TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Regelungstechnik

Versuchsplanung und Methoden zur Identifikation zeitkontinuierlicher Zustandsraummodelle am Beispiel des Verbrennungsmotors

Dipl.-Ing. Univ. Michael Deflorian

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines Doktor-Ingenieurs genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. Hartmut Spliethoff
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. habil. Boris Lohmann
	2. Univ.-Prof. Dr.-Ing. Georg Wachtmeister

Die Dissertation wurde am 07.09.2011 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 01.12.2011 angenommen.

Kurzfassung

Diese Arbeit beschreibt neue Methoden der Online-Versuchsplanung und Online-Identifikation dynamischer Zusammenhänge am Verbrennungsmotorenprüfstand. Diese Methoden sollen die Steuergeräteapplikation bei der Modellierung und Optimierung dynamischer Vorgänge unterstützen. Die Ergebnisse der Arbeit fließen in den modellbasierten Online-Optimierungsalgorithmus ein, welcher zur Optimierung des Stationärverhaltens des Verbrennungsmotors verwendet wird, und ermöglichen diesen zukünftig auch zur Optimierung des Instationärverhaltens einzusetzen.

Die wesentlichen Neuerungen dieser Arbeit betreffen die modellbasierte Online-Versuchsplanung und Online-Modellbildung. Die vorgestellten Methoden der Online-Versuchsplanung ermöglichen es unter Berücksichtigung von Einschränkungen an die Ein- und Ausgänge mit möglichst wenigen Messungen einen maximalen Informationsgewinn aus diesen zu ziehen und die Versuchsplanung hinsichtlich des späteren Einsatzzwecks der Modelle zu optimieren. Die Methoden der Online-Modellbildung ermöglichen die Identifikation eingangs-/zustandsstabiler zeitkontinuierlicher Zustandsraummodelle unter Verwendung von Neuronalen Netzen, die mittels Runge-Kutta Methoden zeitdiskretisiert werden. Es kann gezeigt werden, dass sowohl die Methoden der Online-Versuchsplanung als auch die der Online-Modellbildung zu einer Verbesserung der erzielten Modellgüte führen und den bisher am Verbrennungsmotorenprüfstand eingesetzten Methoden der Versuchsplanung und Identifikation dynamischer Modelle überlegen sind. Die Versuchsplanung und Modellbildung eignet sich auch für den konventionellen Offline-Betrieb.

Anhand von zwei Beispielen wird die Anwendbarkeit der entwickelten Methoden am Verbrennungsmotorenprüfstand gezeigt.

Abstract

This thesis describes methods to calculate online design of experiments and to identify online the dynamic behaviour of combustion engines. These methods are developed to support the electronic control unit calibration in terms of modelling and optimizing the dynamics of the combustion engine. The results are used to extend an existing model-based online optimization algorithm, which is used to optimize the steady state behaviour of the combustion engine, such that this algorithm can also be used to optimize the transient behaviour of the combustion engine.

The thesis introduces new model-based online designs of experiments and online-identification algorithms. The model-based online design of experiments allows gathering maximum information with minimum measurements, to consider constraints on the inputs and outputs and to optimize the design of experiment in respect to the indented use of the models. The online-identification enables to identify continuous time, input-to-state stable state-space models using neural networks, which are disretized by Runge-Rutta methods. It is possible to show, that the proposed online design of experiment and the online identification lead to better model quality and are superior to state of the art methods.

Two examples show the application of the proposed methods at the testbed.

Vorwort

Die Arbeit entstand während meiner Zeit als Doktorand im Bereich „Entwicklung Versuchsmethoden“ der BMW Group in München in Zusammenarbeit mit dem Lehrstuhl für Regelungstechnik der Technischen Universität München.

An dieser Stelle möchte ich mich bei Prof. Dr.-Ing. Lohmann, Dr.-Ing. Michael Buhl und Dr.-Ing. Florian Klöpper für die fachliche Betreuung und wertvollen Diskussionen während der Entstehung dieser Arbeit bedanken. Zudem danke ich den Studenten Martin Ronis, Susanne Zaglauer und Matthias Moll für ihren Einsatz und Unterstützung.

Besonderer Dank gilt meiner Familie und vor allem auch meiner Freundin für die fortwährende Unterstützung und Geduld!

München, August 2011 Michael Deflorian

Inhaltsverzeichnis

Notation

Abkürzungen

APRBS	Amplitudenmoduliertes Pseudo Randomisiertes Binäres Signal
DoE	Design of Experiment
FIM	Fisher-Informationsmatrix
LHC	Latin HyperCube
LM	Levenberg-Marquardt
LOLIMOT	Local Linear Model Tree
MLP	Multi Layer Perceptron
NARX	Nonlinear AutoRegressive with eXogenous input
NMSE	Normalized Mean Square Error
NMAE	Normalized Mean Absolute Error
NOE	Nonlinear Output Error
QbC	Query by Committee
RBF	Radial Basis Function
RK	Runge-Kutta
RKNN	Runge-Kutta Neuronales Netz
SNR	Signal to Noise Ratio
TS	Takagi-Sugeno
ZDNN	Zeitdiskretes Neuronales Netz

Symbole

N_x	Anzahl an Zuständen
N_R	Anzahl an Regressoren
N_I	Anzahl an Eingängen
N_O	Anzahl an Ausgängen
N_N	Anzahl an Neuronen in der versteckten Schicht
N_p	Anzahl an Parametern
u	Eingang
$x, \hat{x}$	System- / Modellzustand
$y, \hat{y}$	System- / Modellausgang
θ	Modellgewichte / -parameter
φ	Regressor
η	Messrauschen
e	Abweichung zwischen Modell- und Systemausgang
t_s	Abtastrate
h_{rk}	Runge-Kutta Schrittweite

1 Einleitung

Der Verbrennungsmotor ist ein komplexes dynamisches System, welches durch Verbrennung die chemische Energie eines Kraftstoffes in mechanische Leistung umwandelt. Der Ablauf der Verbrennung wird dabei durch zahlreiche steuerbare Größen, wie beispielsweise Zündzeitpunkt und eingespritzte Treibstoffmenge, kontrolliert. Über diese Stellgrößen ist es möglich den Verbrennungsvorgang zu beeinflussen und sicherzustellen, dass der Verbrennungsmotor in allen Betriebsarten, wie zum Beispiel Warmlauf und Notlauf, sicher und ohne Fehlzündungen (Zündaussetzer, Klopfen) betrieben werden kann. Zudem soll die Verbrennung möglichst optimal, das heißt mit hohem Wirkungsgrad ablaufen, um aus einer gegebenen Treibstoffmenge die maximale mechanische Leistung zu gewinnen und somit den Verbrauch zu minimieren. Des weiteren muss darauf geachtet werden, dass die gesetzlichen Vorgaben hinsichtlich der Emissionen (Stickoxide, Kohlenwasserstoffe, Rußwerte) erfüllt werden. Demzufolge sollen die Steuergeräte des Verbrennungsmotors dahingehend parametriert werden, dass die Verbrennung in allen Betriebszuständen verlässlich und mit hohem Wirkungsgrad abläuft und die Emissionen die gesetzlichen Rahmenbedingungen einhalten. Die gefundenen Parameter werden in Form von Kennwerten, Kennlinien, Kennfeldern und Funktionsrahmen in den Steuergeräten abgespeichert. Dieser Vorgang der Parametrierung der Steuergeräte wird auch Applikation genannt. Das Auffinden möglichst optimaler Parameterkombinationen ist ein hoch komplexer und zeitaufwendiger Vorgang, der aufgrund der genannten Freiheitsgrade bei der Steuerung der Verbrennung nur unter Verwendung moderner mathematischer Methoden beherrschbar ist. In den Vorgängerarbeiten [65], [48], [47], [82] sind Methoden erarbeitet worden, um Stationärwerte des Verbrennungsmotors zu modellieren und zu optimieren, wodurch eine (teil-) automatisierte Applikation möglich wird. Analog dazu ist das Ziel dieser Arbeit die Entwicklung eines Methodenbaukastens zur Modellierung dynamischer Zusammenhänge, der die Applikation bei bestimmten Aufgabenstellungen unterstützt.

1.1 Einführung in die Aufgabenstellung und Motivation

Wenn die Verbrennung nur durch ein oder zwei Stellgrößen beeinflusst wird, ist es dem erfahrenen Applikateur möglich durch Versuchs-und-Irrtums-Methoden zufriedenstellende Parameterkombinationen zu finden. Durch die in den letzten Jahren stark gestiegene Komplexität der Stellglieder des Verbrennungsmotors und die damit einhergehenden Freiheitsgrade ist mittlerweile ein systematisches und automatisiertes Vorgehen notwendig, um mit vertretbarem Zeitaufwand die Applikation durchführen zu können.

Da sich der Verbrennungsmotor durch seinen komplizierten mechanischen Aufbau und die komplexen chemischen Reaktionen während der Verbrennung mit aktuellen Methoden nur sehr schwer hinreichend genau beschreiben lässt, ist es nicht möglich ein physikalisches

Modell zu erstellen und dieses im laufenden Prüfstandsbetrieb zur Ermittlung optimaler Parameterkombinationen zu verwenden. Die aktuell verfügbaren Simulationsmodelle sind sehr zeitaufwändig in der Auswertung und daher für die Optimierung von Steuergerätegrößen gegenwärtig nicht einsetzbar.

Aktuelle Methoden der Motorapplikation erstellen daher messdatenbasierte Ersatzmodelle, die anschließend stellvertretend für den Verbrennungsmotor optimiert werden. Dabei erfolgt die Optimierung im wesentlichen in drei Schritten: Im ersten Schritt werden mittels Methoden der statistischen Versuchsplanung signifikante Messpunkte im Versuchsraum bestimmt und vermessen. Dabei ist nicht in allen Bereichen des Versuchsraums, der durch die variablen Stellgrößen aufgespannt wird, ein sicherer Betrieb des Verbrennungsmotors möglich. Bestimmte Stellgrößenkombinationen können zu Problemen bei der Verbrennung oder zu hohen Temperaturen beziehungsweise Drücken führen, die eine Beschädigung von Bauteilen verursachen können. Daher muss der Versuchsraum durch Limitmodelle, welche die Grenzen des sicheren Betriebs modellieren, eingeschränkt werden. In einem zweiten Schritt werden mit den gewonnenen Messdaten Modelle gebildet, die den Zusammenhang zwischen Stellgrößen und den zu optimierenden Größen (Zielgrößen) abbilden. Schließlich werden die Parameter anhand des Modells, das stellvertretend für die zu optimierende Zielgröße des Verbrennungsmotors steht, unter Berücksichtigung der Limitmodelle optimiert. Damit ist es möglich für verschiedene Betriebsbereiche optimale Stellgrößenkombinationen zu finden, die anschließend am Prüfstand verifiziert werden.

Die bisher eingesetzten Verfahren der Modellbildung basieren auf Stationärmessungen und eignen sich ausschließlich zur Erstellung *stationärer* Modelle. Werden hingegen *dynamische* Zusammenhänge modelliert, welche die zeitliche Abhängigkeit der Messgrößen berücksichtigen, versprechen sich neue Möglichkeiten in der Applikation zu eröffnen. Diese betreffen die Modellbildung, Limitmodellierung und Optimierung.

Modellbildung
Dynamische Modelle erlauben, neben der modellbasierten Optimierung des dynamischen Motorverhaltens, das Modellwissen für weitere Applikationsaufgaben zu verwenden. So können die dynamischen Modelle zum Entwurf von Reglern oder als virtuelle Sensoren verwendet werden. Daneben eignen sie sich auch für den Einsatz in klassischen Applikationsaufgaben, welche sich der Stationärmessung bedienen. Da sich viele Messgrößen, wie zum Beispiel die Abgastemperatur, erst nach mehreren Minuten nicht mehr ändern, liegt ein wesentlicher Zeitanteil der Applikation in der zeitaufwendigen Stationärvermessung. Dynamische Modelle erlauben es den Stationärzustand der Messgröße abzuschätzen, womit auf die Stationärmessung verzichtet und somit Messzeit eingespart werden kann.

Limitmodellierung
Die Limitmodellierung schränkt den Versuchsraum ein und modelliert Grenzen an die Variationsgrößen. Tritt im laufenden Betrieb beispielsweise eine Verletzung definierter Grenzen an die Laufruhe oder den Abgasdruck auf, werden die Stellgrößenkombinationen, welche zu dieser Grenzwertverletzung geführt haben, in der Limitmodellierung berücksichtigt. Im Gegensatz zu Grenzwertverletzungen, die durch eine ungleichmäßige Verbrennung entstehen, treten Grenzwertverletzungen, die durch zu hohe Abgastemperaturen verursacht werden, erst zeitlich verzögert auf. Die Limitmodellierung von Abgastemperaturen erlaubt somit den Abbruch einer Stationärmessung bevor es aus Bauteilschutzgründen zu einer

Notabschaltung des Prüfstands kommt. Dazu sind dynamische Limitmodelle nötig, welche den zeitlichen Verlauf der Abgastemperaturen vorhersagen können.

Optimierung
Statische Modelle erlauben ausschließlich die Optimierung des Stationärverhaltens des Verbrennungsmotors. Somit ist die Optimierung von dynamischen Fahrsituationen nur indirekt möglich. Dynamische Modelle ermöglichen eine Optimierung des Instationärverhaltens des Verbrennungsmotors, womit optimale Parameterkombinationen für beliebige Fahrmanöver gefunden werden können.

Diese genannten Vorteile in der Modellbildung, Limitmodellierung sowie Optimierung motivieren den Einsatz dynamischer Modelle. Es sollen in dieser Arbeit die Einflüsse der Verwendung von dynamischen Modellen auf die Versuchsplanung und Modellbildung untersucht und ein Methodenbaukasten zur Versuchsplanung und Identifikation dynamischer Modelle entwickelt werden.

1.2 Stand der Technik in der Applikation

Die Aufgabe der Motorsteuergeräteapplikation besteht darin, die Steuergeräte des Verbrennungsmotors so zu parametrieren, dass dieser in allen Betriebsbereichen hinsichtlich bestimmter Kriterien optimal läuft. Dabei unterscheiden sich die Optimalitätskriterien abhängig vom Betriebsbereich. So steht in der Teillast die Minimierung von Verbrauch und Emissionen im Vordergrund, während es in der Volllast die Leistung zu maximieren gilt. Im Leerlauf steht hingegen die Laufruhe für bestmöglichen Komfort im Vordergrund. Die Optimierung erfolgt innerhalb des Betriebsbereiches $\mathcal{X}_b$ des Verbrennungsmotors, der durch Drehzahl und Last aufgespannt wird. Dabei wird dieser durch die Leerlauf- beziehungsweise Abregeldrehzahl und die Null- beziehungsweise Volllast begrenzt (Abbildung 1.1). Festgelegte Drehzahl-Last Kombinationen werden als Betriebspunkte x_b bezeichnet. Erfolgt die Optimierung der Zielgrößen y für einen festen Betriebspunkt, wird von einer *lokalen* Optimierung gesprochen. Fließen Drehzahl und Last mit in die Optimierung ein, wird dies als *globale* Optimierung bezeichnet. Die optimalen Stellgrößenkombinationen für einen Betriebspunkt x_b werden dabei innerhalb eines Variationsraums $\mathbb{U}_V$ gesucht, der durch physikalische Einschränkungen (zum Beispiel Klopfen, Laufruhe) auf einen fahrbaren Bereich $\mathbb{U}_F \subseteq \mathbb{U}_V$ eingeschränkt wird. Die für die betrachteten Betriebspunkte gefundenen Stellgrößenkombinationen werden anschließend mittels Kenngrößen, Kennlinien und Kennfeldern im Steuergerät hinterlegt. Um möglichst glatte Kennfelder zu erreichen, werden oftmals die Ergebnisse der Optimierung nachträglich angepasst [61]. Glatte Kennfelder stellen sicher, dass die Stellbewegung der Aktuatoren bei Betriebspunktwechseln gering bleibt und somit das Material geschont und ein Maximum an Laufruhe sowie Komfort erreicht wird.

Bei den Optimierungsverfahren kann zwischen messdatenbasierter und modellbasierter Optimierung unterschieden werden. Bei komplexen Applikationsaufgaben kommt zur Optimierung des Stationärverhaltens in der Regel die modellbasierte Optimierung zum Einsatz, während die messdatenbasierte Optimierung nur noch in Ausnahmefällen angewandt wird.

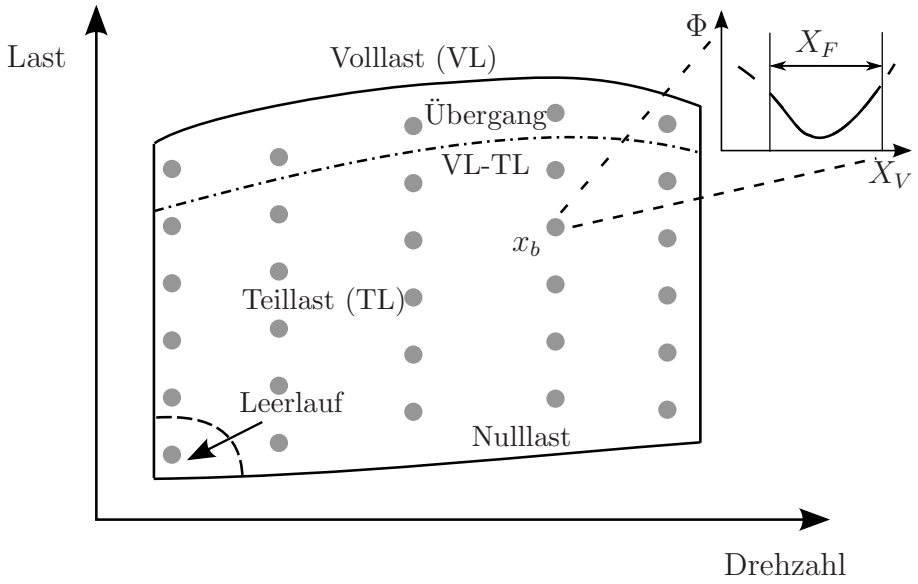


Abbildung 1.1: Umfang und Aufgabenbereiche der Applikation.

Bei allen im folgenden beschriebenen Optimierungsverfahren wird der Verbrennungsmotor im eingeschwungenen respektive *stationären* Zustand vermessen.

1.2.1 Messdatenbasierte Optimierung

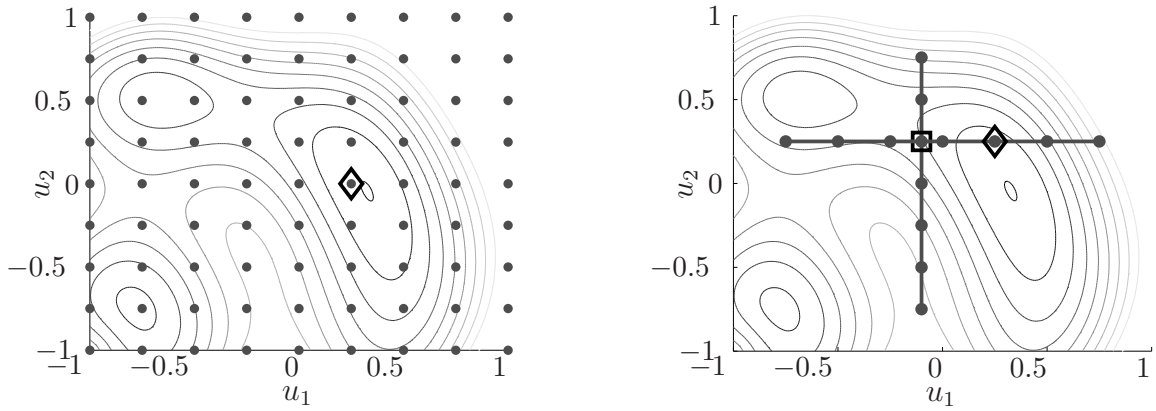
Wenn die Optimierung in einem niedrigdimensionalen Variationsraum (2-3 Dimensionen) durchgeführt wird, kamen und kommen häufig messdatenbasierte Optimierungsmethoden zum Einsatz. Zum Einem sind diese einfach umzusetzen und zum Anderen sehr anschaulich. Bei der Optimierung moderner Verbrennungsmotoren mit ihrer Vielzahl an Variationsparametern sind sie hingegen kaum anwendbar. Auf die messdatenbasierte Optimierung wird im Folgenden nur kurz eingegangen und für eine weiterführende Diskussion derselben auf [47] verwiesen. Die beiden hier vorgestellten Verfahren sind bei gewissen Aufgabenstellungen noch gebräuchlich und werden der Vollständigkeit halber kurz beschrieben.

Vollfaktorielle Optimierung

Bei dieser Methode wird ein vollfaktorieller Versuchsplan, das heißt ein Vollraster im Variationsraum, erzeugt und der Variationsraum vermessen. Die Anzahl der zu vermessenden Punkte N und somit auch der Versuchsaufwand wird dabei durch die Anzahl der Faktorstufen FS und die Anzahl der Faktoren d, bestimmt:

$$N = FS^d.$$

Das Optimum lässt sich bei der vollfaktoriellen Vermessung durch Sortieren der Messwerte auffinden. Abbildung 1.2(a) verdeutlicht das Vorgehen an einem zweidimensionalen Variationsraum. Die Isolinien geben den Verlauf der Zielgröße wieder, die grauen Punkte die Messpunkte des Vollrasters. Der Messpunkt mit minimalem Wert ist durch eine Raute gekennzeichnet. Da der Versuchsaufwand exponentiell mit der Dimension des Variationsraums ansteigt, ist die vollfaktorielle Optimierung für $d > 2$ nicht mehr praktikabel. Zudem



(a) Vollrastervermessung (b) One Factor at a time

Abbildung 1.2: Messdatenbasierte Optimierungsverfahren

hängt die Lage des Optimums von der gewählten Rasterung des Variationsraums ab, was methodisch gesehen unbefriedigend ist.

One Factor at a Time

Bei diesem Verfahren wird jede Variationsgröße einzeln angepasst, siehe Abbildung 1.2(b). Der Reihe nach wird jeweils eine Variationsgröße zur Optimierung verwendet, während die anderen Variationsgrößen konstant gehalten werden. Bei Bedarf wird das Verfahren mehrmals wiederholt. Da immer entlang einer Dimension das lokale Optimum gesucht wird, kann die Wechselwirkung mehrerer Parameter nicht erfasst werden. Dies führt vor allem bei hochdimensionalen Variationsräumen nur nach einer Vielzahl von Messungen zu befriedigenden Ergebnissen. Daher scheidet auch dieses Verfahren bei komplexeren Optimierungsaufgaben aus.

1.2.2 Modellbasierte Optimierung

Von einer modellbasierten Optimierung wird gesprochen, wenn auf Grundlage von Messungen Ersatzmodelle erstellt werden, die anschließend stellvertretend für das System optimiert werden. Die modellbasierte Optimierung kann dabei in Verfahren der Offline- und Online-Optimierung unterteilt werden. Bei der Offline-Optimierung findet eine strikte Trennung zwischen Messdatenerfassung und Modellbildung statt. In einem ersten Schritt erfolgen die gesamten Messungen und in einem zweiten nachgelagerten Schritt die Modellbildung und Optimierung. Da keine Interaktion zwischen dem System und der Modellbildung erfolgt, wird von Offline-Modellbildung und Optimierung gesprochen. Bei der Online-Optimierung werden die Modelle hingegen mit jedem neu vorliegenden Datensatz angepasst und somit ist auch eine Interaktion zwischen Modellbildung, Optimierung und Messdatenerfassung möglich. In diesem Zusammenhang wird oft von adaptiver Modellbildung und Versuchsplanung gesprochen. In dieser Arbeit wird der Begriff Online bevorzugt, um die permanente Interaktion mit der Prüfstandsumgebung zu unterstreichen. Durch Berücksichtigung der auf Grundlage der anfallenden Daten erstellten Modelle in der

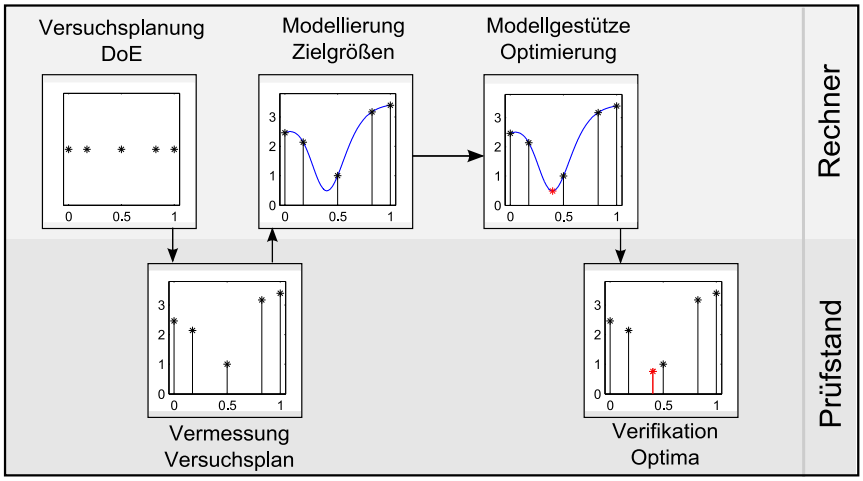


Abbildung 1.3: Ablauf modellbasierte Offline-Optimierung.

Versuchsplanung, kann eine bessere Modellqualität und somit auch ein besseres Optimierungsergebnis erzielt werden [47].

Modellbasierte Offline-Optimierung

Bei der Offline-Optimierung erfolgt die Messdatenerfassung und die Modellbildung in getrennten Schritten. Dies schlägt sich, wie in Abbildung 1.3 schematisch dargestellt, auch im Ablauf der Optimierung nieder. Zunächst wird am Rechner ein Versuchsplan (*engl. DoE: Design of Experiments*) erzeugt. Anschließend erfolgt mit den festgelegten Parameterkombinationen eine Vermessung am Prüfstand. Basierend auf den Messdaten werden am Rechner für die gewünschten Zielgrößen Modelle erzeugt. Anhand dieser Modelle lassen sich mit Hilfe von Optimierungsverfahren Eingangskombinationen finden, welche die vom Modell abgebildete Zielgröße minimieren beziehungsweise maximieren. Die gefundenen Optima werden anschließend durch Messungen am Prüfstand verifiziert. Mit diesen verifizierten Optima findet schließlich die Kennfeldbedatung statt. Ein wesentlicher Nachteil der Offline-Optimierung ist die fehlende Kontrolle der Modellgüte während des Testlaufs. Ob die Modelle für die Aufgabenstellung genau genug sind, lässt sich erst anhand der Verifikationsmessungen beurteilen, die am Ende der Offline-Optimierung erfolgen. Sind die gefundenen Optima nicht zufriedenstellend, muss teilweise von vorne begonnen und mit einem neu erstellten Versuchsplan eine weitere Vermessung gestartet werden. Diesen Nachteil behebt die modellbasierte Online-Optimierung.

Modellbasierte Online-Optimierung

Bei der modellbasierten Online-Optimierung wird die Trennung zwischen Prüfstands- und Auswerteumgebung aufgehoben. Der Optimierungsalgorithmus steht somit in permanenter Interaktion mit dem Prüfstand. Im Gegensatz zur Offline-Optimierung, bei der die Erstellung des Versuchsplans und die Versuchsdurchführung vom der nachfolgenden Modellierung und Optimierung unbeeinflusst sind, wird bei der Online-Optimierung die Versuchsdurchführung durch die Modellierung und Optimierung bestimmt. Auf diese Weise ist es möglich, Messpunkte so zu platzieren, dass die Modelle im laufenden Versuch bestmöglich verfeinert und angepasst werden. Die aufgefundenen Optima können zudem direkt verifiziert und die Modelle bei Bedarf in deren Nähe verfeinert werden.

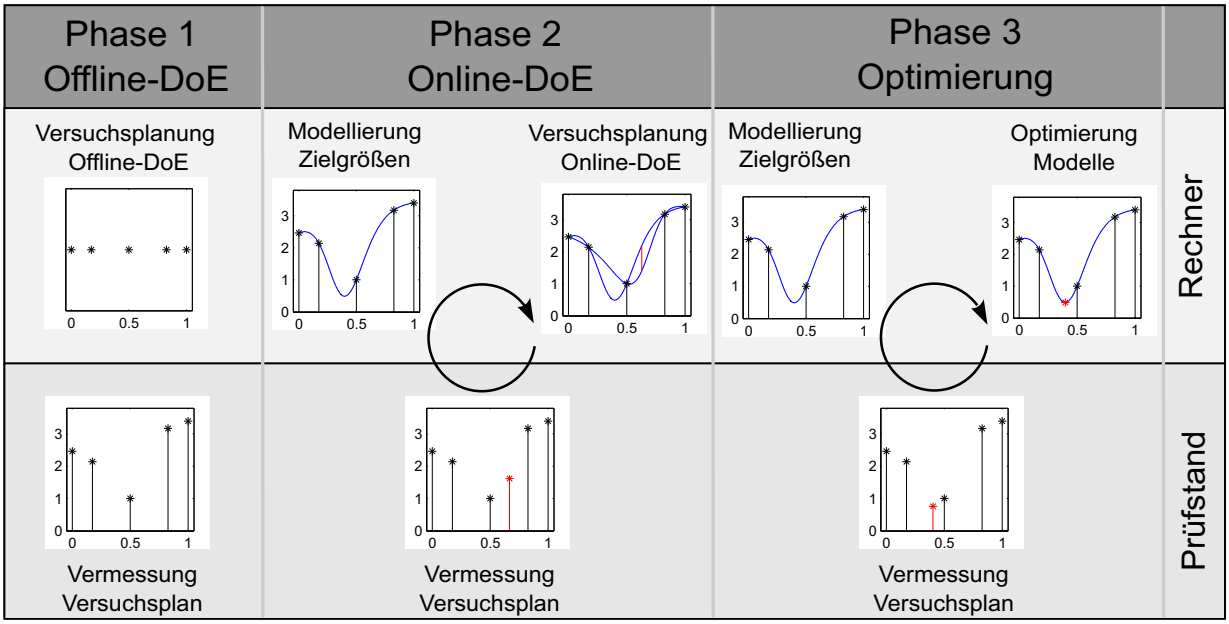


Abbildung 1.4: Ablauf modellbasierte Online-Optimierung.

Die modellbasierte Online-Optimierung läuft, wie in Abbildung 1.4 dargestellt, in drei Phasen ab. In einer ersten Phase werden Offline-Versuchspläne erzeugt und der Eingangsraum vermessen. Mit diesen Messdaten werden anschließend die initialen Modelle erzeugt. Basierend auf diesen erfolgt in einer zweiten Phase eine adaptive Versuchsplanung, welche die weiteren Messpunkte so platziert, dass die vorliegenden Modelle bestmöglich verfeinert werden. Dabei erfolgt nach jeder Messung eine Anpassung der Modelle und der Versuchsplanung. In einer dritten und letzten Phase werden anhand der Modelle Optima ermittelt und in der Nähe derselben weitere Messpunkte platziert, um die Modelle in diesen Bereichen nochmals zu verfeinern.

Durch die modellbasierte Online-Optimierung kann der Versuchsaufwand bei gleichbleibender Optimierungsgüte erheblich reduziert werden [47]. Das größte Potential, um die Prüfstandszeit weiter zu reduzieren, liegt nun in den verwendeten Messverfahren.

1.2.3 Messdatenerfassung

Das erzielte Optimierungsergebnis hängt entscheidend von der Qualität der Messungen ab. Verrauschte Messwerte führen zu Unsicherheiten in der Modellbildung, die im Allgemeinen nur durch eine größere Anzahl an Messungen ausgeglichen werden können. Auf diese Problematik wird im Kapitel 2, welches die Grundlagen der Modellbildung behandelt, näher eingegangen. Um den Versuchsaufwand gering zu halten, sollten daher die Messwerte möglichst unverrauscht sein und die Messzeit pro Messpunkt kurz gehalten werden. Je nach Art der Vermessung können diese beiden Forderungen konträr sein. Bei der Messdatenerfassung kann zwischen stationärer Messung und instationärer Messung unterschieden werden.

Stationäre Vermessung

Die Zielgrößen des Verbrennungsmotors werden meist hinsichtlich dessen Stationärverhaltens optimiert. Daher ist es naheliegend auch bei der Messung darauf zu achten, dass die Messgrößen stationär sind. Da die Messgrößen immer mit Rauschen behaftet sind, werden mehrere Messungen benötigt, damit durch Mittelung der Messgrößen die Effekte des Rauschens unterdrückt werden können. Dabei wird mit steigender Anzahl an Messwerten, die zur Mittelung zur Verfügung stehen, das Ergebnis besser. Dies widerspricht allerdings der Forderung nach möglichst kurzer Messzeit. In Abbildung 1.5 ist schematisch der zeitliche Verlauf einer verrauschten Messgröße dargestellt. Die Führungsgröße ist meist eine Regelgröße (zum Beispiel Drehmoment), welche innerhalb der Regelzeit (RZ) eingeregelt wird. Dementsprechend verzögert folgt die Zielgröße. Ist diese nach der Stabilisierungszeit (SZ) stationär, kann mit der Messung begonnen werden. Anschließend werden noch innerhalb der Mittelungszeit (MZ) die anfallenden Messwerte gewichtet (zum Beispiel Mittelwert oder Median bestimmt), um das Rauschen zu unterdrücken.

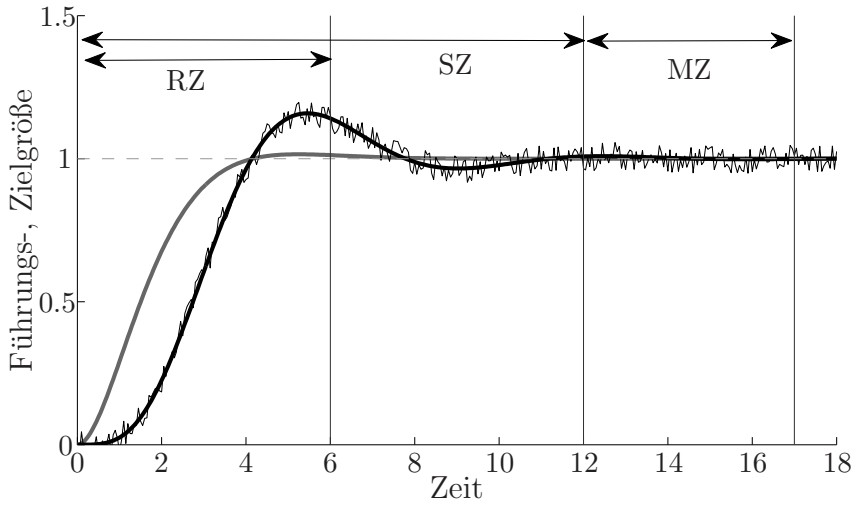


Abbildung 1.5: Stationärmessung: Regel- (RZ), Stabilisierungs- (SZ) und Mittelungszeit (MZ).

Die Messzeit pro Betriebspunkt setzt sich somit aus der Stabilisierungszeit (SZ) und der Mittelungszeit (MZ) zusammen. Erstere muss abgewartet werden, damit die Messgrößen stationär sind, und letztere um den störenden Einfluss des Rauschens durch Mittelung der Messwerte unterdrücken zu können. Je nach betrachteter Messgröße kann die Stabilisierungszeit bis zu mehreren Minuten in Anspruch nehmen (zum Beispiel Abgastemperaturen). Pro Messpunkt muss folglich mit einigen Minuten Messzeit gerechnet werden, wobei die während der Einschwingzeit anfallenden Messdaten nicht zur Modellbildung verwendet werden können. Um die zeitaufwändigen Stationärmessungen zu vermeiden, sind in den letzten Jahren mehrere instationäre Vermessungsstrategien entwickelt worden.

Instationäre Vermessung

Bei der instationären Vermessung wird nicht darauf geachtet, dass die Messgrößen stationär sind. Daher muss die Dynamik der Messgrößen berücksichtigt werden. Je nach Ansatz müssen entweder die Messwerte um den Einfluss der Dynamik bereinigt oder ein dynami-

sches Modell gebildet werden. Den ersten Ansatz zur Bildung *statischer* Modelle verfolgen die folgenden Arbeiten:

- Sweepen [75]: Bei dieser Verstellstrategie wird der Eingangsraum quasistationär vermessen und das System mittels einer Doppelrampe angeregt. Die dynamischen Anteile der Messung werden durch Mittelung der Hin- und Rückmessung unterdrückt, womit die Stationärmessung approximiert wird.
- Rapid Measurement [72]: Die Versuchsplanung erfolgt wie bei der Stationärvermessung. Allerdings wird nicht gewartet bis die Zielgröße stationär ist, sondern es wird Online die lokale Dynamik modelliert, um den Stationärwert zu prädizieren.
- Slow Dynamic Slope [45]: Die Verstellung erfolgt mit einer sehr langsamen Rampenfunktion, so dass die Zielgrößen mit schneller Dynamik (zum Beispiel Moment) quasistationär bleiben. Das Verfahren ist allerdings für langsame Größen wie Abgastemperaturen ungeeignet.

In diesen Arbeiten wird die Versuchsplanung derart gestaltet, dass die Dynamik des Systems möglichst wenig angeregt wird. Der Einfluß der Systemdynamik wird durch Modellansätze oder Mittelungsverfahren bestimmt und anschließend direkt auf den Stationärwert geschlossen.

Der zweite Ansatz besteht darin die Dynamik des Systems gezielt anzuregen und ein *dynamisches* Modell zu identifizieren. Anhand des globalen dynamischen Modells kann auf den Stationärwert geschlossen werden. Dieses Vorgehen verfolgen beispielsweise die Arbeiten in [64] und [3]. Allerdings erfolgt die Versuchsplanung und die Modellbildung offline. Im Rahmen dieser Arbeit sollen erstmals Online-Versuchspläne verwendet werden, um die vorgestellten Vorteile der adaptiven Versuchsplanung nützen zu können.

Gemeinsam ist den Ansätzen, dass alle anfallenden Messwerte in der Modellbildung berücksichtigt werden, wodurch ein erheblich größerer Datensatz zur Modellbildung verfügbar ist. Diesem positiven Effekt steht der Nachteil eines wesentlich komplexeren Modells gegenüber, da nun die Zeitabhängigkeit der Zielgröße mit berücksichtigt werden muss.

1.3 Ziele der Arbeit

Das Ziel der Arbeit besteht darin einen Methodenbaukasten zu entwickeln, welcher es erstmals erlaubt im laufenden Betrieb *dynamische Modelle* zu erstellen und diese bereits während der Modellbildungsphase hinsichtlich ihres späteren Einsatzzweckes zu optimieren. Der eingangs vorgestellte Ablauf der modellbasierten Online-Optimierung des *stationären* Verhaltens ist der aktuelle Stand der Technik und hat sich in der Praxis bewährt. In dieser Arbeit soll daher die Online-Modellbildung und deren Ablauf übernommen und an die Anforderungen der Modellierung dynamischer Modelle angepasst werden. Betrachtet man die dynamischen Modelle als Ersatz für die statischen Modelle, können diese einerseits verwendet werden, um zeitaufwändige stationäre Messungen zu vermeiden und andererseits um den zeitlichen Verlauf der Zielgröße zu prädizieren. Dadurch kann sowohl der Messprozess schneller und durch Prädiktion von Limitverletzungen

(zum Beispiel Notabschaltung aus Bauteilschutzgründen bei zu hohen Abgastemperaturen oder Drücken) stabiler gestaltet werden. Die dynamischen Modelle eröffnen zudem die Möglichkeit neben dem Stationär- auch das Instationärverhalten des Verbrennungsmotors zu modellieren und zu optimieren. Dies führt neben der modellbasierten Optimierung des Instationärverhaltens zu gänzlich neuen Möglichkeiten in der Applikation. Applikationsmethoden und -werkzeuge basierend auf dynamischen Modellen, wie beispielsweise der Einsatz modellbasierter Reglerentwurfstechniken oder der Einsatz dynamischer Modelle als virtuelle Sensoren, versprechen einen großen Innovationsschub in der Applikation, wohingegen die Methoden zur Modellierung und Optimierung des stationären Motorverhaltens weitgehend ausgereizt sind.

Aus dieser Aufgabenstellung ergeben sich folgende Zielsetzungen:

- Modellbildung: Es existieren zahlreiche Ansätze zur Erstellung dynamischer Modelle. Neben dem exakten Abbilden des dynamischen Verhaltens, ergeben sich aus der Aufgabenstellung noch weitere Anforderungen an das Modellbildungsverfahren. Damit die Optimierung hinsichtlich des Stationärverhaltens erfolgen kann, muss das Modellbildungsverfahren in der Lage sein, neben dem Instationärverhalten auch das Stationärverhalten exakt abzubilden. Die Modelle sollten zudem schnell, das heißt mit wenig Rechenaufwand, auswertbar sein, da die adaptive Versuchsplanung eine Vielzahl an Modellauswertungen benötigt. Zudem ist es entscheidend, dasss die Modellbildungsverfahren robust gegenüber Störeinflüssen wie zum Beispiel Messrauschen und robust gegenüber der Parametrierung sind. Idealerweise passen sich die Modelle weitgehend selbständig der Komplexität der Aufgabenstellung an. Nur so ist es gewährleistet, dass ein einfacher Einsatz im Prüfstandsbetrieb möglich ist. Bestehende Modellbildungsverfahren sollen hinsichtlich dieser Kriterien ausgewählt und gegebenenfalls angepasst werden.

- Versuchsplanung: Bei der Anregung dynamischer Systeme muss darauf geachtet werden, dass diese im relevanten Frequenzbereich erfolgt, so dass die Dynamik dort gut erfasst werden kann. Zudem ist es erforderlich alle relevanten Bereiche im Eingangsraum abzudecken, damit es bei der Modellauswertung nicht zur Extrapolation kommt. Darüber hinaus muss die Anregung so erfolgen, dass eventuell auftretende Limitverletzungen erkannt und behoben werden können, bevor es zu einer Beschädigung des Prüflings kommt. Unterschiedliche Anregungsmethoden sollen untersucht und hinsichtlich ihrer Eignung überprüft und gegebenenfalls angepasst werden. Außerdem soll die Versuchsplanung hinsichtlich des späteren Einsatzzwecks der Modelle ausgelegt werden, so dass beispielsweise die Modelle möglichst exakt in der Nähe potentieller stationärer beziehungsweise instationärer Optima sind. Zudem soll, wie auch bei der Modellbildung gefordert, die erzielte Güte der Daten robust sein gegenüber der Parametrierung der Versuchsplanung.

- Optimierung: Die Optimierung soll auch hinsichtlich des Stationärverhaltens erfolgen. Dementsprechend muss sichergestellt werden, dass neben dem Instationär- auch das Stationärverhalten durch das Modell hinreichend genau abgebildet wird. Ist dies gewährleistet, können die Methoden der Optimierung übernommen werden, die erfolgreich in [47] eingesetzt werden. Die während der Online-Optimierung anfallenden Ergebnisse sollen dabei in die Versuchsplanung mit einfließen.

1.4 Aufbau der Arbeit

Der Aufbau der Arbeit gliedert sich in 7 Kapitel. In den Kapiteln 2 bis 4 werden die Grundlagen der Identifikation dynamischer Systeme und der Versuchsplanung behandelt. Dabei ist die Einführung so gestaltet, dass die Abschnitte, welche für die nachfolgenden Kapitel von wesentlicher Bedeutung sind, ausführlicher behandelt werden. Abschnitte, welche der Beschreibung alternativer, für die Aufgabenstellung weniger relevanter Verfahren dienen, sind der Vollständigkeit halber aufgeführt und unter Verweis auf weiterführende Literatur knapp gehalten. Die Kapitel 5 und 6 stellen den Kern der Arbeit dar, wobei ersteres die erarbeiteten Methoden der Versuchsplanung und Modellbildung vorstellt und letzteres deren Anwendung in der Praxis behandelt. Abschließend werden die anhand von Simulationen und Prüfstandsmessungen erzielten Ergebnisse diskutiert und ein Ausblick gegeben.

In Kapitel 2 wird auf die Grundlagen der Modellbildung und deren wesentlichen Probleme eingegangen. Neben der Darstellung der Systemdynamik wird das Bias-Varianz-Dilemma, das grundlegende Problem der Modellbildung, eingeführt und verschiedene Arten der Modellbildung behandelt. Dabei liegt der Schwerpunkt in der Einführung in die Thematik ohne Details zu behandeln.

In Kapitel 3 und 4 wird auf den aktuellen Stand der Technik in der Versuchsplanung und Modellbildung eingegangen. Der Fokus bei der Versuchsplanung liegt speziell auf der Anwendbarkeit in der Praxis, das heißt beim Einsatz am Verbrennungsmotorenprüfstand. Bei der Modellbildung erfolgt die Auswahl der vorgestellten Verfahren und Methoden einerseits hinsichtlich deren bereits erfolgtem Einsatz in der Praxis und andererseits hinsichtlich derer theoretischen Eignung. Die in diesen beiden Kapiteln vorgestellten Versuchsmethoden und Modellbildungsverfahren werden im Hinblick auf Zielsetzung der vorliegenden Arbeit untersucht und eventuelle Defizite diskutiert.

Kapitel 5 stellt den Kern der Arbeit dar, in dem ein neues Modellbildungsverfahren und neue Kriterien zur Versuchsplanung vorgestellt werden. Zunächst erfolgt die Behandlung der Modellbildung, da die Modelle die Grundlage der modellbasierten Versuchsplanung darstellen. Es wird das Runge-Kutta Neuronale Netz vorgestellt und Lernverfahren entwickelt, die deren Einsatz in der Praxis ermöglichen. Dabei werden Kriterien an die Parameter des Netzes formuliert, welche die Stabilität des Modells in der Simulation sicherstellen. Bei der Versuchsplanung werden erstmals Online-Verfahren vorgestellt, welche es erlauben, die Systemanregung hinsichtlich des späteren Einsatzzweckes der Modelle zu optimieren. Mit den in diesem Kapitel entwickelten Methoden der Modellbildung und Versuchsplanung ist es erstmals möglich den in den Zielsetzungen formulierten Anforderungen an die Modellbildung, Versuchsplanung und Optimierung zu entsprechen.

In Kapitel 6 wird die Anwendung der in Kapitel 5 entwickelten Methoden der Versuchsplanung und Modellbildung am Motorprüfstand behandelt. Es wird der Ablauf der Online-Modellbildung am Prüfstand unter Berücksichtigung von Limits erläutert und die nötigen Heuristiken zur Umsetzung in der Praxis vorgestellt.

Im letzten Kapitel werden die erzielten Ergebnisse hinsichtlich der Aufgabenstellung diskutiert und mögliche Anwendungsgebiete der dynamischen Modelle bei der Applikation von Verbrennungsmotoren sowie noch offene Fragestellungen aufgezeigt.

2 Grundlagen der Modellbildung

Die Erstellung eines Modells, welches das Systemverhalten abbildet, ist die Grundvoraussetzung zur Behandlung der eingangs formulierten Aufgabenstellungen. Erst ein hinreichend genaues Modell des Systems ermöglicht das Modell stellvertretend für das System auszuwerten beziehungsweise zu optimieren. In diesem Zusammenhang wird der Begriff *Modell* nach folgender Definition gebraucht:

> Unter einem Modell versteht man allgemein ein (vereinfachtes) Abbild einer (partiellen) Realität [8].

In dieser Arbeit soll das nichtlineare *dynamische* Verhalten des Verbrennungsmotors modelliert werden. Da es sich hier um das Modell eines unbekannten Systems handelt, entspricht der Begriff der Modellbildung dem der Systemidentifikation und wird im Folgenden synonym verwendet. Dabei wird angenommen, dass sich das zeitkontinuierliche dynamische Verhalten des zu identifizierenden Systems im Zustandsraum

$$\dot{x}(t) = f_s\left(x(t), u(t)\right) \tag{2.1a}$$
$$y(t) = h_s\left(x(t), u(t)\right) + \eta(t) \tag{2.1b}$$

beschreiben lässt, wobei $t \geq t_0 \geq 0$ und $x(t_0) = x_0$. Es wird vorausgesetzt, dass die Eingänge $u(t) \in \mathbb{R}^{N_I}$ und Ausgänge $y(t) \in \mathbb{R}^{N_O}$ des Systems bekannt beziehungsweise messbar sind, wobei die Systemausgänge einem Messrauschen $\eta(t)$ unterliegen. Die Zustände $x(t) \in \mathbb{R}^{N_x}$ und die Funktionen $f_s : \mathbb{R}^{N_x} \times \mathbb{R}^{N_I} \to \mathbb{R}^{N_x}$ und $h_s : \mathbb{R}^{N_x} \times \mathbb{R}^{N_I} \to \mathbb{R}^{N_O}$ sind im Allgemeinen nicht oder nur teilweise bekannt. Da die Messungen der Zielgrößen mittels digitaler Messgeräte erfolgt, sind die vorliegenden Daten immer zeitdiskret und das zeitkontinuierliche System (2.1) kann bei geeigneter Abtastzeit $t_s > 0$ durch das zeitdiskrete System

$$x_{k+1} = f_d\left(x_k, u_k\right) \tag{2.2a}$$
$$y_k = h_d\left(x_k, u_k\right) + \eta_k \tag{2.2b}$$

beschrieben werden. Dabei gilt der Zusammenhang $u(kt_s + t_0) = u(t_k) = u_k$ und $h_d(\cdot) = h_s(\cdot)$, $f_d(\cdot) = \int_{t_k}^{t_{k+1}} f_s(\cdot)\,\mathrm{d}t$, $k \in \mathbb{N}$. Aufgrund der zeitdiskreten Messungen entscheidet die Abtastrate, ob die Messdaten als quasikontinuierlich betrachtet werden können oder nicht.

Das Systemidentifizierungsproblem kann nun wie folgt definiert werden: Es liegen die beobachteten Eingänge $u \in \mathbb{R}^{N_I}$ und die (verrauschten) Messgrößen $y \in \mathbb{R}^{N_O}$ eines dynamischen Systems vor:

$$u^k = [u_1^\top, u_2^\top, \ldots, u_{k-1}^\top]^\top$$
$$y^k = [y_1^\top, y_2^\top, \ldots, y_{k-1}^\top]^\top,$$

wobei die Daten mit der Abtastrate t_s abgetastet werden. Es wird nach einem zu modellierenden Zusammenhang zwischen vergangenen Beobachtungen $[u^k, y^k]$ und zukünftigen Ausgang y_k gesucht:

$$y_k = f(u^k, y^k) + \nu_k. \tag{2.3}$$

Der additive Term ν_k berücksichtigt die Tatsache, dass der zu modellierende Zusammenhang keine exakte Funktion der vergangenen (verrauschten) Beobachtungen ist.

Um die Funktion (2.3) darstellen zu können, muss diese innerhalb einer Funktionenschar gefunden werden. Wird nun diese Funktionenschar über einen endlich dimensionalen Vektor θ, dessen Einträge auch als *Gewichte* bezeichnet werden, parametriert

$$\hat{y}_k = f(u^k, y^k, \theta), \tag{2.4}$$

entspricht dies in der Regel einer *Approximation*. Ist die Modellstruktur - auch Funktionsapproximator genannt - gewählt und ein Datensatz $[u^K, y^K]$ mit K Messungen vorhanden, kann θ durch Lösen des folgenden Optimierungsproblems bestimmt werden

$$\theta^* = \arg\min_{\theta} \left(\sum_{k=1}^{K} \left|\left| y_k - f(u^k, y^k, \theta) \right|\right| \right),$$

wobei die Norm und das verwendete Optimierungsverfahren je nach Aufgabenstellung variieren können. Die Optimierung der Parameter eines Funktionsapproximators wird in der Literatur häufig auch als *Training* bezeichnet. Der für das Training verfügbare Datensatz $[u^K, y^K]$ wird entsprechend *Trainingsdatensatz* genannt.

Um die mit k anwachsende Anzahl der Beobachtungen handhaben zu können, wird (2.4) üblicherweise durch zwei aufeinander folgende Abbildungen beschrieben [78]

$$\begin{aligned} \varphi_k &= \varphi(u^k, y^k) \\ f(\varphi_k, \theta) &= f(u^k, y^k, \theta), \end{aligned}$$

wobei die erste Abbildung dazu dient, die wachsende Anzahl an Beobachtungen in einem endlich dimensionalen Vektor $\varphi_k \in \mathbb{R}^{N_R}$ mit fixer Dimension abzubilden. Dabei wird φ_k als Regressorvektor und die Komponenten als Regressoren bezeichnet. Die Abbildung in den Regressorvektor kann auch dazu verwendet werden, die vorhandenen Eingänge zu transformieren. So ist es beispielsweise von Vorteil, physikalische Vorkenntnisse in die Wahl der Regressoren einfließen zu lassen. Wenn als Eingänge beispielsweise das Drehmoment M und die Drehzahl N zur Verfügung stehen und bekannt ist, dass die Zielfunktion von der Leistung abhängt, wird als Regressor $\varphi_k = M_{k-1}N_{k-1}$ gewählt werden.

Die Wahl des Regressors hängt zudem von der gewählten Darstellung der Dynamik ab, auf die im Folgenden eingegangen wird.

2.1 Darstellung der Dynamik

Um ein dynamisches Modell erzeugen zu können, muss vor der Modellbildung entschieden werden, wie die Zeitabhängigkeit der Zielgrößen berücksichtigt wird. Ein System n-ter Ordnung kann direkt als solches beschrieben oder in ein Gleichungssystem erster Ordnung überführt werden. Ersteres wird als Ein-/Ausgangsmodell und letzteres als Zustandsraummodell bezeichnet. Welches verwendet wird, hängt in der Regel davon ab, ob ein zeitdiskretes oder zeitkontinuierliches Modell erstellt werden soll.

Neben dem Modellansatz gilt es zudem zwischen unterschiedlichen Modell-Konfigurationen zu wählen. Je nach gewählter Konfiguration kann das Modell als Ein-Schritt-Prädiktor oder als Simulator trainiert werden. Von der Wahl der Konfiguration hängt auch entscheidend der Aufbau des Regressors ab, der als Modelleingang dient. Auf die unterschiedlichen Konfigurationen und Modellansätzen wird in den nächsten Abschnitten eingegangen.

2.1.1 Modell-Konfiguration

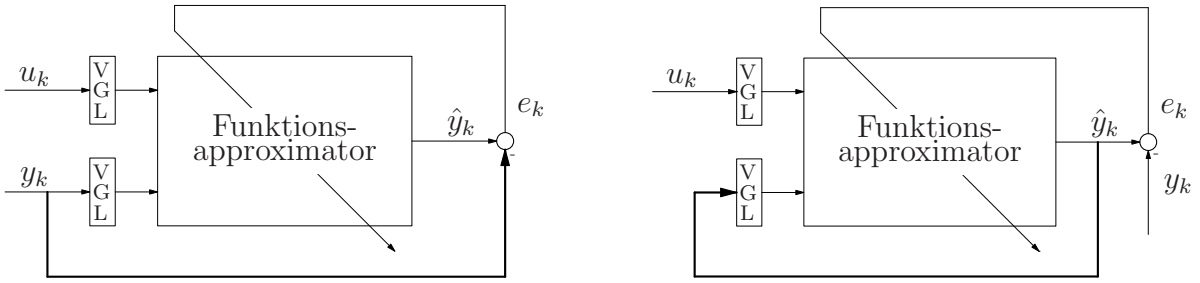
Unabhängig von der Betrachtung zeitkontinuierlicher und zeitdiskreter Modelle und dem verwendeten Modellansatz, gibt es zwei unterschiedliche Konfigurationen, in denen dynamische Modelle erstellt werden können. Zum einen die *Prädiktor*-Konfiguration, in der dem Modell neben den Systemeingängen die bisherigen Messwerte zur Verfügung stehen. Die zweite Konfiguration ist die *Simulator*-Konfiguration, in der dem Modell ausschließlich die Systemeingänge bekannt sind.

Prädiktor

Die Aufgabe eines Prädiktors besteht darin, auf Grundlage der bisherigen Messwerte und Systemeingänge die Messwerte y_k zu *prädizieren*. Die Dynamik wird in dieser Konfiguration durch die vergangenen Beobachtungen extern, das heißt außerhalb der Modellstruktur, erzeugt. Das Modell selbst ist statisch, da die Zeitabhängigkeit nicht im Modell abgebildet wird. Das Ziel eines Trainingsverfahrens ist die Minimierung des Ein-Schritt-Prädiktionsfehler des Modells. Eine schematische Darstellung des Signalflusses eines Modells in Prädiktor-Konfiguration ist in Abbildung 2.1(a) dargestellt.

Simulator

Ein Simulator ist ein frei laufendes Modell, bei dem kein Abgleich mit den Messwerten erfolgt. Die Simulator-Konfiguration und deren Signalfluss sind schematisch in Abbildung 2.1(b) dargestellt. Schon aufgrund der Modellierungsform ist klar, dass die Identifikation von Modellen in Simulator-Konfiguration durch die rekurrente Struktur des Modells erheblich komplizierter ist als die Identifikation von Modellen in Prädiktor-Konfiguration, bei denen keine rekurrenten Strukturen vorhanden sind. Beim Training eines Modells in Simulator-Konfiguration wird somit die Differenz zwischen Systemausgang und simulierten Modellausgang minimiert. Da ein Modell in dieser Konfiguration selbst ein dynamisches System darstellt, ist es wichtig während des Trainings dessen Ein-/Ausgangsstabilität sicherzustellen.



(a) Modell in Prädiktor-Konfiguration (b) Modell in Simulator-Konfiguration

Abbildung 2.1: Modellkonfigurationen mit Verzögerungsgliedern (VGL).

In dieser Arbeit sollen dynamische Modelle auch verwendet werden, um das Stationärverhalten des Verbrennungsmotors vorherzusagen. Um den Stationärwert eines dynamischen Modells zu ermitteln, muss dieses bei konstantem Eingang simuliert werden bis sich ein Stationärzustand einstellt. Wird ein Modell, welches in der Prädiktor-Konfiguration erstellt wurde, als Simulator verwendet, stellt die Fehlerakkumulation ein großes Problem dar, womit ein guter Prädiktor ein schlechter Simulator sein kann [60]. Daher liegt das Hauptaugenmerk der Arbeit in der Identifikation von dynamischen Modellen in Simulator-Konfiguration.

2.1.2 Zeitdiskrete Modelle

Zur Modellierung nichtlinearer dynamischer Systeme kommen üblicherweise zeitdiskrete Modelle zum Einsatz. Dies hängt einerseits mit den in zeitdiskreter Form vorliegenden Messwerten und andererseits mit den verfügbaren Modellbildungsverfahren zusammen, welche sich zum allergrößten Teil nur zur Identifikation zeitdiskreter Systeme eignen. Um den dynamischen Zusammenhang zwischen zeitdiskreten Ein- und Ausgängen beschreiben zu können, wird eine Darstellungsform für die Modelldynamik benötigt.

Zur Darstellung der Dynamik sind zwei grundsätzlich unterschiedliche Ansätze gebräuchlich: zum einen Funktionsapproximatoren mit interner Dynamik und zum anderen Funktionsapproximatoren mit externer Dynamik [60]. Funktionsapproximatoren mit interner Dynamik werden hauptsächlich zur Identifikation von Zeitreihen verwendet, während Funktionsapproximatoren mit externer Dynamik vor allem im Bereich der Systemidentifikation zum Einsatz kommen.

Externe Dynamik: Ein-/Ausgangmodelle

Der Ansatz der externen Dynamik verwendet, wie in Abbildung 2.2 dargestellt, eine externe dynamische Filterbank und einen statischen Funktionsapproximator [60]. Diese Darstellung realisiert eine Differenzengleichung, wobei die Ordnung derselben durch die Anzahl der Verzögerungsglieder festgelegt wird. Wie bereits in der Einleitung beschrieben, können zwei unterschiedliche Konfigurationen zur Darstellung der Dynamik verwendet werden: zum einen die Prädiktor-Konfiguration und zum anderen die Simulator-Konfiguration. Beide Konfigurationen werden durch die Verwendung einer externen dynamischen Filterbank realisiert.

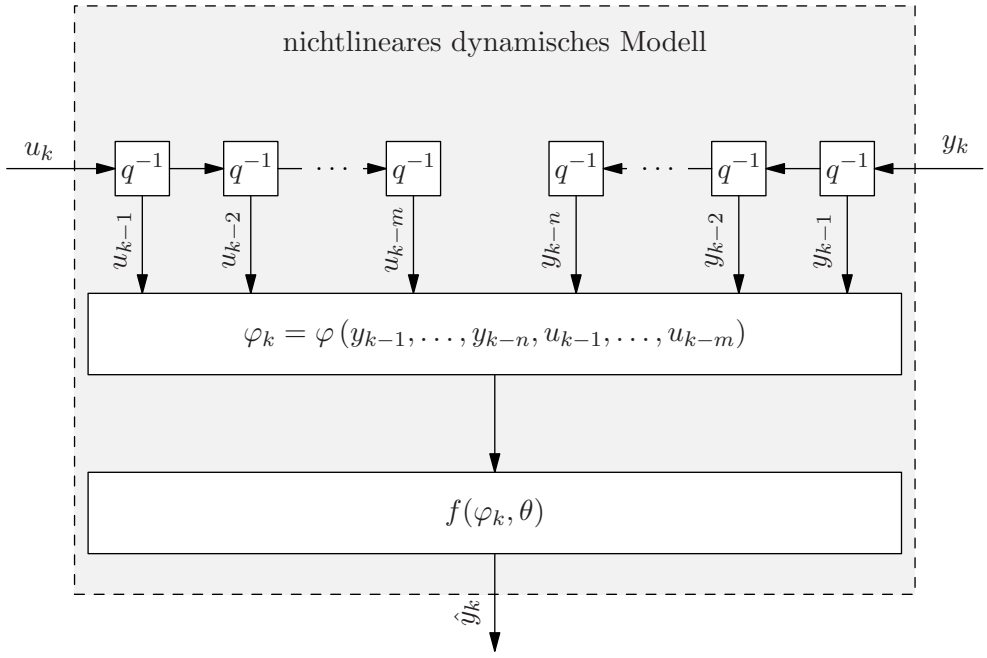


Abbildung 2.2: NARX Modell. Zeitverzögerungsoperator q^{-1}: $q^{-1}u_k = u_{k-1}$.

