42

4.1 Systemarchitektur zur Erstellung eines onlinefähigen und datengetriebenen Zustandsbeobachter. 44

5.1 Kapitelaufbau und Vorgehensweise bei der Erstellung eines nichtlinearen dynamischen Zustandsbeobachters. 48
5.2 Exemplarische Reduktion von zwei Zustandsvariablen $\boldsymbol{x} \in \mathbb{R}^2$ (Datenpunkte: blau) auf ein Zustandsmerkmal $z \in \mathbb{R}$ (Koordinatenwerte: schwarz). 49
5.3 Vereinfachte Darstellung der Topologien von NLPCA und PFA mit Netzein- und ausgabe. 50
5.4 Topologie des sequenziellen Bottleneck neuronalen Netzes, welches zur Reduktion der Zustandsvariablen für alle Stichproben X verwendet wird. 52
5.5 Topologie des bottleneck neuronalen Netzes für das erste Zustandsmerkmal $\boldsymbol{z}_1$, mit Zustandsvariablen des aktuellen und vorherigen Zeitschrittes als Eingabe des Netzes. 55
5.6 Unteranpassung (oben) und Überanpassung (unten): Approximation der Trainingsdaten (links) und Approximation der Testdaten (rechts) bei einer beispielhaften Regressionsanwendung. 57
5.7 Gradientenabstieg für unterschiedliche Ausgangspunkte. 59
5.8 Pretraining des Netzes in Schritt 1 und 2 mit anschließendem Training des gesamten Netzes in Schritt 3. 60

6.1 Versuchsaufbau Wärmeleitung nach [96]. 72
6.2 Temperaturverlauf für unterschiedliche Frequenzen $\tilde{\omega}$ an Position $\tilde{p} = 0{,}5$. 74
6.3 Temperatur im Bauteil für unterschiedliche Zeitschritte mit Frequenz $\tilde{\omega} = 6$. . . . 75
6.4 Schematischer Aufbau einer Tiefziehvorrichtung. 75
6.5 Schematische Darstellung des Spannungs-Dehnungs-Diagramm. 76
6.6 Zweidimensionales (links) und dreidimensionales (rechts) FE-Modell eines tiefgezogenen Napfes mit von Mises Spannung als Zustandsbeschreibung. Die gelb markierten Elemente werden für die Ergebnisse und Evaluation genauer betrachtet. 78
6.7 Verfestigungsverhalten des Materials für das zweidimensionale (links) und dreidimensionale (rechts) Tiefziehmodell. 79
6.8 2D-Tiefziehmodell: Zeitlicher Verlauf der von Mises Spannung in je einem Element im Flansch, Zarge, Boden und Bodenrundung des Werkstückes (vgl. Abbildung 6.6). 81
6.9 2D-Tiefziehmodell: Ergebnisse für unterschiedliche Niederhalterkräfte (F_{NK}). Links: Zeitlicher Verlauf der Matrizenkraft in Ziehrichtung. Rechts: Stempelkraft versus Stempelverschiebung. 81
6.10 2D-Tiefziehmodell: Zeitlicher Verlauf der Dehnung in Ziehrichtung in je einem Element in Flansch, Zarge, Boden und Bodenrundung des Werkstückes (vgl. Abbildung 6.6). 82
6.11 3D-Tiefziehmodell: Zeitlicher Verlauf der von Mises Spannung in je einem Element in Flansch (Zipfel und Zipfeltal), Zarge, Bodenrundung und Boden des Werkstückes (vgl. Abbildung 6.6). 83
6.12 3D-Tiefziehmodell: Ergebnisse für unterschiedliche Niederhalterkräfte (F_{NK}). . . 84

6.13 3D-Tiefziehmodell: Zeitlicher Verlauf der Dehnung in Ziehrichtung in je einem Element in Flansch (Zipfel und Zipfeltal), Zarge, Bodenrundung und Boden des Werkstückes (vgl. Abbildung 6.6). 85

