

Bachelorarbeit

Maschinenbau

Identifikation mit vorstrukturierten rekurrenten Netzen

vorgelegt von

Zhiping Li
Matr.-Nr. 6835235

Betreuer:

Prof. Dr.-Ing. habil. Ansgar Trächtler
Dr.-Ing. Julia Timmermann

Paderborn, 10.03.2020

Bachelorarbeit

Identifikation mit vorstrukturierten rekurrenten Netzen

am: 10. März 2020

HEINZ NIXDORF INSTITUT
Universität Paderborn
Regelungstechnik und Mechatronik
Prof. Dr.-Ing. habil. Ansgar Trächtler
Fürstenallee 11
D-33102 Paderborn

Bachelorarbeit**Identifikation mit vorstrukturierten rekurrenten Netzen****Motivation**

Rekurrente neuronale Netze eignen sich dazu, das Ein-/Ausgangsverhalten nichtlinearer dynamischer Systeme zu approximieren, ohne dass die Möglichkeit besteht, aus dem Identifikationsergebnis auf interne Zustände, lineare Parameter oder auch nichtlineare Charakteristiken schließen zu können. Um sowohl die linearen Parameter als auch die vorhandenen Nichtlinearitäten zu identifizieren, wird in dieser Arbeit ein rekurrentes neuronales Netz verwendet, bei dem das Vorwissen über die Struktur des zu identifizierenden Systems mit berücksichtigt wird. Mit diesem vorstrukturierten rekurrenten Netz wird zudem die physikalische Interpretierbarkeit des Identifikationsergebnisses erreicht.

Aufgabenstellung

Die Identifikation nichtlinearer Systeme mit vorstrukturierten rekurrenten Netzen wird im Buch „Intelligente Verfahren“ von Schröder und Buss vorgestellt. Dieses Kapitel ist die Grundlage dieser Bachelorarbeit und soll von Herrn Li bearbeitet werden. Um dieses Thema bearbeiten zu können ist im Vorfeld die Einarbeitung in das Thema künstliche neuronale Netze notwendig. Es wird dann von der Idee ausgegangen, den bekannten Signalflussplan als ein strukturiertes rekurrentes Netz aufzufassen. Dieses rekurrente Netz wird in einer Beobachterstruktur implementiert. Durch eine geeignete Wahl der Beobachterkoeffizienten kann außerdem eine Filterung der Messdaten erfolgen.

Folgende Arbeitspakete sind zu bearbeiten

- Einarbeitung in das Thema künstliche neuronale Netze
- Bearbeitung des Kapitels über vorstrukturierte rekurrente Netze
- Implementierung der Beispiele aus dem Buchkapitel (Rekurrentes Netz zur Identifikation der Mechanik einer elektrischen Maschine als Beobachter und Identifikation des losebehafteten Zweimassensystems)
- Anwendung der Methodik auf ein mechanisches Anwendungsbeispiel

Über Vorgehen und Ergebnisse ist eine schriftliche Ausarbeitung anzufertigen.

Arbeitsaufwand: 360 h

Bearbeiter: Zhiping Li

Bearbeitungszeit: 20 Wochen

Betreuer: J. Timmermann

Beginn der Bearbeitung: 28.10.2019



Prof. Dr.-Ing. habil. Ansgar Trächtler

Erklärung:

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne unerlaubte fremde Hilfe angefertigt, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Paderborn, 10. März 2020

Zusammenfassung

Bei manchen nichtlinearen mechatronischen Antriebssystemen sind sowohl die linearen Systemparameter als auch die Nichtlinearitäten unbekannt. Einzig die Struktur des nichtlinearen Systems ist bekannt und die Inputs und Outputs des Systems sind messbar. Um dieses Problem zu lösen, beschäftigt sich die vorliegende Arbeit mit der Identifikation von linearen Systemparametern und der Nichtlinearität von nichtlinearen mechatronischen Antriebssystemen. Dazu werden die Strukturkenntnisse in ein auf General-Regression-Neural(GRN)-Netzwerke basierendes vorstrukturiertes rekurrentes neuronales Netz, auch als strukturiertes rekurrentes Netz genannt wird, übertragen und diese neue rekurrente Struktur wird in eine Beobachterstruktur eingebunden. In den Kapiteln 4 und 5 dieser Arbeit wird vorstrukturiertes rekurrentes neuronales Netz in zwei nichtlinearen Systeme mit unterschiedlichen Strukturen, d.h. leerlaufende elektrische Maschinen und Duffing-Oszillatoren, eingeführt und analysiert, wie die Struktur und die Parameter vom vorstrukturierten rekurrenten neuronalen Netz geregelt werden sollen, damit es sich besser an unterschiedliche nichtlineare Systeme anpassen kann. Durch die Analyse der Identifikationsergebnisse kann erkannt werden, dass die zu identifizierenden Systemparameter nicht nur in kurzer Zeit richtig approximiert werden können, sondern auch physikalisch interpretierbar sind. Auf dieser Grundlage ist es empfehlenswert, um die unbekannten Parameter komplexerer nichtlinearer Systeme zu identifizieren, das GRNN durch tieferes neuronales Netzwerk wie Multilayer Perceptron(MLP) ersetzt werden kann.

Summary

In some non-linear mechatronic systems, both the linear system parameters and the non-linearities are unknown. Only the structure of the non-linear system is known and the inputs and outputs of the system are measurable. To solve this problem, the present work deals with the identification of linear system parameters and the non-linearity of non-linear mechatronic systems. For this purpose, the structural knowledge is transferred to a pre-structured recurrent neural network based on general regression neural networks (GRNN) and this new recurrent structure is integrated into an observer structure. Chapters 4 and 5 of this work introduce pre-structured recurrent neural networks in two nonlinear systems with different structures and analyze how the structure and parameters of the pre-structured recurrent neural network are to be regulated so that it can adapt better to different nonlinear systems. By analyzing the identification results, it can be recognized that the system parameters to be identified can not only be correctly approximated in a short time, but can also be interpreted physically. On this basis, it is recommended to identify the unknown parameters of more complex nonlinear systems that GRNN can be replaced by a deeper neural network like multilayer perceptron (MLP).

Inhaltsverzeichnis

| | |
|---|-----------|
| 1 Einleitung | 1 |
| 1.1 Motivation | 1 |
| 1.2 Zielsetzung | 2 |
| 1.3 Vorgehensweise | 2 |
| 2 Grundlagen | 3 |
| 2.1 Künstliche Neuronale Netze | 3 |
| 2.2 Statische Neuronale Netze | 4 |
| 2.3 Radial Basis Function (RBF) Netz | 7 |
| 2.4 General-Regression-Neural-Netzwerk (GRNN) | 10 |
| 2.5 Kombination von zwei GRNNs | 15 |
| 2.6 Gradientenabstiegsverfahren | 17 |
| 2.7 Lerngesetz für das RBF-Netz und GRNN | 20 |
| 2.8 Lernfähiger Luenberger-Beobachter | 21 |
| 3 Erstellung strukturiertes rekurrentes Netzes | 23 |
| 3.1 Strukturierte rekurrente Netze | 23 |
| 3.2 Anwendung der Transformation | 24 |
| 3.3 Parameteradaption | 26 |
| 3.4 Zustandsdarstellung | 28 |
| 3.5 Partielle Ableitungen | 32 |
| 4 Identifikation einer elektrischen Maschine | 36 |
| 4.1 Anwendung der Beobachterstruktur | 36 |
| 4.2 Durchführung der Identifikation | 37 |
| 4.3 Simulative Parameteridentifikation | 39 |
| 4.4 Ergebnisse und Beurteilung | 45 |
| 4.5 Kurzzusammenfassung | 47 |
| 5 Identifikation eines Duffing-Oszillators | 50 |
| 5.1 Duffing-Oszillator | 50 |
| 5.2 Durchführung der Identifikation | 50 |
| 5.3 Modellbildung des strukturierten rekurrenten Netzes | 55 |
| 5.4 Bestimmung vom Parameter | 56 |

Inhaltsverzeichnis

| | |
|---|-----------|
| 5.5 Ergebnisse und Beurteilung | 61 |
| 5.6 Kurzzusammenfassung | 64 |
| 6 Zusammenfassung und Ausblick | 65 |
| 7 Literaturverzeichnis | 67 |

1 Einleitung

In diesem Kapitel werden die Motivation, die Zielsetzung sowie die Vorgehensweise dieser Arbeit beschrieben.

1.1 Motivation

In zahlreichen dynamischen mechatronischen Systemen sind Nichtlinearitäten wie Reibung und Lose unvermeidbar. Sie mathematisch festzustellen, ist normalerweise eine zu große Aufgabe und zudem zeit- und gelaufwändig. Es ist bekannt, dass neuronale Netze verwendet werden können, um die Gleichung von der Nichtlinearität zu approximieren. Deshalb tritt es eine Idee auf, dass durch Nutzen von neuronalen Netzen ein Modell vom vorhandenen nichtlinearen System gebildet wird, so dass das Modell wie normale neuronale Netze gelernt werden kann, und am Ende die gleichen Funktionen wie das tatsächliche System erreichen kann, d.h. unter derselbe Eingabe die gleiche Ausgabe zu erhalten. Weil der Lernprozess durch iterative Operationen des Computers ausgeführt wird, können die Ergebnisse schneller erhalten werden. Das mit den neuronalen Netzen aufgebaute Modell heißt in dieser Arbeit vorstrukturiertes rekurrentes Netz.

In Abbildung 1.1 wird ein konkretes Beispiel mit solch einem Problem dargestellt.

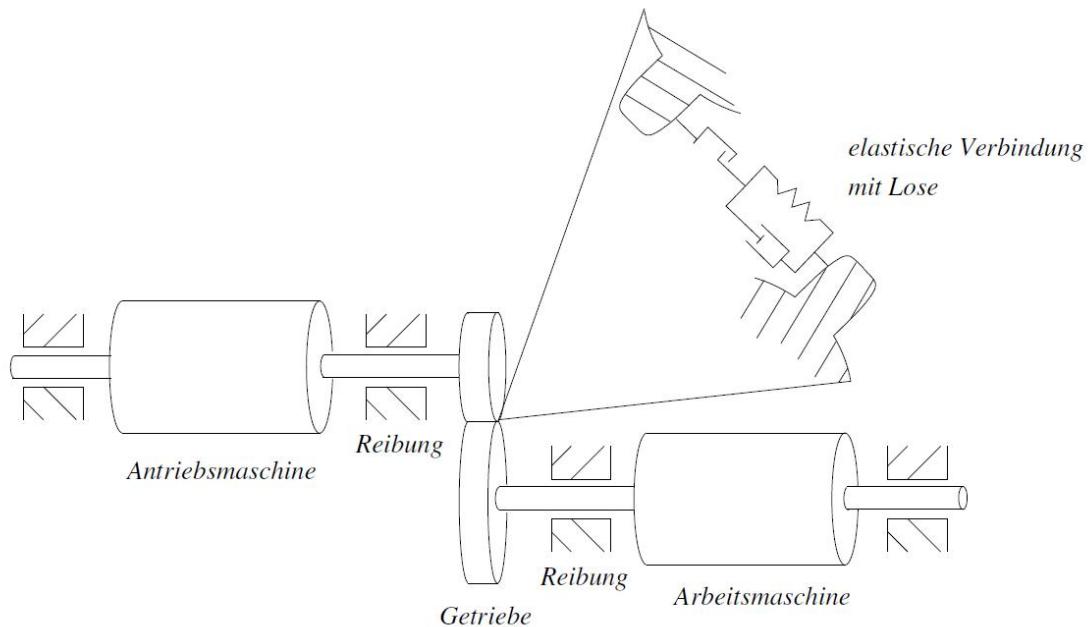


Abb. 1.1: Elastische Verbindung der Antriebs- und Arbeitsmaschine mit Lose [C03]

Die Nichtlinearitäten in diesem Beispiel, d.h. die Lagerreibung und die in Getrieben oder Kupplungen vorhandene Lose, wirken sich negativ auf die Güte der Regelung und damit auf die Qualität der Anwendung (z. B. der Positionierung eines Roboterarmes).

Besonders wenn die Lose sich nicht im Eingriff, d.h. kein Kontakt zwischen zwei benachbarte Zähne bei unterschiedlichen Zahnrädern, befindet, sind Antriebs- und Arbeitsmaschine entkoppelt. Um den negativen Einfluss der Nichtlinearitäten zu verringern, können Lager mit geringerer Reibung oder spielfreie Getriebe eingesetzt werden. Diese konstruktiven Maßnahmen haben den Nachteil, aufwendig und damit kostenintensiv zu sein. Im Vergleich dazu ist es eine bessere Lösung, die direkte mathematische Ableitung von Nichtlinearität zu umgehen und durch regelungstechnische Maßnahmen einen Funktionsapproximator, d.h. in dieser Arbeit vorgestelltes strukturiertes rekurrentes Netz, herzuleiten.

1.2 Zielsetzung

Ziel dieser Arbeit ist es, die linearen Parameter und die vorhandenen Nichtlinearitäten eines Single-Input-Single-Output (SISO)-Systems gleichzeitig durch das strukturierte rekurrente Netz zu identifizieren und dabei eine physikalische Interpretierbarkeit der Identifikationsergebnisse zu erhalten. Z.B. die Differentialgleichung von der leerlaufenden elektrischen Maschine im Beispiel vom Abschnitt 3.2 ist wie Gleichung 1-1 zu sehen. Das Reibmoment $M_w(\Omega)$ ist hierbei eine Nichtlinearität von Winkelgeschwindigkeit Ω und ihre Gleichung ist unbekannt. Der Wert vom Parameter $\frac{1}{J}$ ist auch unbekannt.

$$\dot{\Omega} = \frac{1}{J} \cdot (M_L - M_w(\Omega)) \# (1 - 1)$$

Das Ziel dieser Arbeit ist dann, durch strukturiertes rekurrentes Netz und vorgegebene Ein- und Ausgangswerte dieses Systems der Parameter $\frac{1}{J}$ und die Nichtlinearität $M_w(\Omega)$ zu identifizieren und mit den identifizierten Ergebnissen das strukturierte rekurrente Netz gleiche Funktion wie die Gleichung 1-1 erreichen kann, d.h. beim Eingeben von gleicher externer Anregung M_L gleiche Antwort $\dot{\Omega}$ und Ω zu erhalten.

1.3 Vorgehensweise

In Kap. 2 dieser Arbeit werden die für die Erstellung der Modelle notwendigen Grundlagen und Vorwissen, wie Neuronale Netze und Luenberger-Beobachter, beschrieben.

In Kap. 3 wird alle notwendigen und allgemeinen Rechnungen zur Modellbildung des strukturierten rekurrenten Netz abgeleitet.

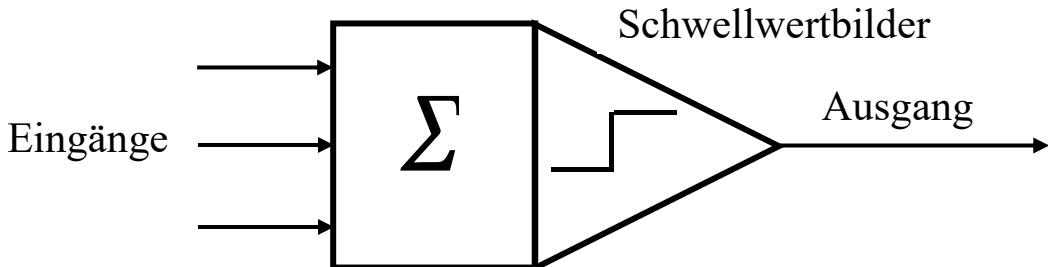
Zwei konkrete Aufgabe werden durch die Kombination von vorhandenen Begriffen und Rechnungen separat in Kap. 4 und Kap. 5 unter der Umgebung Matlab/Simulink modelliert. Am Ende jedes Kapitels, d.h. im Abschnitt 4.4 und 5.5 werden die Ergebnisse analysiert.

2 Grundlagen

Dieses Kapitel beginnt mit einer Einführung über neuronale Netze und beschreibt zwei Arten, die in dieser Arbeit diskutiert werden: Radial-Basis-Function(RBF)-Netze und General-Regression-Neural-Netzwerke (GRNN). Im Anschluss wird das bedeutsame Gradientenabstiegsverfahren dargestellt. Dieses wird in Abschnitt 2.6 sowohl im RBF-Netz als auch im GRNN eingesetzt und dadurch wird ein vollständiges neuronales Netz in einem Matlab-Skript programmiert. Danach wird eine spezielle Struktur für die Konvergenz des Systems eingeführt: Lernfähiger Luenberger-Beobachter. Die Inhalte dieses Kapitels sind von [S10] entnommen.

2.1 Künstliche Neuronale Netze

Künstliche neuronale Netze basieren meist auf der Vernetzung einer großen Zahl an McCulloch-Pitts-Neuronen oder leichten Abwandlungen davon. Auf diese Weise wurde 1943 von W. McCulloch und W. Pitts das wohl erste statische neuronale Netz entwickelt [MCP88]. Bei diesem Neuron werden die Eingangssignale addiert und ein Schwellwert-Gatter erzeugt anschließend ein binäres Ausgangssignal. Eine



schematische Darstellung ist in Abbildung 2.1 zu sehen.

Abb. 2.1: Schematische Darstellung eines McCulloch-Pitts-Neuron [S10]

Die erste Blütezeit der neuronalen Netze verlief von 1955 bis 1969. Sie begann mit dem ersten erfolgreichen Neurocomputer (Mark I Perceptron), der in den Jahren 1957 und 1958 am Massachusetts Institute of Technology (MIT) entwickelt worden war [ROS88], und endete 1969 mit M. Minskys und S. Paperts Buch Perceptrons [MIP88]. In den stillen Jahren (1969–1982) fand das Gebiet der neuronalen Netze kaum Aufmerksamkeit, allerdings wurden in dieser Zeit zahlreiche Grundlagen für die neueren Entwicklungen gelegt. So entwickelte P. Werbos 1974 das Backpropagation-Verfahren [W74]. Mit der Wiederentdeckung des Backpropagation-Algorithmus durch D. Rumelhart, G. Hinton und R. J. Williams im Jahr 1986 begann die Renaissance neuronaler Netze [RHW88], die bis heute andauert.

Seit Beginn der neunziger Jahre des 20. Jahrhunderts haben sich mehrere statische neuronale Netze durchgesetzt. Es gibt solche, die ihre Approximationsfähigkeit durch Überlagerung gewichteter lokaler Basisfunktionen erlangen, wie das RBF-Netz und die beiden daraus abgeleiteten Funktionsapproximatoren GRNN und Harmonic-Activated-Neural-Network (HANN). Beim Local-Linear-Model-Tree (LOLIMOT) können mehrere unterschiedliche Basisfunktionen vorhanden sein, die jeweils über Zugehörigkeitsfunktionen bestimmten Bereichen zugeordnet sind. Die größte Bedeutung haben allerdings die Multi-Layer-Perzeptronen(MLP)-Netzwerke erlangt. Eine Übersicht über die oben genannten Arten von neuronalen Netzen zeigt Abbildung 2.2. Darin sind die gelb markierten Quadrate (RBF-Netze und GRNN) die Teile, die in dieser Arbeit besprochen und verwendet werden.

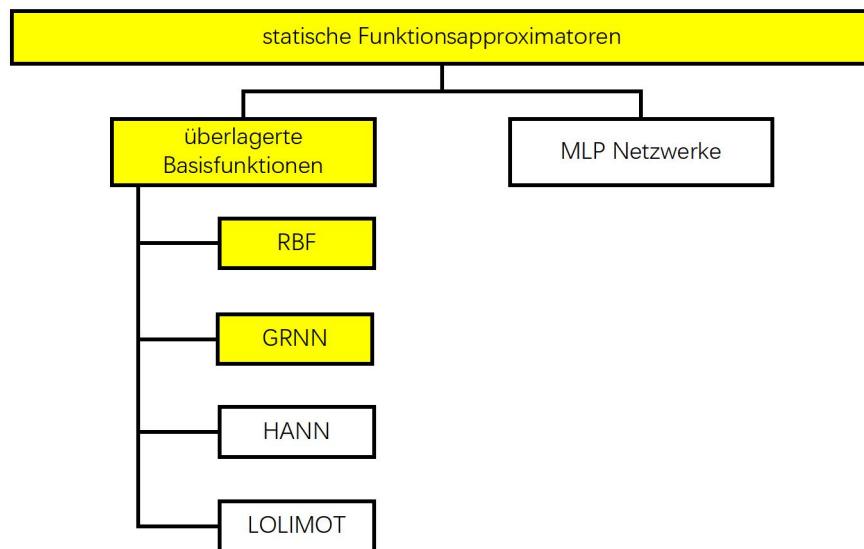


Abb. 2.2: Übersicht über wichtige statische Funktionsapproximatoren

2.2 Statische Neuronale Netze

In diesem Abschnitt werden einige in dieser Arbeit häufig verwendete Begriffe definiert. Zudem findet sich die Darstellung einer normalen Struktur eines neuronalen Netzes in Abbildung 2.3.

In dieser Arbeit wird der nichtlineare Teil des mechanischen Systems durch die Struktur des neuronalen Netzwerks ersetzt und simuliert, deshalb haben die *n Eingänge* ($x_1, x_2 \dots x_n$) und *Ausgang* O_j in der Abbildung dieselbe physikalische Bedeutung wie die Ein- und Ausgänge des realen nichtlinearen Systems. Hier wird nur die einfachste Struktur des neuronalen Netzes beschrieben, d. h., es gibt nur einen Ausganswert und alle Eingaben sind in einer Reihe. Im Gegensatz dazu gibt es im MLP-Netzwerk mehrere Reihen an Eingaben und Gewichten.