Wesentliche Vorteile der externen Dynamik sind dabei die üblicherweise geringere Anzahl der Modellparameter, da innerhalb des Modells keine zu modellierenden rekurrenten Verbindungen vorhanden sind, und die Möglichkeit die Komplexität der Dynamik über die Filterbank festzulegen. Als Eingänge für die Filterbank werden die verzögerten Systemeingänge und je nach Modellierungsform verzögerte Messgrößen (Ein-Schritt Prädiktor: *Nonlinear Auto Regressive with Exogenous Input*, NARX-Modell)

$$\hat{y}_k = f(\varphi_k, \theta) \qquad \varphi_k = \varphi\left(y^k, u^k\right) \tag{2.5}$$

oder verzögerte Modellausgänge (Simulator: *Nonlinear Output Error*, NOE-Modell)

$$\hat{y}_k = f(\varphi_k, \theta) \qquad \varphi_k = \varphi\left(\hat{y}^k, u^k\right) \tag{2.6}$$

verwendet [78]. Komplexere Modelle, die beispielsweise den Prädiktionsfehler als zusätzlichen Eingang berücksichtigen, kommen kaum zum Einsatz, da die zusätzliche Flexibilität in der Regel nicht den Nachteil der erweiterten Eingangsdimensionalität kompensieren kann [60]. Das Training eines NARX-Modells (Prädiktor) ist erheblich einfacher als das Training eines NOE-Modells (Simulator), da ersteres keine rekurrenten Komponenten aufweist und somit mit Standard-Trainingsalgorithmen für statische Modelle trainiert werden kann.

Dient das identifizierte Modell als Simulator, stellt die Fehlerakkumulation bei NARX-Modellen ein großes Problem dar, wodurch ein guter Prädiktor ein sehr schlechter Simulator sein kann. Darüber hinaus sind NARX-Modelle gegenüber NOE-Modellen sensibler bezüglich der Wahl der Abtastzeit und hohe Frequenzen werden stärker betont [60]. In der Praxis werden daher meist verschiedene NARX-Modelle trainiert und die Parameter des besten Modells als Initialwerte für das Training des NOE-Modells verwendet, wodurch das Lernverfahren schneller konvergiert [60].

Neben den genannten Vorteilen ist ein wesentlicher Nachteil der externen Dynamik die mit der Ordnung der Dynamik ansteigende Eingangsdimensionalität. Zudem kann mit einem Modell der Struktur (2.5) beziehungsweise (2.6) nur eine Unterklasse der von (2.2) beschriebenen Systemklasse identifizieren werden. Dies wird leicht ersichtlich, wenn die Voraussetzungen für die Rekonstruierbarkeit der Zustände des Systems (2.2) betrachtet werden. Sollen aus den gegebenen Messungen:

$$\begin{aligned}\varphi_{y,k} &= [y_{k-1}^\top, \ldots, y_{k-N_x}^\top]^\top \\ \varphi_{u,k} &= [u_{k-1}^\top, \ldots, u_{k-N_x}^\top]^\top\end{aligned}$$

die Zustände $x \in \mathbb{R}^{N_x}$ rekonstruieren werden, muss folgendes Gleichungssystem [90]

$$\begin{aligned}\varphi_{y,k} &= \left[h_s\left(x_{k-1}, u_{k-1}\right)^\top, \ldots, h_s\left(x_{k-N_x}, u_{k-N_x}\right)^\top\right]^\top \\ &= \begin{bmatrix} h_s\left(f_s\left(\ldots f_s\left(x_{k-N_x}, u_{k-N_x}\right), \ldots, u_{k-2}\right), u_{k-1}\right) \\ \vdots \\ h_s\left(f_s\left(x_{k-N_x}, u_{k-N_x}\right), u_{k-N_x+1}\right) \\ h_s\left(x_{k-N_x}, u_{k-N_x}\right) \end{bmatrix} \\ &=: \tilde{h}\left(x_{k-N_x}, \varphi_{u,k}\right)\end{aligned}$$

nach x_{k-N_x} aufgelöst werden. Dies ist nur möglich, wenn die Funktion $\tilde{h}(\cdot)$ bijektiv (eindeutig umkehrbar) ist. Dadurch ist es beispielsweise mit einem Modell der Struktur (2.5) beziehungsweise (2.6) nicht möglich Hystereseeffekte und Lose (mechanisches Spiel) darzustellen [60]. Diese Problematik wird anhand einer nichtinvertierbaren Nichtlinearität in Beispiel 1 veranschaulicht. Dessen ungeachtet erlauben Ein-/Ausgangsmodelle eine große Klasse an Systemen zu approximieren [49].

Beispiel 1: Dieses Beispiel ist aus [60] entnommen und zeigt die Restriktionen der Klasse der Ein-/Ausgangsmodelle auf. Als Beispiel dient ein Wiener Modell, welches aus einem linearen dynamischen System gefolgt von einem nichtlinearen statischen System besteht

$$\begin{aligned}x_k &= b_1 u_{k-1} - a_1 x_{k-1} \\ y_k &= g\left(x_k\right)\end{aligned}$$

Der Ausgang des Wienerprozesses [60] kann nun geschrieben werden als

$$y_k = g\left(b_1 u_{k-1} - a_1 x_{k-1}\right) = g\left(b_1 u_{k-1} - a_1 g^{-1}\left(y_{k-1}\right)\right)$$

wobei $g^{-1}(\cdot)$ die Inverse von $g(\cdot)$ bezeichnet. Ist $g(\cdot)$ nicht invertierbar, kann der Wienerprozess nicht durch ein dynamisches Ein-/Ausgangsmodell beschrieben werden. Allerdings kann ein Ein-/Ausgangsmodell jeden Prozess bis zu einem gewissen Grad approximieren, wobei dies für Wienerprozesse generell schwierig ist, da die Invertierbarkeit von $g(\cdot)$ Monotonie erfordert. Dies stellt eine Einschränkung dar, womit der Approximationsfehler von $g(\cdot)$ abhängt.

Die genannten Nachteile der erweiterten Eingangsdimensionalität und der beschränkten Systemklasse können durch Verwendung von Zustandsraummodellen vermieden werden.

Interne Dynamik: Zustandsraummodelle

Die Verwendung der internen Dynamik realisiert eine nichtlineare Zustandsdarstellung ohne Informationen über die wahren Systemzustände, das heißt die rekurrenten Strukturen sind innerhalb des Modells vorhanden. Modelle mit interner Dynamik können nach (2.2) mit zwei unabhängigen Parametervektoren θ_f beziehungsweise θ_h wie folgt beschrieben werden

$$\hat{x}_{k+1} = f(\hat{x}_k, u_k, \theta_f) \tag{2.7a}$$
$$\hat{y}_k = h(\hat{x}_k, u_k, \theta_h), \tag{2.7b}$$

wobei $\hat{x}$ die internen Modellzustände bezeichnet, welche häufig von der gewählten Modellstruktur abhängen und oftmals viel höher gewählt werden als die angenommene dynamische Ordnung des Systems. Daher ist die Wahl der Ordnung nicht so entscheidend wie beim Ansatz der externen Dynamik [60].

Im Gegensatz zum Ansatz der externen Dynamik ist es bei Zustandsraummodellen nicht notwendig, mehrmals verzögerte Ein- und Ausgänge als Modelleingänge zu verwenden, wodurch die Dimension des Eingangsraums unabhängig von der Modellordnung konstant bleibt. Diese im Gegensatz zur externen Dynamik reduzierte Eingangsraumdimensionalität stellt einen erheblichen Vorteil der internen Dynamik dar. Dem gegenüber steht die üblicherweise höhere Anzahl an zu bestimmender Modellparameter (durch die rekurrente Strukturen des Modells) und der Nachteil, dass ein Modell mit interner Dynamik bei unbekannten Systemzuständen nicht als Prädiktor verwendet werden kann. Dies erschwert auch das Training, da das Modell nur in der stets nichtlinearen Simulator-Konfiguration trainierbar ist. Aus den genannten Gründen werden in der Praxis hauptsächlich zeitdiskrete Modelle mit externer Dynamik verwendet.

2.1.3 Zeitkontinuierliche Modelle

Im Gegensatz zur zeitdiskreten Modellbildung findet sich wenig Literatur zur zeitkontinuierlichen Systemidentifikation. Der Großteil der bisherigen Forschung konzentriert sich hierbei auf lineare Modelle und die Ergebnisse zur Identifikation zeitkontinuierlicher nichtlinearer dynamischer Modelle sind überschaubar. Einen guten Überblick über verschiedene Verfahren zur Identifikation zeitkontinuierlicher Systeme findet man in [68]. Das Ein-/Ausgangmodell ist bei zeitkontinuierlichen Modellen in der Regel nicht anwendbar, da hierzu die n-te Ableitung der Ausgänge bekannt sein müsste, um eine Differentialgleichung n-ter Ordnung

$$y^{(n)} = f\left(u, y, y', \ldots, y^{(n-1)}\right)$$

darstellen zu können. Daher ist im Gegensatz zu den zeitdiskreten Modellen der Ansatz des Zustandsraumodells (2.1) gebräuchlicher. Um eine Prädiktor-Konfiguration realisieren zu können, müssen alle Systemzustände messbar sein. Da dies in den allermeisten Fällen nicht der Fall ist, kommt bei zeitkontinuierlichen Modellen in der Regel nur die Simulator-Konfiguration zum Einsatz. Der Einsatz zeitkontinuierlicher Modelle wird in der Praxis durch die erforderliche Abtastrate erschwert, um die Ableitungen der Systemzustände beziehungsweise Systemausgängen gut schätzen zu können. Diese muss sehr hoch gewählt werden, damit die Systemausgänge als quasikontinuierlich angesehen werden können, oder

spezielle Anregungssignale gewählt werden [68]. Dementsprechend groß ist die anfallende Datenmenge, welche das Training durch die mindestens linear mit den Trainingsdaten ansteigende Trainingszeit sehr aufwändig werden lässt, oder die Wahl des Anregungssignales eingeschränkt, wodurch nicht auf die spezifischen Anforderungen im Prüfstandsumfeld eingegangen werden kann.

2.1.4 Wahl der Abtastzeit

Die Wahl der Abtastzeit ist bei zeitdiskreten Modellen recht kritisch, wobei die Problematik bei Modellen in Prädiktor-Konfiguration verschärft wird [60]. Ist die Abtastfrequenz gegenüber der Systemdynamik sehr hoch gewählt, ändert sich der Ausgang zwei aufeinander folgender Abtastzeitpunkte kaum. Im extremsten Fall gilt $y(k+1) \approx y(k)$ und es kann kein Zusammenhang zwischen Ein- und Ausgang identifiziert werden. Wird die Prädiktor-Konfiguration verwendet, führt eine zu hohe Abtastrate dazu, dass die zu identifizierende Systemdynamik im Messrauschen untergeht. Ist die Abtastzeit hingegen zu niedrig, befindet sich das System schon im Stationärzustand und die Systemdynamik kann nicht identifiziert werden. Als Daumenregel gilt, dass die Abtastzeit 1/10 bis 1/20 der Systemeinschwingzeit betragen sollte [60]. Somit muss über Sprungantworten des Systems eine geeignete Abtastzeit ermittelt werden. Anhand des nachfolgenden Beispiels 2 wird die Problematik nochmals verdeutlicht.

Beispiel 2: Es wird folgendes lineares System 1.Ordnung (PT1-Glied) mit Zeitkonstante T und Verstärkungsfaktor K betrachtet

$$T\dot{y} = y + Ku.$$

Durch Separation der Variablen ergibt sich als Lösung für die Differentialgleichung

$$y_{k+1} = y_k e^{-t_s/T} + (1 - e^{-t_s/T})Ku_k,$$

wobei t_s die Abtastzeit bezeichnet und angenommen wird, dass u für $t \in [t_k\ t_{k+1}[$ konstant ist. Die Lösung der Differentialgleichung für unterschiedliche Abtastzeiten und die Zusammenhänge zwischen Eingangsraum und Zielfunktion sind für $T = 1$, $K = 1$ in Abbildung 2.3 dargestellt.

Wie ersichtlich, ist für niedrige Abtastraten die Lösung der Differentialgleichung hauptsächlich vom Eingang abhängig, während für hohe Abtastraten der Eingang kaum eine Rolle spielt, da $y_{k+1} \to y_k$, $t_s \to 0$ gilt. Um die Parameter gut identifizieren zu können, sollte die Abhängigkeit vom vorhergegangenen Systemzustand y_k und dem Eingang u_k in derselben Größenordnung liegen

$$e^{\frac{-t_s}{T}} = K\left(1 - e^{\frac{-t_s}{T}}\right) \Rightarrow t_s = -T\log\left(\frac{K}{1+K}\right)$$

Wie in Abbildung 2.3 graphisch verdeutlicht, sollte die Abtastzeit groß gewählt werden, wenn die Zeitkonstante T groß beziehungsweise der Verstärkungsfaktor K klein ist. Allerdings nimmt mit steigender Abtastzeit die Anzahl der Messungen ab, wodurch die Varianz in der Parameterschätzung zunimmt (siehe Diskussion im nachfolgenden Abschnitt 2.2). Daher wird in [60] als Faustregel für die Wahl der Abtastrate circa 1/10 bis 1/20 der größten interessierenden Zeitkonstante angegeben. Diese Wahl wird umso problematischer, desto größer der Verstärkungsfaktor des Systems ist.

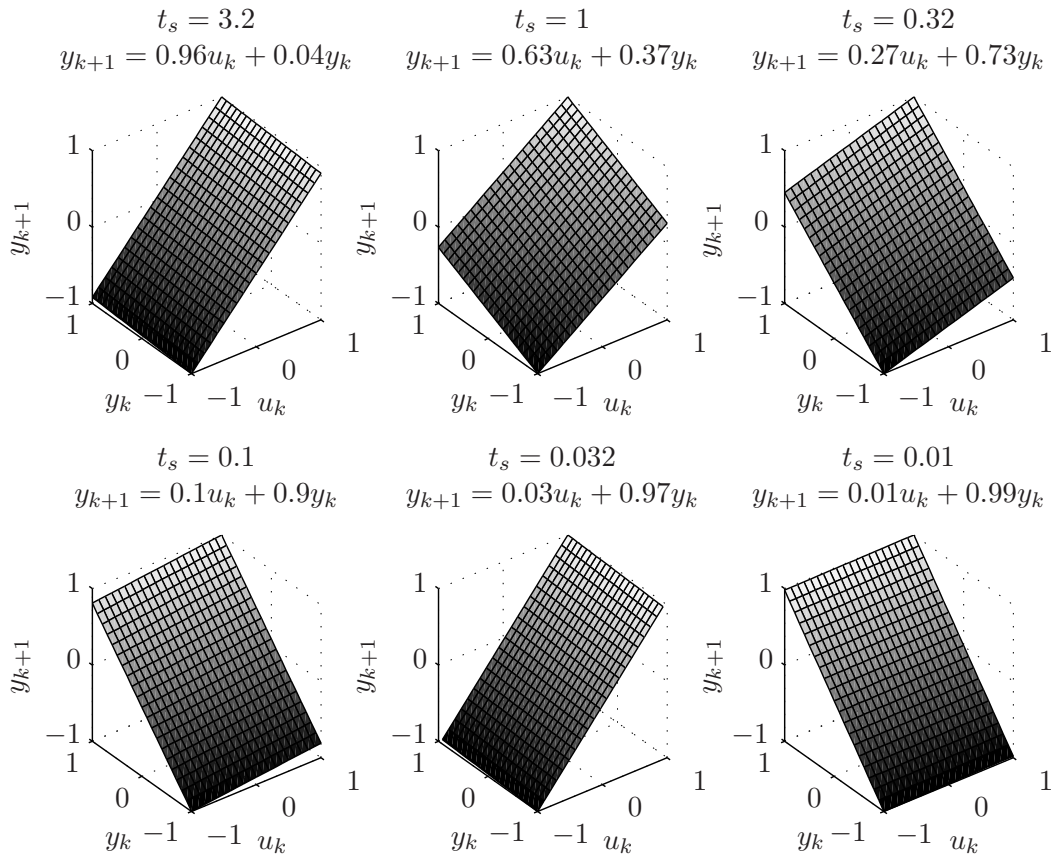


Abbildung 2.3: Abhängigkeit der Ein-/Ausgangsabbildung von der Abtastzeit.

2.1.5 Diskussion im Sinne der Aufgabenstellung

Die Aufgabenstellung erfordert Modelle, welche für unterschiedlichste Identifikationsaufgaben verwendet werden können. Es sollen sowohl Zielgrößen mit unterschiedlicher Dynamik modelliert, als auch die Stationärwerte von Zielgrößen vorausgesagt werden. Das Modell muss somit in Simulator-Konfiguration trainiert werden, um es frei laufend, zum Beispiel zur Stationärwertprädiktion, verwenden zu können. Neben der Modellgüte ist auch der zeitliche Aufwand des Modelltrainings und der Modellauswertung ein entscheidender Faktor, da die Online-Versuchsplanung ein ständiges Auswerten und Anpassen der Modelle erfordert. Die angesprochene Problematik der Wahl der Abtastzeit lässt sich durch die Verwendung von zeitkontinuierlichen Modellen vermeiden, was allerdings der Forderung nach einem schnellen Modelltraining und somit geringem Trainingsdatenumfang widerspricht.

2.2 Bias-Varianz Dilemma

Unabhängig von der Wahl des Modellbildungsverfahrens, der Modellstruktur sowie Modellkonfiguration tritt immer das grundlegende Problem der Systemidentifikation auf; das sogenannte Bias-Varianz Dilemma.

Das Bias-Varianz Dilemma resultiert aus der notwendigen Wahl der Modellkomplexität beziehungsweise -flexibilität. Ein Modell, dessen Komplexität zu gering ist, um ein Systemverhalten zu modellieren, wird einen systematischen Fehler aufweisen, auch als Bias bezeichnet. Im Gegensatz dazu führt ein Modell mit zu hoher Komplexität zu einer Schwankung beziehungsweise Streuung um den zu modellierenden Wert, das heißt zu einer Varianz. Ersteres Phänomen wird auch als *Underfitting* und letzteres als *Overfitting* bezeichnet.

Ein Modell wird um so komplexer, über desto mehr zu identifizierende Parameter es verfügt. Um den Einfluss der Anzahl der Parameter zu untersuchen, wird der *Modellfehler* in den *Biasfehler* und den *Varianzfehler* aufgeteilt. Wie in Abbildung 2.4 dargestellt, wird die Differenz zwischen dem Modellausgang $\hat{y}$ und dem Messwert y betrachtet. Dabei ist y der mit Rauschen η behaftete Systemausgang y_s. Dabei wird angenommen, dass das Rauschen $\eta \sim \mathcal{N}(0, \sigma^2)$ normalverteilt mit Varianz σ^2 ist (siehe Anhang A). Als Kostenfunktion, welche beim Modelltraining minimiert werden soll, wird bei der Systemidentifikation üblicherweise die Summe der quadratischen Fehler verwendet.

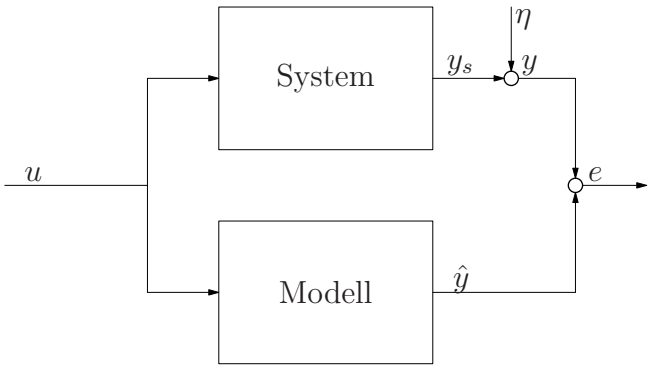


Abbildung 2.4: Zerlegung des Fehlers e in Rauschen η und den Bias- bzw. Varianzfehler

Der Erwartungswert des quadratischen Fehlers kann in zwei Teile aufgespalten werden

$$E[e^2] = E[(y - \hat{y})^2] = E[(y_s + \eta - \hat{y})^2] = E[(y_s - \hat{y})^2] + E[\eta^2] + 2E[(y_s - \hat{y})\eta],$$

wobei der Term $E[(y_s - \hat{y})\eta]$ vernachlässigbar ist, da das Messrauschen unkorreliert ist mit dem System- und Modellausgang. Der Modellfehler kann weiter aufgespalten werden [60]

$$\underbrace{E\left[(y_s - \hat{y})^2\right]}_{\text{Modellfehler}^2} = \underbrace{\left(y_s - E\left[\hat{y}\right]\right)^2}_{\text{Biasfehler}^2} + \underbrace{E\left[(\hat{y} - E\left[\hat{y}\right])^2\right]}_{\text{Varianzfehler}}. \tag{2.8}$$

Wie aus der Gleichung (2.8) ersichtlich, stellt der Biasfehler die Abweichung des erwarteten Modellausgang vom Systemausgang dar. Dieser systematische Fehler wird durch die mangelnde Flexibilität des Modells verursacht. In der Praxis kann immer ein durch den

Biasfehler verursachter Approximationsfehler erwartet werden. Typischerweise kann nur erreicht werden, dass bei steigender Modellkomplexität der Biasfehler *gegen* Null geht. Ein Approximator der diese Eigenschaft erfüllt, wird als *universeller Approximator* bezeichnet [60]. Daher besteht ein Modellierungsziel darin, dem Modell möglichst viel Flexibilität und somit Parameter zu geben. Allerdings nimmt der Einfluss zusätzlicher Parameter auf den Biasfehler bei steigender Parameterzahl ab. Eine schematische Darstellung, welche diesen Zusammenhang verdeutlicht, findet sich in Abbildung 2.5(a).

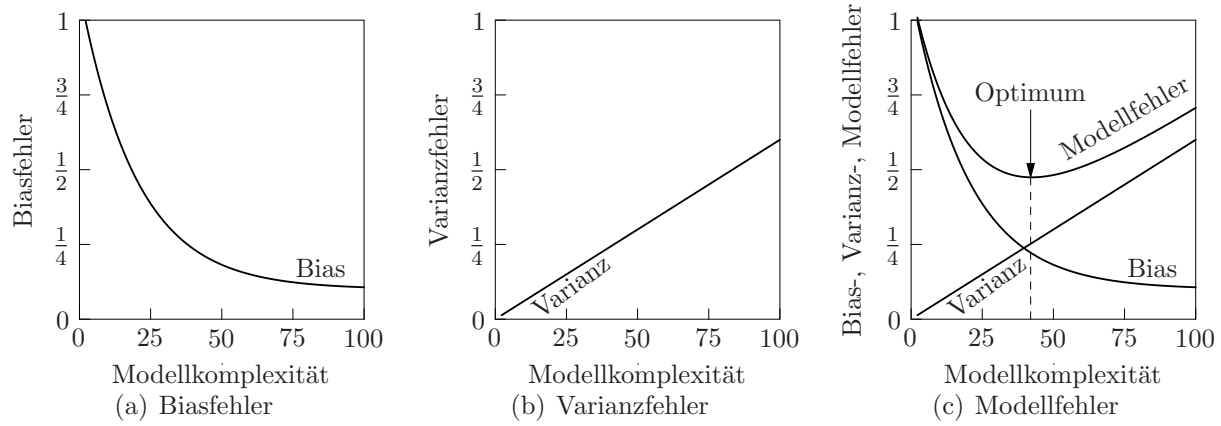
Der Varianzfehler kann hingegen wie folgt interpretiert werden. Wenn eine identische Eingangssequenz mehrmals zur Systemanregung verwendet wird, erhält man aufgrund des Messrauschens unterschiedliche Datensätze. Werden basierend auf diesen verschiedene Modelle erzeugt, dann gibt der Varianzfehler die mittlere quadratische Abweichung der unterschiedlichen Modellausgänge vom mittleren Modellausgang an. Ein Modell mit zu hoher Flexibilität modelliert das Rauschen im Messsignal mit und somit führt ein Modell mit zu hoher Flexibilität zu einer großen Varianz. Es kann nun gezeigt werden, dass der Varianzfehler für große Trainingsdatensätze mit N Trainingspaaren annähernd linear mit der Anzahl der Parameter N_p ansteigt [50]

$$E\left[(\hat{y} - E[\hat{y}])^2\right] \approx \sigma^2 \frac{N_p}{N}.$$

Dieser Zusammenhang gilt bei ausreichend großem Trainingsdatensatz unabhängig von der Modellierung. Der Zusammenhang ist in Abbildung 2.5(b) dargestellt. Die Steigung der Geraden hängt dabei vom Messrauschen mit Varianz σ^2 und der Anzahl der Trainingsdaten ab. Somit wird der Modellfehler signifikant beeinflusst von der Anzahl der verfügbaren Trainingsdaten und deren Qualität (Varianz der Messdaten). Der Varianzfehler kann reduziert werden, indem die Flexibilität des Modells eingeschränkt oder die Anzahl der Trainingsdaten erhöht wird. Dies bedeutet, dass für Modelle, die ausreichend flexibel sind um das zu approximierende Systemverhalten zu beschreiben, der Modellfehler mit $\sqrt{N}$ abnimmt. Aus dieser Diskussion ist die Bedeutung des Umfangs des Trainingsdatensatzes deutlich ersichtlich.

Die Abbildung 2.5(c) fasst die Effekte des Bias und Varianzfehlers zusammen. Ein sehr einfaches Modell hat einen geringen Varianzfehler und einen hohen Biasfehler; ein sehr flexibles Modell hingegen einen geringen Bias- und hohen Varianzfehler. Da sich der Bias- und Varianzfehler nicht gleichzeitig minimieren lassen und somit in Konflikt stehen, wird dieser Zielkonflikt als Bias-Varianz Dilemma bezeichnet. Dabei entscheidet das Messrauschen und die Anzahl der Trainingsdaten wie ausgeprägt der Konflikt ist.

Der Zielkonflikt in der Modellbildung liegt somit darin, dem Modell ausreichend Flexibilität zu geben, damit es keinen systematischen Fehler aufweist (Biasfehler) und andererseits die Flexibilität soweit zu begrenzen, dass keine Überanpassung erfolgt (Varianzfehler). Um den Zielkonflikt aufzulösen, gibt es im Wesentlichen drei Möglichkeiten. Entweder wird das Training abgebrochen bevor es zu einer Überanpassung kommen kann (*engl. early stopping*) oder die Flexibilität der Modelle durch sogenannte Regularisierungsverfahren beschränkt, so dass eine Überanpassung vermieden wird. Eine dritte Möglichkeit ist die Verwendung von Modellkomitees, welche durch Kombination mehrerer Einzelmodelle, die Varianz der Modellausgänge reduzieren. Auf die unterschiedlichen Verfahren wird in den nächsten Abschnitten eingegangen.



(a) Biasfehler (b) Varianzfehler (c) Modellfehler

Abbildung 2.5: Bias-, Varianz- und Modellfehler

2.2.1 Early Stopping

Das Ziel dieses Verfahrens ist es, das Training beziehungsweise die Optimierung abzubrechen, sobald der Varianzfehler zunimmt. Um einen geeigneten Zeitpunkt für den Trainingsabbruch zu finden, werden meist Validierungsdatensätze verwendet, welche als Maß für den Varianzfehler dienen. Kommt es während des Trainings ab einem gewissen Punkt nur noch zu einer Reduktion des Trainingsfehlers, ohne dass der Validierungsfehler reduziert wird, kann das Training abgebrochen und die Parameter verwendet werden, bei denen der Validierungsfehler am geringsten war. Durch den frühzeitigen Abbruch des Trainings wird eine Überanpassung vermieden und die beste Generalisierungsfähigkeit bezogen auf den Validierungsdatensatz sichergestellt. Der Nachteil dieses Verfahren ist, dass neben dem Trainings- auch ein Validierungsdatensatz erstellt werden muss. Dies kann bei aufwändigen Messungen unerwünscht sein und kann durch den Einsatz der folgenden Verfahren vermieden werden.

2.2.2 Regularisierungsverfahren

Regularisierungsverfahren verfolgen den Ansatz, ein flexibles Modell während des Trainings in seiner Flexibilität zu beschränken, um eine Überanpassung zu verhindern. Dies wird in der Regel dadurch erreicht, dass eine neue Kostenfunktion eingeführt wird, die neben einer Norm an den von den Modellparametern θ abhängigen Trainingsfehler $e(\theta)$ auch eine Norm an die Modellparameter respektive -gewichte θ minimiert

$$S = \beta \left\| e(\theta) \right\| + \alpha \left\| \theta \right\| \tag{2.9}$$

Dabei hängt die Wahl der Norm vom Regularisierungsverfahren ab. Die Gewichtung des Trainingsfehlers beziehungsweise die Bestrafung der Gewichte wird über die sogenannten Hyperparameter α und β festgelegt. Durch die Beschränkung der Gewichte mit dem Term $\alpha \left\| \theta \right\|$ (auch *weight decay* Term genannt) wird die gewünschte Reduktion der Modellflexibilität [6] erreicht. Die angestrebte Beschränkung der Gewichte beruht auf der Beobachtung, dass sich große Gewichte und ein glatter Modellausgang gegenseitig ausschließen, da große

Gewichte meist große Gradienten bedingen. Die Wahl der Parameter kann durch die Verwendung eines Validierungsdatensatzes ermittelt werden oder durch Bayes'sche Verfahren, welche in Abschnitt 4.1.3 eingeführt werden.

Growing-Verfahren

Bei den sogenannten Growing Verfahren (auch Vorwärtsselektion) wird die Komplexität des Funktionsapproximators durch Hinzunahme von Gewichten schrittweise erhöht, bis der Trainingsfehler klein genug oder ein Abbruchkriterium erreicht wird. Der Vorteil der Growing-Verfahren liegt darin, dass mit einem einfachen, schnell zu trainierendem Modell begonnen werden kann und dieses während des Trainings an die Komplexität des Problems angepasst wird. In der linearen Regression fallen hierunter die orthogonalen least squares Verfahren [11] und bei nichtlinearen Modellen können wiederum Verfahren eingesetzt werden, welche auf der Bayes'schen Methode aufbauen (siehe Abschnitt 4.1.3).

Pruning-Verfahren

Die Pruning Verfahren (auch Rückwärtsselektion) verfolgen den umgekehrten Ansatz und starten mit einem sehr flexiblen Modell und reduzieren nach und nach die Modellfexibilität durch Reduktion der Anzahl der Modellgewichte. Das Ziel ist somit eine Minimierung der Netzwerkarchitektur, wobei dies während oder nach dem Training erfolgen kann. Im Gegensatz zu den Growing-Verfahren weisen die Pruning-Verfahren einen erhöhten Trainingsaufwand auf, da das Training mit einer Vielzahl an Gewichten gestartet wird und (in der Startphase) entsprechend langsam ist. Dem Nachteil des erhöhten Trainingsaufwands steht der Vorteil gegenüber mit meist weniger Anpassungen der Architektur ein gutes Modell zu finden [39].

2.2.3 Modellkomitees

Während Regularisierungsverfahren durch Beschränkung der Modellflexibilität die Überanpassung zu verhindern suchen, reduzieren Modellkomitees durch geschickte Kombination mehrerer Modelle die Varianz des Komiteeausgangs. Die Varianz der Einzelmodelle um den „wahren" Wert ist von den Modellgewichten abhängig. Bei nichtlinearen Modellen stellt die Optimierung der Gewichte ein nicht-konvexes Optimierungsproblem dar, dessen Lösung von der zufälligen Initialisierung der Startgewichte abhängt. Die von diesen zufälligen Startwerten abhängige Varianz der Modellausgänge lässt sich daher durch geschickte Kombination verschiedener Einzelmodelle weitgehend „herausmitteln" [6]. Auf die Erstellung von Modellkomitees wird in Abschnitt 5.3 noch näher eingegangen.

2.3 Modellbildungsverfahren

Bei den Modellbildungsverfahren kann im Wesentlichen zwischen zwei Modelltypen unterschieden werden. Modelle, welche eine feste Struktur mit einer definierten Anzahl von freien Parametern aufweisen, werden als parametrische Modelle bezeichnet. Ist die Modellstruktur hingegen vom Trainingsdatensatz abhängig, wird von nicht-parametrischen Modellen gesprochen. Entscheidend ist neben der Modellstruktur auch die im Training verwendete Fehlerfunktion für die erzielte Modellgüte.

2.3.1 Parametrische Modelle

Die Ein-/Ausgangsabbildung parametrischer Modelle wird über einen adaptierbaren Parametervektor kontrolliert. Die Anpassung des Parametervektors anhand der Trainingsdaten erfolgt dabei abhängig von der gewählten Modellstruktur mittels linearer oder nichtlinearer Optimierungsalgorithmen. Das Ziel des Modelltrainings ist das Modell dahingehend zu optimieren, dass der Modellausgang die vorliegenden Trainingsdaten möglichst exakt beschreibt und die Generalisierungsfähigkeit der Modelle erhalten bleibt. Ist das Training abgeschlossen, werden die Trainingsdaten nicht mehr benötigt, da die gesamte Modellinformation im Parametervektor liegt. Zu diesen Modellen zählen beispielsweise Neuronale Netze und (lokal) lineare (Polynomial-)Modelle. Wird die Anzahl der Parameter der Modelle während des Trainings mittels Growing- und/oder Pruning-Verfahren an die Komplexität der Problemstellung angepasst, ist der Übergang zu den nicht-parameterischen Modellen fließend, da die finale Modellstruktur abhängig von den Trainingsdaten ist.

2.3.2 Nicht-Parametrische Modelle

Zu den nicht-parametrischen Modellbildungsverfahren zählen im Wesentlichen Kernel-Methoden [6] wie Support Vector Machines (SVM) [73] und Gauß-Prozesse [69]. Diese Modelle weisen keine feste Modellstruktur auf und basieren auf den Trainingsdaten selbst. Dies bedeutet, dass nicht-parametrische Modelle die gesamten Trainingsdaten abspeichern und basierend auf diesen neue Datenpunkte prädizieren. Die Vorhersage basierend auf den Trainingsdaten ermöglicht, wie beispielsweise in [6] und [69] beschrieben, eine einfachere Beherrschung des Bias-Varianz-Dilemmas und erlaubt Aussagen über die Modellunsicherheit zu treffen. Aufgrund dieser Möglichkeit stellen vor allem Gauß-Prozesse einen vielversprechenden Ansatz bei der Black-Box Modellierung von statischen Zusammenhängen des Verbrennungsmotors dar [4]. Allerdings hängt bei nicht-parametrischen Modellen die Modellkomplexität von der Anzahl der Trainingsdaten beziehungsweise Messungen ab. Bei einem geringen Trainingsdatenumfang ist dies unproblematisch. Nimmt die Anzahl der Messungen jedoch stark zu (mehrere tausend Trainingspunkte), wird das Modelltraining wie auch die Modellauswertung extrem zeitaufwändig [69]. Dies macht den Einsatz nicht-parametrischer Modelle bei der Online-Identifikation dynamischer Systeme weitgehend inpraktikabel, da hier mit sehr großen Datenmengen umgegangen werden muss. Aus diesem Grund werden die nicht-parametrischen Modelle im Weiteren nicht mehr betrachtet.

2.3.3 Kostenfunktionen

Die Kostenfunktion, welche durch die Optimierung der Modellgewichte minimiert wird, beeinflusst entscheidend die erzielte Modellgüte. Deren Wahl hängt meist von der Art des Messrauschens ab.

Das populärste Fehlermaß ist die Summe der Fehlerquadrate, welches ein Sonderfall des Minkowski Fehlers darstellt. Dieser ist definiert als [5]

$$E = \sum \sum \|\hat{y}(k)_i - y(k)_i\|^R \tag{2.10}$$

und entspricht für $R = 2$ der Summe der Fehlerquadrate. Die Norm $\|\hat{y}_k - y_k\|^R$ wird als L_R Norm bezeichnet. Die partielle Ableitung nach den Gewichten ist

$$\frac{\partial E}{\partial w_{ij}} = \sum \sum \|\hat{y}(k)_i - y(k)_i\|^{R-1} sign(\hat{y}(k)_i - y(k)_i) \frac{\partial y(k)_i}{\partial w_{ij}}. \tag{2.11}$$

Für $R = 1$ wird der Fehlerbetrag und somit der Median minimiert

$$E = \|\hat{y}_k - y_k\|.$$

Die Minimierung von E bezüglich y gibt

$$\sum sign(\hat{y}_k - y_k) = 0,$$

was erfüllt ist, wenn $\hat{y}$ der Median der Messwerte y ist, das heißt, wenn die Anzahl der Werte für die y größer als $\hat{y}$ sind, denselben Wert hat wie die Anzahl der Werte, die kleiner als $\hat{y}$ sind. Wenn nun keine beziehungsweise wenige Ausreißer vorhanden sind, hat dies keinen Einfluss auf die Lösung von y.

Für $R = 2$ wird der mittlere quadratische Fehler minimiert. Dieses Fehlermaß ist die beste Wahl, wenn keine Ausreißer vorhanden sind. Kleine Abweichungen werden kaum bestraft (z.B. Messrauschen, kleine Varianz) während größere Abweichungen (Biasfehler) stark gewichtet werden.

2.3.4 Diskussion im Sinne der Aufgabenstellung

Die Aufgabenstellung erfordert eine Online-Modellbildung sowie die Berücksichtigung der Modellgüte in der Versuchsplanung. Dies erfordert ein häufiges Auswerten und Anpassen der Modelle im laufenden Betrieb. Somit sind parametrische Modelle den nichtparametrischen Modellen vorzuziehen, da parametrische Modelle bei großem Trainingsdatenumfang geringere Trainingszeit aufweisen und der Auswerteaufwand unabhängig vom Trainingsdatenumfang gleich bleibend und somit abschätzbar ist. Um das Bias-Varianz Dilemma zu beherrschen, werden in dieser Arbeit Modellkomitees, Regularisierungs- und Growing-Verfahren eingesetzt, die sich bei der Identifikation statischer Systeme bewährt haben [47].

3 Stand der Technik: Versuchsplanung

Die Versuchsplanung bestimmt wie das zu identifizierende System angeregt wird und somit welcher Trainingsdatensatz zur Modellbildung verfügbar ist. Dabei ist der Trainingsdatensatz von entscheidender Bedeutung für die Qualität des identifizierten Modells. Denn nur ein Trainingsdatensatz, der ausreichend Information über das System enthält, erlaubt es ein gutes Modell zu erstellen. Da das Anregungssignal die einzige Möglichkeit ist, den Informationsgehalt des Trainingsdatensatzes zu beeinflussen, stellt dieses unabhängig von der gewählten Modellarchitektur und -struktur eine untere Grenze an die Güte des identifizierten Modells dar. Das Ziel der Versuchsplanung besteht nun darin, mit jedem Messwert ein Maximum an Information über das zu untersuchende System zu erlangen, um mit einem möglichst geringen Messaufwand eine maximale Modellgüte zu erzielen. Bei der Versuchsplanung muss sowohl die Art des Anregungssignals als auch die Parametrierung desselben hinsichtlich der genannten Zielsetzung festgelegt werden.

3.1 Anregungssignale für dynamische Systeme

Wie bereits erwähnt, ist die Wahl des Anregungssignals von entscheidender Bedeutung für die erzielte Modellgüte. Das System soll mit diesem so angeregt werden, dass alle interessierenden Bereiche im Eingangsraum und alle relevanten Frequenzen abgedeckt werden.

Gebräuchliche Anregungssignale zur Identifikation nichtlinearer Systeme sind *multi-valued* PRBS (*engl. pseudo random binary signals*), amplitudenmodulierte PRBS (APRBS), Sinus-Signale und gefiltertes Gauß'sches Rauschen. In [50] und [60] findet sich eine Diskussion der genannten Anregungssignale. In industriellen Identifikationsaufgaben ist es oft von Bedeutung eine zu starke Aktuierung zu vermeiden, um den Prüfling und die Aktuatoren zu schonen [71]. Wenn die Aktuierung mit hohen Kosten verbunden ist, sind APRBS und multi-valued PRBS die am besten geeigneten Anregungssignale, da Sinus-Signale und gefiltertes Gauß'sches Rauschen eine Aktuierung zu jedem Abtastzeitpunkt erfordern. Da das multi-valued PRBS für die lineare Systemidentifikation entwickelt wurde und nur eine geringe Anzahl an Amplituden abdeckt, ist das APRBS das am weitesten verbreitetste Anregungssignal der Systemidentifikation an Motorenprüfständen [30], [64], [96].

Amplitudenmoduliertes Pseudobinäres Rauschsignal
Das APRBS ist ein amplitudenmoduliertes PRBS, welches ein periodisches deterministisches Signal mit Eigenschaften ähnlich dem weißen Rauschen ist [84]. Da die Amplituden frei zu wählende Werte im Eingangsraum darstellen, werden sie entsprechend der Terminologie der Versuchsplanung im Folgenden Designpunkte genannt. Dementsprechend kann das APRBS als eine Stufenfunktion (auch Heaviside-Funktion) bestehend aus N Intervallen

beziehungsweise Stufen aufgefasst werden

$$u(k) = \sum_{s=1}^{N} d_s \chi_{A_s}(k),$$

wobei A_s die Intervalle sind, die Designpunkte $d_s \in [u_{\min}, u_{\max}]$ beschränkt sind und χ_A die Indikatorfunktion von A ist:

$$\chi_{A_s}(k) = \begin{cases} 1 & \text{falls } k \in A_s, \\ 0 & \text{falls } k \notin A_s. \end{cases}$$

Durch die Haltezeit $T_{h,s}$, das heißt die Zeiten für welche das Signal u konstant gehalten wird, und durch die Abtastzeit t_s wird die Intervallänge

$$A_s = \left\{ \frac{1}{t_s} \sum_{i=1}^{s-1} T_{h,i} + 1, \ldots, \frac{1}{t_s} \sum_{i=1}^{s} T_{h,i} \right\}$$

festgelegt (siehe Abbildung 3.1). Dabei ist zu beachten, dass die Haltezeit $T_{h,s}$ ein ganzzahliges Vielfaches der Abtastzeit t_s sein muss. Über die Designpunkte und die Haltezeiten lässt sich das APRBS vollständig beschreiben und parametrieren. Neben dem zulässigen Bereich des Eingangsraums stellen die Signallänge und die minimale Haltezeit zu wählende Design-Parameter dar. Bei gegebener Signaldauer T beeinflusst die minimale Haltezeit $T_{h,\min} \leq T_{h,s}\ \forall s$ die Anzahl der Sprünge und somit die Frequenzcharakteristik des Anregungssignals. Dabei ist es wichtig, eine geeignete minimale Haltezeit zu wählen, damit das System genügend Zeit hat einzuschwingen. Ist dies nicht der Fall, werden nur Bereiche um $y \approx (u_{\max} - u_{\min})/2$ abgedeckt sein. Ein mittels solcher Daten identifiziertes Modell kann das Stationärverhalten des Systems nicht gut beschreiben. Die minimale Haltezeit eines APRBS sollte daher in der Größenordnung der dominanten (größten) Zeitkonstante des Prozesses liegen [60].

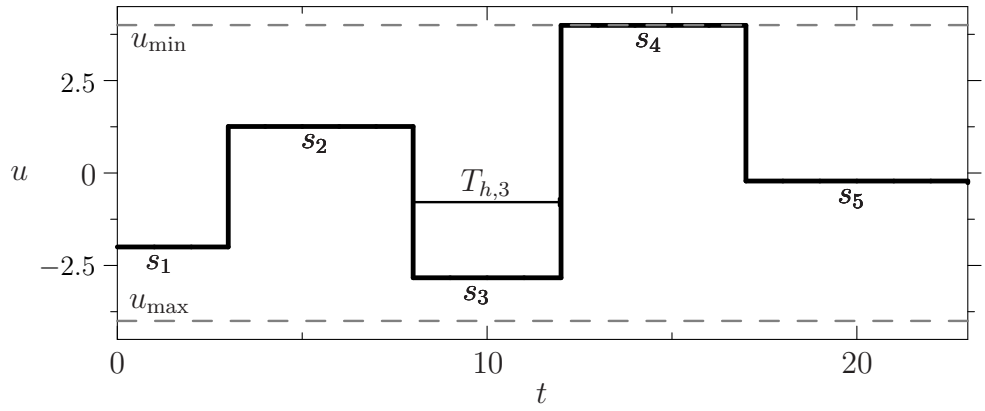


Abbildung 3.1: APRBS im Zeitbereich.

In [60] wird vorgeschlagen das APRBS wie folgt zu erstellen. Zunächst wird ein PRBS erzeugt. Dann wird die Anzahl der Sprünge gezählt und jede Dimension des Eingangsraums in die entsprechende Anzahl von Intervallen unterteilt. Schlussendlich wird jedem Sprung des PRBS für jede Dimension zufällig ein Wert aus diesen Intervallen zugeordnet. Wie in

[60] hervorgehoben wird, ist der Eingangsraum in der Regel gut abgedeckt, wobei aufgrund der zufälligen Wertezuweisung einige „Löcher“ existieren können. Eine gute Abdeckung des Eingangsraums ist daher nur bei einer Vielzahl von Sprüngen und somit entsprechender Anregungsdauer zu erwarten.

Um eine bessere Verteilung der Amplituden zu erreichen, können die im folgenden vorgestellten Methoden der Versuchsplanung verwendet werden.

3.2 Versuchsplanung: modellfreie und modellbasierte Verfahren

Die Methoden der Versuchsplanung lassen sich in modellfreie und modellbasierte Verfahren unterteilen. Erstere machen keine Annahmen über die verwendete Modellstruktur und versuchen den Eingangsraum möglichst gleichförmig mit Designpunkten abzudecken. Somit kann gewährleistet werden, dass über alle Bereiche des Eingangsraums Informationen vorliegen, welche durch das Modell abgebildet werden sollen. Modellbasierte Verfahren zielen hingegen darauf ab, die Designpunkte so im Eingangsraum zu verteilen, dass die Parameterschätzung möglichst unempfindlich gegenüber dem Rauschen wird. Dieses Vorgehen wird im folgenden Beispiel 3 anhand eines linearen Modells verdeutlicht.

Beispiel 3: Es wird ein lineares Modell der Form $y = \theta_0 + \theta_1 u$ mit dem Eingang u betrachtet (Abbildung 3.2). Zur Bestimmung der Modellparameter werden zumindest zwei Punkte benötigt, die frei im Bereich $u \in [-3, 3]$ zu wählen sind. Die Messungen unterliegen dabei einem maximalen Messrauschen im Bereich $[-0.4, 0.4]$, dargestellt durch die grauen Balken. Wird nun die Linienschar betrachtet, die sich abhängig von der Platzierung der Punkte ergibt, führen die Punkte am Rand des Definitionsbereichs (Abbildung 3.2(b)) zu einer wesentlich geringeren Varianz der Modelle und somit der Modellparameter als die Punkte näher am Ursprung (Abbildung 3.2(a)).

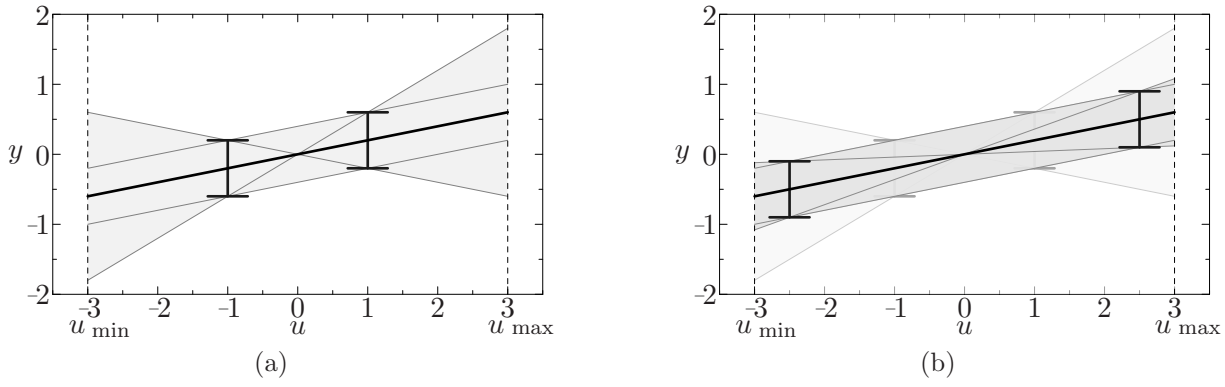


Abbildung 3.2: Veranschaulichung Versuchsplanung anhand eines linearen Modells.

Dieselben Überlegungen gelten für beliebige parametrische Modelle und lassen sich für die modellbasierte Versuchsplanung ausnützen.

Die Varianz der Parameterschätzung eines beliebigen parametrischen Modells kann mittels der Fisher-Information abgeschätzt werden. Diese basiert auf der Wahrscheinlichkeitstheorie und wird im folgenden Abschnitt vorgestellt. Die wichtigsten Grundlagen und Begriffe der Wahrscheinlichkeitstheorie, welche für das Verständnis der Fisher-Information relevant sind, finden sich im Anhang A.

Fisher-Information
Dieses Informationsmaß beschreibt den Informationsgehalt von Zufallsvariablen x über den Parametervektor θ, von dem die Likelihoodfunktion $L(\theta|x) = p(x|\theta)$, das heißt die Wahrscheinlichkeitsdichtefunktion betrachtet in Abhängigkeit des Parametervektors θ, abhängt (siehe auch Anhang A). Unter der Annahme, dass der Zufallsvektor x eine multivariate Normalverteilung aufweist und unter der Annahme von unkorreliertem Gauß'schen Rauschen $\Sigma(\theta) = \sigma^2 I$, ist die Fisher-Informationsmatrix (FIM) gegeben durch [44]

$$\mathcal{I}(\theta) = \frac{1}{\sigma^2} \frac{\partial \mu^T(\theta)}{\partial \theta} \frac{\partial \mu(\theta)}{\partial \theta}. \tag{3.1}$$

Dabei ist $\mu(\theta)$ der Mittelwert von $f(\theta)$ für Parameterschätzungen basierend auf denselben Trainingsdaten, welche mit unterschiedlichem Rauschen $\eta \sim \mathcal{N}(0, \sigma^2 I)$ behaftet sind. Im Anhang A.2 findet sich die Herleitung und weiterführende Erläuterungen zum besseren Verständnis. Nun besagt die Cramer-Rao Ungleichung [44]

$$\operatorname{cov}(\theta(x)) \geq \mathcal{I}(\theta, x)^{-1}, \tag{3.2}$$

dass die Varianz der Parameterschätzung nach unten hin durch die Inverse der Fisher-Informationsmatrix beschränkt ist. Dabei gilt ein Parameterschätzverfahren als effizient, wenn die untere Schranke der Cramer-Rao Ungleichung gilt. Für ein Schätzverfahren, welches für die Problemstellung geeignet ist, ist es sinnvoll anzunehmen, dass es asymptotisch effizient ist und daher die untere Schranke der Cramer-Rao Ungleichung erfüllt [29]. Um die Varianz der Parameterschätzung zu reduzieren, muss somit die Fisher-Informationsmatrix maximiert werden. Ist der Zusammenhang $\mu = f(\theta, u)$ bekannt oder beliebig genau approximierbar (das heißt die Struktur des Approximators ist geeignet um μ beliebig genau zu approximieren), lässt sich die FIM für die Zufallsvariablen $y_{k+1} = f(\theta, u_k) + \eta_k$ und $\eta \sim \mathcal{N}(0, \sigma I)$ berechnen

$$\mathcal{I}_k(\theta, u_k) = \frac{1}{\sigma^2} \frac{\partial f(\theta, u_k)^T}{\partial \theta} \frac{\partial f(\theta, u_k)}{\partial \theta}. \tag{3.3}$$

Die (mittlere) Fisher-Information für K Paare (y_k, u_k) wird definiert mit

$$\mathcal{I}\left(\theta, \mathcal{U}_1^K\right) = \frac{1}{K} \sum_{k=1}^{K} \mathcal{I}_k(\theta, u_k). \tag{3.4}$$

Dabei wird die Eingangssequenz $\mathcal{U}_1^K = \{u_1, \ldots, u_K\}$ als kompakte Teilmenge von $\mathbb{R}^{N_I}$ angenommen. Das Ziel, der auf der Fisher-Information basierenden Versuchspläne, besteht nun darin, eine Eingangssequenz zu bestimmen, welche den Informationsgehalt der Daten über die zu schätzenden Parameter maximiert.

Die Versuchsplanung lässt sich, wie erläutert, in modellfreie und modellbasierte Verfahren unterteilen. Hinsichtlich der Aufgabenstellung ist es zudem zweckmäßig, die Methoden der Versuchsplanung in Offline- und Online-Verfahren zu unterteilen.

3.3 Versuchsplanung: Offline-Verfahren

Bei der Offline-Versuchsplanung wird bereits vor Beginn der Messungen der Versuchsplan vollständig festgelegt. Dabei kann zwischen modellbasierter und modellfreier Versuchsplanung unterscheiden werden. Modellfreie Versuchspläne beziehen die gewählte Modellstruktur nicht mit ein, sondern versuchen den Eingangsraum möglichst gleichmäßig abzudecken. Modellbasierte Versuchspläne basieren hingegen auf Informationskriterien und maximieren den Informationsgehalt der Daten unter Berücksichtigung der gewählten Modellstruktur. Die beiden bekanntesten Methoden für beide Ansätze werden im Folgenden beschrieben.

3.3.1 Maxmin Latin Hypercube Verteilung

Bei der Latin Hypercube (LHC) Verteilung wird der Eingangsraum in K Intervalle unterteilt. Um die Notation zu erleichtern, wird angenommen, dass jeder Eingang auf das Einheitsintervall normiert ist und der zulässige Eingangsraum der N-dimensionale Einheitshypercubus ist. Jeder der d_i Designpunkte ist ein Element aus dem Set

$$V = \{0, 1/(K-1), 2/(K-2), \ldots, 1\},$$

so dass für jede Dimension j alle $d_{i,j}$ unterschiedlich sind. Damit wird erreicht, dass jeder Eingang mit einer Schrittweite von $1/K$ variiert wird und somit die Projektion des Eingangsraums auf jeden Eingang einer Rasterung von $1/K$ entspricht. Ein Latin Hypercube Design deckt den Raum gut ab, wenn der minimale Abstand der Designpunkte untereinander maximal ist. Der minimale Euklidische Abstand $\min_{i \neq j} d(d_i, d_j)$ der K Designpunkte kann beispielsweise über die in [58] vorgestellten Methoden maximiert werden. Die Verwendung von APRBS mit einer Maxmin LHC Verteilung der Designpunkte wird beispielsweise in [20] vorgeschlagen.

3.3.2 Fisher-Information basierte Verteilung

Für parametrische Modelle lässt sich basierend auf der Fisher-Informationsmatrix die optimale Verteilung der Designpunkte im Eingangsraum ermitteln. Um den Informationsgehalt für eine Eingangssequenz $\mathcal{U}_1^K$ zu maximieren und somit die Varianz des Schätzers zu minimieren, muss die Inverse der Fisher-Information $\mathcal{I}\left(\theta, \mathcal{U}_1^K\right)$ minimiert werden. Da es kein eindeutiges Kriterium zur Minimierung einer Matrix gibt, werden verschiedene Kriterien $J(\theta, \mathcal{U}_1^K) = g(\mathcal{I}(\theta, \mathcal{U}_1^K))$ verwendet [2],[24]:

- A-optimales Kriterium:

$$J\left(\theta, \mathcal{U}_1^K\right) = \operatorname{trace}\left(\mathcal{I}\left(\theta, \mathcal{U}_1^K\right)^{-1}\right)$$

- C-optimales Kriterium:

$$J\left(\theta,\mathcal{U}_1^K\right) = c^\top \left(\mathcal{I}\left(\theta,\mathcal{U}_1^K\right)^{-1}\right) c,$$

  bezüglich einem vorgegebenen Vektor c.

- D-optimales Kriterium:

$$J\left(\theta,\mathcal{U}_1^K\right) = \det\left(\mathcal{I}\left(\theta,\mathcal{U}_1^K\right)^{-1}\right)$$

- E-optimales Kriterium:

$$J\left(\theta,\mathcal{U}_1^K\right) = \lambda_{\max}\left(\mathcal{I}\left(\theta,\mathcal{U}_1^K\right)^{-1}\right),$$

  wobei $\lambda_{max}(\cdot)$ für den maximalen Eigenwert des Arguments steht.

Die optimale Eingangssequenz ermittelt sich durch Minimierung des Kriteriums

$$\mathcal{U}_1^{K*} = \arg\min_{\mathcal{U}_1^K} J\left(\theta,\mathcal{U}_1^K\right).$$

Für eine Diskussion der verschiedenen Kriterien sei hierbei auf [29] und [24] verwiesen. Das bekannteste Kriterium in der Versuchsplanung ist das D-optimale [86]. Durch die Relation $\det\left(A^{-1}\right) = \det\left(A\right)^{-1}$ für eine beliebige Matrix A, ist es möglich, die Minimierung des D-optimalen Kriteriums in ein Maximierungsproblem umzuformulieren

$$\mathcal{U}_1^{K*} = \arg\max_{\mathcal{U}_1^K} \det\left(\mathcal{I}(\theta,\mathcal{U}_1^K)\right) = \arg\min_{\mathcal{U}_1^K} \det\left(\mathcal{I}\left(\theta,\mathcal{U}_1^K\right)^{-1}\right),$$

wodurch die Invertierung der Fisher-Informationsmatrix vermieden werden kann. Dadurch ist das D-optimale Kriterium mit weniger Rechenaufwand auswertbar als die zuvor vorgestellten Kriterien.

Allerdings lässt sich die Fisher-Informationsmatrix nur bei Modellen, die linear in ihren Parametern sind, unabhängig von den zu schätzenden Parametern bestimmen. Kommen nun Modelle, die nichtlinear von ihren Parametern abhängen, zum Einsatz und ist kein Vorwissen über die Parameter vorhanden, kann die Fisher-Information nicht für eine Offline-Versuchsplanung verwendet werden, da hierzu die noch zu schätzenden Parameter bereits bekannt sein müssten. Daher wird in der Praxis die Komplexität des zu approximierenden Problems abgeschätzt und ein Versuchsplan berechnet, der auf einem entsprechend komplexen Polynommodell basiert. Für einen linearen Schätzer $\hat{y} = X\theta$ ergibt sich somit

$$\mathcal{I}\left(\theta\right) = \frac{1}{\sigma^2} X^\top X,$$

wobei die Matrix X die aus den Eingängen gebildeten Polynomterme enthält. Durch die Verwendung eines Polynommodells, das linear in seinen Parametern ist, ist es möglich den Versuchsplan bereits vor Beginn der Messungen vollständig festzulegen. Der Unsicherheit im Modellansatz kann dabei Rechnung getragen werden und über einen Bayes'schen Ansatz

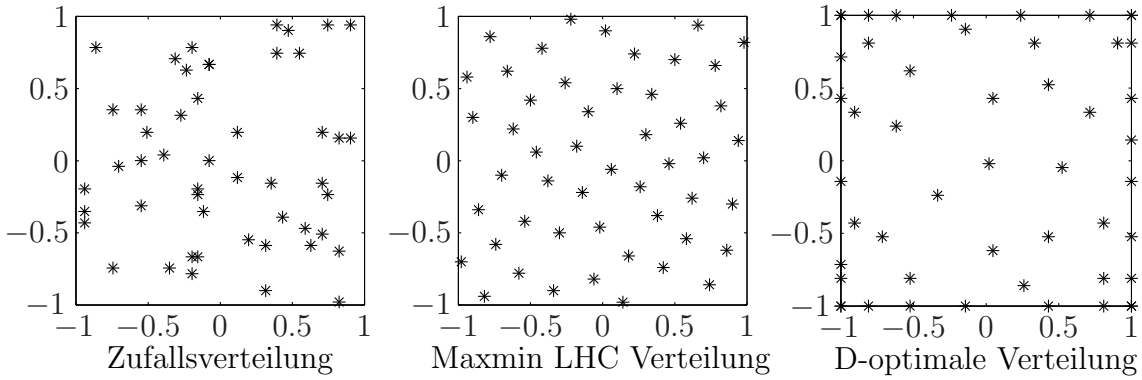


Abbildung 3.3: Verteilungen für verschiedene Offline-Verfahren.

diese Unsicherheit im D-optimalen Versuchsplan berücksichtigt werden. Dies führt zu einer geringfügigen Änderung im Optimalitätskriterium [93]. Das Kriterium der D-Optimalität lässt sich in der Regel nicht analytisch bestimmen. Ein bekanntes iteratives Verfahren zum Erzeugen von D-optimalen Versuchsplänen bei gegebener Anzahl von Designpunkten ist der *DETMAX-Algorithmus* [55], der auch in dieser Arbeit eingesetzt wird.

In Abbildung 3.3 ist eine typische D-optimale Verteilung mit 50 Designpunkten basierend auf einem Polynommodell 3. Ordnung für zwei Eingangsgrößen dargestellt. Im Gegensatz zur LHC-Verteilung wird vor allem der Randbereich des Eingangsraums mit Designpunkten gut abgedeckt. Dadurch kann sichergestellt werden, dass am Randbereich keine Extrapolation auftritt. Die Verwendung einer D-optimalen Verteilung der Designpunkte zur Erstellung eines APRBS findet sich beispielsweise in [74].