7.1 Wärmeleitung: Vergleich des Rekonstruktionsfehlers für die jeweilige Anzahl an Merkmalen für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{temp}$ und SBNN$_{temp_{pre}}$ auf normierten Daten. 89
7.2 Wärmeleitung: Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{temp}$ für Prozessparameter $\omega = 1$ (blau) und $\omega = 12$ (rot). 90
7.3 Wärmeleitung: Zeitliche Verläufe für die zwei Merkmale jeweils für Prozessparameter $\omega = 1$ (durchgezogene Linie), $\omega = 6$ (gestrichelte Linie) und $\omega = 12$ (gepunktete Linie), die durch PCA ((**a**), (**b**)) NLPCA ((**c**), (**d**)) und SBNN$_{temp_{pre}}$ ((**e**), (**f**)) extrahiert wurden. 91
7.4 2D-Tiefziehmodell (von Mises Spannung): Vergleich des Rekonstruktionsfehlers (MSE) pro Merkmal für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{temp}$. . 92
7.5 2D-Tiefziehmodell (von Mises Spannung): Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{temp}$ für Niederhalterkraft 70 kN (blau) und 100 kN (rot). 92
7.6 2D-Tiefziehmodell (von Mises Spannung): Zeitliche Verläufe der ersten zwei Merkmale für die Prozessparameter 70 kN, 85 kN und 100 kN mittels Berechnung nach PCA ((**a**), (**b**)), NLPCA ((**c**), (**d**)), SBNN ((**e**), (**f**)) und SBNN$_{temp}$ ((**g**), (**h**)). . . . 93
7.7 2D-Tiefziehmodell (Dehnung): Vergleich des Rekonstruktionsfehlers (MSE) pro Merkmal für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{temp}$. 94
7.8 2D-Tiefziehmodell (Dehnung): Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{temp}$ für Niederhalterkraft 70 kN (blau) und 100 kN (rot). 95
7.9 2D-Tiefziehmodell (Dehnung): Zeitliche Verläufe der ersten zwei Merkmale für die Prozessparameter 70 kN, 85 kN und 100 kN für jeweils Berechnung nach PCA ((**a**), (**b**)), NLPCA ((**c**), (**d**)), SBNN ((**e**), (**f**)) und SBNN$_{temp}$ ((**g**), (**h**)). 96
7.10 3D-Tiefziehmodell (von Mises Spannung): Vergleich des Rekonstruktionsfehlers pro Merkmal für die Verfahren PCA, NLPCA, SBNN sowie SBNN$_{pre}$. 97
7.11 3D-Tiefziehmodell (von Mises Spannung): Vergleich ursprünglicher Zustand im Bauteil mit der Rekonstruktion aus PCA, NLPCA, SBNN und SBNN$_{pre}$ für Niederhalterkraft 45 kN (blau) und 150 kN (rot). 97
7.12 3D-Tiefziehmodell (von Mises Spannung): Zeitliche Verläufe der ersten zwei Merkmale für die Prozessparameter 45 kN, 97 kN und 150 kN für jeweils Berechnung nach PCA ((**a**), (**b**)) NLPCA ((**c**), (**d**)), SBNN ((**e**), (**f**)) und SBNN$_{pre}$ ((**g**), (**h**)). . 98
7.13 Ergebnisse Reduzierung von Mises Spannung für das zweidimensionale Modell für unterschiedliche Anzahl Neuronen in den verdeckten Schichten bei dreifacher Kreuzvalidierung ohne Pretraining. 100
7.14 Vergleich zeitlicher Verlauf des zweiten und dritten Merkmals für unterschiedliche Anzahl Neuronen in den verdeckten Schichten bei dreifacher Kreuzvalidierung. . 100