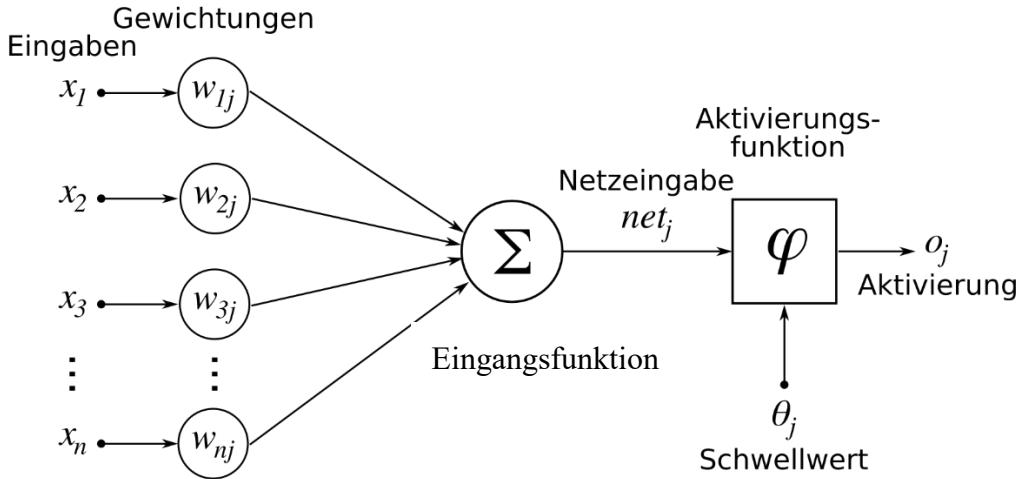


Abb. 2.3: Schema eines künstlichen Neurons: Künstliches Neuron mit Index j [C06]

- Jede *Eingabe* hat ein unterschiedliches Einflussniveau auf den Ausgang des Systems. Dieser unterschiedliche Einfluss wird durch die Multiplikation jedes *Eingangs* mit einer unterschiedlichen *Gewichtung* erzielt. Letztere sind im neuronalen Netz von entscheidender Bedeutung. Die Auswahl geeigneter Gewichtungen beeinflusst die Qualität des gesamten Netzes.
- Danach soll jeder *Eingang* des Systems durch eine *Eingangsfunktion* miteinander verbunden werden. Im Gegensatz zu realen nichtlinearen Systemen wird hier nur eine Operation ausgeführt, nämlich die Addition.
- Die resultierende Summe wird durch eine *Aktivierungsfunktion* mit Schwellwert überprüft, und der überprüfte Wert ist der *Ausgang* dieses neuronalen Netzwerks sowie der *identifizierte Ausgabewert* des entsprechenden nichtlinearen Systems. Ein Beispiel ist als Aktivierungsfunktion die Treppenfunktion. Sie wird hauptsächlich im logischen neuronalen Netz genutzt. Wenn die Summe die Überprüfung besteht, entspricht der Ausgangswert 1, umgekehrt, d.h. die Summe kleiner als der Aktivierungswert ist, wird der Ausgangswert als 0 bezeichnet. Diese zwei Werte, 0 und 1, entsprechen der falschen und der richtigen Logik.
- Für der *Aktivierungsfunktion* steht eine große Zahl unterschiedlicher Funktionen zur Auswahl, von denen die Abbildung 2.4 nur einige zeigt.

Durch Kombination unterschiedlicher Gewichtungen und Aktivierungsfunktionen können unterschiedliche nichtlineare Systeme simuliert werden. Dieser Ausdruck, bei dem eine nichtlineare Funktion durch die Summe mehrerer Terme ersetzt ist, ähnelt der Taylor-Reihe und der Fourier-Reihe. Die beiden in dieser Arbeit verwendeten neuronalen Netze, nämlich das RBF-Netz und das GRNN, werden in den folgenden Abschnitten erörtert.

| | $T(\hat{u}_\Sigma)$ | $T(\hat{u}_\Sigma)$ |
|---------------------------|---|---------------------|
| | $\tanh(\hat{u}_\Sigma) = \frac{1 - e^{-2\hat{u}_\Sigma}}{1 + e^{2\hat{u}_\Sigma}}$ <p>tanh Funktion</p> | |
| weiche Transferfunktionen | $\frac{O - U}{1 + e^{-c\hat{u}_\Sigma}} + U$ <p>verallg. Logistikfunktion</p> | |
| | $c \cdot \hat{u}_\Sigma$ <p>lineare Funktion</p> | |
| harte Transferfunktionen | $\text{sign}(\hat{u}_\Sigma)$ <p>Signumfunktion</p> | |
| | $= \begin{cases} +1, & \hat{u}_\Sigma \geq 0 \\ 0, & \text{sonst} \end{cases}$ <p>Stufenfunktion</p> | |

Abb. 2.4: Typische Transferfunktionen [C03]

2.3 Radial Basis Function (RBF) Netz

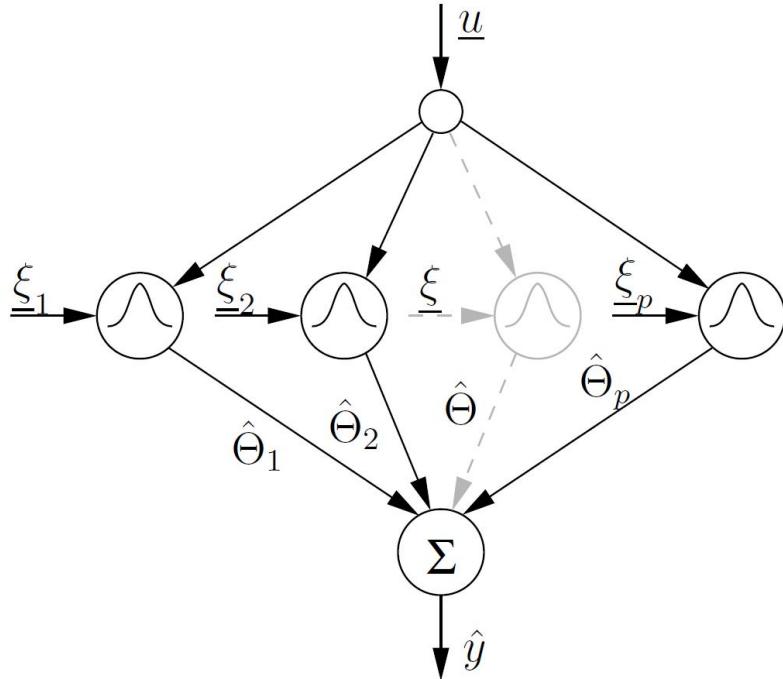


Abb. 2.5: Struktur des RBF-Netzes mit p Stützstellen [S10]

Die mathematische Beschreibung eines RBF-Netzes mit r Neuronen, den Gewichten $\hat{\theta}_j$ und den Stützstellen ξ_j mit $1 \leq j \leq r$ lautet

$$\hat{y}_{RBF} = \sum_{j=1}^r \hat{\theta}_j \cdot e^{-\frac{(u-\xi_j)^2}{2\sigma^2}} \cdot \#(2-1)$$

Hierbei werden die Gaußschen Glockenkurven $A_j(u)$ als Basis- bzw. Aktivierungsfunktionen. Die Darstellung der Glockenkurven ist an die der Standardverteilung mit σ^2 angeglichen. Die Parameter der Aktivierungsfunktionen sind der Glättungsfaktor σ , der den Grad der Überlappung zwischen benachbarten Aktivierungen bestimmt, und die Stützstellen ξ_j , die die Lage der Zentren der Gaußglocken im Eingangsraum beschreiben [C03].

Bei den in dieser Arbeit eingesetzten Approximatoren sind die Stützstellen im Eingangsbereich äquidistant verteilt, was unterschiedlich und einfacher wie in allgemeinen Situationen ist. Dadurch ist der Abstand zwischen zwei Stützstellen $\Delta\xi$ konstant

$$\Delta\xi = \frac{u_{max} - u_{min}}{r-1} \cdot \#(2-2)$$

Ausgehend von dem Maximalwert u_{max} und Minimalwert u_{min} der Eingabewerte und der Anzahl r der Stützstellen kann auch deswegen die Größe jeder Stützstelle ermittelt werden

$$\xi_j = u_{min} + (j - 1) \cdot \frac{u_{max} - u_{min}}{r - 1} \cdot \#(2 - 3)$$

Damit muss für die Anzahl r der Stützstellen $r \geq 2$ gelten.

Zur besseren Vergleichbarkeit wird der Glättungsfaktor σ auf den Abstand zwischen zwei Stützstellen normiert. Damit gilt $\sigma_{norm} = \frac{\sigma}{\Delta\xi}$.

Mit dem normierten Glättungsfaktor ergibt sich die mathematische Beschreibung des RBF-Netzes zu

$$\hat{y}_{RBF} = \sum_{j=1}^r \hat{\theta}_j \cdot A_j(u) \quad mit \quad A_j(u) = e^{-\frac{(u-\xi_j)^2}{2 \cdot \sigma_{norm}^2 \cdot 4 \xi_j^2}} \#(2 - 4)$$

Diese Formel kann auch in Matrixform ausgedrückt werden. Mit Gewichts- oder Parametervektor $\hat{\theta}$

$$\hat{\theta} = [\hat{\theta}_1 \ \dots \ \hat{\theta}_r]^T$$

und der Aktivierungsfunktionen $A_j(u)$

$$\underline{A}(u) = [A_1(u) \ \dots \ A_r(u)]^T$$

ergibt sich die mathematische Beschreibung des RBF-Netzes in vektorieller Form zu

$$\hat{y}_{RBF} = \hat{\theta}^T \cdot \underline{A}(u) \#(2 - 5)$$

Der obige Inhalt wird durch die folgende Abbildung intuitiver beschrieben.

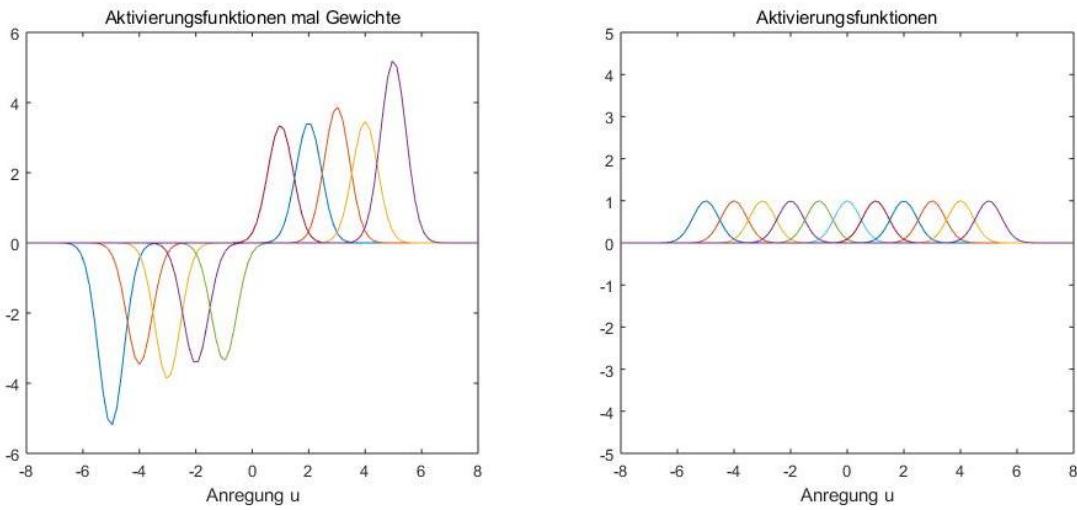


Abb. 2.6: reine Aktivierungsfunktionen und Aktivierungsfunktionen mal Gewichte im RBF-Netz mit $u_{max} = 5$, $u_{min} = -5$, $r = 11$ und $\sigma_{norm} = 0.45$

Wie in der Abbildung 2.6 dargestellt, stellt die Anregung u auf der Abszisse den Eingabewert des Systems dar. Elf Stützstellen sind gleichmäßig von u_{min} nach u_{max} verteilt. Auf jeder Stützstelle ist eine entsprechende Aktivierungsfunktion zentriert,

deshalb gibt es derer insgesamt elf. Das Produkt der Aktivierungsfunktion und das entsprechende Gewicht zeigen an, wie diese unterschiedlichen Eingaben beeinflusst. Für einen gewählten Eingangswert gibt es auf den elf mit Gewicht multiplizierten Aktivierungsfunktionen insgesamt elf Ausgangswerte. Deren Summe ist der identifizierte Ausgangswerte des RBF-Netzes entsprechend dem gewählten Eingangswert.

Die folgenden Probleme bestehen bei der tatsächlichen Verwendung des neuronalen RBF-Netzes: Einerseits kann die Interpolation zwischen den Stützwerten nicht genau den realen Werte folgen, andererseits ist das Extrapolationsverhalten aufgrund der fehlenden Monotonie-Erhaltung ebenfalls ungünstig. Wenn sich der Eingabewert vom angegebenen Bereich entfernt, tendieren alle Aktivierungsfunktionen zu 0, sodass die Gesamtausgabe des neuronalen Netzwerks ebenfalls zu 0 tendiert, anstatt auf einem festen Wert zu bleiben. Um diese Probleme zu optimieren, wird ein weiteres neuronales Netz eingeführt, nämlich GRNN.

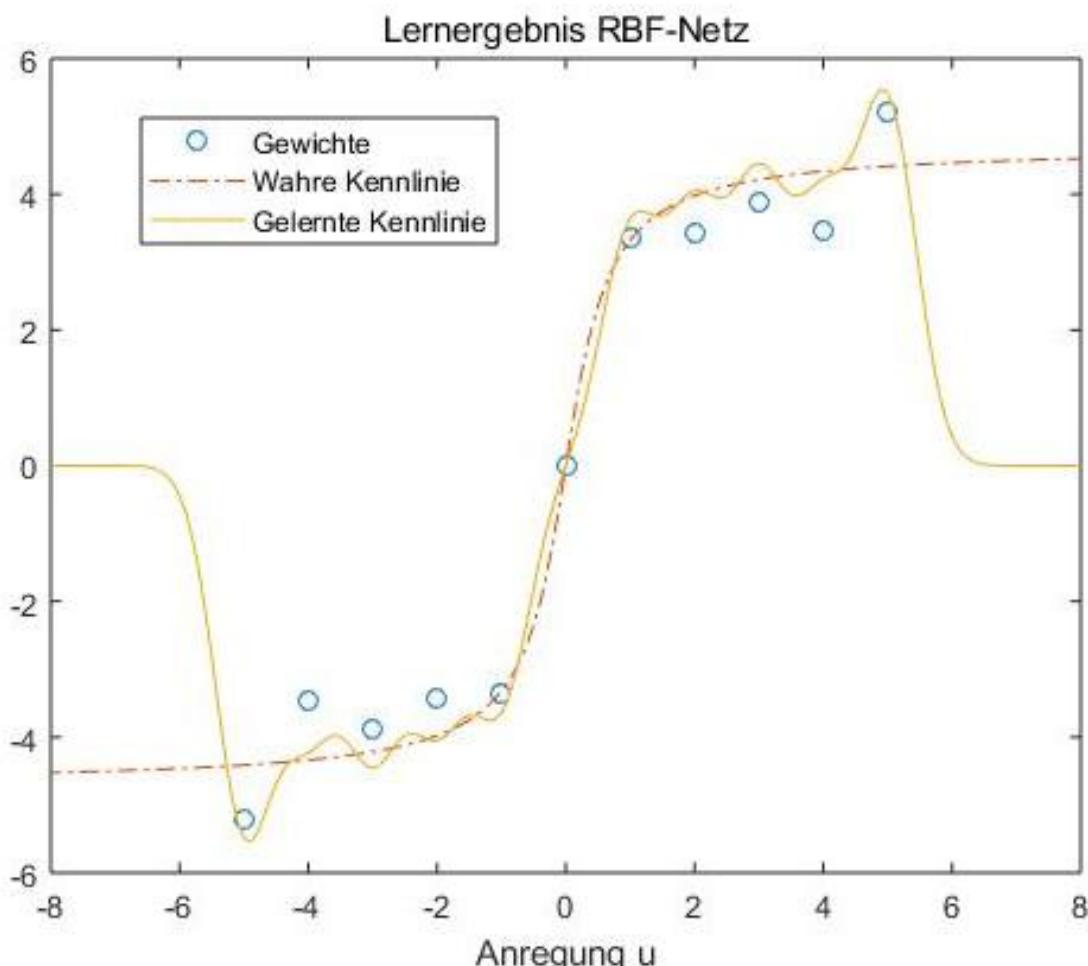


Abb. 2.9.1: Lernergebnisse von dem RBF-Netz mit ursprünglicher Gleichung
 $NL(u) = 3 \cdot \arctan(2 \cdot u)$ mit $r=11$ und $\sigma_{norm}=0.45$

2.4 General-Regression-Neural-Netzwerk (GRNN)

Das GRNN stellt eine Weiterentwicklung des RBF-Netzes dar. Der wesentliche Unterschied besteht in einer Normierung aller Aktivierungsfunktionen auf deren Summe.

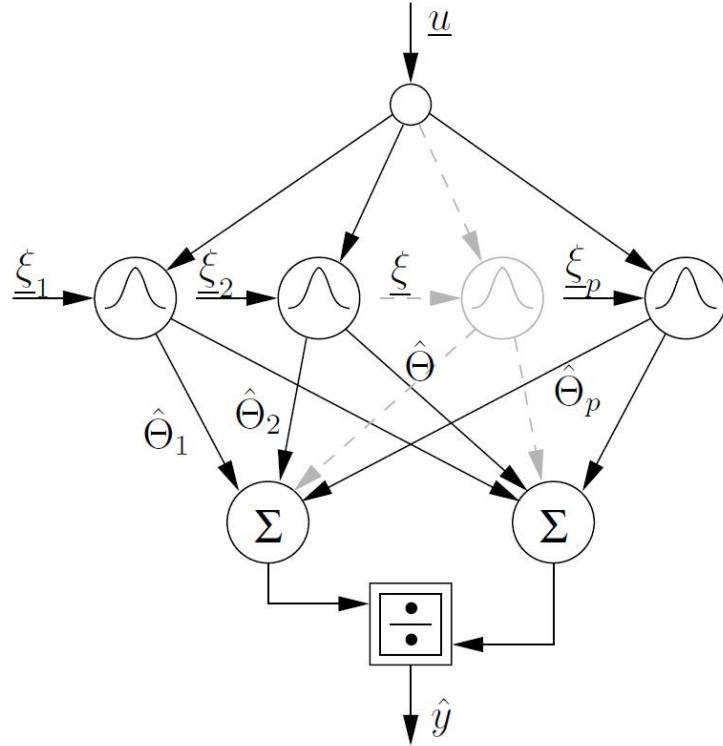


Abb. 2.7: Struktur des GRNN mit p Stützstellen [S10]

Die mathematische Beschreibung des GRNN lautet:

$$\hat{y}_{GRNN} = \sum_{j=1}^r \hat{\theta}_j \cdot A_j(u) \quad \text{mit} \quad A_j(u) = \frac{e^{-\frac{(u-\xi_j)^2}{2 \cdot \sigma_{norm}^2 \cdot \Delta \xi^2}}}{\sum_{k=1}^r e^{-\frac{(u-\xi_k)^2}{2 \cdot \sigma_{norm}^2 \cdot \Delta \xi^2}}} \#(2 - 6)$$

In vektorieller Schreibweise kann die oben genannte Gleichung geschrieben werden als

$$\hat{y}_{GRNN} = \hat{\underline{\theta}}^T \cdot \underline{A}(u) \#(2 - 7)$$

mit

$$\hat{\underline{\theta}} = [\hat{\theta}_1 \ \dots \ \hat{\theta}_r]^T \quad \text{und} \quad \underline{A}(u) = [A_1(u) \ \dots \ A_r(u)]^T$$

Die physikalische Bedeutung der in der Formel enthaltenen speziellen Parameter, z. B. normierter Glättungsfaktor σ_{norm} , Stützstellen ξ_j , Abstand zwischen zwei

Stützstellen $\Delta\xi$ sowie Eingangswerte u , entspricht genau der Formel im neuronalen RBF-Netz.