3.4 Versuchsplanung: Online-Verfahren

Im Gegensatz zu den Offline-Verfahren berücksichtigen Online-Verfahren den aus den Messungen resultierenden Informationsgewinn, um den weiteren Versuchsablauf anzupassen. Durch die Verwendung von adaptiven Versuchsplänen ist es möglich, Modelle an bestimmten Bereichen im Eingangsraum zu verfeinern, die sich im Laufe der Modellbildung für die Aufgabenstellung als relevant erweisen. Da bisher noch keine Online-Verfahren bei der Versuchsplanung für dynamische Modelle verwendet werden, werden im Folgenden zwei Verfahren aus der statischen Modellbildung beschrieben.

3.4.1 Query by Committee

Ein in der Praxis bewährtes Online-Verfahren ist die adaptive Versuchsplanung mit Modellkomitees, auch Query by Committee genannt. Die Idee besteht darin, Designpunkte im Eingangsraum zu platzieren, wo die Modelle des Komitees die größte Unsicherheit aufweisen. Die Varianz des Modellkomitees kann als Maß für die Modellunsicherheit verwenden werden. Formal wird nach einem Designpunkt gesucht, bei dem die Varianz eines aus m

Modellen $f_i(u, \theta_k^{(i)})$ bestehenden Modellkomitees maximal ist [65]

$$u_k^* = \arg\max_{u_k} \sum_{i=1}^{m} \left(f_i(u_k, \theta_k^{(i)}) - \frac{1}{m} \sum_{j=1}^{m} f_j(u_k, \theta_k^{(j)}) \right)^2 ,$$

wobei $\theta_k^{(i)}$ die Parameterschätzung des i-ten Modells zum Zeitpunkt k ist. Ein Designpunkt, der im Eingangsraum an einer Stelle maximaler Modellunsicherheit liegt, verspricht einen maximalen Informationsgewinn [65]. Ein weiterer Vorteil der Verwendung von Modellkomitees liegt in der reduzierten Varianz der kombinierten Modellausgänge (siehe Bias-Varianz-Dilemma). Dieser Ansatz wird in [47] erfolgreich zur Identifikation statischer Modelle eingesetzt.

3.4.2 Sequentielles Fisher-Information basiertes Design

Wie bereits erwähnt, hängt die Fisher-Informationsmatrix bei nichtlinearen Modellen von den Modellparametern ab. Bei einer Online-Versuchsplanung müssen daher mit jedem neuen Designpunkt die Modelle und mit ihnen das Fisher-Information basierte Design angepasst werden. Dazu wird, basierend auf der Parameterschätzung zum Zeitpunkt k, ein oder mehrere optimale Eingänge bestimmt:

$$\mathcal{U}^{*K}_k = \arg\min_{\mathcal{U}_k^K} J\left(\mathcal{U}_k^K, \theta_k\right) .$$

Dabei muss die Fisher-Information für die bereits angewandte Eingangssequenz $\mathcal{U}_1^{k-1}$ und die zu optimierende Eingangssequenz $\mathcal{U}_k^K$ berechnet werden. Aufgrund dieses iterativen Vorgehens und da der Versuchsplan immer nur für die aktuelle Parameterschätzung optimal ist, wird von einem sequentiellen Fisher-Information basiertem Design gesprochen. Das sequentielle D-optimale Design wurde kürzlich zur Identifikation statischer Neuronaler Netze vorgeschlagen [91]. Eine Diskussion der Auswirkungen der sequentiellen D-optimalen Versuchsplanung auf die Parameterschätzung findet sich in [67].

3.5 Limitbehandlung

Bestimmte Bereiche des Eingangsraumes können zu Limitverletzungen führen. So verursachen mitunter bestimmte Ventilsteuerzeitenkombinationen kritische Werte bei der Laufruhe (Zündaussetzer). Um eine Beschädigung des Prüflings zu vermeiden, gilt es Limitverletzungen bei der Vermessung zu verhindern. Werden während der Datenerfassung Limitmodelle erstellt, welche die zulässigen Bereiche des Eingangsraums begrenzen, können diese als Nebenbedingungen an den Eingangsraum in der weiteren Versuchsplanung berücksichtigt werden. Damit ist es möglich Limitverletzungen auf ein Mindestmaß zu reduzieren. Bei der Wahl der Limitmodelle spielt das Vorwissen um die zulässigen Bereiche im Eingangsraum eine erhebliche Rolle. Die Limitmodellierung wird in [48] ausführlich untersucht und im Rahmen dieser Arbeit wird auf die dort entwickelten Limitmodelle zurückgegriffen, ohne diese hier näher zu beschreiben.

3.6 Diskussion im Sinne der Aufgabenstellung

Um die Dynamik der Zielgrößen des Verbrennungsmotors identifizieren zu können, muss diese hinreichend angeregt werden. Dabei ist allerdings zu beachten, dass der Motor immer in einem sicheren Bereich, das heißt ohne Fehlzündungen (Klopfen, Zündaussetzer) und im zulässigen Temperatur- sowie Druckbereich, betrieben wird. Da üblicherweise die Bereiche des Eingangsraums, die zu Limitverletzungen führen, nicht exakt bekannt sind, muss während der Systemanregung flexibel auf Limitverletzungen reagiert oder anfangs durch aufwändige Variationsraumvermessung der zulässige Eingangsraum bestimmt werden [70]. Da die beim Ausloten der Eingangsraumgrenzen anfallenden Messdaten nicht zum Training verwendet werden können, ist ein Verzicht auf die Variationsraumvermessung und eine flexible Reaktion auf Limitverletzungen während der Systemanregung zu bevorzugen. Dies ist bei Verwendung des APRBS nicht möglich, da dieses Sprünge im Eingangsraum erfordert, welche unter Umständen durch Anspringen eines unzulässigen Bereiches zur Beschädigung des Prüflings führen können. Die vorgestellten Verfahren zur Offline-Versuchsplanung können auch zur Verteilung der Designpunkte des APRBS verwendet werden. Allerdings können die Online-Verfahren nicht übernommen werden, da diese nicht den zeitlichen Verlauf des Anregungssignals berücksichtigen.

4 Stand der Technik: Identifikation dynamischer nichtlinearer Systeme

Mit den im vorherigen Abschnitt vorgestellten Verfahren der Versuchsplanung ist es möglich, ein für die Problemstellung geeignetes Anregungssignal zu erzeugen. Auf Basis der damit gewonnenen Messdaten soll ein Modell gebildet werden, welches das dynamische Verhalten des zu identifizierenden System möglichst exakt nachbildet.

In [1] wird eine Vielzahl verschiedener Modellbildungsverfahren zur Identifikation dynamischer Systeme miteinander verglichen. Die Modellbildungsverfahren werden anhand zahlreicher Datensätze evaluiert und die Autoren zeigen, dass sich besonders Neuronale Netze und Gauß'sche Prozesse zur Identifikation dynamischer Zusammenhänge eignen. Ähnliche Untersuchungen wurden auch in den Arbeiten [47], [82] durchgeführt, die sich mit der stationären Online-Optimierung beschäftigt haben. Dabei hat sich gezeigt, dass sich besonders (lokal) lineare Modelle, die in [1] nicht berücksichtigt werden, und Neuronale Netze zur Modellierung von statischen Zusammenhängen eignen. Die Gauß'schen Prozesse sind in der Funktionsauswertung rechenaufwändig (die Rechenzeit steigt kubisch mit der Anzahl der Trainingsdaten [69] an), wodurch die Online-Versuchsplanung und die Optimierung der Modelle zu zeitintensiv [82] wird (siehe hierzu auch die Diskussion in Abschnitt 2.3.2). Daher werden in dieser Arbeit ausschließlich Neuronale Netze und lokal lineare Modelle betrachtet, deren Rechenzeit linear mit dem Trainingsdatenumfang ansteigt [14], [60].

Bei der Identifikation dynamischer Zusammenhänge am Motorprüfstand kommen bisher ausschließlich Ein-/Ausgangsmodelle zum Einsatz, welche stets ein Training der Modelle in Prädiktor-Konfiguration ermöglichen. Diese Konfiguration wird bei lokal linearen Modellen als auch Neuronalen Netzen verwendet. Bei lokal linearen Modellen können so die Gewichte über lineare Optimierungsverfahren bestimmt werden. Bei den Neuronalen Netzen wird durch das initiale Training in der Prädiktor-Konfiguration das nachfolgende Training in Simulator-Konfiguration beschleunigt [21]. Deshalb erfolgt an dieser Stelle die Beschreibung der Modellbildungsverfahren für Ein-/Ausgangsmodelle. Das Training von Neuronalen Netzen in Zustandsraumdarstellung wird im nachfolgenden Kapitel 5 beschrieben.

Das Ziel dieses Kapitels ist es Neuronale Netze und lokal-lineare Modelle hinsichtlich der Anforderungen, welche aus der Aufgabenstellung resultieren, zu untersuchen. Dementsprechend hängt der Detaillierungsgrad der Beschreibungen davon ab, ob die in den folgenden Absätzen vorgestellten Ergebnisse im weiteren Verlauf der Arbeit noch benötigt werden.

4.1 Neuronale Netze

Neuronale Netze stellen eine weit verbreitete Klasse von Funktionsapproximatoren dar. Das sogenannte Multi-Layer-Perceptron (MLP) ist, wie in [1] gezeigt, besonders gut zur Identifikation dynamischer Systeme geeignet. Der Vorteil des MLP liegt darin, dass es ein universeller Approximator ist und in der Regel mit einer geringen Anzahl von Parametern auskommt. Dadurch lässt sich das Modell einfach auswerten. Dies ist ein wichtiger Gesichtspunkt bei der Versuchsplanung und Optimierung. Allerdings ist das MLP stark nichtlinear in den Parametern, wodurch das Ergebnis der Parameteroptimierung vom Startpunkt der Optimierung abhängt und gegebenenfalls unterschiedliche Startwerte (initiale Parameter) getestet werden müssen, um ein befriedigendes Ergebnis zu erhalten.

4.1.1 Zeitdiskrete Neuronale Netze

Ein MLP mit einer versteckten Schicht ist wie folgt definiert

$$\hat{y}_k = f_{MLP}(\varphi_k, \theta) = W^o \sigma\left(W^h \varphi_k\right), \tag{4.1}$$

wobei $\hat{y} \in \mathbb{R}^{N_O}$ der Netzausgang und der Regressor $\varphi_k \in \mathbb{R}^{N_R}$ der Eingang des Neuronalen Netzes zum Zeitpunkt k ist. Der Vektor $\sigma(\cdot) = [\tanh(\cdot), \ldots, \tanh(\cdot), 1]^\top$ ist der Ausgang der versteckten Schicht mit N_N Neuronen. Die Matrizen $W^o \in \mathbb{R}^{N_O \times N_N + 1}$, $W^h \in \mathbb{R}^{N_N \times N_R}$ sind Gewichtsmatrizen der versteckten Schicht beziehungsweise der Ausgangsschicht. Eine schematische Darstellung für $N_R = 4$, $N_N = 6$ und $N_O = 3$ findet sich in Abbildung 4.1.

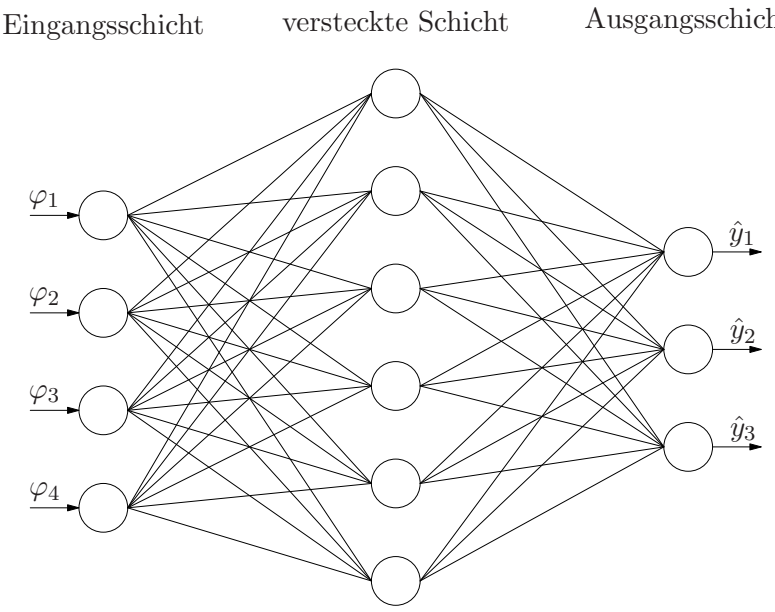


Abbildung 4.1: Aufbau Multi-Layer-Perceptron: Kanten repräsentieren die Gewichte und Knoten die Ausgänge der jeweiligen Schicht. Die Ausgänge zum Zeitpunkt k dienen in Simulator-Konfiguration im darauf folgenden Zeitschritt $k+1$ als Eingänge. Siehe auch Abbildung 2.1 und Abbildung 2.2

Um das MLP in die parametrische Form von (2.5) beziehungsweise (2.6) zu bringen, werden die Zeilen der Gewichtsmatrizen im Parametervektor $\theta = [W_1^o \dots W_{N_O}^o \; W_1^h \dots W_{N_R}^h]^\top \in \mathbb{R}^{N_p}$ zusammengefasst, wobei $N_p = N_N(N_O + N_R) + N_O$ ist. Der Regressorvektor φ_k und seine Komponenten hängen dabei von der Konfiguration ab. In Simulator-Konfiguration (2.6) ist er gegeben durch:

$$\varphi_k = \left[\hat{y}_{k-1}^\top, \dots, \hat{y}_{k-m}^\top, u_{k-1}^\top, \dots, u_{k-n}^\top, 1\right]^\top \tag{4.2}$$

und in Prädiktor-Konfiguration (2.5) durch

$$\varphi_k = \left[y_{k-1}^\top, \dots, y_{k-m}^\top, u_{k-1}^\top, \dots, u_{k-n}^\top, 1\right]^\top. \tag{4.3}$$

In der Simulator-Konfiguration weist das Neuronale Netz im Gegensatz zur Prädiktor-Konfiguration rekurrente Strukturen auf, da der Modellausgang zum Zeitpunkt k als Modelleingang zum Zeitpunkt $k+1$ dient. Dementsprechend ist die Parameterschätzung eines Neuronalen Netzes in Simulator-Konfiguration rechenaufwändiger als in Prädiktor-Konfiguration.

4.1.2 Parameterschätzung

Zum Training Neuronaler Netze stehen eine Vielzahl an Algorithmen zur Verfügung. Ein guter Überblick findet sich beispielsweise in [5] und [62]. Die Wahl des Trainingsalgorithmus ist dabei oft problemspezifisch und abhängig von der Topologie des Neuronalen Netzes. Zum Training vom MLP-Netzen mit wenigen hundert Gewichten ist der im Anhang B beschriebene und in dieser Arbeit verwendete Levenberg-Marquardt (LM) Algorithmus meist die beste Wahl [33]. Der Algorithmus benötigt wie die meisten in der Praxis eingesetzten Lernverfahren den Gradienten der Kostenfunktion. An dieser Stelle wird der Gradient für den quadratischen Fehler des Netzausganges hergeleitet und auf die Verwendung des Minkowski Fehlers verzichtet, da der LM-Algorithmus eine quadratische Fehlerfunktion voraussetzt. Soll der Minkowski Fehler in Verbindung mit einem anderen Optimierungsverfahren verwendet werden, lässt sich dies durch Anpassung der Kostenfunktion und des Gradienten nach (2.10) beziehungsweise (2.11) erreichen.

Die Ableitung des j-ten Ausgangs des Neuronalen Netzes nach den Gewichten ist gegeben durch

$$\frac{\partial \hat{y}_{k,j}}{\partial w_{p,q}^o} = \begin{cases} \sigma\left(W^h \varphi_k\right)_q & \text{wenn } p = j \\ 0 & \text{sonst} \end{cases}$$

$$\frac{\partial \hat{y}_{k,j}}{\partial w_{p,q}^h} = w_{j,p}^o \left(1 - \tanh\left(W^h \varphi_k\right)_p^2\right) \varphi_{k,q}.$$

In Simulator-Konfiguration wird aufgrund der rekurrenten Struktur des Neuronalen Netzes zusätzlich die totale Ableitung und die Ableitung nach den verzögerten Modellausgängen benötigt

$$\frac{\mathrm{d}\hat{y}_{k,j}}{\mathrm{d}w_{p,q}^o} = \frac{\partial \hat{y}_{k,j}}{\partial w_{p,q}^o} + \sum_{i=1}^{m} \sum_{l=1}^{N_O} \frac{\partial \hat{y}_{k,j}}{\partial \hat{y}_{k-i,l}} \frac{\mathrm{d}\hat{y}_{k-i,l}}{\mathrm{d}w_{p,q}^o}$$

$$\frac{\mathrm{d}\hat{y}_{k,j}}{\mathrm{d}w^h_{p,q}} = \frac{\partial \hat{y}_{k,j}}{\partial w^h_{p,q}} + \sum_{i=1}^{m}\sum_{l=1}^{N_O} \frac{\partial \hat{y}_{k,j}}{\partial \hat{y}_{k-i,l}} \frac{\mathrm{d}\hat{y}_{k-i,l}}{\mathrm{d}w^h_{p,q}}$$

$$\frac{\partial \hat{y}_{k,j}}{\partial \hat{y}_{k-i,l}} = \sum_{n=1}^{N_N} w^o_{j,n} \left(1 - \tanh\left(W^h \varphi_k\right)^2_n\right) w^h_{n,(i-1)N_O+l}.$$

Um das Neuronale Netz in die parametrische Form (2.6) zu bringen, werden die Zeilen der Gewichtsmatrizen im Parametervektor θ zusammengefasst. Der zu minimierende quadratische Approximationsfehler ist gegeben durch

$$E(\theta) = \frac{1}{2}\sum_{k=1}^{K}\left(\sum_{i=1}^{N_O}\left(y_i(k) - \hat{y}_i(k)\right)^2\right). \tag{4.4}$$

Der Gradient der Kostenfunktion (4.4) berechnet sich somit wie folgt

$$\nabla E(\theta) = -J(\theta)^\top e(\theta) \tag{4.5}$$

mit dem Fehlervektor $e(\theta) = [e_1^\top(\theta), \ldots, e_K^\top(\theta)]^\top$, $e_k(\theta) = (y_k - \hat{y}_k)$ und der Jakobimatrix

$$J(\theta) = \begin{bmatrix} \frac{\mathrm{d}f_{MLP}(\varphi_1,\theta)}{\mathrm{d}\theta_1} & \frac{\mathrm{d}f_{MLP}(\varphi_1,\theta)}{\mathrm{d}\theta_2} & \cdots & \frac{\mathrm{d}f_{MLP}(\varphi_1,\theta)}{\mathrm{d}\theta_p} \\ \frac{\mathrm{d}f_{MLP}(\varphi_2,\theta)}{\mathrm{d}\theta_1} & \frac{\mathrm{d}f_{MLP}(\varphi_2,\theta)}{\mathrm{d}\theta_2} & \cdots & \frac{\mathrm{d}f_{MLP}(\varphi_2,\theta)}{\mathrm{d}\theta_p} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathrm{d}f_{MLP}(\varphi_K,\theta)}{\mathrm{d}\theta_1} & \frac{\mathrm{d}f_{MLP}(\varphi_K,\theta)}{\mathrm{d}\theta_2} & \cdots & \frac{\mathrm{d}f_{MLP}(\varphi_K,\theta)}{\mathrm{d}\theta_p} \end{bmatrix}. \tag{4.6}$$

4.1.3 Regularisierung

Wie bereits im Abschnitt 2.2 über das Bias-Varianz-Dilemma diskutiert, erlauben es die Hyperparameter α und β die Modellflexibilität festzulegen. Dabei ist die optimale Modellflexibilität einerseits vom Umfang der Trainingsdaten und andererseits vom Modellierungsfehler abhängig. Eine elegante adaptive und datenbasierte Methode zur Ermittlung der Parameter α und β stellt die Bayes'sche Regularisierung dar, die in [51] vorgestellt und in [28] erstmals mit dem LM-Training kombiniert wird. Die Idee der Bayes'schen Regularisierung besteht darin, die bedingte Wahrscheinlichkeit (a posteriori Verteilung) der Gewichte abhängig von den vorhandenen Beobachtungen $\mathcal{D} = \{y_1, \ldots, y_K\}$ mittels des Bayes'schen Theorems zu berechnen [6]

$$p(\theta|\mathcal{D}, \alpha, \beta) = \frac{p(\mathcal{D}|\theta, \alpha, \beta)p(\theta)}{p(\mathcal{D})}. \tag{4.7}$$

Dabei bezeichnet $p(\mathcal{D}|\theta, \alpha, \beta)$ die Wahrscheinlichkeit bei gegebenem Parametervektor θ und gegebenen Regularisierungsparametern α, β den Datensatz $\mathcal{D}$ zu beobachten und $p(\theta)$ die a priori Verteilung der Gewichte. Der Nenner $p(\mathcal{D}) = \int p(\mathcal{D}|\theta, \beta)p(\theta)\,\mathrm{d}\theta$ ist eine Normalisierungskonstante und stellt sicher, dass die a posteriori Verteilung der Gewichte eine gültige Wahrscheinlichkeitsdichtefunktion ist. Da in der Regel kein Vorwissen über die

Modellgewichte vorhanden ist, wird für $p(\theta)$ die multivariate Normalverteilung (A.2)

$$p(\theta) = \mathcal{N}\left(\theta|0, \alpha^{-1}I\right)$$

und für $p(\mathcal{D}|\theta, \beta, f_{MLP}(\theta)) = \prod_{k=1}^{K} \mathcal{N}\left(y_k|f_{MLP}(\varphi_k, \theta), \alpha^{-1}I\right)$ angesetzt, womit sich die Lösung für (4.7) in Abhängigkeit von den Regularisierungsparametern berechnen lässt.

Auf eine Herleitung der Regularisierungsparameter, welche (4.7) maximieren, wird an dieser Stelle verzichtet und auf [51] beziehungsweise [6] verwiesen. Die Regularisierungsparameter ermitteln sich mit der Hessematrix $H = 2\beta J^\top J + 2\alpha I$ (siehe Anhang B) in jeder Iteration wie folgt:

$$\begin{aligned}
\alpha &= \frac{\gamma}{2\left|\left|\theta\right|\right|_2^2 + \operatorname{trace}\left(H^{-1}\right)}, \\
\beta &= \frac{N_p - \gamma}{2\left|\left|e\right|\right|_2^2}, \\
\gamma &= N_p - \alpha \operatorname{trace}\left(H^{-1}\right).
\end{aligned}$$

Dabei kann γ als die Anzahl der aktiven Parameter aufgefasst werden [6]. Mittels γ kann somit überprüft werden, ob das Modell für die gegebene Problemstellung genügend Flexibilität, das heißt eine genügend große Anzahl an Parametern, besitzt. Während des Trainings kann diese Information, wie in [65] vorgeschlagen, zur automatischen Adaption der Neuronenzahl genutzt werden. Überschreitet das Verhältnis γ, bezogen auf die Gesamtzahl der Parameter N_p, einen bestimmten Schwellwert (beispielsweise $\gamma > 0.8N_p$), wird die Neuronenzahl erhöht. Um dies im laufenden Training durchführen zu können, muss sichergestellt werden, dass das neu hinzugefügte Neuron nicht den Modellausgang beeinflusst. Dies kann beispielsweise erreicht werden, indem die Eingangsgewichte des Neurons zufällig initialisiert werden, während das Neuron in der Ausgangsschicht mit Null gewichtet wird.

4.1.4 Bifurkation und Stabilität

Im Gegensatz zu linearen Systemen können nichtlineare Systeme abhängig vom zeitlichen Verlauf beziehungsweise Ausgangswert x_0 für einen konstanten Eingang mehrere Ruhelagen respektive Stationärwerte aufweisen. Dieses Verhalten wird auch Bifurkation genannt. Da die Zielgrößen des Verbrennungsmotors nicht dieses Verhalten aufweisen, ist sicherzustellen, dass auch der nichtlineare Funktionsapproximator für jeden konstanten Eingang eine eindeutige Ruhelage besitzt. Die lässt sich durch Bedingungen an die Gewichtsmatrizen erreichen.

Behauptung 1. *Ein MLP nach Gleichung (4.1) besitzt für jeden beliebigen aber konstanten Eingang $u := u_k \ \forall k$, genau dann eine eindeutige Ruhelage $y := y_k \ \forall k$, falls die Gewichtsmatrizen der versteckten Schicht $W^h := [W^{h,y},\ W^{h,u},\ b^h]$ und der Ausgangsschicht W^o folgender Bedingung genügen:*

$$||W^o||\ \left|\left|W^{h,y} J^\top_{N_O \times m}\right|\right| < 1,$$

wobei $J_{N_O,m} := [I_{N_O}, \ldots I_{N_O}] \in \mathbb{R}^{N_O(1\times m)}$.

Beweis. Aus der geforderten Ruhelage folgt mit $J_{i,j} := [I_i, \dots I_i] \in \mathbb{R}^{i(1\times j)}$ für den Regressor

$$\varphi = \left[y^\top J_{N_O\times m},\ u^\top J_{N_I\times m},\ 1\right]^\top. \tag{4.8}$$

Um eine eindeutige Ruhelage zu gewährleisten, muss die vom Funktionsapproximator beschriebene Ein-/Ausgangsabbildung eine Kontraktion nach Theorem 7 (Anhang D) darstellen:

$$||T(y_1) - T(y_2)|| = \left|\left|W^o\sigma\left(W^h\varphi_1\right) - W^o\sigma\left(W^h\varphi_2\right)\right|\right| < ||y_1 - y_2||\,.$$

Dies führt mit $\|\tanh(x_1) - \tanh(x_2)\| \leq \|x_1 - x_2\|$ zu folgender Abschätzung, die sich in ähnlicher Form auch in [90] findet:

$$\begin{aligned}
||T(y_1) - T(y_2)|| &= \left|\left|W^o\sigma\left(W^h\varphi_1\right) - W^o\sigma\left(W^h\varphi_2\right)\right|\right| \\
&\leq ||W^o||\,\left|\left|\sigma\left(W^h\varphi_1\right) - \sigma\left(W^h\varphi_2\right)\right|\right| \\
&\leq ||W^o||\,\left|\left|W^h\varphi_1 - W^h\varphi_2\right|\right| \\
&= ||W^o||\,\left|\left|W^{h,y}\left(J^\top_{N_O\times m}y_1 - J^\top_{N_O\times m}y_2\right)\right|\right| \\
&\leq ||W^o||\,\left|\left|W^{h,y}J^\top_{N_O\times m}\right|\right|\,||y_1 - y_2||\,.
\end{aligned}$$

Aus dieser Abschätzung folgt unmittelbar die Behauptung 1. □

Neben der Vermeidung von Bifurkation gilt es die Stabilität von dynamischen Modellen sicherzustellen. Dabei bedeutet in dieser Arbeit stabil, dass beschränkte Eingänge stets beschränkte Zustände und Ausgänge zur Folge haben und bei konstantem Eingang sich das System einer Ruhelage nähert (siehe Anhang D). Um die Stabilität von (4.1) gemäß dem in [90] verwendeten Ansatz zu untersuchen, wird die Differenzengleichung (4.1) m-ter Ordnung in eine Differenzengleichung 1. Ordnung mit Ruhelage im Ursprung überführt:

$$x_{k+1} = Ax_k + BW^o\Theta(x_k) \tag{4.9a}$$

$$y_k = B^\top x_k + y. \tag{4.9b}$$

Dabei ist y die Ruhelage des Systems (4.1) und

$$\begin{aligned}
x_{k-1} &= [(\,y_{k-1} - y)^\top, \dots, (y_{k-m} - y)^\top]^\top \\
A &= \begin{bmatrix} 0 & 0 \\ I_{N_O(m-1)} & 0 \end{bmatrix} \quad \in \mathbb{R}^{N_Om\times N_Om} \\
B &= [I_{N_O}, 0_{N_O}, \dots, 0_{N_O}]^\top \quad \in \mathbb{R}^{N_O\times N_Om} \\
\Theta(x_k) &= \sigma\left(W^{h,y}x_k + W^h\varphi\right) - \sigma(W^h\varphi).
\end{aligned}$$

Der Vektor φ ist der Regressor (4.8) des Systems (4.1) in Ruhelage für einen konstanten, aber beliebigen Eingang $u_{k+1} = u_k := u \quad \forall k$ und $\Theta(x_k)$ ergibt sich durch die Überführung der Ruhelage in den Ursprung:

$$\begin{aligned}
y_k - y &= W^o\sigma\left(W^h\varphi_k\right) - y \\
&= W^o\sigma\left(W^{h,y}x_{k-1} + W^h\varphi\right) - W^o\sigma\left(W^h\varphi\right).
\end{aligned}$$

Um die Bedingungen für Stabilität der Ruhelage zu zeigen, wird die Lyapunov-Funktion (siehe Abschnitt Stabilitätsanalyse im Anhang D)

$$V(x_k) = x_k^\top P x_k$$

mit positiv definiter Matrix P angesetzt. Offensichtlich genügt diese Lyapunov-Funktion $V(x_{k+1}) - V(x_k) < 0$, wenn

$$\begin{aligned} 0 &< x_k^\top P x_k - (Ax_k + BW^o\Theta(x_k))^\top P\,(Ax_k + BW^o\Theta(x_k)) \\ 0 &< x_k^\top (P - A^\top PA)x_k - \Theta(x_k)^\top W^{o\top} B^\top PBW^o\Theta(x_k) - 2\left(x_k^\top A^\top PBW^o\Theta(x_k)\right). \end{aligned} \tag{4.10}$$

Über eine Normabschätzung mit positiv-definiter Diagonalmatrix P lässt sich folgende Bedingung an die Gewichte ermitteln [90]:

Theorem 1. *Ein MLP nach Gleichung (4.1) besitzt für jeden beliebigen aber konstanten Eingang $u := u_k\ \forall k$, genau dann eine global exponentiell stabile Ruhelage $y := y_k\ \forall k$, falls die Gewichtsmatrizen der versteckten Schicht $W^h := [W^{h,y},\ W^{h,u},\ b^h]$ und der Ausgangsschicht W^o folgender Bedingung genügen:*

$$||W^o||_2 \left|\left|W^{h,y}\right|\right|_2 < \sqrt{\frac{2(N_O N_N - 1)^{N_O N_N - 1}}{(N_O N_N - 1 + \sqrt{2})((N_O N_N - 1 + \sqrt{2})^{N_O N_N} - (N_O N_N - 1)^{N_O N_N})}}.$$

Der Beweis findet sich in [90]. Wird gefordert, dass P eine positiv-definite Diagonalmatrix ist, lässt sich auch ein alternativer Stabilitätsbeweis finden:

Theorem 2. *Ein MLP nach Gleichung (4.1) besitzt für jeden beliebigen aber konstanten Eingang $u := u_k\ \forall k$, genau dann eine global exponentiell stabile Ruhelage $y := y_k\ \forall k$, falls die Gewichtsmatrizen der Ausgangsschicht W^o folgender Bedingung genügen:*

$$W^{o\top} P_r W^o \leq 0, \tag{4.11}$$

wobei $P_r = \operatorname{diag}(p_1, \ldots, p_{N_O})$ und $p_1 > p_2 > \ldots > p_{N_O}$.

Beweis. Ist P eine Diagonalmatrix, ist der Ausdruck $A^\top PB$ eine Nullmatrix. Dies ist leicht ersichtlich, wenn $P = I$ gesetzt wird:

$$\begin{bmatrix} 0 & I_{N_O(m-1)} \\ 0 & 0 \end{bmatrix} [I_{N_O}, 0_{N_O}, \ldots, 0_{N_O}]^\top = 0 \in \mathbb{R}^{N_O m \times N_O}.$$

Für $P = \operatorname{diag}(p_1, \ldots, p_{N_O m})$ mit $p_1 > \ldots > p_{N_O m}$ gilt zudem $P - A^\top PA > 0$, da $A^\top PA = \operatorname{diag}(p_{N_O+1}, \ldots, p_{N_O m}, 0, \ldots, 0)$. Somit muss nur der Ausdruck $W^{o\top} B^\top PBW^o$ des Gleichungssystems 4.10 negativ definit sein, damit Stabilität gewährleistet ist. Da $B^\top PB = P_r$ ist, folgt Theorem 2. □

Der Vorteil des Theorems 1 gegenüber Theorem 2 ist, dass nicht eine Matrixungleichung als Nebenbedingung formuliert werden muss, wodurch sich der LM-Algorithmus dahingehend modifizieren lässt, dass die Nebenbedingungen aus Theorem 1 stets eingehalten werden [90].

Die hier vorgestellten Nebenbedingungen an die Gewichte stellen zwar die Stabilität der Ruhelage sicher, aber nicht dass beschränkte Eingänge auch beschränkte Ausgänge zur Folge haben. Bei zeitdiskreten Neuronalen Netzen der Form 4.1 ist allerdings leicht ersichtlich, dass der Ausgang stets begrenzt ist, da $||y||$ mit (C.1) stets durch $||W^o||$ beschränkt ist.

4.2 Lokal lineare Modelle

Verfahren zur Bildung lokal linearer Modelle unterteilen den Eingangs- beziehungsweise Regressorraum in Teilbereiche. Jeder dieser Teilbereiche wird mittels eines lokal linearen Modells beschrieben. Lokal linear bedeutet hier, dass die Modelle lokal gültig und linear in ihren Parametern sind, somit können zum Beispiel auch Polynomialmodelle Verwendung finden. Der Vorteil lokaler Modelle liegt einerseits in der einfachen Parameterschätzung in Prädiktor-Konfiguration, da es sich um ein lineares Optimierungsproblem handelt [60]. Andererseits lässt sich auch das globale Optimum einfach bestimmen, indem die Extrema der Einzelmodelle bestimmt und diese miteinander vergleichen werden.

Bei den lokalen Ansätzen wird das Modell abhängig vom aktuellen Betriebspunkt (definiert durch den Regressor oder Teilen davon) in lokale Untermodelle aufgeteilt. Dadurch wird eine größere Flexibilität erreicht und die Modelle können (im Gegensatz zu globalen Polynomialmodellen) eine große Klasse von nichtlinearen Systemen beschreiben. Ein weiterer Vorteil ist, dass sie sich durch zusätzliche lokale Modelle selbst an die Komplexität des Problems anpassen können [60]. Dies führt zu robusten Modellen mit schnellen Trainingszeiten. Allerdings ist die Flexibilität eines lokal linearen Modells durch die Anzahl der Trainingsdaten eingeschränkt. Da jedes Teilmodell für jeden Parameter eines lokalen Modells zumindest einen Trainingsdatenpunkt benötigt, können nur soviele Teilmodelle erstellt werden, wie es der Trainingsdatenumfang zulässt. Dies stellt bei hochdimensionalen Eingangsräumen $N_I >> 1$ und Polynomen l-ter Ordnung ein Problem dar, da dort die Eingangsdimension ein exponentielles Ansteigen der Koeffizienten mit sich bringt [60]

$$N_p = \frac{(N_I + l)!}{N_I! l!}.$$

Daher werden in der Praxis meist nur lineare Modelle verwendet oder die Parameterzahl bei Polynomialmodellen reduziert, indem Mischterme vernachlässigt werden.

Ist der Übergang zwischen den lokalen Modellen „unscharf“ (englisch: *fuzzy*), wird das Modell auch als Neuro Fuzzy Modell bezeichnet. Die bekannteste Modellklasse von Neuro Fuzzy Modellen stellen die regelbasierten Takagi-Sugeno Fuzzy Modelle dar, welche in [83] vorgeschlagen und im Folgenden näher vorgestellt werden.

4.2.1 Aufbau des Takagi-Sugeno Fuzzy Modells

Das Takagi-Sugeno Fuzzy Modell, kurz TS-Modell, gehört zu den regelbasierten Modellen. Bei dieser Modellklasse wird das zu approximierende System durch Regeln oder Teilmodelle dargestellt. Die Regeln M_i eines TS-Modells basieren auf dem Modelleingang (Regressor)

φ und haben die folgende Form

$$\begin{aligned} R_i: \quad & if \quad (\varphi_1 \; is \; M_{i,1}) \; and \; ... \; and \; (\varphi_{N_R} \; is \; M_{i,N_R}) \\ & then \quad y_i = \beta_{i,1}\varphi_1 + ... + \beta_{i,\nu}\varphi_{N_R}. \end{aligned}$$

Dabei ist $M_{i,j}$ der Bereich der j-ten Dimension, in dem das i-te lokale Modell gültig ist. Der Rückgabewert y_i der i-ten Regel oder des i-ten Teilmodells hängt linear von den Parametern β_i ab

$$y_i(\varphi) = \beta_i^T \varphi = \sum_{j=1}^{N_R} \beta_{i,j}\varphi_j. \tag{4.12}$$

Jedes dieser lokalen Modelle hat ein Zentrum und eine Umgebung, in dem es gültig ist und das System approximiert. Der Einfluss eines lokalen Modells im jeweiligen Bereich des Regressorraumes wird durch eine Gauß-Funktion mit Zentrum φ^* und Standardabweichung σ, die in jeder Dimension verschieden sein kann, festgelegt

$$\mu_i(\varphi) = exp\left(-\frac{1}{2} \cdot \sum_{j=1}^{N_R} \frac{\left(\varphi_j - \varphi_{i,j}^*\right)^2}{\sigma_{i,j}^2}\right).$$

Damit sich die Gauß-Funktionen, im Rahmen der lokalen Modelle auch Zugehörigkeitsfunktionen genannt, überall im Regressorraum konstant auf 1 addieren, ist es nötig diese zu normieren. Der Aktivitätsgrad Φ_i einer Regel oder eines lokalen Modells ermittelt sich nun wie folgt

$$\Phi_i(\varphi) = \frac{\mu_i(\varphi)}{\sum_{j=1}^{N} \mu_j(\varphi)} \qquad , \qquad \sum_{i=1}^{N} \Phi_i(\varphi) = 1.$$

Um den Ausgang y des gesamten Modells zu erhalten, werden die Ausgänge y_i der N lokalen Modelle, gewichtet mit dem Aktivitätsgrad Φ_i, aufaddiert

$$y(\varphi) = \sum_{i=1}^{N} \Phi_i(\varphi) \cdot y_i(\varphi).$$

Wird für eine Regel M_i in der eben beschriebenen Form weitere Unterregeln definiert, ergeben sich sehr flexible Modellstrukturen. Meist kommen jedoch die beiden folgenden Strukturen zum Einsatz:

Flache Struktur

Der Modellausgang wird beispielsweise durch $y = \Phi_1 y_1 + \Phi_2 y_2 + \ldots + \Phi_9 y_9$ gebildet. Eine schematische Darstellung findet sich in Abbildung 4.2. Bei dieser Struktur muss die Normalisierung der Zugehörigkeitsfunktionen Φ_i global ausgeführt werden, wodurch es zu Normalisierungsnebeneffekten wie Reaktivierung eines Untermodells oder zu nicht-dominanten Untermodellen kommen kann [60]. Um dies zu vermeiden, muss die Standardabweichung der Zugehörigkeitsfunktionen klein gewählt und sichergestellt werden, dass der Einzugsbereich eines jeden Modells von relevanter Größe ist.

Hierarchische Struktur

Der Modellausgang wird zum Beispiel durch $y = \Phi_1(\Phi_{11}y_1 + \Phi_{12}y_2) + \Phi_2(\Phi_{21}y_3 + \Phi_{22}(\Phi_{221}y_4 + \Phi_{222}y_4))$ gebildet, wobei $\Phi_2 = 1 - \Phi_1$. Bei dieser Struktur sind die Normalisierungsnebeneffekte weniger schwerwiegend. Die Normalisierung wird für jede Unterteilung getrennt durchgeführt, wodurch die Nebeneffekte unwahrscheinlich werden [60]. Falls diese dennoch auftreten, können diese durch das Anpassen der Standardabweichung der Aktivierungsfunktionen vermieden werden.

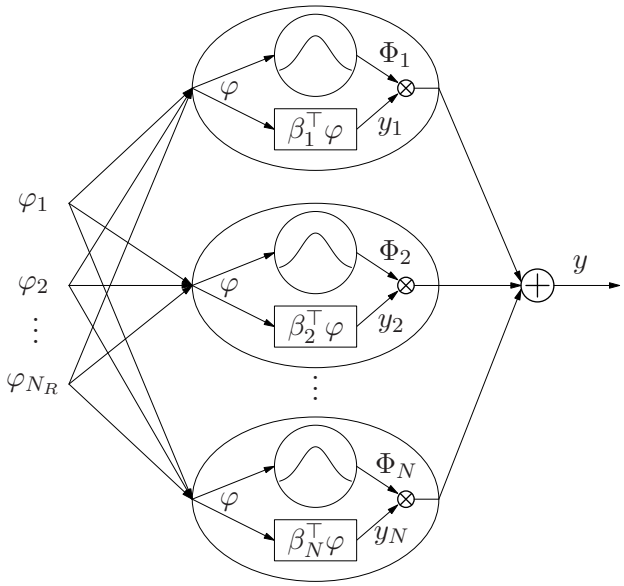


Abbildung 4.2: Aufbau Takagi-Sugeno Fuzzy Modell: flache Struktur.

Entsprechend der verwendeten Modellstruktur können auch die im folgenden vorgestellten lokalen Modellbildungsverfahren unterschieden werden.

4.2.2 LOLIMOT

Die Abkürzung LOLIMOT steht für LOcal LInear MOdel Tree und wie schon aus dem Namen hervorgeht, wird das Modell baumförmig aufgebaut und hat dementsprechend eine hierarchische Struktur [60]. Die Modellbildung erfolgt dabei offline, indem der Unterraum mit dem größten Trainingsfehler partitioniert wird. Der Algorithmus teilt den Unterraum achsen-orthogonal in alle möglichen Dimensionen und behält die Teilung, die zur größten Verbesserung geführt hat. Neben linearen Teilmodellen können bei diesem Verfahren auch polynomiale Modelle eingesetzt werden. Das Verfahren ist weit verbreitet und wird beispielsweise in den Arbeiten [31], [64] und in modifizierter Form in [74] eingesetzt.

4.2.3 Hinging Hyperplanes

Im Gegensatz zu LOLIMOT erlauben Hinging Hyperplanes eine beliebige Unterteilung des Eingangsraumes. Dabei wird wiederum zwischen flacher [7] und hierarchischer Struktur

[26], [85] unterschieden. Allerdings erlauben die Ansätze, die auf Hinging Hyperplanes basieren, nur lokal lineare Modelle. Daher liegen die Optima der Teilmodelle am Randbereich, wo die Modelle tendenziell ungenauer sind und die Überschneidung mit anderen Teilmodellen am Größten ist. Zudem führt das Partitionierungsverfahren bei großen Gradienten respektive Unstetigkeitsstellen zu Problemen. Da der zu partitionierende (Teil-)Raum immer in zwei Teilgebiete unterteilt wird, welche durch zwei sich schneidende lineare Modelle beschrieben werden, kann mitunter eine Stufenfunktion nur schwer beschrieben werden. Hierzu sind genügend Daten nahe der Unstetigkeitsstelle erforderlich, um ein lokales Modell erstellen zu können, welches die Flanke beschreibt. Solch große Gradienten treten beispielsweise bei Abgastemperaturen von Dieselmotoren auf, wenn im Teillastbereich die Abgasrückführung aktiv ist und somit die Abgastemperatur sprunghaft ansteigt.

4.2.4 Lokale Neuro-Fuzzy Modelle

In [42] und [41] wird ein weiteres Modellbildungsverfahren vorgestellt, das im Umfeld der Identifikation dynamischer Zusammenhänge am Verbrennungsmotor eingesetzt wird und eine beliebige Unterteilung des Eingangsraumes zulässt. Über einen EM-Algorithmus (Expectation-Maximization) wird die Lage der Zugehörigkeitsfunktion abhängig von der Parameterschätzung des lokalen Teilmodells bestimmt. Neben lokal linearen Modellen können auch Polynomialmodelle eingesetzt werden. Der Algorithmus weist, verglichen mit den beiden zuvor vorgestellten Verfahren, die größte Flexibilität auf, benötigt aber wie diese bereits zu Trainingsbeginn die gesamten Trainingsdaten.

Da die vorgestellten Modelltypen linear in den Parametern sind, können die Modellparameter in Prädiktor- beziehungsweise NARX-Konfiguration durch lineare Optimierungsverfahren bestimmt werden, wodurch ein schnelles Training möglich wird. Dabei muss das Messrauschen der verzögerten Systemausgänge in der Parameterschätzung berücksichtigt werden, um unverzerrte Schätzergebnisse zu erhalten. Dazu können verschiedene Schätzverfahren wie Instrumentalvariablen-Schätzer [60] oder Total Least Squares [52], [41] eingesetzt werden.

4.3 Diskussion im Sinne der Aufgabenstellung

Dynamische Neuronale Netze sind wie in [1] gezeigt besonders gut zur Identifikation dynamischer Systeme geeignet. Die Neuronalen Netze stellen universelle Approximatoren dar und können, sofern genügend Neuronen vorhanden sind, jedes System beliebig genau approximieren [77]. Die Verwendung der Bayes'schen Regularisierung beim Training Neuronaler Netze ist ein in der Praxis bewährtes Verfahren, welches es erlaubt, die Flexibilität des Neuronalen Netzes abhängig vom Trainingsdatensatz anzupassen [28]. Darüber hinaus existieren Nebenbedingungen an die Gewichte, welche die Simulationsstabilität des Neuronalen Netzes sicherstellen können [90]. Die feste parametrische Struktur der Neuronalen Netze erlaubt es zudem, die Komplexität der Modellauswertung abzuschätzen. Dies ist ein wichtiger Aspekt in der Online-Versuchsplanung, da die Zeit zur Modellauswertung berücksichtigt werden und somit bekannt sein muss.

Im Gegensatz zu Neuronalen Netzen sind lokal lineare Modelle und vor allem LOLIMOT Modelle bei Identifikationsaufgaben am Motorenprüfstand weit verbreitet. Lokal lineare Modelle unterteilen den Eingangsraum und approximieren das Systemverhalten durch Teilmodelle. Um die gewünschte Genauigkeit zu erzielen, kann eine Vielzahl von Teilmodellen und eine entsprechende starke Partitionierung des Eingangsraums erforderlich sein. Ist dies der Fall, stehen pro Teilmodell eine entsprechend begrenzte Anzahl an Trainingsdaten zur Verfügung, wodurch es bei verrauschten Messdaten zu Unsicherheiten in der Parameterschätzung kommt. Daher sollte das Partitionierungsverfahren eine flexible Unterteilung des Eingangsraums und die Verwendung von Polynomialmodellen erlauben, um mit möglichst wenig Teilmodellen bestmögliche Ergebnisse zu erzielen. Dies ist beispielsweise bei den lokalen Neuro-Fuzzy Modellen der Fall. Allerdings wächst bei allen lokal linearen Modellen, unabhängig von der Partitionierungsstrategie des Eingangsraums, die benötigte Anzahl der Messungen exponentiell mit der Anzahl der Eingänge [4]. Aus diesem Grund ist der Einsatz lokal linearer Modelle bei hochdimensionalen Eingangsräumen mit einem hohen Messaufwand verbunden.

In dieser Arbeit sollen die Modelle in der Simulator-Konfiguration verwendet werden, wodurch ein Training in NOE-Konfiguration dem in NARX-Konfiguration vorzuziehen ist. Darüber hinaus weisen, wie in Abschnitt 2.1.4 diskutiert, Modelle in NOE-Konfiguration eine wesentlich geringere Abhängigkeit von der Abtastzeit auf, wodurch Modelle in NOE-Konfiguration für die Aufgabenstellung der Arbeit generell die bessere Wahl darstellen. Werden lokal lineare Modelle in NOE-Konfiguration trainiert, müssen nichtlineare Optimierungsalgorithmen eingesetzt werden, wodurch der Geschwindigkeitsvorteil der lokal linearen Modelle beim Training verloren geht. Zudem ist der Rechenaufwand bei der Auswertung lokal linearer Modelle höher als bei Neuronalen Netzen. Um ein lokal lineares Modell simulieren zu können, müssen neben den lokal linearen Modellen auch deren Zugehörigkeitsfunktionen berechnet werden. Der Rechenaufwand bei der Auswertung wirkt sich entsprechend in der Modellauswertung und -optimierung aus. Beides sind sehr rechenintensive und oft benötigte Operationen in der Online-Versuchsplanung und Online-Optimierung. Zudem finden sich bisher keine Ergebnisse hinsichtlich der Stabilität von lokal linearen Modellen.

Aus den genannten Gründen stellen die Neuronalen Netze hinsichtlich der Aufgabenstellung den vielversprechenderen Ansatz dar. Im nachfolgenden Kapitel 5, in welchem die theoretischen Ergebnisse der Arbeit präsentiert werden, wird daher ausschließlich das MLP Neuronale Netz verwendet.

5 Online-Versuchsplanung und Identifikation dynamischer Zusammenhänge

Dieses und das folgende Kapitel beinhalten die wesentlichen Neuerungen der Arbeit und es werden die notwendigen Methoden der Versuchsplanung und Modellbildung vorgestellt, um eine Online-Versuchsplanung und Online-Identifikation dynamischer Zusammenhänge am Prüfstand realisieren zu können. Nachdem in der praktischen Anwendung eine Reihe von Rahmenbedingungen, wie schwankende Abtastraten, Limitverletzungen und Messausreißer berücksichtigt werden müssen, werden zunächst nochmals die Anforderungen formuliert. Anschließend werden auf Basis der Kapitel 3 und 4 die notwendigen Methoden der Versuchsplanung und Modellbildung erarbeitet und simulativ evaluiert. Im nächsten Kapitel wird der Ablauf der Online-Modellbildung am Prüfstand unter Verwendung der in diesem Kapitel vorgestellten Methoden erläutert und die nötigen Heuristiken zur Umsetzung in der Praxis vorgestellt.

5.1 Anforderungen

Nachdem in den vorhergehenden Kapiteln die Grundlagen der Modellbildung und Versuchsplanung diskutiert werden, wird an dieser Stelle nochmals auf die Anforderungen an diese eingegangen. Es soll eine Umgebung zur Online-Identifikation dynamischer Zusammenhänge am Motorprüfstand geschaffen werden. Um die Vorteile der Online-Modellbildung ausnützen zu können, muss die Versuchsplanung zur Laufzeit die Systemanregung so gestalten, dass der Informationsgewinn für die Modelle maximal ist. Zudem muss bei der Versuchsplanung gewährleistet werden, dass der Motor immer in einem sicheren Betriebszustand läuft, das heißt Limitverletzungen wie beispielsweise Laufunruhe und/oder zu hohe Temperaturen/Drücke müssen rechtzeitig erkannt und behoben werden. Dies erfordert einerseits ein Eingangssignal, welches das System so anregt, dass eine Limitbehandlung jederzeit möglich ist, und andererseits muss das Anregungssignal unter Berücksichtigung der zur Laufzeit erstellten Limitmodelle den optimalen Informationsgewinn gewährleisten. Die Gestaltung optimaler Versuchsplanung unter Berücksichtigung von Limits wird in Abschnitt 5.4 diskutiert.

Die Modellbildung muss mit den Betriebsbedingungen am Prüfstand zurechtkommen. Die Kommunikation am Prüfstand, wie in Abbildung 5.1 schematisch veranschaulicht, sieht eine Echtzeitkommunikation nur zwischen dem Prüfstandsautomatisierungssystem, der E-Maschine und den Steuer- beziehungsweise schnellen Messgeräten vor. Daher kann bei der Kommunikation zwischen dem Prüfstandsautomatisierungssystem und dem Optimierungssystem, auf welchem die Online-Versuchsplanung und -Identifikation läuft, mit keiner gleich bleibenden Abtastrate gerechnet werden. Die Schwankungen der Abtastrate sind, wie Tests

am Prüfstand gezeigt haben, relativ klein. Dennoch sollten sie berücksichtigt werden, um bestmögliche Ergebnisse zu erzielen. Eine weitere wesentliche Anforderung stellt die Sicherstellung der Stabilität der Modelle dar. Da die dynamischen Modelle zur Simulation verwendet, sowie die Stationärzustände aus ihnen abgeleitet werden sollen, ist es notwendig die Simulationsstabilität derselben sicherzustellen. Die Berücksichtigung schwankender Abtastraten ist mit den klassischen zeitdiskreten Neuronalen Netzen nicht möglich, weshalb im folgenden Abschnitt 5.2 die Runge-Kutta Neuronalen Netze (RKNN) eingeführt werden. Es wird erstmals ein Algorithmus zum Training von RKNN in Simulator-Konfiguration hergeleitet und Bedingungen an die Gewichte formuliert, um die Simulationsstabilität der RKNN sicherzustellen.

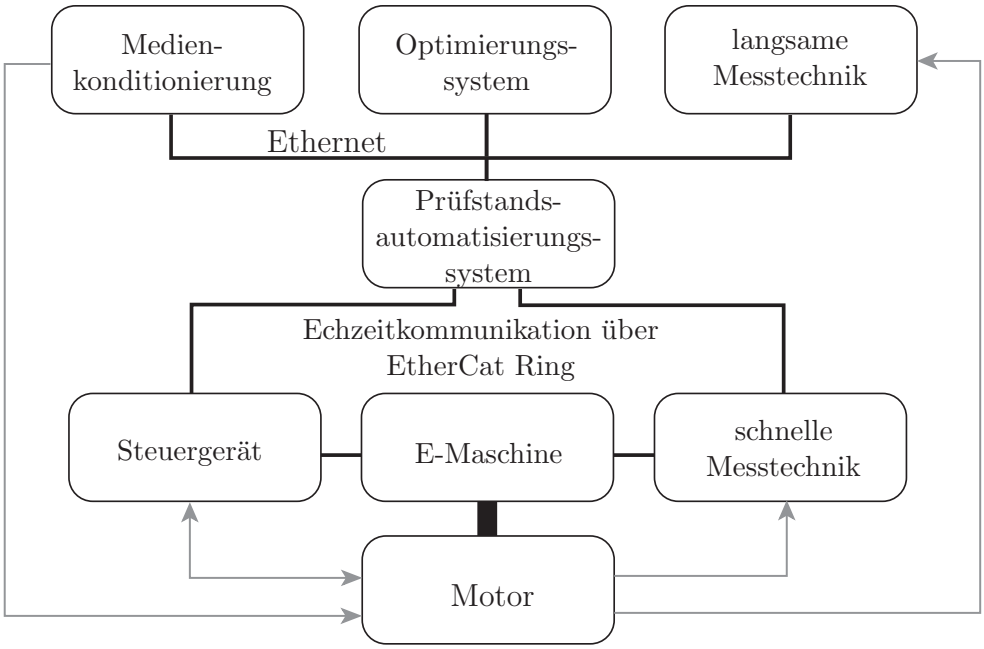


Abbildung 5.1: Schematische Darstellung der Prüfstandskommunikation.

5.2 Identifikation von Runge-Kutta Neuronalen Netzen

In Kapitel 4 werden unterschiedliche Ansätze für Modelle, welche im Bereich der Identifikation dynamischer Zusammenhänge am Verbennungsmotorenprüfstand zum Einsatz kommen, vorgestellt und hinsichtlich der Aufgabenstellung diskutiert. Wie in [1] gezeigt, eignen sich Neuronale Netze sehr gut zur Identifikation dynamischer Zusammenhänge und werden auch in vielen unterschiedlichen Bereichen zur Identifikation dynamischer Systeme eingesetzt. Die parametrische Struktur der Neuronalen Netze und deren in der Regel geringe Anzahl an Gewichten macht sie für die gegebene Aufgabenstellung ebenfalls sehr attraktiv (siehe Diskussion in Kapitel 4).

Um ein dynamisches System mit Neuronalen Netzen zu approximieren, wird das System gewöhnlich als zeitdiskretes oder als in erster Ordnung zeitdiskretisiertes System angesehen, welches durch gewöhnliche Differenzengleichungen beschrieben wird. Da Zustandsraummodelle eine größere Systemklasse als Ein-/Ausgangsmodelle beschreiben können (sie-

he Diskussion Kapitel 2), werden im Folgenden Neuronale Netze in Zustandsraumdarstellung verwendet.

Wie in [88] betont wird, bringen Modelle, welche das System als zeitdiskretes oder als in erster Ordnung zeitdiskretisiertes System ansehen, eine Reihe von Problemen mit sich: Größere Approximationsfehler werden durch die angenommene Diskretisierung erster Ordnung verursacht, die Genauigkeit der Langzeitprädiktion ist üblicherweise nicht so gut und eine Prädiktion kann nur zu festen Zeitintervallen erfolgen. Der Grund für die genannten Probleme ist die Tendenz von zeitdiskreten Neuronalen Netzen die Systemzustände anstatt der Änderung derselben zu lernen [88].

In [88] wird das Runge-Kutta Neuronale Netz (RKNN) zum Modellieren von gewöhnlichen Differentialgleichungen (*engl.: ordenary differential equation (ODE)*) der Form $\dot{x} = f(x, u)$ mit unbekanntem $f(\cdot)$ eingeführt. Als Neuronales Netz wird ein RBF (*engl. radial basis function*) - Netz [5] verwendet und dieses entsprechend der Runge-Kutta Approximationsmethode für ODE's [9] in Prädiktor-Konfiguration konstruiert. Dadurch wird, wie in [88] gezeigt, eine genaue Schätzung der Zustandsänderung der Systemzustände (das heißt der rechten Seite der Differentialgleichung) möglich. Daher weisen RKNN's nicht die zuvor erwähnten Probleme auf und sind hinsichtlich der Generalisierung und Langzeitprädiktion zeitdiskreten Modellen überlegen. Dies wird in [88] theoretisch bewiesen und in [25] anhand von Simulationsbeispielen gezeigt.

Obwohl RKNNs zeitdiskreten Black-Box Modellen hinsichtlich Approximationsgüte und Generalisierungsfähigkeit überlegen sind [88],[25], sind die in [88] getroffenen Annahmen sehr restriktiv, da eine rauschfreie Messung *aller* Systemzustände vorausgesetzt wird. Eine rauschfreie Messung ist in der Praxis kaum gegeben und zudem können meist nicht alle Systemzustände erfasst beziehungsweise als bekannt vorausgesetzt werden.

In diesem Abschnitt wird der Ansatz [88], um RKNN vierter Ordnung in Prädiktor-Konfiguration zu erzeugen, auf allgemeine Differentialgleichungen n-ter Ordnung erweitert. Im Gegensatz zu [88] wird als Neuronales Netz das MLP-Netz verwendet, welches als bestes Black-Box Modell zur Modellierung dynamischer Systeme gilt [1]. Es wird die Jakobimatrix und der Gradient des dynamischen Neuronalen Netzes in Prädiktor- und *Simulator*-Konfiguration hergeleitet. Dadurch wird die Verwendung von Standardtrainingsalgorithmen, wie der bereits vorgestellte Levenberg-Marquardt Algorithmus, möglich. Nachdem in der Simulator-Konfiguration nur die Anfangszustände bekannt oder abgeschätzt werden müssen, ist es im Gegensatz zur Prädiktor-Konfiguration nicht länger nötig, zu jedem Abtastzeitpunkt die unter Umständen nicht verfügbaren Systemzustände (rauschfrei) zu messen. Durch die Herleitung einer Lernregel für RKNN in Simulator-Konfiguration wird die breite Anwendbarkeit der RKNN in der Praxis erst möglich und die Güte der Modelle bei verrauschten Trainingsdatensätzen und Auswertung in Simulator-Konfiguration entscheidend verbessert. Zudem werden erstmals Nebenbedinungen an die Netzgewichte hergeleitet, welche die Stabilität des RKNN sicherstellen.

5.2.1 Problemformulierung

Das Ziel ist es, ein System in Zustandsraumdarstellung (2.1) zu approximieren. Im Gegensatz zu [88] wird die Ausgangsgleichung $h(\cdot)$ auch berücksichtigt, da nur die Ausgänge als nicht zwangsläufig rauschfrei messbar angenommen werden. In vielen Fällen wird $h(\cdot)$ eine Auswahlfunktion darstellen, welche die messbaren Zustände angibt.

Das Ziel ist nun die Approximation des Ein-/Ausgangsverhaltens des Systems (2.1), welches durch die unbekannten Funktionen $f_s(\cdot)$ und $h_s(\cdot)$ beschrieben wird. Dabei sind der Ausgang $y(t)$ und Eingang $u(t)$ nur zu bestimmten Abtastzeitpunkten bekannt, wobei die Abtastrate $t_s = t_s(t)$ nicht konstant sein muss. Für die weitere Betrachtung werden folgende Annahmen getroffen [88]:

1. Die Funktionen $f_s(\cdot)$ und $h_s(\cdot)$ sind unabhängig von der Zeit.
2. Die Funktionen $f_s(\cdot)$ and $h_s(\cdot)$ sind Lipschitz-stetig.
3. Das System (2.1) ist stabil.
4. Die Zustände x sind beobachtbar und abhängig von der Konfiguration (Prädiktor/Simulator) auch messbar.

Der Begriff der Stabilität wird in Abschnitt 5.2.5 eingeführt.

Eine Darstellung des Aufbaus eines RKNN in Simulator-Konfiguration ist in Abbildung 5.2 dargestellt. Die unbekannten Funktionen $f_s(\cdot)$ und $h_s(\cdot)$ sollen jeweils durch ein Neuronales Netz (MLP) approximiert werden und die Integration von $\hat{x}$ soll durch numerische Integration mittels eines Runge-Kutta Verfahrens erfolgen. Beides wird in den nächsten Abschnitten näher beschrieben.