7.15 Ergebnisse Reduzierung von Mises Spannung für das zweidimensionale Modell für unterschiedliche Anzahl Kreuzvalidierungen bei 8 Merkmalen in den verdeckten Schichten mit und ohne Pretraining $_{pre}$. 101
7.16 2D-Tiefziehmodell (von Mises Spannung): Vergleich zeitlicher Verlauf der ersten vier Merkmale für Training mit und ohne Pretraining und dreifacher Kreuzvalidierung. 101
7.17 Wärmeleitung: Pareto-Front für $f^{(E)}$. 104
7.18 Wärmeleitung: Pareto-Front für $f^{(N)}$. 105
7.19 2D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(Z)}$. 107
7.20 2D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(N)}$. 108
7.21 2D-Tiefziehmodell (Dehnung): Pareto-Front für $f^{(N)}$. 110
7.22 3D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(E)}$. 112
7.23 3D-Tiefziehmodell (von Mises Spannung): Pareto-Front für $f^{(N)}$. 113
7.24 Wärmeleitung: Grün markiert die Überschreitungen des Fehlerschwellwertes für $\omega = 6$ zu den Zeitpunkten 0,005, 0,5 und 1. 118
7.25 2D-Tiefziehmodell (von Mises Spannung): Anzahl der Zeitschritte, die je Parameter durchlaufen wurden. Die Niederhalterkraft als Parameter variiert 200 mal äquidistant zwischen 70 und 100 kN. 118
7.26 2D-Tiefziehmodell (von Mises Spannung): Vergleich Endzustand in FE-Simulation (links) und nach Zustandsverfolgung mittels reduziertem Modell und EKF (rechts) für Niederhalterkraft 70 kN. 119
7.27 2D-Tiefziehmodell (von Mises Spannung): Bereiche der Fehlerschwellwertüberschreitungen von mehr als 50 MPa (weiß) nach Zustandsverfolgung mittels reduziertem Modell und EKF für Niederhalterkraft 70 kN. 119
7.28 2D-Tiefziehmodell (Dehnung): Anzahl der Zeitschritte, die je Parameter durchlaufen wurden. Die Niederhalterkraft als Parameter variiert 200 mal zwischen 70 und 100 kN. 120
7.29 2D-Tiefziehmodell (Dehnung): Vergleich Endzustand in FE-Simulation (links) und nach Zustandsverfolgung mittels reduziertem Modell und EKF (rechts) für Niederhalterkraft 70 kN. 120
7.30 2D-Tiefziehmodell (Dehnung): Fehlerwerte nach Zustandsverfolgung mittels reduziertem Modell und EKF für Niederhalterkraft 70 kN. Der Fehlerwert liegt für jedes Element unter dem Schwellwert von 0,05. 120
7.31 3D-Tiefziehmodell (von Mises Spannung): Anzahl der Zeitschritte, die je Parameter durchlaufen wurden. Die Niederhalterkraft als Parameter variiert 100 mal zwischen 45 und 150 kN. 121

C.1 2D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $70kN$, $85kN$ und $100kN$ im Vergleich zu gemittelten von Mises Spannung in Elementen im Bauteil. 130
C.2 3D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $45kN$, $97kN$ und $150kN$ im Vergleich zu gemittelten von Mises Spannung in Elementen im Bauteil. 131

C.3 2D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $70kN$, $85kN$ und $100kN$ im Vergleich zu gemittelten plastischen Dehnung in Elementen im Bauteil (Teil 1). 132
C.4 2D-Tiefziehmodell: Zeitliche Verläufe der extrahierte Merkmale für die Prozessparameter $70kN$, $85kN$ und $100kN$ im Vergleich zu gemittelten plastischen Dehnung in Elementen im Bauteil (Teil 2). 133

Tabellenverzeichnis

2.1 Ordnungsschema mathematischer Modelle anhand ihrer Eigenschaften. 11
2.2 Vergleich feedforward-Netz und rekurrentes Netz. 17

6.1 Größen der Simulationsstudie. 74
6.2 Geometrieeigenschaften der Tiefziehmodelle. 78
6.3 Übersicht der Kennwerte der Simulationsstudien für das zwei- sowie dreidimensionale Tiefziehmodell. 80

7.1 Wärmeleitung: Fehler $(1 - R^2)$ und Komplexität der gewählten Ergebnisse der symbolischen Regression für unterschiedliche Strukturvorgaben. 103
7.2 2D-Tiefziehmodell: Fehler $(1 - R^2)$ und Komplexität der gewählten Ergebnisse der symbolischen Regression für die Merkmale der von Mises Spannung. 106
7.3 2D-Tiefziehmodell: Fehler $(1 - R^2)$ und Komplexität der gewählten Ergebnisse der sym. Regression für die Merkmale der Dehnung in Ziehrichtung. 109
7.4 3D-Tiefziehmodell: Fehler und Komplexität der gewählten Ergebnisse der symbolischen Regression der reduzierten von Mises Spannungswerte. 111
7.5 Vergleich E_{MSE} für Messmodell mittels symbolischer Regression (SR) und künstlichem neuronalen Netz (KNN). 114
7.6 Ergebnisse der nichtlinearen Kalman Filter als Zustandstracker. 117