Die folgende Abbildung 2.8 zeigt die Verteilung aller Aktivierungsfunktionen des neuronalen GRNN-Netzes unter gegebenen Bedingungen:

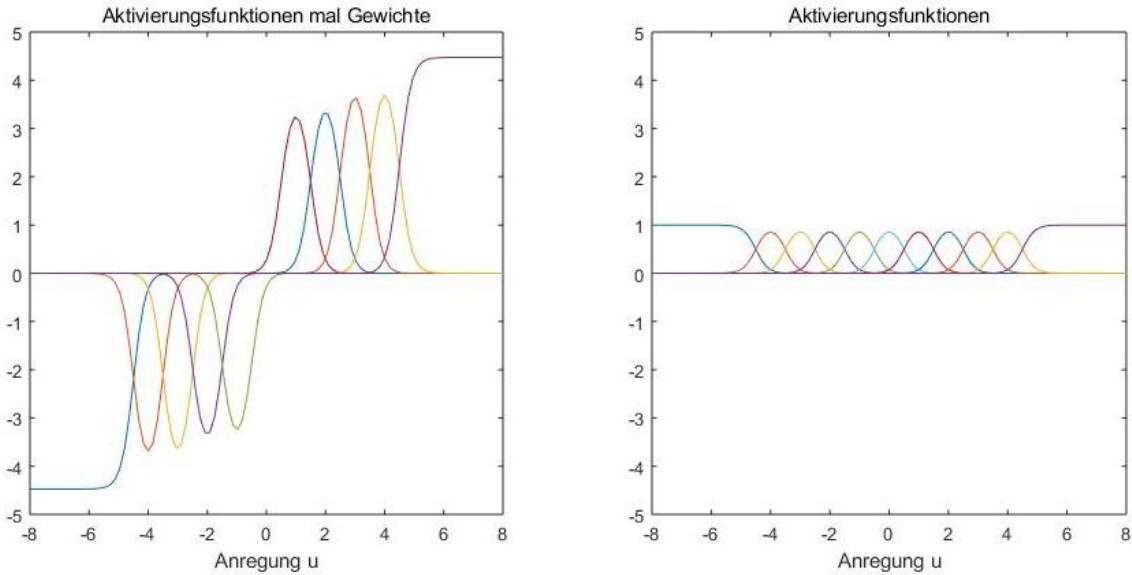


Abb. 2.8: reine Aktivierungsfunktionen und Aktivierungsfunktionen mal Gewichte im GRNN mit $u_{max} = 5$, $u_{min} = -5$, $r = 11$ und $\sigma_{norm} = 0.45$

Wie aus den Abbildungen 2.6 und 2.8 ersichtlich ist, verhält sich die Aktivierungsfunktionen von RBF-Netzen und GRNN im gegebenen Eingangsbereich ähnlich. Nur der maximale Wert von jeder Aktivierungsfunktion im GRNN kleiner oder gleich 1 sein kann. Es ist abhängig von den Größen von dem Glättungsfaktor σ_{norm} und der Anzahl der Aktivierungsfunktion r . Bei der Extrapolation kann der Wert der Aktivierungsfunktion auf 1 beim GRNN beibehalten während beim RBF-Netz der Wert der Aktivierungsfunktion automatisch auf 0 abnehmen. Bei der Reproduktion der ursprünglichen Funktion ist die Qualität des neuronalen RBF-Netzes und des GRNN jedoch sehr unterschiedlich.

Im Vergleich zu den Verhalten beider neuronalen Netze in der Abbildung 2.9.1 und 2.9.2 verhält sich die Lernkurve von GRNN innerhalb eines vorgegebenen Bereiches von Eingabewerten $-5 \leq u \leq 5$, der Interpolation, zwischen benachbarten Stützstellen glatter und schwingungsfreier. Der Grund dafür ist, dass der Wert einer RBF-Kennlinie immer die Tendenz hat, zu 0 an- oder abzunehmen, wenn es sich von letztem Gewicht entfernt. Nur wenn es nah genug liegt, kann er sich unter dem Einfluss der nächsten Aktivierungsfunktion wieder der Kennlinie nähern. Außerhalb des vorgegebenen Bereiches von Eingabewerten $5 \leq u$ oder $u \leq -5$, der Extrapolation, kann sich die Lernkurve von GRNN bei einer weiteren Zu- oder Abnahme des Eingabewerts auf einem festen Wert halten, d. h. die Gewichte an den beiden Grenzen des

Eingangsbereiches, während die Lernkurve des neuronalen RBF-Netzes zu 0 tendiert. Aus diesem Grund ist das GRNN für Regelungstechnische Anwendungen entscheidend.

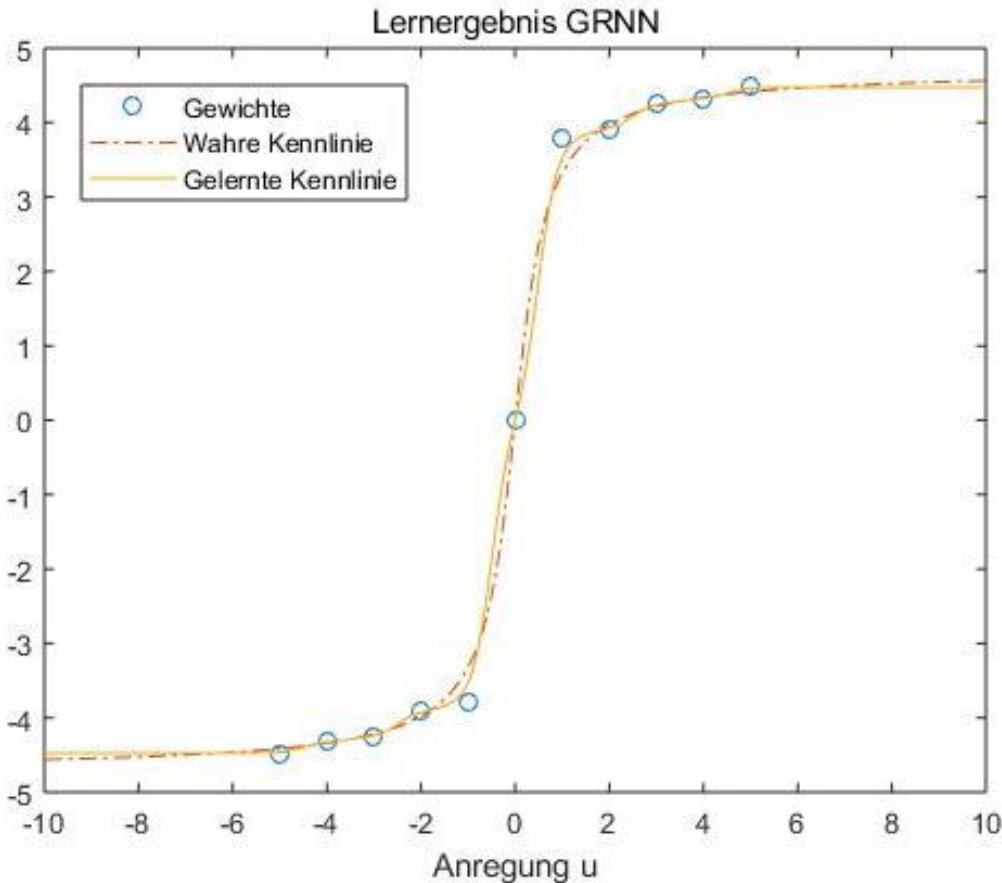


Abb. 2.9.2: Lernergebnisse von dem GRNN mit ursprünglicher Gleichung
 $NL(u) = 3 \cdot \arctan(2 \cdot u)$ mit $r = 11$ und $\sigma_{norm} = 0.45$

Des Weiteren ist die Auswahl der Werte wie die Anzahl der Aktivierungsfunktionen r und der normierte Glättungsfaktor σ_{norm} für die Qualität des Approximationsergebnisses von großer Bedeutung. Dies wird nachfolgend durch zwei Beispiele erklärt.

Zuerst wird der normierte Glättungsfaktor σ_{norm} auf 1 geändert. Die Ergebnisse sind in den Abbildungen 2.10 und 2.11 dargestellt. Die Aktivierungsfunktionen sind zu breit, aber nicht hoch genug. Jedes Gewicht dient nicht speziell für einen bestimmten, sondern nahezu allgemein für den ganzen Eingangsbereich. Aufgrund dieser hohen Korrelation zwischen angrenzenden Gewichten kostet der Lernprozess mehr Zeit für die Approximation einer Kennlinie mit Steigungen. Ein anderes Resultat ist (siehe Abbildung 2.10), dass eine große Zahl von Gewichten weit entfernt von der Kennlinie liegt, selbst wenn das Aussehen der wahren Kennlinie noch gut ist. Bei der weiteren Annahme des normierten Glättungsfaktor σ_{norm} werden die Figuren aller Aktivierungsfunktionen zu einer fast geraden Linie. In diesem Fall sind sie nicht in der Lage, eine Kennlinie zu approximieren.

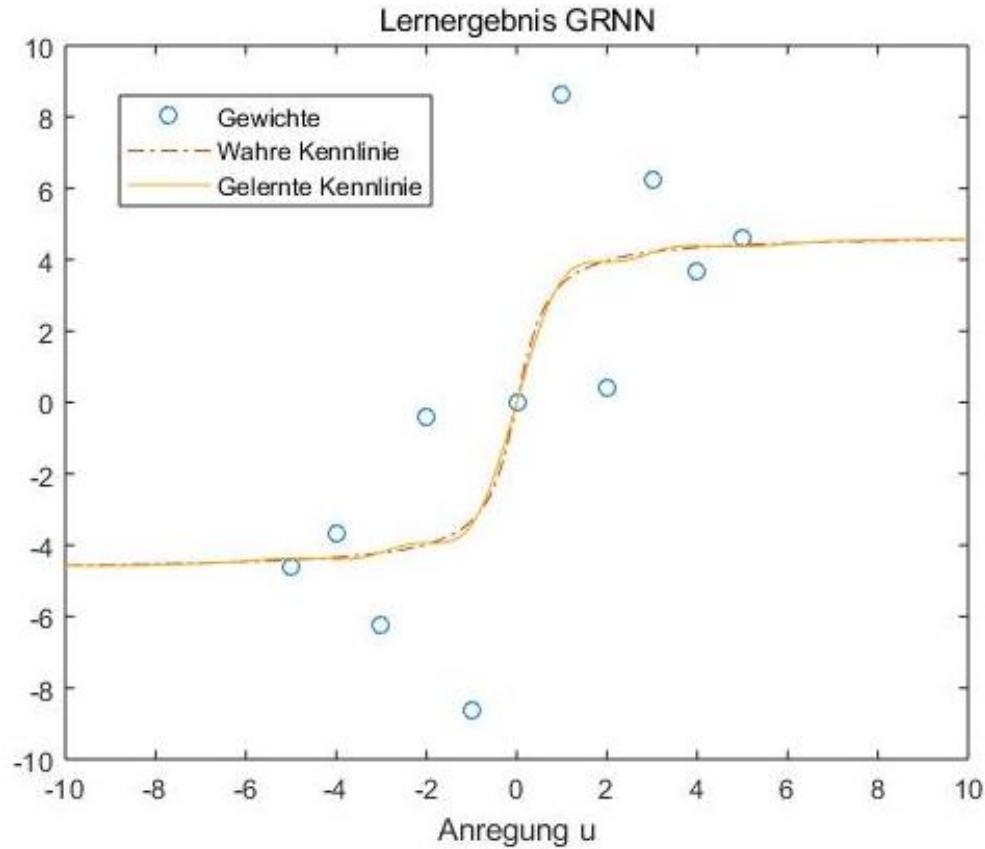


Abb. 2.10: Lernergebnisse GRNN mit ursprünglicher Gleichung mit $r = 11$ und $\sigma_{\text{norm}} = 1$

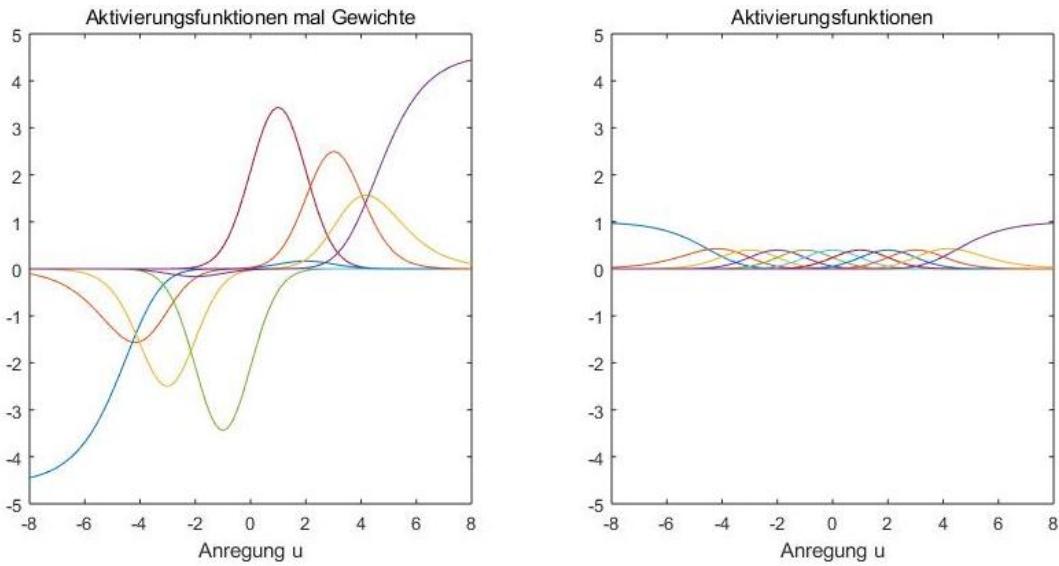


Abb. 2.11: reine Aktivierungsfunktionen und Aktivierungsfunktionen mal Gewichte im GRNN mit $r = 11$ und $\sigma_{\text{norm}} = 1$

Dann wird die Anzahl der Aktivierungsfunktionen r auf 5 geändert. Im Gegensatz zum letzten Beispiel sind fünf Aktivierungsfunktionen nicht ausreichend, um die vorgegebene Kennlinie zu approximieren. Wie in der Abbildung 2.12 zu sehen ist,

entfernt sich die gelernte Kennlinie von der wahren Kennlinie. Die identifizierte Kennlinie hierbei kann nur die Hauptmerkmale der wahren Kennlinie nachbilden, wie die Grenzwerte an beiden Seiten und den zentralen Wert.

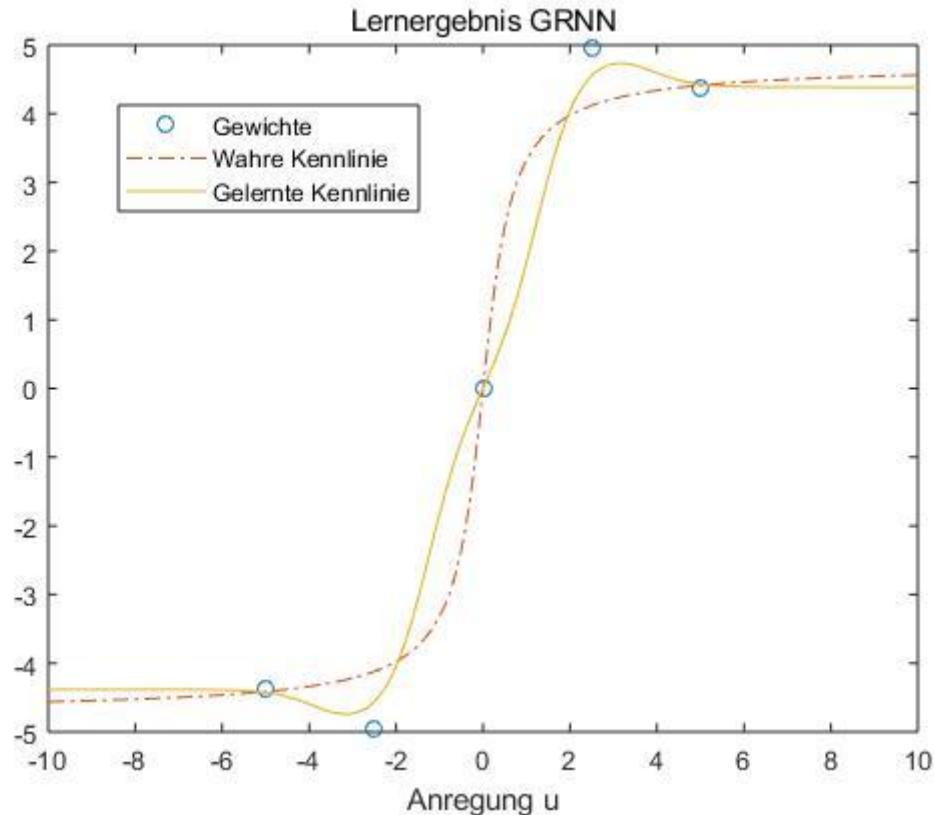


Abb. 2.12: Lernergebnisse GRNN mit ursprünglicher Gleichung mit $r = 5$
und $\sigma_{norm} = 0.45$

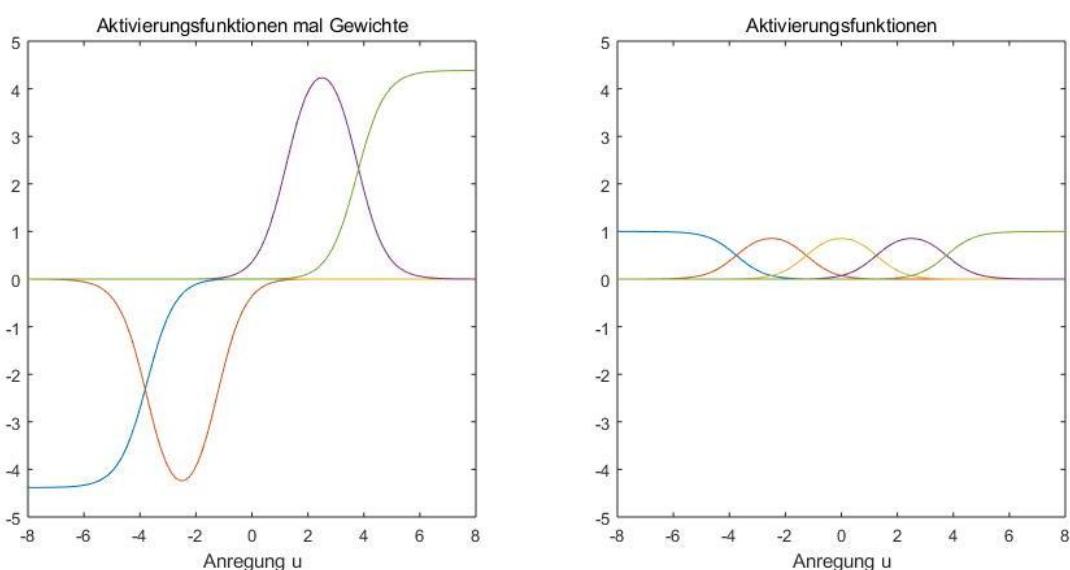


Abb. 2.13: reine Aktivierungsfunktionen und Aktivierungsfunktionen mal Gewichte im
GRNN mit $r = 5$ und $\sigma_{norm} = 0.45$

2.5 Kombination von zwei GRNNs

Darüber hinaus funktioniert ein einzelnes RBF-Netz oder ein GRNN für die Approximation der Unstetigkeit nicht gut. In der Abbildung 2.9 ist schon ersichtlich, dass um den Bereich ($u = 0$) die gelernte Kennlinie die wahre Kennlinie nicht perfekt approximieren kann. Zur Kompensation des Fehlers kommt es bei den letzten und nächsten Gewichten noch zu einem kleinen Überschreiten. Um dieses Problem zu lösen, können zwei GRNNs, eines für einen positiven Eingangsbereich und einen für negativen Eingangsbereich, für die Approximation einer Kennlinie eingesetzt werden [S10]. Hier wird durch ein Beispiel durch Nutzen vom GRNN mit gleichem Parameter wie in der Abbildung 2.9.2 aber unterschiedlicher Funktion, es zu darstellen.

Um eine Unstetigkeit in der originalen Funktion zu realisieren, wird diese Funktion durch zwei Funktionen darstellt, eine das Verhalten der Funktion im positiven Eingangsbereich und die andere das Verhalten im negativen Eingangsbereich zu beschreiben, nämlich

$$y = 0.29 \cdot \arctan(0.3 \cdot u) + 0.4 \quad (\text{für } u \geq 0)$$

und

$$y = 0.35 \cdot \arctan(0.3 \cdot u) - 0.15 \quad (\text{für } u \leq 0)$$

Durch das Nutzen vom vorhandenen Skript sind die Approximationsergebnisse wie Abb. 2.14 zu sehen.

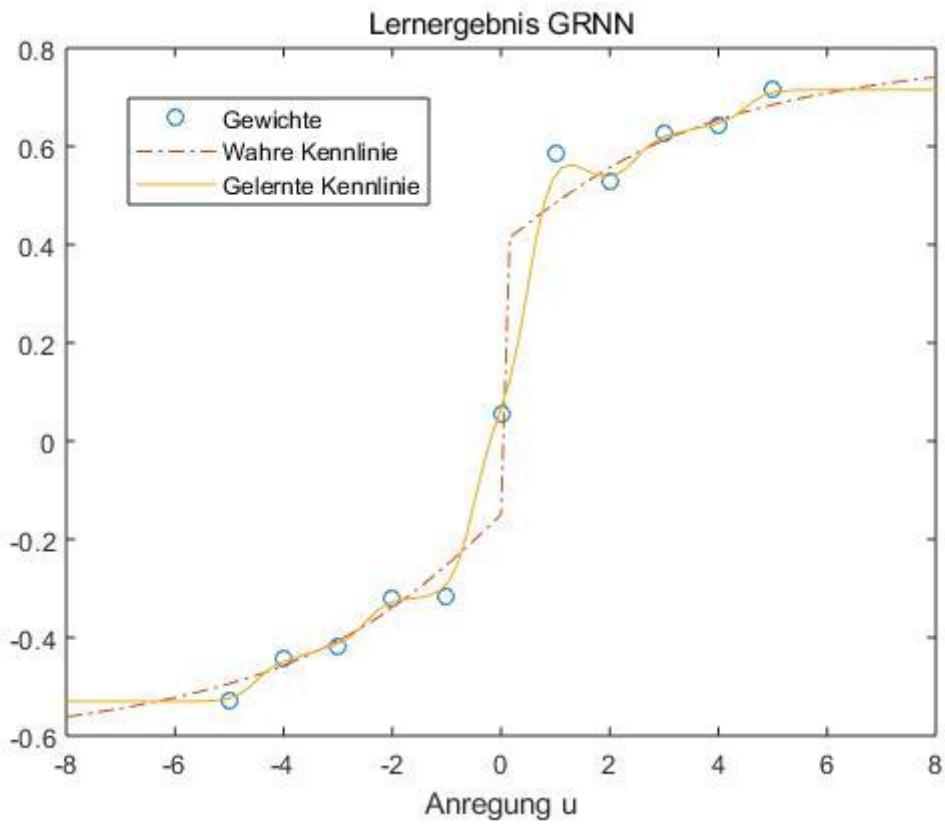


Abb. 2.14: Lernergebnisse vom vorhandenem GRNN mit der Unstetigkeit

Nach der Optimierung des GRNNs mit der Idee am Anfang dieses Abschnittes, nämlich durch zwei GRNNs die Funktion mit der Unstetigkeit zu approximieren, werden die Ergebnisse wie Abb. 2.15 zu sehen.