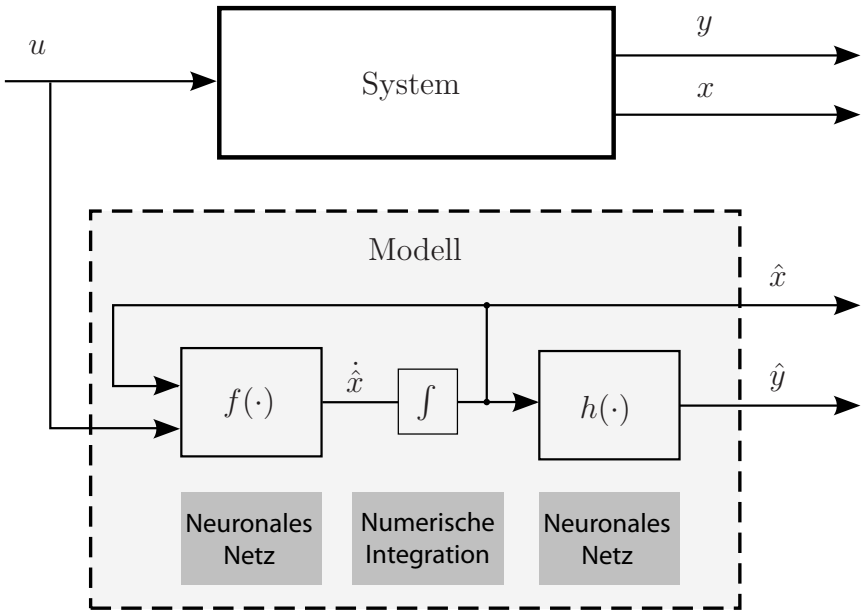


Abbildung 5.2: Aufbau des RKNN.

5.2.2 Struktur der Runge-Kutta Neuronalen Netze

Um die rechte Seite der Differentialgleichung (2.1) zu lösen, wird in [88] das Runge-Kutta Verfahren vorgeschlagen. Die Ein-/Ausgangsabbildung eines s-stufigen Runge-Kutta Verfahrens wird beschrieben durch [9]

$$x((i+1)h_{rk}) = x(ih_{rk}) + h_{rk}\sum_{j=1}^{s} b_j k_j(ih_{rk}) = x(ih_{rk}) + \psi(x(ih_{rk}), u(ih_{rk})) \tag{5.1a}$$

$$k_j(ih_{rk}) = f_s\left(x(ih_{rk}) + h_{rk}\sum_{l=1}^{s-1} a_{jl}k_l(ih_{rk}), u(ih_{rk} + c_l h_{rk})\right). \tag{5.1b}$$

Dabei hängt die Wahl der Koeffizienten a und b vom Runge-Kutta Verfahren ab [9]. Ein s-stufiges Runge-Kutta Verfahren ist dabei eindeutig durch den zugehörigen Butcher Array beschrieben

$$\begin{array}{c|c} c & A \\ \hline & b^\top \end{array},$$

wobei $\sum_{j=1}^{s} b_j = 1$, $c_j = \sum_{l=1}^{s} a_{jl}$ gilt und a_{jl} die Einträge der Matrix A bezeichnen. Hier wird anstelle des ansonsten üblichen zeitdiskreten Indek k der Index i verwendet, um eine Verwechslung mit den Runge-Kutta Schritten zu vermeiden. Dabei ist bei einem expliziten Runge-Kutta Verfahren A eine untere Dreiecksmatrix. Wird nun der Eingang u über das Abtastintervall als konstant angenommen, werden die Runge-Kutta Schritte eines expliziten Verfahrens beschrieben durch

$$\begin{aligned}
k_1(ih_{rk}) &= f_s\left(x(ih_{rk}), u(ih_{rk})\right) \\
k_2(ih_{rk}) &= f_s\left(x(ih_{rk}) + h_{rk}a_{21}k_1(ih_{rk}), u(ih_{rk})\right) \\
k_3(ih_{rk}) &= f_s\left(x(ih_{rk}) + h_{rk}\left[a_{31}k_1(ih_{rk}) + a_{32}k_2(ih_{rk})\right], u(ih_{rk})\right) \\
&\vdots = \vdots \\
k_s(ih_{rk}) &= f_s\left(x(ih_{rk}) + h_{rk}\sum_{j=1}^{s-1} a_{si}k_j(ih_{rk}), u(ih_{rk})\right)
\end{aligned}$$

und lassen sich somit rekursiv berechnen. Der Zustandsvektor $x((i+1)h_{rk})$ zum Zeitpunkt $t = (i+1)h_{rk}$ ist bestimmt durch die Eingänge $u(ih_{rk})$ und Zustände $x(ih_{rk})$ zum Zeitpunkt $t = ih_{rk}$ plus dem Produkt der Intervallänge h_{rk} und der gewichteten Summe der Steigungen k_i. Um die Notation einfach zu halten, wird die Intervallänge h_{rk} als konstant angenommen. Ist dies nicht der Fall, muss (5.1a) ersetzt werden durch

$$\begin{aligned}
x\left(\sum_{l=1}^{i+1} h_{rk,l}\right) &= x\left(\sum_{l=1}^{i} h_{rk,l}\right) + h_{rk,i+1}\sum_{j=1}^{s} b_j k_j\left(\sum_{l=1}^{i} h_{rk,l}\right) \\
&= x\left(\sum_{l=1}^{i} h_{rk,l}\right) + \psi\left(x\left(\sum_{l=1}^{l} h_{rk,l}\right), u\left(\sum_{l=1}^{i} h_{rk,l}\right)\right)
\end{aligned}$$

und die Runge-Kutta Schritte zu jedem Zeitpunkt entsprechend angepasst werden.

Nachdem $f_s(\cdot)$ und $h_s(\cdot)$ unbekannt sind, werden sie mit zwei parametrischen Modellen $f(x, u, \theta_f)$ und $h(x, u, \theta_h)$ approximiert. Die Intervallänge wird sinnvollerweise gleich der Abtastzeit $h_{rk} = t_s$ gesetzt, womit im Folgenden wieder die Notation $u(kt_s) = u_k$, $k \in \mathbb{N}$ verwendet wird. Es wird angenommen, das die Parametervektoren θ_f und θ_h von fester und bekannter Dimension $\theta_f \in \mathbb{R}^{N_{pf}}$ und $\theta_h \in \mathbb{R}^{N_{ph}}$ sind. Die Gesamtzahl der Parameter ist gegeben durch $N_p = N_{pf} + N_{ph}$ und hängt und der gewählten Modellstruktur ab.

Modellstruktur

Die Modellstruktur wird vom gewählten Funktionsapproximator vorgegeben. Grundsätzlich kann jedes beliebige Black-Box Modell verwendet werden, welches über ein Gradientenabstiegsverfahren trainiert werden kann. In [88] wird das RBF-Netzwerk vorgeschlagen, dessen Struktur und Parameter von den Trainingsdaten abhängen. Wie in Kapitel 4 dargestellt, ist für die Aufgabenstellung das MLP-Netz mit seiner definierten Anzahl an Parametern als Funktionsapproximator vorzuziehen und wird dementsprechend verwendet, um die Funktionen $f_s(\cdot)$ und $h_s(\cdot)$ zu approximieren.

Das verwendete MLP Netzwerk (4.1) mit linearem Term wird durch folgende Gleichung

$$\dot{\hat{x}} = f_{MLP}(\varphi_k, \theta) = W^l \varphi_k + W^o \sigma\left(W^h \varphi_k\right), \tag{5.2}$$

beschrieben, wobei $f_{MLP}(\varphi, \theta) : \mathbb{R}^{N_R} \to \mathbb{R}^{N_x}$ und $\varphi(\cdot) \in \mathbb{R}^{N_R}$ der Eingang des Neuronalen Netzes ist. Die Matrizen $W^l \in \mathbb{R}^{N_x \times N_R}$, $W^o \in \mathbb{R}^{N_x \times N_N}$, $W^h \in \mathbb{R}^{N_N \times N_R}$ sind Gewichtungsmatrizen, die während des Trainings angepasst werden. Um das Neuronale Netz in die parametrische Form von (5.3) und (5.4) zu bringen, werden die Zeilen der Gewichtungsmatrizen im Parametervektor

$$\theta = [W^l_1, \ldots, W^l_{N_x}, W^o_1, \ldots W^o_{N_x}, W^h_1, \ldots, W^h_{N_N}]^\top$$

zusammengefasst. Die Komponenten des Regressorvektors φ sind abhängig von der Konfiguration und in Prädiktor-Konfiguration definiert als

$$\varphi_k = \left[x_k^\top,\ u_k^\top,\ 1\right]^\top$$

und in Simulator-Konfiguration als

$$\varphi_k = \left[\hat{x}_k^\top,\ u_k^\top,\ 1\right]^\top.$$

Mit der gewählten Modellstruktur 5.2 können einfach lineare Systeme approximiert werden, indem man $W^o = 0$ und $W^l = [A\, B\, 0]$ setzt und somit

$$\dot{\hat{x}} = A\hat{x} + Bu$$

erhält. Zudem deckt die Runge-Kutta Methode, kombiniert mit dem Neuronalen Netz (5.2), eine Vielzahl an zeitdiskreten Modellansätzen ab. Wird ein Ein-Schritt Runge-Kutta Verfahren (Euler-Verfahren mit Koeffizienten $a_1 = b_1 = 1$) verwendet und $W^l = 0$, $h_{rk} = 1$ gesetzt, erhält man ein zeitdiskretes Zustandsraum Neuronales Netz [59]

$$\hat{x}_{k+1} = \hat{x}_k + W^o \sigma\left(W^h \varphi_k\right).$$

Das Euler-Verfahren mit $W^l = -\frac{1}{h_{rk}}[I,\ 0]$ stellt ein zeitdiskretes vollrekurrentes Neuronales Netz dar [89]

$$\hat{x}_{k+1} = W^o \sigma \left(W^h \varphi_k \right).$$

Wird das Euler-Verfahren verwendet und die Gewichtungen wie in (4.9) gesetzt, erhält man ein Neuronales Netz, welches ein Ein-/Ausgangsmodell in NARX- oder NOE-Konfiguration beschreibt. Da das RKNN verschiedene Ansätze für Neuronale Netze als Sonderfall abdeckt, kann erwartet werden, dass die Approximationsgüte des RKNN den genannten Ansätze für Neuronalen Netzen zumindest ebenbürtig ist.

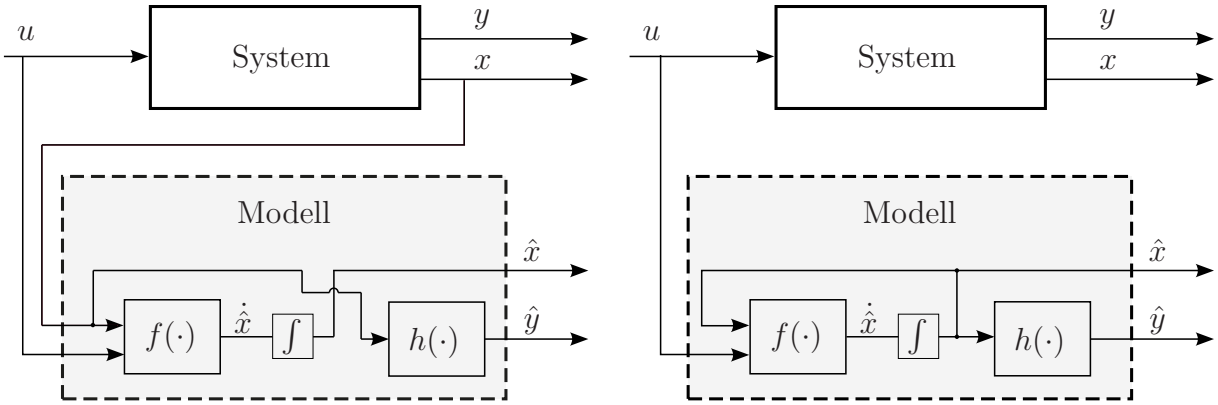


Abbildung 5.3: RKNN in Prädiktor-(links) und Simulator- (rechts) Konfiguration. Die numerische Integration wird durch ein Runge-Kutta Verfahren realisiert.

5.2.3 Berechnung der Jakobi-Matrix

Wie bereits im Kapitel 2 über die Grundlagen der Modellbildung erläutert, kann zwischen Prädiktor- und Simulator-Konfiguration unterschieden werden. Ein Modell in Prädiktor-Konfiguration wird beschrieben durch

$$\dot{\hat{x}}(t) = f(x(t), u(t), \theta_f) \tag{5.3a}$$
$$\hat{y}(t) = h(x(t), u(t), \theta_h), \tag{5.3b}$$

während ein Modell in Simulator-Konfiguration gegeben ist durch

$$\dot{\hat{x}}(t) = f(\hat{x}(t), u(t), \theta_f) \tag{5.4a}$$
$$\hat{y}(t) = h(\hat{x}(t), u(t), \theta_h). \tag{5.4b}$$

Sind die Anfangszustände x_0 bekannt oder abschätzbar, können in Simulator-Konfiguration alle folgenden Modellzustände und -ausgänge in Abhängigkeit vom Eingangssignal simuliert werden. Die beiden Konfigurationen sind in Abbildung 5.3 dargestellt. Dabei ist zu beachten, dass ein RKNN in beiden Konfigurationen innerhalb der Runge-Kutta Schritte in Simulator-Konfiguration verwendet wird, da die Systemzustände zwischen den Abtastzeitpunkten immer unbekannt sind.

Zunächst wird die Jakobi-Matrix für das Modell in Simulator-Konfiguration hergeleitet. Gesucht wird die totale Ableitung der Zustände $\hat{x}_k$ und Ausgänge $\hat{y}_k$ nach dem Parametervektor $\theta^\top = \left[\theta_f^\top, \theta_h^\top\right]$ unter Verwendung eines s-stufigen Runge-Kutta Verfahrens (5.1):

$$\frac{\mathrm{d}\hat{x}_{k+1}}{\mathrm{d}\theta} = \frac{\mathrm{d}\hat{x}_k}{\mathrm{d}\theta} + \frac{\mathrm{d}\psi(\hat{x}_k, u_k)}{\mathrm{d}\theta} = \frac{\mathrm{d}\hat{x}_k}{\mathrm{d}\theta} + t_s \sum_{i=1}^{s} b_i \frac{\mathrm{d}\hat{k}_i(k)}{\mathrm{d}\theta} \tag{5.5}$$

$$\frac{\mathrm{d}\hat{y}_k}{\mathrm{d}\theta} = \frac{\mathrm{d}h(\hat{x}_k, u_k, \theta_2)}{\mathrm{d}\theta}. \tag{5.6}$$

Um die Notation zu vereinfachen, werden folgende zu jedem Zeitpunkt k zu evaluierenden Vektoren definiert:

$$v_1(k) = \hat{x}_k \tag{5.7a}$$

$$v_2(k) = \hat{x}_k + h_{rk} a_{21} \hat{k}_1(k) \tag{5.7b}$$

$$v_3(k) = \hat{x}_k + h_{rk}\left[a_{31}\hat{k}_1(k) + a_{32}\hat{k}_2(k)\right] \tag{5.7c}$$

$$\vdots = \vdots$$

$$v_s(k) = \hat{x}_k + h_{rk} \sum_{i=1}^{s-1} a_{si} \hat{k}_i(k). \tag{5.7d}$$

Dabei wird die Abhängigkeit von $\hat{k}_j(k) = \hat{k}_j(k, \theta_f)$, $v_j(k) = v_j(k, \theta_f)$ und $\psi(\hat{x}_k, u_k) = \psi(\hat{x}_k, u_k, \theta_f)$ vom Parametervektor θ_f unterdrückt, um die Beschreibung kompakt zu halten. Mit den definierten Vektoren (5.7) können die Steigungen $\hat{k}_j$ beschrieben werden als:

$$\hat{k}_j(k) = f\left(v_j(k), u_k, \theta_f\right). \tag{5.8}$$

Die totale Ableitung von (5.8) ist daher gegeben durch

$$\frac{\mathrm{d}\hat{k}_j(k)}{\mathrm{d}\theta} = \frac{\partial f\left(v_j(k), u_k, \theta_f\right)}{\partial \theta} + \frac{\partial f\left(v_j(k), u_k, \theta_f\right)}{\partial v_j(k)} \frac{\mathrm{d}v_j(k)}{\mathrm{d}\theta} \tag{5.9}$$

mit

$$\frac{\mathrm{d}v_j(k)}{\mathrm{d}\theta} = \frac{\mathrm{d}\hat{x}_k}{\mathrm{d}\theta} + h_{rk} \sum_{l=1}^{j-1} a_{jl} \left(\frac{\mathrm{d}\hat{k}_l(k)}{\mathrm{d}\theta}\right) \quad , j > 1 \tag{5.10}$$

$$\frac{\mathrm{d}v_1(k)}{\mathrm{d}\theta} = \frac{\mathrm{d}\hat{x}_k}{\mathrm{d}\theta}. \tag{5.11}$$

Mit den Beziehungen (5.9), (5.10), (5.11) und gegebenem $\hat{x}_0$ respektive $\mathrm{d}\hat{x}_0/\mathrm{d}\theta = 0$ kann die totale Ableitung der Zustände (5.5) rekursiv berechnet werden. Die totale Ableitung der Ausgänge kann unter Verwendung von (5.5) und des folgenden Zusammenhanges

$$\frac{\mathrm{d}\hat{y}_k}{\mathrm{d}\theta} = \frac{\partial h\left(x_k, u_k, \theta_h\right)}{\partial \theta} + \frac{\partial h\left(\hat{x}_k, u_k, \theta_h\right)}{\partial \hat{x}_k} \frac{\mathrm{d}\hat{x}_k}{\mathrm{d}\theta}$$

ebenfalls rekursiv berechnet werden.

In Prädiktor-Konfiguration (wie in [88] verwendet) sind die Zustände x_k bekannt und das Modell wird als Ein-Schritt-Prädiktor verwendet. Somit ist die Ableitung gegeben durch

$$\begin{aligned}\frac{\mathrm{d}\hat{x}_{k+1}}{\mathrm{d}\theta} &= \frac{\mathrm{d}x_k}{\mathrm{d}\theta} + \frac{\mathrm{d}\psi(x_k, u_k, \theta_f)}{\mathrm{d}\theta} = t_s \sum_{i=1}^{s} b_i \frac{\mathrm{d}k_i(k)}{\mathrm{d}\theta} \\ \frac{\mathrm{d}\hat{y}_k}{\mathrm{d}\theta} &= \frac{\partial h(x_k, u_k, \theta_h)}{\partial \theta}.\end{aligned} \tag{5.12}$$

Im Gegensatz zur Simulator-Konfiguration ist $\mathrm{d}\hat{x}_k/\mathrm{d}\theta = 0$ und $\mathrm{d}v_1(k)/\mathrm{d}\theta = 0$. Dabei ist zu beachten, dass die letzten Einträge in (5.5) und in Prädiktor-Konfiguration auch die ersten Einträge von (5.6) gleich Null sind. Die Ableitungen von $f(\cdot)$ und $h(\cdot)$ nach den Gewichten und Zuständen hängen ihrerseits von der gewählten Struktur des Funktionsapproximators ab und sind für das MLP 5.2 gegeben durch

$$\begin{aligned}\frac{\partial f_{MLP}(\varphi_k, \theta)_j}{\partial w^l_{m,l}} &= (\varphi_k)_l \\ \frac{\partial f_{MLP}(\varphi_k, \theta)_j}{\partial w^o_{m,l}} &= \begin{cases} \sigma\left(W^h \varphi_k\right)_l & \text{wenn } m = j \\ 0 & \text{sonst} \end{cases} \\ \frac{\partial f_{MLP}(\varphi_k, \theta)_j}{\partial w^h_{m,l}} &= w^o_{j,m}\left(1 - \sigma\left(W^h \varphi_k\right)^2_m\right)(\varphi_k)_l \\ \frac{\partial f_{MLP}(\varphi_k, \theta)_j}{\partial \hat{x}_{k,l}} &= w^l_{j,l} + \sum_{n=1}^{N_N} w^o_{j,n}\left(1 - \sigma\left(W^h \varphi_k\right)^2_n\right) w^h_{n,l}.\end{aligned}$$

Nachdem zwei Neuronale Netze zur Approximation der Zustands- und Ausgangsgleichung verwendet werden, werden diese durch Indizierung des Parametervektors θ unterschieden. Die Vektoren θ_f und θ_h sind entsprechend die Gewichte der Zustands- und Ausgangsgleichung, $f(\cdot) = f_{MLP}(\varphi_\cdot, \theta_f)$ und $h(\cdot) = f_{MLP}(\varphi_\cdot, \theta_h)$.

5.2.4 Gradientenberechnung der Kostenfunktion

Für die vom Anfangszustand x_0 abhängigen Trainingstrajektorien $\{x(t_k, x_0)|k = 1, \ldots, K\}$ und $\{y(t_k, x_0)|k = 1, \ldots, K\}$ des Systems (2.1) werden x_k und y_k zu z_k zusammengefasst, wobei $\mathcal{X}$ das Set an Indizes $i \in \mathcal{X}$ für messbare Zustände und $\mathcal{Y}$ das Set an Indizes $i \in \mathcal{Y}$ der per Definition messbaren Ausgänge angibt. Die Indizes sind so gewählt, dass

$$z_k = \begin{cases} x_k, & \text{wenn } i \in \mathcal{X} \\ y_k, & \text{wenn } i \in \mathcal{Y}. \end{cases} \tag{5.13}$$

Der Approximationsfehler ist

$$E(\theta) = \frac{1}{2} \sum_{k=1}^{K} \left(\sum_{i \in \mathcal{X} \cup \mathcal{Y}} (z_{k,i} - \hat{z}_{k,i})^2 \right),$$

wobei alle Ausgänge messbar sein müssen. In der Prädiktor-Konfiguration müssen alle Zustände messbar sein, während in Simulator-Konfiguration auch $\mathcal{X} = \emptyset$ möglich ist. Der Gradient der Kostenfunktion (5.14) ist für $\theta = [\theta_f^\top, \theta_h^\top]^\top$ somit

$$\nabla E(\theta) = -J(\theta)^\top e(\theta) \tag{5.14}$$

mit dem Fehlervektor $e(\theta) = [e_1^\top(\theta), \ldots, e_K^\top(\theta)]^\top$, $e_k(\theta) = z_k - \hat{z}_k$ und der Jakobi-Matrix

$$J(\theta) = \begin{bmatrix} \frac{\mathrm{d}z_1(\theta)}{\mathrm{d}\theta_1} & \frac{\mathrm{d}z_1(\theta)}{\mathrm{d}\theta_2} & \cdots & \frac{\mathrm{d}z_1(\theta)}{\mathrm{d}\theta_p} \\ \frac{\mathrm{d}z_2(\theta)}{\mathrm{d}\theta_1} & \frac{\mathrm{d}z_2(\theta)}{\mathrm{d}\theta_2} & \cdots & \frac{\mathrm{d}z_2(\theta)}{\mathrm{d}\theta_p} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathrm{d}z_K(\theta)}{\mathrm{d}\theta_1} & \frac{\mathrm{d}z_K(\theta)}{\mathrm{d}\theta_2} & \cdots & \frac{\mathrm{d}z_K(\theta)}{\mathrm{d}\theta_p} \end{bmatrix}. \tag{5.15}$$

Mit der Jakobi-Matrix (5.15) und dem Gradienten (5.14) kann der im Anhang B beschriebene LM-Algorithmus zum Training der RKNN in Prädiktor- und Simulator-Konfiguration verwendet werden.

5.2.5 Stabilität

Der in dieser Arbeit verwendete Begriff der Stabilität bezieht sich auf die Eingangs-/Zustandssstabilität (*engl.: ISS. Input to State Stability*) und Ein-/Ausgangsstabilität (*engl.: IOS. Input to Output Stability*) des zeitdiskreten RKNN (5.1). Diese Bedingungen stellen sicher, dass beschränkte Eingänge stets beschränkte Zustände beziehungsweise Ausgänge und konstante Eingänge eine asymptotische Annäherung des Systems zu einer stabile Ruhelage zur Folge haben (siehe Anhang D für eine genauere Definition).

In diesen Abschnitt werden Bedingungen an die Gewichte des zeitkontinuierlichen MLP (5.2) formuliert, welche ISS und IOS des RKNN sicherstellen. Um die Stabilität des zeitkontinuierlichen Neuronalen Netzes mit N Neuronen zu untersuchen, wird der Modellansatz 5.2 umformuliert und die Offsetterme durch b^h und b^o explizit gekennzeichnet, wodurch die folgende Notation von der bisherigen kompakten Notation abweicht:

$$\dot{x} = W^{l,x}x + W^o\sigma(W^{h,x}x + W^{h,u}u + b^h) + b^o, \tag{5.16}$$

wobei $x \in \mathbb{R}^n$, $\sigma : v \mapsto [\tanh(v_1), \ldots, \tanh(v_N)]^\top$ und $-1 \leq \tanh(x) = 1 - \frac{2}{1+e^{-2x}} \leq 1$. Zunächst wird die Existenz und Eindeutigkeit der Lösung der Differentialgleichung (5.16) diskutiert und anschließend verschiedene Nebenbedingungen an die Netzgewichte präsentiert, um die Stabilität des Netzes sicherzustellen. Um eine gute Lesbarkeit zu gewährleisten, werden im Folgenden nur die Ergebnisse präsentiert. Die umfangreiche Herleitung und genaue Definitionen finden sich in Anhang D.

Existenz und Eindeutigkeit

Existenz und Eindeutigkeit der Lösung von (5.16) ist gewährleistet, wenn die Lipschitzbedingung

$$||f_{NN}(x, \theta) - f_{NN}(y, \theta)|| \leq L\,||x - y||$$

erfüllt ist (Theorem 9, Anhang D). Mit der Abschätzung

$$\begin{aligned}
||f_{NN}(x,\theta) - f_{NN}(y,\theta)|| &= \big\|W^{l,x}x + W^{l,u}u + W^o\sigma(W^{h,x}x + W^{h,u}u + b^h) \\
&\quad -W^{l,x}y - W^{l,u}u - W^o\sigma(W^{h,x}y + W^{h,u}u + b^h)\big\| \\
&\leq \big|\big|W^{l,x}\big|\big|\,||x-y|| + \\
&\quad ||W^o||\,\big|\big|\sigma(W^{h,x}x + W^{h,u}u + b^h) - \sigma(W^{h,x}y + W^{h,u}u + b^h)\big|\big| \\
&\leq \big|\big|W^{l,x}\big|\big|\,||x-y|| + ||W^o||\,\big|\big|W^{h,x}x - W^{h,x}y\big|\big| \\
&\leq \big|\big|W^{l,x}\big|\big|\,||x-y|| + ||W^o||\,\big|\big|W^{h,x}\big|\big|\,||x-y|| = L\,||x-y||
\end{aligned}$$

ist für

$$L = \big|\big|W^{l,x}\big|\big| + ||W^o||\,\big|\big|W^{h,x}\big|\big| \tag{5.17}$$

die Lipschitzbedingung erfüllt und somit Existenz und Eindeutigkeit der Lösung der Differentialgleichung (5.16) gewährleistet. Dies bedeutet allerdings nicht, dass RKNN (5.16) für einen gegebenen Eingang u nur eine Ruhelage besitzt. Abhängig vom Startpunkt x_0 können sich durchaus verschiedene Ruhelagen für denselben Eingang u einstellen, welche dann eindeutig und somit reproduzierbar sind.

Stabilität von Runge-Kutta Neuronalen Netzen

In diesem Abschnitt wird das Theorem vorgestellt, welches die Bedingungen an die Netzgewichte formuliert, um ISS des RKNN (5.1) sicherzustellen. Die Herleitung dieses Satzes ist umfangreich und daher in Anhang D ausgelagert. Neben dem Beweis und den dazu nötigen Definitionen finden sich dort die Definitionen der ISS für zeitdiskrete Systeme (Definition 6) sowie der Bezug zur Stabilität nach Lyapunov.

Der Satz basiert auf der Annahme, dass das verwendete RK-Verfahren (k,l,m)-algebraisch stabil ist. Dabei sind die Koeffizienten k, l und m abhängig vom gewählten RK-Verfahren, wobei für implizite RK-Verfahren der Koeffizient $m = 0$ ist. Für ein (k,l,m)-algebraisches RK-Verfahren gilt bei geeigneter Wahl der Schrittweite h_{rk} und $u = const$, dass

$$\langle x, f(x,u)\rangle \leq -\beta(t)\langle x,x\rangle \Rightarrow \langle x_{k+1}, x_{k+1}\rangle \leq k\langle x_k, x_k\rangle,$$

wobei $\beta(t) \geq 0\ \forall t$, x_k beziehungsweise x_{k+1} Lösungen des RK-Verfahrens und der Faktor $0 < k < 1$ ein Koeffizient des RK-Verfahrens sind [13]. Anschaulich formuliert, bedeutet dies, dass der Integrationsfehler des RK-Verfahrens über die Zeit gegen Null konvergiert, wenn sich die Systemtrajektorie der Ruhelage (Ursprung) nähert. Mit diesem Satz lässt sich zwar direkt die Stabilität der Ruhelage des RKNN nach Lyapunov zeigen, allerdings nicht ISS, da der Einfluss eines sich ändernden Eingangs nicht berücksichtigt wird. Die Stabilität der Ruhelage ist zwar eine notwendige aber keine hinreichende Bedingung für ISS [81]. Für eine ausführliche Erläuterung sei an dieser Stelle auf Anhang D verwiesen.

Das folgende Theorem formuliert die Bedingungen an die Netzgewichte des zeitkontinuierlichen RKNN (5.16), um die Eingangs-/Zustandsstabilität des RKNN (D.30) sicherzustellen. Der Beweis und eine ausführlichere Diskussion findet sich wiederum in Anhang (D.4.2).

Theorem 3 (ISS von RKNN)**.** *Das RKNN (D.30) ist für ein (k, l, m)-algebraisch stabiles RK-Verfahren mit $l < 0$ und $0 < k < 1$ ISS, wenn positiv definite Matrizen P, Λ, $Q = (P + P\Lambda P)$ und Gewichtungsvariablen $\sum \alpha_i = 1$, $\alpha_i \geq 0\ \forall i$ existieren, so dass entweder*

$$\begin{aligned}
P\Lambda P + PW^{l,x} + {W^{l,x}}^\top P + 2h_{rk}m\left({W^{l,x}}^\top Q W^{l,x} + C\right) <& 0 \\
\left(2h_{rk}m{W^{l,x}}^\top Q + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk}m{W^{l,x}}^\top Q + P\right) W^o W_i^{h,x}\right)^\top & \\
+\alpha_i P\Lambda P + \alpha_i PW^{l,x} + \alpha_i {W^{l,x}}^\top P + 2\alpha_i h_{rk}m\left({W^{l,x}}^\top Q W^{l,x} + C\right) \leq& 0 \quad \forall i
\end{aligned}$$

erfüllt sind, oder

$$\begin{aligned}
P\Lambda P + PW^{l,x} + {W^{l,x}}^\top P + 2h_{rk}m\left({W^{l,x}}^\top Q W^{l,x} + C\right) \leq& 0 \\
\left(2h_{rk}m{W^{l,x}}^\top Q + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk}m{W^{l,x}}^\top Q + P\right) W^o W_i^{h,x}\right)^\top & \\
+\alpha_i P\Lambda P + \alpha_i PW^{l,x} + \alpha_i {W^{l,x}}^\top P + 2\alpha_i h_{rk}m\left({W^{l,x}}^\top Q W^{l,x} + C\right) \leq& 0 \quad \forall i \\
\left(2h_{rk}m{W^{l,x}}^\top Q + P\right) W^o W_q^{h,x} + \left(\left(2h_{rk}m{W^{l,x}}^\top Q + P\right) W^o W_q^{h,x}\right)^\top & \\
+\alpha_q P\Lambda P + \alpha_q PW^{l,x} + \alpha_q {W^{l,x}}^\top P + 2\alpha_q h_{rk}m\left({W^{l,x}}^\top Q W^{l,x} + C\right) <& 0 \quad \exists q,
\end{aligned}$$

erfüllt sind. Darin sind

$$C = \left|\left|{W^o}^\top Q W^o\right|\right| \left|\left|W^{h,x}\right|\right|^2$$

und

$$W_i^{h,x} = \begin{bmatrix} & 0_{i-1\times N_x} & \\ W_{i,1}^{h,x} & \dots & W_{i,N_x}^{h,x} \\ & 0_{N_N-i\times N_x} & \end{bmatrix}.$$

Der Koeffizient m ist ein Parameter der RK-Verfahren der bei den klassischen expliziten RK-Verfahren $m = 1$ und bei den impliziten RK-Verfahren $m = 0$ ist [13]. Wie aus dem Theorem 3 ersichtlich, kann die Stabilität leichter sichergestellt werden, wenn die Schrittweite h_{rk} des Runge-Kutta Verfahrens klein gewählt wird, da dann die positiv definite Matrix ${W^{l,x}}^\top Q W^{l,x} + C$ weniger stark gewichtet wird. Dies gilt ebenso für implizite RK-Verfahren, da dann $m = 0$ ist. Allerdings kann für implizite RK-Verfahren nicht die Ableitung des Ausganges des RKNN nach den Gewichten berechnet werden, wodurch der Einsatz von Standardtrainingsalgorithmen unmöglich wird.

Anzumerken ist an dieser Stelle, dass ein ISS-RKNN stets global asymptotisch stabile Ruhelagen besitzt und gleichzeitig Ein-/Ausgangsstabil (IOS) ist. Eine ausführlichere Diskussion hierzu findet sich ebenfalls im Anhang (Abschnitt D.2.1 und D.2.2).

5.2.6 Wahl des expliziten Runge-Kutta Verfahrens

Der Vergleich von numerischen Verfahren zur Lösung von Differentialgleichungen basiert meist auf der Konsistenzordnung. Diese gibt an mit welcher Ordnung der Diskretisierungsfehler $\epsilon(x) = x((i+1)h_{rk}) - x(ih_{rk}) - \psi(x(ih_{rk}), u(ih_{rk}))$ in Abhängigkeit der Schrittweite

h_{rk} fällt. Dabei gilt

$$\epsilon(x) \leq K h_{rk}^p = \mathcal{O}(h_{rk}^p)$$

und p bezeichnet die Konsistenzordnung. Für explizite Runge-Kutta Verfahren mit s-Stufen gilt für $s \leq 4$, dass die Konsistenzordnung $p = s$ ist [34]. Daher werden meist Runge-Kutta Verfahren mit 4 Stufen verwendet, da hier die größtmögliche Konsistenzordnung im Verhältnis zum Aufwand der Auswertung erzielt werden kann. In dieser Arbeit wird entsprechend das klassische Runge-Kutta Verfahren 4. Ordnung mit dem Butcher-Array

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

verwendet [34]. Für diesen Butcher-Array ist das RK-Verfahren 4. Ordnung $(k(l), l, m)$-algebraisch stabil, mit $m = 1$, $\alpha \leq l < 0$ und $k(l) = 1 + 2l + 2l^2 + \frac{4}{3}l^3 + \frac{2}{3}l^4 < 0$, wobei $\alpha = -0.64779...$ die reale Wurzel von $4 + 4x + 2x^2 + x^3$ ist [13].

Die Konsistenzordnung zeigt auch deutlich den Unterschied zwischen der Euler-Methode (RK-Verfahren 1. Ordnung) und RK-Verfahren höherer Ordnung auf, da der Diskretisierungsfehler exponentiell mit der Anzahl der Stufen des RK-Verfahren fällt. Auf der Konsistenzordnung basiert auch der Beweis in [88], der zeigt, dass RKNN zeitdiskreten Neuronalen Netzen hinsichtlich Approximationsgüte und Generalisierungsfähigkeit überlegen sind.

5.2.7 Initialisierung

Um das Training mit den zuvor formulierten Bedingungen an die Gewichte durchführen zu können, müssen die Gewichte so initialisiert werden, dass die Stabilitätskriterien erfüllt sind. Zudem sollte für eine schnelle Konvergenz des Levenberg-Marquardt Algorithmus die Initialisierung nahe (lokaler) Optima erfolgen. Um eine gute Initialisierung zu erreichen, kann das RKNN in die Modellstruktur (4.9) überführt und in NARX-Konfiguration betrachtet werden. Dies erlaubt den Einsatz der in [21] vorgestellten linearen Optimierungsverfahren, um mit möglichst geringem Trainingsaufwand möglichst gute Anfangsgewichte zu finden. Diese Methode der Initialisierung wird in dieser Arbeit auch eingesetzt, wodurch das Training erheblich beschleunigt werden kann.

5.2.8 Einschränkungen der Runge-Kutta Neuronalen Netze

Die RKNN sind zeitkontinuierliche Neuronale Netze, welche über das Runge-Kutta Verfahren diskretisiert werden. Dementsprechend sind die RKNN den Restriktionen der numerischen Lösung von Differentialgleichungen unterworfen. Die maximale Schrittweite des Runge-Kutta Verfahrens muss so gewählt werden, dass eine Konvergenz der Lösung sichergestellt werden kann. Somit ist die minimale Abtastrate, für welche die RKNN ein-

gesetzt werden können, problemspezifisch und kann erst nach dem Training abgeschätzt werden. Um die Konvergenz der numerischen Integration sicherzustellen, muss während des Trainings die Abtastrate hoch gewählt werden, wobei mit steigender Abtastrate der Trainingsdatenumfang und somit die Trainingsdauer ansteigt. Entsprechend muss ein guter Kompromiss zwischen zu niedriger (keine Konvergenz der numerischen Integration) und zu hoher (lange Trainingszeiten) Abtastrate gefunden werden. Da die Trainingszeit von MLP-Netzen linear mit der Anzahl der Trainingsdaten steigt [14], wirkt sich die Wahl einer hohen Abtastrate bei niedrig dimensionalen Approximationsproblemen (einige hundert Parameter) in der Regel nicht kritisch auf die Rechenzeit aus und bewegt sich innerhalb weniger Sekunden.

5.2.9 Vorteile der Runge-Kutta Neuronalen Netze

Die Verwendung von RKNN anstatt zeitdiskreten Neuronalen Netzen bringt neben der höheren Modellgüte bei den folgenden Problemstellungen Vorteile mit sich:

Integration in Simulationsmodelle: Da RKNN für jede beliebige Abtastrate ausgewertet werden können, ist eine einfache Integration der RKNN in zeitkontinuierliche Simulationsmodelle möglich. Zudem kann das im Training verwendete Runge-Kutta Verfahren bei der Auswertung durch beliebige Integrationsverfahren ersetzt werden, wodurch die Vorteile der adaptiven Schrittweitenregelung von ODE-Solver bei der Simulation genutzt werden können.

Reglerentwurf: Die Abtastrate beim Training der RKNN kann sich von der Abtastrate in der Auswertung unterscheiden. Dies ist vor allem bei der Verwendung der Neuronalen Netze zum Reglerentwurf [32] relevant, da eine hohe Abtastrate entscheidend für die erzielte Regelgüte ist. Zeitdiskrete Neuronale Netze erlauben keine problemspezifische Anpassung der Abtastrate in der Auswertung und sind an die im Training gewählte Abtastrate gebunden. Zudem stehen durch die Verwendung der RKNN nicht nur die Modellinformation zu den Abtastzeitpunkten zur Verfügung. Somit kann das RKNN als datenabgetastetes Modell betrachtet werden, das heißt als zeitkontinuierliches Modell, welches die Signalabtastung berücksichtigt. Beim Reglerentwurf auf Basis eines datenabgetasteten Modells steht neben dem Systemzustand x zu den Abtastzeitpunkten t_k und t_{k+1} zusätzliches Wissen über das Systemverhalten zwischen den Abtastzeitpunkten zu Verfügung. Beim Reglerentwurf kann somit, ergänzend zum Reglerentwurf auf Basis eines zeitdiskreten Modells, die zeitkontinuierliche Natur des Systems mit berücksichtigt und somit auf das Verhalten des Systems zwischen den Abtastzeitpunkten eingegangen werden. Durch den zusätzlichen Informationsgehalt des Ersatzmodells kann das Regelverhalten gegenüber zeitdiskreten Ersatzmodellen verbessert werden [12].

Schwankende Abtastrate: Ein RKNN ist unabhängig von der gewählten Abtastrate gültig, sofern diese nicht zu groß wird. Kann eine gleich bleibende Abtastrate nicht sichergestellt werden, wie dies bei nicht echtzeitfähigen Messsystemen der Fall ist, kann die schwankende Abtastzeit beim Training der RKNN berücksichtigt werden. Dies stellt einen großen Vorteil gegenüber zeitdiskreten Neuronalen Netzen dar, welche ein zeitdiskretes System mit gleich bleibender Abtastrate modellieren. Die Auswirkungen einer schwankenden Abtastrate wird in Abschnitt 5.5 simulativ untersucht.

5.3 Modellkomitee

Wie im Kapitel 2 erläutert, besteht das Ziel eines Modellkomitees darin, die Varianz des Komiteeausgangs durch Kombination von m Modellen zu reduzieren. Der Ansatz eines Modellkomitees basiert auf der Annahme, dass die Modelle die Trainingsdaten in der Regel sehr gut beschreiben können und somit einen geringen Biasfehler aufweisen. Die Abweichung der Modelle vom wahren Funktionswert, in den Bereichen wo keine Trainingsdaten vorliegen, stellen den Varianzfehler dar. Dieser ist abhängig vom gewählten Modelltyp beziehungsweise bei Modellen, die nichtlinear in ihren Parametern sind, von der zufälligen Initialisierung der Gewichte, die bei der nicht-konvexen Optimierung der Modellgewichte als Startwerte verwendet werden. Durch geschickte Kombination der Modelle lässt sich nun die zufällige Varianz der einzelnen Modelle „herausmitteln“ und somit der Varianzfehler bei gleich bleibenden Biasfehler reduzieren [6].

Die Ausgänge $\hat{y}_k^{(i)}$ der Einzelmodelle sollen über die Parameter λ_i so gewichtet werden, dass der Komiteeausgang

$$\hat{y}_k = \sum_{i=1}^{m} \lambda_i \hat{y}_k^{(i)} \qquad \text{mit } 0 \leq \lambda_i \leq 1 \text{ und } \sum \lambda_i = 1$$

einen geringeren Varianzfehler (bei gleich bleibenden Biasfehler) als die Einzelmodelle aufweist. Die einfachste Methode der Komiteebildung besteht darin, die Modelle über das arithmetische Mittel zu gewichten

$$\hat{y}_k = \sum_{i=1}^{m} \frac{1}{m} \hat{y}_k^{(i)}.$$

Dieser Ansatz führt, wie in [5] theoretisch und in [47] anhand der Praxis gezeigt, zu besseren Ergebnissen als das beste Einzelmodell des Komitees.

Da die Modellgüte der Einzelmodelle Schwankungen unterworfen ist, sollten Modelle, für welche eine höhere Modellgüte erwartet werden kann, stärker gewichtet werden [63]. Zudem kann gezeigt werden, dass die Verwendung der besten Modelle zu besseren Ergebnissen führt als die Verwendung aller [95]. Um die besten Modelle zu selektieren, muss die erwartete Modellgüte abgeschätzt werden. Dies lässt sich unter anderem durch Bayes'sche Methoden erreichen [5]. Andere Ansätze bestehen darin, die Generalisierungsfähigkeit der Modelle oder den Informationsgehalt der Trainingsdaten für die Einzelmodelle abzuschätzen. Eine ausführliche Gegenüberstellung und simulative Untersuchung der einzelnen Methoden findet sich in [92]. Die in [92] erzielten Simulationsergebnisse deuten darauf hin, dass die Bayes'sche Methode [5] zur besten Generalisierungsfähigkeit des Komitees führt. Eine weitere Verbesserung lässt sich in der Praxis erzielen, wenn Modelle, deren Gewichtung einen definierten Schwellwert unterschreiten, aus dem Komitee entfernt werden [92].

Analog zur Ermittlung der Regularisierungsparameter über das Bayes'sche Theorem (4.7),

lässt sich berechnen, wie wahrscheinlich ein Modell für einen gegebenen Datensatz ist

$$p\left(f_{MLP}(\theta^{(i)})|\mathcal{D}\right) = \frac{p\left(\mathcal{D}|f_{MLP}(\theta^{(i)})\right) p\left(f_{MLP}(\theta^{(i)})\right)}{p(\mathcal{D})}$$

und somit die Gewichtung der Einzelmodelle im Komitee ermitteln

$$f_{com}(\varphi_k) = \sum_{i=1}^{m} p\left(f_{MLP}(\theta^{(i)})|\mathcal{D}\right) f_{MLP}(\varphi_k, \theta^{(i)}).$$

Dabei gilt

$$\begin{aligned} p\left(f_{MLP}(\theta^{(i)})|\mathcal{D}\right) \sim & -\alpha\frac{1}{2}\|\theta^{(i)}\|_2^2 - \beta E(\theta^{(i)}) - \frac{1}{2}\ln\det\left(H^{(i)}\right) + \frac{N_p}{2}\ln\alpha^{(i)} + \frac{K}{2}\ln\beta^{(i)} \\ & + \ln N_N! + 2\ln N_N + \frac{1}{2}\ln\left(\frac{2}{\gamma^{(i)}}\right) + \frac{1}{2}\ln\left(\frac{2}{K-\gamma^{(i)}}\right). \end{aligned}$$

Modelle, welche einen festzulegenden Schwellwert für die Modellwahrscheinlichkeit unterschreiten, werden üblicherweise aus dem Modellkomitee entfernt. Damit die resultierende Gewichtung der Modelle sich auf Eins summiert, muss die Wahrscheinlichkeit der Einzelmodelle entsprechend normiert werden. Für die Herleitung und eine weiterführende Diskussion sei an dieser Stelle an [5] und [92] verwiesen.

5.4 Versuchsplanung

Die in Kapitel 3 diskutierten Versuchspläne sind alle aus der Welt der statischen Modelle entliehen. Sie dienen zur Ermittlung einer Verteilung der Designpunkte im Eingangsraum, wobei die Dynamik des Systems nicht berücksichtigt wird. In diesem Abschnitt wird zunächst ein parametrisches Anregungssignal vorgestellt, welches im industriellen Einsatz für viele Identifikationsaufgaben besser geeignet ist als das klassische APRBS, und anschließend werden Online-Versuchspläne entwickelt, welche die Dynamik des Systems berücksichtigen. Die Diskussion der Versuchsplanung erfolgt für beliebige Anregungssignale, da für bestimmte Strecken eine APRBS-förmige Anregung nicht geeignet ist. So muss zur Identifikation integrierender Strecken, wie zum Beispiel hydraulische Strecken, das parametrische Anregungssignal durch Impulse ersetzt werden. Allerdings zeigt sich, dass sich die Kriterien der Versuchsplanung für das am APRBS angelehnte parametrische Anregungssignal erheblich vereinfachen.

5.4.1 Parametrisches Anregungssignal

Problematisch beim APRBS im industriellen Einsatz sind die Sprünge im Eingangsraum. Zum einen sind die Aktuatoren nicht beliebig schnell und zum anderen führen bei vielen Systemen gewisse Eingangskombinationen zu unsicheren Betriebsbedinungen. Werden anstelle der Sprünge Rampen mit einer maximal zulässigen Steigung gewählt, lässt sich das Anregungssignal als eine Sequenz von N Rampenfunktionen beschreiben, welche durch die zugehörige Haltezeit $T_{h,s}$ und die zu verbindenden Designpunkte d_{s-1} und d_s im Eingangsraum bestimmt sind

$$\mathcal{U} = \bigcup_{s=1}^{N} \mathcal{U}_s \left(d_s, d_{s-1}, T_{h,s}\right).$$

Die einzelnen Rampenfunktionen $\mathcal{U}_s$ bestehen aus zeitdiskreten Eingangssequenzen, welche durch die Sequenzdauer T_s und die Abtastzeit t_s festgelegt sind [22]

$$\mathcal{U}_s \left(d_s, d_{s-1}, T_{h,s}\right) = \left\{u_{k+i}\right\}_{i=1}^{T_s/t_s}, \tag{5.20}$$

wobei $t_k = \frac{1}{t_s} \sum_{j=1}^{s-1} T_j$ und

$$u_{k+i} = \begin{cases} d_s, & \frac{T_{r,s}}{t_s} \leq i \leq \frac{T_s}{t_s} \\ gi + d_{s-1}, & 1 \leq i < \frac{T_{r,s}}{t_s} \end{cases}. \tag{5.21}$$

Die Sequenzdauer muss dabei ein ganzzahliges Vielfaches der Abtastzeit sein. Die Steigung g und die Rampenzeit $T_{r,s}$ werden durch die maximal zulässige Steigung $g \leq g_{\max}$ bestimmt und berechnen sich wie folgt:

$$g = \frac{1}{T_{r,s}} \Delta d \qquad T_{r,s} = \left\lceil \frac{1}{t_s} \max_j \left(\left| \frac{\Delta d_j}{g_{\max,j}} \right| \right) \right\rceil t_s \qquad \Delta d = d_s - d_{s-1}. \tag{5.22}$$

Die Rampenzeit $T_{r,s}$ und die Haltezeit $T_{h,s}$ sind dabei ein Vielfaches von der Abtastzeit t_s. Somit hängt die Länge einer Rampenfunktion (5.21) von der Designpunkt abhängigen

Rampenzeit und der Haltezeit ab: $T_s = T_{h,s} + T_{r,s}$. In Abbildung 5.4 ist eine schematische Darstellung einer Rampenfunktion dargestellt.

Dabei bestimmt die Steilheit der Flanke der Rampenfunktion (5.21) die Frequenzcharakteristik des Signals. Bei steilen Flanken, das heißt bei kleiner Rampenzeit $T_{r,s}$, werden insbesondere die hohen Frequenzen stärker angeregt [40]. Die relative Abweichung des Amplitudendichtespektrums des Anregungssignals mit Rampenfunktion gegenüber einem Anregungssignal mit Sprungfunktion lässt sich mit $\leq 5\%$ beziehungsweise $\leq 1\%$ abschätzen, wenn folgende Bedingung erfüllt ist [40]:

$$T_{r,s} \leq \frac{1.1}{2\pi} T_{\min} \quad \text{beziehungsweise} \quad T_{r,s} \leq \frac{0.5}{2\pi} T_{\min} \qquad \forall s.$$

Dabei bezeichnet $T_{\min}$ die kleinste für die Identifikationsaufgabe relevante Zeitkonstante des Prozesses. Da die minimale Haltezeit des APRBS üblicherweise hinsichtlich der zu identifizierenden Prozessdynamik gewählt wird, kann in der Praxis die Rampenzeit so gewählt werden, dass

$$T_{r,s} < \frac{1.1}{2\pi} T_{h,\min} \quad \text{beziehungsweise} \quad T_{r,s} < \frac{0.5}{2\pi} T_{h,\min} \qquad \forall s$$

zu jeder Zeit erfüllt ist. Wird $T_r = 0$ gesetzt, erhält man wieder das klassische APRBS.

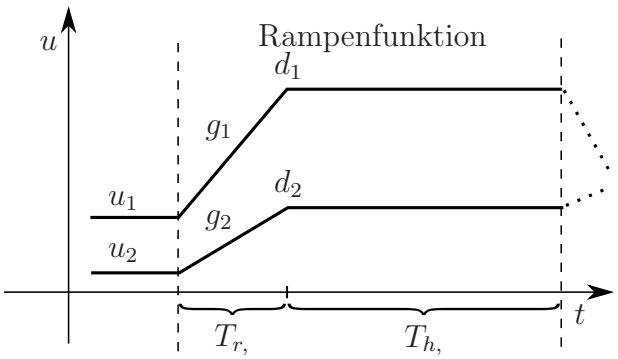


Abbildung 5.4: Zeitlicher Verlauf des Eingangs u: Designpunkt d mit zugehöriger Steigung g, Rampen- $T_{r,}$ und Haltezeit $T_{h,}$.

Unabhängig von der Wahl der Rampen- und Haltedauer stellt sich die Frage, wie die Amplituden zu wählen sind. In [60] wird vorgeschlagen, das APRBS wie folgt zu erzeugen. Zunächst wird eine PRBS berechnet. Dann wird die Anzahl der Sprünge gezählt und jede Dimension des Eingangsraums in die entsprechende Anzahl von Intervallen unterteilt. Schlussendlich wird jedem Sprung des PRBS für jede Dimension zufällig ein Wert aus diesen Intervallen zugeordnet. Wie in [60] hervorgehoben wird, ist der Eingangsraum in der Regel gut abgedeckt, wobei allerdings aufgrund der zufälligen Wertezuweisung einige „Löcher“ existieren können. Eine gute Abdeckung des Eingangsraums ist daher nur bei einer Vielzahl von Sprüngen und somit entsprechender Anregungsdauer zu erwarten. Unterschiedliche Methoden zur Verteilung von Designpunkten im Eingangsraum werden im Kapitel 3 diskutiert. Neben der Verteilung der ist auch die Reihenfolge von entscheidender Bedeutung. Wie auch bei der Versuchsplanung für statische Systeme, wird im Folgenden zwischen Offline- und Online-Versuchsplanung unterschieden.

5.4.2 Offline-Versuchsplanung für dynamische Systeme

Bei der Versuchsplanung für dynamische Systeme ist die Wahl der Reihenfolge der Designpunkte von entscheidender Bedeutung. Bei den bisher diskutierten Offline-Versuchsplänen geben sowohl die LHC- als auch D-optimale Verteilung keinen Aufschluss über die optimale Reihenfolge der Designpunkte. Wenn kein physikalisches Vorwissen in die Versuchsplanung eingebracht werden kann, werden daher in [74] folgende Kriterien zur Reihung vorgeschlagen:

- Sortierung nach minimalem Abstand,
- Sortierung nach maximalem Abstand,
- Sortierung nach mittlerem Abstand.

Diese heuristischen Kriterien können die Dynamik des zu identifizierenden Systems nicht berücksichtigen. Diese Problematik lässt sich anhand des Beispiels 4 leicht veranschaulichen.

Beispiel 4: Für das folgende lineare System sollen 2 D-optimale Eingänge bestimmt werden

$$x_{k+1} = \theta(u_k - x_k) \qquad y_k = x_k + \eta_k, \tag{5.23}$$

wobei $\eta \sim \mathcal{N}(0, 0.1)$ und die Eingänge auf $u \in [0, 10]$ beschränkt sind. Werden nun 2 D-optimale Eingänge u_1 und u_2 ohne Berücksichtigung der Systemdynamik über den Ansatz eines affinen Modells $x = [u^\top, 1]\theta$ bestimmt, ist das D-Optimalitätskriterium $\det(\mathcal{I}(\theta, u)) = u_1^2 + u_2^2 - 2u_1u_2$ und somit die D-optimalen Eingänge $\mathcal{U} = \{0, 10\}$ beziehungsweise $\mathcal{U} = \{10, 0\}$. Das D-optimale Kriterium für (5.23) berechnet sich mit $x_0 = 3$ und $\theta = 0.5$ allerdings wie folgt:

$$\begin{aligned} J(\mathcal{U}, \theta) &= \det\left(\mathcal{I}_1(\theta, u_1) + \mathcal{I}_2(\theta, u_2)\right) \\ &= (u_1 - 3)^2 + (u_2 - 0.5\,(u_1 - 3))^2\,. \end{aligned}$$

Wählt man die Eingangssequenz $\mathcal{U}_1 = \{10, 0\}$ ergibt sich $J(\mathcal{U}_1, \theta) = 61.25$. Werden die Eingänge einzeln optimiert, ergibt sich $u_1 = 10 = \max_{u_1}(u_1 - 3)^2$ und $u_2 = 10 = \max_{u_2} 49 + (u_2 - 3.5)^2$. Damit ist $J(\mathcal{U}_2, \theta) = 91.25$. Allerdings führt die einzelne Optimierung der Eingänge nicht zur optimalen Eingangssequenz $\mathcal{U}_3 = \{0, 10\}$, für welche sich $J(\mathcal{U}_3, \theta) = 141.25$ ergibt. Um somit den optimalen Informationsgehalt zu erreichen, muss die Systemdynamik (in Form verzögerter Modellausgänge) berücksichtigt werden und es ist zudem nicht ausreichend nur für den nächsten Schritt den optimalen Eingang zu berechnen. Die Varianz der Parameterschätzung $\hat{\theta}$ bei 10^4 Parameterschätzungen mit verschiedenen Rauschrealisierungen ist entsprechend unterschiedlich: $\mathrm{var}(\hat{\theta}(\mathcal{U}_1)) = 0.0016$, $\mathrm{var}(\hat{\theta}(\mathcal{U}_2)) = 0.0012$ und $\mathrm{var}(\hat{\theta}(\mathcal{U}_3)) = 0.0007$.

Aus diesem einfachen Beispiel wird ersichtlich, dass auch der Einfluss der Eingänge auf die zukünftige Bildung der Fisher-Informationsmatrix berücksichtigt werden muss. Dies lässt sich durch den Einsatz von Online-Verfahren erreichen, welche das Modellwissen in der Versuchsplanung mit berücksichtigen.

5.4.3 Online-Versuchspläne für dynamische Systeme

Im Gegensatz zu den zuvor diskutierten Offline-Verfahren berücksichtigen die im folgenden eingeführten Online-Verfahren den aus den Messungen resultierenden Informationsgewinn, um den weiteren Versuchsablauf zu bestimmen. Durch die Verwendung von adaptiven Versuchsplänen ist es möglich, Modelle an bestimmten Bereichen im Eingangsraum zu verfeinern, die sich im Laufe der Modellbildung für die Aufgabenstellung als relevant erweisen.

Da, wie anhand des Beispiels 4 veranschaulicht, die Wahl der Reihenfolge der Designpunkte von entscheidender Bedeutung ist, werden die Kriterien der Versuchspläne als modellprädiktive Regelung mit Nebenbedingungen definiert [53]. Die unter Berücksichtigung von Nebenbedingungen (N.B.) über den Zeithorizont T_s und somit über $N_s = T_s/t_s$ Abtastzeitpunkte zu optimierende Kostenfunktion ist

$$\begin{aligned} V(\mathcal{U}_s, k, T_s) &= \sum_{j=k+1}^{k+N_s} \sum_{i=1}^{m} \lambda_i \ell(\hat{x}_j^{(i)}, u_j) \\ \text{N.B. } \hat{x}_{j+1}^{(i)} &= f\left(\hat{x}_j^{(i)}, u_j, \theta_{f,k}^{(i)}\right) \quad u_j \in \mathbb{U} \\ \hat{y}_j^{(i)} &= h\left(\hat{x}_j^{(i)}, u_j, \theta_{h,k}^{(i)}\right) \quad \hat{y}_j \in \mathbb{Y} \\ \hat{y}_j &= \sum_{i=1}^{m} \lambda_i \hat{y}_j^{(i)} \quad \sum_{i=1}^{m} \lambda_i = 1 \end{aligned} \tag{5.24}$$

mit modellabhängigen Kosten $\ell(\hat{x}_j^{(i)}, u_j)$ und der Parameterschätzung $\theta_{f,k}^{(i)}$, $\theta_{h,k}^{(i)}$ zum Zeitpunkt k, wobei $\mathbb{U} \subseteq \mathbb{R}^{N_I}$ und $\mathbb{Y} \subseteq \mathbb{R}^{N_O}$ die zulässigen Ein- und Ausgangsmengen bezeichnen, welche durch Limitmodelle eingeschränkt werden können [48]. Dabei stellen Limitmodelle an den Eingang meist unsichere Betriebsbedingungen wie beispielsweise Klopfen und zu hohe Drücke dar, während Begrenzungen an den Ausgang bekannte Einschränkungen der zu modellierenden Zielgröße (zum Beispiel Temperaturgrenzen) berücksichtigen. Wird nun das parametrische Anregungssignal (5.21) verwendet, kann die Anzahl der zu optimierenden Parameter von $N_I N_s$ auf N_I reduziert werden

$$\begin{aligned} V(d_s, d_{s-1}, k, T_{h,s}) &= \frac{1}{N_s} \sum_{j=k+1}^{k+N_s} \sum_{i=1}^{m} \lambda_i \ell(\hat{x}_j^{(i)}, u_j) \\ \text{N.B. } \hat{x}_{j+1}^{(i)} &= f\left(\hat{x}_j^{(i)}, u_j, \theta_{f,k}^{(i)}\right) \quad u_j \in \mathbb{U} \\ \hat{y}_j^{(i)} &= h\left(\hat{x}_j^{(i)}, u_j, \theta_{h,k}^{(i)}\right) \quad \hat{y}_j \in \mathbb{Y} \\ \hat{y}_j &= \sum_{i=1}^{m} \lambda_i \hat{y}_j^{(i)} \quad \sum_{i=1}^{m} \lambda_i = 1 \\ u_{k+i} &= \begin{cases} d_s, & \frac{T_{r,s}}{t_s} \leq i < \frac{T_{h,s}+T_{r,s}}{t_s} \\ gi + d_{s-1}, & 1 \leq i \leq \frac{T_{r,s}}{t_s} \end{cases}, \end{aligned} \tag{5.25}$$

wobei $N_s = (T_{h,s} + T_{r,s})/t_s$ und sich g wie in (5.22) berechnet. Man beachte die Normierung des Optimierungskriteriums, da sich die Länge der Eingangssequenz durch die

Designpunkt abhängige Rampendauer ändern kann. Die Maximierung des Informationsgehalts pro Abtastzeitpunkt erlaubt zudem die Haltezeit $T_{h,s} \in [T_{h,\min}, T_{h,\max}]$ als weiteren Optimierungsparameter aufzufassen, anstatt die Haltezeit bereits von Beginn an festzulegen. Die gesuchte optimale Eingangssequenz respektive Designpunkt ist entsprechend

$$\mathcal{U}_s^* = \arg\max_{\mathcal{U}_s} V(\mathcal{U}_s, k, T_s) \quad d_s^* = \arg\max_{d_s} V(d_s, d_{s-1}, k, T_{h,s}).$$

Im Folgenden wird das Optimierungskriterium $\ell(\hat{x}_j^{(i)}, u_j)$ anhand des Query by Committee (QbC)-Kriterium und der Fisher-Information beschrieben,

Query by Committee

Beim QbC-Kriterium wird bei statischen Systemen nach einem Eingang gesucht der zur größten Abweichung der Modellausgänge führt [76]. Dementsprechend wird bei dynamischen Modellen nach einer Eingangssequenz gesucht, die innerhalb des betrachteten Zeithorizonts T_s die größte „Uneinigkeit“ der Modelle verursacht [22], [23]. Die Eingangssequenz, welche zur größten Modellunsicherheit führt, verspricht, wie in Absatz 3.4.1 diskutiert, den größten Informationsgewinn:

$$\ell_Q(\hat{x}_j^{(i)}, u_j) = \left|\left| h\left(\hat{x}_j^{(i)}, u_j, \theta_{h,k}^{(i)}\right) - \sum_{l=1}^{m} \lambda_l h\left(\hat{x}_j^{(l)}, u_j, \theta_{h,k}^{(i)}\right) \right|\right|_2^2. \tag{5.26}$$

Die optimale Eingangssequenz $\mathcal{U}_s^*$ beziehungsweise Designpunkt d_s^* führt zur größten Varianz der einzelnen Modellausgänge bezogen auf den Komiteeausgang. Die Ausgänge der Einzelmodelle können unterschiedlich gewichtet werden, indem die quadrierte Zwei-Norm durch eine gewichtete Summe der quadrierten Abweichung der Einzelmodellausgänge vom Modellkomiteeausgang ersetzt wird.