Literaturverzeichnis

[1] Ashhab, M. S. ; Breitsprecher, T. ; Wartzack, S. : Neural network based modeling and optimization of deep drawing – extrusion combined process. In: *Journal of Intelligent Manufacturing* 25 (2014), Nr. 1, S. 77–84

[2] Banabic, D. : *Sheet Metal Forming Processes: Constitutive Modelling and Numerical Simulation*. Springer, 2010

[3] Barone, M. F. ; Kalashnikova, I. ; Brake, M. R. ; Segalman, D. J.: Reduced Order Modeling of Fluid/Structure Interaction / Sandia National Laboratories Report. 2009 (2009-7189). – Forschungsbericht. – Albuquerque, USA

[4] Belkin, M. ; Niyogi, P. : Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. In: *Advances in Neural Information Processing Systems 14*, MIT Press, 2001, S. 585–591

[5] Camacho, E. F. ; Bordons, C. : *Model Predictive Control*. Springer London, 2007

[6] Carpenter, W. C. ; Hoffman, M. E.: Selecting the Architecture of a Class of Back-propagation Neural Networks used as Approximators. In: *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing* 11 (1997), 1, S. 33–44

[7] Chinesta, F. ; Keunings, R. ; Leygue, A. : *The Proper Generalized Decomposition for Advanced Numerical Simulations*. Springer International Publishing, 2014 (SpringerBriefs in Applied Sciences and Technology)

[8] Chinesta, F. ; Leygue, A. ; Bordeu, F. ; Aguado, J. ; Cueto, E. ; Gonzalez, D. ; Alfaro, I. ; Ammar, A. ; Huerta, A. : PGD-Based Computational Vademecum for Efficient Design, Optimization and Control. In: *Archives of Computational Methods in Engineering* 20 (2013), Nr. 1, S. 31–59

[9] Christmann, A. ; Steinwart, I. : *Support Vector Machines*. Springer New York, 2008

[10] Cole, K. D. ; Brettmann, L. : *Slab body with cosine-periodic temperature variation at x=0, and zero initial temperature at x=L*. Exact Analytical Conduction Toolbox, exact.unl.edu, 2014

[11] Cordier, L. ; Noack, B. R. ; Tissot, G. ; Lehnasch, G. ; Delville, J. ; Balajewicz, M. ; Daviller, G. ; Niven, R. K.: Identification Strategies for Model-based Control. In: *Experiments in Fluids* 54 (2013), Nr. 8

[12] Cox, T. F. ; Cox, M. : *Multidimensional Scaling, Second Edition*. 2. Chapman and Hall/CRC, 2000

[13] Deutsches Institut für Normung e.V.: *Manufacturing Processes Forming – Classification; Subdivision, Terms and Definitions, Alphabetical Index*. DIN 8582:2003-09. Beuth, Berlin, 2003

[14] Doege, E. ; Behrens, B. : *Handbuch Umformtechnik: Grundlagen, Technologien, Maschinen*. 2. Springer Berlin Heidelberg, 2010 (VDI-Buch)

[15] Duchi, J. ; Hazan, E. ; Singer, Y. : Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. In: *J. Mach. Learn. Res.* 12 (2011), S. 2121–2159

[16] Elliott, R. J. ; Aggoun, L. ; Moore, J. B.: *Hidden Markov models : estimation and control*. Springer, 1995 (Stochastic Modelling and Applied Probability 29)

[17] Elman, J. L.: Finding Structure in Time. In: *Cognitive Science* 14 (1990), Nr. 2, S. 179–211

[18] Endelt, B. ; Tommerup, S. ; Danckert, J. : A novel feedback control system – Controlling the material flow in deep drawing using distributed blank-holder force. In: *Journal of Materials Processing Technology* 213 (2013), Nr. 1, S. 36–50

[19] Erhan, D. ; Bengio, Y. ; Courville, A. ; Manzagol, P.-A. ; Vincent, P. ; Bengio, S. : Why Does Unsupervised Pre-training Help Deep Learning? In: *J. Mach. Learn. Res.* 11 (2010), S. 625–660

[20] Fahlman, S. E. ; Lebiere, C. ; Touretzky, D. S. (Hrsg.): *Advances in Neural Information Processing Systems 2*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1990. – 524–532 S.