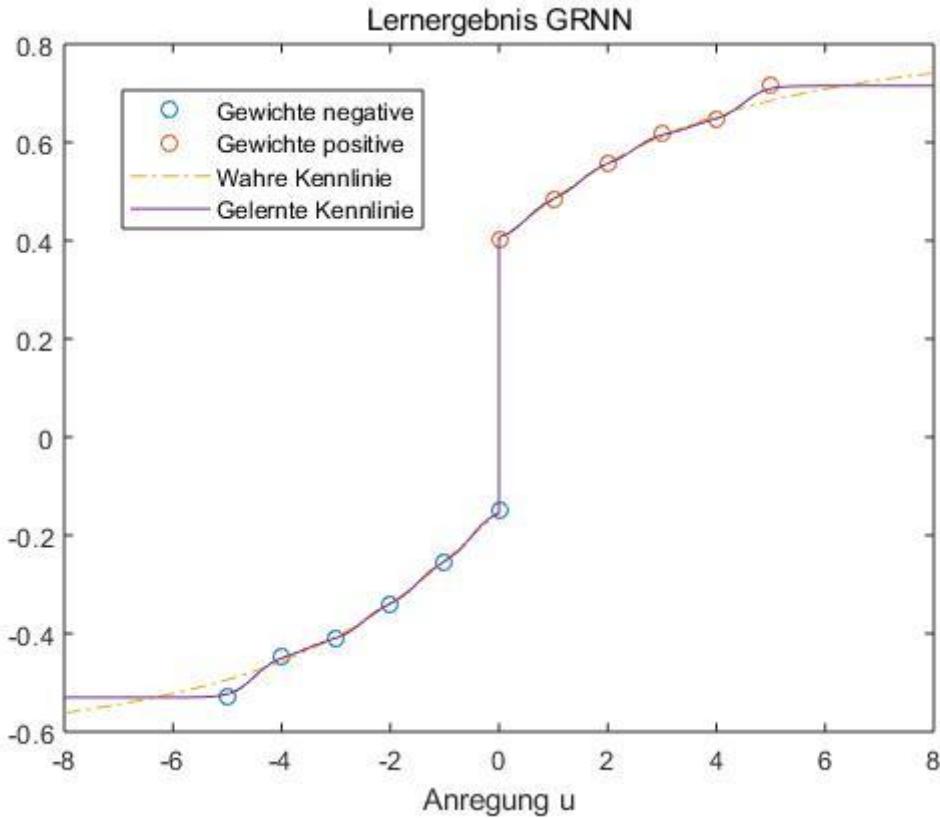


Abb. 2.15: Lernergebnisse vom korrigierten GRNN mit der Unstetigkeit

Im Vergleich zu den Abbildungen 2.14 und 2.15 ist es ersichtlich, dass nach der Optimierung die Lernergebnisse um die Unstetigkeit (um $x=0$) viel besser aussehen.

Das Problem bei dieser Methode besteht darin, dass vor der Programmierung zuerst die genaue Position der Unstetigkeit im Eingangsbereich bekannt werden soll und diese als Trennung verschiedenes GRNNs verwenden müssen. Wenn die ursprüngliche Funktion mehrere Unstetigkeiten enthält, müssen dann entsprechend mehrere GRNNs eingestellt werden.

2.6 Gradientenabstiegsverfahren

Um das spezifische nichtlineare Problem mit dem obigen neuronalen Netzwerk zu lösen, ist es auch notwendig, einen geeigneten Gewichtsvektor $\hat{\theta}$ zu finden. Die Adaption der Gewichte der statischen Neuronalen Netze erfolgt mit Hilfe eines Lernalgorithmus, dem sogenannten *Lerngesetz*. Das wohl bekannteste Lernverfahren stellt das *Gradientenverfahren* dar. Dadurch sollen die Parameter bzw. die Gewichte des neuronalen Netzes so angepasst werden, dass die Abweichung zwischen dem Ausgang y des zu identifizierenden Systems und dem Ausgang \hat{y} des Neuronalen Netzes minimiert wird. Diese Abweichung zwischen wahren und geschätztem Wert wird als *Ausgangsfehler* [S10] bezeichnet.

$$e(\hat{\underline{\theta}}) = (y - \hat{y}(\hat{\underline{\theta}})) \#(2 - 8)$$

Ausgangspunkt für die folgenden Überlegungen ist das quadratische *Fehlermaß* $E(\hat{\underline{\theta}})$:

$$E(\hat{\underline{\theta}}) = \frac{1}{2} \cdot e^2(\hat{\underline{\theta}}) = \frac{1}{2} \cdot (y - \hat{y}(\hat{\underline{\theta}}))^2 \#(2 - 9)$$

Die Einführung des Faktors $\frac{1}{2}$ ist für die Adaption der Gewichte unerheblich, führt jedoch zu einem übersichtlicheren Lerngesetz.

Der Zweck des obigen Lernverfahrens besteht darin, eine Beziehungsgleichung zwischen dem Ausgabefehler $E(\hat{\underline{\theta}})$ und dem Gewicht $\hat{\underline{\theta}}$ herzustellen und danach die Beziehungsgleichung zu analysieren und zu optimieren. Nach dem Optimierungsprozess wird die günstigste Gewichtsmatrix erhalten, mit der das Fehlermaß $E(\hat{\underline{\theta}})$ zu 0 oder dem Minimumswert wird. Da im Allgemeinen das Fehlermaß $E(\hat{\underline{\theta}})$ nicht analytisch vorliegt, bzw. dessen Ableitung bezüglich der Parameter $\hat{\underline{\theta}}$ nicht analytisch berechnet werden kann, ist der Nutzer hier auf eine iterative Lösung angewiesen [P91].

Die grundsätzliche algorithmische Struktur besteht aus nachfolgenden Schritten und wird in folgender Abbildung 2.14 für den zweidimensionalen Fall illustriert.

1. Wahl eines Startpunktes $\hat{\underline{\theta}}^{(0)}$ und Festlegung des Iterationsindexes zu $l = 0$.
2. Bestimmung einer Suchrichtung $\underline{s}^{(l)}$
3. Bestimmung einer skalaren Schrittweite $\eta^{(l)} > 0$ durch Lösung des folgenden eindimensionalen Minimierungsprobleme:

$$\min_{\eta > 0} E = (\hat{\underline{\theta}}^{(l)} + \eta^{(l)} \underline{s}^{(l)}) \#(2 - 10)$$

anschließend ergibt sich der $(l + 1)$ -te Parametervektor aus

$$\hat{\underline{\theta}}^{(l+1)} = \hat{\underline{\theta}}^{(l)} + \eta^{(l)} \underline{s}^{(l)} \#(2 - 11)$$

4. Ist ein geeignetes Abbruchkriterium erfüllt, dann wird die Iteration gestoppt. Ansonsten:
5. Muss mit einer neuen Iteration $l := l + 1$ und einer Rücksprung nach 2 begonnen werden.

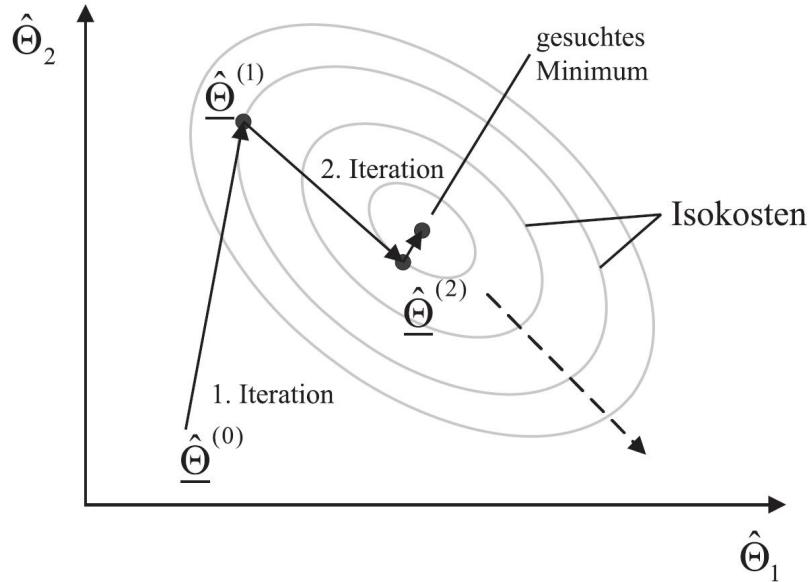


Abb. 2.16: Iterative Suche eines lokalen Minimums [S10]

Das in [PM91] beschriebene Gradientenabstiegsverfahren verwendet für die Suchrichtung am jeweiligen Iterationspunkt die Richtung des steilsten Abstiegs, d. h. die negative Gradientenrichtung $-g(\hat{\theta})$

$$\underline{s}^{(l)} = -\frac{\partial E(\hat{\theta}^{(l)})}{\partial \hat{\theta}} = -g(\hat{\theta}) \#(2-12)$$

Es sei an dieser Stelle ausdrücklich darauf hingewiesen, dass die Suchrichtung nicht immer der negative Gradient sein muss. In dieser Arbeit wird beim Gradientenabstiegsverfahren auf die Bestimmung der skalaren Schrittweite η für jeden Iterationsschritt verzichtet. Es wird eine geeignet gewählte Schrittweite η als konstant angesetzt. Diese Schrittweite wird oft auch als *Lernschrittweite* oder auch *Lernfaktor* bezeichnet. Für die Änderung der Gewichte des Neuronalen Netzes ergibt sich somit in zeitdiskreter Schreibweise folgendes Lerngesetz

$$\hat{\theta}[k+1] = \hat{\theta}[k] - \eta \cdot \frac{\partial E(\hat{\theta}[k])}{\partial \hat{\theta}} \#(2-12)$$

In zeitkontinuierlicher Form lautet das Lerngesetz

$$\frac{d\hat{\theta}}{dt} = -\eta' \cdot \frac{\partial E(\hat{\theta})}{\partial \hat{\theta}} \#(2-13)$$

wobei die Lernschrittweite η' der mit der Abtastzeit gewichteten Lernschrittweite für den zeitdiskreten Fall entspricht [S10].

Um die Gradientenabstiegsverfahren besser zu verstehen, bietet diese Arbeit hier einen anderen eindimensionalen Verständnisgedanke.

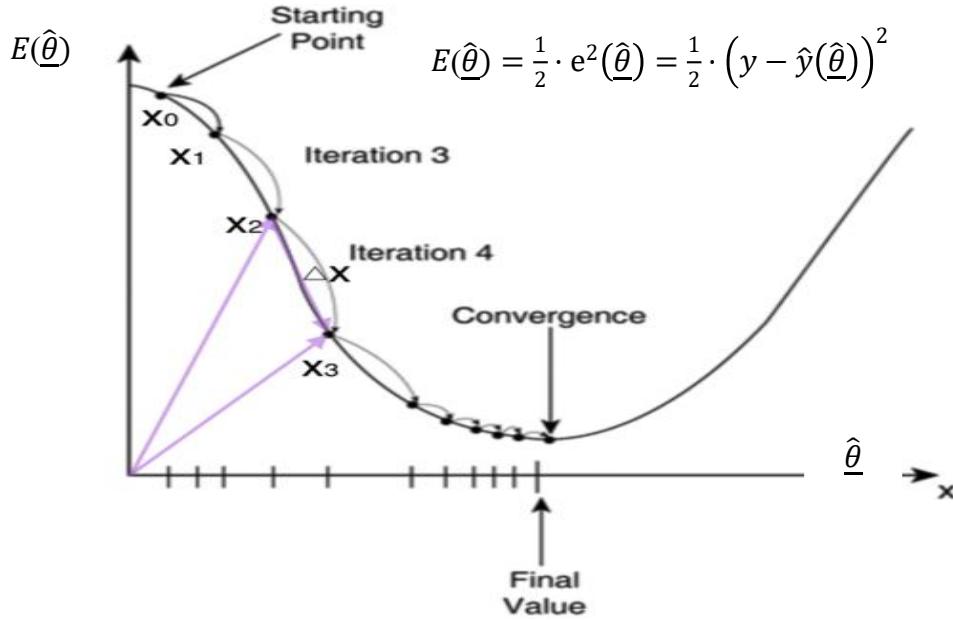


Abb. 2.17: Eindimensionale iterative Suche eines lokalen Minimums [W16]

In der Abbildung 2.11 stellt die Gleichungskurve in 2-ter Ordnung die Beziehung zwischen Fehlermaß $E(\hat{\theta})$ und Gewichtsvektor $\hat{\theta}$ dar. Es zeigt sich, dass das Problem des Suchens nach minimalem Fehlermaß mit dem Problem des Suchens nach dem Extrempunkt der quadratischen Gleichung gleichgesetzt ist. Das Ableitungsverfahren wird in dieser Arbeit nicht detailliert beschrieben. Die Ergebnisse sind gleich wie die in erstem Verfahren.

2.7 Lerngesetz für das RBF-Netz und GRNN

Ausgehend von den Gleichungen im Abschnitt 2.5 für den Schätzwert \hat{y} am Ausgang eines RBF-Netzes

$$\hat{y}(\underline{u}) = \sum_{i=1}^p \hat{\theta}_i \cdot A_i(\underline{u}) \quad (2-14)$$

wird zur Adaption der Gewichte der bereits oben eingeführte Lernfehler e bzw. der daraus abgeleitete quadratische Fehler E herangezogen.

$$e(\underline{u}) = \sum_{i=1}^p \hat{\theta}_i \cdot A_i(\underline{u}) - y(\underline{u}) \quad (2-15)$$

$$E(\underline{u}) := \frac{1}{2} e^2 = \frac{1}{2} \left(\sum_{i=1}^p \hat{\theta}_i \cdot A_i(\underline{u}) - y(\underline{u}) \right)^2 \quad (2-16)$$

Dazu wird der quadratische Fehler nach dem jeweiligen Gewicht abgeleitet.

$$\frac{\partial E(\underline{u})}{\partial \hat{\theta}_i} = \left(\sum_{k=1}^p \hat{\theta}_k \cdot A_k(\underline{u}) - y(\underline{u}) \right) \cdot A_i(\underline{u}) = e(\underline{u}) \cdot A_i(\underline{u}) \# (2-17)$$

Somit bestimmen sich die notwendigen Änderungen der Gewichte zueinander wie es dem Beitrag jedes Gewichts zum Schätzwert \hat{y} und damit zum Lernfehler e entspricht. Eine zusätzliche Skalierung mit einem Lernfaktor η dient der Einstellung einer gewünschten Lerngeschwindigkeit bzw. Glättungswirkung bei der Adaption. Das negative Vorzeichen stellt eine Anpassung der Gewichte in Richtung kleinerer Fehler sicher. Das vollständige Lerngesetz für jedes Gewicht lautet

$$\frac{d}{dt} \hat{\theta}_i = -\eta \cdot e \cdot A_i \# (2-18)$$

Dadurch wird der quadratische Fehler nach mehrmals Iterationen minimiert.

Zusammenfassend sind die iterativen gewichteten Gleichungen des diskreten RBF-Netzes und des GRNN wie folgt

$$\hat{\theta}[k+1] = \hat{\theta}[k] - \eta \cdot e(\underline{u}) \cdot A_i(\underline{u}) \# (2-19)$$

2.8 Lernfähiger Luenberger-Beobachter

In diesem Kapitel wird die Identifikation statischer Nichtlinearitäten mit einer Systemidentifikation für eine spezielle Klasse von nichtlinearen dynamischen Systemen kombiniert. Zu diesem Zweck wird ein lernfähiger Beobachter [S95] entworfen, der zwei Aufgaben erfüllen soll:

- Schätzung aller Systemzustände
- Identifikation einer statischen Nichtlinearität

Diese Struktur wird für rein mathematische Modelle ohne Messwerte verwendet. Zunächst soll ein Modell mit allen bekannten Parametern im Matlab/Simulink erstellt werden, alle erforderlichen Eingangswerte gehen in das Modell ein und anschließend werden entsprechende Ausgangswerte aus dem Modell entnommen. Diese Ein- und Ausgangswerte entsprechen dem gemessenen Wert eines realen Systems und werden als Ein- und Ausgabe vom mitverbundenen neuronalen Netzwerkmodell übernommen, um die Parameter darin zu trainieren.

Die für die weiteren Inhalte der Arbeit nützliche lernfähigere Zustandsbeobachterstruktur ist in Abbildung 2.12 zu sehen.

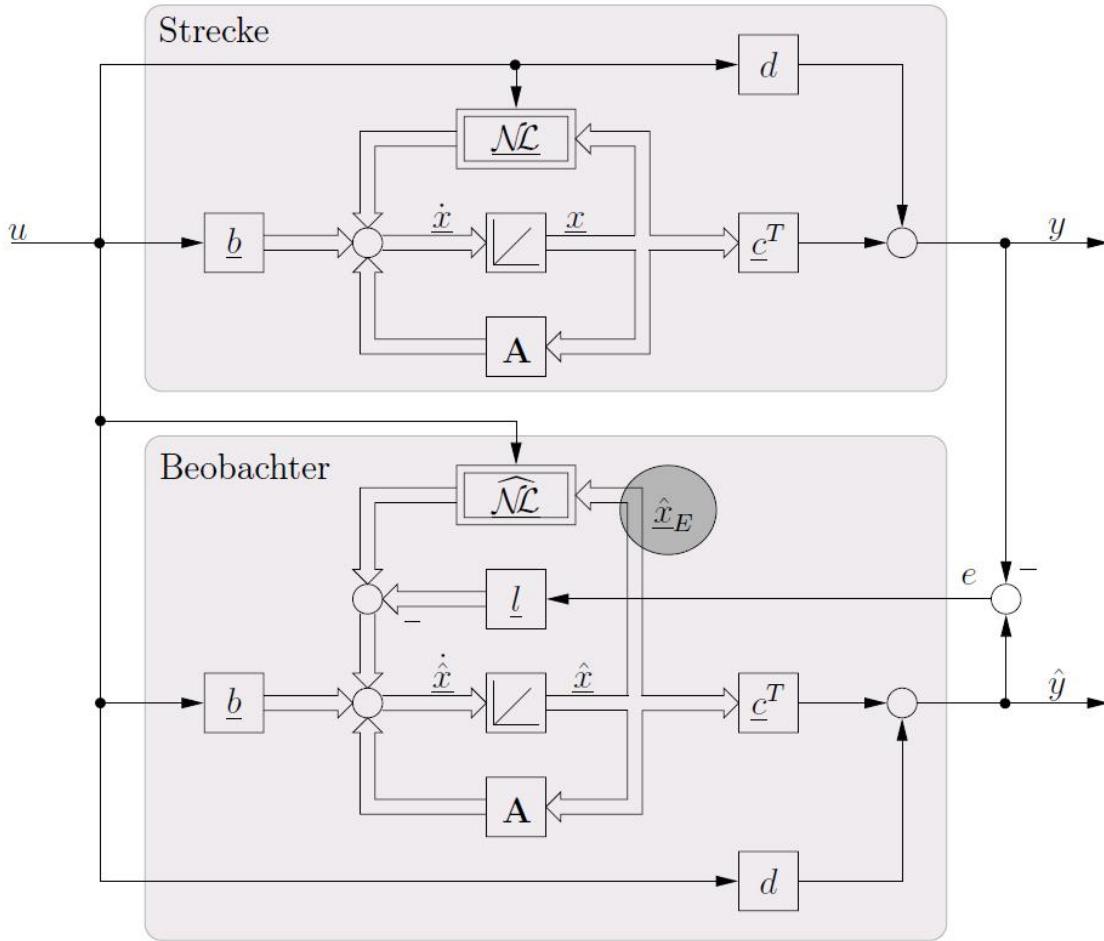


Abb. 2.18: Lernfähiger Zustandsbeobachter mit isolierter Nichtlinearität; Ziel:
Beobachtung nichtmessbarer Zustände und Identifikation der Nichtlinearität
([S95], Patent)

Die entsprechende differenzielle Darstellung als Differentialgleichung lautet:

$$\dot{\underline{x}} = (A - l \cdot \underline{c}^T) \cdot \underline{x} + \underline{b} \cdot u + \underline{k}_{NL} \cdot \widehat{NL}(\hat{x}_E, u) + \underline{l} \cdot \underline{c}^T \cdot \underline{x}$$

$$\text{und } \hat{y} = \underline{c}^T \cdot \hat{x} + d \cdot u \#(2-20)$$

Dabei sind die Parameter A , \underline{b} , \underline{c} , \underline{k}_{NL} und d lautet der Beobachteransatz bekannt.