Sequentieller FIM-basierter Versuchsplan

Um eine hinsichtlich der Fisher-Information (siehe Abschnitt 3.4.2) optimale Eingangssequenz zu finden, muss die FIM der einzelnen Modelle über den betrachteten Zeithorizont maximiert werden

$$\begin{aligned}
\sum_{j=k+1}^{k+N_s} \ell_{FIM}(\hat{x}_j^{(i)}, u_j) =& g\left(\sum_{j=k+1}^{k+N_s} \mathcal{I}(\hat{x}_j^{(i)}, u_j, \theta_k^{(i)}) + \mathcal{I}_0^{(i)}(\theta_k^{(i)})\right) \\
\mathcal{I}(\hat{x}_j^{(i)}, u_j, \theta_k^{(i)}) =& \frac{\mathrm{d}h\left(\hat{x}_j^{(i)}, u_j, \theta_k^{(i)}\right)^\top}{\mathrm{d}\theta_k^{(i)}} \frac{\mathrm{d}h\left(\hat{x}_j^{(i)}, u_j, \theta_k^{(i)}\right)}{\mathrm{d}\theta_k^{(i)}} \\
\mathcal{I}_0^{(i)}(\theta_k^{(i)}) =& \sum_{j=1}^{k} \mathcal{I}(\hat{x}_j^{(i)}, u_j, \theta_k^{(i)}).
\end{aligned} \tag{5.27}$$

Dabei ist $\mathcal{I}_0^{(i)}(\theta_k^{(i)})$ die FIM, welche von den bisherigen Trainingsdaten und der aktuellen Parameterschätzung $\theta_k^{(i)}$ abhängt. Das bekannteste Kriterium $g(\cdot)$ zur Maximierung der FIM ist das D-optimale (siehe Abschnitt 3.3.2). Eine Diskussion des D-optimalen Kriteriums bei der Versuchsplanung für dynamische Modelle findet sich in [20] und [22].

5.4.4 Aufgabenbezogene Modifikationen

Die Beurteilung der Modellqualität hängt vom späteren Einsatzzweck der Modelle ab. Insofern ist es zweckmäßig den späteren Einsatzzweck der Modelle bereits bei der Versuchsplanung zu berücksichtigen.

Dynamische Modelle werden oft verwendet, um eine zeitaufwändige Stationärmessung zu vermeiden und stattdessen den Stationärwert zu prädizieren. Daher ist vor allem die Qualität des Stationärverhaltens entscheidend. Ein weiteres Einsatzgebiet ist die modellbasierte Optimierung, bei der entweder das Stationär- oder Instationärverhalten des Systems optimiert werden soll.

Verbesserung Stationärverhalten

Um das Stationärverhalten des Modellkomitees zu verbessern, kann die Modellunsicherheit im Stationärzustand

$$V_S(d_s) = \sum_{i=1}^{m} \lambda_i \left\| h(\hat{x}_s^{(i)}, d_s) - \sum_{l=1}^{m} \lambda_l h(\hat{x}_s^{(l)}, d_s) \right\|_2^2 \tag{5.28}$$

zum Kriterium (5.25) addiert und über $\mu \in [0, 1]$ entsprechend gewichtet werden [23]

$$d_s^* = \arg\max_{d_s} \left(\mu V(d_s, d_{s-1}, k, T_{h,s}) + (1 - \mu) V_S(d_s) \right).$$

Dabei steht $\hat{x}_s^{(i)}$ für den Stationärzustand des i-ten Modells im Designpunkt d_s. Sollen die Ausgänge der Einzelmodelle unterschiedlich gewichtet werden, muss die quadrierte Zwei-Norm entsprechend durch eine gewichtete Summe der quadrierten Abweichung der Einzelmodellausgänge vom Modellkomiteeausgang ersetzt werden.

Optimierung

Die gleichzeitige Optimierung des Modellausgangs und der Parameter stellt besondere Anforderungen an die Versuchsplanung. Zum einen muss das System möglichst stark angeregt werden, damit eine gute Parameterschätzung erfolgen kann. Zum anderen soll sich die Versuchsplanung auf Bereiche der vermuteten Optima konzentrieren, da nur hier eine hohe Modellgüte erforderlich ist. Da dies widersprüchliche Anforderungen an die Versuchsplanung sind, muss zunächst sicher gestellt werden, dass die Modellgüte ausreichend ist, um eine modellbasierte Systemoptimierung durchführen zu können. Daher muss das Optimierungskriterium dahingehend abgewandelt werden, dass sich im Laufe der Versuchsplanung der Schwerpunkt von einer (optimalen) Systemanregung zu einer Systemoptimierung verschiebt. Je nachdem, ob das Stationär- oder Instationäverhalten optimiert werden soll, ist das erste beziehungsweise zweite Kriterium besser geeignet

$$d_s^* = \arg\max_{d_s} \left(\mu V(d_s, d_{s-1}, k, T_{h,s}) + (1 - \mu) V_{opt}(d_s) \right) \tag{5.29a}$$

$$\mathcal{U}_s^* = \arg\max_{\mathcal{U}_s} \left(\mu V(\mathcal{U}_s, k, T_s) + (1 - \mu) V_{opt}(\mathcal{U}_s, k, T_s) \right). \tag{5.29b}$$

Dabei ist der Gewichtungsfaktor $\mu \in [0, 1]$ von entscheidender Bedeutung für die erzielten Optimierungsergebnisse und ist aktueller Gegenstand der Forschung. Weiterführende

Literatur findet sich unter dem Begriff *Self-Tuning Optimisation*; siehe [66] und die dort enthaltenen Referenzierungen.

Heuristische Kriterien zur Wahl von μ, die sich in der Praxis zur Optimierung statischer Systeme bewährt haben, finden sich in [65] und [47]. Diese berücksichtigen die Modellqualität in dem die Gewichtung der Kriterien abhängig von der laufend evaluierten Prädiktionsqualität der Modelle erfolgt.

Optimierung des Stationärverhaltens Bei der Optimierung des Stationärverhaltens, wird derjenige Designpunkt gesucht, welcher zum minimalen Modellausgang führt

$$V_{opt}(d_s) = -\sum_{j=1}^{N_O} \left(\sum_{i=1}^{m} \lambda_i h(\hat{x}_s^{(i)}, d_s, \theta_{h,k}^{(i)}) \right)_j , \tag{5.30}$$

wobei $\hat{x}_s^{(i)}$ wiederum der Stationärzustand des i-ten Modells im Designpunkt d_s ist. Sollen die Ausgänge des Modellkomitees unterschiedlich gewichtet werden, muss die Summe der Modellkomiteeausgänge entsprechend durch eine gewichtete Summe ersetzt werden.

Optimierung Instationärverhalten Bei der Optimierung des Instationärverhaltens wird die Eingangssequenz $\mathcal{U}_s$ gesucht, welche über den betrachteten Zeithorizont N_s zu den geringsten Kosten führt

$$V_{opt}(\mathcal{U}_s, k, T_s) = -\sum_{j=1}^{N_O} \left(\sum_{j=k}^{k+N_s} \sum_{i=1}^{m} \lambda_i h(\hat{x}_s^{(i)}, u_s, \theta_{h,k}^{(i)}) \right)_j , \tag{5.31}$$

Dabei werden meist zusätzliche Nebenbedingungen an den Eingangsvektor beziehungsweise an Teile des Eingangsvektors definiert, wie beispielsweise Optimierung der Ventilsteuerzeiten hinsichtlich Verbrauch bei vorgegebenem Fahrzyklus. Eine individuelle Gewichtung der Modellkomiteeausgänge, kann wiederum durch Verwendung einer gewichteten Summe erreicht werden.

5.5 Simulative Evaluierung: Versuchsplanung und Modellbildung

Im Folgenden wird die Versuchsplanung und Modellbildung anhand des im Anhang E vorgestellten Benchmarksystems evaluiert. Das Benchmarksystem besteht aus einem nichtlinearen statischen MISO (*engl. multi input single output*) System gefolgt von einer Dynamik erster Ordnung, welche durch ein PT1-Glied mit einer nichtlinear von den Eingangsgrößen abhängigen Zeitkonstante modelliert wird. Dieses Verhalten entspricht dem vieler Messgrößen am Verbrennungsmotorenprüfstand, die ebenfalls ein PT1-Verhalten aufweisen (zum Beispiel Abgastemperaturen), deren Zeitkonstante allerdings vom betriebspunktabhängigen Entalpiestrom im Abgas- oder Einlasssystem abhängen. Das nichtlineare statische System, welches zur Evaluierung der modellbasierten Online-Optimierung statischer Systeme in [65] und [48] verwendet wird, erlaubt eine einfache Überprüfung der Güte des Stationärverhaltens der Modelle.

Zunächst werden die Modellbildungsverfahren untersucht, da die modellbasierte Versuchsplanung auf diese angewiesen ist. Anschließend werden die in Abschnitt 5.4 vorgestellten Methoden der Versuchsplanung anhand des Benchmarksystems evaluiert.

5.5.1 Modellbildung

In diesem Abschnitt erfolgt ein Vergleich zwischen dem Runge-Kutta Neuronalen Netz (RKNN) und dem zeitdiskreten Neuronalen Netz (ZDNN) in Simulator- beziehungsweise Prädiktor-Konfiguration. Das RKNN wird dabei stets mittels dem klassischen Runge-Kutta Verfahren 4. Ordnung trainiert und simuliert (siehe Abschnitt 5.2.6). Um die Generalisierungsfähigkeit der Netze zu gewährleisten, wird zunächst das Verfahren des Early Stoppings verwendet. Auf diese Weise wird ein von Modelltyp und Trainingsverfahren unabhängiges Trainingsabbruchkriterium festgelegt und somit die Netze unter vergleichbaren Bedingungen trainiert.

Entscheidend für ein Modellbildungsverfahren ist dessen Robustheit gegenüber den Trainingsdaten, wie Abtastzeit und Trainingssignallänge, der Parametrierung der Modelle durch die Anzahl der Neuronen und Modellordnung und Störeinflüssen wie Messrauschen, sowie - für die gegebenen Randbedingungen am Verbrennungsmotoren-Prüfstand - Schwankungen in der Abtastzeit. Zunächst werden die unterschiedlichen Konfigurationen untersucht, anschließend die Abhängigkeit von der Wahl der Neuronen, wodurch die geeignete Neuronenzahl für die weiteren Simulationen ermittelt wird. Schließlich wird der Einfluss der Trainingssignallänge untersucht und die Abhängigkeit vom Messrauschen, Schwankungen in der Abtastzeit und der Wahl der Modellordnung. Die Generalisierungsfähigkeit der Netze bei Verwendung der Bayes'schen-Regularisierung und Modellkomitees im Vergleich zum Early Stopping wird am Ende untersucht.

Zum Training der RKNN und ZDNN wird der im Anhang B beschriebene LM-Algorithmus verwendet, mit dem in Algorithmus 1 beschriebenen Ablauf und Parametern.

Simulationsbedingungen

Die verwendeten Simulationseinstellungen, wenn nicht anders angegeben, sind für alle Simulationen die folgenden: Als Anregungssignal wird stets ein APRBS mit LHC-Verteilung (siehe Abschnitt 3.3.2) beziehungsweise zufälliger Verteilung der Designpunkte verwendet mit $T = 1000s$ und $t_s = 0.3s$. Mit 50 verschiedenen Trainingsdatensätzen mit LHC-Verteilung der Designpunkte werden RKNN und ZDNN trainiert, welche anschließend anhand von 100 Testdatensätzen mit zufälliger Verteilung der Designpunkte evaluiert werden. Als Validierungsdatensatz dient wiederum ein APRBS mit LHC-Verteilung der Designpunkte. Die Trainings- und Validierungsdaten können, wenn angegeben, verrauscht sein, die Testdaten sind hingegen immer unverrauscht. Das Training der Netze erfolgt bei der Untersuchung der Modellbildung offline und wird abgebrochen, wenn der Validierungsfehler 10 mal in Folge angestiegen ist. Anschließend werden die Parameter verwendet, welche zum geringsten Validierungsfehler geführt haben. Sowohl die RKNN als auch die ZDNN sind MLP-Netze mit 12 Neuronen und werden als Systeme 1. Ordnung mit dem LM-Algorithmus trainiert, das heißt beim RKNN wird die Ausgangsgleichung nicht verwendet und beim ZDNN werden einfach verzögerte Eingänge und ein einfach verzögerter Ausgang als Regressorvektor verwendet. Simulationen für RKNN höherer Ordnung mit Ausgangsgleichung finden sich in [15] und [18].

Training mit Stabilitätsnebenbedingungen

Der Einfluss der Nebenbedingungen aus Theorem 3 an die Netzgewichte, welche die Simulationsstabilität garantieren, werden hier kurz diskutiert. Es muss sichergestellt werden, dass die Nebenbedingungen die erzielte Modellgüte nicht beeinflussen und untersucht werden unter welchen Bedingungen dies sichergestellt werden kann. Da die Nebenbedingungen die Wahl der Netzgewichte einschränkt, kann davon ausgegangen werden, dass meist mehr Neuronen benötigt werden als beim Training ohne Nebenbedingungen. Zudem kann aus diesem Grund und aufgrund des komplizierteren Optimierungsproblem erwartet werden, dass die Optimierung der Netzgewichte mehr Rechenzeit in Anspruch nimmt, als eine Optimierung ohne Nebenbedingungen an die Netzgewichte. Das Training der RKNN ohne Nebenbedingungen erfolgt mittels des Levenberg-Marquardt Algorithmus 1, Anhang B. Werden hingegen die Stabilitätsnebenbedingungen verwendet, wird ein *interior-point* Verfahren ([10], [87]) verwendet, welches in Matlab 2010b durch den `fmincon` Befehl implementiert ist.

Die Trainings- und Testfehler finden sich in Abbildung 5.5 und sind in Form von Boxplots dargestellt. Wie aus den Simulationsergebnissen ersichtlich, schränken die Nebenbedingungen die Netzkomplexität nicht wesentlich ein, da der minimale Trainingsfehler bei RKNN trainiert mit und ohne Stabilitätsnebenbedingungen in derselben Größenordnung liegt. Allerdings sind die durchschnittlichen Optimierungsergebnisse beim Training mit Stabilitätsnebenbedingungen schlechter, als beim Training ohne Nebenbedingungen. Dies kann auf die genannte Einschränkung bei der Wahl der Gewichte durch die Nebenbedingungen und das kompliziertere Optimierungsproblem zurückgeführt werden. Bei den Testfehlern hingegen sind keine Unterschiede zu erkennen. Im Gegensatz zum Training mit Stabilitätsnebenbedingungen, werden Netze, welche ohne Stabilitätsnebenbedingungen trainiert werden, bei der Auswertung gelegentlich instabil. Bei etwa 3% der Auswertungen der RKNN, welche ohne Stabilitätsnebenbedingungen trainiert wurden, geht der Testfehler gegen unendlich.

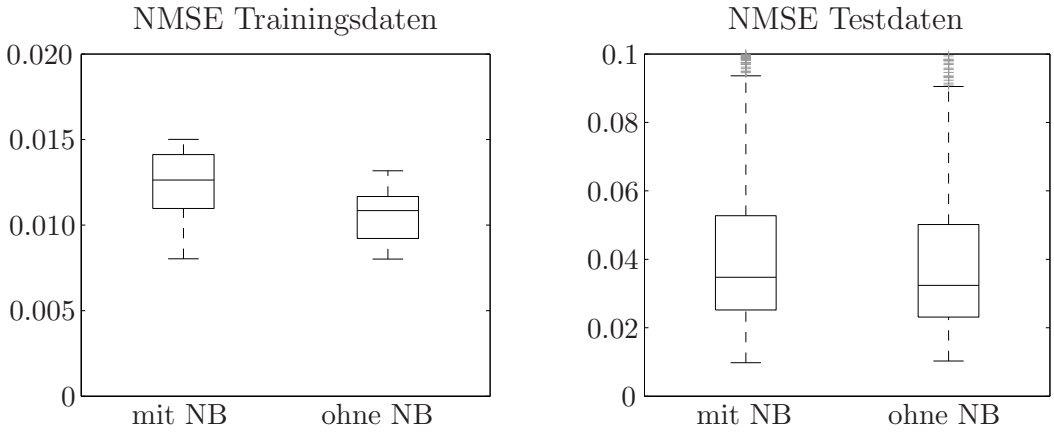


Abbildung 5.5: RKNN trainiert mit und ohne Stabilitätsnebenbedingungen.

Der Testfehler bei RKNN trainiert mit Stabilitätsnebenbedingungen weist hingegen einen maximalen NMSE von 0.3 auf.

Das Training der RKNN mit Nebenbedingungen dauert dabei im Mittel 4.5 mal so lange wie das Training ohne Nebenbedingungen.

Bei der Evaluierung der Generalisierungsfähigkeit anhand der Testdaten werden kaum Netze instabil. Dies liegt allerdings, wie Simulationen gezeigt haben, an der Verwendung eines Validierungsdatensatzes beim Training. Erfolgt das Training ohne Validierungsdatensatz, werden Netze regelmäßig instabil. Da das Training unter Berücksichtigung der Nebenbedingungen erheblich länger dauert, werden die folgenden umfangreichen Simulationen ohne Berücksichtigung der Nebenbedingungen durchgeführt. Um den Einfluss eventuell instabil gewordener Netze auf den Testfehler auszuschließen, werden RKNN, welche einen Testfehler von NMSE > 1 aufweisen, aus dem Trainings- und Testdatensatz gelöscht.

Im Gegensatz zum RKNN weisen die ZDNN geringere Stabilitätsprobleme auf, da der Netzausgang des ZDNN durch seine Struktur 4.1 ohnehin durch $||W^o||$ beschränkt ist. Daher erfolgt an dieser Stelle keine Evaluierung der ZDNN trainiert mit den Stabilitäsnebenbedingungen (4.11).

Abtastzeit

Die Zeitkonstanten des Benchmarksystems schwanken abhängig vom Eingang im Intervall $T = [6, 13.2]$. Entsprechend der im Abschnitt 2.1.4 vorgestellten Daumenregel, sollte die Abtastzeit $t_s \approx 1/10 - 1/20$ der kleinsten Zeitkonstante betragen und somit im Bereich $t_s = [0.3, 0.6]$ liegen. Der Daumenregel folgend wird auch die minimale Haltezeit des APRBS mit $T_{h,\min} = 6$ festgelegt. Um die Robustheit gegenüber der Wahl der Abtastzeit zu testen, wird diese zwischen $t_s = [0.05, 2]$ variiert. Dabei muss darauf geachtet werden, dass die minimale Haltezeit ein ganzzahliges Vielfaches der Abtastrate ist. Entsprechend werden Anregungssignale mit folgenden Abtastzeiten erzeugt:

$$\begin{aligned} t_s =& T_{h,\min} \left\{ \frac{1}{120}, \frac{1}{46}, \frac{1}{28}, \frac{1}{20}, \frac{1}{16}, \frac{1}{13}, \frac{1}{11}, \frac{1}{10}, \frac{1}{9}, \frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3} \right\} \\ \approx& \{0.05, 0.1304, 0.2143, 0.3, 0.375, 0.4615, 0.5455, 0.6, 0.6667, 0.75, 0.8571, 1, 1.2, 1.5, 2\} . \end{aligned}$$

Die Empfindlichkeit der Netze gegenüber der Wahl der Abtastzeit wird in Simulator- und Prädiktor-Konfiguration untersucht.

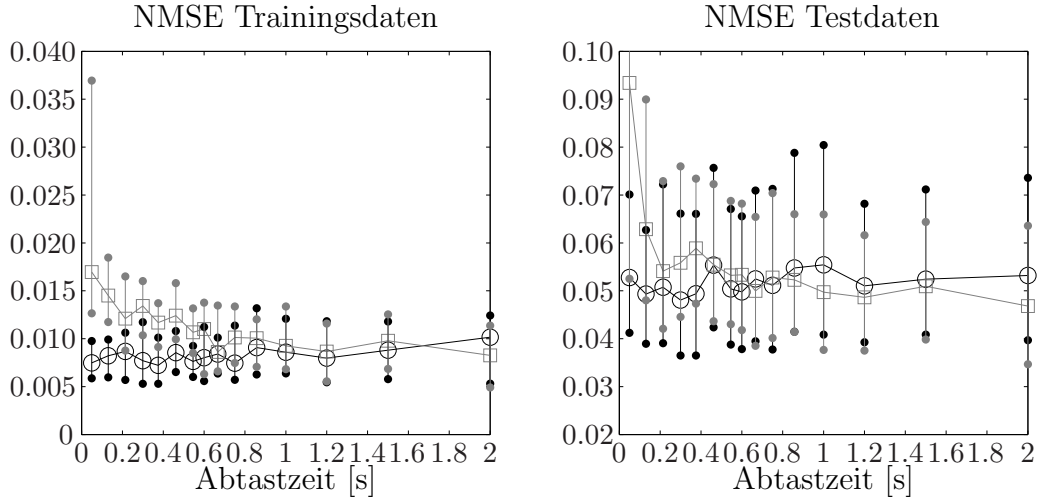


Abbildung 5.6: Simulator, unverrauschte Trainingsdaten. ZDNN: grau, RKNN: schwarz.

Zunächst werden die Unterschiede des RKNN und ZDNN in der für diese Arbeit relevanten Simulator-Konfiguration anhand unverrauschter Trainingsdaten diskutiert und mit der Prädiktor-Konfiguration verglichen, um anschließend auch einen Vergleich zu Modellen, trainiert mit verrauschten Trainingsdaten, ziehen zu können. Dazu werden die Trainingsdaten mit einem weißen Rauschen mit einem Signal zu Rauschverhältnis von 100 beaufschlagt (*engl. signal to noise ratio*: SNR). Im Anhang findet sich in Abbildung E.2 der typische Verlauf des verrauschten beziehungsweise unverrauschten Ausgangs des Benchmarksystems.

In Abbildung 5.7 und 5.6 ist der NMSE für die unterschiedlichen Abtastzeiten jeweils für Trainings- und Testdaten des ZDNN und RKNN in Prädiktor- und Simulator-Konfiguration dargestellt. Dabei ist die Darstellung an die Boxplots angelehnt. Unterhalb des schwarzen Kreises (RKNN) beziehungsweise grauen Quadrats (ZDNN) liegen 50% der NMSE-Werte. Die über schwarzen Linien mit den Median-Wert verbundenen kleinen schwarzen und grauen Kreise kennzeichnen den Bereich, indem 50% der NMSE-Werte liegen. Dies bedeutet, dass unterhalb des unteren schwarzen/grauen Kreises die 25% besten NMSE-Werte liegen und oberhalb des oberen schwarzen/grauen Kreises die 25% schlechtesten NMSE-Werte. Diese Darstellung hat den Vorteil, dass eventuelle Ausreißer nicht die Resultate verzerren. Weist das Ergebniss eine Normalverteilung auf, sind die schwarzen/grauen Kreise symmetriesch zum Median-Wert, welcher dann mit dem Mittelwert übereinstimmt.

Beim Trainingsfehler in Simulator-Konfiguration ist zu beobachten, dass der Trainingsfehler des ZDNN mit abnehmender Abtastzeit zunimmt, während der Trainingsfehler des RKNN gleich bleibt beziehungsweise tendenziell abnimmt. Dabei beträgt bei niedrigen Abtastzeiten der Trainingsfehler des RKNN ein Drittel des Trainingsfehlers des ZDNN. Bei hohen Abtastzeiten ist der Trainingsfehler beider Netztypen beinahe gleich und erst bei einer Abtastzeit ab 2 Sekunden ist der Trainingsfehler des ZDNN geringer als der des RKNN. Bei dem NMSE der Testdaten ist der Unterschied etwas geringer ausgeprägt, allerdings qualitativ gleich wie beim Trainingsfehler. Den höheren NMSE bei den Trainings- als auch Testdaten kann auf die schlechtere Generalisierungsfähigkeit der ZDNN zurück-

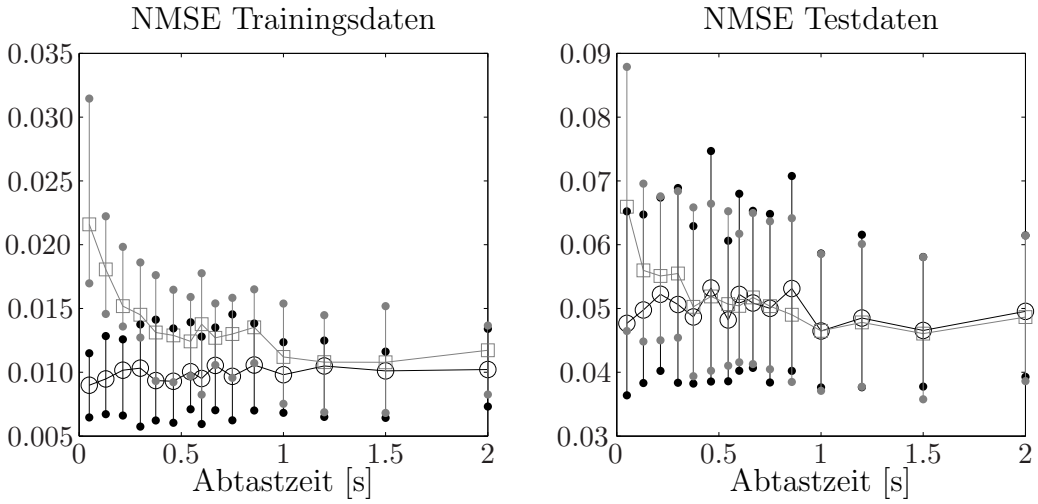


Abbildung 5.7: Prädiktor, unverrauschte Trainingsdaten. ZDNN: grau, RKNN: schwarz.

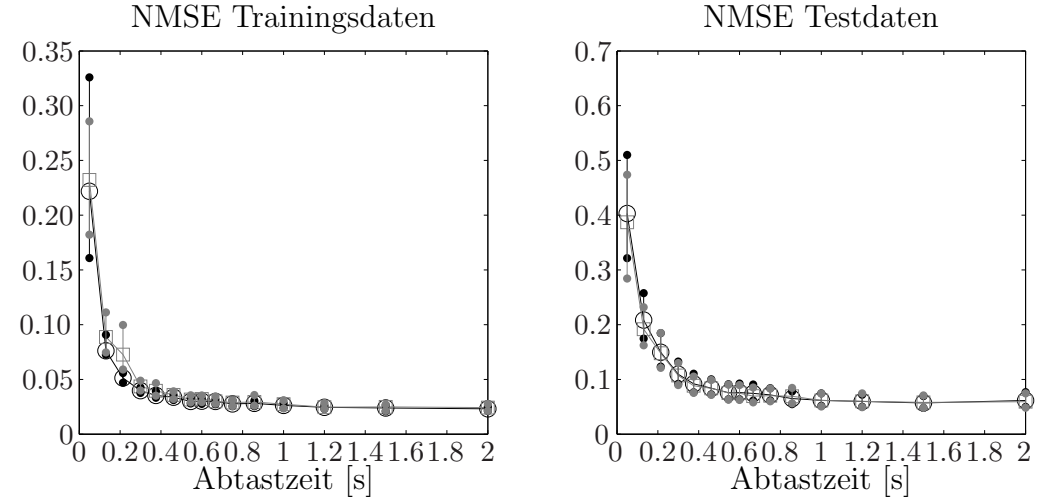


Abbildung 5.8: Prädiktor, verrauschte Trainingsdaten. ZDNN: grau, RKNN: schwarz.

geführt werden, wodurch das Training bei Verwendung eines Validierungssignals früher abgebrochen wird. Die höheren Trainings- und Testdatenfehler des RKNN bei hohen Abtastzeiten kann auf Ungenauigkeiten der numerischen Integration der RKNN bei zu großen Schrittweiten zurückgeführt werden.

In der Prädiktor-Konfiguration (Abbildung 5.7) ist der Zusammenhang ähnlich wie bei der Simulator-Konfiguration. Der Trainingsfehler, ausgewertet in der Simulator-Konfiguration, ist entsprechend höher als in der Simulator-Konfiguration, da die Trainings-Konfiguration nicht mit der Auswerte-Konfiguration übereinstimmt. Auch in der Prädiktor-Konfiguration ist das RKNN dem ZDNN überlegen. Dies kann unter anderem darauf zurückgeführt werden, dass das RKNN bei den Runge-Kutta Integrationsschritten in Simulator-Konfiguration ausgewertet und trainiert wird, so dass im Gegensatz zum RKNN eine 4-Schritte-Prädiktion trainiert wird. Dies kommt der Simulator-Konfiguration wesentlich näher als eine 1-Schritt-Prädiktion, wovon die Modellqualität profitiert.

In den Abbildungen 5.8 und 5.9 finden sich die Simulationsergebnisse für Modelle, trainiert in Prädiktor- und Simulator-Konfiguration anhand verrauschter Trainingsdaten. In

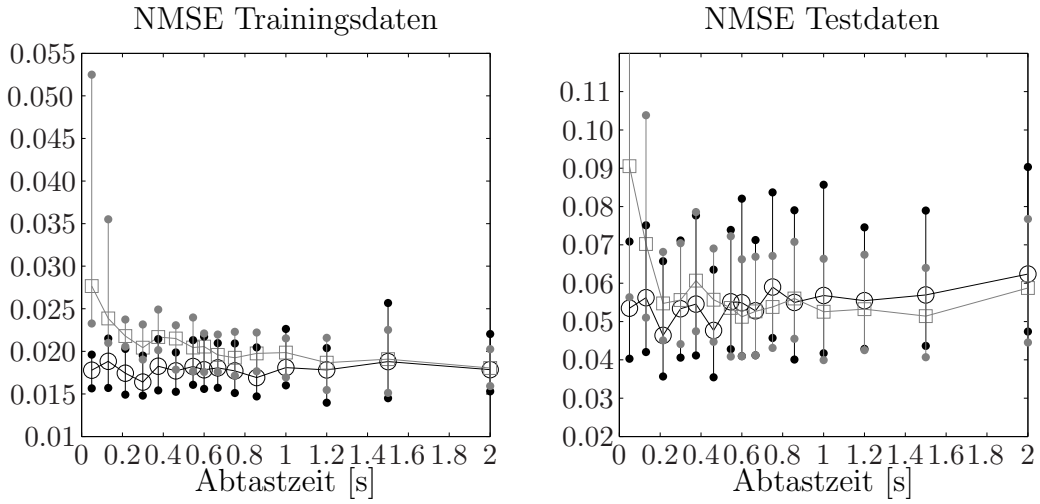


Abbildung 5.9: Simulator, verrauschte Trainingsdaten. ZDNN: grau, RKNN: schwarz.

der Prädiktor-Konfiguration ist, wie aus Abbildung 5.8 ersichtlich, die Abhängigkeit von der Abtastzeit sowohl beim RKNN als auch ZDNN sehr ausgeprägt. Der Trainings- und Testfehler wird bei zunehmender Abtastzeit zunehmend größer. Dies ist darauf zurückzuführen, dass sich der Systemausgang zwei aufeinander folgender Zeitschritte kaum ändert ($y(k) \approx y(k-1)$) und diese vom Modell abzubildende Dynamik im Rauschen unter geht und somit nicht mehr erfasst werden kann. Im Gegensatz dazu bleibt die Modellqualität in der Simulator-Konfiguration (Abbildung 5.9) vom Messrauschen weitgehend unberührt, wobei der Fehler des RKNN wiederum geringer ist als der des ZDNN. Die Abhängigkeiten von der Abtastzeit sind qualitativ und quantitativ dieselben wie im unverrauschten Fall (Abbildung 5.6). Dabei ist zu beachten, dass der Trainingsfehler höher ist, da dieser im Gegensatz zum Testfehler basierend auf verrauschten Daten berechnet wird.

Im Gegensatz zum ZDNN ist beim RKNN in Simulator-Konfiguration die korrekte Wahl der Abtastzeit weniger kritisch, da die Generalisierungsfähigkeit der RKNN bei ausreichend niedrigen Abtastzeiten unabhängig von der Wahl der Abtastzeit erhalten bleibt, während beim ZDNN bei zu niedrigen beziehungsweise zu hohen Abtastzeiten die Modellqualität abnimmt. Im Schnitt beträgt die Trainingszeit für die RKNN in etwa das Vierfache der Trainingszeit der ZDNN, wie dies aufgrund der vier Runge-Kutta Schritte pro Trainingspaar auch zu erwarten ist.

Neuronen

Um die Abhängigkeit der Modellgüte von der Anzahl der Neuronen zu ermitteln, werden RKNN und ZDNN Modelle mit unterschiedlicher Anzahl an Neuronen entsprechend der Standard-Simulationsbedingungen in Simulator-Konfiguration trainiert und getestet. Die Prädiktor-Konfiguration wird im Folgenden nicht mehr näher betrachtet, da die erzielte Modellgüte für die Aufgabenstellung relevante Auswertung in Simulator-Konfiguration nicht ausreichend ist. In Abbildung 5.10 finden sich die Trainings- und Testfehler der RKNN und ZDNN Modelle für unterschiedliche Anzahl an Neuronen.

Wie aus Abbildung 5.10 hervorgeht nimmt der Trainingsfehler mit steigender Anzahl an Neuronen ab während der Testfehler zunächst sinkt und ab 17 Neuronen wieder leicht zu steigen beginnt. Ab 11 Neuronen reicht die Flexibilität der Netze aus, um die Trai-

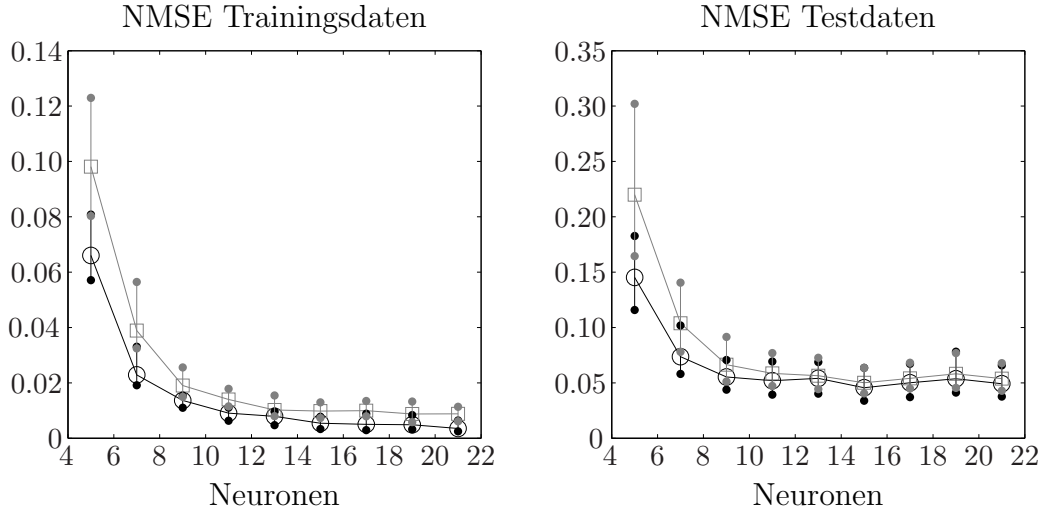


Abbildung 5.10: NMSE für verschiedene Neuronenanzahlen. ZDNN: grau, RKNN: schwarz.

ningsdaten ausreichend gut approximieren zu können und der Trainingsfehler nimmt nur noch wenig ab. Der Generalisierungsfehler nimmt hingegen trotz der Verwendung von Validierungsdaten als Trainingsabbruchkriterium ab 17 Neuronen wieder leicht zu und der Median des Trainingsfehlers des RKNN 0.0054 beträgt circa 12% des Testfehlers 0.0456. Unabhängig von der Anzahl der verwendeten Neuronen liegt der Trainings- und Testfehler des RKNN stets unter dem des ZDNN.

Trainingssignallänge

Wie in Abschnitt 2.2 diskutiert, ist die Trainingssignallänge von entscheidender Bedeutung für die Generalisierungsfähigkeit der Modelle. Der Varianzfehler der Modelle fällt proportional zum Trainingsdatenumfang mit $\propto \frac{1}{\sqrt{K}}$, wobei K die Anzahl der Trainingsdaten angibt. Dadurch nimmt wie erwartet bei steigender Anzahl an Trainingsdaten die Generalisierungsfähigkeit der Modelle zu. Die Ergebnisse der Trainings- und Testfehler für APRBS unterschiedlicher Dauer finden sich in Abbildung 5.11.

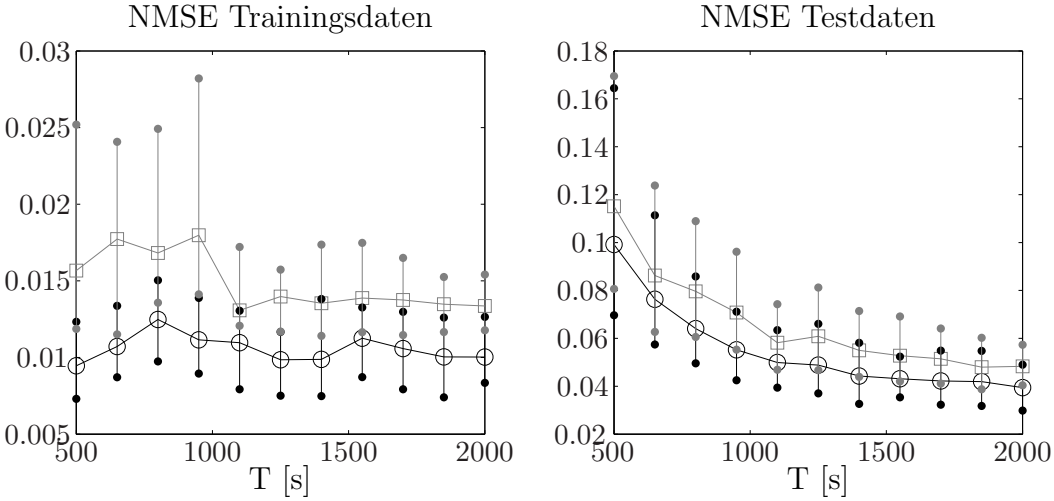


Abbildung 5.11: NMSE für verschiedene Trainingssignallänge. ZDNN: grau, RKNN: schwarz.

Wie aus den Simulationsergebnissen ersichtlich, bleibt der Trainingsfehler unabhängig von

der Trainingssignallänge annähernd konstant, während der Testfehler bei steigendem Trainingsdatenumfang sinkt. Unabhängig von der Trainingssignallänge weist das RKNN stets einen geringeren Testfehler als das ZDNN auf. Die Differenz des Testfehlers des RKNN und ZDNN ist dabei unabhängig von der Trainingssignallänge gleich bleibend.

Rauschen

Um die Abhängigkeit der Modellgüte von der Stärke des Rauschsignals zu untersuchen, werden die Trainings- und Validierungsdaten mit einem weißen Rauschen mit unterschiedlichem Signal zu Rauschverhältnissen beaufschlagt. Die Testdaten sind immer unverrauscht. In Abbildung E.2 ist ein typischer verrauschter und unverrauschter Ausgangsverlauf des Benchmarksystems abgebildet.

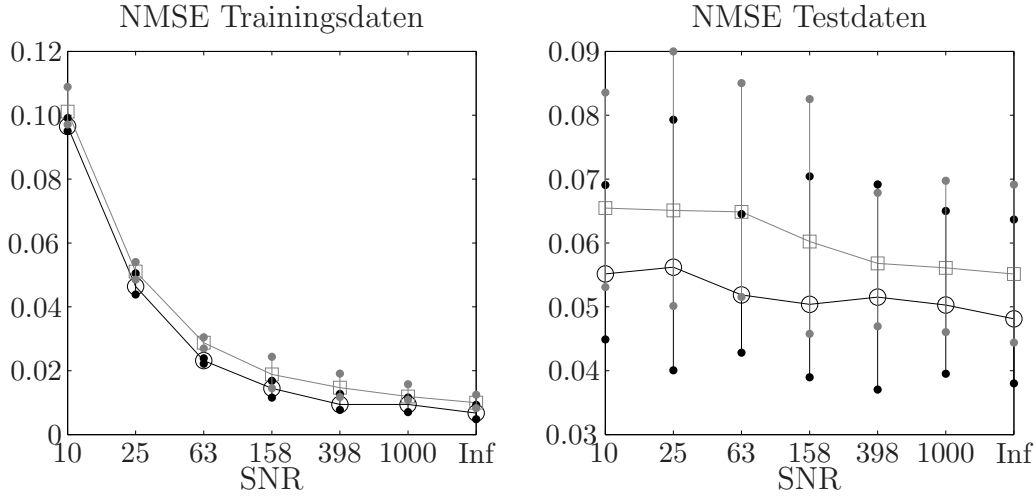


Abbildung 5.12: NMSE für verschiedene SNR. ZDNN: grau, RKNN: schwarz.

Wie aus Abbildung 5.12 ersichtlich, nimmt der Trainingsfehler in etwa exponentiell mit steigendem SNR ab. Der Testfehler, welcher anhand unverrauschter Datensätze erfolgt, weist hingegen eine relativ geringe Abhängigkeit vom SNR auf. Der Unterschied zwischen dem NMSE von Testdaten basierend auf Modellen trainiert mit unverrauschten Trainingsdaten und trainiert mit Trainingsdaten mit einem SNR=10 beträgt in etwa 10%. Somit können RKNN und ZDKNN trainiert in Simulator-Konfiguration das Systemverhalten auch bei stark verrauschen Trainingsdaten gut erfassen. Der Verlauf des Testfehlers deutet allerdings darauf hin, dass das RKNN noch robuster hinsichtlich verrauschter Trainingsdaten ist als das ZDNN. Analog zu den vorherigen Beispielen ist das RKNN dem ZDNN hinsichtlich der Modellgüte überlegen.

Schwankung der Abtastzeit

Wie zu Beginn des Kapitels bereits angesprochen, kann eine Echzeitkommunikation zwischen Optimierungssystem und Steuergerät über die in Abbildung 5.1 dargestellte Prüfstandskommunikationsstruktur nicht sichergestellt werden. Abhängig von der Auslastung des Netzwerks und des Prüfstandsautomatisierungssystems kann es zu Schwankungen in der Abtastzeit kommen. Diese sind, wie sich bei Messungen gezeigt hat, nicht sonderlich groß, sollten aber dennoch berücksichtigt werden, um eine optimale Modellgüte sicherzustellen. Die Schwankungen in der Abtastzeit können, wie im Abschnitt 5.2 diskutiert, bei

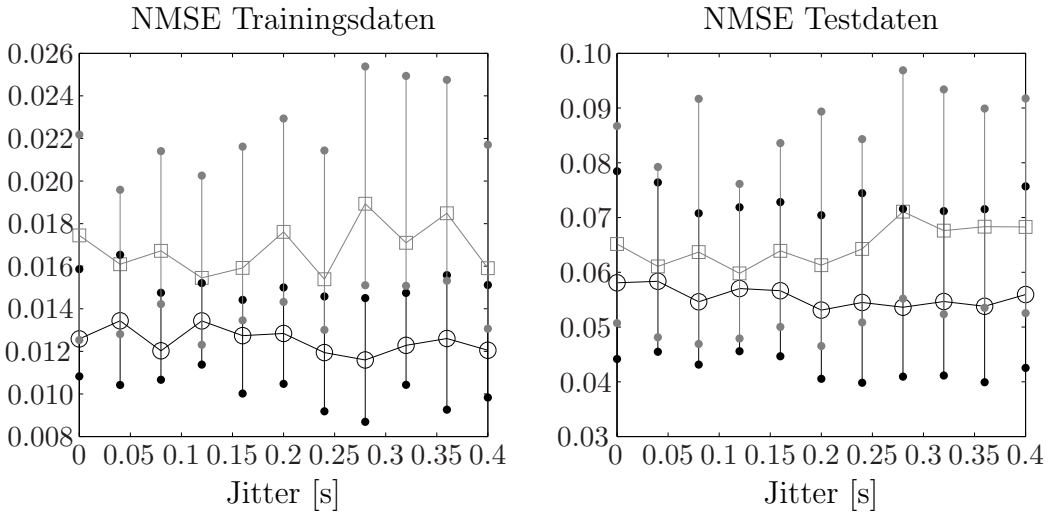


Abbildung 5.13: NMSE für verschiedene Jitter. ZDNN: grau, RKNN: schwarz.

den ZDNN im Gegensatz zu den RKNN nicht berücksichtigt werden. Um die Auswirkungen einer schwankenden Abtastzeit zu untersuchen, wird diese in den folgenden Beispiel für die Trainingsdaten und Validierungsdaten zufällig variiert, während die Abtastzeit für die Testdaten konstant bleibt. Die Schwankung der Abtastzeit wird auch als Jitter bezeichnet und beträgt in der Simulationen zwischen $0 - 0.4$. Bei einem Jitter von 0.4 schwanken die Abtastzeiten um maximal 40% um die gewünschte Abtastzeit von 0.25, womit die tatsächliche Abtastzeit im Bereich $t_s = [0.15, 0.35]$ liegt.

In Abbildung 5.13 ist der Fehlerverlauf für die Trainings- und Testdaten dargestellt. Wie aus den Abbildungen ersichtlich, bleibt der Testfehler bei den RKNN unabhängig von der Schwankung der Abtastzeit annähernd konstant während beim Testfehler des ZDNN bei zunehmendem Jitter der Median des Testfehler wie auch die Varianz des Testfehlers zunimmt. Das Ergebnis unterstreicht den Vorteil des RKNN gegenüber dem ZDNN bei schwankenden Abtastzeiten.

Überschätzung der Systemordnung

Die Systemordnung kann aufgrund von physikalischen Systemwissen in der Regel abgeschätzt werden. So weisen viele dynamischen Größen des Verbrennungsmotor ein PT1-Verhalten aus (zum Beispiel Temperaturen). Ist die Systemordnung allerdings unbekannt, kann ein Unterschätzen derselben zu schlechten Approximationsergebnissen führen. Ein Überschätzen der Systemordnung bringt zusätzliche Freiheitsgrade des Modells mit sich, welche die Approximationsgüte und Generalisierungsfähigkeit des Modells nicht beeinträchtigen sollten. Um die Auswirkung einer Überschätzung der Systemordnung auf die Modellbildungsverfahren zu untersuchen, werden Modelle mit unterschiedlicher Ordnung trainiert und simuliert.

Die in Abbildung 5.14 dargestellten Ergebnisse zeigen, dass beide Modellbildungsverfahren robust sind gegenüber einer Überschätzung der Systemordnung. Ab der vierten Modellordnung nimmt beim RKNN sowohl das Trainingsfehler als auch der Generalisierungsfehler zu. Dies lässt darauf schließen, dass es nicht zu einem Overfitting kommt, sondern dass die Generalisierungsfähigkeit des RKNN abnimmt und es durch den höheren Validierungsfehler zu einem vorzeitigen Trainingsabbruch kommt. Bei einer hohen Überschätzung der

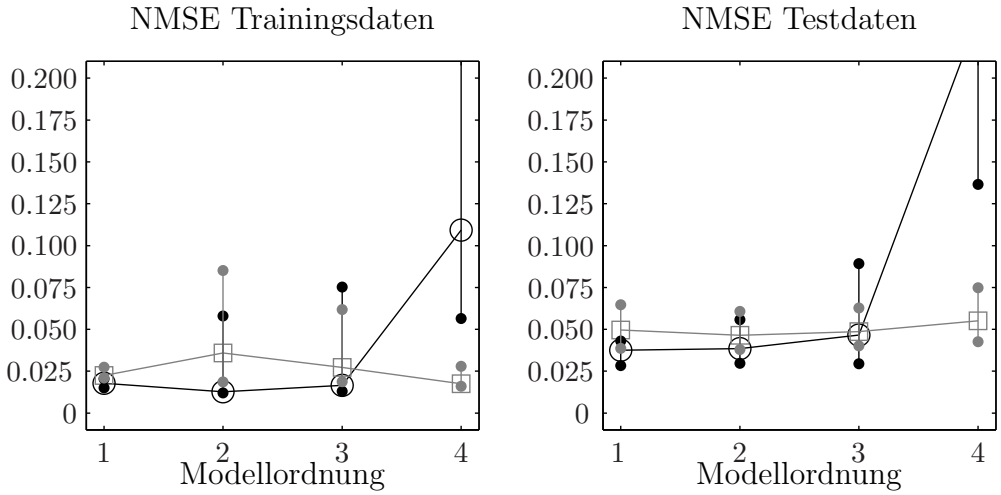


Abbildung 5.14: NMSE für verschiedene Modellordnungen. ZDNN: grau, RKNN: schwarz.

Systemordnung ist daher das ZDNN robuster gegenüber der Wahl der Modellordnung als das RKNN.

Generalisierungsfähigkeit

Um den Einfluss der Regularisierungsverfahren (siehe Abschnitt 4.1.3) auf die Generalisierungsfähigkeit zu überprüfen, werden RKNN mit und ohne Bayes'sche-Regularisierung beziehungsweise mit und ohne Validierungssignal (Early Stopping) trainiert. Die Ergebnisse werden mit jenen eines Modellkomitees bestehend aus $m = 4$ RKNN verglichen. Die Gewichtung der Einzelmodelle wird anhand der Bayes'schen-Regularisierungsparameter berechnet (siehe Abschnitt 5.3), wobei Einzelmodelle, deren Gewichtung $\lambda_i < \frac{1}{2m}$ ist, nicht bei der Auswertung des Komitees berücksichtigt werden. Dadurch wird sichergestellt, dass schlechte Modelle die Güte des Modellkomitees nicht verzerren. Die ZDNN werden an dieser Stelle nicht mehr berücksichtigt, da sich in den vorhergehenden Simulationen durchwegs die Überlegenheit der RKNN gezeigt hat.

Da beim Training ohne Validierungssignal die Netze bei der Auswertung häufig instabil werden, werden die RKNN, im Gegensatz zu den bisherigen Simulationen, mit den Stabilitätsnebenbedingungen trainiert.

In Abbildung 5.15 finden sich die Simulationsergebnisse. Sowohl das Training mit Validierungssignal (val) als auch die Bayes'sche-Regularisierung (br) führen zu einer verbesserten Generalisierungsfähigkeit der RKNN. Die besten Ergebnisse liefert dabei das Modelkomitee (kom).

Zusammenfassung

Wie aus den zahlreichen Simulationen ersichtlich, weist das RKNN eine bessere Generalisierungsfähigkeit als das ZDNN auf und ist zudem robuster bezüglich Schwankungen in der Abtastzeit, wodurch die erzielte Modellgüte der RKNN den der ZDNN im realen Prüfstandsbetrieb überlegen ist. Zudem ist die Wahl der Abtastzeit bei RKNN weniger kritisch, da eine niedrige Abtastzeit, im Gegensatz zum ZDDN, nicht die Modellgüte nachteilig beeinflussen kann. Ein Nachteil der RKNN ist die im Mittel vierfach längere Trainingszeit

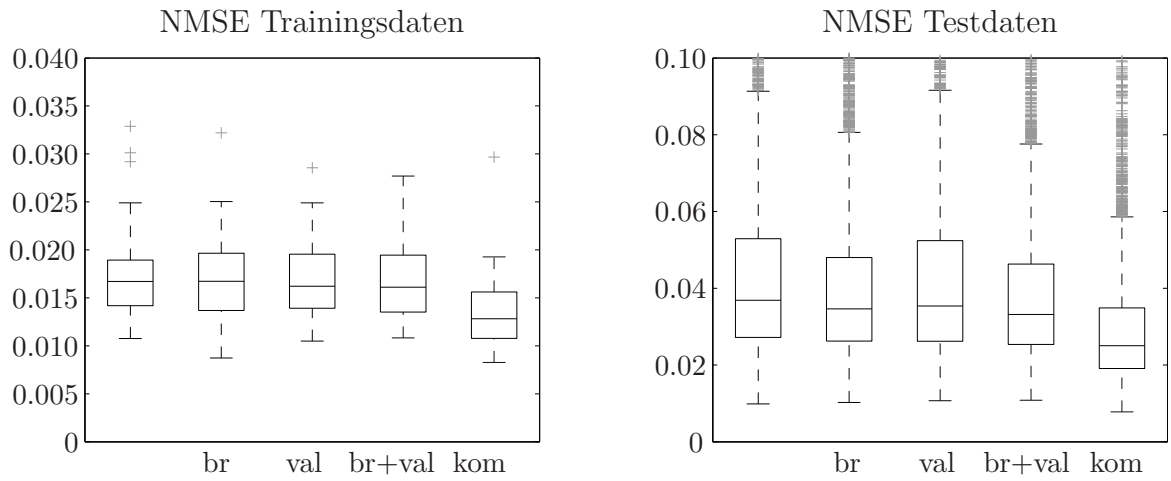


Abbildung 5.15: NMSE für verschiedene Regularisierungsmethoden.

als die der ZDNN. Zudem nimmt die Trainingszeit der RKNN bei der Verwendung der Stabilitätsnebenbedingungen an die Gewichte weiter zu.

Die Generalisierungsfähigkeit der RKNN lässt sich, wie anhand der Simulationen gezeigt, erhöhen, indem Modellkomitees in Verbindung mit Bayes'scher Regularisierung verwendet werden. Die Verwendung eines Validierungssignals zur Bestimmung des Trainingsabbruchkriteriums kann dabei die Modellgüte weiter verbessern und die Trainingszeiten verkürzen, da mit einem Validierungsdatensatz sehr wahrscheinlich ein stabiles Modell gefunden wird, ohne die Stabilitätsnebenbedingungen an die Gewichte im Training berücksichtigen zu müssen.

Tabelle 5.1: Bewertung des RKNN und ZDNN.

	RKNN	ZDNN
Wahl der Abtastzeit	++	-
Wahl der Neuronenzahl	++	++
Robustheit gegen Rauschen	+	+
benötigte Trainingsdaten	+	+
Schwankende Abtastzeit	++	–
Wahl der Modellordnung	+	++
Generalisierungsfähigkeit	++	+
Trainingsgeschwindigkeit	-	+

In Tabelle 5.5.1 findet sich eine Zusammenfassung/Bewertung der Eigenschaften der RKNN und ZDNN trainiert in Simulator-Konfiguration mit Validierungsdaten.

5.5.2 Online-Versuchsplanung

Die in Abschnitt 5.5.1 durchgeführten Simulationen zur Evaluierung der Modellgüte des RKNN basieren auf einem offline berechneten APRBS mit LHC-Verteilung der Designpunkte. In diesem Abschnitt werden die in Abschnitt 5.4 eingeführten Methoden der Offline-Versuchsplanug mit denen der Online-Versuchsplanung anhand des Benchmark-Systems verglichen. Es wird ein Modellkomitee bestehend aus $m = 4$ RKNN verwendet, deren Gewichtung basierend auf den Bayes'schen Regularisierungsparametern berechnet wird (siehe Abschnitt 5.3). Dabei wird die Gewichtung der Einzelmodelle bei der Versuchsplanung und Evaluierung unterschiedlich gehandhabt. Einzelmodelle, deren Gewichtung $\lambda_i < \frac{1}{2m}$ ist, werden nur bei der Versuchsplanung, nicht aber bei der Auswertung des Komitees, berücksichtigt. Damit wird sichergestellt, dass Modelle deren Genauigkeit weit unter der der anderen Modellen liegt, nicht die Genauigkeit des Gesamtkomitees beeinflussen [95]. Bei der Versuchsplanung werden sie dennoch berücksichtigt, damit auch diese von der Versuchsplanung profitieren und gegebenenfalls im weiteren Versuchsverlauf wieder in das Komitee aufgenommen werden können.

Um die Simulation möglichst realistisch zu halten, wird der zulässige Eingangsraum begrenzt. Dabei ist bei Beginn der Simulation die Begrenzung nicht bekannt und erst während der Simulation werden Limitmodelle erstellt, welche die Beschränkung des Eingangsraums nachbilden. Als Limitmodell wird das in [48] eingeführte Kartoffelmodell verwendet und die Eingangsraumgrenzen durch die in Abbildung 6.1 dargestellten Einschränkung modelliert [48]. Die Verstellstrategie, Limitbehandlung und Limitmodellierung wird im nachfolgenden Kapitel 6.1.2 genauer beschrieben. Die während der Laufzeit erstellten Limitmodelle werden bei der Online-Versuchsplanung berücksichtigt.

Als Fehlermaß zur Evaluierung des dynamischen Verhaltens wird der normalisiere mittlere quadratische Fehler (NMSE) und zur Evaluierung des Stationärverhaltens (welcher nach Theorem 7 berechnet wird) der normalisierte mittlere absolute Fehler (NMAE) verwendet. Im Anhang C.3 findet sich eine genauere Beschreibung der Fehlermaße.

Simulative Evaluierung

Die Evaluierung erfolgt wiederum anhand des Benchmark-Systems. Zunächst werden verschiedene Strategien zur Designpunkteverteilung miteinander verglichen. Es werden jeweils 50 verschiedene APRBS Versuchspläne mit zufälliger, LHC- und D-optimaler (Polynom 3. Ordnung) Verteilung mit $T_{h,\min} = 6s$, $T = 1000s$ und $T_r = 0$ erstellt. Die Online-Verfahren mit den Kosten (5.26) beziehungsweise (5.27) benötigen vortrainierte Modelle. Daher werden zunächst LHC-Versuchspläne mit $T = 500s$ erstellt und dann in den nächsten $500s$ die Designpunkte entsprechend der verwendeten Kriterien gewählt. Die Modellkomitees werden immer online adaptiert, das heißt mit den bei einem neuen Designpunkt anfallenden Daten trainiert. Die Optimierung erfolgt stets für einen Designpunkt und zur Maximierung der FIM wird unter Berücksichtigung der Limitmodelle das Kriterium der D-Optimalität verwendet. Der Ausgang des Testsystems wird mit einem Rauschen mit Signal-Rausch-Verhältnis von SNR=25 beaufschlagt. Alle Datensätze, die nicht zum Training des Modellkomitees verwendet werden, dienen als Validierungsdatensätze für das dynamische Verhalten. Zur Validierung des Stationärverhaltens wird für 1000 LHC-verteilte Punkte der Mittelwert und die Standardabweichung des NMAE berechnet.

Tabelle 5.2: instationäre und stationäre Validierungsdaten.

	instationär: NMSE	stationär: NMAE
randomisiert	0.1851 ± 0.2513	0.0294 ± 0.0161
LHC	0.1494 ± 0.1298	0.0265 ± 0.0066
D-opt (offl.)	0.1001 ± 0.0089	0.0294 ± 0.0065
D-opt (onl.)	0.0876 ± 0.0048	0.0225 ± 0.0043
QbC	0.0917 ± 0.0093	0.0262 ± 0.0093

Wie aus den in Tabelle 5.5.2 aufgeführten Ergebnissen ersichtlich, führt die zufällige Verteilung der Designpunkte sowohl beim Instationär- als auch beim Stationärverhalten zum größten Fehler. Die Latin-Hypercube Verteilung führt durch die gleichmäßige Abdeckung des Eingangsraums zu einem reduzierten Fehler, der durch die auf einem Polynom 3. Grades basierende D-optimale Verteilung weiter verringert werden kann. Allerdings kommt das Testsystem E.1 der D-optimalen Verteilung entgegen, da diese die Designpunkte am Randbereich des Eingangsraums konzentriert und die Funktionswerte dort die höchsten Werte annehmen. Daher haben Modellierungsfehler am Randbereich den größten Einfluss auf das Fehlermaß. Die modellbasierten Versuchspläne führen sowohl beim instationären als auch beim stationären Fehler zu einer weiteren Verbesserung der Ergebnisse, wobei der D-optimale Versuchsplan zu den besten Ergebnissen führt.