[21] Fischer, S. ; Hensgen, O. ; Elshaabiny, M. ; Link, N. : Generating Low-dimensional, Nonlinear Process Representations by Ordered Features. In: *IFAC-PapersOnLine* 48 (2015), Nr. 3, S. 1037–1042. – 15th IFAC Symposium on Information Control Problems in Manufacturing 2015

[22] Föllinger, O. (Hrsg.) ; Konigorski, U. (Hrsg.): *Regelungstechnik : Einführung in die Methoden und ihre Anwendung*. 11. Berlin : VDE Verlag, 2013

[23] Fritzen, F. : *Microstructural modeling and computational homogenization of the physically linear and nonlinear constitutive behavior of micro-heterogeneous materials*, Karlsruher Institut für Technologie, Diss., 2011

[24] Golden, R. M.: *Mathematical Methods for Neural Network Analysis and Design*. 1st. Cambridge, MA, USA : MIT Press, 1996

[25] Goodfellow, I. ; Bengio, Y. ; Courville, A. : *Deep Learning*. MIT Press, 2016. – `http://www.deeplearningbook.org`

[26] Gordon, N. J. ; Salmond, D. J. ; Smith, A. F. M.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In: *IEE Proceedings F - Radar and Signal Processing* 140 (1993), Nr. 2, S. 107–113

[27] Gorinevsky, D. : Radial Basis Function Network Approximation and Learning in Task-dependent Feedforward Control of Nonlinear Dynamical Systems. In: *Neural Network System Techniques and Applications* (1998)

[28] Hagan, M. T. ; Menhaj, M. B.: Training feedforward networks with the Marquardt algorithm. In: *IEEE Transactions on Neural Networks* 5 (1994), Nr. 6, S. 989–993

[29] Hao, W. ; Duncan, S. : Constrained model predictive control of an incremental sheet forming process. In: *CCA*, IEEE, 2011, S. 1288–1293

[30] Hassan, M. A. ; Yamaguchi, K. ; Takakura, N. : A fuzzy model for prediction of material flow-stress in metal forming. In: Mori, K. (Hrsg.): *Simulation of Materials Processing: Theory, Methods and Applications*, 2001, S. 303–308. – Proceedings of the 7th International Conference on Numerical Methods in Industrial Forming Processes

[31] Hedrick, J. K. ; Girard, A. : *Control of nonlinear dynamic systems: theory and applications*. Berkeley press, 2005

[32] Hensgen, O. : *Datengetriebene Optimierung nicht-linearer Regressionsstruktur*, Karlsruhe University of Applied Sciences, Masterarbeit, 2013

[33] Hinton, G. E. ; Salakhutdinov, R. R.: Reducing the dimensionality of data with neural networks. In: *Science* 313 (2006), Nr. 5786, S. 504 – 507

[34] Hochreiter, S. ; Schmidhuber, J. : Long Short-Term Memory. In: *Neural Computation* 9 (1997), Nr. 8, S. 1735–1780

[35] Hopfield, J. J.: Neural networks and physical systems with emergent collective computational abilities. In: *Proceedings of the National Academy of Sciences* 79 (1982), Nr. 8, S. 2554–2558

[36] Hornik, K. : Approximation capabilities of multilayer feedforward networks. In: *Neural Networks* 4 (1991), Nr. 2, S. 251–257

[37] Hsu, C.-W. ; Ulsoy, A. ; Demeri, M. : An approach for modeling sheet metal forming for process controller design. In: *Journal of Manufacturing Science and Engineering, Transactions of the ASME* 122 (2000), Nr. 4, S. 717–724

[38] Hsu, C.-W. ; Ulsoy, A. ; Demeri, M. : Development of process control in sheet metal forming. In: *Journal of Materials Processing Technology* 127 (2002), Nr. 3, S. 361–368