Es ist hilfreich, bei der Modellierung und Analyse des nichtlinearen Systems das rekurrente Netz zu einem Luenberger-Beobachter zu erweitern. Dadurch wird zum einen die Anfangswertproblematik gelöst, und zum anderen werden die Zustände so beibehalten, dass der Ausgangsfehler klein bleibt und stationär zu Null, nämlich Konvergenz, wird. In nachfolgenden Kapiteln der Arbeit erfolgt immer Erweiterung der Modelle zum Luenberger-Beobachter.

3 Erstellung strukturiertes rekurrentes Netzes

Von diesem Kapitel wird begonnen, eine spezielle Modellierungsstruktur, d. h. rekurrentes neuronales Netz, zu verwenden, um den Ein-/Ausgangsverhalten einfacher nichtlinearer dynamischer Systeme zu approximieren, ohne dass die Möglichkeit besteht, aus dem Identifikationsergebnis auf interne Zustände, lineare Parameter oder auch nichtlineare Charakteristiken schließen zu können.

In einem einfachen Beispiel, der Identifikation der Mechanik einer elektrischen Maschine, wird die Verwendung eines rekurrenten neuronalen Netzes vorgestellt. Hierbei wird das Vorwissen über die Struktur des zu identifizierenden Systems berücksichtigt. Dieses rekurrente Netz wird in eine Beobachterstruktur implementiert, wodurch das Problem der Anfangswertfindung der Systemzustände vermieden wird [DC10]. Mit diesem vorstrukturierten bzw. strukturierten rekurrenten Netz wird zudem die physikalische Interpretierbarkeit des Identifikationsergebnisses erreicht. Die Inhalte dieses Kapitels sind von [S10] entnommen.

3.1 Strukturierte rekurrente Netze

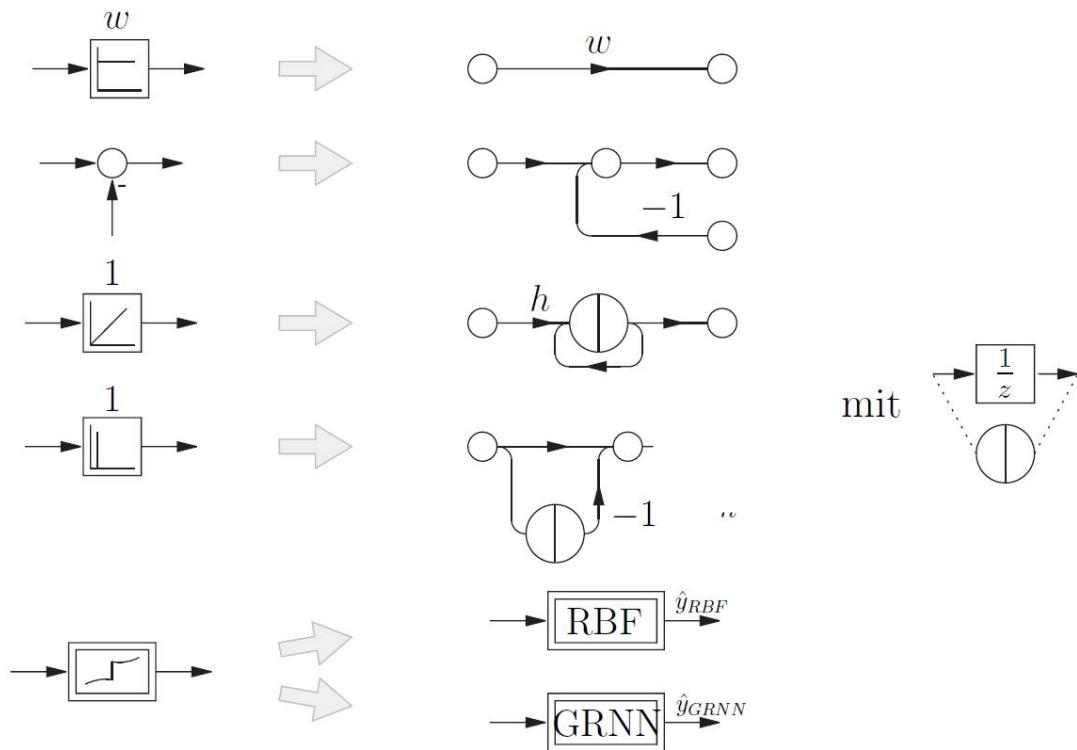


Abb. 3.1: Elementare Operatoren eines Signalflussplanes und die äquivalenten Operatoren des strukturierten rekurrenten Netzes mit der Abtastzeit h [S10]

Wie in der obigen Abbildung gezeigt, ist das strukturierte rekurrente Netz ähnlich wie ein Signalflussplan aufgebaut und besteht ebenfalls aus den elementaren Operatoren (Verstärker, Addierer, Integrierer, Differenzierer und Multiplikator) sowie den unbekannten Nichtlinearitäten. Der Unterschied besteht darin, dass im strukturierten rekurrenten Netz die diskrete Form von elementaren Operatoren verwendet wird. Hierbei werden die Summationspunkte „ \circ “ des Signalflussplanes zu *Neuronen* und die linearen Parameter ω zu den *Gewichten* zwischen den Neuronen des rekurrenten Netzes. Die Integratoren werden mit Hilfe von Zeitverzögerungsgliedern gemäß der Integrationsregel

$$y_{int}[k+1] = h \cdot u_{int}[k+1] + y_{int}[k]\#(3-1)$$

mit dem Eingangswert des Integrators u_{int} , dem Ausgangswert des Integrators y_{int} und der Abtastzeit h nach *L. Euler* (häufig als *Euler-Vorwärts-Approximation* oder auch als *Rechteckapproximation* bezeichnet) implementiert. Entsprechend ist für die numerische Differentiation auch die Formel

$$y_{diff}[k+1] = \frac{1}{h} \cdot (u_{diff}[k+1] - u_{diff}[k])\#(3-2)$$

möglich. Dabei sind u_{diff} der Eingang und y_{diff} der Ausgang des Differentiationsgliedes. Außerdem werden statische Nichtlinearitäten, wie das RBF-Netz und GRNN, als Subnetze (ein Subsystem) im strukturierten rekurrenten Netz berücksichtigt.

3.2 Anwendung der Transformation

Um die Struktur vom strukturierten rekurrenten Netz intuitiver zu verstehen, wird in diesem Abschnitt ein einfaches Beispiel angewandt, das einer leerlaufenden elektrischen Maschine. Die Bewegungsdifferentialgleichung für dieses System lautet

$$\dot{\Omega} = \frac{1}{J} \cdot (M_L - M_w(\Omega))\#(3-3)$$

mit

- der Winkelgeschwindigkeit Ω in $\frac{rad}{s}$,
- dem Massenträgheitsmoment J in $kg \cdot m^2$,
- dem Luftspaltdrehmoment (antreibendes Drehmoment) M_L in $N \cdot m$ und
- dem Reibungsdrehmoment (Widerstandsdrehmoment) $M_w(\Omega)$ in $N \cdot m$ abhängig von der Winkelgeschwindigkeit.

Der entsprechende Signalflussplan und das strukturierte rekurrente Netz sind in Abbildung 3.2 zu sehen.

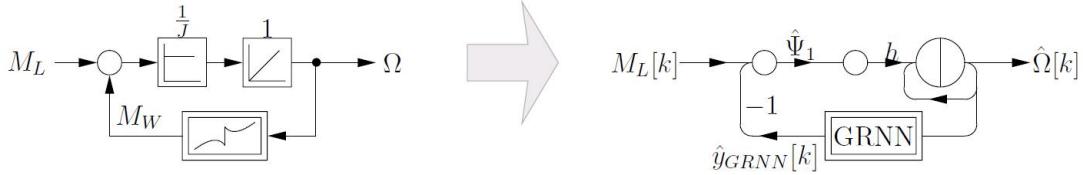


Abb. 3.2: Transformation der Mechanik einer elektrischen Maschine in ein rekurrentes Netz [S10]

Der lineare Parameter in diesem System im Signalflussplan ist $\frac{1}{J}$, der dem Kehrwert des Massenträgheitsmoments entspricht und als ein Gewicht $\hat{\psi}_1$ im strukturierten rekurrenten Netz dargestellt ist. Der GRNN-Block entspricht die Reibungskennlinie $M_w(\Omega)$. Da hier zur Approximation der Reibungskennlinie GRNN genutzt wird (siehe Abschnitt 2.4), sind darin r nichtlineare Parameter enthalten, von $\hat{\theta}_1$ bis $\hat{\theta}_r$.

Für das rekurrente Netz ergibt sich entsprechend der Transformation aus Abbildung 3.2 die diskrete Differenzengleichung

$$\hat{\Omega}[k+1] = \hat{\Omega}[k] + h \cdot \hat{\psi}_1 \cdot (M_L[k] - \hat{y}_{GRNN}(\hat{\Omega}[k], \underline{\hat{\theta}})) \quad \#(3-4)$$

mit

- der geschätzten Winkelgeschwindigkeit $\hat{\Omega}[k]$
- dem Luftspaltdrehmoment in der k-ten Iteration $M_L[k]$
- dem geschätzten Reibungsdrehmoment \hat{y}_{GRNN} abhängig von $\hat{\Omega}[k]$ und $\underline{\hat{\theta}}$
- des linearen Parameters, auch Kehrwert des Massenträgheitsmoments $\hat{\psi}_1$
- des nichtlinearen Parameters, nämlich die Stützwerte des GRNN $\underline{\hat{\theta}} = [\hat{\theta}_1 \dots \hat{\theta}_r]^T$

In der Gleichung (3-4) bestehen neben dem Luftspaltdrehmoment, dem Eingangswert, $M_L[k]$ und der geschätzten Winkelgeschwindigkeit, dem Zustandsparameter $\hat{\Omega}[k]$, auch die Parameter, die vom rekurrenten neuronalen Netz bestimmt werden müssen, aus zwei Teilen: dem linearen Parameter $\hat{\psi}_1$ und der nichtlinearen Parametervektor $\underline{\hat{\theta}} = [\hat{\theta}_1 \dots \hat{\theta}_r]^T$, die zusammen zum Parametervektor

$$\underline{\hat{\omega}} = [\hat{\omega}_1 \dots \hat{\omega}_p]^T = [\hat{\psi}_1 \ \hat{\theta}_1 \dots \hat{\theta}_r]^T \quad \#(3-5)$$

des rekurrenten Netzes zusammengefasst werden. Darin sind entsprechend alle zu identifizierenden Gewichte enthalten.

In dieses Beispiel mit nur einem linearen Parameter liegt in der Gleichung (3-5) das folgende Verhältnis vor:

$$\hat{\omega}_1 = \hat{\psi}_1 \text{ und } \hat{\omega}_n = \hat{\theta}_{n-1} \text{ für } 1 < n \leq p$$

Wenn es mehr als einem linearen Parameter, z.B. s linearen Parameter, und einem nichtlinearen Parametervektor gibt, ist das Verhältnis wie in Gleichung (3-6) gegeben:

$$\hat{\omega}_n = \hat{\theta}_{n-s} \text{ für } 1 < n \leq p \#(3 - 6)$$

Im Allgemeinen setzt sich der Parametervektor aus p Elementen zusammen und beinhaltet mehrere lineare Parameter sowie einer oder mehrere nichtlineare Parametervektor, wenn es im dynamischen System mehr als einen nichtlinearen Teil gibt.

3.3 Parameteradaption

Für die Parameteradaption wird die in Abschnitt 2.6 vorgestellte Ausgangsfehleranordnung, nämlich Gradientenabstiegsverfahren, verwendet. Dies ist in Abbildung 3.3 noch einmal anhand des zusammengefassten Parametervektors dargestellt.

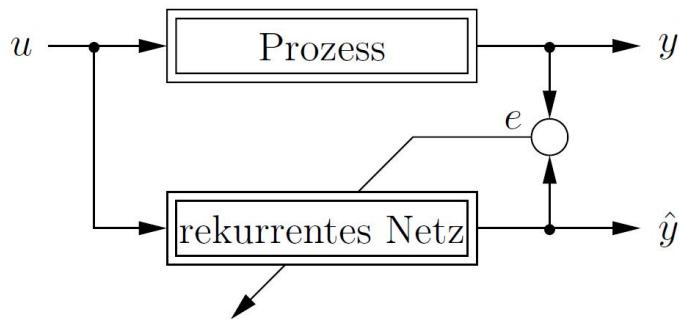


Abb. 3.3: Prinzip der Ausgangsfehleranordnung [S10]

Ausgangspunkt ist der quadratische Fehler

$$E(\hat{\underline{\omega}}) = \frac{1}{2} \cdot e^2(\hat{\underline{\omega}}) = \frac{1}{2} \cdot (y - \hat{y}(\hat{\underline{\omega}}))^2 \#(3 - 7)$$

Analog zu Gleichung (2-12) ergibt sich folgende Gleichung für die Parameteradaption

$$\hat{\underline{\omega}}[k + 1] = \hat{\underline{\omega}}[k] - \eta \cdot \left[\frac{\partial E[\hat{\underline{\omega}}, k]}{\partial \hat{\omega}_1} \dots \frac{\partial E[\hat{\underline{\omega}}, k]}{\partial \hat{\omega}_p} \right] \Big|_{\hat{\underline{\omega}}[k]} \#(3 - 8)$$

η ist hierbei die an das Problem angepasste Lernschrittweite, wobei die Bedingung $\eta > 0$ erfüllt sein muss.¹⁾

Zur übersichtlicheren Schreibweise wird der Nabla-Operator eingeführt

¹⁾ Mit $\eta < 0$ führt das Gradientenverfahren auf ein lokales Maximum in der Funktion E , womit eine konvergente Identifikation nicht möglich ist. [S10]

$$\nabla = \left[\frac{\partial}{\partial \hat{\omega}_1} \cdots \frac{\partial}{\partial \hat{\omega}_P} \right]^T$$

womit die Gleichung zur Parameteradaption in der Form

$$\underline{\hat{\omega}}[k+1] = \underline{\hat{\omega}}[k] - \eta \cdot \nabla E[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]} \#(3-9)$$

angegeben werden kann. Der Gradient $\nabla E[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]}$ kann entsprechende der Definition von E nach Gleichung (3-7) weiter zerlegt werden:

$$\nabla E[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]} = \frac{1}{2} \cdot 2 \cdot \left(\underbrace{y[k] - \hat{y}[\underline{\hat{\omega}}, k]}_{e[\underline{\hat{\omega}}, k]} \right) \cdot \left(-\nabla \hat{y}[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]} \right) = -e[\underline{\hat{\omega}}, k] \cdot \nabla \hat{y}[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]} \#(3-10)$$

Wird dieser Gradient in die Gleichung zur Parameteradaption eingesetzt, ergibt sich das Lerngesetz nach dem Gradientenabstiegsverfahren als

$$\underline{\hat{\omega}}[k+1] = \underline{\hat{\omega}}[k] + \eta \cdot e[\underline{\hat{\omega}}, k] \cdot \nabla \hat{y}[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]} \#(3-11)$$

Dieses Lerngesetz wird dahingehend erweitert, dass die Lernschrittweite η nicht ein skalarer Wert ist, sondern für jedes Gewicht $\hat{\omega}_i$ eine eigene Lernschrittweite η_i vorhanden ist. Damit erweitert sich η zu einem Vektor. Zusätzlich wird für jeden Parameter ein *Momentumterm* $0 \leq \alpha_i < 1$ eingeführt, der in die Berechnung der aktuellen Gewichtsänderung auch die vergangenen Gewichtsänderungen einbezieht. Der *Momentumterm* hat den Vorteil, dass die Gewichtsanpassung unempfindlicher gegenüber Plateaus in der Fehlerebene E und Rauschanteilen im Gradienten ∇E wird. [S10]

Nach dem Einsetzen dieser Ergänzung verändert die Form des Lerngesetzes sich zu

$$\underline{\hat{\omega}}[k+1] = \underline{\hat{\omega}}[k] + \Delta \underline{\hat{\omega}}[k] \#(3-12)$$

mit

$$\Delta \underline{\hat{\omega}}[k] = e[k] \cdot \text{diag}(\underline{\eta}) \cdot \nabla \hat{y}[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]} + \text{diag}(\underline{\alpha}) \cdot \Delta \underline{\hat{\omega}}[k-1]$$

Durch diesem Lerngesetz kann der neue Parametervektor $\underline{\hat{\omega}}$ im Abtastschritt $k+1$ nicht nur von dem Parametervektor $\underline{\hat{\omega}}[k]$, dem Ausgangsfehler $e[k]$ und den partiellen Ableitungen $\nabla \hat{y}[\underline{\hat{\omega}}, k] \Big|_{\underline{\hat{\omega}}[k]}$ im letzten Abtastschritt k sondern auch von der vergangenen Gewichtsänderung $\Delta \underline{\hat{\omega}}[k-1]$ abhängig sein.

Weil bei dem Lerngesetz eine iterative Methode genutzt wird, ist die Bestimmung der Startwerte des Parametervektors $\underline{\hat{\omega}}[0]$ erforderlich. Die Startwerte stellen Vorwissen dar, das bei der Identifikation zusätzlich zur Struktur des rekurrenten Netzes eingebracht werden muss. Wegen der Initialisierung des Gradientenverfahrens mit konstanten Startwerten $\underline{\hat{\omega}}[0]$ ergibt sich für die Gewichtsänderung $\underline{\hat{\omega}}[0] = \underline{0}$ [S10]. Der Nachteil dieses Verfahrens ist, dass die Werte der Lernparameter $\underline{\eta}$ und $\underline{\alpha}$ eine große Wirkung auf die Konvergenzgeschwindigkeit der Gewichte haben, aber manuell anhand eigener

Erfahrungen bestimmt werden müssen. Außerdem ist die Berechnung von der partiellen Ableitung $\nabla \hat{y}[\hat{\omega}, k] \Big|_{\hat{\omega}[k]}$ zum Abtastschritt k notwendig.

In den nachfolgenden Abschnitten wird zuerst die Zustandsdarstellung für die strukturierten rekurrenten Netze eingeführt und dann anhand der Zustandsdarstellung die partielle Ableitung allgemein und für RBF-Netze und GRNN berechnet.

3.4 Zustandsdarstellung

Im Folgenden wird eine allgemeine Vorschrift zur Berechnung der Zustandsdarstellung für das normale nichtlineare SISO-System entwickelt.

In der zeitkontinuierlichen Darstellung ist die Zustandsbeschreibung ein System aus nichtlinearen Differentialgleichungen erster Ordnung, das in der Form

$$\dot{x} = A \cdot x + b \cdot u + K_{NL} \cdot NL(u, x) \quad \text{und} \quad y = c^T \cdot x \quad (3-13)$$

dargestellt werden kann. Der Anteil $K_{NL} \cdot NL(u, x)$ repräsentiert die isoliert eingreifenden Nichtlinearitäten. A , b und c bilden den linearen Systemanteil. In dieser Form wird durch die Zustandsbeschreibung ein nicht sprungfähiges System beschrieben (Durchgriff $d = 0$). Die Zustandsbeschreibung ist in Abbildung 3.4 graphisch dargestellt.

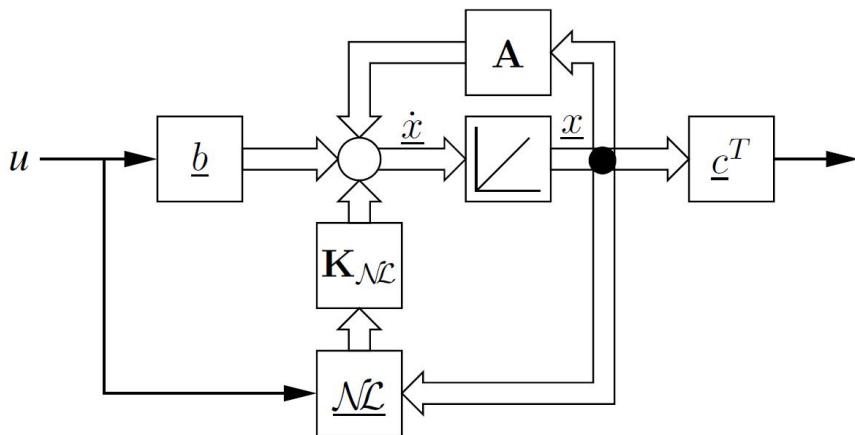


Abb. 3.4: Nichtlineare kontinuierliche Zustandsbeschreibung [S10]

Dabei sind

- u der skalare Systemeingang,
- $x \in \mathbb{R}^n$ der Zustandsvektor mit n Zuständen,
- $A \in \mathbb{R}^{n \times n}$ die Systemmatrix,
- $b \in \mathbb{R}^n$ der Einkopplungsvektor,
- $K_{NL} \in \mathbb{R}^{n \times q}$ die Kopplungsmatrix der Nichtlinearitäten,

- $\underline{NL}(u, \underline{x}): R \times R^n \rightarrow R^q$ der Vektor der statischen Nichtlinearitäten mit q skalaren Funktionen,
- $\underline{c} \in R^n$ der Auskopplungsvektor und
- y der skalare Systemausgang.