Tabelle 5.3: stationäre Validierungsdaten.

	global (NMAE)	lokal (NMAE)
D-opt (offl.)	0.0578 ± 0.0089	0.1460 ± 0.0236
D-opt	0.0274 ± 0.0134	0.0433 ± 0.0132
D-opt+(5.28)	0.0245 ± 0.0034	0.0477 ± 0.0107
D-opt+(5.30)	0.0251 ± 0.0070	0.0426 ± 0.0146
D-opt+(5.28)+(5.30)	0.0236 ± 0.0060	0.0450 ± 0.0144

Zum Testen der Kriterien (5.28) und (5.30) in Kombination mit dem D-optimalen Kriterium (5.27) wird das Stationärverhalten des Modellkomitees einmal global (1000 LHC-verteilte Punkte) und einmal lokal um die drei lokalen Minima evaluiert (10x10 Raster im Bereich $[-0.1, 0.1]$ um die lokalen Minima). Die Normierung des lokalen Fehlers erfolgt anhand des Wertebereichs der RadCos-Funktion in diesem Bereich. Der Faktor μ wird so angepasst, dass dieser zu Beginn der Online-Versuchsplanung 0 und am Ende 1 ist. Der Übergang wird durch eine sigmoide Funktion festgelegt [65]. Die Ergebnisse finden sich in Tabelle 5.5.2

Wie aus den Ergebnisse hervorgeht, führen die Kriterien (5.28) und (5.30) zu einem global beziehungsweise lokal verbesserten Stationärverhalten, wobei die Konzentration der Versuchsplanung auf die Umgebung um potentielle Minima die Stationärgenauigkeit dort weiter steigern kann. Eine Kombination aus beiden Kriterien stellt einen Kompromiss zwischen beiden dar.

5.6 Diskussion im Sinne der Aufgabenstellung

In diesem Kapitel werden neue Verfahren der Modellbildung und Versuchsplanung entwickelt, welche den Anforderungen aus der Aufgabenstellung an die Modellbildung und Versuchsplanung Rechnung tragen.

Wie durch umfangreiche Simulationen gezeigt wird, ist die Generalisierungsfähigkeit der Modelle sowie deren Genauigkeit im Stationärverhalten sehr gut. Durch die üblicherweise geringe Anzahl an Parametern der RKNN lassen sich die Modelle schnell auswerten, sodass sie für den Online-Einsatz geeignet sind. Es kann zudem gezeigt werden, dass die Modelle robust sind gegenüber der Parametrierung, welche die Wahl der Neuronenzahl, der Abtastzeit und der Modellordnung sowie die Anzahl der Trainingsdaten betreffen. Durch die Bayes'sche Regularisierung und die Adaption der Neuronenzahl kann eine selbstständige Anpassung an die Komplexität der Aufgabenstellung realisiert werden. Zudem sind die Modelle robust gegenüber Störeinflüssen wie Rauschen und schwankenden Abtastzeiten.

Die Versuchsplanung ermöglicht den späteren Einsatzzweck der Modelle zu berücksichtigen und mittels der vorgestellten Kriterien die zu erreichende Modellgüte festzulegen. Diese als untere Schranken an das Kriterium definierten Werte, dienen als Abbruchkriterium für die Versuchsplanung. Dadurch kann die Anzahl der Trainingsdaten in Abhängigkeit der gestellten Anforderungen an die Modellqualität und der Komplexität der Aufgabenstellung selbständig von der Versuchsplanung bestimmt werden. Einschränkungen an den Versuchsraum sowie Grenzen an die System- beziehungsweise Modellausgänge können in Form von Nebenbedingungen berücksichtigt werden, womit ein sicherer Prüfstandsbetrieb gewährleistet werden kann.

Durch diese neu entwickelten Verfahren der Modellbildung und Versuchsplanung ist ein einfache Identifikation und ein robuster Einsatz der Modelle am Prüfstand möglich.

6 Online-Identifikation am Motorprüfstand

In diesem Kapitel wird die Umsetzung der im letzten Kapitel erarbeiteten Methoden der Versuchsplanung und Modellbildung am Motorprüfstand behandelt. Zudem wird die Verstellstrategie, Limitbehandlung und Phasenplanung beschrieben. Der Ablauf der Online-Identifikation und -Versuchsplanung basiert auf Heuristiken, die weitestgehend von den Vorgängerarbeiten zur Identifikation statischer Modelle [65], [48] und [47] übernommen werden. Im ersten Abschnitt wird die Implementierung und Parametrierung der Online-Identifikations- und Optimierungsumgebung behandelt. Zudem werden Heuristiken vorgestellt, die es erlauben mit Totzeiten, Messausreißern, unterschiedlichen Zielgrößendynamiken und Limitverletzungen umzugehen. Anschließend erfolgt die Evaluierung der Versuchsplanung und Modellbildung am Motorprüfstand.

Die Anbindung der Online-Identifikationsumgebung an den Prüfstand erfolgt über einen eigenen Rechner, welcher an das Prüfstandsautomatisierungssystem angebunden ist (siehe Abbildung 5.1 und Abschnitt 5.1).

6.1 Implementierung und Heuristiken

In diesem Abschnitt wird der Ablauf der Online-Modellbildung am Prüfstand erläutert und die nötigen Heuristiken zur Umsetzung in der Praxis vorgestellt. Neben der Parametrierung der Phasenplanung, welche den Versuch in initiale Offline-Versuchsplanung, modellbasierte Online-Versuchsplanung und eventuell anschließender Optimierungsphase unterteilt, müssen in der Praxis Limitverletzungen und die Messunsicherheiten berücksichtigt werden.

6.1.1 Phasenplanung

Analog zum *mbminimize*-Algorithmus, welcher in [65], [48] entwickelt und in [47] hinsichtlich der Anwendung am Prüfstand erweitert und optimiert wurde, wird der Ablauf der Modellbildung und Versuchsplanung in drei Phasen unterteilt.

Startphase

In der Startphase erfolgt eine grobe Erfassung des dynamischen Systemverhaltens des Motors durch einen initialen Versuchsplan. Standardmäßig werden die Hälfte der Designpunkte der Startphase durch eine LHC-Verteilung und die andere Hälfte durch eine D-optimale Verteilung festgelegt (siehe Kapitel 3.3). Dadurch wird sicher gestellt, dass der Randbereich des Versuchsraums abgedeckt wird, sowie eine gleichmäßige Verteilung der Designpunkte im Versuchsraum erfolgt. Vom Anwender ist die Anzahl der Messpunkte

sowie der Modellansatz für das D-optimale Design (quadratisch, kubisch,...) festzulegen. Eventuell vorhandenes Vorwissen, z. B. in Form bereits bekannter Limitgrenzen, kann bei der Erstellung der Versuchspläne berücksichtigt werden. In Abbildung 6.1 findet sich eine Darstellung einer solchen Verteilung von 50 Designpunkten unter Berücksichtigung bekannter Einschränkungen des Eingangsraums. Neben den Einschränkungen des Eingangsraums (schwarz umrandete und grau gepunktete Flächen) und den Designpunkten (schwarze Punkte) ist der Zentralpunkt dargestellt (schwarzer Kreis), welcher als Ausgangs- und Rückzugspunkt der Versuchsplanung fungiert. Tritt eine Limitverletzung auf, wird stets der Zentralpunkt angefahren und dort auf das Abklingen der Limitverletzung gewartet. Dabei wird vorausgesetzt, dass im Zentralpunkt ein sicherer Motorbetrieb möglich ist. Entsprechend sorgfältig muss der Zentralpunkt vor Beginn der Versuchsplanung gewählt werden.

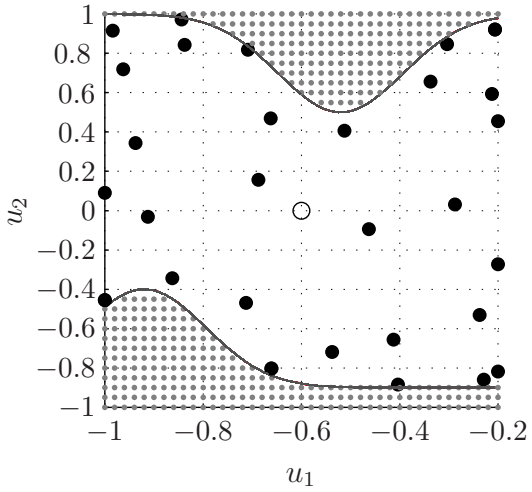


Abbildung 6.1: Beispiel: initialer Versuchsplan (schwarze Punkte) basierend auf LHC- und D-optimalen Versuchsplänen mit Eingangsraumbeschränkungen (graue Flächen) und Zentralpunkt (schwarzer Kreis).

Explorationsphase

Nach Ablauf der Startphase wird anhand der Messdaten ein Modellkomitee für die zu approximierende(n) Zielgröße(n) erstellt. Das Komitee besteht normalerweise aus 4 Neuronalen Netzen, welche wie in Abschnitt 5.2 beschrieben, erstellt und gewichtet werden. Die erstmalige Berechnung des Modellkomitees kann einige Zeit in Anspruch nehmen (wenige Minuten). Ist das Modellkomitee nach der Startphase berechnet, ist die Adaption in der Explorationsphase viel weniger zeitintensiv, da die Optimierungen mit den in der Regel bereits sehr guten Ergebnissen aus der vorherigen Berechnungen initialisiert werden können. In der Explorationsphase werden nun weitere Designpunkte bestimmt, um das Modellkomitee iterativ zu verbessern. Dies kann entweder nach dem D-optimalen oder QbC Kriterium erfolgen (siehe Abschnitt 5.4). Die Optimierung dieser Kriterien kann mit den zur Verfügung stehenden Optimierungsrechnern nicht in Echtzeit realisiert werden, so dass die für einen Designpunkt zur Verfügung stehende Zeit in zwei Bereiche unterteilt wird (siehe Abbildung 6.2). Im ersten Zeitbereich wird das Modellkomitee adaptiert. Im

nachfolgenden Zeitbereich wird auf Grundlage des angepassten Modellkomitees die nächste optimale Eingangssequenz ermittelt.

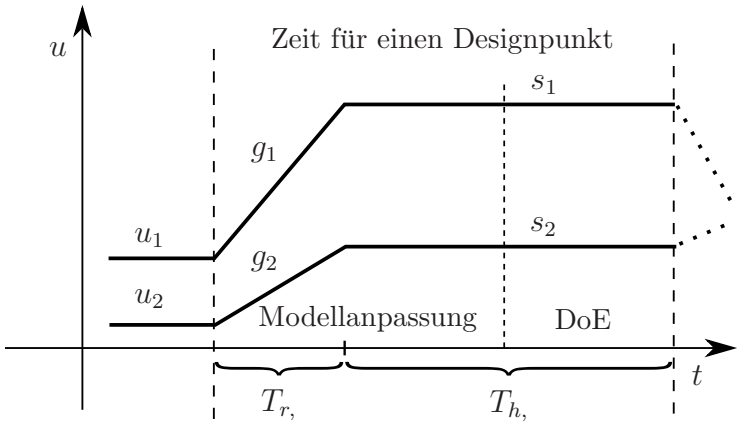


Abbildung 6.2: Eingang u im Zeitbereich: Designpunkt d mit Rampenzeit $T_{r,}$, Steigung g und Haltezeit $T_{h,}$. Unterteilung der Rechenzeit (fein strichlierte Linie) für die Modellanpassung und DoE Berechnung.

Mit den neu angefallenen Daten werden die Modelle des Komitees angepasst. Dies wird so lange wiederholt, bis die Modellgüte ein benutzerdefiniertes Kriterium unterschreitet. Die Modellgüte wird dabei an den mit jedem Designpunkt neu anfallenden Daten bestimmt, bevor diese zum Modelltraining verwendet werden. Somit dienen die neuen Daten zur Evaluierung der Generalisierungsfähigkeit des Modelkomitees, welche als Abbruchkriterium dient.

Zielt die Versuchsplanung auf eine gleichzeitige Optimierung des Systemverhaltens ab, sind zum Ende der Explorationsphase jene Bereiche im Parameterraum, bei denen die Zielgröße keine niedrigen Werte (Minimierungsproblem) oder hohe Werte (Maximierungsproblem) annimmt, von untergeordnetem Interesse. Dementsprechend muss die Gewichtung μ des Versuchsplanungskriteriums und Optimierungskriteriums in (5.29) angepasst werden. Um dies zu erreichen, wird der Parameter μ schrittweise reduziert, so dass bei den ersten 25% der Designpunkte der Explorationsphase $\mu = 1$ nach 75% der Designpunkte $\mu = 0.25$ gilt.

Optimierungsphase

Wird eine Optimierung des Systemverhaltens angestrebt, wird in der letzten Phase bei $\mu = 0$ nach Optima anhand des Modellkomitees gesucht und diese analog der Explorationsphase anhand der Messdaten validiert. Dadurch kann bereits im laufenden Betrieb die Güte der gefundenen Optima validiert werden. Sind die Optima bis auf einen festzulegenden Schwellwert genau bestimmt, endet die Optimierungsphase.

6.1.2 Verstellstrategie und Limitbehandlung

Die Verstellung der Eingänge wird mittels drei Komponenten realisiert. Zum einem die DoE-Komponente, welche das optimale Design unter Berücksichtigung des Limitmodells berechnet und die Rampen erzeugt, auf welchen die Designpunkte angefahren werden.

Zum anderen ein Limitmodell [48], welchem Eingänge übergeben werden, die zu einer Limitverletzung geführt haben. Als letzte Komponente kommt die Versteller-Komponente (Variationsversteller [47]) zum Einsatz, welcher der anzufahrende Designpunkt übergeben wird.

Mit Hilfe der DoE-Komponente berechnet der Variationsversteller die Rampe, welche die Designpunkte miteinander verbindet. Anschließend werden die Limitmodelle und Zielgrößenmodelle befragt, ob die Rampe zu einer Limitverletzung führen wird. Dabei werden die Limitmodelle hinsichtlich Beschränkungen des Eingangsraum und die Zielgrößenmodelle hinsichtlich der Verletzung von bekannten Beschränkungen der Zielgrößen befragt. Bei auftretenden Limitverletzungen weicht der Variationsversteller von der vorgegebenen Trajektorie ab, um sicher einen alternativen Designpunkte anfahren zu können.

Der gesamte Ablauf der Verstellung ist in Abbildung 6.3 dargestellt. Die einzelnen Aktionen der verschiedenen Komponenten werden im Folgenden beschrieben.

Bei Limitverletzungen sind im Variationsversteller folgende Strategien implementiert [47]:

- Ist 70 Prozent der vorgegebenen Trajektorie (Rampe 70%) zwischen vorherigem und anzufahrendem Designpunkt zurückgelegt, wenn die Limitverletzung auftritt (Limit Rampe), wird der letzte fahrbare Punkt als neuer Designpunkt angesehen (Rückzug, neuer Designpunkt) und über die Dauer der Haltezeit gehalten (Halte Designpunkt). Der neue Designpunkt wird anschließend der DoE-Komponente und die Eingangskombination bei der die Limitverletzung stattfand, an das Limitmodell übergeben (Update Limitmodell).
- Sind weniger als 70 Prozent zurückgelegt, wird über eine Rampe der Zentralpunkt angefahren (Rampe Zentralpunkt), wobei in dieser Phase weitere Limitverletzungen nicht berücksichtigt, aber an das Limitmodell übergeben werden. Der Zentralpunkt stellt dabei einen definierten Punkt dar, für den sichergestellt ist, dass keine Limitverletzungen auftreten. Vom Zentralpunkt aus wird erneut versucht, den Designpunkt anzufahren (Rampe Designpunkt). Tritt erneut eine Limitverletzung auf, wird unabhängig von der zurückgelegten Strecke der letzte fahrbare Punkt als neuer Designpunkt betrachtet. Alle Limitverletzungen werden an das Limitmodell übergeben sowie die zurückgelegte Trajektorie an die DoE-Komponente.

Nach Limitverletzungen wird immer das Limitmodell aktualisiert, bevor die DoE-Komponente nach dem nächsten Designpunkt befragt wird. Das aktualisierte Limitmodell wird von der DoE-Komponente in der Startphase für 2 Aktionen verwendet:

- Wenn einer oder mehrere der noch anzufahrenden Designpunkte das aktualisierte Limitmodell verletzen (Limit Design), wird für die verbliebenen Designpunkte unter Berücksichtigung der bereits angefahrenen Designpunkte, ein neues LHC oder D-optimales Design berechnet (Neues Design).
- Führt die Rampe, welche zum nächsten Designpunkt führt, zu einer Limitverletzung, wird durch Vertauschung der Reihenfolge der Designpunkte versucht einen anfahrbaren Designpunkt zu finden. Sind alle verbliebenen Designpunkte durchprobiert und jeder führt zu einer Limitverletzung (Max Komb.), wird der nächste Designpunkt über den Umweg des Zentralpunktes angefahren.

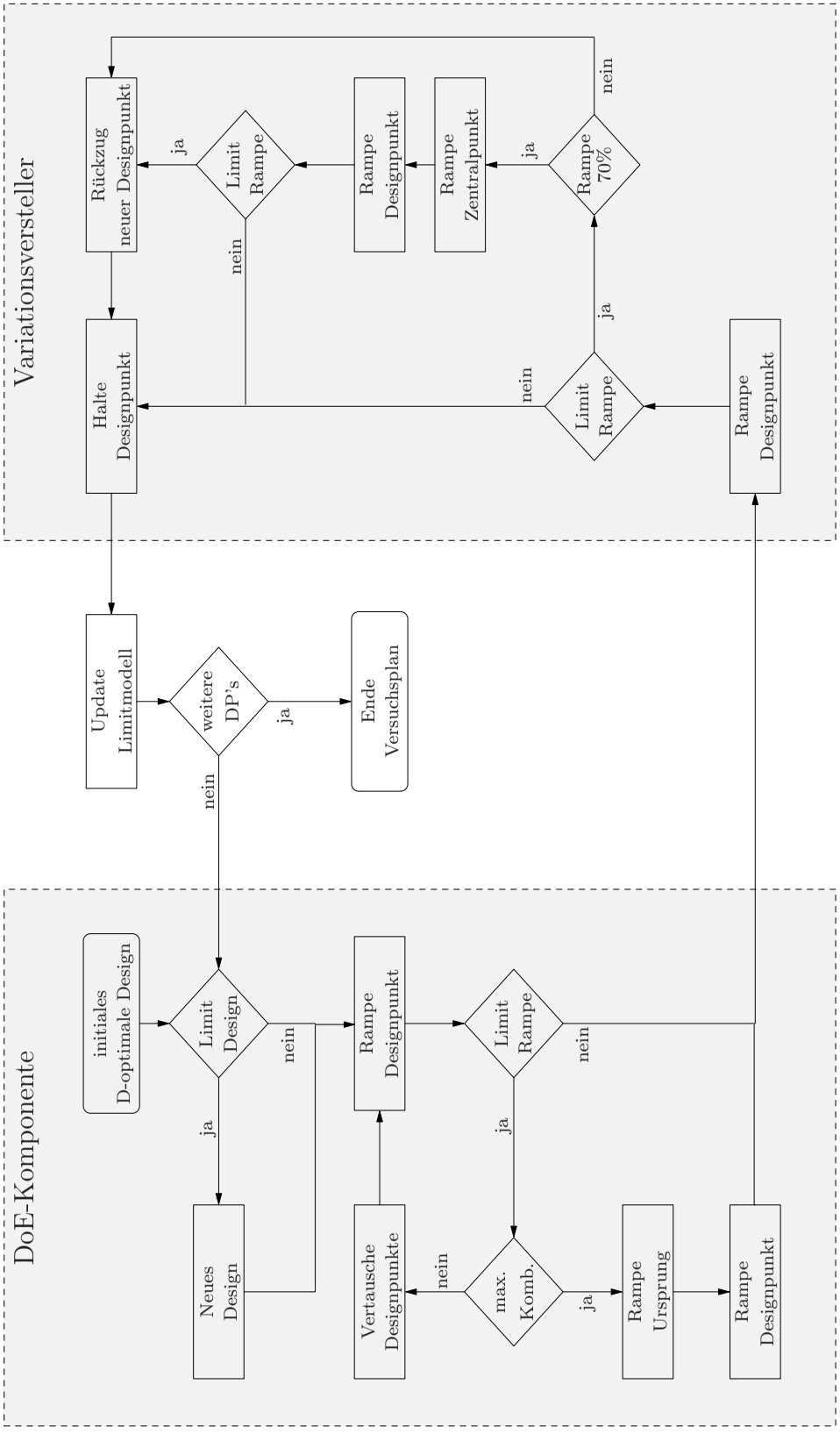


Abbildung 6.3: Ablauf der Erstellung des initialen Designs.

In der Explorationsphase respektive Optimierungsphase ist der Versuchsraum durch die Limitmodelle bereits hinreichend eingegrenzt, so dass die Rampenzeit verkürzt werden kann und die Limitmodelle als Nebenbedingung an das Optimierungskriterium (5.29) formuliert werden können.

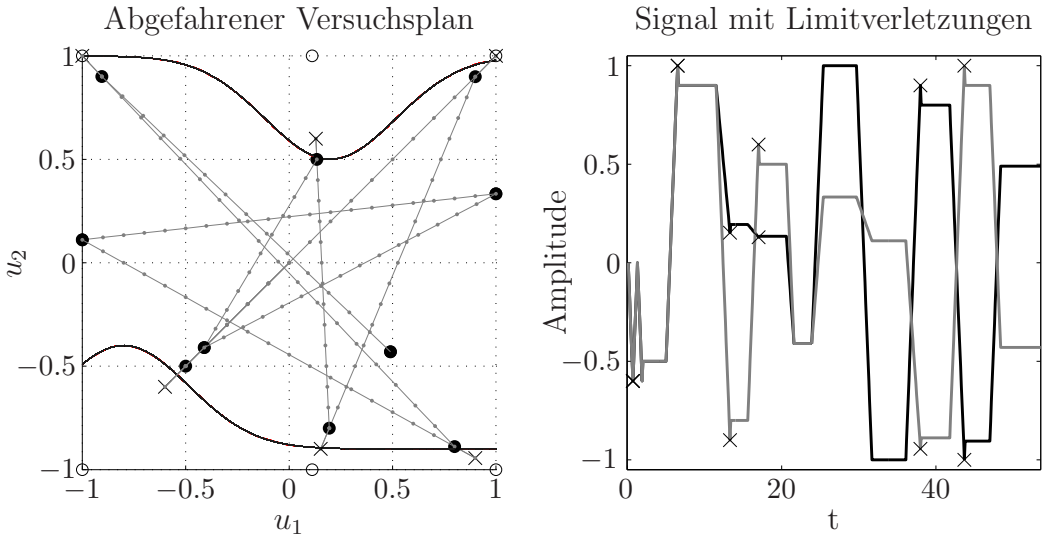


Abbildung 6.4: DoE in der Startphase.

In Abbildung 6.4 ist ein zweidimensionales Beispiel dargestellt. Die Kreise kennzeichnen das anfangs vorgesehene D-optimale Design, die Kreuze die Limitverletzungen und die schwarzen Punkte das am Ende verwendete modifizierte D-optimale Design. Ebenso ist der zeitliche Verlauf des Anregungssignals dargestellt, wobei die Kreuze die auftretenden Limitverletzungen kennzeichnen. Am Anfang der Anregung wird ein Designpunkt über den Umweg des Ursprungs angefahren. Das Simulationsergebnis basiert auf dem im Anhang beschriebenen Testsystem und in [48] beschriebene Limitmodell.

6.1.3 Totzeiten, Ausreißer und unterschiedliche Zielgrößendynamik

Die Messgrößen am Verbrennungsmotorprüfstand sind in der Regel mit Totzeiten behaftet und neben dem Messrauschen ist außerdem mit Messausreißern zu rechnen. Zudem werden meist zeitgleich mehrere Zielgrößen modelliert, wobei sich die Zeitkonstanten der betrachteten Zielgrößen um Größenordnungen unterscheiden können.

Totzeiten

Mit Totzeiten muss vor allem bei den Emissionen gerechnet werden, da zum einen die Übertragung des Abgases zu den Messgeräten als auch dessen Analyse zu Zeitverzögerungen führt. Da die Abgasanalysegeräte mit Pumpen zur Abgasansaugung ausgestattet sind, ist die Totzeit nicht vom Abgasdruck abhängig und es kann von einer betriebspunktunabhängigen Totzeit ausgegangen werden.

Zur Ermittlung der Totzeit wird ein einfacher Ansatz verfolgt. Im Anschluss an die Startphase wird nach jedem Designpunkt die Sprung- beziehungsweise Rampenantwort der Ziel-

größen mittels eines linearen dynamischen Modells 2. Ordnung approximiert und N_d verschiedene Totzeiten $T_d = [0\,t_s, \ldots, N_d\,t_s]$ getestet. Die Totzeit, welche für alle Designpunkte zum geringsten mittleren Approximationsfehler führt, wird dann verwendet, um die Eingangsdaten entsprechend zeitlich zu verschieben.

Ausreißer

Ausreißer lassen sich beispielsweise anhand der in [35] vorgeschlagenen Methoden erkennen. Um nach jedem Designpunkt eventuell auftretende Ausreißer zu unterdrücken, wird die quadratische Kostenfunktion (B.1) dahingehend angepasst, dass der quadratische Fehler eines jeden Modellausgangs zu jedem Zeitpunkt gewichtet wird

$$S_3(\theta) = \beta \left(\sum_{k=1}^{K} \sum_{i=1}^{N_O} (1 - \gamma_{i,k}) \left(y_{i,k} - f(\varphi, \theta)_{i,k} \right)^2 \right) + \alpha \, ||\theta||_2^2 \,. \tag{6.1}$$

Mittels $\gamma_{i,k} \in [0\ 1]$ ist es möglich, die i-te Komponente des Ausgangs $y_{i,k}$ zum Zeitpunkt k hinsichtlich seiner Wahrscheinlichkeit ein Ausreißer zu sein zu gewichten.

Dabei ist zu beachten, dass sich die Gewichtung der Ausreißer abhängig von der Konfiguration unterscheidet. Modelle in Prädiktor-Konfiguration gewichten den Trainingsfehler geringer zum Zeitpunkt des Auftretens des Ausreißers. Bei Modellen in Simulator-Konfiguration hingegen, dient der Ausreißer n-mal als Modelleingang, so dass alle n dem Ausreißer nachfolgenden Trainingsfehler ebenfalls geringer gewichtet werden müssen.

Unterschiedliche Zielgrößendynamik

Werden verschiedene Zielgrößen gleichzeitig modelliert, wird die Versuchsplanung hinsichtlich der langsamsten Zielgröße ausgelegt. Ist nun eine Zielgröße wesentlich schneller als die zu modellierende, befindet sich diese nach relativ kurzer Zeit bereits im eingeschwungenem Zustand, wodurch der Modellfehler im Stationärzustand gegenüber dem Modellfehler im Instationären übermäßig gewichtet wird. Um dies zu vermeiden, kann die Zielgrößenstabilität mittels der in [47] vorgeschlagenen Methoden erkannt und der Modellfehler (6.1) ab diesem Zeitpunkt mittels $\gamma_{i,k}$ entsprechend geringer gewichtet werden. Eine schematische Darstellung hierzu findet sich in Abbildung 6.5.

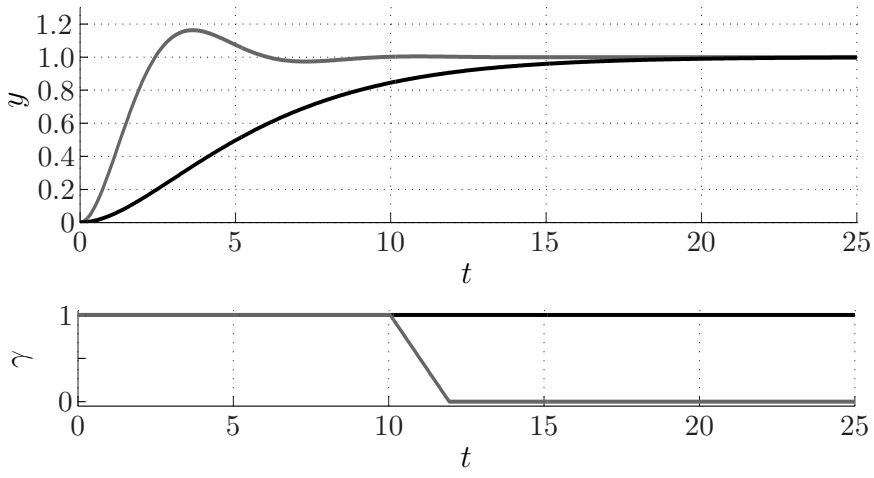


Abbildung 6.5: Gewichtung unterschiedlicher Dynamiken.

6.2 Evaluierung im Prüfstandsbetrieb

Anhand von zwei Beispielen wird in diesem Abschnitt die Eignung von Runge-Kutta Neuronalen Netzen zur Approximation dynamischer Zusammenhänge am Motorprüfstand gezeigt. Da kein Validierungsdatensatz verwendet wird, erfolgt die Optimierung unter Berücksichtigung der Stabilitätsnebenbedingungen an die Gewichte. Zur Berechnung des Anregungssignals werden die Methoden der Online-Versuchsplanung, welche in Kapitel 5 eingeführt werden, verwendet.

Im ersten Beispiel werden die Methoden der Online-Versuchsplanung anhand einer Kennfelddokumentation, welche den Zusammenhang zwischen Drehzahl / Last und Messgrößen darstellt, untersucht. In einem zweiten Beispiel wird ein RKNN zum Prüfstandsreglerentwurf verwendet.

6.2.1 Kennfelddokumentation

Für die Kennfelddokumentation wird ein aufgeladener 4-Zylinder Ottomotor verwendet, der im Teillastbereich angeregt wird. Als Eingänge dienen Motordrehzahl und der Ventilhub, welcher die Last vorgibt. Die Drehzahl wird im Bereich 1000-3000 U/min und der Ventilhub zwischen 15%-50% variiert. Die zu modellierenden Größen sind das resultierende Moment und die Abgastemperatur vor Turbolader (Temp. VT) und vor Katalysator (Temp. VK). Die Ermittlung des Zusammenhangs zwischen Drehzahl/Last und Messgrößen wird häufig zum Protokollieren des Applikationsfortschritts verwendet. Sollen dabei auch die stationären Abgastemperaturen erfasst werden, ist die Erstellung einer Kennfelddokumentation sehr zeitaufwändig. Eine Prädiktion der Stationärwerte von Abgastemperaturen beinhaltet somit ein großes Potential bei Kennfelddokumentationen Prüfstandszeit einzusparen.

Anhand dieses Beispiels soll die Online-Versuchsplanung evaluiert und die Eignung von RKNN zur Abschätzung der stationären Abgastemperaturwerte beurteilt werden. Als Eingangssignal dient das in Abschnitt 5.4.1 eingeführte parametrische Anregungssingal. Die Flanke des parametrischen Eingangssignals wird so gewählt, dass die maximale Rampendauer 2 Sekunden beträgt und die minimale Haltedauer wird auf $T_{h,\min} = 10s$ festgesetzt. Die Signaldauer beträgt 1350 Sekunden und die Verteilung der Designpunkte im Eingangsraum wird mittels Offline-Versuchsplänen (siehe Abschnitt 3.3) respektive Online-Versuchsplänen (siehe Abschnitt 5.4.3) festgelegt. Die Länge der Haltezeit stellt einen Kompromiss dar, da die Dynamik der Temperaturen wesentlich träger ist als die des Moments. Um optimale Ergebnisse zu erzielen, müsste zur Identifikation der Temperaturen längere Haltezeiten angesetzt werden.

Als Modell dient ein Modellkomitee bestehend aus vier RKNN mit 8 bis 11 Neuronen. Als Trainingsalgorithmus kommt der Levenberg-Marquardt Algorithmus 1 mit Bayes'scher Regularisierung zum Einsatz.

Die Offline-Versuchspläne werden über eine Zufallsverteilung (zufällig), Latin Hypercube Verteilung (LHC) und einer D-optimalen Verteilung (D-opt. offline), basierend auf einem Polynom 4. Grades, bestimmt. Bei der Online-Versuchsplanung kommt der sequentielle D-

Tabelle 6.1: NMSE: Validierungsdaten.

	zufällig	LHC	D-opt. (offline)	D-opt. (online)	QbC
Moment	0.036 ± 0.031	0.021 ± 0.016	0.017 ± 0.011	0.015 ± 0.002	0.017 ± 0.002
Temp. VK	0.321 ± 0.242	0.381 ± 0.278	0.261 ± 0.176	0.246 ± 0.154	0.239 ± 0.155
Temp. VT	0.208 ± 0.138	0.212 ± 0.163	0.187 ± 0.116	0.156 ± 0.104	0.173 ± 0.220

optimale Versuchsplan (D-opt. online) und die Query by Committee Methode (QbC) zum Einsatz. Jede Messung wird 5 Mal wiederholt und das Training des Modellkomitees erfolgt unabhängig vom eingesetzten Versuchsplan stets online. Jede Messung, welche nicht zum Training des Modellkomitees verwendet wird, dient bei der Evaluierung als Validierungsdatensatz.

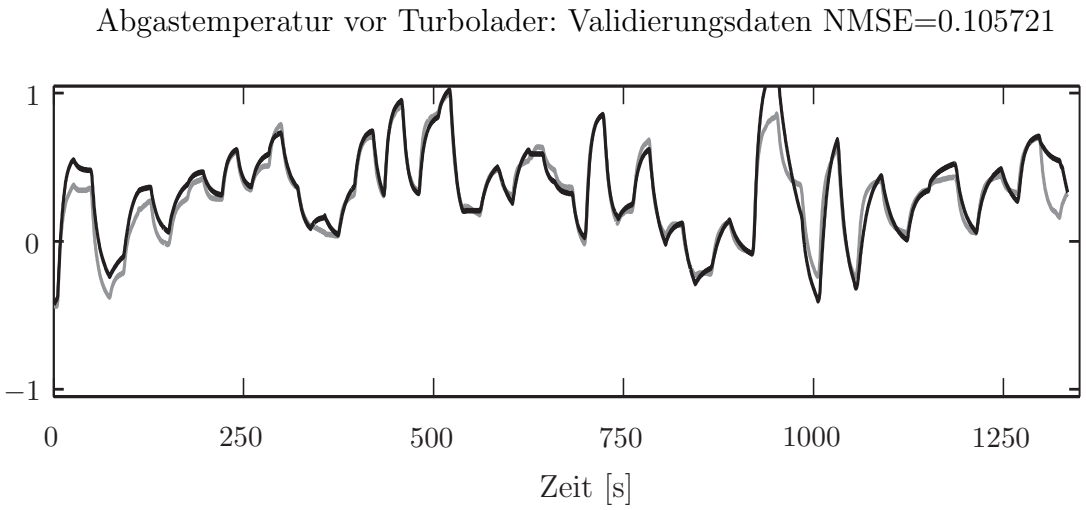


Abbildung 6.6: Gegenüberstellung System- (grau) und Modellkomiteeausgang (schwarz).

Die Ergebnisse für die Validierungsdatensätze mit Angabe des Mittelwerts und der Standardabweichung des NMSE finden sich in Tabelle 6.1. In Abbildung 6.6 wird beispielhaft der normierte zeitliche Verlauf der Temperatur vor Turbolader und der Modellausgang des Komitees gegenübergestellt, um eine Vorstellung von der Systemdynamik und der erzielten Modellgüte zu ermitteln.

Die Ergebnisse der Abschätzung der stationären Abgastemperaturen finden sich in Tabelle 6.2. Als Validierungsdaten dienen 40 Stationärmessungen, deren Drehzahl und Last Kombinationen über die LHC-Verteilung bestimmt werden. Das Fehlermaß ist der auf den Wertebereich der Stationärmessung normierte mittlere absolute Fehler (NMAE), welcher wiederum mit Mittelwert und Standardabweichung angegeben wird.

Wie aus den Ergebnissen ersichtlich, führen die Online-Versuchspläne zur besten Modellgüte. Dies trifft sowohl auf das instationäre als auch stationäre Modellverhalten zu. Das D-optimale Kriterium führt im Mittel zu besseren Ergebnissen als das QbC Kriterium. Der mittlere absolute Fehler der Abschätzung der Abgastemperatur beträgt bei den

Tabelle 6.2: NMAE: Stationärwertschätzung.

	zufällig	LHC	D-opt. (offline)	D-opt. (online)	QbC
Temp.VK	0.199 ± 0.124	0.184 ± 0.050	0.175 ± 0.066	0.150 ± 0.012	0.151 ± 0.025
Temp.VT	0.139 ± 0.025	0.168 ± 0.044	0.132 ± 0.017	0.131 ± 0.018	0.282 ± 0.193

D-optimalen Versuchsplänen um die 52°C (Temperatur vor Turbolader) beziehungsweise 60°C (Temperatur vor Katalysator). Die Stationärwerte schwanken zwischen 350°C und 750°C. Die Genauigkeit der Abschätzung der Stationärwerte lässt sich weiter steigern, indem anstatt der minimalen Haltedauer $T_{h,\min} = 10s$ einen höheren Wert ansetzt wird, wodurch die tieferen Frequenzen stärker anregt werden.

Die Ergebnisse unterstreichen die Anwendbarkeit der Versuchsplanung in der Praxis und zeigen die potentielle Zeitersparnis bei einer Kennfelddokumentation auf, wenn anstatt mehrere Minuten auf das Einschwingen der Temperaturen zu warten, der Stationärwert der Temperaturen abgeschätzt wird. Allerdings hängt die potentielle Zeitersparnis von der Anforderung an die Genauigkeit der Stationärwertschätzung ab, da nur mit ausreichend großer minimaler Haltezeit eine sehr hohe stationäre Modellgenauigkeit erzielt werden kann.

6.2.2 Prüfstandsreglerentwurf

Das zweite Beispiel besteht in einem linearisierenden Prüfstandsreglerentwurf basierend auf RKNN. Es werden die Vorteile der RKNN gegenüber ZDNN beim linearisierenden Reglerentwurf untersucht.

Um eine Ein-/Ausgangslinearisierung diskutieren zu können, wird zunächst ein eingangsaffines Modell erster Ordnung der Form

$$f_{MLP}(\varphi, u) = {w^l}^\top \varphi + {w_1^o}^\top \sigma(W_1^h \varphi) + {w_2^o}^\top \sigma(W_2^h \varphi) u \tag{6.2}$$

angesetzt. Dabei ist u der Eingang, welcher als Stellgröße dient und nicht im Regressorvektor φ enthalten ist. Beim ZDNN kann eine beliebige Ordnung angesetzt und die verzögerten Ein- und Ausgänge im Regressorvektor gesammelt werden. Wird nun den Zusammenhang zwischen Führungsgröße w_k und linearisierender Stellgröße u_k betrachtet

$$\begin{aligned} w_k &= {w^l}^\top \varphi_k + {w^o}_1^\top \sigma(W_1^h \varphi_k) + {w_2^o}^\top \sigma(W_2^h \varphi_k) u_k \\ \Rightarrow \quad u_k &= \frac{w_k - {w^l}^\top \varphi_k + {w_1^o}^\top \sigma(W_1^h \varphi_k)}{{w_2^o}^\top \sigma(W_2^h \varphi_k)}, \end{aligned} \tag{6.3}$$

erkennt man, dass für hohe Abtastraten der Term ${w_2^o}^\top \sigma(W_2^h \varphi_k)$ gegen 0 geht. Dies ist leicht ersichtlich, da bei ausreichender Modellgüte und hohen Abtastraten (bezogen auf die Systemdynamik) für jede beliebige Stellgröße u_k der Zusammenhang $f_{MLP}(\varphi_{k+1}, u_{k+1}) \approx f_{MLP}(\varphi_k, u_k)$ gültig sein muss. Somit ist der Nenner im linearisierenden Regelgesetz (6.3) abhängig von der Abtastzeit. Dies führt zu einer widersprüchlichen Zielsetzung bei der Wahl der Abtastzeit, da einerseits die Abtastrate möglichst hoch gewählt werden sollte,

um eine hohe Regelgüte zu gewährleisten, und andererseits eine zu hohe Abtastrate den Nenner im linearisierenden Regelgesetz (6.5) gegen Null gehen lässt, womit dieser extrem empfindlich auf Regelungsabweichungen reagiert. Dadurch kann es zu oszillierendem Regelverhalten kommen. Dies stellt ein allgemeines Problem beim linearisierenden Reglerentwurf basierend auf ZDNN dar [32].

Für den zeitkontinuierlichen Fall wird hingegen als Modell eine Integratorkette der Form

$$x^{(n)} = {w^l}^\top \varphi + {w^o}_1^\top \sigma(W_1^h \varphi) + w_2^{o\top} \sigma(W_2^h \varphi) u$$

angesetzt, wobei der Regressor nun die System- beziehungsweise Modellzustände (je nach Konfiguration) und alle Eingänge bis auf u enthält. Nun kann über den Eingang u dem System das gewünschte Einschwingverhalten aufprägt werden. Um dies zu erreichen, kann ein Führungsfilter verwendet und dem System beispielsweise ein PT2-Verhalten aufprägen werden. Dies wird erreicht, indem für einen gewünschten Stationärwert w_s mit $\overline{w}_1 = w$, $\overline{w}_2 = \dot{w}$ ein lineares Wunscheinschwingverhalten ansetzt wird

$$\begin{bmatrix} \dot{\overline{w}}_1 \\ \dot{\overline{w}}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{1}{T^2} & -\frac{2d}{T} \end{bmatrix} \overline{w} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w_s \quad w = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \overline{w} \tag{6.4}$$

und als linearisierenden Eingang

$$u = \frac{\dot{\overline{w}}_2 - {w^l}^\top \varphi + w_1^{o\top} \sigma(W_1^h \varphi)}{w_2^{o\top} \sigma(W_2^h \varphi)} \tag{6.5}$$

wählt. Wenn die Annahme einer Integratorkette zutrifft und die Modellierung exakt ist, gilt $\ddot{y} = \ddot{w}$ und somit wird das gewünschte Systemverhalten erzielt. Mit dem Dämpfungsfaktor d kann das Einschwingverhalten und mit T die Einschwinggeschwindigkeit festgelegt werden. Dabei muss bei der Wahl der Zeitkonstante T darauf geachtet werden, dass das zu linearisierende System in allen Bereichen schnell genug ist, um ein Einschwingverhalten mit Zeitkonstante T darstellen zu können.

Der Ansatz eines RKNN ermöglicht somit einen linearisierenden Reglerentwurf ohne sich die Probleme des zeitdiskreten Ansatzes einzuhandeln: Die Terme des RKNN sind unabhängig von der Abtastzeit und lediglich abhängig von der Systemgeschwindigkeit.

Nachdem das zu regelnde Motorverhalten Störungen unterworfen ist und die Modellierung des Motorverhaltens niemals exakt sein wird, ist eine linearisierende Vorsteuerung, welche dem System die gewünschte Dynamik aufprägt, nicht ausreichend. Daher wird stattdessen das Fehlerabklingverhalten festgelegt. Für ein System 2. Ordnung setzt man über die Linearisierung (6.5) beispielsweise

$$\ddot{y} = \ddot{w} + a(\dot{w} - \dot{y}) + b(w - y) + c \int (w - y) \, \mathrm{d}t$$

an und erhält mit $\int e = y - w \, \mathrm{d}t$ das über die Koeffizienten a, b, c festlegbare Fehlerabklingverhalten

$$e^{(3)} = -a\ddot{e} - b\dot{e} - ce.$$

Nachdem die zeitliche Ableitung des Systemausgangs in der Regel unbekannt ist, können

die Modellzustände zur Berechnung von (6.5) verwendet werden oder man bedient sich Ableitungsschätzverfahren wie beispielsweise die in [94] vorgestellten echtzeitfähigen Methoden.

Der linearisierende Reglerentwurf wird anhand einer Momentenregelung für einen aufgeladenen 8 Zylinder Ottomotor evaluiert. Das Ziel ist ein möglichst schneller Regler, welcher in der gesamten Betriebsebene einsetzbar ist und das Moment über die Drossenklappenstellung einregelt. Um einen Datensatz mit möglichst hohem Informationsgehalt für den Modellbildungsschritt zu erhalten, wird ein D-optimales parametrisches Anregungssignals mit minimalen Haltedauer $T_{h,\min} = 8s$, Rampendauer $T_r = 2s$ und Signallänger $T = 360s$ erzeugt. Die Abtastzeit für die Identifikation beträgt $t_s = 0.1s$ und für die Regelung $t_s = 0.01$. Um eine Echtzeitanbindung mit einer Abtastzeit von $t_s = 0.01s$ sicherzustellen, wird der Regler in das Prüfstandsautomatisierungssystem integriert. Als Modell wird ein eingangsaffines RKNN (6.2) mit 10 Neuronen und einer RK-Schrittweite von $h_{rk} = 0.1s$ trainiert (es wird nur jeder 10. Messwert für das Training verwendet).

Zur Validierung der erzielten Regelgüte werden Momentensprünge bei gleich bleibenden Drehzahlen, Drehzahlsprünge bei gleich bleibenden Momenten sowie gleichzeitig durchgeführte rampenförmige Drehzahl- und Momentenänderungen vorgegeben. Um ein vom Motor darstellbares Einschwingverhalten vorgeben zu können, wird durch Lastsprünge bei unterschiedlichen Drehzahlen die langsamste Zeitkonstante für das Einschwingverhalten des Moments bestimmt und entsprechend für den Führungsfilter (6.4) verwendet. In der Abbildung 6.7 finden sich normierte Darstellungen des erzielten Regelverhaltens. Die obere Grafik zeigt das vorgegebene Sollmoment (graue Linie) und das tatsächlich eingeregelte Moment (schwarze Linie). In der unteren Grafik ist der Drehzahlverlauf dargestellt. Das Moment ist dabei bei -1 knapp über 0 Nm und bei 1 bei Volllast, die Drehzahl deckt den gesamten Drehzahlbereich ab der Leerlaufdrehzahl ab.

Wie aus den Ergebnissen ersichtlich, kann der Regler den Vorgaben sehr gut folgen und auch Störungen in Form von Drehzahlsprüngen schnell ausregeln. Bei den niedrigen Drehzahlen kann teilweise das geforderte Moment nicht erreicht werden ($t \in [40, 50]$) und teilweise kommt es im Zeitbereich $t \in [100, 130]$ zu einem Überschwingen beim Einschwingvorgang. Dies ist nicht durch den Führungsfilter vorgegeben und wird durch Modellierungsfehler verursacht. Diese resultieren aus dem Ansatz eines eingangsaffinen Modells 1. Ordnung, obwohl der Zusammenhang zwischen Drehzahl/Last und Moment einem Verhalten 2. Ordnung entspricht und nicht zwangsläufig eingangsaffin ist. Der Modellansatz 1. Ordnung wird motiviert durch die Tatsache, dass ein Modell 2. Ordnung eine numerische Ableitung des Moments oder einen Zustandsbeobachter benötigt, wodurch zusätzlicher Parametrieraufwand erforderlich wird. Die erzielte Regelgüte rechtfertigt den einfacheren Modellansatz, um den Prüfstandsreglerentwurf in der Praxis so einfach wie möglich zu halten.

Die erzielten Ergebnisse zeigen die Eignung des linearisierenden Reglerentwurfs basierend auf RKNN. Der linearisierende Regler kann auf Basis von Prüfstandsmessungen einfach und automatisiert erstellt und über die Reglerkoeffizienten kann komfortabel das Fehlerabklingverhalten vorgegeben werden. Weiterführende Diskussionen und ein Vergleich mit kommerziellen Reglern findet sich in [56].

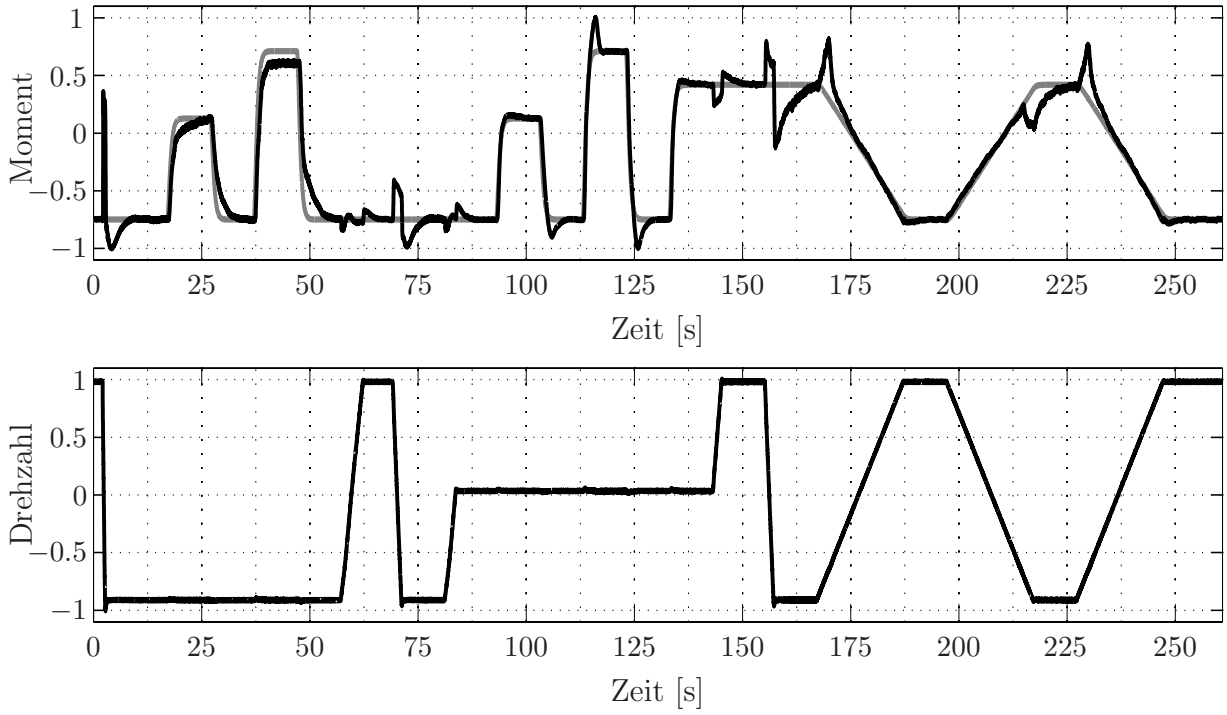


Abbildung 6.7: Regelverhalten (graue Linie: Sollmoment, schwarze Linien: Drehmoment).

6.3 Einschränkungen und Ausblick

Die in dieser Arbeit entwickelten Methoden der Online-Versuchsplanung und -Identifikation liefern unter geeigneten Rahmenbedingungen gute Ergebnisse. Allerdings unterliegt die Anwendbarkeit gewissen Einschränkungen. So bereiten die in den Steuergeräten hinterlegten Bauteilschutzfunktionen erhebliche Probleme. Der Bauteilschutz überwacht unter anderem die Temperaturen im Abgassystem und verhindert durch Veränderung der Verbrennungsbedingungen zu hohe Temperaturen. Die Regelwerke des Bauteilschutz sind oftmals hysteresebehaftet und greifen durch Veränderung des Luft-Gasgemischverhältnisses direkt in die Verbrennung ein, wodurch die Rahmenbedingungen grundlegend verändert werden. Diese im Motorsteuergerät hinterlegte temperaturabhängige Funktionalität macht die Identifikation der dynamischen Zusammenhänge im Übergangsbereich zwischen aktiviertem und deaktiviertem Bauteilschutz in vertretbaren Zeitaufwand äußerst schwierig. Zum einen muss dieser Bereich stark angeregt werden, um die Hystereseeffekte und Unstetigkeiten zu erfassen, zum anderen kann die genaue Parametrierung sich im Laufe des Motorprojektes ändern, wodurch eine neuerliche Identifikation des dynamischen Verhaltens nötig wird. Um unabhängig von den in den Steuergeräten hinterlegten Funktionalitäten ein Modell zu erstellen, müsste der Eingangsraum massiv erweitert werden und alle Stellmöglichkeiten, welche durch das Steuergerät beeinflusst werden, als Modelleingänge nutzen. Dies führt zu einem 8-10 dimensionalen Eingangsraum, wodurch die Identifikation in einem vertretbaren Zeitraum erheblich erschwert wird.

Zur Vermeidung dieser Schwierigkeiten würde sich ein physikalisches (Teil-)Modell unter Berücksichtigung der Steuergerätebedatung anbieten. Anstelle des RKNN könnten (teil-)

physikalische Modelle zur modellbasierten Optimierung verwendet oder das Wissen um die Steuergerätebedatung in das Modell integriert werden. Die vorgestellten Verfahren der FIM-basierten Kriterien der Versuchsplanung lassen sich hierbei nutzen, um einen möglichst informativen Datensatz zur Identifikation der Parameter des physikalischen Modells zu erzeugen. Die Verwendung eines physikalischen Modells würde es zudem erlauben, direkt mit der Online-Versuchsplanung die Identifikation des dynamischen Verhaltens zu beginnen, da die initiale Modellbildungsphase durch die Verwendung des physikalischen Modells unnötig ist. Ein weiterer Vorteil liegt darin, dass die aufwändige Sicherstellung der Stabilität dynamischer Black-Box Modelle entfällt.

Da die physikalische Modellierung des gesamten Motorverhaltens aufgrund der komplexen chemischen Reaktionen bei der Verbrennung nicht möglich ist, bietet sich die Kombination von physikalischen Teilmodellen mit dynamischen Black-Box Modellen unter Berücksichtigung der Steuergerätebedatung an. Um die hier vorgestellten Verfahren bei einem solchen Ansatz nützen zu können, müssen allerdings die Ein-/Ausgänge der zu modellierenden Teilstrecke messbar sein.

7 Zusammenfassung und Ausblick

In dieser Arbeit werden Verfahren entwickelt, welche eine einfache und robuste Identifikation dynamischer Zusammenhänge am Motorprüfstand ermöglichen.

Der vorgestellte Modellbildungsalgorithmus erlaubt erstmals die Identifikation ein-/zustandsstabiler zeitkontinuierlicher Zustandsraummodelle mittels Runge-Kutta Neuronaler Netze in Simulator-Konfiguration anhand von zeitdiskreten Messungen. Es werden neue Lernverfahren für das Runge-Kutta Neuronale Netz (RKNN) entwickelt und Nebenbedingungen an die Gewichte desselben formuliert, um die Eingangs-/Zustandsstabilität sicherzustellen. Um dem Bias-Varianz Dilemma zu begegnen, erfolgt die Parameterschätzung unter Verwendung der Bayes'schen Regularisierung sowie automatischen Adaption der Neuronenzahl und es werden mehrere Modelle in einem Modellkomitee verwendet, deren Gewichtung basierend auf einem Bayes'schen Ansatz bestimmt werden. Die Modellbildung ist dabei, wie anhand von Simulationen gezeigt, robust gegenüber äußeren Störeinflüssen wie Messrauschen und schwankender Abtastzeit, sowie gegenüber der Wahl der Modellparameter wie Modellordnung und Anzahl der Neuronen. Zudem ist das RKNN im Gegensatz zu zeitdiskreten Modellen robust gegenüber der Wahl der Abtastzeit.

Die Qualität der Modelle wird dabei durch den Informationsgehalt der Trainingsdatensätze begrenzt. Um möglichst viel Information über das zu identifizierende System zu erhalten, werden neue Verfahren der modellbasierten Online-Versuchsplanung basierend auf dem amplitudenmodulierten pseudo-binären Rauschsignal (APRBS) entwickelt. Die Online-Versuchsplanung basiert dabei auf Kriterien, welche es erlauben die Versuchsplanung hinsichtlich des späteren Einsatzzwecks zu optimieren. So kann beispielsweise die gewünschte Qualität der Modelle hinsichtlich des Stationärverhaltens oder im Bereich der Optima im Stationärverhalten über die Versuchsplanung sichergestellt werden. Diese als untere Schranken an das Kriterium definierten Werte, dienen als Abbruchkriterium für die Versuchsplanung. Dadurch wird die Anzahl der Trainingsdaten in Abhängigkeit der gestellten Anforderungen an die Modellqualität und der Komplexität der Aufgabenstellung selbständig von der Versuchsplanung bestimmt. Die Kriterien der Versuchsplanung ermöglichen zudem die Berücksichtigung von Einschränkungen an den Ein-/Ausgangsraum, wodurch ein sicherer Prüfstandsbetrieb möglich wird.

Die vorgestellten Verfahren der Modellbildung und Versuchsplanung werden anhand von umfangreichen Simulationen und Prüfstandsmessungen evaluiert.

Ausblick und offene Problemstellungen

In dieser Arbeit sind die Grundlagen für die Online-Identifikation dynamischer Modelle und die modellbasierte Online-Versuchsplanung für dynamische Modelle am Motorenprüfstand gelegt worden. Für bestimmte Applikationsaufgaben sind die hier vorgestellten Verfahren nicht direkt einsetzbar:

Versuchsplanung Die in dieser Arbeit entwickelten Verfahren eignen sich zur Identifikation stabiler Systeme. Instabile Systeme, die nur in geregelter Form betrieben werden, können mit den vorgestellten Verfahren der Versuchsplanung nicht optimal angeregt werden. Die modellbasierte Online-Versuchsplanung im geschlossenen Regelkreis muss in weiterführenden Arbeiten untersucht werden.

In einigen Fällen sind physikalische Teilstreckenmodelle mit ausreichender Genauigkeit vorhanden (zum Beispiel Abgastemperaturmodelle), die über verschiedene Parameter an die Messdaten angepasst werden können. Solche teil-physikalische Modelle mit Adaptionsparametern nennt man auch Grey-Box Modelle, da die Struktur bekannt ist und nur einige Parameter zu bestimmen sind. Aufgrund der komplexen physikalischen Modelle im Prüfstandsumfeld ist es allerdings in der Regel nicht möglich, den Gradienten der physikalischen Modelle analytisch zu bestimmen. Daher müssen die vorgestellten Methoden der Versuchsplanung bei der Verwendung dieser Grey-Box Modelle angepasst werden.

Modellbildung Bestimmte Aufgaben, welche dynamische Black-Box Modelle verwenden, benötigen die Abschätzung der Modellungenauigkeit. Die hier vorgestellten Methoden der Modellbildung lassen keine direkten Aussagen über die Modellunsicherheit zu. Die Modellunsicherheit kann nur durch die Abweichungen der Einzelmodelle im Modellkomitee oder die Varianz der Modelparameter abgeschätzt werden. Sind beispielsweise bekannte Schranken an die Modellgenauigkeit erforderlich, sind die hier vorgestellten Modellbildungsverfahren nur bedingt geeignet.

Wissenschaftliche Zusammenfassung

Die Arbeit deckt mit der Versuchsplanung und Modellbildung zwei große Gebiete der Identifikation dynamischer Systeme ab. Zu beiden Gebieten werden Beiträge geleistet:

1. Für das RKNN wird ein Trainingsalgorithmus hergeleitet, welcher das Training in Simulator-Konfiguration mit Standardoptimierungsverfahren erlaubt [15].
2. Es werden Bedingungen formuliert, welche die Eingangs-/Zustandsstabilität des RKNN für gegebene Abtastzeiten sicherstellen, wenn das zugrundeliegende zeitkontinuierliche Neuronale Netz einer bestimmten Lyapunov-Funktion genügt [16].
3. Das RKNN wird erstmals mit einem zeitkontinuierlichen MLP-Netz konstruiert und Bedingungen an die Gewichte des MLP hergeleitet, welche der genannten Lyapunov-Funktion genügen und somit die Stabilität des RKNN sicherstellen [17].

4. Ein empirischer Vergleich zwischen dem RKNN und zeitdiskreten MLP-Netzen wird durchgeführt und die erzielte Modellgüte (Instationär- und Stationärverhalten) in Abhängigkeit der Abtastzeit, dem Signal-zu-Rauschverhältnis, dem Trainingsdatenumfang, der Modellordnung und schwankender Abtastrate untersucht. Dabei kann gezeigt werden, dass das RKNN dem zeitdiskreten MLP-Netz bei allen Untersuchungen hinsichtlich der Approximationsgüte überlegen ist.

5. Es wird eine modellbasierte Online-Versuchsplanung zur Identifikation dynamischer Systeme vorgestellt, welche die durch das Modell abgebildete Systemdynamik berücksichtigt und die Online-Versuchsplanung als modellprädiktives Regelungsproblem unter Berücksichtigung von Limitmodellen formuliert [23], [19].

6. Die modellbasierte Online-Versuchsplanung als modellprädiktives Regelungsproblem wird für verschiedene Optimierungskriterien, welche das Query by Committee Kriterium und die Maximierung der Fisher-Information beinhalten, simulativ und anhand von Prüfstandsmessungen evaluiert und mit den bestehenden Kriterien der Offline-Versuchsplanung verglichen. Die Simulationsergebnisse und Prüfstandsmessungen zeigen, dass die modellbasierte Online-Versuchsplanung zu einer wesentlich besseren Modellgüte führt, als die weit verbreitete Offline-Versuchsplanung.

7. Es werden Kriterien für die modellbasierte Online-Versuchsplanung eingeführt, welche es erlauben über die Versuchsplanung die Modelle hinsichtlich ihres späteren Einsatzzwecks zu optimieren [19].

Die wissenschaftlichen Neuerungen bestehen somit einerseits im RKNN in Simulatorkonfiguration, für welches ein Trainingsalgorithmus und Bedingungen für die Eingangs-/Zustandsstabilität hergeleitet werden, und andererseits in der Einführung neuer Methoden der Versuchsplanung für dynamische Systeme, welche Optimalitätskriterien hinsichtlich dem Informationsgehalt der Messdaten genügen.

Zusammenfassung aus Sicht der Anwendung

Mit den in der Arbeit erzielten Ergebnissen wird das für statische Modelle in der Praxis bewährte Verfahren der Online-Versuchsplanung und Systemidentifikation auch für dynamische Modelle möglich. Dadurch werden in der Applikation neue Möglichkeiten eröffnet:

1. Die Modellierung von dynamischen Zusammenhängen am Motorprüfstand ist erstmals ohne Systemwissen möglich, da das Modellbildungsverfahren äußerst robust gegenüber der Parametrierung ist.