[39] Jacobs, R. A.: Increased rates of convergence through learning rate adaptation. In: *Neural Networks* 1 (1988), Nr. 4, S. 295–307

[40] Jakumeit, J. ; Herdy, M. ; Nitsche, M. : Parameter optimization of the sheet metal forming process using an iterative parallel Kriging algorithm. In: *Structural and Multidisciplinary Optimization* 29 (2005), Nr. 6, S. 498–507

[41] *Kapitel* Splines. In: Johnson, S. A.: *Encyclopedia of Operations Research and Management Science*. Boston, MA : Springer US, 2013, S. 1443–1446

[42] Jolliffe, I. T.: A Note on the Use of Principal Components in Regression. In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 31 (1982), Nr. 3, S. 300–303

[43] Jordan, M. I.: Artificial Neural Networks. In: Diederich, J. (Hrsg.): *Artificial Neural Networks*. Piscataway, NJ, USA : IEEE Press, 1990, Kapitel Attractor Dynamics and Parallelism in a Connectionist Sequential Machine, S. 112–127

[44] Julier, S. J. ; Uhlmann, J. K.: Unscented filtering and nonlinear estimation. In: *Proceedings of the IEEE* 92 (2004), Nr. 3, S. 401–422

[45] Julier, S. J. ; Uhlmann, J. K.: New extension of the Kalman filter to nonlinear systems. In: Kadar, I. (Hrsg.): *Signal Processing, Sensor Fusion, and Target Recognition VI* Bd. 3068, SPIE Proceedings, 1997, S. 182–193

[46] Kalman, R. E.: A New Approach to Linear Filtering and Prediction Problems. In: *Journal of Fluids Engineering* 82 (1960), Nr. 1, S. 35–45

[47] Kalman, R. E. ; Falb, P. L. ; Arbib, M. A.: *Topics in mathematical system theory*. McGraw-Hill, 1969 (International series in pure and applied mathematics)

[48] Karhunen, K. : *Über lineare Methoden in der Wahrscheinlichkeitsrechnung*. Universitat Helsinki, 1947 (Annales Academiae scientiarum Fennicae: Mathematica - Physica)

[49] Kingma, D. P. ; Ba, J. : Adam: A Method for Stochastic Optimization. In: *CoRR* abs/1412.6980 (2014)

[50] Kohonen, T. (Hrsg.) ; Schroeder, M. R. (Hrsg.) ; Huang, T. S. (Hrsg.): *Self-Organizing Maps*. 3rd. Secaucus, NJ, USA : Springer-Verlag New York, Inc., 2001

[51] Kommenda, M. ; Kronberger, G. ; Wagner, S. ; Winkler, S. ; Affenzeller, M. : On the Architecture and Implementation of Tree-based Genetic Programming in HeuristicLab. In: *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*. New York, NY, USA : ACM, 2012 (GECCO '12), S. 101–108

[52] Kosambi, D. : Statistics in function space. In: *Journal of Indian Mathematical Society* 7 (1943), S. 76–88

[53] Koza, J. R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992

[54] Kramer, M. : Nonlinear Principal Component Analysis Using Autoassociative Neural Networks. In: *AIChE Journal* 37 (1991), S. 233–243

[55] Krige, D. G.: A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand. In: *Journal of the Chemical, Metallurgical and Mining Society of South Africa* 52 (1951), Nr. 6, S. 119–139

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

[57] Lee, J. A. N. ; Verleysen, M. : *Nonlinear dimensionality reduction*. 1. Ed. New York, NY : Springer, 2007 (Information Science and Statistics)

[58] Loeve, M. : Fontions aléatoire de second ordre. In: *Compte Rend. Acad. Sci. (Paris)* 220 (1945)

[59] Luenberger, D. G.: Observing the State of a Linear System. In: *IEEE transactions on military electronics* 8 (1964), Nr. 2, S. 74–80

[60] Lumley, J. L.: The Structure of Inhomogeneous Turbulent Flows. In: Yaglom, A. M. (Hrsg.) ; Tatarski, V. I. (Hrsg.): *Atmospheric turbulence and radio propagation*. Moscow : Nauka, 1967, S. 166–178

[61] Lutz, H. ; Wendt, W. : *Taschenbuch der Regelungstechnik mit MATLAB und Simulink*. EuropaLehrmittel, 2014