In derselben Weise wie die kontinuierlichen Zustandsgleichungen aus dem Signalflussplan abgeleitet werden, können die diskreten Zustandsgleichungen aus dem rekurrenten Netz bestimmt werden. Dazu wird von einer kontinuierlichen Abbildung der Strecke

$$\dot{\underline{x}} = \tilde{A} \cdot \underline{x} + \tilde{b} \cdot u + \tilde{K}_{NL} \cdot \underline{NL}(u, \underline{x}) \quad \text{und} \quad \tilde{y} = \underline{c}^T \cdot \underline{x} \#(3 - 14)$$

ausgegangen. Die Tilde kennzeichnet, dass es sich um geschätzte kontinuierliche Größen handelt. Die Werte von \tilde{A} , \tilde{b} , \tilde{K}_{NL} und \underline{NL} sind zunächst unbekannt und sollen im weiteren Verlauf identifiziert werden. Der Auskopplungsvektor c stellt dabei eine Ausnahme dar, da die Berechnung der partiellen Ableitungen davon ausgeht, dass die Elemente des Auskopplungsvektors bekannt sind. Damit wird die Beziehung $\underline{c} = \underline{c}$ für die Berechnung der partiellen Ableitungen vorausgesetzt. Dies ist aber nicht absolut und sollte nach spezifischen Problemen analysiert werden. Die Funktionen im Vektor \underline{NL} sind durch statische Funktionsapproximatoren ersetzt.

Mit der Rechteckapproximation der Integratoren gilt für einen Zustand des rekurrenten Netzes $\hat{x}_i[k]$ die Beziehung

$$\hat{x}_i[k+1] = h \cdot \hat{u}_{int}[k] + \hat{x}_i[k] \#(3 - 15)$$

Wird diese Gleichung auf alle Zustände erweitert, ergibt sich mit den Bedingungen $\hat{x}[k] = \underline{x}[k]$ und $\hat{u}_{int}[k] = \tilde{A} \cdot \underline{x}[k] + \tilde{b} \cdot u[k] + \tilde{K}_{NL} \cdot \underline{NL}(u[k], \underline{x}[k])$, die für kleine Abtastzeiten h gelten, die Gleichung

$$\hat{x}[k+1] = h \cdot \left(\tilde{A} \cdot \hat{x}[k] + \tilde{b} \cdot u[k] + \tilde{K}_{NL} \cdot \underline{NL}(u[k], \hat{x}[k]) \right) + \hat{x}[k] \#(3 - 16)$$

Durch Zusammenfassen kann diese Gleichung auf die Form

$$\hat{x}[k+1] = \underbrace{[h \cdot \tilde{A} + E]}_{\hat{A}} \cdot \hat{x}[k] + \underbrace{h \cdot \tilde{b}}_{\hat{b}} \cdot u[k] + \underbrace{h \cdot \tilde{K}_{NL} \cdot \underline{NL}(u[k], \hat{x}[k])}_{\hat{K}_{NL}} \#(3 - 16)$$

gebracht werden. Dies ergibt die diskrete Zustandsbeschreibung

$$\hat{x}[k+1] = \hat{A} \cdot \hat{x}[k] + \hat{b} \cdot u[k] + \hat{K}_{NL} \cdot \underline{NL}(u[k], \hat{x}[k])$$

$$\text{und} \quad \hat{y}[k] = \underline{c}^T \cdot \hat{x}[k] \#(3 - 17)$$

mit

- $\hat{A} = h \cdot \tilde{A} + E$

- $\hat{b} = h \cdot \tilde{b}$
- $\hat{K}_{NL} = h \cdot \tilde{K}_{NL}$
- $\widehat{NL}(u[k], \hat{x}[k]) = \widetilde{NL}(u[k], \underline{\hat{x}}[k])$ und
- $\hat{c} = \tilde{c} = c$.

Diese Gleichungen gelten nur für die Euler-Vorwärts-Approximation der Integratoren und müssen für andere Methoden der numerischen Integration getrennt bestimmt werden. Die Gleichungen (3-17) sind in Abbildung 3.5 graphisch dargestellt.

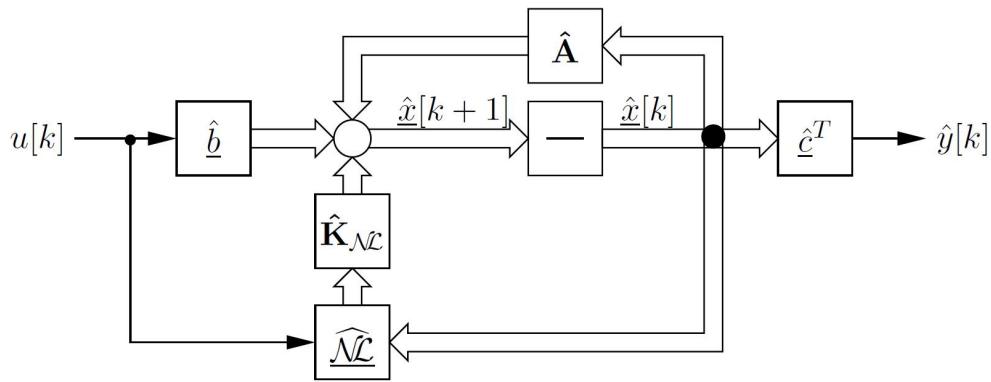


Abb. 3.5: Diskrete Zustandsbeschreibung [S10]

Im Anschluss kann dieses Modell zum Luenberger-Beobachter erweitert werden. Dies führt auf folgende kontinuierliche Beschreibung des Rekurrenten Netzes:

$$\dot{\tilde{x}} = \underbrace{(\tilde{A} + \tilde{l} \cdot \tilde{c}^T)}_{\tilde{A}_{beo}} \cdot \tilde{x} + \tilde{b} \cdot u - \tilde{l} \cdot y + \tilde{K}_{NL} \cdot \widetilde{NL}(u, \tilde{x})$$

und $\tilde{y} = \tilde{c}^T \cdot \tilde{x}$ (3-18)

Ähnlich wie Gleichung (3-15) kann diese Formel in eine diskrete Form umgewandelt werden

$$\hat{x}[k+1] = h \cdot \left((\tilde{A} + \tilde{l} \cdot \tilde{c}^T) \cdot \hat{x}[k] + \tilde{b} \cdot u[k] - \tilde{l} \cdot y[k] + \tilde{K}_{NL} \cdot \widetilde{NL}(u[k], \hat{x}[k]) \right) + \hat{x}[k] \# (3-19)$$

Durch einfache Zusammenfassungen und durch Berücksichtigung der Ausgangsgleichung ergibt sich aus dieser Gleichung die diskrete Zustandsbeschreibung des zum Luenberger-Beobachter erweiterten rekurrenten Netzes

$$\hat{x}[k+1] = \hat{A}_{rek} \cdot \hat{x}[k] + \hat{b} \cdot u[k] - \hat{l} \cdot y[k] + \hat{K}_{NL} \cdot \widetilde{NL}(u[k], \hat{x}[k])$$

und $\hat{y}[k] = \hat{c}^T \cdot \hat{x}[k] \# (3-20)$

mit

- $\widehat{A}_{rek} = h \cdot \widetilde{A}_{beo} + E = h \cdot (\widetilde{A} + \tilde{l} \cdot \tilde{c}^T) + E$
- $\widehat{b} = h \cdot \tilde{b}$
- $\widehat{l} = h \cdot \tilde{l}$
- $\widehat{K}_{NL} = h \cdot \widetilde{K}_{NL}$
- $\underline{\widehat{NL}}(u[k], \underline{\widehat{x}}[k]) = \underline{\widetilde{NL}}(u[k], \underline{\widehat{x}}[k])$ und
- $\widehat{c} = \tilde{c} = c$.

Diese Gleichungen gelten nur für die Euler-Vorwärts-Approximation der Integratoren und müssen für exaktere Integrationsmethoden separat bestimmt werden.

Abbildung 3.6 beschreibt das entsprechende rekurrente Netz als Luenberger-Beobachter. Darin ist die Kombination des realen Systems mit dem diskreten Beobachter über Digital-Analog- bzw. Analog-Digital-Wandler dargestellt.

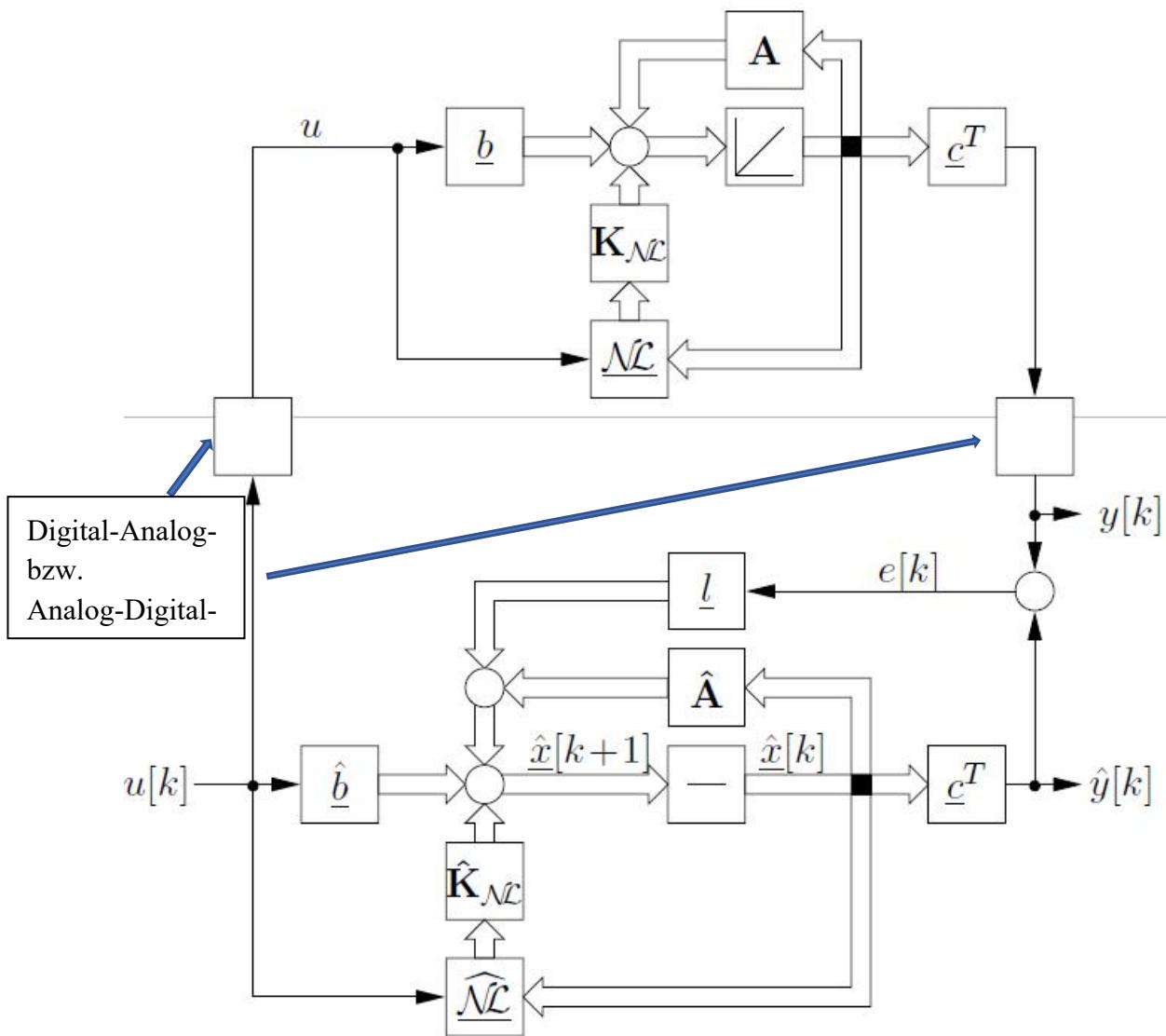


Abb. 3.6: Rekurrentes Netz als Luenberger-Beobachter [S10]

3.5 Partielle Ableitungen

Aus der diskreten Zustandsbeschreibung können die partiellen Ableitungen $\Delta\hat{y}$ zur Implementierung des Lerngesetzes berechnet werden. Sie ergeben sich nach Gleichung (3-20) wie folgt:

$$\nabla\hat{y} = \frac{\partial\hat{c}^T}{\partial\hat{\omega}} \cdot \hat{x} + \hat{c}^T \cdot \frac{\partial\hat{x}}{\partial\hat{\omega}} \quad \#(3-21)$$

Mit $\hat{c} = c = \text{konst}$ vereinfacht sich Gleichung (3-21) zu:

$$\nabla\hat{y} = c^T \cdot \frac{\partial\hat{x}}{\partial\hat{\omega}} \quad \#(3-22)$$

Somit reduziert sich die Bestimmung von $\Delta\hat{y}$ auf die Berechnung von $\frac{\partial\hat{x}}{\partial\hat{\omega}}$. Diese partiellen Ableitungen der Zustände nach den gesuchten Gewichten können mit Hilfe der Zustandsberechnung aus Gleichung (3-20) bestimmt werden. Hierfür ist es jedoch zunächst zweckmäßig neben dem bereits eingeführten Nabla-Operator die Jacobi-Matrix $\hat{J} \in R^{n_d \times p}$ mit n_d diskreten Zuständen und p Parametern einzuführen [DC10].

$$\frac{\partial\hat{x}[k]}{\partial\hat{\omega}} = \hat{J}_{\hat{x}[k]} = \begin{bmatrix} (\nabla\hat{x}_1[k])^T \\ \vdots \\ (\nabla\hat{x}_{n_d}[k])^T \end{bmatrix} = \begin{bmatrix} \frac{\partial\hat{x}_1[k]}{\partial\hat{\omega}_1} & \dots & \frac{\partial\hat{x}_1[k]}{\partial\hat{\omega}_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial\hat{x}_{n_d}[k]}{\partial\hat{\omega}_1} & \dots & \frac{\partial\hat{x}_{n_d}[k]}{\partial\hat{\omega}_p} \end{bmatrix} \#(3-23)$$

Die partielle Differentiation der Systembeschreibung (3-20) nach den einzelnen Gewichten führt auf

$$\begin{aligned} \frac{\partial\hat{x}[k+1]}{\partial\hat{\omega}} &= \frac{\partial}{\partial\hat{\omega}}(\hat{A}_{rek} \cdot \hat{x}[k]) \\ &\quad + \frac{\partial}{\partial\hat{\omega}}(\hat{b} \cdot u[k]) \\ &\quad - \frac{\partial}{\partial\hat{\omega}}(\hat{l} \cdot y[k]) \\ &\quad + \frac{\partial}{\partial\hat{\omega}}(\hat{K}_{NL} \cdot \underline{NL}(u[k], \hat{x}[k])) \#(3-24) \end{aligned}$$

Wegen der Unabhängigkeit von den wahren Größen u und y gilt $\frac{\partial u[k]}{\partial\hat{\omega}} = 0$ und $\frac{\partial y[k]}{\partial\hat{\omega}} = 0$ sowie $\frac{\partial\hat{l}}{\partial\hat{\omega}} = 0$. Mit diesen Bedingungen und der Produktregel kann die obige Gleichung weiter umgeformt werden:

$$\begin{aligned} \frac{\partial\hat{x}[k+1]}{\partial\hat{\omega}} &= \frac{\partial\hat{A}_{rek}}{\partial\hat{\omega}} \cdot \hat{x}[k] \\ &\quad + \hat{A}_{rek} \cdot \frac{\partial\hat{x}[k]}{\partial\hat{\omega}} \\ &\quad + \frac{\partial\hat{b}}{\partial\hat{\omega}} \cdot u[k] \\ &\quad + \frac{\partial\hat{K}_{NL}}{\partial\hat{\omega}} \cdot \underline{NL}(u[k], \hat{x}[k]) \\ &\quad + \hat{K}_{NL} \cdot \frac{\partial\underline{NL}(u[k], \hat{x}[k])}{\partial\hat{\omega}} \end{aligned}$$

Mit der eingeführten Jacobi-Matrix $\hat{J}_{\hat{x}[k]} = \frac{\partial \hat{x}[k]}{\partial \hat{\omega}}$ und elementaren Umformungen ergibt sich:

$$\begin{aligned}\hat{J}_{\hat{x}[k+1]} &= \hat{A}_{rek} \cdot \hat{J}_{\hat{x}[k]} \\ &+ \frac{\partial \hat{A}_{rek}}{\partial \hat{\omega}} \cdot \hat{x}[k] \\ &+ \frac{\partial \hat{b}}{\partial \hat{\omega}} \cdot u[k] \\ &+ \frac{\partial \hat{K}_{NL}}{\partial \hat{\omega}} \cdot \underline{NL}(u[k], \hat{x}[k]) \\ &+ \hat{K}_{NL} \cdot \frac{\partial \underline{NL}(u[k], \hat{x}[k])}{\partial \hat{\omega}} \#(3-25)\end{aligned}$$

Durch Zusammenfassen kann diese Gleichung in die Form

$$\hat{J}_{\hat{x}[k+1]} = \hat{A}_{rek} \cdot \hat{J}_{\hat{x}[k]} + \hat{F} \quad \text{mit} \quad \hat{F} = \left[\hat{f}_1 \dots \hat{f}_i \dots \hat{f}_p \right] \#(3-26)$$

gebracht werden. Die Spalten der Matrix \hat{F} ergeben sich dabei zu

$$\begin{aligned}\hat{f}_i &= \frac{\partial \hat{A}_{rek}}{\partial \hat{\omega}} \cdot \hat{x}[k] \\ &+ \frac{\partial \hat{b}}{\partial \hat{\omega}} \cdot u[k] \\ &+ \frac{\partial \hat{K}_{NL}}{\partial \hat{\omega}} \cdot \underline{NL}(u[k], \hat{x}[k]) \\ &+ \hat{K}_{NL} \cdot \frac{\partial \underline{NL}(u[k], \hat{x}[k])}{\partial \hat{\omega}} \#(3-27)\end{aligned}$$

Entsprechend der Ausgangsgleichung $\hat{y}[k] = \hat{c}^T \cdot \hat{x}[k]$ werden aus der Jacobi-Matrix $\hat{J}_{\hat{x}[k]}$ die gesuchten partiellen Ableitungen $\nabla \hat{y}[k]$ gewonnen:

$$(\nabla \hat{y}[k])^T = \left[\frac{\partial \hat{y}[k]}{\partial \hat{\omega}_1} \dots \frac{\partial \hat{y}[k]}{\partial \hat{\omega}_p} \right] = \hat{c}^T \cdot \hat{J}_{\hat{x}[k]} \#(3-28)$$

Aus Gleichung (3-25) ist auch ersichtlich, dass die Jacobi-Matrix $\hat{J}_{\hat{x}[k+1]}$ aus der aktuellen Jacobi-Matrix $\hat{J}_{\hat{x}[k]}$ berechnet wird. Entsprechend ist zur Berechnung von $\hat{J}_{\hat{x}[1]}$ die Matrix $\hat{J}_{\hat{x}[0]}$ notwendig. Da für $k \leq 0$ alle Zustände $\hat{x}[k]$ und alle Gewichte $\hat{\omega}$ konstant sind, gilt für die Jacobi-Matrix $\hat{J}_{\hat{x}[0]} = 0$. Die Jacobimatrix wird also nicht durch explizites Differenzieren entsprechend Gleichung (3-23), sondern durch eine einfache Rekursion bestimmt [S10].

Eine ausführliche Herleitung der partiellen Ableitungen würde den Rahmen dieser Studie sprengen und kann in [CH03] nachgelesen werden. Die partiellen Ableitungen für das GRNN und RBF-Netz sind aus Gründen der Vollständigkeit in Tabelle 3.1 zusammengefasst.

| Netz | partielle Ableitung |
|------|--|
| RBF | Für $\hat{w}_i \neq \hat{\Theta}_l$ mit $1 \leq i \leq p$ und $1 \leq l \leq r$ $\frac{\partial \hat{y}_{RBF}}{\partial \hat{w}_i} = -\frac{\partial \hat{u}}{\partial \hat{w}_i} \cdot \sum_{j=1}^r \hat{\Theta}_j \cdot \mathcal{A}_j(\hat{u}) \cdot \frac{\hat{u} - \xi_j}{\sigma_{norm}^2 \cdot \Delta \xi^2}$ |
| | Für $\hat{w}_i = \hat{\Theta}_l$ mit $1 \leq i \leq p$ und $1 \leq l \leq r$ $\frac{\partial \hat{y}_{RBF}}{\partial \hat{w}_i} = \mathcal{A}_l(\hat{u}) - \frac{\partial \hat{u}}{\partial \hat{w}_i} \cdot \sum_{j=1}^r \hat{\Theta}_j \cdot \mathcal{A}_j(\hat{u}) \cdot \frac{\hat{u} - \xi_j}{\sigma_{norm}^2 \cdot \Delta \xi^2}$ |
| GRNN | Für $\hat{w}_i \neq \hat{\Theta}_l$ mit $1 \leq i \leq p$ und $1 \leq l \leq r$ $\frac{\partial \hat{y}_{GRNN}}{\partial \hat{w}_i} = \frac{\partial \hat{u}}{\partial \hat{w}_i} \cdot \sum_{j=1}^r \mathcal{A}_j(\hat{u}) \cdot \frac{\hat{u} - \xi_j}{\sigma_{norm}^2 \cdot \Delta \xi^2} \cdot (\hat{y}_{GRNN} - \hat{\Theta}_j)$ |
| | Für $\hat{w}_i = \hat{\Theta}_l$ mit $1 \leq i \leq p$ und $1 \leq l \leq r$ $\frac{\partial \hat{y}_{GRNN}}{\partial \hat{w}_i} = \mathcal{A}_l(\hat{u}) + \frac{\partial \hat{u}}{\partial \hat{w}_i} \cdot \sum_{j=1}^r \mathcal{A}_j(\hat{u}) \cdot \frac{\hat{u} - \xi_j}{\sigma_{norm}^2 \cdot \Delta \xi^2} \cdot (\hat{y}_{GRNN} - \hat{\Theta}_j)$ |

Tabelle 3.1: Zusammenstellung der partiellen Ableitungen der statischen Neuronalen Netze [S10]

Bemerkenswert ist in der Tabelle den Term $\frac{\partial \hat{u}}{\partial \hat{w}_i}$ gleich die Jacobi-Matrix $\hat{J}_{\hat{x}[k]}$. Das heißt in Gleichung (3-27), dass die Ableitung von Nichtlinearität nach Gewicht auch abhängig von der Jacobi-Matrix $\hat{J}_{\hat{x}[k]}$ im letzten Abtastschritt.