2. Eine modellbasierte Optimierung des dynamischen Motorverhaltens kann anhand der dynamischen Modelle realisiert werden [20], [22].

3. Der Prüfstandsreglerentwurf kann automatisiert auf Grundlage von Messdaten identifizierten dynamischen Modellen erfolgen [18].

4. Dynamische Modelle ermöglichen die Prädiktion der Stationärwerte, wodurch die zeitaufwändige Stationärmessung vermieden werden kann [22].

5. Dynamische Modelle ermöglichen zudem die Prädiktion von Limitverletzungen durch träge Messgrößen, wie beispielsweise Abgastemperaturen, wodurch Prüfstands(not)abschaltungen vermieden werden können.

Die Modellierung dynamischer Zusammenhänge am Verbrennungsmotor ermöglicht neben der direkten Verwendung der dynamischen Modelle (wie zum Beispiel Reglerentwurf oder Optimierung des dynamischen Verhaltens) auch Verbesserungen bei Applikationsaufgaben mit Stationärmessungen. Durch die genannten Anwendungen der Stationärwert- und Limitprädiktion kann bei vielen Applikationsaufgaben durch die Verwendung von dynamischen Modellen ein schnellerer und sicherer Messprozess realisiert werden, wodurch teure Prüfstandszeit gespart und unnötige Prüfstandsabschaltungen vermieden werden können.

A Wahrscheinlichkeitstheorie

Die Wahrscheinlichkeitstheorie gibt Wahrscheinlichkeitsverteilungen für Variablen an, das heißt wie wahrscheinlich es ist, eine Zufallsvariable beziehungsweise -variablenvektor (in einem Intervall) zu beobachten.

Die Wahrscheinlichkeit eine Variable x zu beobachten wird dabei mit $p(x)$ angegeben. Hängt die Wahrscheinlichkeit von x von einem Parameter θ ab, wird von *bedingter Wahrscheinlichkeit* $p(x|\theta)$ gesprochen. Die Wahrscheinlichkeit mehrere Variablen $p(\theta_1, \ldots, \theta_n) = p(\theta)$ gemeinsam zu beobachten, wird als *gemeinsame Wahrscheinlichkeit* bezeichnet. Diese Wahrscheinlichkeit lässt sich auch über die Produktregel ausdrücken [6]

$$p(\theta) = p(\theta_1)p(\theta_2|\theta_1)\ldots p(\theta_n|\theta_{n-1}, \ldots, \theta_1),$$

wobei $p(\theta_n|\theta_{n-1}, \ldots, \theta_1)$ die Wahrscheinlichkeit ist, θ_n zu beobachten, wenn $\theta_1, \ldots, \theta_{n-1}$ gegeben ist. Sind die beobachteten Variablen unabhängig, gilt entsprechend $p(\theta) = \prod_{i=1}^{n} p(\theta_i)$. Soll die Verteilung von Zufallsvariablen beschrieben werden, wird die im folgenden beschriebene Wahrscheinlichkeitsverteilung verwendet.

A.1 Wahrscheinlichkeitsverteilungen

Bei stetigen Verteilungen lassen sich die Wahrscheinlichkeitsverteilungen als Integral über die Wahrscheinlichkeitsdichtefunktion darstellen. So wird die Wahrscheinlichkeit, dass die Zufallsvariable x im Bereich $x \in [a, b]$ liegt über die Wahrscheinlichkeitsdichtefunktion $p(x)$ beschrieben [6]

$$p(x \in [a, b]) = \int_a^b p(x)\,\mathrm{d}x.$$

A.1.1 Wahrscheinlichkeitsdichtefunktion

Die Wahrscheinlichkeitsdichtefunktion erfüllt stets

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1 \quad p(x) \geq 0,$$

da die Wahrscheinlichkeit immer positiv ist und maximal den Wert 1 annehmen kann. Die bekannteste und einzig in dieser Arbeit verwendete Wahrscheinlichkeitsdichtefunktion ist die Gauß-Funktion, die Wahrscheinlichkeitsdichtefunktion der Normalverteilung. Eine

stetige Zufallsvariable x mit der Gauß-Funktion $p(x) : \mathbb{R} \to \mathbb{R}_{>0}$

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right). \tag{A.1}$$

heißt (μ, σ^2)-normalverteilt, auch $x \sim \mathcal{N}(x|\mu, \sigma^2)$ geschrieben, wobei μ der Erwartungswert und σ die Standardabweichung sind [6].

Die Wahrscheinlichkeitsdichtefunktion einer mehrdimensionalen beziehungsweise multivariaten Normalverteilung mit $x \in \mathbb{R}^{N_p}$ Variablen ist entsprechend gegeben durch [6]

$$p(x, \theta) = \frac{1}{(2\pi)^{N_p/2} \left(\det(\Sigma)\right)^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \tag{A.2}$$

und $x \sim \mathcal{N}(x|\mu, \Sigma)$. Dabei bezeichnet Σ die Kovarianzmatrix der multivariaten Normalverteilung.

A.1.2 Likelihood-Funktion

Hängen die beobachteten Daten von einem Set von Parametern ab, wird die Funktion, welche die Wahrscheinlichkeit des Parametersets bei gegebenen beobachteten Daten beschreibt, als Likelihood-Funktion bezeichnet. Bei einem gegebenen Set an beobachteten Daten $\mathcal{D}$ und Parametern θ, kann die bedingte Wahrscheinlichkeit $p(\mathcal{D}|\theta)$, das heißt die Wahrscheinlichkeit $\mathcal{D}$ bei gegebenen Parametern θ zu beobachten, als Likelihood-Funktion $L(\theta|\mathcal{D}) = p(\mathcal{D}|\theta)$ interpretiert werden. Wird $f(\mathcal{D}|\theta)$ interpretiert als Funktion von $\mathcal{D}$ mit festem θ spricht man von einer Wahrscheinlichkeitsdichtefunktion und wenn $f(\mathcal{D}|\theta)$ interpretiert wird als Funktion von θ mit festem $\mathcal{D}$ entsprechend von einer Likelihood-Funktion. Die Likelihood-Funktion erlaubt bei einem gegebenen Datensatz den wahrscheinlichsten zugehörigen Parametervektor zu finden. Sind die Beobachtungen unabhängig voneinander ergibt sich für einen Datensatz $\mathcal{D} = \{x_1, \ldots, x_n\}$ entsprechend der Produktregel folgende Likelihood-Funktion

$$L(\theta|\mathcal{D}) = \prod_{i=1}^{n} p(x_1|\theta).$$

A.2 Fisher-Information

Die Fisher-Information kann für eine Familie von Wahrscheinlichkeitsdichten definiert werden und liefert Aussagen über die bestmögliche Qualität von Parameterschätzungen eines Modells.

Für einen erwartungstreuen Schätzer $\hat{\theta}(x)$ gilt:

$$\mathrm{E}\left[\hat{\theta}(x) - \theta\right] = \int \left[\hat{\theta}(x) - \theta\right] p(x|\theta)\,\mathrm{d}x = 0. \tag{A.3}$$

Die Likelihood-Funktion $L(\theta|x) = p(x|\theta)$ beschreibt die Wahrscheinlichkeit, dass für ein gegebenes θ die Zufallsvariable x beobachtet wird. Wenn $L(\theta|x)$ einen „scharfen Peak“ hat, ist es einfach aus den gegebenen Daten den „korrekten“ Parameter θ abzulesen; das heißt die Daten beinhalten viel Information über θ. Wenn die Likelihood-Funktion flach und ausladend ist, benötigt man viele Daten, um den Parameter θ zu schätzen (die Daten beinhalten wenig Information über θ) [44].

Durch Differenzieren von (A.3) ergibt sich

$$\begin{aligned}\frac{\partial}{\partial\theta}\int\Big[\hat{\theta}(x)-\theta\Big]p(x|\theta)\,\mathrm{d}x &= \frac{\partial}{\partial\theta}\int\hat{\theta}(x)p(x|\theta)\,\mathrm{d}x - \frac{\partial}{\partial\theta}\int\theta p(x|\theta)\,\mathrm{d}x\\ &= \int\hat{\theta}(x)\frac{\partial}{\partial\theta}p(x|\theta)\,\mathrm{d}x - \int\left[p(x|\theta)+\theta\frac{\partial}{\partial\theta}p(x|\theta)\right]\mathrm{d}x\\ &= \int\left(\hat{\theta}(x)-\theta\right)\frac{\partial}{\partial\theta}p(x|\theta)\,\mathrm{d}x - \int p(x|\theta)\,\mathrm{d}x.\end{aligned}$$

Da die Likelihood-Funktion eine Wahrscheinlichkeitsverteilung ist, gilt

$$\int p(x|\theta)\,\mathrm{d}x = 1$$

und durch den Zusammenhang $(\ln x)' = \frac{1}{x}$ gilt

$$\frac{\partial}{\partial\theta}p(x|\theta) = p(x|\theta)\frac{\partial}{\partial\theta}\ln p(x|\theta).$$

Dadurch ergibt sich der Zusammenhang

$$\int(\hat{\theta}(x)-\theta)p(x|\theta)\frac{\partial}{\partial\theta}\ln p(x|\theta)\,\mathrm{d}x = 1.$$

Durch Faktorisierung des Integranten erhält man

$$\int\left((\hat{\theta}(x)-\theta)\sqrt{p(x|\theta)}\right)\left(\sqrt{p(x|\theta)}\frac{\partial}{\partial\theta}\ln p(x|\theta)\right)\mathrm{d}x = 1.$$

Wenn man die Gleichung quadriert und die Cauchy-Schwarz Ungleichung anwendet, ergibt sich

$$\left[\int\left(\hat{\theta}(x)-\theta\right)^2 p(x|\theta)\,\mathrm{d}x\right]\cdot\left[\int\left(\frac{\partial}{\partial\theta}\ln p(x|\theta)\right)^2 p(x|\theta)\,\mathrm{d}x\right]\geq 1.$$

Der rechte Teil dieser Ungleichung ist die Fisher-Information

$$\mathcal{I}(\theta) = \int\left(\frac{\partial\ln p(x|\theta)}{\partial\theta}\right)^2 p(x|\theta)\;\mathrm{d}x.$$

Der linke Teil ist der erwartete quadratische Fehler des Schätzers θ, da

$$\mathrm{E}\left[\left(\hat{\theta}(x)-\theta\right)^2\right] = \int\left(\hat{\theta}-\theta\right)^2 p(x|\theta)\,dx.$$

Diese Ungleichung besagt, dass die Präzision mit welcher θ abgeschätzt werden kann, durch die Fisher-Information beschränkt ist

$$\mathrm{Var}\left[\hat{\theta}(x)\right] \geq 1/\mathcal{I}(\theta).$$

Die Unsicherheit in den Parametern ist also gekennzeichnet durch die Inverse der Fisher-Information.

Bei einem Parametervektor mit $\theta = \left[\theta_1, \theta_2, \cdots, \theta_{N_p}\right]^\top$ ist die Fisher-Information entsprechend eine $N_p \times N_p$ Matrix

$$\mathcal{I}(\theta) = \mathrm{E}\left[\frac{\partial}{\partial\theta}\ln p(x;\theta)\frac{\partial}{\partial\theta}\ln p(x;\theta)^T\right].$$

Weist der Zufallsvektor x eine multivariate Normalverteilung auf $y \sim \mathcal{N}(\mu(\theta), \Sigma(\theta))$ und ist der Mittelwert $\mu(\theta)$ eine bekannte Funktion des unbekannten Parametervektors θ, ist die Fisher-Informationsmatrix gegeben durch [44]

$$\mathcal{I}(\theta) = \frac{\partial}{\partial\theta}\mu^T(\theta)\Sigma^{-1}\frac{\partial}{\partial\theta}\mu(\theta) + \frac{1}{2}\,\mathrm{tr}\left(\Sigma^{-1}\frac{\partial\Sigma}{\partial\theta}\Sigma^{-1}\frac{\partial\Sigma}{\partial\theta}\right). \tag{A.4}$$

Unter der Annahme von unkorreliertem Gauß'schen Rauschen $\Sigma(\theta) = \sigma^2 I$ folgt somit

$$\mathcal{I}(\theta) = \frac{1}{\sigma^2}\frac{\partial\mu^T(\theta)}{\partial\theta}\frac{\partial\mu(\theta)}{\partial\theta}.$$

Ist der Zusammenhang $\mu = f(\theta, u)$ bekannt oder beliebig genau approximiert, das heißt die Struktur des Approximators ist geeignet um μ beliebig genau zu approximieren, lässt sich die FIM für die Zufallsvariablen $x_k = f(\theta, u_k) + \epsilon_k$ wie folgt berechnen

$$\mathcal{I}_k(\theta, u) = \frac{1}{\sigma^2}\frac{\partial f(\theta, u_k)^T}{\partial\theta}\frac{\partial f(\theta, u_k)}{\partial\theta}.$$

B Levenberg-Marquardt Algorithmus

Der Levenberg-Marquardt (LM) Algorithmus ist ein numerischer Optimierungsalgorithmus zum Lösen von nichtlinearen Least-Squares Problemen (das heißt zur Minimierung der Summe der Fehlerquadrate von nichtlinearen Modellen). Dabei interpoliert der LM-Algorithmus zwischen dem Gauß-Newton-Verfahren und dem Gradientenverfahren. Dadurch ist der LM-Algorithmus deutlich robuster als das Gauß-Newton-Verfahren, das heißt, er konvergiert mit einer hohen Wahrscheinlichkeit auch bei schlechten Startbedingungen [57].

Das Minimierungsproblem kann analog zu (2.9) mit der 2-Norm und

$$S_2(\theta) = \beta \left(\sum_{k=1}^{K} ||y_k - f(\varphi_k, \theta)||_2^2 \right) + \alpha \, ||\theta||_2^2 \tag{B.1}$$

wie folgt formuliert werden

$$\theta^* = \arg\min_{\theta} S_2(\theta).$$

Der Gradient der Kostenfunktion (B.1) ist gegeben durch

$$\nabla S_2(\theta) = -2\beta J(\theta)^\top e(\theta) + 2\alpha\theta. \tag{B.2}$$

Mittels eines Gradientenabstiegsverfahrens lässt sich ein (lokales) Minimum von (B.1) finden

$$\theta^{i+1} = \theta^{(i)} - \mu^{(i)} \nabla S_2(\theta^{(i)}).$$

Wird die Schrittweite $\mu^{(i)}$ klein genug gewählt, lässt sich garantieren, dass $S_2(\theta^{(i+1)}) \leq S_2(\theta^{(i)})$ ist. Die erzielte Konvergenzrate ist allerdings nur linear und somit ist das Verfahren in der Praxis recht langsam.

Wird hingegen direkt die optimalen Parameteränderung $\Delta\theta$ zur Minimierung der Kostenfunktion (B.1) gesucht, ergibt sich folgendes Minimierungsproblem

$$\Delta\theta^* = \arg\min_{\Delta\theta} S_2(\theta + \Delta\theta) = \arg\min_{\Delta\theta} \Big(\beta \, ||e(\theta + \Delta\theta)||_2^2 + \alpha \, ||\theta + \Delta\theta||_2^2 \Big). \tag{B.3}$$

Die üblicherweise nichtlineare Kostenfunktion (B.1) wird in einer Umgebung von $e(\theta)$ linarisiert, wodurch sich folgender Ausdruck ergibt

$$S_2(\theta + \Delta\theta) \approx \min_{\theta} \Big(\beta \, ||e(\theta) - J(\theta)\Delta\theta||_2^2 + \alpha \, ||\theta + \Delta\theta||_2^2 \Big). \tag{B.4}$$

Damit lässt sich direkt die optimale Gewichtsänderung bestimmen

$$\begin{aligned}\frac{\mathrm{d}S_2(\theta+\Delta\theta)}{\mathrm{d}\Delta\theta} =& \frac{\mathrm{d}}{\mathrm{d}\Delta\theta}\Big(\beta\left(||e||_2^2 - 2\Delta\theta^\top J^\top e + \Delta\theta^\top J^\top J\Delta\theta\right) + \alpha\left(||\theta||_2^2 + 2\Delta\theta^\top\theta + ||\Delta\theta||_2^2\right)\Big)\\ =& 2\beta J^\top J\Delta\theta - 2\beta J^\top e + 2\alpha\theta + 2\alpha\Delta\theta = 0.\end{aligned}$$

Um die Notation kompakt zu halten, wird die Abhängigkeit der Jakobimatrix $J(\theta) = J$ und des Fehlervektors $e(\theta) = e$ unterdrückt. Für die gesuchte Gewichtsänderung ergibt sich folglich

$$\Delta\theta^* = \left(2\beta J^\top J + 2\alpha I\right)^{-1}\left(2\beta J^\top e - 2\alpha\theta\right).$$

Dabei entspricht der erste Term der Gauß-Newton Approximation der Hessematrix

$$\nabla^2 S_2(\theta) \approx H = 2\beta J^\top J + 2\alpha I.$$

Da in der Praxis die Netzwerkstruktur eines Neuronalen Netzes meist sehr flexibel gewählt wird, ist stets eine Teilmenge der Gewichte für die Problemstellung nicht signifikant. Diese sehr kleinen Gewichte führen dazu, dass die Gauß-Newton Approximation der Hessematrix nahezu singulär wird und somit bei kleinem α numerische Probleme bei der Berechnung auftreten.

Um numerische Probleme zu vermeiden, modifiziert der LM-Algorithmus die Approximation der Hessematrix. Eine Iteration kann nun wie folgt beschrieben werden

$$\theta^{(i+1)} = \theta^{(i)} - (H + \lambda D^\top D)^{-1}\nabla S_2(\theta). \tag{B.5}$$

Dabei ist D eine beliebige nicht-singuläre Matrix und $\lambda > 0$ der LM-Parameter, der die Interpolation zwischen Gauß-Newton-Verfahren und dem Gradientenverfahren bestimmt und bei jeder Iteration angepasst wird. Nimmt λ sehr große Werte an, wird mit λ^{-1} ein sehr kleiner Schritt entlang des negativen Gradienten gemacht, womit der Fehler immer abnimmt, aber die Konvergenz langsam ist. Für kleine λ entspricht der LM-Algorithmus dem Gauß-Newton-Verfahren, welches eine schnelle Konvergenz (nur) in der Nähe des Minimums ermöglicht.

Der LM-Algorithmus kann zudem als Trust-Region Algorithmus betrachtet werden, der den LM-Parameter abhängig vom Vertrauensbereich wählt [57]. Diese Linearisierung (B.4) von (B.3) ist natürlich nur in der Umgebung von $e(\theta)$ gültig. Somit ergibt sich folgendes restringiertes Optimierungsproblem

$$\Delta\theta^* = \arg\min_\theta\Big(\beta\,||e(\theta) - J(\theta)\Delta\theta||_2^2 + \alpha\,||\theta + \Delta\theta||_2^2 \;:\quad ||D\Delta\theta||_2\, 2^2 \le \rho^2\Big) \tag{B.6}$$

mit einer beliebigen nicht-singulären Matrix D, welche den hyperellipsoiden Vertrauensbereich definiert. Ist $\Delta\theta^*$ eine Lösung von (B.6), dann gilt mit einem LM-Parameter $\lambda \ge 0$ der Zusammenhang [57]

$$\Delta\theta^* = -(H + \lambda D^\top D)^{-1}\nabla S_2(\theta). \tag{B.7}$$

Da $\Delta\theta^*$ monoton vom LM-Parameter λ abhängt, muss dieser so gewählt werden, dass der Vertrauensbereich ρ^2 nicht verlassen wird.

Die Wahl des Vertrauensbereichs hängt vom Verhältnis zwischen tatsächlicher und prädi-

zierter Fehlerreduktion ab

$$r(\Delta\theta) = \frac{S_2(\theta) - S_2(\theta + \Delta\theta)}{S_2(\theta) - \beta \left|\left|e(\theta) - J(\theta)\Delta\theta\right|\right|_2^2 - \alpha \left|\left|\theta + \Delta\theta\right|\right|_2^2}.$$

Mit dem Zusammenhang (B.7) und (B.2) ergibt sich der einfach auszuwertende Ausdruck:

$$r(\Delta\theta) = \frac{S_2(\theta) - S_2(\theta + \Delta\theta)}{\beta\|J(\theta)\Delta\theta\|_2^2 + \lambda\|D\Delta\theta\|_2^2 + \alpha\|\Delta\theta\|_2^2}. \tag{B.8}$$

Der Vorteil von (B.8) liegt im, unabhängig von eventuell auftretenden Rundungsfehlern, garantiert positiven Nenner. Die Herleitung von (B.8) findet sich im Abschnitt über die Fehlerreduktionsrate. Somit gibt (B.8) die Übereinstimmung zwischen dem linearen Modell und der (nichtlinearen) Kostenfunktion an.

Wird der in [27] vorgestellte Algorithmus 1 verwendet, lässt sich mit $D = S_2(\theta)I$ (lokal) quadratische Konvergenz garantieren. Dabei haben sich die hier im Algorithmus 1

Algorithmus 1 Levenberg-Marquardt Algorithmus

while $\left|\left|\nabla S_2\left(\theta^i\right)\right|\right|_2 > \epsilon_1$ and $i < \epsilon_2$ **do**

$\Delta\theta = -\left(H(\theta^i) + \lambda S\left(\theta^i\right)^2 I\right)^{-1} \nabla S_2(\theta^i)$

if $r(\Delta\theta) > 10^{-4}$ **then**

$\theta^{(i+1)} = \theta^{(i)} + \Delta\theta$

end if

$$\lambda^{(i+1)} = \begin{cases} 5\lambda^{(i)} & r(\Delta\theta) < 0.15 \\ \lambda^{(i)} & r(\Delta\theta) \in [0.15\,, 0.75] \\ \max\left(\frac{1}{2}\lambda^{(i)}\,, 10^{-8}\right) & r(\Delta\theta) > 0.75] \end{cases}$$

$i = i + 1$

end while

gewählten Parameter in der Praxis bewährt, sind aber innerhalb bestimmter Schranken frei wählbar [27]. Die Parameter ϵ_1 und ϵ_2 sorgen für einen Abbruch der Optimierung, wenn die Schrittweite zu klein beziehungsweise die Anzahl der Iterationen zu groß wird, und stellen aufgabenspezifische Abbruchkriterien dar.

Fehlerreduktionsrate

Um die Notation einfach zu halten, wird die Abhängigkeit der Jakobimatrix $J(\theta) = J$ und des Fehlervektors $e(\theta) = e$ unterdrückt. Das Verhältnis zwischen tatsächlicher und prädizierter Fehlerreduktion ist gegeben durch

$$r(\Delta\theta) = \frac{S_2(\theta) - S_2(\theta + \Delta\theta)}{S_2(\theta) - \beta \left|\left|e - J\Delta\theta\right|\right|_2^2 - \alpha \left|\left|\theta + \Delta\theta\right|\right|_2^2}.$$

Dabei gilt

$$\beta \left|\left|e - J\Delta\theta\right|\right|_2^2 = \beta \left|\left|e\right|\right|_2^2 - 2\beta\Delta\theta^\top J^\top e + \beta \left|\left|J\Delta\theta\right|\right|_2^2$$

und

$$\alpha \left|\left|\theta + \Delta\theta\right|\right|_2^2 = \alpha \left|\left|\theta\right|\right|_2^2 + \alpha \left|\left|\Delta\theta\right|\right|_2^2 + 2\alpha\Delta\theta^\top\theta.$$

Mit dem Zusammenhang (B.7) und (4.5) ergibt sich

$$2\beta J(\theta)^\top e(\theta) = (H + \lambda D^\top D)\Delta\theta + 2\alpha\theta$$

und unter der Verwendung der Gauß-Newton Approximation der Hessematrix

$$\begin{aligned} 2\beta\Delta\theta^\top J^\top e &= \Delta\theta^\top(H + \lambda D^\top D)\Delta\theta + 2\alpha\Delta\theta^\top\theta \\ &= \Delta\theta^\top \left(2\beta J^\top J + 2\alpha I + \lambda D^\top D\right)\Delta\theta + 2\alpha\Delta\theta^\top\theta \\ &= 2\beta \left|\left|J\Delta\theta\right|\right|_2^2 + 2\alpha \left|\left|\Delta\theta\right|\right|_2^2 + \lambda \left|\left|D\Delta\theta\right|\right|\right|_2^2 + 2\alpha\Delta\theta^\top\theta. \end{aligned}$$

Damit gilt

$$\beta \left|\left|e - J\Delta\theta\right|\right|_2^2 = \beta \left|\left|e\right|\right|_2^2 - \beta \left|\left|J\Delta\theta\right|\right|_2^2 - 2\alpha \left|\left|\Delta\theta\right|\right|_2^2 - \lambda \left|\left|D\Delta\theta\right|\right|_2^2 - 2\alpha\Delta\theta^\top\theta$$

und mit $S(\theta)_2 = \beta\|e\|_2^2 + \alpha\|\theta\|_2^2$ für

$$r(\Delta\theta) = \frac{S_2(\theta) - S_2(\theta + \Delta\theta)}{S_2(\theta) - \beta \left|\left|e + J\Delta\theta\right|\right|_2^2 - \alpha \left|\left|\theta + \Delta\theta\right|\right|_2^2} = \frac{S_2(\theta) - S_2(\theta + \Delta\theta)}{\beta \left|\left|J(\theta)\Delta\theta\right|\right|_2^2 + \lambda \left|\left|D\Delta\theta\right|\right|_2^2 + \alpha \left|\left|\Delta\theta\right|\right|_2^2}.$$

C Definitionen

C.1 Totale Ableitung

Die totale Ableitung einer von mehreren Variablen abhängenden Funktion $f(t, x, y)$ nach einen ihrer Variablen ist verschieden von der partiellen Ableitung. Die Berechnung der totalen Ableitung von f basiert nicht auf der Annahme, dass, wenn sich eine Variable ändert, die anderen Variablen konstant bleiben. Die totale Ableitung trägt dieser indirekten Abhängigkeit Rechnung, indem die Kettenregel angewandt wird. So ist zum Beispiel die totale Ableitung von $f(t, x, y)$ nach t

$$\frac{\mathrm{d}\, f}{\mathrm{d}\, t} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\frac{\mathrm{d}\, x}{\mathrm{d}\, t} + \frac{\partial f}{\partial y}\frac{\mathrm{d}\, y}{\mathrm{d}\, t}.$$

Die totale Ableitung wird bei der Berechnung des Gradienten und der Jakobimatrix verwendet.

C.2 Matrix- und Vektoreigenschaften

Die folgenden Aussagen, Definitionen und Theoreme sind alle [36] entnommen. Auf die hier aufgeführten Ergebnisse, wird vor allem bei den Stabilitätsbeweisen in Anhang D zurückgegriffen.

Theorem 4 (Rayleigh-Ritz)**.** *Für eine hermetische Matrix $A \in \mathbb{C}^{n\times n}$ gilt:*

$$\lambda_{\min} x^H x \leq x^H A x \leq \lambda_{\max} x^H x \quad \forall x \in \mathbb{C}^n$$

$$\lambda_{\max} \max_{x\neq 0} \frac{x^H A x}{x^H x} = \max_{x^H x = 1} x^H A x$$

$$\lambda_{\min} \min_{x\neq 0} \frac{x^H A x}{x^H x} = \min_{x^H x = 1} x^H A x.$$

Definition 1 (positiv definite Matrix)**.** *Eine $n \times n$ hermetische Matrix A ist positiv definit, wenn*

$$x^H A x > 0$$

für alle $x \in \mathbb{C}^n$ ungleich Null gilt.

Wird die Forderung abgeschwächt zu $x^H A x \geq 0$ spricht man von positiv semi-definit. Wird die Gleichung beziehungsweise Ungleichung umgedreht, spricht man entsprechend

von negativ definit beziehungsweise negativ semi-definit. Ist eine Matrix weder positiv noch negativ (semi-) definit, wird sie als indefinit bezeichnet.

Dabei gilt

Theorem 5. *Jeder Eigenwert einer positiv (negativ) definiten Matrix ist eine positive (negative) Zahl.*

Matrix- und Vektornormen

In diesem Abschnitt werden die in der Arbeit verwendeten Vektor- und Matrizennormen eingeführt [36]

Definition 2. *Die p-Normen eines Vektors $x \in \mathbb{R}^n$ sind durch*

$$||x||_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

definiert. Dabei ist $p \geq 1$ eine reelle Zahl und $|x_i|$ der Absolutbetrag der i-ten Koordinate des Vektors x.

Die bekanntesten Normen sind dabei:

- Betragssummennorm: $p = 1$
$$||x||_1 = \sum_{i=1}^{n} |x_i|$$
- Euklidische Norm: $p = 2$
$$||x||_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$$
- Maximumsnorm: $p \to \infty$
$$||x||_\infty = \max_i |x_i|$$

Als Normen auf endlich dimensionalen Vektorräumen sind alle p-Normen inklusive der Maximumsnorm zueinander äquivalent. Dabei gelten für einen n-dimensionalen Raum folgende Ungleichungen:

$$||x||_p \leq ||x||_1 \leq \sqrt[p]{n^{p-1}}\ ||x||_p$$
$$||x||_\infty \leq ||x||_p \leq \sqrt[p]{n}\ ||x||_\infty .$$

Außerdem gilt für $q > p \geq 1 : ||x||_p \geq ||x||_q$.

Die Maxmimumsnorm ist dabei ein Sonderfall der Supremumsnorm für endlich dimensionale Vektoren. Die Supremumsnorm weist einer reellen beschränkten Funktion f definiert

für eine Menge S eine nicht negative Zahl zu

$$||f||_\infty = \sup\{|f(x)|, x \in S\} < \infty.$$

Die Menge aller durch die Supremumsnorm beschränkten Funktionen $f : \mathbb{R} \to \mathbb{R}^m$ wird dabei mit ℓ_∞^m bezeichnet und somit gitl $f \in \ell_\infty^m$.

Neben den Vektornormen wird noch die Matrixnorm und deren Eigenschaften benötigt. Als Matrixnorm wird die induzierte p-Norm verwendet.

Definition 3. *Für eine Matrixnorm $||\cdot||_p$ induziert von einer Vektornorm $||\cdot||_p$ gilt*

$$||A||_p = \sup_{x \neq 0} \frac{||Ax||_p}{||x||_p} = \sup_{||x||_p = 1} ||Ax||_p .$$

Eine induzierte Matrixnorm ist stets submultiplikativ

$$||AB|| \leq ||A||\,||B|| .$$

Jede induzierte Norm erfüllt folgende Bedingung

Theorem 6. *Wenn $||\cdot||$ eine induzierte Matrixnorm ist, dann gilt*

$$||A|| \geq \rho(A),$$

wobei $\rho(A)$ der Spektralradius von A ist.

Dieser ist wie folgt definiert

Definition 4. *Der Spektralradius $\rho(A)$ einer Matrix A ist*

$$\rho(A) := \max_i(|\lambda_i|),$$

wobei λ_i ein Eigenwert von A ist.

Jede von einer Vektornorm induzierte Matrixnorm ist mit dieser Vektornorm verträglich

Definition 5. *Eine Matrixnorm $||\cdot||_M$ heißt mit einer Vektornorm $||\cdot||_V$ verträglich, wenn*

$$||Ax||_V \leq ||A||_M\,||x||_V \tag{C.1}$$

gilt.

C.3 Fehlermaße

Die folgenden Fehlermaße werden in der Arbeit verwendet.

NMSE: Normalised Mean Squared Error
Der NMSE ist definiert als mittlerer quadratischer Fehler normiert mit der Varianz des Referenzsignals

$$NMSE = \frac{\frac{1}{K}\sum_{k=1}^{K}(x_k - \hat{x}_k)^2}{\sigma(x)^2}$$

$$\sigma(x)^2 = \frac{1}{K}\sum_{k=1}^{K}\left(x_k - \frac{1}{K}\sum_{j=1}^{K}x_j\right)^2.$$

RMSE: Root Mean Squared Error
Der RMSE ist die Wurzel des mittleren quadratischen Fehlers

$$RMSE = \sqrt{\frac{1}{K}\sum_{k=1}^{K}(x_k - \hat{x}_k)^2}.$$

NMAE: Normalized Mean Absolute Error
Der NMAE skaliert den mittleren absoluten Fehler auf den Wertebereich, welcher von den Messdaten abgedeckt wird

$$NMAE = \frac{\sum_{k=1}^{K}|e_k|}{K(y_{\max} - y_{\min})}.$$

D Stabilitätsanalyse

In diesem Abschnitt werden die nötigen Definitionen und Theoreme eingeführt, die für die Stabilitätsuntersuchung zeitdiskreter und zeitkontinuierlicher Systeme mittels Lyapunov-Funktionen nötig sind. Sofern nicht anders angegeben, sind die Definitionen und Theoreme [46] entnommen. Anschließend werden Bedingungen an die Netzgewichte hergeleitet, welche die Eingangs-/Zustandsstabilität und Ein-/Ausgangsstabilität von RKNN sicherstellen.

Wird die Stabilität der Ruhelage untersucht, ist man sowohl an deren Eindeutigkeit wie auch an einer Methode zur Ermittlung derselben interessiert. Das Theorem der Kontraktion gibt die hinreichende Bedingung hierzu an.

Theorem 7 (Kontraktion)**.** *Sei $\mathcal{S}$ eine geschlossene Teilmenge eines Banach-Raumes[1] $\mathcal{X}$, dann heißt eine Abbildung $T : \mathcal{S} \to \mathcal{S}$ Kontraktion, wenn es eine Zahl $\rho \in [0, 1)$ gibt, mit der für alle $x, y \in \mathcal{S}$ gilt:*

$$||T(x) - T(y)|| \leq \rho \, ||x - y|| \, .$$

Für eine Kontraktion gilt außerdem:

- *Es existiert ein eindeutiger Vektor $x^* \in \mathcal{S}$ für den gilt $x^* = T(x^*)$. Dabei nennt man x^* einen Fixpunkt der Abbildung $T(\cdot)$, da $T(\cdot)$ den Fixpunkt x^* invariant lässt.*
- *x^* kann über die Iterationsfolge $x_{n+1} = T(x_n)$ mit einem beliebigen Startwert $x_0 \in \mathcal{S}$ berechnet werden.*

Die lokale Existenz und Eindeutigkeit der Ruhelage wird über folgendes Theorem definiert:

Theorem 8 (Lokale Existenz und Eindeutigkeit)**.** *Sei $f(x, t)$ stückweise konstant in t und die Lipschitzbedingung*

$$||f(x, t) - f(y, t)|| \leq L \, ||x - y|| \tag{D.1}$$

für $\forall x, y \in B = \left\{ x \in \mathbb{R}^{N_x} | \ ||x - x_0|| \leq r \right\}$, $\forall t \in [t_0, t_1]$ erfüllt, dann existiert ein $\delta > 0$ so dass für die Zustandsgleichung

$$\dot{x} = f(x, t), \quad x(t_0) = x_0$$

eine eindeutige Lösung in $[t_0, t_0 + \delta]$ existiert.

Eine Funktion, welche die Lipschitzbedingung (D.1) erfüllt, bezeichnet man als Lipschitz-stetig in x und die positive Konstante L als Lipschitzkonstante. Je nach Gültigkeitsbereich der Lipschitzbedingung wird von lokal oder global Lipschitz-stetig in x gesprochen. Aus der Lipschitzbedingung wird klar, dass eine Funktion, welche in irgendeinem Punkt eine Unstetigkeit in der Steigung besitzt, in diesem nicht lokal Lipschitz-stetig ist. Somit können nur kontinuierliche Funktionen ohne Unstetigkeiten global Lipschitz-stetig sein.

[1] Ein Banach-Raum ist ein vollständig normierter Raum, beispielsweise der Euklidische Raum.

Theorem 9 (Globale Existenz und Eindeutigkeit). *Sei $f(x,t)$ stückweise konstant in t und die Bedingungen*

$$||f(x,t) - f(y,t)|| \leq L\,||x-y|| \tag{D.2}$$
$$||f(x_0,t)|| \leq c \tag{D.3}$$

für $\forall x, y \in \mathbb{R}^{N_x}$, $\forall t \in [t_0, t_1]$, $c \in \mathbb{R}$ erfüllt, dann besitzt

$$\dot{x} = f(x,t), \quad x(t_0) = x_0$$

eine eindeutige Lösung in $[t_0, t_1]$.

Für die Stabilitätsanalyse wird zusätzlich der Formalismus für Vergleichsfunktionen benötigt: Eine Funktion der Klasse $\mathcal{K}$ ist eine Funktion $\alpha : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ welche kontinuierlich, strikt steigend ist und $\alpha(0) = 0$ erfüllt. Ist die Funktion eine Funktion der Klasse $\mathcal{K}$ und es gilt zudem $\alpha(s) \to \infty$, wenn $s \to \infty$, ist sie eine Funktion der Klasse $\mathcal{K}_\infty$. Eine Funktion der Klasse $\mathcal{KL}$ ist eine Funktion $\beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+$ so dass gilt $\beta(\cdot, t) \in \mathcal{K}_\infty$ für jedes $t \geq 0$ und $\beta(s,t) \to 0$ für jedes $s \geq 0$, wenn $t \to \infty$.

D.1 Stabilitätsanalyse zeitdiskreter Systeme

Die Stabilität von Systemen in Zustandsraumdarstellung wird meist über die sogenannte Eingangs-/Zustandsstabilität (*engl.: input-to-state stability, ISS*) definiert.

Definition 6 (ISS zeitdiskreter Systeme [43]). *System (2.2a) ist ISS, wenn eine $\mathcal{KL}$ Funktion $\beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+$ und eine $\mathcal{K}$ Funktion γ existiert, so dass für jeden Eingang $u \in \ell_\infty^{N_I}$ und jeden Anfangswert $x(0) \in \mathbb{R}^{N_x}$ gilt, dass*

$$||x(n, x(0), u)|| \leq \beta(||x(0)||, n) + \gamma(||u||_{\sup}) \tag{D.4}$$

für jedes $n \in \mathbb{N}_+$ erfüllt ist.

ISS für zeitdiskrete Systeme lässt sich durch diverse Aussagen sicherstellen.

Theorem 10 (Input-to-State Stabilität (ISS) [43]). *Für ein System $x_{k+1} = f(x_k, u_k)$ sind die folgenden Aussagen equivalent:*

- *Das System ist ISS.*
- *Das System ist robust stabil.*
- *Für das System existiert eine glatte ISS-Lyapunov Funktion.*

Die Stabilität wird meist durch den Ansatz einer glatten ISS-Lyapunov Funktion geprüft. Diese ist wie folgt definiert:

Definition 7 (ISS-Lyapunov Funktion). *Eine kontinuierliche Funktion $V : \mathbb{R}^{N_x} \to \mathbb{R}_{\geq 0}$ ist eine ISS-Lyapunov Funktion für $x_{k+1} = f(x_k, u_k)$, wenn die folgenden Aussagen zutreffen:*

- *Es existieren $\mathcal{K}_\infty$-Funktionen α_1, α_2 so dass*

$$\alpha_1(||x||_2) \leq V(x) \leq \alpha_2(||x||_2) \quad \forall x \in \mathbb{R}^{N_x}.$$

- *Es existiert eine $\mathcal{K}_\infty$-Funktion α_3 und eine $\mathcal{K}$-Funktion σ, so dass*

$$V(f(x,u)) - V(x) \leq -\alpha_3(||x||_2) + \sigma(||u||_2) \quad \forall x \in \mathbb{R}^{N_x}, \forall u \in \mathbb{R}^{N_I}.$$

D.2 Stabilitätsanalyse zeitkontinuierlicher Systeme

Analog zur Stabilitätsanalyse zeitdiskreter Systeme wird ISS für zeitkontinuierliche Systeme definiert.

Definition 8 (ISS zeitkontinuierlicher Systeme [80])**.** *System (2.1a) ist ISS, wenn eine $\mathcal{KL}$ Funktion $\beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+$ und eine $\mathcal{K}$ Funktion γ existieren, so dass für jeden Eingang $u \in \ell_\infty^{N_I}$ und jeden Anfangswert $x(0) \in \mathbb{R}^{N_x}$, gilt dass*

$$||x(t, x(0), u)|| \leq \beta(||x(0)||, t) + \gamma(||u||_\infty) \tag{D.5}$$

für alle $t \geq 0$ erfüllt ist.

Wie im zeitdiskreten Fall, ist ein System (2.1a) dann und nur dann ISS, wenn es eine ISS-Lyapunov Funktion besitzt [80].

Definition 9 (ISS-Lyapunov Funktion [80])**.** *Eine glatte Funktion $V : \mathbb{R}^{N_x} \to \mathbb{R}_0^+$ wird ISS-Lyapunov Funktion des Systems (2.1a) genannt, wenn $\mathcal{K}_\infty$ Funktionen α_1, α_2, α_3 und eine $\mathcal{K}$-Funktion σ existieren, so dass*

$$\alpha_1(||x||_2) \leq V(x) \leq \alpha_2(||x||_2) \tag{D.6}$$

und für jedes $x \in \mathbb{R}^{N_x}$ und

$$\dot{V}(x) \leq -\alpha_3(||x||_2) + \sigma(||u||_2) \tag{D.7}$$

für jedes $x \in \mathbb{R}^{N_x}$ und jedes $u \in \mathbb{R}^{N_I}$ gilt.

D.2.1 Zusammenhang zwischen ISS und Stabilität nach Lyapunov

Die Stabilität von Ruhelagen eines Systems wird meist nach Lyapunov definiert. Um den Zusammenhang zwischen ISS und Stabilität nach Lyapunov herzustellen, wird zunächst die Stabilität nach Lyapunov eingeführt.

Eine Ruhelage x^* heißt stabil im Sinne von Lyapunov, wenn jede Trajektorie, die in der Nähe von x^* beginnt, für alle $t \geq 0$ in der Nähe von x^* verweilt. Konvergieren oder enden alle Trajektorien, die in der Nahe von x^* beginnen, in der Ruhelage x^*, heißt die Ruhelage asymptotisch stabil. Eine Ruhelage ist instabil im Sinne von Lyapunov, wenn sie nicht stabil ist.

Definition 10 (Stabilität in Sinne von Lyapunov)**.** *Eine Ruhelage $x^* = 0$ des Systems (2.1a) heißt*

- *stabil im Sinne von Lyapunov, wenn für jedes $\epsilon > 0$ ein $\delta = \delta(\epsilon) > 0$ existiert, so dass gilt*
$$||x(0)|| < \delta \Rightarrow x(t) < \epsilon, \forall t \geq 0.$$
- *asymptotisch stabil im Sinne von Lyapunov, wenn x^* stabil und $\delta(\epsilon) > 0$ so gewählt werden kann, dass gilt*
$$||x(0)|| < \delta \Rightarrow \lim_{t\to\infty} x(t) = 0.$$
- *instabil, wenn x^* nicht stabil ist.*

Stabilität im Sinne von Lyapunov kann gezeigt werden, wenn für das System (2.1a) mit konstantem Eingang u eine Lyapunov-Funktion $V(x)$, $\dot{V}(x) = \frac{\partial V}{\partial x} f(x) \leq 0$ gefunden werden kann, sodass folgendes Theorem erfüllt ist:

Theorem 11 (Lokale Stabilität im Sinne von Lyapunov)**.** *Sei $x = 0$ eine Ruhelage des Systems (2.1a) und $D \subset \mathbb{R}^{N_x}$ eine Menge die $x = 0$ enthält. Sei $V : D \to \mathbb{R}$ eine stetige differenzierbare Funktion, sodass*

$$V(0) = 0 \text{ und } V(x) > 0 \text{ in } D - \{0\},$$

$$\dot{V}(x) \leq 0 \text{ in } D,$$

dann ist $x(0)$ stabil. Gilt

$$\dot{V}(x) < 0 \text{ in } D - \{0\},$$

dann ist $x = 0$ asymptotisch stabil.

Für globale asymptotische Stabilität ist es allerdings nicht ausreichend, wenn V positiv definit und $\dot{V}$ negativ (semi-) definit ist. Zudem muss V radial unbeschränkt sein, das heißt $||x|| \to \infty \Rightarrow V(x) \to \infty$ muss erfüllt sein:

Theorem 12 (Globale asymptotische Stabilität im Sinne von Lyapunov)**.** *Sei $x^* = 0$ eine Ruhelage des Systems (2.1a) und $D \subset \mathbb{R}^{N_x}$ eine Menge die $x = 0$ enthält. Sei $V : \mathbb{R}^{N_x} \to \mathbb{R}$ eine stetige differenzierbare Funktion, sodass*

$$V(0) = 0 \text{ und } V(x) > 0, \quad \forall x \neq 0, \tag{D.8}$$

$$||x|| \to \infty \Rightarrow V(x) \to \infty,$$
$$\dot{V}(x) < 0, \quad \forall x \neq 0 \tag{D.9}$$

dann ist $x = 0$ global asymptotisch stabil.

Die geforderte Ruhelage im Ursprung stellt dabei keine Einschränkung dar, da die Ruhelage immer in den Ursprung überführt werden kann [46].

Mit dem Theorem für globale asymptotische Stabilität (GAS) kann nun ISS (D.5) mit GAS verglichen werden. Für einen konstanten Eingang kann stets durch eine Koordinatentransformation der konstante Eingang in die Ruhelage überführt werden, sodass mit (D.5)

$$||x(t, x(0)|| \leq \beta(||x(0)||, t)$$

bei einem ISS-System stets GAS sichergestellt ist. Dies spiegelt sich auch in der ISS-Lyapunov Funktion (D.6), (D.7) wieder, welche für $u = 0$ der Laypunov-Funktion (D.8), (D.9) entspricht.

Ist der Eingang nicht konstant gilt mit (D.5)

$$\lim_{t\to\infty} ||x(t, x_0, u)|| \leq \gamma(||u||_\infty) \quad \forall x(0), u(\cdot),$$

womit sich die Systemtrajektorie über die Zeit beliebig nahe einer Sphäre annähert, deren Radius proportional ist zur Amplitude des Eingangs, und somit analog zur Stabilität nach Lyapunov eine ϵ-Umgebung unabhängig vom Eingang u nicht mehr verlässt. Das heißt ein beschränkter Eingang hat stets beschränkte Zustände zur Folge. Diese Zusammenhänge sind entsprechend auch für zeitdiskrete Systeme gültig.

Neben diesen leicht ersichtlichen Zusammenhängen zwischen Stabilität nach Lyapunov und ISS, existieren noch eine Reihe weiterer Stabilitätseigenschaften, die zueinander äquivalent sind. Ein Überblick findet sich beispielsweise in [81] und [79].

D.2.2 Zusammenhang ISS und Ein-/Ausgangsstabilität

Ist das System (2.1a) ISS ist das Gesamtsystem (2.1a), (2.1b) auch IOS (*engl. input to output stable*), das heißt es existiert ein $\beta \in \mathcal{KL}$ und $\gamma \in \mathcal{K}_\infty$, so dass

$$||y(t)|| \leq \beta(||x(0)||, t) + \gamma(||u||_\infty)$$

hält, und zudem auch IOSS (*engl. input/output to state stable*) für $\beta \in \mathcal{KL}$ und $\gamma_u, \gamma_y \in \mathcal{K}_\infty$ erfüllt

$$||x(t)|| \leq \beta(||x(0)||, t) + \gamma_u(||u||_\infty) + \gamma_y(||y||_\infty).$$

Dabei gilt allgemein der Zusammenhang IOS+IOSS $\Leftrightarrow$ ISS [79]. Da ISS die Ein-/Ausgangsstabilität garantiert, wird an dieser Stelle nicht näher auf diese eingegangen.

D.3 Stabilität von Runge-Kutta Verfahren

Die algebraische Stabilität von expliziten Runge-Kutta Verfahren, deren Parameter durch den zugehörigen Butcher Array

$$\begin{array}{c|c} c & A \\ \hline & b^\top \end{array}$$

beschrieben werden (siehe Abschnitt 5.2.2), lässt sich wie folgt sicherstellen:

Definition 11 ((k,l,m)-algebraische Stabilität von expliziten Runge-Kutta Methoden [13])**.** *Für reale k, l und m ist eine Runge-Kutta Methode (k,l,m) algebraisch stabil, wenn eine Diagonalmatrix $D \geq 0$ existiert, so dass die Matrix*

$$M(k,l,m) = \begin{bmatrix} k-1-2le^\top De & e^\top D - b^\top - 2le^\top DA \\ De - b - 2lA^\top De & mD + DA + A^\top D - bb^\top - 2lA^\top DA \end{bmatrix}$$

nicht negativ definit ist.

Der Vektor e ist ein Vektor mit Einsen als Einträgen und passender Dimension, A und b sind durch den Butcher-Array des RK-Verfahrens vorgegeben. Die Koeffizienten k, l und m sind abhängig vom gewählten RK-Verfahren, wobei m für impliziten RK-Verfahren gleich 0 ist [13]. Für ein (k,l,m)-algebraisches RK-Verfahren gilt [13]:

Theorem 13. *Wird ein (k,l,m)-algebraisches RK-Verfahren mit den Funktionen für ein System mit den Eigenschaften*

$$\langle \dot{x}, \dot{x} \rangle \leq \mu(t) \langle x, x \rangle \tag{D.10}$$

und

$$\langle x, \dot{x} \rangle \leq -\beta(t) \langle x, x \rangle$$

mit $\beta : \mathbb{R}^+ \to \mathbb{R}_0^+$ und $\mu : \mathbb{R}^+ \to \mathbb{R}^+$ angewandt, und für die Schrittweite h_{rk} ist

$$-2h_{rk}\beta(t) - 2l + h_{rk}^2 \mu(t) m \leq 0 \tag{D.11}$$

erfüllt, dann gilt

$$\langle x_{k+1}, x_{k+1} \rangle \leq k \langle x_k, x_k \rangle.$$

Dabei gilt $\beta(t) \geq 0 \ \forall t$, x_k beziehungsweise x_{k+1} sind Lösungen des RK-Verfahrens und der Faktor $0 < k < 1$ ein Koeffizient des RK-Verfahrens. Dies bedeutet, dass ein System, welches stabil im Sinne vom Lyapunov ist, bei geeigneter Wahl der Schrittweite h_{rk} auch (k,l,m)-algebraisch stabile RK-Verfahren ermöglicht. Das jedes Lipschitz-stetige System die Bedingung D.10 erfüllt, wird in Lemma 1 gezeigt. Die Stabilität der Ruhelage nach Lyapunov ist zwar eine notwendige aber keine hinreichende Bedingung für ISS [81]. Ziel wird es im folgenden sein, Bedingungen zu formulieren, um ISS für das explizite RK-Verfahren sicherzustellen.

Analog zur ISS existiert folgende Definition der B-ISS für implizite RK-Methoden:

Definition 12 (B-ISS Runge-Kutta Methoden [37])**.** *Eine Runge-Kutta Methode wird als B-eingangs-/zustandsstabil (B-ISS) bezeichnet, wenn die Runge-Kutta Methoden auf das nichtlineare System (2.1a) mit Hypothese (D.5) angewandt wird und eine $\mathcal{KL}$ Funktion $\beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+$ und eine $\mathcal{K}$ Funktion γ existiert, so dass für jeden Eingang $u \in \ell_\infty^{N_I}$ und jeden Anfangszustand $x(0) \in \mathbb{R}^{N_x}$ gilt, dass*

$$||x(n, x(0), u)|| \leq \beta(||x(0)||, n) + \gamma(||u||_{\sup})$$

für alle $t \geq 0$ erfüllt ist.

B-ISS bedeutet somit, dass sowohl das zeitkontinuierliche als auch das über eine Runge-Kutta Methode zeitdiskretisierte System ISS ist. Die Definition der B-ISS von Runge-Kutta Methoden wird für den nachfolgenden Stabilitätsbeweis verwendet.

D.4 Stabilität von Runge-Kutta Neuronalen Netzen

Die Bedingungen an die Netzgewichte um ISS des RKNN sicherzustellen, wird in zwei Schritten durchgeführt. Zunächst werden Bedingungen für B-ISS explizite RK-Methoden hergeleitet und anschließend untersucht, welche Bedingungen an die Gewichte des RKNN erfüllt sein müssen, um B-ISS sicherzustellen. Um den Beweis besser lesbar zu gestalten, wird nicht auf die Gleichungen in den einzelnen Kapiteln verwiesen, sondern diese erneut aufgeführt.

D.4.1 B-ISS expliziter Runge-Kutta Methoden

In diesem Abschnitt wird B-ISS von expliziten Runge-Kutta Methoden untersucht und Bedingungen an das System und die Schrittweite formuliert, welche B-ISS sicherstellen. Die Herleitung der Bedingungen besteht aus zwei Schritten:

1. Zunächst wird bewiesen, dass für gegebenes m die RK-Methode (k, l, m)-algebraisch stabil ist, wenn l einer Funktion von m, h_{rk}, $\beta(t)$, der Lipischtz-Konstante L und den Eigenwerten einer positiv definiten Matrix P genügt.

2. Anschließend wird gezeigt wie $\beta(t)$ gewählt werden kann, um eine von h_{rk} und m abhängige Lyapunov Funktion zu erhalten. Erfüllt das System beziehungsweise das Modell (siehe auch (5.4))

$$\dot{x}(t) = f(x(t), u(t)) \tag{D.12a}$$
$$y(t) = h(x(t), u(t)) \tag{D.12b}$$

   diese Lyapunov Funktion, stellt dies unabhängig von den Eigenwerten von P und der Lipschitz-Konstante L B-ISS sicher.

Um die algebraische Stabiliät von expliziten Runge-Kutta Methoden zu zeigen, wird folgende ISS-Lyapunov Funktion verwendet

$$V(x) = \frac{1}{2} x^\top P x \tag{D.13a}$$
$$\dot{V}(x) = \dot{x}^\top P x \leq -\beta(t) V(x) + \gamma \langle u, u \rangle \tag{D.13b}$$

mit symmetrischer positiv definiter Matrix $P > 0$, wobei $\beta(t) > 0, \gamma \geq 0$. Man beachte dass (D.13) die dissipative Charakterisierung der ISS Eigenschaft erfüllt, nachdem immer ein $\beta_{min} \leq \beta(t)\ \forall t$ existiert.

Um die Notation kompakt zu halten, wird im folgenden das innere Produkt $\langle \cdot, \cdot \rangle$ verwendet, welches definiert ist als $\langle x, y \rangle = x^\top P y$.

Bevor das Theorem vorgestellt wird, welches die B-ISS expliziter RK-Methoden sicherstellt, wird das folgende Lemma benötigt, welches sicherstellt, dass jedes Lipschitz-stetige System nach (D.10) μ-begrenzt ist.

Lemma 1. *Jedes Lipschitz-stetige System ist μ-begrenzt in dem Sinne, dass immer ein $0 < \mu(t) \leq \sigma(P)L^2$ existiert, so dass gilt*

$$\langle \dot{x}, \dot{x} \rangle \leq \mu(t)\langle x, x \rangle + \sigma(P)L_u^2\langle u, u \rangle \qquad t \in [k, k+1]. \tag{D.14}$$

Dabei ist $\sigma(P)$ das Verhältnis aus größtem und kleinstem Eigenwert von P: $\sigma(P) = 2\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}$.

Beweis. Nachdem das System (D.12) im ersten sowie zweitem Argument Lipschitz-stetig ist, gilt allgemein mit den Lipschitz-Konstanten L und L_u

$$\begin{aligned} ||f(x,u) - f(y,u)||_2 &\leq L\,||x-y||_2 \\ ||f(y,u) - f(y,v)||_2 &\leq L_u\,||u-v||_2\,. \end{aligned}$$

Somit gilt auch mit der Dreiecksungleichung $||a||_2 + ||b||_2 \geq ||a+b||_2$

$$\begin{aligned} L\,||x-y||_2 + L_u\,||u-v||_2 &\geq ||f(x,u) - f(y,u)||_2 + ||f(y,u) - f(y,v)||_2 \\ &\geq ||f(x,u) - f(y,u) + f(y,u) - f(y,v)||_2 \\ &\geq ||f(x,u) - f(y,v)||_2\,. \end{aligned}$$

Da das System zudem per Definition $f(0,0) = 0$ erfüllt, gilt

$$||\dot{x}||_2 \leq L\,||x||_2 + L_u\,||u||_2\,.$$

Mit der Abschätzung $(||a||_2 - ||b||_2)^2 \geq 0$ lässt sich leicht der Zusammenhang $2\,||a||_2^2 + 2\,||b||_2^2 \geq (||a||_2 + ||b||_2)^2$ herleiten:

$$\begin{aligned} (||a||_2 - ||b||_2)^2 &\geq 0 \\ ||a||_2^2 + ||b||_2^2 &\geq 2\,||a||_2\,||b||_2 \\ 2\,||a||_2^2 + 2\,||b||_2^2 &\geq 2\,||a||_2\,||b||_2 + ||a||_2^2 + ||b||_2^2 \\ 2\,||a||_2^2 + 2\,||b||_2^2 &\geq (||a||_2 + ||b||_2)^2 \end{aligned}$$

und somit gilt folgende Abschätzung

$$||\dot{x}||_2^2 \leq 2L^2\,||x||_2^2 + 2L_u^2\,||u||_2^2\,.$$

Nun kann nach Rayleigh-Ritz (Theorem 4) die Abschätzung

$$\lambda_{\min}(P)\,||x||_2^2 \leq x^\top P x \leq \lambda_{\max}(P)\,||x||_2^2$$

verwendet werden, um folgendes zu berechnen:

$$\begin{aligned} ||\dot{x}||_2^2 \leq & 2L^2 \, ||x||_2^2 + 2L_u^2 \, ||u||_2^2 \\ \langle \dot{x}, \dot{x} \rangle \leq & \lambda_{\max}(P) \, ||\dot{x}||_2^2 \leq \lambda_{\max}(P)(2L^2 \, ||x||_2^2 + 2L_u^2 \, ||u||_2^2) \\ \lambda_{\min}(P)\langle \dot{x}, \dot{x} \rangle \leq & \lambda_{\min}(P)\lambda_{\max}(P) \left(2L^2 \, ||x||_2^2 + 2L_u^2 \, ||u||_2\right) \\ \leq & \lambda_{\max}(P) 2L^2 \langle x, x \rangle + \lambda_{\max}(P) 2L_u^2 \langle u, u \rangle \end{aligned}$$

womit der Zusammenhang

$$\langle \dot{x}, \dot{x} \rangle \leq \frac{\lambda_{\max}(P)}{\lambda_{\min}(P)} 2(L^2 \langle x, x \rangle + L_u^2 \langle u, u \rangle) = \sigma(P)(L^2 \langle x, x \rangle + L_u^2 \langle u, u \rangle).$$

erfüllt ist. Damit gilt (D.14). □

Das folgende Ergebnis erweitert das Resultat aus [37] auf explizite RK-Methoden. Für $m = 0$, $\beta(t) = const$ und $P = I$ entspricht das folgende Theorem dem Ergebnis in [37] für implizite RK-Methoden.

Theorem 14 (B-ISS von expliziten RK-Methoden [16])**.** *Angenommen*

- *für das nichtlineare ISS System (D.12) gilt mit* $V(x) = 0.5\langle x, x \rangle$ *die Hypothese*

$$\dot{V}(x) \leq -\beta(t)V(x) + \gamma\langle u, u \rangle, \tag{D.15}$$

- *für* $0 < k < 1$*, ist die RK-Methode* (k, l, m)*-algebraisch stabil, wobei*

$$l = -h_{rk}\beta(t) + h_{rk}^2 \mu(t) m < 0, \tag{D.16}$$

dann ist für das nichtlineare Modell (D.12) das explizite RK-Verfahren B-ISS.

Beweis. Das Runge-Kutta Verfahren (5.1) kann auch wie folgt geschrieben werden [37]:

$$x((n+1)h_{rk}) = x(nh_{rk}) + h_{rk} \sum_{i=1}^{s} b_i f(Y(i), u(nh_{rk} + c_i h_{rk})) \tag{D.17a}$$

$$Y(i) = x(nh_{rk}) + h_{rk} \sum_{j=1}^{s} a_{ij} f(Y(j), u(nh_{rk} + c_j h_{rk})). \tag{D.17b}$$

Unter Verwendung des Kronecker Produkts $\otimes$, der Vektoren

$$Y = \begin{bmatrix} Y(1) \\ \vdots \\ Y(s) \end{bmatrix}, \quad F(Y, nh_{rk} + h_{rk}c) = \begin{bmatrix} f(Y(1), u(nh_{rk} + c_1 h_{rk})) \\ \vdots \\ f(Y(s), u(nh_{rk} + c_s h_{rk})) \end{bmatrix}$$

und

$$e_s = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \quad c = Ae_s = \begin{bmatrix} c_1 \\ \vdots \\ c_s \end{bmatrix}$$

kann das zeitdiskrete System (D.17) kompakt formuliert werden [13]

$$x((n+1)h_{rk}) = x(nh_{rk}) + h_{rk}\left(b^\top \otimes I_s\right) F(Y, nh_{rk} + h_{rk}c) \tag{D.18}$$

$$Y = e_s \otimes x(nh_{rk}) + h_{rk}\left(A \otimes I_s\right) F(Y, nh_{rk} + h_{rk}c), \tag{D.19}$$

wobei I_s die $s \times s$ Einheitsmatrix ist.