[62] Manoochehri, M. ; Kolahan, F. : Integration of artificial neural network and simulated annealing algorithm to optimize deep drawing process. In: *The International Journal of Advanced Manufacturing Technology* 73 (2014), Nr. 1, S. 241–249

[63] Marquardt, D. W.: An algorithm for least-squares estimation of nonlinear parameters. In: *SIAM Journal on Applied Mathematics* 11 (1963), Nr. 2, S. 431–441

[64] Matheron, G. : Principles of geostatistics. In: *Economic Geology* 58 (1963), Nr. 8, S. 1246–1266

[65] MathWorks: *Design Time Series NARX Feedback Neural Networks*. Online. `https://de.mathworks.com/help/nnet/ug/design-time-series-narx-feedback-neural-networks.html`. Version: Januar 2017

[66] Merz, C. J. ; Pazzani, M. J.: A Principal Components Approach to Combining Regression Estimates. In: *Machine Learning* 36 (1999), Nr. 1, S. 9–32

[67] Narendra, K. S. ; Parthasarathy, K. : Identification and control of dynamical systems using neural networks. In: *IEEE Transactions on Neural Networks* 1 (1990), Nr. 1, S. 4–27

[68] Nelles, O. : *Nonlinear System Identification*. 1. Springer Berlin Heidelberg, 2001

[69] nutonian: *Eureqa: The A.I.-Powered Modeling Engine*. Online. `https://www.nutonian.com/products/eureqa/`. Version: Dezember 2017

[70] Press, W. H. ; Teukolsky, S. A. ; Vetterling, W. T. ; Flannery, B. P.: *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. 3. Cambridge University Press, 2007

[71] Quarteroni, A. (Hrsg.) ; Rozza, G. (Hrsg.): *Reduced Order Methods for Modeling and Computational Reduction*. Springer, 2013 (MS&A, Modeling, Simulation and Applications)

[72] Rao, C. R. ; Heumann, C. ; Shalabh ; Toutenburg, H. : *Linear models : least squares and alternatives*. 3. Springer, 2008

[73] Rasmussen, C. E.: Gaussian Processes in Machine Learning. In: Bousquet, O. (Hrsg.) ; Luxburg, U. von (Hrsg.) ; Rätsch, G. (Hrsg.): *Advanced Lectures on Machine Learning*. Springer Berlin Heidelberg, 2004, S. 63–71

[74] Riedmiller, M. ; Braun, H. : A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: *IEEE International Conference on Neural Networks*, 1993, S. 586–591

[75] Rieger, F. : *Work-hardening of dual-phase steel:*. Karlsruher Institut für Technologie, 2016 (Schriftenreihe Kontinuumsmechanik im Maschinenbau / Karlsruher Institut für Technologie, Institut für Technische Mechanik - Bereich Kontinuumsmechanik)

[76] Rojas, R. : *Neural Networks: A Systematic Introduction*. Springer New York, 1996

[77] Roweis, S. T. ; Saul, L. K.: Nonlinear dimensionality reduction by locally linear embedding. In: *Science* 290 (2000), S. 2323–2326

[78] Rumelhart, D. E. ; Hinton, G. E. ; Williams, R. J.: Learning Representations by Back-propagating Errors. In: *Nature* (1986)

[79] Schmidhuber, J. : Deep Learning in Neural Networks: An Overview. In: *Neural Networks* 61 (2015), S. 85–117

[80] Schmidt, M. ; Lipson, H. : Distilling Free-Form Natural Laws from Experimental Data. In: *Science* 324 (2009), Nr. 5923, S. 81–85

[81] Schneider, D. : *Phasenfeldmodellierung mechanisch getriebener Grenzflächenbewegungen in mehrphasigen Systemen*, Karlsruher Institut für Technologie, Diss., 2016

[82] Schölkopf, B. ; Smola, A. ; Müller, K.-R. ; Gerstner, W. (Hrsg.) ; Germond, A. (Hrsg.) ; Hasler, M. (Hrsg.) ; Nicoud, J.-D. (Hrsg.): *Kernel principal component analysis*. Berlin, Heidelberg : Springer Berlin Heidelberg, 1997. – 583–588 S.