4 Identifikation einer elektrischen Maschine

Im Folgenden wird anhand des bereits eingeführten Beispiels der Mechanik einer elektrischen Maschine anschaulich erläutert, wie das rekurrente Netz zum Beobachter erweitert wird und wie sich diese Erweiterung auf die Parameteradaption auswirkt.

4.1 Anwendung der Beobachterstruktur

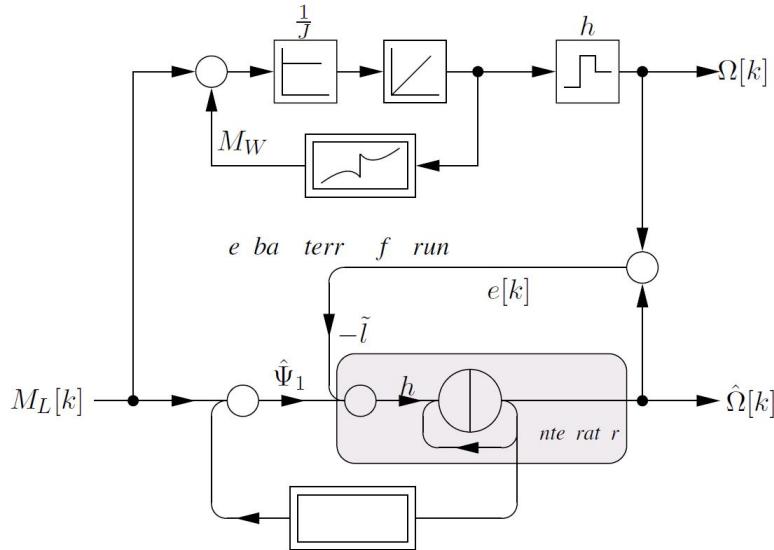


Abb. 4.1: Neuronaler Beobachter der Mechanik einer elektrischen Maschine [S10]

In Abbildung 4.1 ist das rekurrente Netz aus Abbildung 3.2 inklusive der Strecke und der Beobachterrückführung dargestellt. Der Eingriff der Beobachterrückführung erfolgt am Eingang der Integrator-Approximation, daher wird der Beobachterkoeffizient \tilde{l} des kontinuierlichen Beobachters eingesetzt. Bei dieser Darstellung der Beobachterrückführung kann die vorwärts Rechteckapproximation durch eine exaktere Methode ersetzt werden, indem der Block für den Integrator im rekurrenten Netz entsprechend ersetzt wird.

Aus Abbildung 4.1 kann die Differenzengleichung

$$\hat{\Omega}[k+1] = (1 + h \cdot \tilde{l}) \cdot \hat{\Omega}[k] - h \cdot \tilde{l} \cdot \Omega[k] + h \cdot \hat{\Psi}_1 \cdot M_L[k] - h \cdot \hat{\Psi}_1 \cdot \hat{y}_{GRNN}(\hat{\Omega}[k]) \#(4 - 1)$$

abgelesen werden. Durch einen Vergleich der Differenzengleichung mit der diskreten Zustandsbeschreibung des neuronalen Beobachters nach Gleichung (3-20) ergeben sich die Elemente der Zustandsbeschreibung, wie in Tabelle 4.1 zusammengefasst.

| dis. Zustandsbeschr. | u | \hat{x} | $\hat{\mathbf{A}}_{rek}$ | \hat{b} | \hat{l} | $\hat{\mathbf{K}}_{NL}$ | $\widehat{\mathcal{NL}}(u, \underline{x})$ | \hat{c} | \hat{y} |
|----------------------|-------|----------------|--------------------------|------------------------|---------------------|-------------------------|--|-----------|----------------|
| Differenzengleichung | M_L | $\hat{\Omega}$ | $1 + h \cdot \tilde{l}$ | $h \cdot \hat{\Psi}_1$ | $h \cdot \tilde{l}$ | $-h \cdot \hat{\Psi}_1$ | $\hat{y}_{GRNN}(\hat{\Omega})$ | 1 | $\hat{\Omega}$ |

Tabelle 4.1: Elemente der diskrete Zustandsbeschreibung [S10]

4.2 Durchführung der Identifikation

Mit den Elementen der diskreten Zustandsdarstellung aus Tabelle 4.1 und dem Parametervektor

$$\hat{\omega} = [\hat{\psi}_1 \ \hat{\theta}_1 \ \dots \ \hat{\theta}_r]^T$$

können die partiellen Ableitungen $\nabla \hat{\Omega}$ unter Berücksichtigung der Beobachterrückführung nach Gleichungen (3-26) und (3-28) bestimmt werden. Die Ergebnisse dieser Berechnungen sind

$$\hat{J}_{\hat{x}[k+1]} = \hat{A}_{rek} \cdot \hat{J}_{\hat{x}[k]} + \hat{F} = \hat{J}_{\hat{\Omega}[k+1]} = (1 + h \cdot \tilde{l}) \cdot \hat{J}_{\hat{\Omega}[k]} + \hat{F} \# (4 - 2)$$

mit

$$\begin{aligned} \hat{f}_1 &= \frac{\partial \hat{A}_{rek}}{\partial \hat{\omega}_1} \cdot \hat{x}[k] \\ &+ \frac{\partial \hat{b}}{\partial \hat{\omega}_1} \cdot u[k] \\ &+ \frac{\partial \hat{K}_{NL}}{\partial \hat{\omega}_1} \cdot \widehat{\mathcal{NL}}(u[k], \hat{x}[k]) + \hat{K}_{NL} \cdot \frac{\partial \widehat{\mathcal{NL}}(u[k], \hat{x}[k])}{\partial \hat{\omega}_1} \\ &= \frac{\partial (1 + h \cdot \tilde{l})}{\partial \hat{\omega}_1} \cdot \hat{\Omega}[k] \\ &+ \frac{\partial (h \cdot \hat{\varphi}_1)}{\partial \hat{\omega}_1} \cdot M_L[k] \\ &+ \frac{\partial (-h \cdot \hat{\varphi}_1)}{\partial \hat{\omega}_1} \cdot \hat{y}_{GRNN}(\hat{\Omega}[k]) + (-h \cdot \hat{\varphi}_1) \cdot \frac{\partial \hat{y}_{GRNN}(\hat{\Omega}[k])}{\partial \hat{\omega}_1} \\ &= 0 \cdot \hat{\Omega}[k] \\ &+ h \cdot M_L[k] \\ &- h \cdot \hat{y}_{GRNN}(\hat{\Omega}[k]) - h \cdot \hat{\varphi}_1 \cdot \frac{\partial \hat{y}_{GRNN}(\hat{\Omega}[k])}{\partial \hat{\omega}_1} \\ &= h \cdot (M_L[k] - \hat{y}_{GRNN}(\hat{\Omega}[k])) - h \cdot \hat{\varphi}_1 \cdot \frac{\partial \hat{y}_{GRNN}(\hat{\Omega}[k])}{\partial \hat{\omega}_1} \end{aligned}$$

und für $2 \leq i \leq p$

$$\begin{aligned}
\widehat{f}_1 &= \frac{\partial \widehat{A}_{rek}}{\partial \widehat{\omega}_1} \cdot \underline{\hat{x}}[k] \\
&+ \frac{\partial \widehat{b}}{\partial \widehat{\omega}_1} \cdot u[k] \\
&+ \frac{\partial \widehat{K}_{NL}}{\partial \widehat{\omega}_1} \cdot \underline{\widehat{NL}}(u[k], \underline{\hat{x}}[k]) + \widehat{K}_{NL} \cdot \frac{\partial \underline{\widehat{NL}}(u[k], \underline{\hat{x}}[k])}{\partial \widehat{\omega}_1} \\
&= \frac{\partial(1 + h \cdot \tilde{l})}{\partial \widehat{\omega}_1} \cdot \widehat{\Omega}[k] \\
&+ \frac{\partial(h \cdot \widehat{\varphi}_1)}{\partial \widehat{\omega}_1} \cdot M_L[k] \\
&+ \frac{\partial(-h \cdot \widehat{\varphi}_1)}{\partial \widehat{\omega}_1} \cdot \widehat{y}_{GRNN}(\widehat{\Omega}[k]) + (-h \cdot \widehat{\varphi}_1) \cdot \frac{\partial \widehat{y}_{GRNN}(\widehat{\Omega}[k])}{\partial \widehat{\omega}_1} \\
&= 0 \cdot \widehat{\Omega}[k] \\
&+ 0 \cdot M_L[k] \\
&+ 0 \cdot \widehat{y}_{GRNN}(\widehat{\Omega}[k]) - h \cdot \widehat{\varphi}_1 \cdot \frac{\partial \widehat{y}_{GRNN}(\widehat{\Omega}[k])}{\partial \widehat{\omega}_1} \\
&= -h \cdot \widehat{\varphi}_1 \cdot \frac{\partial \widehat{y}_{GRNN}(\widehat{\Omega}[k])}{\partial \widehat{\omega}_1}
\end{aligned}$$

Das graphische Verhältnis dieser Berechnungen ist in Abbildung 4.2 enthalten. Diese zeigt, wie die Strecke, das rekurrente Netz, die Berechnung der partiellen Ableitungen und das Lerngesetz kombiniert werden, um die Identifikation der Maschinenparameter (Massenträgheitsmoment und Reibungskennlinie) durchzuführen.

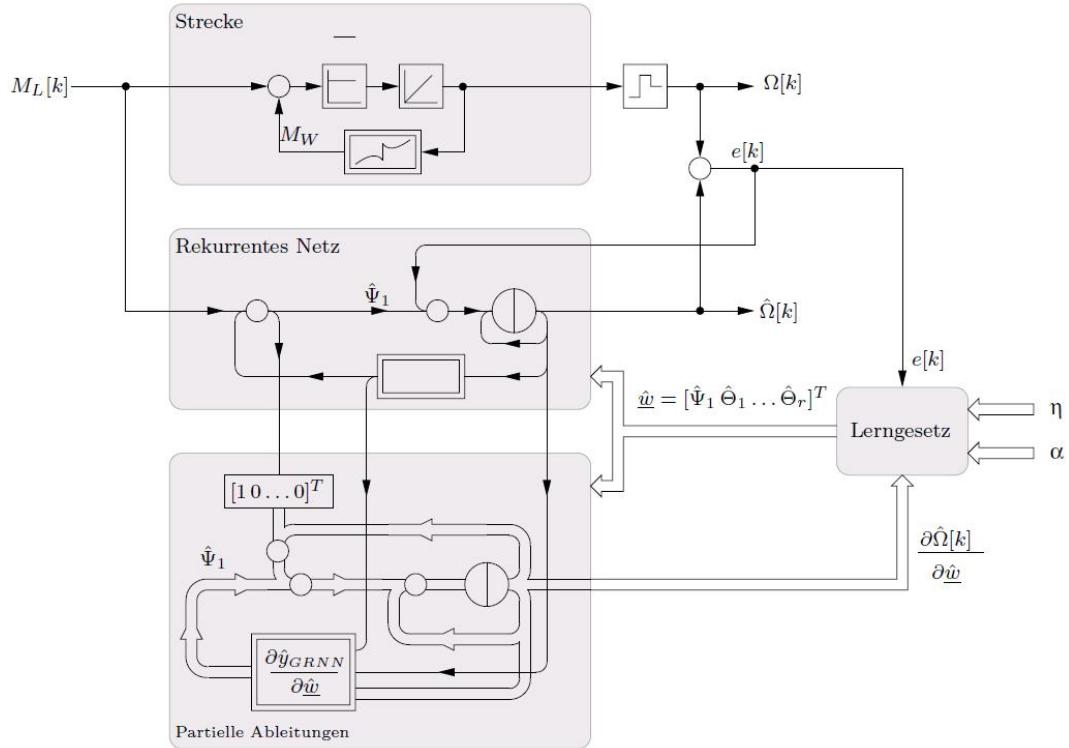


Abb. 4.2: Rekurrentes Netz zur Identifikation der Mechanik einer elektrischen Maschine als Beobachter [S10]

4.3 Simulative Parameteridentifikation

Die Anregung M_L der Strecke erfolgt mit der Zwei-Punkt-Regelung. Überschreitet die Winkelgeschwindigkeit $\Omega[k]$ +20 rad/s, wird das Luftspaltdrehmoment M_L mit -15 Nm vorgegeben. Werden -20 rad/s unterschritten, wird das Luftspaltdrehmoment auf +15 Nm gesetzt. Damit erfolgt die Anregung der Strecke mit einem rechteckförmigen Drehmomentverlauf.

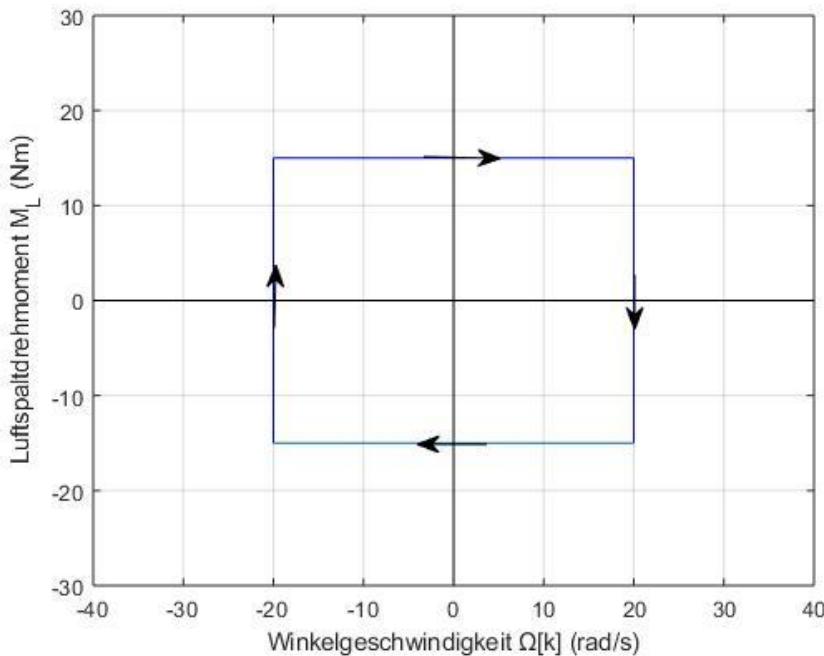


Abb. 4.3: Rechteckförmiger Drehmomentverlauf von Zwei-Punkt-Regelung

Das Massenträgheitsmoment wird auf $J_1 = 0.166 \text{ kg/m}^2$ gesetzt. Im rekurrenten Netz wird der zugehörige Parameter mit $\hat{\psi}_1[0] = \frac{1}{0.2 \text{ kg m}^2} = 5 \frac{1}{\text{kg m}^2}$ initialisiert. Die Abtastzeit wird in dieser Anwendung mit $h = 1 \text{ ms}$ festgelegt.

Hier wird bei dieser Anwendung neben dem Massenträgheitsmoment auch die Reibungskennlinie über die Approximation durch ein GRNN identifiziert. Für dieses werden zusätzlich der Eingangsbereich $-20 \frac{\text{rad}}{\text{s}} \leq \hat{u}_{GRNN} \leq +20 \frac{\text{rad}}{\text{s}}$, der Glättungsfaktor $\sigma_{1,norm} = 1.3$ und die Anzahl der Stützstellen $r_1 = 36$ festgelegt.

Die Beobachterrückführung wird mit $\tilde{l} = 30$ festgelegt. Der lineare Systemanteil hat damit einen reellen Eigenwert bei $\lambda = -15$, während der lineare Anteil der zu identifizierenden Strecke einen Eigenwert bei $\lambda = 0$ und damit global integrierendes Verhalten aufweist.

Durch die Berücksichtigung der Reibung und der Erweiterung zum Beobachter müssen die Lernparameter manuell festgelegt werden. Für den linearen Parameter wird $\eta_1 = 1 \cdot 10^{-3}$ und $\alpha_1 = 0.95$ bestimmt. Für die nichtlinearen Parameter, d. h. die Stützstellen der Reibungskennlinie, wird $\eta_{2...31} = 15 \cdot 10^{-2}$ und $\alpha_{2...31} = 0.95$ gewählt.

Durch Nutzen von Matlab/Simulink wird das vorstrukturierte rekursive Netz, wie in Abbildung 4.4 gezeigt, realisiert.

Strecke

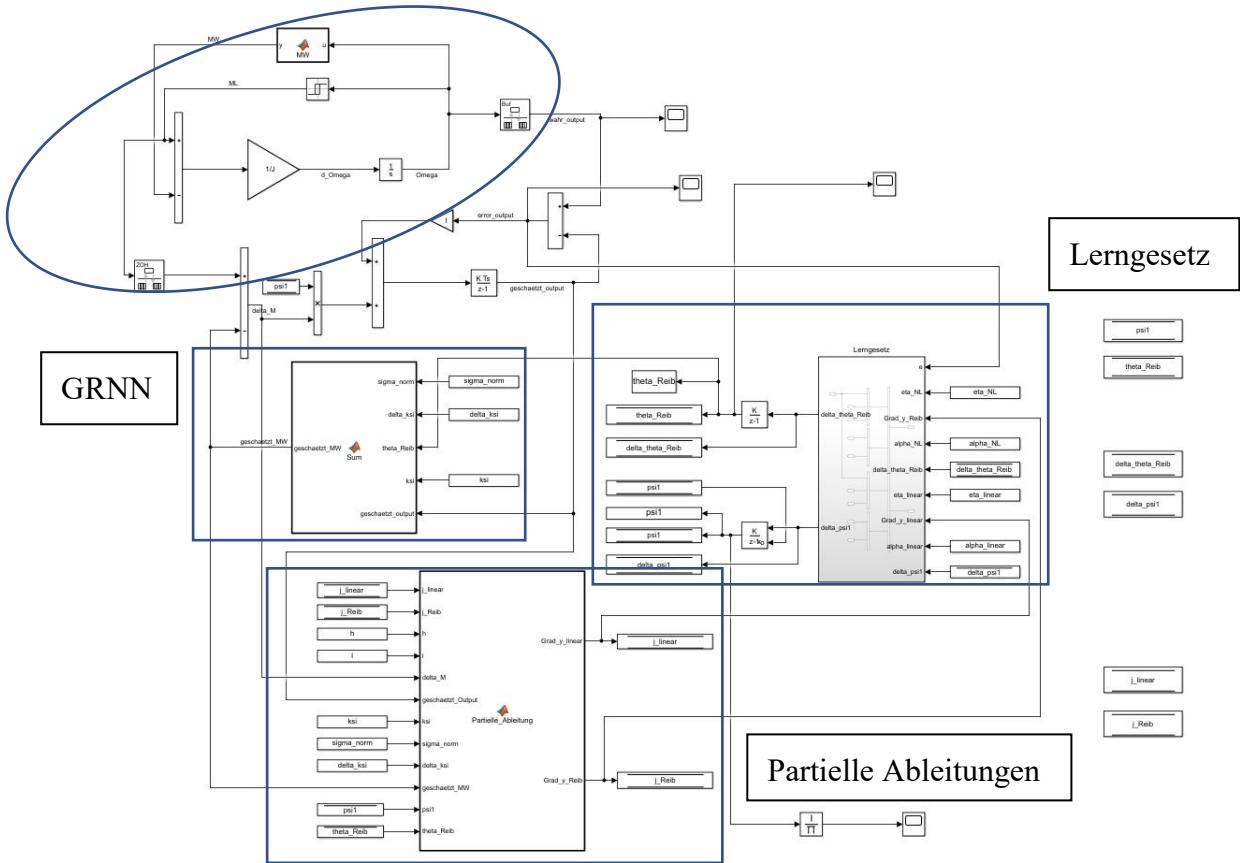


Abb. 4.4: Vorstrukturiertes rekurrentes Netz zur Identifikation der Mechanik einer elektrischen Maschine als Beobachter in Matlab/Simulink

Darin ist die obere rechte Strecke (siehe auch Abbildung 4.5) die originale Regelungsstrecke von der elektrischen Maschine. Die Funktion von diesem Teil ist zur Versorgung von Ein- und Ausgangswerte, mit den die Parameter durch das vorstrukturierte rekurrente Netz erlernt werden können.