Der Ansatz ist ähnlich zu dem in [37] verwendeten, um die ISS von impliziten RK-Methoden zu zeigen und basiert auf der Abschätzung der rechten Seite von

$$\begin{aligned} &\langle x((n+1)h_{rk}), x((n+1)h_{rk})\rangle - k\langle x(nh_{rk}), x(nh_{rk})\rangle \\ &= \underbrace{\langle x((n+1)h_{rk}), x((n+1)h_{rk})\rangle - k\langle x(nh_{rk}), x(nh_{rk})\rangle}_{1.} \\ &\underbrace{-2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle}_{2.} + \underbrace{2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle}_{3.} \end{aligned} \tag{D.20}$$

mit beliebigen Zahlen $d_1, \ldots, d_s$ wobei gilt $d_i \geq 0$, $0 < k < 1$ und

$$f(i) = f(Y(i), u(nh_{rk} + c_i h_{rk})). \tag{D.21}$$

Die Addition und Subtraktion des 2. bzw. 3. Terms der rechten Seite von (D.20) ist nötig, damit die Runge-Kutta Parameter expizit erscheinen und die Annahmen aus Theorem 14 angewandt werden können.

Das Ziel ist es die Abhängigkeiten der rechten Seite von (D.20) auf den Funktionswert $x(nh_{rk})$ und u beziehungsweise $\dot{x} = f(x, u)$ innerhalb des Intervalls h_{rk} zu reduzieren. Das heißt die Abhängigkeit der rechten Seite von (D.20) von $x((n+1)h_{rk})$ und von $Y(i)$ muss vermieden werden. Dies wird erreicht indem im 1. und 2. Term der rechten Seite von (D.20) $x((n+1)h_{rk})$ und $Y(i)$ substituiert werden und der 3. Term abgeschätzt wird. Die Abschätzung wird möglich, indem berücksichtigt wird, dass (D.12) Lipschitz-stetig und ISS ist sowie (D.15) und (D.16) gilt.

Entsprechend gliedert sich der Beweis in 4 Schritte:

1. Substitution von $x((n+1)h_{rk})$ im 1. Term.
2. Substitution von $Y(i)$ im 2. Term.
3. Abschätzung des 3. Terms mit (D.15), (D.16) und (D.14).
4. Abschätzung von (D.20) mithilfe der Ergebnisse der ersten drei Schritte, wodurch es möglich wird B-ISS zu zeigen.

1. Schritt: Substitution von $x((n+1)h_{rk})$ im 1. Term.
Mit (D.17a) kann berechnet werden

$$\begin{aligned}
&\langle x((n+1)h_{rk}), x((n+1)h_{rk})\rangle - k\langle x(nh_{rk}), x(nh_{rk})\rangle \\
&= \langle x(nh_{rk}), x(nh_{rk})\rangle + 2\langle x(nh_{rk}), h_{rk}\sum_{i=1}^{s} b_i f(i)\rangle + \langle h_{rk}\sum_{i=1}^{s} b_i f(i), h_{rk}\sum_{i=1}^{s} b_i f(i)\rangle \\
&\quad - k\langle x(nh_{rk}), x(nh_{rk})\rangle \\
&= (1-k)\langle x(nh_{rk}), x(nh_{rk})\rangle + 2h_{rk}\sum_{i=1}^{s} b_i\langle x(nh_{rk}), f(i)\rangle + h_{rk}^2\sum_{i,j=1}^{s} b_i b_j\langle f(i), f(j)\rangle \quad \text{(D.22)}
\end{aligned}$$

2. Schritt: Substitution von $Y(i)$ im 2. Term
Mit (D.17b) kann berechnet werden

$$-2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle = -2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), x(nh_{rk})\rangle - 2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), h_{rk}\sum_{j=1}^{s} a_{ij} f(j)\rangle \tag{D.23}$$

3. Schritt: Abschätzung des 3. Terms mit (D.15), (D.16) und (D.14)
Nachdem für das Modell (D.17) die Annahme (D.15) als erfüllt vorausgesetzt wird und $f(i) = f(Y(i), u(nh_{rk} + c_i h_{rk}))$, gilt in Analogie zu (D.13b) die folgende Abschätzung

$$\langle f(i), Y(i)\rangle \leq -\beta(t)\langle Y(i), Y(i)\rangle + \gamma\langle u(i), u(i)\rangle \quad \forall i,$$

mit $u(i) = u(nh_{rk} + c_i h_{rk})$. Entsprechend gilt für den 3. Term von (D.20)

$$2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle \leq 2h_{rk}\sum_{i=1}^{s} d_i\left(-\beta(t)\langle Y(i), Y(i)\rangle + \gamma\langle u(i), u(i)\rangle\right).$$

Um die unbekannte Vergleichsfunktion $\beta(t)$ zu substituieren, wird der Zusammenhang (D.16) nach $\beta(t)$ aufgelöst

$$-\beta(t) = \frac{l}{h_{rk}} - h_{rk}\mu(t)m.$$

Setzt man dies für $\beta(t)$ ein, gilt mit dem Zusammenhang (D.14)

$$-\langle \dot{x}, \dot{x}\rangle + \sigma(P)L_u^2\langle u, u\rangle \geq -\mu(t)\langle x, x\rangle$$

und somit mit $\overline{\gamma} = \gamma + \sigma(P)L_u^2$

$$\begin{aligned}
2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle \leq& 2l\sum_{i=1}^{s} d_i\left(\langle Y(i), Y(i)\rangle + \gamma\langle u(i), u(i)\rangle\right) - 2h_{rk}^2 m\sum_{i=1}^{s} d_i\mu(i)\langle Y(i), Y(i)\rangle \\
\overset{\text{(D.14)}}{\leq}& 2l\sum_{i=1}^{s} d_i\left(\langle Y(i), Y(i)\rangle + \overline{\gamma}\langle u(i), u(i)\rangle\right) - 2h_{rk}^2 m\sum_{i=1}^{s} d_i\langle f(i), f(i)\rangle.
\end{aligned} \tag{D.24}$$

Nun wird mit (D.17b) $Y(i)$ substituiert

$$\begin{aligned}
2l\sum_{i=1}^{s} d_i\langle Y(i),Y(i)\rangle =& 2l\sum_{i=1}^{s} d_i\langle x(nh_{rk})+h_{rk}\sum_{j=1}^{s} a_{ij}f(j), x(nh_{rk})+h_{rk}\sum_{j=1}^{s} a_{ij}f(j)\rangle \\
=& 2l\sum_{i=1}^{s} d_i\langle x(nh_{rk}),x(nh_{rk})\rangle + 4l\sum_{i=1}^{s} d_i\langle x(nh_{rk}),h_{rk}\sum_{j=1}^{s} a_{ij}f(j)\rangle \\
&+ 2l\sum_{m=1}^{s} d_m\langle h_{rk}\sum_{j=1}^{s} a_{mj}f(j), h_{rk}\sum_{j=1}^{s} a_{mj}f(j)\rangle \\
=& 2l\sum_{i=1}^{s} d_i\langle x(nh_{rk}),x(nh_{rk})\rangle + 4hl\sum_{i=1}^{s} d_i\sum_{j=1}^{s}\langle x(nh_{rk}),a_{ij}f(j)\rangle \\
&+ 2h_{rk}^2 l\sum_{m=1}^{s} d_m\sum_{i,j=1}^{s} a_{mi}a_{mj}\langle f(i),f(j)\rangle. \qquad \text{(D.25)}
\end{aligned}$$

Wird (D.25) in (D.24) eingesetzt, gilt folgende Abschätzung

$$\begin{aligned}
2h_{rk}\sum_{i=1}^{s} d_i\langle f(i),Y(i)\rangle \leq& 2l\sum_{i=1}^{s} d_i\left(\langle Y(i),Y(i)\rangle+\overline{\gamma}\langle u(i),u(i)\rangle\right) - 2h_{rk}^2 m\sum_{i=1}^{s} d_i\langle f(i),f(i)\rangle \\
\leq& 2l\sum_{i=1}^{s} d_i\langle x(nh_{rk}),x(nh_{rk})\rangle + 4h_{rk}l\sum_{i=1}^{s} d_i\sum_{j=1}^{s}\langle x(nh_{rk}),a_{ij}f(j)\rangle \\
&+ 2h_{rk}^2 l\sum_{m=1}^{s} d_m\sum_{i,j=1}^{s} a_{mi}a_{mj}\langle f(i),f(j)\rangle - 2h_{rk}^2 m\sum_{i=1}^{s} d_i\langle f(i),f(i)\rangle \\
&+ 2l\sum_{i=1}^{s} d_i\overline{\gamma}\langle u(i),u(i)\rangle. \qquad \text{(D.26)}
\end{aligned}$$

4. Schritt: Abschätzung von (D.20)

Mit den Zusammenhängen (D.22), (D.23) und (D.26) erhält man folgende Abschätzung

$$
\begin{aligned}
&\langle x((n+1)h_{rk}), x((n+1)h_{rk})\rangle - k\langle x(nh_{rk}), x(nh_{rk})\rangle \\
&= \underbrace{\langle x((n+1)h_{rk}), x((n+1)h_{rk})\rangle - k\langle x(nh_{rk}), x(nh_{rk})\rangle}_{1.} \\
&\quad \underbrace{-2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle}_{2.} + \underbrace{2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), Y(i)\rangle}_{3.} \\
&\leq (1-k)\langle x(nh_{rk}), x(nh_{rk})\rangle + 2h_{rk}\sum_{i=1}^{s} b_i\langle x(nh_{rk}), f(i)\rangle + h_{rk}^2\sum_{i,j=1}^{s} b_ib_j\langle f(i), f(j)\rangle \\
&\quad - 2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), x(nh_{rk})\rangle - 2h_{rk}\sum_{i=1}^{s} d_i\langle f(i), h_{rk}\sum_{j=1}^{s} a_{ij}f(j)\rangle \\
&\quad + 2l\sum_{i=1}^{s} d_i\langle x(nh_{rk}), x(nh_{rk})\rangle + 4hl\sum_{i=1}^{s} d_i\sum_{j=1}^{s}\langle x(nh_{rk}), a_{ij}f(j)\rangle \\
&\quad + 2h^2l\sum_{m=1}^{s} d_m\sum_{i,j=1}^{s} a_{mi}a_{mj}\langle f(i), f(j)\rangle - 2h_{rk}^2m\sum_{i=1}^{s} d_i\langle f(i), f(i)\rangle + 2l\sum_{i=1}^{s} d_i\overline{\gamma}\langle u(i), u(i)\rangle \\
&= \left(1-k+2l\sum_{i=1}^{s} d_i\right)\langle x(nh_{rk}), x(nh_{rk})\rangle + 2h_{rk}\sum_{i=1}^{s}\langle x(nh_{rk}), f(i)\rangle\left(b_i - d_i + 2l\sum_{j=1}^{s} d_ja_{ij}\right) \\
&\quad + h_{rk}^2\sum_{i,j=1}^{s}\langle f(i), f(j)\rangle\left(b_ib_j - 2d_ia_{ij} + 2l\sum_{m=1}^{s} d_ma_{mi}a_{mj}\right) - 2h_{rk}^2\sum_{i=1}^{s} md_i\langle f(i), f(i)\rangle \\
&\quad + 2l\sum_{i=1}^{s} d_i\overline{\gamma}\langle u(i), u(i)\rangle \\
&= -\alpha\langle x(nh_{rk}), x(nh_{rk})\rangle - 2h_{rk}\sum_{i=1}^{s}\nu_i\langle x(nh_{rk}), f(i)\rangle - h_{rk}^2\sum_{i,j=1}^{s} w_{ij}\langle f(i), f(j)\rangle \\
&\quad + 2l\overline{\gamma}\sum_{i=1}^{s} d_i\langle u(i), u(i)\rangle.
\end{aligned}
$$

mit den Abkürzungen

$$
\begin{aligned}
\alpha &= k - 1 - 2l\sum_{i=1}^{s} d_i \\
\nu_i &= d_i - b_i - 2l\sum_{j=1}^{s} d_ja_{ij} \\
w_{ij} &= \begin{cases} d_ia_{ij} + d_ja_{ji} - b_ib_j - 2l\sum_{m=1}^{s} d_ma_{mi}a_{mj} + md_i & i = j \\ d_ia_{ij} + d_ja_{ji} - b_ib_j - 2l\sum_{m=1}^{s} d_ma_{mi}a_{mj} & \text{sonst} \end{cases},
\end{aligned}
$$

die den Einträgen der Matrix

$$M(k,l,m) = \begin{bmatrix} \alpha & \nu^\top \\ \nu & W \end{bmatrix}$$

aus Definition 11 entsprechen. Dabei sind in Definition 11 die Matrizen D eine positiv semidefinite Diagonalmatrix mit den Einträgen $d_1, \ldots, d_s$ und A die Matrix des Butcher-Arrays mit den den Einträgen a_{ij}. Da das Runge-Kutta Verfahren (k,l,m)-algebraisch stabil ist, ist die Matrix M nicht-negativ. Mit dem Zusammenhang für positiv definite Matrizen $M, N > 0 \Rightarrow M \otimes N > 0$ [36] erhält man mit $z = [x^\top(nh_{rk}), hf^\top(1), \ldots, hf^\top(s)]^\top$

$$\begin{aligned}\langle x((n+1)h_{rk}), x((n+1)h_{rk})\rangle \leq & k\langle x(nh_{rk}), x(nh_{rk})\rangle - z^\top (M \otimes I)\, z + 2l\overline{\gamma}\sum_{i=1}^{s} d_i\langle u(i), u(i)\rangle \\ \leq & k\langle x(nh_{rk}), x(nh_{rk})\rangle + 2l\overline{\gamma}\sum_{i=1}^{s} d_i\langle u(i), u(i)\rangle.\end{aligned}$$

Mit $V(n) = \langle x(nh_{rk}), x(nh_{rk})\rangle$, $u(i) = u(nh_{rk} + c_i h_{rk})$ und der Annahme $0 < k < 1$ wird folgende Abschätzung durchgeführt

$$\begin{aligned}V(n+1) - V(n) \leq & (k-1)\langle x(nh_{rk}), x(nh_{rk})\rangle + 2l\overline{\gamma}\sum_{i=1}^{s} d_i\langle u(nh_{rk} + c_i h_{rk}), u(nh_{rk} + c_i h_{rk})\rangle \\ \leq & (k-1)\lambda_{\max}(P)\,||x(nh_{rk})||^2 + 2l\lambda_{\max}(P)\overline{\gamma}\sum_{i=1}^{s} d_i\,||u(nh_{rk})||_\infty^2,\end{aligned}$$

womit $V(n)$ nach Definition 7 eine gültige Lyapunov-Funktion ist und nach Theorem 10 der Zusammenhang (D.4) erfüllt ist. Nach Definition 12 ist der Beweis erbracht.

□

Die Annahmen des Theorems 14, dass $l = -h_{rk}\beta(t) + h_{rk}^2\mu(t)m$ garantiert, dass (k,l,m)-algebraische Stabilität stets sichergestellt ist. Dies erkennt man leicht, indem man l in (D.11) einsetzt. Dies muss auch so sein, da ISS stets GAS sicherstellt [81].

Theorem (14) erweitert die Ergebnisse von [43] auf explizite RK-Methoden und beliebige innere Produkte. Zudem erlaubt Theorem (14) die Verwendung einer zeitabhängigen Vergleichsfunktionen $\beta(t)$. Für implizite RK Methoden gilt $m = 0$ und mit $P = I$ und $\beta(t) = const$, ist das Ergebnis identisch zu [43].

Die in Theorem (14) formulierte Bedingung an l ist abhängig vom unbekannten μ. Diese Abhängigkeit lässt sich durch eine geeignete Wahl der Vergleichsfunktion $\beta(t)$ vermeiden.

Wahl der Vergleichsfunktion $\beta(t)$

Die Abtastzeit muss so gewählt werden, dass gilt

$$\langle x, f(x,u)\rangle \leq -\beta(t)\langle x, x\rangle + \gamma\langle u, u\rangle \qquad \forall t \in T, \tag{D.27}$$

wobei T das betrachtete Zeitintervall ist. Der Koeffizient l einer (k,l,m)-algebraisch stabilen RK-Methode ist immer begrenzt durch $l_{\min} \leq l = -h_{rk}\beta(t) + h^2\mu(t)m < 0$ [13].

Dabei hängt der Koeffizient $l_{\min}$ von der gewählten RK-Methode ab. Diese Einschränkung an $\beta(t)$ durch den Koeffizient l kann berücksichtigt werden, indem man

$$-\beta(t) = -h_{rk}\mu(t)m - \alpha,$$

wählt. Der Koeffizient $\alpha > 0$ wird hier als beliebig klein angenommen. Mit dieser Wahl der Vergleichsfunktion β ist die Bedingung $l_{\min} < l < 0$ immer erfüllt, da $l = -h_{rk}\alpha$. Nun kann $\mu(t) = \frac{\langle \dot{x}, \dot{x} \rangle}{\langle x, x \rangle}$ gesetzt werden und da α beliebig klein ist, kann Bedingung (D.27) formuliert werden als

$$\langle x, f(x,u) \rangle < -h_{rk} m \langle \dot{x}, \dot{x} \rangle + \gamma \langle u, u \rangle \qquad \forall t \in T, \tag{D.28}$$

Somit erhält man folgendes Theorem:

Theorem 15. *Ein (k, l, m)-algebraisch stabiles RK-Verfahren mit $l < 0$ und $0 < k < 1$ ist B-ISS, wenn das Modell (D.12)*

$$\langle x, f(x,u) \rangle < -h_{rk} m \langle f(x,u), f(x,u) \rangle + \gamma \langle u, u \rangle \tag{D.29}$$

zu jeder Zeit erfüllt.

Nun gibt es zwei Möglichkeiten:

1. die Abtastzeit h_{rk} wird fixiert und die Modellgewichte so gewählt, dass (D.29) hält,
2. die Modellgewichte müssen so gewählt werden, dass $\langle x, f(x,u) \rangle < 0$ erfüllt ist und für jede Abtastung wird das Abtastintervall h_{rk} so festgelegt, dass (D.29) gilt.

Im zweiten Fall ist die Schrittweite h_{rk} abhängig von x und u. Um die maximale Schrittweite zu bestimmen, so dass noch ISS garantiert ist, können die Verfahren des *self-triggered control* verwendet (siehe beispielsweise [54] und die darin enthaltenen Referenzen) oder der folgende simple Algorithmus verwendet werden:

Algorithmus 2 Schrittweitenselektion Runge-Kutta Verfahren

In jedem Schritt der Berechnung:

$t_{rk} = 2h_{rk}$
if $\langle x, f(x,u) \rangle < -t_{rk} m \langle f(x,u), f(x,u) \rangle + \gamma \langle u, u \rangle$ **then**
  $h_{rk} = t_{rk}$
else
  while $\langle x, f(x,u) \rangle >= -t_{rk} m \langle f(x,u), f(x,u) \rangle + \gamma \langle u, u \rangle$ **do**
    $t_{rk} = t_{rk}/2$
  end while
  $h_{rk} = t_{rk}$
end if

Da die Kommunikationsstruktur am Prüfstand eine Festlegung der Abtastzeit bereits zu Beginn nötig macht, beschränken sich die folgenden Abschnitte auf die erste Möglichkeit.

Das allgemeine Ergebnis aus Theorem 15 wird im Folgendem auf RKNN angewandt, um Bedingungen an die Netzgewichte zu gewinnen, welche B-ISS sicherstellen.

D.4.2 Stabilitätsnebenbedingungen an die Gewichte des Runge-Kutta Neuronalen Netzes

In diesem Abschnitt wird ISS von zeitkontinuierlichen nichtlinearen Black-Box Modellen, welche durch eine RK-Methode diskretisiert werden, untersucht. Als Black-Box Modell wird das RKNN bestehend aus einem MLP-Netz und einem s-stufigen RK-Verfahren verwendet [15]. Das RKNN wird beschrieben durch

$$\hat{x}((i+1)h_{rk}) = \hat{x}(ih_{rk}) + h_{rk}\sum_{j=1}^{s} b_j k_j, \tag{D.30}$$

wobei

$$\begin{aligned}
k_1 =& f\left(\hat{x}(ih_{rk}), u(ih_{rk})\right)\\
k_2 =& f\left(\hat{x}(ih_{rk}) + h_{rk}a_{21}k_1, u(ih_{rk})\right)\\
k_3 =& f\left(\hat{x}(ih_{rk}) + h_{rk}\left[a_{31}k_1 + a_{32}k_2\right], u(ih_{rk})\right)\\
\vdots =& \vdots\\
k_s =& f\left(\hat{x}(ih_{rk}) + h_{rk}\sum_{j=1}^{s-1} a_{si}k_j, u(ih_{rk})\right).
\end{aligned}$$

Als nichtlineares zeitkontinuierliches Modell wird das MLP verwendet (dies entspricht (5.2)), welches durch folgende Differentialgleichung beschrieben wird [15]

$$\dot{\hat{x}} = f(\hat{x}, u) = W^{l,x}\hat{x} + W^{l,u}u + W^{o}\sigma(W^{h,x}\hat{x} + W^{h,u}u + b^h) + b^o,$$

wobei $x \in \mathbb{R}^n$, $\sigma : v \mapsto [\tanh(v_1), \ldots, \tanh(v_N)]^\top$, N die Anzahl der Neuronen ist und $-1 \leq \tanh(x) = 1 - \frac{2}{1+e^{-2x}} \leq 1$. Die Gewichtsmatrizen W und die Vektoren b sind zu optimieren, so dass der Ausgang des RKNN (D.30) die Systemzustände (2.2) so gut wie möglich approximiert und ISS von (D.30) sichergestellt ist.

Um die Bedingungen an die Modellgewichte herzuleiten, muss zunächst der Ruhezustand x^*, u^* durch Koordinatentransformation in den Ursprung überführt werden. Der Eingang u wird dabei als beliebig angenommen. Für den Ruhezustand gilt

$$W^{l,x}x^* + W^{l,u}u^* + W^{o}\sigma(W^{h,x}x^* + W^{h,u}u^* + b^h) + b^o = 0$$

und die Variablentransformation $y = \hat{x} - x^*$, $w = u - u^*$ führt zu

$$\begin{aligned}
\dot{y} =& W^{l,x}(y + x^*) + W^{l,u}(w + u^*) + W^{o}\sigma(W^{h,x}(y + x^*) + W^{h,u}(w + u^*) + b^h) + b^o\\
&- \left(W^{l,x}x^* + W^{l,u}u^* + W^{o}\sigma(W^{h,x}x^* + W^{h,x}u^* + b^h) + b^o\right)\\
\dot{y} =& W^{l,x}y + W^{l,u}w + W^{o}\sigma(W^{h,x}(y + x^*) + W^{h,u}(w + u^*) + b^h)\\
&- W^{o}\sigma(W^{h,x}x^* + W^{h,x}u^* + b^h).
\end{aligned}$$

Nun kann ausgenützt werden, dass der Tangens Hyperbolicus $\tanh(\cdot)$ Lipschitz-stetig mit

Lipschitz-Konstante l ist

$$(\tanh(x) - \tanh(y)) \leq l\,(x - y)\,.$$

Zudem ist $\tanh(\cdot)$ eine monoton steigende Funktion, wodurch für $x \neq y$

$$0 < \frac{\tanh(x) - \tanh(y)}{x - y} < \bar{l} = 1$$

hält. Damit gilt

$$\sigma(W^{h,x}(y+x^*)+W^{h,u}(w+u^*)+b^h)-\sigma(W^{h,x}x^*+W^{h,x}u^*+b^h) = L(y,w)(W^{h,x}y+W^{h,u}w)$$

mit $L(y,w) = \operatorname{diag}\left((l_1(y,w),\ldots,l_{N_N}(y,w))\right)$ $, l_i(y,w) \in]0;1[\ \forall y, \forall w.$

Nach Theorem 15 müssen die Gewichte so gewählt werden, dass

$$\langle y, f(y,w)\rangle + h_{rk} m \langle f(y,w), f(y,w)\rangle < \gamma \langle w, w\rangle \tag{D.31}$$

hält, wobei

$$f(y,w) = W^{l,x}y + W^{l,u}w + W^o L(y,w)(W^{h,x}y + W^{h,u}w).$$

Mit diesem Ergebnis können die Bedingungen an die Gewichte bestimmt werden, welche ISS des RKNN sicherstellen.

Theorem 16 (ISS von RKNN)**.** *Das RKNN (D.30) ist für ein (k,l,m)-algebraisch stabiles RK-Verfahren mit $l < 0$ und $0 < k < 1$ ISS, wenn positiv definite Matrizen P, Λ, $Q = (P + P\Lambda P)$ und Gewichtungsvariablen $\sum \alpha_i = 1$, $\alpha_i \geq 0\ \forall i$ existieren, so dass entweder*

$$P\Lambda P + PW^{l,x} + W^{l,x\top}P + 2h_{rk}m\left(W^{l,x\top}QW^{l,x} + C\right) < 0 \tag{D.32a}$$

$$\begin{aligned}&\left(2h_{rk}mW^{l,x\top}Q + P\right)W^oW_i^{h,x} + \left(\left(2h_{rk}mW^{l,x\top}Q + P\right)W^oW_i^{h,x}\right)^\top \\ &+\alpha_i P\Lambda P + \alpha_i PW^{l,x} + \alpha_i W^{l,x\top}P + 2\alpha_i h_{rk}m\left(W^{l,x\top}QW^{l,x} + C\right) \leq 0 \quad \forall i\end{aligned} \tag{D.32b}$$

erfüllt sind, oder

$$P\Lambda P + PW^{l,x} + W^{l,x\top}P + 2h_{rk}m\left(W^{l,x\top}QW^{l,x} + C\right) \leq 0 \tag{D.33a}$$

$$\begin{aligned}&\left(2h_{rk}mW^{l,x\top}Q + P\right)W^oW_i^{h,x} + \left(\left(2h_{rk}mW^{l,x\top}Q + P\right)W^oW_i^{h,x}\right)^\top \\ &+\alpha_i P\Lambda P + \alpha_i PW^{l,x} + \alpha_i W^{l,x\top}P + 2\alpha_i h_{rk}m\left(W^{l,x\top}QW^{l,x} + C\right) \leq 0 \quad \forall i\end{aligned} \tag{D.33b}$$

$$\begin{aligned}&\left(2h_{rk}mW^{l,x\top}Q + P\right)W^oW_q^{h,x} + \left(\left(2h_{rk}mW^{l,x\top}Q + P\right)W^oW_q^{h,x}\right)^\top \\ &+\alpha_q P\Lambda P + \alpha_q PW^{l,x} + \alpha_q W^{l,x\top}P + 2\alpha_q h_{rk}m\left(W^{l,x\top}QW^{l,x} + C\right) < 0 \quad \exists q,\end{aligned} \tag{D.33c}$$

erfüllt sind. Darin sind

$$C = \left|\left|W^{o\top}QW^o\right|\right| \left|\left|W^{h,x}\right|\right|^2 I$$

mit Einheitsmatrix I und

$$W_i^{h,x} = \begin{bmatrix} & 0_{i-1\times N_x} & \\ W_{i,1}^{h,x} & \dots & W_{i,N_x}^{h,x} \\ & 0_{N_N-i\times N_x} & \end{bmatrix}.$$

Beweis. Der Beweis basiert auf einer Abschätzung von (D.31), so dass Matrixungleichungen als Nebenbedingungen an die Gewichte formuliert werden können. Zunächst werden die Mischterme, welche von y und w abhängen abgeschätzt, und anschließend $L(y,w)$ abgeschätzt.

Um die Notation einfach zu halten, werden $L = L(y,w)$, $A = A(y,w) = W^{l,x} + W^o L(y,w) W^{h,x}$ und $B = B(y,w) = W^{l,u} + W^o L(y,w) W^{h,u}$ gesetzt. Nun wird

$$\langle y, f(y,w)\rangle = y^\top PAy + y^\top PBw$$

berechnet und

$$\langle f(y,w), f(y,w)\rangle = y^\top A^\top PAy + 2y^\top A^\top PBw + w^\top B^\top PBw,$$

wodurch sich folgender Ausdruck

$$\begin{aligned}\langle y, f(y,w)\rangle + h_{rk} m \langle f(y,w), f(y,w)\rangle =& y^\top (PA + h_{rk} m A^\top PA) y + y^\top (PB + 2h_{rk} m A^\top PB) w \\ &+ h_{rk} m w^\top B^\top PBw \end{aligned} \tag{D.34}$$

ergibt, der im folgenden abgeschätzt wird. Um den Term $y^\top(PB + 2h_{rk}mA^\top PB)w$ abzuschätzen, wird der folgende Zusammenhang verwendet

$$X^\top Y + (X^\top Y)^\top \leq X^\top \Lambda X + Y^\top \Lambda^{-1} Y, \tag{D.35}$$

der für jedes $X, Y \in \mathbb{R}^{n\times k}$, für jede positiv definite Matrix $0 < \Lambda = \Lambda^\top \in \mathbb{R}^{n\times n}$ gilt. Dieser Zusammenhang folgt leicht aus der Beobachtung 7.1.6 und Theorem 7.7.7 in [36]. Nun kann mit (D.35) und $2y^\top Ay = y^\top (A + A^\top) y$

$$\begin{aligned} &2y^\top (PB + 2h_{rk} m A^\top PB) w = y^\top (PB + (PB)^\top + 2h_{rk} m (A^\top PB) + 2h_{rk} m (A^\top PB)^\top) w \\ &\leq y^\top (P\Lambda P + 2h_{rk} m A^\top P \Lambda PA) y + w^\top (B^\top \Lambda^{-1} B + 2h_{rk} m B^\top \Lambda^{-1} B) w \end{aligned}$$

berechnet werden. Mit diesem Zusammenhang kann (D.34) abgeschätzt werden mit

$$\begin{aligned} &2\langle y, f(y,w)\rangle + 2h_{rk} m \langle f(y,w), f(y,w)\rangle \\ &\leq y^\top (PA + (PA)^\top + 2h_{rk} m A^\top PA) y + y^\top (P\Lambda P + 2h_{rk} m A^\top P\Lambda PA) y \\ &\quad + w^\top (B^\top \Lambda^{-1} B + 2h_{rk} m B^\top \Lambda^{-1} B) w + 2h_{rk} m w^\top B^\top PBw. \end{aligned}$$

Wird nun

$$\gamma = \max_{y,w} \left|\left| B(y,w)^\top (\Lambda^{-1} + 2h_{rk} m (\Lambda^{-1} + P)) B(y,w) \right|\right|$$

gesetzt und mit (D.31) ist leicht ersichtlich, dass

$$y^\top(PA + (PA)^\top + P\Lambda P + 2h_{rk}mA^\top(P + P\Lambda P)A)y < 0 \tag{D.36}$$

ISS sicherstellt. Der nächste Schritt besteht in der Abschätzung von $L(y, w)$. Zunächst wird A resubstituiert und $Q = (P + P\Lambda P)$ eingeführt, um

$$PA + A^\top P = P(W^{l,x} + W^o L W^{h,x}) + (W^{l,x} + W^o L W^{h,x})^\top P$$

und

$$\begin{aligned} A^\top Q A =& (W^{l,x} + W^o L W^{h,x})^\top Q (W^{l,x} + W^o L W^{h,x}) \\ =& {W^{l,x}}^\top Q W^{l,x} + {W^{l,x}}^\top Q W^o L W^{h,x} + (W^o L W^{h,x})^\top Q W^{l,x} + (W^o L W^{h,x})^\top Q W^o L W^{h,x} \end{aligned}$$

zu berechnen. In zwei Schritten wird nun $L(y, w)$ eliminiert. Da $||L(y, w)|| \leq 1$ gilt, wird der positiv definite Term

$$y^\top (W^o L W^{h,x})^\top Q W^o L W^{h,x} y = y^\top C(y, w) y \leq y^\top C y \tag{D.37}$$

mit

$$C(y, w) = (W^o L W^{h,x})^\top Q W^o L W^{h,x} \tag{D.38}$$

$$C = \left|\left|{W^o}^\top Q W^o\right|\right| \left|\left|W^{h,x}\right|\right|^2 I \tag{D.39}$$

abgeschätzt, wobei I die Einheitsmatrix ist. Um die restlichen Terme unabhängig von $L(y, w)$ abzuschätzen, wird dem Ansatz von [38] gefolgt und die folgende Matrix

$$W_i^{h,x} = \begin{bmatrix} & 0_{i-1\times N_x} & \\ W_{i,1}^{h,x} & \cdots & W_{i,N_x}^{h,x} \\ & 0_{N_N - i\times N_x} & \end{bmatrix} \tag{D.40}$$

eingeführt. Damit wird (D.36) zu

$$\begin{aligned} & y^\top \Bigg(P(W^{l,x} + \sum_{i=1}^{N_N} l_i(y, w) W^o W_i^{h,x}) + (W^{l,x} + \sum_{i=1}^{N_N} l_i(y, w) W^o W_i^{h,x})^\top P + 2h_{rk}m {W^{l,x}}^\top Q W^{l,x} \\ & + P\Lambda P + 2h_{rk}m \left(\sum_{i=1}^{N_N} l_i(y, w) {W^{l,x}}^\top Q W^o W_i^{h,x} + \sum_{i=1}^{N_N} l_i(y, w)(W^o W_i^{h,x})^\top Q W^{l,x} + C(y, w) \right) \Bigg) y \\ & = y^\top \Bigg(\sum_{i=1}^{N_N} l_i(y, w) \left(\left(2h_{rk}m {W^{l,x}}^\top Q + P \right) W^o W_i^{h,x} + \left(\left(2h_{rk}m {W^{l,x}}^\top Q + P \right) W^o W_i^{h,x} \right)^\top \right) \\ & \quad + P\Lambda P + P W^{l,x} + {W^{l,x}}^\top P + 2h_{rk}m \left({W^{l,x}}^\top Q W^{l,x} + C(y, w) \right) \Bigg) y < 0. \end{aligned}$$

Indem die Gewichtungsvariablen $\sum \alpha_i = 1,\ 0 \leq \alpha_i \leq 1$ eingeführt werden, kann

$$
\begin{aligned}
&y^\top (PA + (PA)^\top + P\Lambda P + 2h_{rk} m A^\top (P + P\Lambda P) A) y \\
&= y^\top \Bigg(\sum_{i=1}^{N_N} l_i(y,w) \Bigg(\left(2h_{rk} m W^{l,x\top} Q + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk} m W^{l,x\top} Q + P\right) W^o W_i^{h,x}\right)^\top \Bigg) \\
&\qquad + P\Lambda P + P W^{l,x} + W^{l,x\top} P + 2h_{rk} m \left(W^{l,x\top} Q W^{l,x} + C(y,w)\right) \Bigg) y \\
&\leq y^\top \Bigg(\sum_{i=1}^{N_N} l_i(y,w) \Bigg(\left(2h_{rk} m W^{l,x\top} Q + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk} m W^{l,x\top} Q + P\right) W^o W_i^{h,x}\right)^\top \Bigg) \\
&\qquad + P\Lambda P + P W^{l,x} + W^{l,x\top} P + 2h_{rk} m \left(W^{l,x\top} Q W^{l,x} + C\right) \Bigg) y \\
&= y^\top \left(1 - \sum_{i=1}^{N_N} l_i(y,w)\alpha_i\right) \left(P\Lambda P + P W^{l,x} + W^{l,x\top} P + 2h_{rk} m \left(W^{l,x\top} Q W^{l,x} + C\right)\right) y \\
&\quad + \sum_{i=1}^{N_N} l_i(y,w) y^\top \Bigg(\left(2h_{rk} m W^{l,x\top} Q + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk} m W^{l,x\top} Q + P\right) W^o W_i^{h,x}\right)^\top \\
&\qquad + \alpha_i P\Lambda P + \alpha_i P W^{l,x} + \alpha_i W^{l,x\top} P + 2\alpha_i h_{rk} m \left(W^{l,x\top} Q W^{l,x} + C\right) \Bigg) y
\end{aligned}
$$

berechnet werden. Da $\left(1 - \sum_{i=1}^{N_N} l_i(y_i)\alpha_i\right) > 0$ ist mit (D.32a), (D.32b) oder (D.33a), (D.33b), (D.33c) die Bedingung D.36 für alle $0 < l_i(y,w) < 1$ erfüllt und der Beweis vollständig. □

Wenn man nur in GAS des RKNN interessiert ist, wird aus (D.34) und (D.35) ersichtlich, dass alle Terme die Λ enthalten, verschwinden und man erhält folgendes Theorem:

Theorem 17 (GAS von RKNN)**.** *Das RKNN (D.30) ist für ein (k,l,m)-algebraisch stabiles RK-Verfahren mit $l < 0,\ 0 < k < 1$ GAS, wenn eine positiv definite Matrix P und Gewichtungsvariablen $\sum \alpha_i = 1,\ 0 \leq \alpha_i \leq 1\ \forall i$ existieren, so dass entweder*

$$
P W^{l,x} + W^{l,x\top} P + 2h_{rk} m \left(W^{l,x\top} P W^{l,x} + C\right) \leq 0 \tag{D.41a}
$$

$$
\begin{aligned}
&\left(2h_{rk} m W^{l,x\top} P + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk} m W^{l,x\top} P + P\right) W^o W_i^{h,x}\right)^\top \\
&\qquad + \alpha_i P W^{l,x} + \alpha_i W^{l,x\top} P + 2\alpha_i h_{rk} m \left(W^{l,x\top} P W^{l,x} + C\right) \leq 0 \quad \forall i
\end{aligned} \tag{D.41b}
$$

erfüllt sind, oder

$$PW^{l,x} + {W^{l,x}}^\top P + 2h_{rk}m\left({W^{l,x}}^\top PW^{l,x} + C\right) \leq 0 \tag{D.42a}$$

$$\left(2h_{rk}m{W^{l,x}}^\top P + P\right) W^o W_i^{h,x} + \left(\left(2h_{rk}m{W^{l,x}}^\top P + P\right) W^o W_i^{h,x}\right)^\top + \alpha_i PW^{l,x} + \alpha_i {W^{l,x}}^\top P + 2\alpha_i h_{rk}m\left({W^{l,x}}^\top PW^{l,x} + C\right) \leq 0 \quad \forall i \tag{D.42b}$$

$$\left(2h_{rk}m{W^{l,x}}^\top P + P\right) W^o W_q^{h,x} + \left(\left(2h_{rk}m{W^{l,x}}^\top P + P\right) W^o W_q^{h,x}\right)^\top + \alpha_q PW^{l,x} + \alpha_q {W^{l,x}}^\top P + 2\alpha_q h_{rk}m\left({W^{l,x}}^\top PW^{l,x} + C\right) < 0 \quad \exists q, \tag{D.42c}$$

erfüllt sind. Darin sind

$$C = \left|\left|{W^o}^\top PW^o\right|\right| \left|\left|W^{h,x}\right|\right|^2 I$$

mit Einheitsmatrix I und

$$W_i^{h,x} = \begin{bmatrix} & 0_{i-1\times N_x} & \\ W_{i,1}^{h,x} & \dots & W_{i,N_x}^{h,x} \\ & 0_{N_N-i\times N_x} & \end{bmatrix}.$$

Anmerkung: Mit $h = 0$, $\alpha_i = \frac{1}{N_N}$, $N_N = N_x$, $W^o = I$ und $W^{l,x} = \operatorname{diag}(d_1, \dots, d_{N_x})$, $d_i > 0$ entsprechen die Bedingungen (D.41a), (D.41b) beziehungsweise (D.42a), (D.42b), (D.42c) den Ergebnissen von [38]. Dementsprechend ist Theorem 16 eine Verallgemeinerung der in [38] vorgestellten globalen asymptotischen Stabilität (GAS) von zeitkontinuierlichen Neuronalen Netzen.

E Benchmarksystem

Das dynamische Testsystem soll im Verhalten und der Komplexität dem Verbrennungsmotor nahe kommen und sich eignen, die für die Aufgabenstellung relevanten Aspekte zu untersuchen. So soll zum einem das dynamische Verhalten approximiert und andererseits der Stationärwert abgeschätzt und gegebenenfalls optimiert werden. Resultierend aus den Anforderungen der Aufgabenstellung soll das dynamische Testsystem die folgenden Kriterien erfüllen:

- Nichtlinearität,
- Stabililität,
- System in Ruhelage soll als Benchmark für die Optimierung fungieren.

Damit das System in der Ruhelage als Benchmark für eine Optimierung dienen kann, wird das System durch zwei verkettete Abbildungen beschrieben

$$x = f(u) \qquad \dot{y} = h(y, x).$$

Die erste Abbildung bildet $\mathbb{R}^n$ auf $\mathbb{R}$ ab und die zweite Abbildung $h(x) : \mathbb{R} \to \mathbb{R}$ wird durch eine Differentialgleichung erster Ordnung beschrieben. Für die erste Abbildung wird die Radcos-Funktion [65] verwendet. Diese ist für den zweidimensionalen Fall an dem Zusammenhang zwischen Ventilsteuerzeiten und Verbrauch im aufgeladenem Bereich angelehnt. Für $u \in \mathbb{R}^2$ ist die Radcos-Funktion wie folgt definiert

$$x = \cos\left(9\sqrt{u_1^2 + u_2^2} + 2\right) + 0.5\cos(11u_1 + 2) + 15\left((u_1 - 0.4)^2 + (u_2 - 0.4)^2\right)^2.$$

Für $u \in \mathbb{R}^n$, $n > 2$ gilt:

$$\begin{aligned} x = &\cos\left(9\sqrt{u_1^2 + u_2^2} + 2\right) + \frac{1}{2}\cos\left(8x_1 \cdots (1 + x_3 + \ldots + x_n) + 2\right) + \\ &15\left((x_1 - 0.4)^2 + (x_2 - 0.4)^2\right)^2 + \\ &\frac{15}{\sqrt{d-2}}\sum_{j=3}^{n}\left(u_j - a_j \cdot (b_j u_1 + 1 - u_2) - 0.2\right)^2, \end{aligned}$$

wobei $b_j = 1 + 5(j-3)$ und $a_j = \frac{0.6}{1+b_j}$ für $3 \leq j \leq n$.

Die Funktion weist für $n \in \{3, 4, 5\}$ n und für $n \in \{2, 6, 7\}$ $n+1$ im Eingangsraum $u \in \mathbb{R}^n$ verteilte lokale Optima, die nicht an den Rändern des Definitionsbereichs liegen, auf. Die Funktionswerte liegen etwa zwischen -0.7 und -1.4. Ihr Maximum hat etwa den Funktionswert von $9.6\sqrt{n-2+0.5}$ [65]. Um zu garantieren, dass $-1 \leq x \leq 1$ für $-1 \leq u \leq 1$, wird

die Funktion entsprechend verschoben und skaliert. Die nichtlineare Dynamik ist dem Verhalten der Temperaturen angelehnt. Diese weisen ein PT1 Verhalten auf, allerdings ist die Zeitkonstante vom Betriebspunkt und somit von der Entalpie im Abgasstrom abhängig. Im Stationärpunkt soll $h(x, y)$ ein lineares Verhalten aufweisen, so dass eine strikte Trennung zwischen stationärer und dynamischer Nichtlinearität erfolgt. Um dies zu erreichen wird folgende Struktur für die Differentialgleichung angesetzt: $\dot{y}h(x, y) = Kx - y$, womit im eingeschwungenem Zustand $y_0 = Kx$ gilt. Als Differentialgleichung wird folgende Funktion gewählt

$$\dot{y} = \frac{x - y}{2.4\cos(10x + 4) - 0.5y + 4}.$$

Die Übertragungsfunktionen des Benchmarksystems sind in nachfolgender Abbildung E.1 dargestellt.

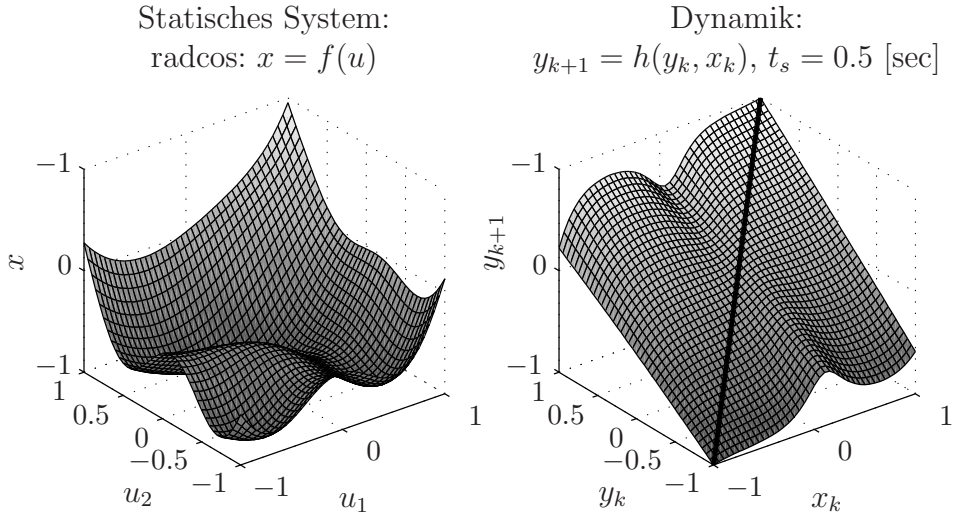


Abbildung E.1: Testsystem.

Die Zeitkonstanten des Systems liegen somit im Bereich $T \in [6,\ 13.2]$.

In Abbildung E.2 findet sich ein typisches Ausgangssignal des Benchmarksystems, welches mit Rauschen unterschiedlicher SNR beaufschlagt ist. Als Eingangssignal dient ein APRBS mit LHC-Verteilung der Punkte.

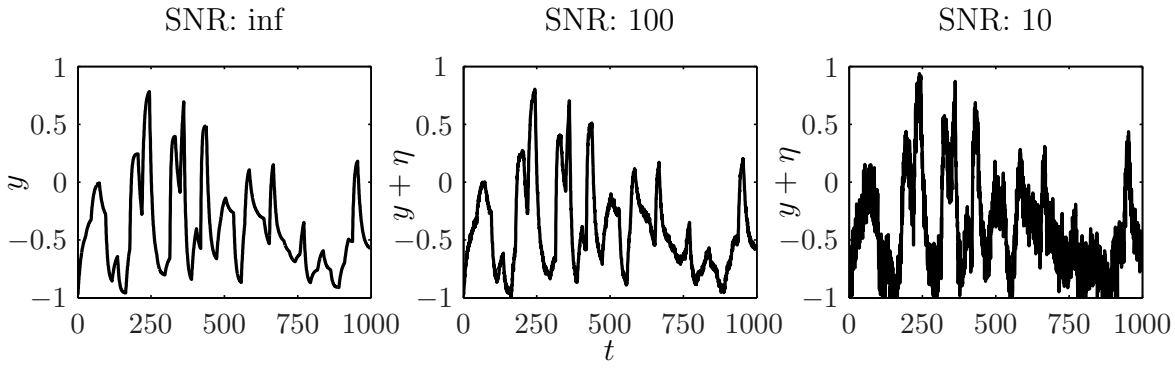


Abbildung E.2: Systemausgang des Benchmarksystems mit unterschiedlichem Rauschen.

Literaturverzeichnis

[1] N.K. Ahmed, A.F. Atiya, N. El Gayar, H. El-Shishiny, and E. Giza. An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29:594 – 621, 2010.

[2] H. Bandemer and A. Bellmann. *Statistische Versuchsplanung*. Teubner Verlag, 1994.

[3] W. Baumann, K. Klug, B.U. Köhler, and K. Röpke. Modeling of transient diesel engine emissions. In *Proceedings of the 5th Conference Design of Experiments (DoE) in Engine Development*, 2009.

[4] B. Berger, F. Rauscher, and B. Lohmann. Analysing Gaussian Processes for Stationary Black-Box Combustion Engine Modelling. In *18th IFAC World Congress, Milan, Italy*, 2011.

[5] C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc. New York, NY, USA, 1995.

[6] C.M. Bishop. *Pattern recognition and machine learning*. Springer New York, 2006.

[7] L. Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999–1013, 1993.

[8] H.-J. Bungartz, S. Zimmer, M. Buchholz, and D. Pflüger. *Modellbildung und Simulation: Eine anwendungsorientierte Einführung*. Springer Verlag, 2009.

[9] J.C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons Inc, 2003.

[10] R.H. Byrd, M.E. Hribar, and J. Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

[11] S. Chen. Multi-output regression using a locally regularised orthogonal least square algorithm. In *IEEE Proceedings Vision, Image and Signal Processing, Vol.149, No.4*, pages 185–195, 2002.

[12] T. Chen and B.A. Francis. *Optimal Sampled-Data Control Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1995.

[13] G.J. Cooper. A Generalization of Algebraic Stability for Runge–Kutta Methods. *IMA Journal of Numerical Analysis*, 4(4):427, 1984.

[14] O. De Jesus and M.T. Hagan. Backpropagation algorithms for a broad class of dynamic networks. *IEEE Transactions on Neural Networks*, 18(1):14–27, 2006.

[15] M. Deflorian. On runge-kutta neural networks: Training in series-parallel and parallel configuration. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 4480–4485. IEEE, 2010.

[16] M. Deflorian. Generalization of an input-to-state stability preserving Runge-Kutta method for nonlinear control systems. *Journal of Computational and Applied Mathematics, submitted*, 2011.

[17] M. Deflorian. Input to state stability of continuous time neural networks discretized by explicit Runge-Kutta methods. *IEEE Transactions on Neural Networks, submitted*, 2011.

[18] M. Deflorian. Runge-kutta neural networks for identification of dynamic models at the testbed. In *Proceedings of the 6th Conference Design of Experiments (DoE) in Engine Development*, 2011.

[19] M. Deflorian. Versuchplanung zur Identifikation nichtlinearer dynamischer Systeme. *at - Automatisierungstechnik*, eingereicht, 2011.

[20] M. Deflorian and F. Klöpper. Design of dynamic experiments. In *Proceedings of the 5th Conference Design of Experiments (DoE) in Engine Development*, 2009.

[21] M. Deflorian and F. Klöpper. Schnelles Training neuronaler Netze zur Identifikation nichtlinearer dynamischer Systeme. *at - Automatisierungstechnik*, 59:75–83, 2011.

[22] M. Deflorian, F. Klöpper, and J. Rückert. Online dynamic black box modelling and adaptive experiment design in combustion engine calibration. In *Proceedings of the IFAC Symposium Advances in Automotive Control*, 2010.

[23] M. Deflorian and S. Zaglauer. Design of experiments for nonlinear dynamic system identification. In *Proceedings of the IFAC World Congress*, volume 18, 2011.

[24] C. M. Douglas. *Design and Analysis of Experiments, 5th Edition*. Wiley New York, 2000.

[25] M.O. Efe and O. Kaynak. A comparative study of neural network structures in identification of nonlinear systems. *Mechatronics*, 9(3):287–300, 1999.

[26] S. Ernst. Hinging hyperplane trees for approximation and identification. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 2, 1998.

[27] J. Fan and J. Pan. A note on the Levenberg–Marquardt parameter. *Applied Mathematics and Computation*, 207(2):351–359, 2009.

[28] F.D. Foresee and M.T. Hagan. Gauss-Newton approximation to Bayesian learning. In *Proceedings of the 1997 International Joint Conference on Neural Networks*, volume 3, pages 1930–1935. Piscataway: IEEE, 1997.

[29] G. Goodwin and R. Payne. *Dynamic System Identification: Experiment Design and Data Analysis*. Academic Press, New York, 1977.

[30] M. Hafner. Effiziente Applikation der Motorsteuerung mit dynamischen Modellen. *at - Automatisierungstechnik*, 51:213–220, May 2003.

[31] M. Hafner, M. Schüler, O. Nelles, and R. Isermann. Fast neural networks for diesel engine control design. *Control Engineering Practice*, 8(11):1211–1221, November 2000.

[32] M.T. Hagan, H.B. Demuth, and O.D. Jesús. An introduction to the use of neural networks in control systems. *International Journal of Robust and Nonlinear Control*, 12(11):959–985, 2002.

[33] M.T. Hagan and M.B. Menhaj. Training feedforward networks with the Marquardt algorithm. *IEEE transactions on Neural Networks*, 5(6):989–993, 1994.

[34] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations I (2nd revised. ed.): nonstiff problems*. Springer-Verlag New York, Inc., New York, NY, USA, 1993.

[35] J. Hartung and B. Elpelt. *Multivariate Statistik: Lehr-und Handbuch der angewandten Statistik,15. Auflage*. Oldenbourg, 2009.

[36] R.A. Horn and C.R. Johnson. *Matrix analysis*. Cambridge Univ Pr, 1990.

[37] G.D. Hu and M. Liu. Input-to-state stability of Runge–Kutta methods for nonlinear control systems. *Journal of Computational and Applied Mathematics*, 205(1):633–639, 2007.

[38] S. Hu and J. Wang. Global stability of a class of continuous-time recurrent neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 49(9):1334–1341, 2002.

[39] X. Huo and X. Ni. When do stepwise algorithms meet subset selection criteria? *Annals of Statistics*, 35(2):870, 2007.

[40] R. Isermann. *Identifikation dynamischer Systeme-Band 1 und 2*, volume 10. Springer Verlag, 1992.

[41] S. Jakubek and C. Hametner. Identification of neurofuzzy models using GTLS parameter estimation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(5):1121–1133, 2009.

[42] S. Jakubek and N. Keuth. Optimierte Neuro-Fuzzy-Modelle für Auslegungsprozesse und Simulation im Automotive-Bereich. *at-Automatisierungstechnik*, 53(9):425–433, 2005.

[43] Z.P. Jiang and Y. Wang. Input-to-state stability for discrete-time nonlinear systems. *Automatica*, 37(6):857–869, 2001.

[44] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[45] N. Keuth, M. Thomas, H. Pflügl, E. Martini, and S. Bergold. Utilization of the Slow Dynamic Slope methology for the calibration of the ECU-functions Air Charge Determination and Torque Prediction in the series production. In *Proceedings of the 5th Conference Design of Experiments (DoE) in Engine Development*, 2009.

[46] H.K. Khalil. *Nonlinear systems*. Prentice hall Englewood Cliffs, NJ, 3. edition, 2002.

[47] F. Klöpper. *Entwicklung und Einsatz modellgestützer Online-Methoden zur Parametereroptimierung von Verbennungsmethoden am Prüfstand.* PhD thesis, Lehrstuhl für Verbrennungskraftmaschinen, TU-München, 2009.

[48] K. Knödler. *Methoden der restringierten Online-Optimierung zur Basisapplikation moderner Verbrennungsmotoren.* PhD thesis, Informations- und Kognitionswissenschaften, Eberhard-Karls-Universität Tübingen, 2004.

[49] I.J. Leontaritis and S.A. Billings. Input-output parametric models for non-linear systems part I: deterministic non-linear systems. *International Journal of Control*, 41(2):303–328, 1985.

[50] L. Ljung. *System Identification: Theory for the User (2nd Edition).* Prentice Hall PTR, December 1999.

[51] D.J.C. MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[52] I. Markovsky and S. van Huffel. Overview of total least squares methods. *Signal Processing*, 87(10):2283–2302, 2007.

[53] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[54] M. Mazo and P. Tabuada. On event-triggered and self-triggered control over sensor/actuator networks. In *47th IEEE Conference on Decision and Control*, pages 435–440. IEEE, 2008.

[55] T.J. Mitchell. An algorithm for the construction of d-optimal experimental designs. *Technometrics*, 16(2):203–210, 1974.

[56] M. Moll. Reglerentwurf basierend auf dynamischen nichtlinearen Black-Box Modellen. Master's thesis, Technische Universität München, 2011.

[57] J.J. Moré. The Levenberg-Marquardt algorithm: implementation and theory. *Lecture notes in mathematics*, 630:105–116, 1977.

[58] M.D. Morris and T.J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43:381–402, 1995.

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

[60] O. Nelles. *Nonlinear system identification.* Springer Verlag, Heidelberg, Germany, 2001.

[61] H. Niedernolte, F. Klöpper, A. Mitterer, and F. Schwarzer. Workflow for data evaluation during basic calibration of combustion engines. In *2006 IEEE International Conference on Control Applications*, pages 2060–2065, 2006.

[62] M. Norgaard, O.E. Ravn, N.K. Poulsen, and L.sK. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.

[63] M.P. Perrone and L.N. Cooper. When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone, editor, *Neural Networks for Speech and Image Processing*, pages 126–142. Chapman and Hall, 1993.

[64] K. Pfeil. Nichtlineare Identifikation des Luftpfads von aufgeladenen Dieselmotoren und automatisierter AGR-/VTG-Reglerentwurf. *at - Automatisierungstechnik*, 55:352 – 359, July 2007.

[65] J. Poland. *Modellgestützte und Evolutionäre Optimierungsverfahren für die Motorentwicklung*. PhD thesis, Fakultät für Informatik der Eberhard-Karls-Universität Tübingen, 2002.

[66] L. Pronzato. Asymptotic properties of nonlinear least squares estimates in stochastic regression models over a finite design space. Application to self-tuning optimisation. In *Proc. 15th IFAC Symposium on System Identification, Saint-Malo, France*, 2009.

[67] L. Pronzato. One-step ahead adaptive D-optimal design on a finite design space is asymptotically optimal. *Metrika*, 71(2):219–238, 2010.

[68] G.P. Rao and H. Unbehauen. Identification of continuous-time systems. *IEE Proceedings-Control theory and applications*, 153(2):185–220, 2006.

[69] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[70] P. Renninger and M. Aleksandrov. Rapid hull determination: a new method to determine the design space for model based approaches. In *Proceedings of the 3th Conference Design of Experiments (DoE) in Engine Development*, 2005.

[71] D.E. Rivera, H. Lee, M.W. Braun, and H.D. Mittelmann. Plant friendly system identification: A challenge for the process industries. In *Proceedings of SYSID*, Rotterdam, Netherlands, 2003.

[72] K. Röpke, M. Knaak., A. Nessler, and S. Schaum. Entwicklung-Mess-und Prüftechnik: Rapid Measurement–Grundbedatung eines Verbrennungsmotors innerhalb eines Tages? *MTZ-Motortechnische Zeitschrift*, 68(4):276–283, 2007.

[73] B. Schölkopf and A.J. Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. the MIT Press, 2002.

[74] A. Schreiber and R. Isermann. Integrated methodology for stationary and dynamic measurement and modeling of combustion engines. In *Challanges in the field of net Powertrain Concepts*, 2009.

[75] A. Schwarte, L. Hack, R. Isermann, H.G. Nitzke, J. Jeschke, and J. Piewek. Automatisierte Applikation von Motorsteuergeräten mit kontinuierlicher Motorvermessung. In *Proceedings of AUTOREG*, 2004.

[76] H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM New York, NY, USA, 1992.

[77] H.T. Siegelmann, B.G. Horne, and C.L. Giles. Computational capabilities of recurrent narx neural networks. Technical report, University of Maryland, College Park, MD, USA, 1995.

[78] J. Sjoberg, Q. Zhang, L. Ljung, A. Benveniste, B. Deylon, P.-Y. Glorennec, H. Hjalmarsson, and A. Juditsky. Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, 31:1691–1724, 1995.

[79] E.D. Sontag. Input to State Stability: Basic Concepts and Results. *Nonlinear and optimal control theory: lectures given at the CIME Summer School held in Cetraro, Italy, June 19-29, 2004*, 1932:163–220, 2008.

[80] E.D. Sontag and Y. Wang. On characterizations of the input-to-state stability property. *Systems Control Letters*, 24(5):351–359, 1995.

[81] E.D. Sontag and Y. Wang. New characterizations of input-to-state stability. *IEEE Transactions on Automatic Control*, 41(9):1283–1294, 1996.

[82] A. Sung. *Dynamische Vermessungsmethoden in der Online-Optimierung moderner Verbennungsmotoren.* PhD thesis, Fakultät für Informations- und Kognitionswissenschaften, TU-Tübingen, 2009.

[83] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15:116–132, 1985.

[84] A.H. Tan and K.R. Godfrey. The generation of binary and near-binary pseudorandom signals: an overview. *IEEE Transactions on Instrumentation and Measurement*, 51:583 – 588, 2002.

[85] S. Töpfer. Approximation nichtlinearer Prozesse mit Hinging Hyperplane Baummodellen (Approximation of Nonlinear Processes with Hinging Hyperplane Trees). *at-Automatisierungstechnik*, 50(4/2002):147, 2002.

[86] E. Walter and L. Pronzato. *Identification of Parametric Models from Experimental Data.* Springer-Verlag, London, 1997.

[87] R.A. Waltz, J.L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, 107(3):391–408, 2006.

[88] Y.J. Wang and C.T. Lin. Runge-Kutta neural network for identification of dynamical systems in high accuracy. *IEEE Transactions on Neural Networks*, 9(2):294–307, 1998.

[89] R.J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[90] T. Winsel. *Stabile neuronale Prozessmodelle: Automatisierte Generierung echtzeitfähiger Modelle zur Nachbildung des dynamischen Verhaltens von Verbrennungsmotoren.* PhD thesis, Institut für elektrische Energietechnik - Antriebstechnik (IEE-AT), 2002.

[91] M. Witczak. Toward the training of feed-forward neural networks with the D-optimum input sequence. *IEEE Transactions on neural networks*, 17(2):357–373, 2006.

[92] S. Zaglauer. Motoroptimierung durch Modellkomitees. Master's thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, 2010.

[93] S. Zaglauer and M. Deflorian. Bayesian d-optimal design. In *Proceedings of the 6th Conference Design of Experiments (DoE) in Engine Development*, 2011.

[94] J. Zehetner, J. Reger, and M. Horn. Echtzeit-Implementierung eines algebraischen Ableitungsschätzverfahrens (Realtime Implementation of an Algebraic Derivative Estimation Scheme). *at-Automatisierungstechnik*, 55(11):553–560, 2007.

[95] Z.H. Zhou, J. Wu, and W. Tang. Ensembling neural networks: Many could be better than all. *Artificial intelligence*, 137(1-2):239–263, 2002.

[96] R. Zimmerschied, M. Weber, and R. Isermann. Stationäre und dynamische Motorvermessung zur Auslegung von Steuerkennfeldern - Eine kurze Übersicht. *at - Automatisierungstechnik*, 53(02):87–94, February 2005.