[83] Scholz, M. ; Vigário, R. : Nonlinear PCA: A new Hierarchical Approach. In: *European Symposium on Artificial Neural Networks*, 2002, S. 439–444

[84] Scholz, M. ; Kaplan, F. ; Guy, C. L. ; Kopka, J. ; Selbig, J. : Non-linear PCA: A Missing Data Approach. In: *Bioinformatics* 21 (2005), Nr. 20, S. 3887–3895

[85] Searson, D. ; Leahy, D. ; Willis, M. : GPTIPS: an open source genetic programming toolbox for multigene symbolic regression. In: *Proceedings of the International Multi-Conference of Engineers and Computer Scientists 2010*, 2010, S. 77–80

[86] Senn, M. ; Link, N. : Hidden State Observation for Adaptive Process Controls. In: *Proceedings of the Second International Conference on Adaptive and Self-adaptive Systems and Applications (ADAPTIVE 2010)* IARIA, 2010, S. 52–57

[87] Senn, M. : *Optimale Prozessführung mit merkmalsbasierter Zustandsverfolgung*, Karlsruhe Institute of Technology (KIT), Diss., 2012

[88] Sheela, K. G. ; Deepa, S. N.: Review on Methods to Fix Number of Hidden Neurons in Neural Networks. In: *Mathematical Problems in Engineering* (2013)

[89] SIMULIA: *Abaqus/CAE*. Online. `https://www.3ds.com/de/produkte-und-services/simulia/produkte/abaqus/`. Version: Dezember 2017

[90] Song, Y. ; Li, X. : Intelligent Control Technology for the Deep Drawing of Sheet Metal. In: *Second International Conference on Intelligent Computation Technology and Automation, 2009. ICICTA '09.* Bd. 1, 2009, S. 797–801

[91] Sutton, R. S. ; Barto, A. G.: *Introduction to Reinforcement Learning*. 1. Cambridge, MA, USA : MIT Press, 1998

[92] Swerling, P. : *A proposed stagewise differential correction procedure for satellite tracking and prediction*. Santa Monica, Calif. : RAND, 1958

[93] Tenenbaum, J. B. ; Silva, V. de ; Langford, J. C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. In: *Science* 290 (2000), Nr. 5500, S. 2319

[94] Thrun, S. ; Burgard, W. ; Fox, D. : *Probabilistic Robotics*. The MIT Press, 2006

[95] Tollenaere, T. : SuperSAB: Fast adaptive back propagation with good scaling properties. In: *Neural Networks* 3 (1990), Nr. 5, S. 561–573

[96] University of Nebraska-Lincoln: *Exact Analytical Conduction Toolbox*. online. `http://exact.unl.edu/`. Version: Januar 2017

[97] VDI-Fachbereich Fabrikplanung und -betrieb: *Simulation von Logistik-, Materialfluss- und Produktionssystemen – Grundlagen*. VDI-Richtlinie 3633, Blatt 1. Beuth Berlin, 1993

[98] Vinzi, V. E. ; Chin, W. W. ; Henseler, J. ; Wang, H. : *Handbook of Partial Least Squares*. Springer, 2010

[99] Wang, H. ; Li, E. ; Li, G. Y.: Parallel boundary and best neighbor searching sampling algorithm for drawbead design optimization in sheet metal forming. In: *Structural and Multidisciplinary Optimization* 41 (2010), Nr. 2, S. 309–324

[100] Watson, G. S.: Smoothing and interpolation by kriging and with splines. In: *Journal of the International Association for Mathematical Geology* 16 (1984), Nr. 6, S. 601–615

[101] Weinan, E. : *Principles of Multiscale Modeling*. Cambridge University Press, 2011

[102] Yoo, J. D. ; Hwang, S. W. ; Park, K.-T. : Factors influencing the tensile behavior of a Fe–28Mn–9Al–0.8C steel. In: *Materials Science and Engineering: A* 508 (2009), Nr. 1–2, S. 234 – 240

[103] Zhao, J. ; Wang, F. : Parameter identification by neural network for intelligent deep drawing of axisymmetric workpieces. In: *Journal of Materials Processing Technology* 166 (2005), Nr. 3, S. 387–391