Damit die Ein- und Ausgangswerte richtig vom rekurrenten Netz akzeptiert werden, sind die Analog-Digital-Umsetzer an den Ein- und Ausgangsschnittstelle der Strecke erforderlich. Dafür wird der Simulink-Operator („Rate Transition“) eingesetzt. Darin kann die Abtastzeit als 1ms eingestellt werden. Die Abtastzeit anderer diskreter Operatoren, wie dem diskreten Integrator, kann dann einfach als -1 (-1 für vererbt, nämlich 1ms) bestimmt werden.

Die Matlab-Funktion „MW“ ist hierbei die Nichtlinearität im dynamischen System. Darin wird der Operator „Relay“ eingesetzt, der die Funktion der Zwei-Punkt-Regelung realisiert und unendlichen Inputs, M_L , an das System liefert.

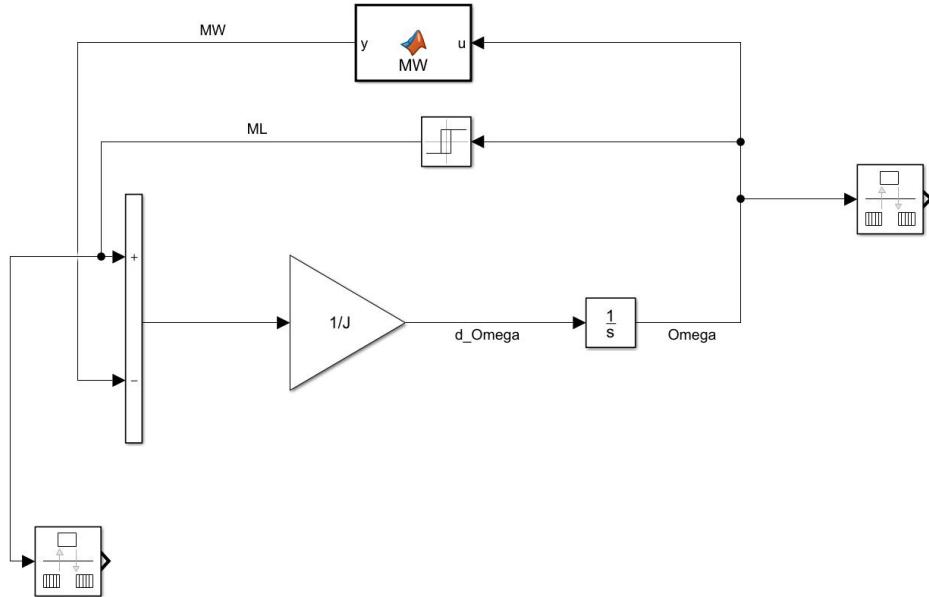


Abb. 4.5: reale Regelungsstrecke mit Analog-Digital-Umsetzer

Bei der realen Regelungsstrecke handelt es sich um das diskrete rekurrente Netz. Es übernimmt die diskreten Ein- und Ausgangswerte in jeder Iteration, rechnet durch einen Add-Operator den Ausgangsfehler aus, und überträgt diesen an dem weiteren Schritt, das Lerngesetz. Wie in dieser Arbeit in Abschnitt 4.1 erklärt, wird hierbei eine Beobachterstruktur verwendet. Mit dem Addieren des Produktes vom Ausgangsfehler und der Beobachterrückführung kann der Einstieg des Ausgangsfehler eingeschränkt werden. Mit dieser Erweiterung ist es möglich, eine stabile Identifikation einer global integrierten Strecke aufzubauen.

Als Integrator wird der Simulink-Operator „Discrete-Time Integrator“ eingesetzt. Durch Wählen von „Integration: Forward Euler“ kann die Integrationsmaßnahme mit der Gleichung

$$y[n] = y[n - 1] + K \times (t[n] - t[n - 1]) \times u(n - 1) \quad (4-3)$$

benutzt werden (wie Gleichung (3-1)). In jeder Iteration wird der Zustandsparameter, d.h. die Winkelgeschwindigkeit $\Omega[k]$, automatisch mit dessen Wert in letzter Iteration addiert, indem der Programmierungsprozess vereinfacht wird.

Weil der lineare Parameter Ψ_1 (in der Abbildung „psi1“) keine Konstante ist und sich nach jeder Iteration erneuert, soll hier der Simulink-Operator „Data Store Memory“ mit entsprechendem Eingang („Data Store Read“) und Ausgang („Data Store Write“) eingesetzt werden. Durch „Data Store Write“ kann er die Werte von Parameter verändern und durch „Data Store Read“ die veränderten Werte in nächstem Schritt wieder in das System importieren. Alle Parameter, die in dieser Situation liegen, soll durch „Data Store Memory“ kontrolliert werden.

Außerdem verwendet die Matlab-Funktion „Sum“ hier die Gleichung (2-6), indem es die erneuerten geschätzten Ausgangswerte des rekurrenten Netzes und die erneuerten Gewichte des Lerngesetzes annimmt. Das GRNN wird an dieser Stelle als Funktionsapproximator genutzt.

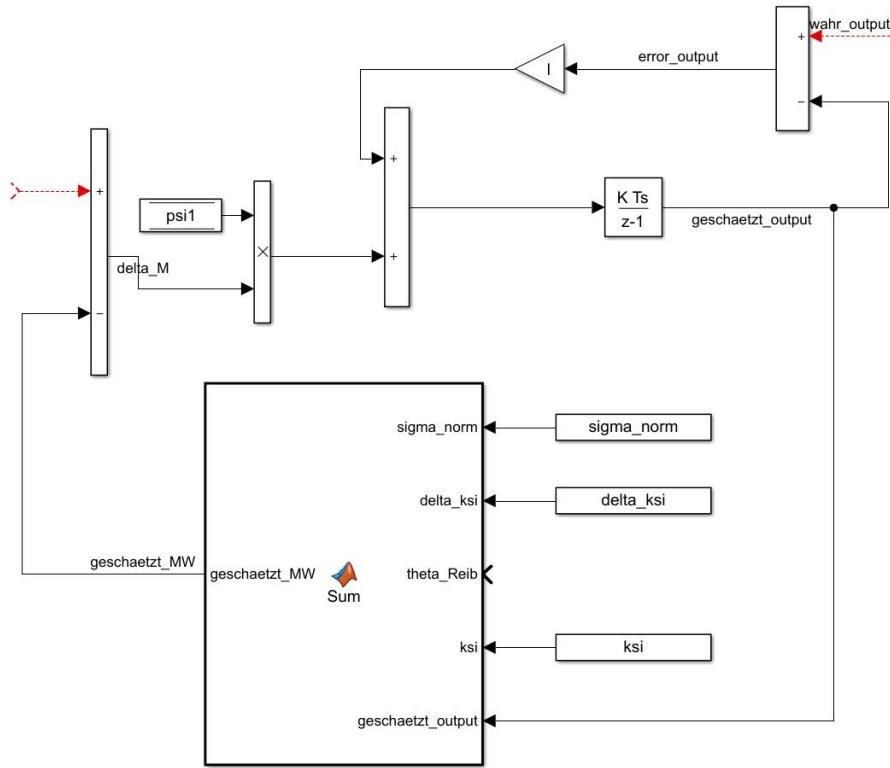


Abb. 4.6: Rekurrentes Netz mit GRNN Funktionsapproximator

Nun muss noch das Problem der Erneuerung der Gewichte gelöst werden. Anhand der Gleichung (3-12) wird dies durch das Subsystem „Lerngesetz“ erreicht.

Um die Struktur besser zu verstehen, werden der lineare Parameter „psi1“ und die nichtlinearen Parameter (die Gewichte „theta_Reib“) in dieser Arbeit separat gelernt. In der Abbildung steht „NL“ für Nichtlinearität und „linear“ für linearen Parameter.

Die Gleichung (3-12) ähnelt Gleichung (3-1), mit der Ausnahme, dass darin die Abtastzeit fehlt. Deshalb wird zusätzlich zur Akkumulation der Gewichte und linearen Parameter noch der Simulink-Operator „Discrete-Time Integrator“ eingesetzt. Nur bei der Auswahl der Integrationsmaßnahme wird „Accumulation: Forward Euler“ eingestellt. Die Gleichung ist

$$y[n + 1] = y[n] + K \times T \times u[n] \#(4 - 4)$$

Im Vergleich zur Gleichung (4-3) ist Gleichung (4-4) unabhängig von der Abtastzeit. Bei der Einstellung des „Discrete-Time Integrator“ von linearem Parameter soll zusätzlich die Anfangsbedingung importiert werden, weil der Anfangswert von „psi“ nicht null beträgt. Diese Funktion kann durch die Auswahl von „external“ in

,Initial condition source‘ und die Verbindung der auftretenden Schnittstelle mit ,Data Store Read‘ von ,psi1‘ realisiert werden.

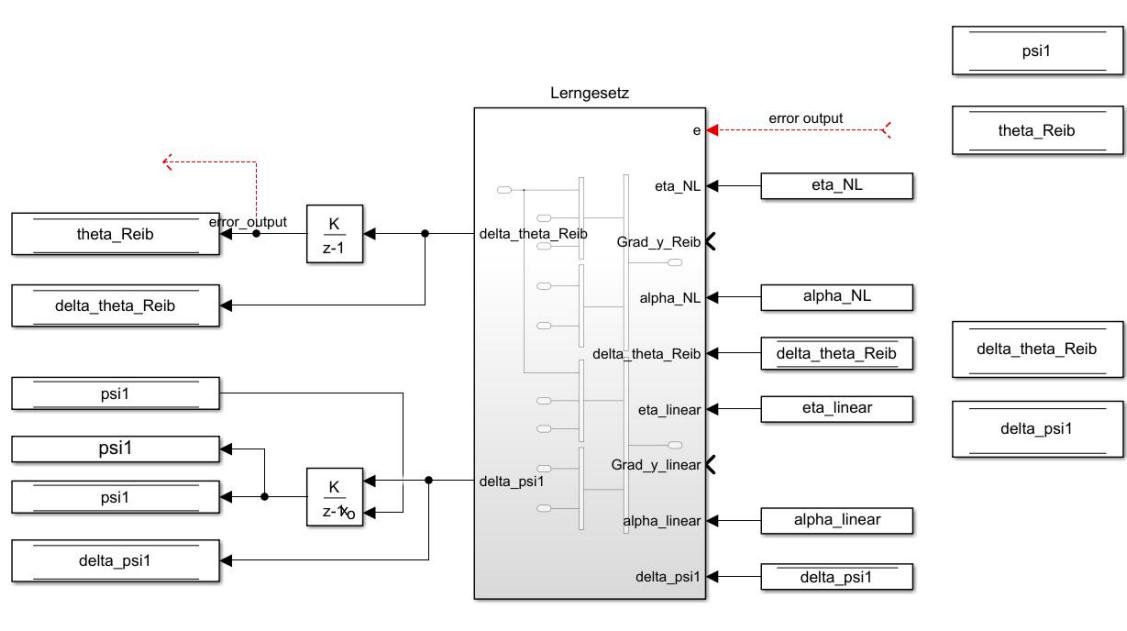


Abb 4.7: Lerngesetz

Am Ende gilt es das letzte Problem, die partiellen Ableitungen, zu lösen. Auf Basis von Tabelle 3.1 und Abschnitt 4.2 kann dies durch ein Matlab-Skript erreicht werden, dessen konkrete Matlab-Skript kann aus Simulink-Dokumenten entnommen werden.

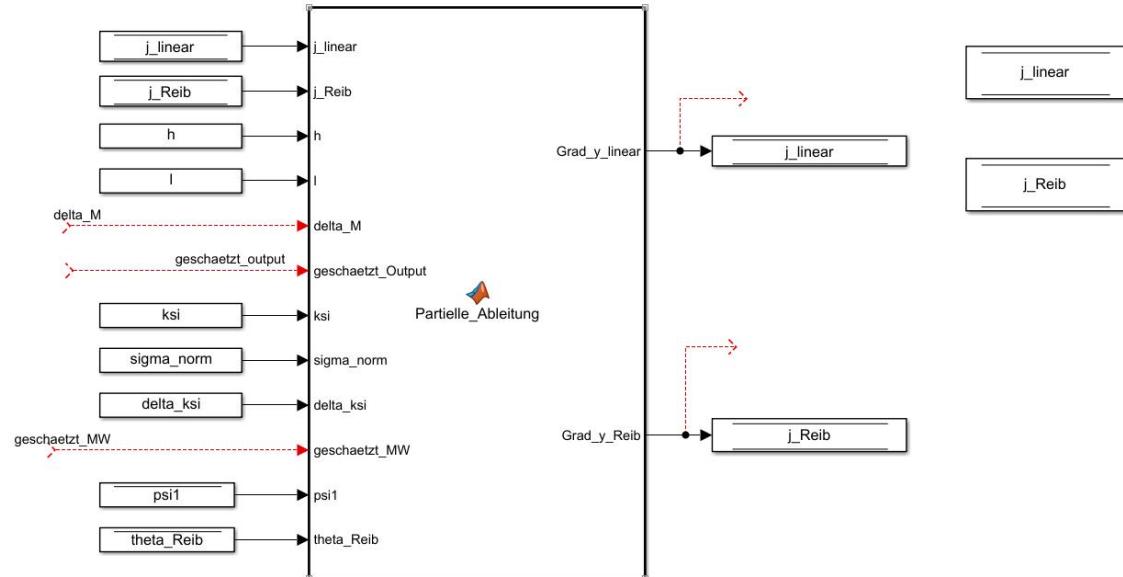


Abb 4.8: Partielle Ableitungen

4.4 Ergebnisse und Beurteilung

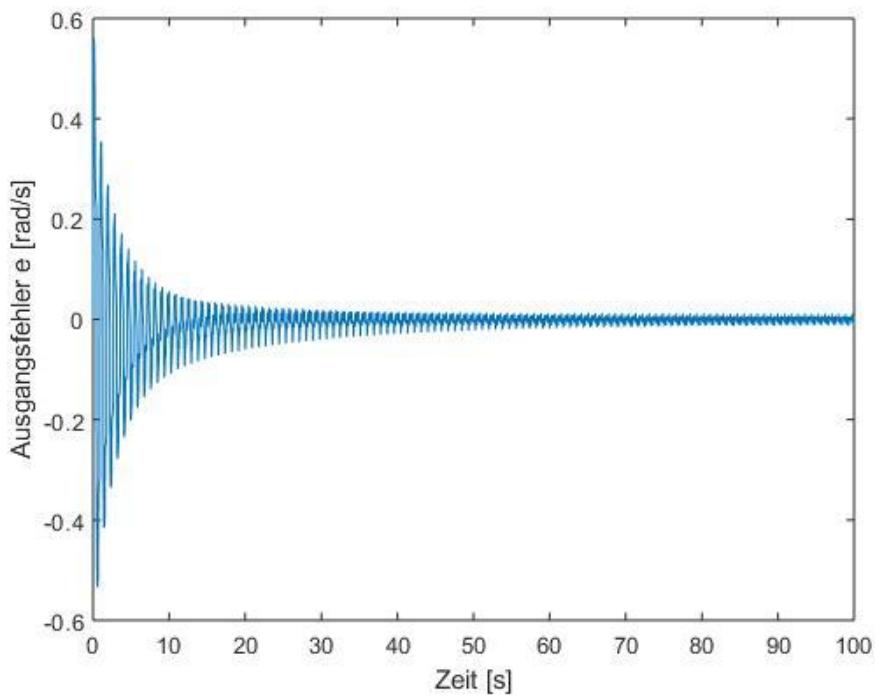


Abb. 4.9: Ausgangsfehlerverlauf während der Identifikation

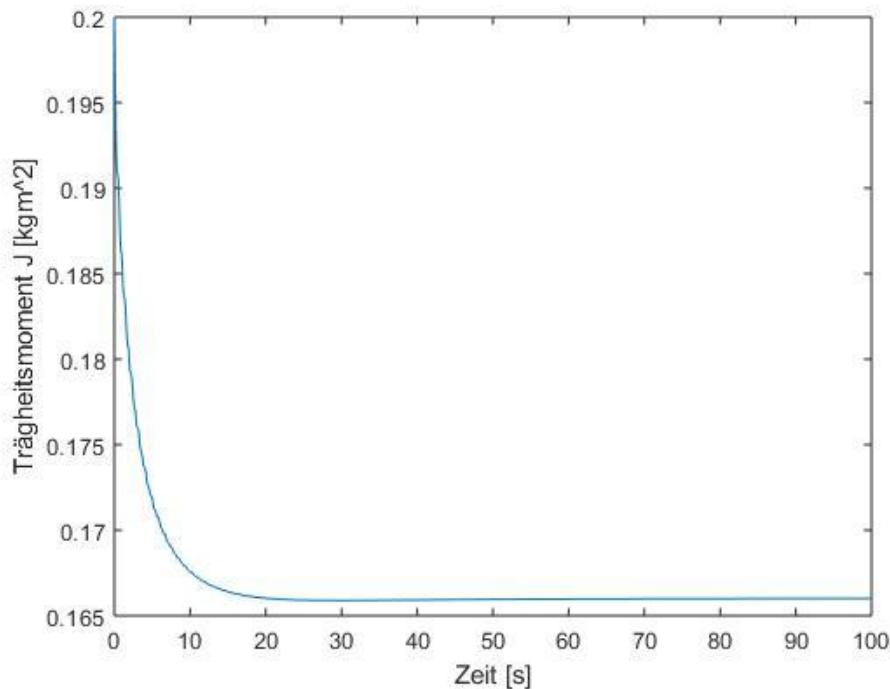


Abb. 4.10: Identifikationsverlauf der Massenträgheit \hat{J}

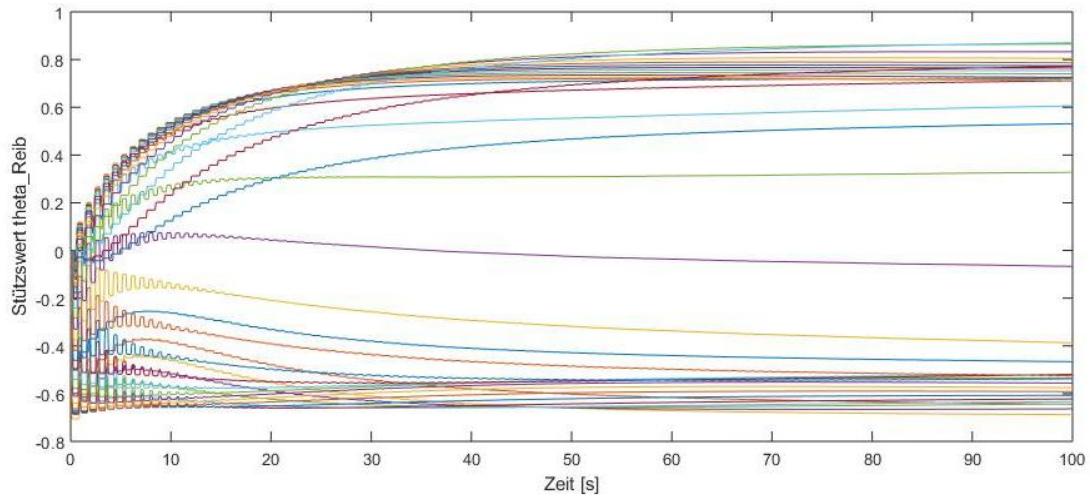


Abb. 4.11: Identifikationsverlauf der Gewichte $\hat{\theta}_{Reib}$ des Funktionsapproximators

Abbildung 4.9 zeigt den Verlauf des Ausgangsfehler während der Identifikation. Nach 30 Sekunden ist der Fehler gegenüber dem Maximalwert deutlich kleiner und nach 50 Sekunden fast zu null geworden.

Der Verlauf des Massenträgheitsmoments während der Identifikation ist in Abbildung 4.10 dargestellt. Der Endwert $J = 0.166 \text{ kg} \cdot \text{m}^2$ ist nach 20 Sekunden erreicht und schwingungsfrei.

Der Verlauf der Gewichte des GRNN ist in Abbildung 4.11 gezeigt. Die Anfangswerte von Gewichten sind 0. Je nach Stützwerten sind die Gewichte nach 20-40 Sekunden stabil. Die meisten Endwerte von Gewichten liegen im Bereich von 0.6-0.8 und um -0.6.

Die zu den Werten nach 100 Sekunden gehörende Reibungskennlinie ist in Abbildung 4.12 dargestellt. Im Bereich der Haftribbung ($\Omega_1 = 0$) und an der Grenze des Eingangsbereiches ($\Omega_1 = -20$ oder 20) weicht die identifizierte Reibungskennlinie leicht von der vorgegebenen Kennlinie ab. Der größte Grund dafür ist, dass der Ausgangsfehler nach 30 Sekunden gegenüber dem Ausgangsfehler bei Identifikationsbeginn klein ist, und nach Gleichung (3-12) hängen die Gewichtsänderungen im rekurrenten Netz vom Ausgangsfehler e ab. Die Gewichtsänderungen sinken mit dem Ausgangsfehler ebenfalls. Außerdem gibt es noch andere Gründe. So ist z. B. das GRNN für Unstetigkeit, d. h. die Haftribbung, nicht praktisch und kann es nicht perfekt approximieren. Um dieses Problem zu lösen, können bei der Identifikation der Reibungskennlinien zwei GRNN eingesetzt werden [S10].

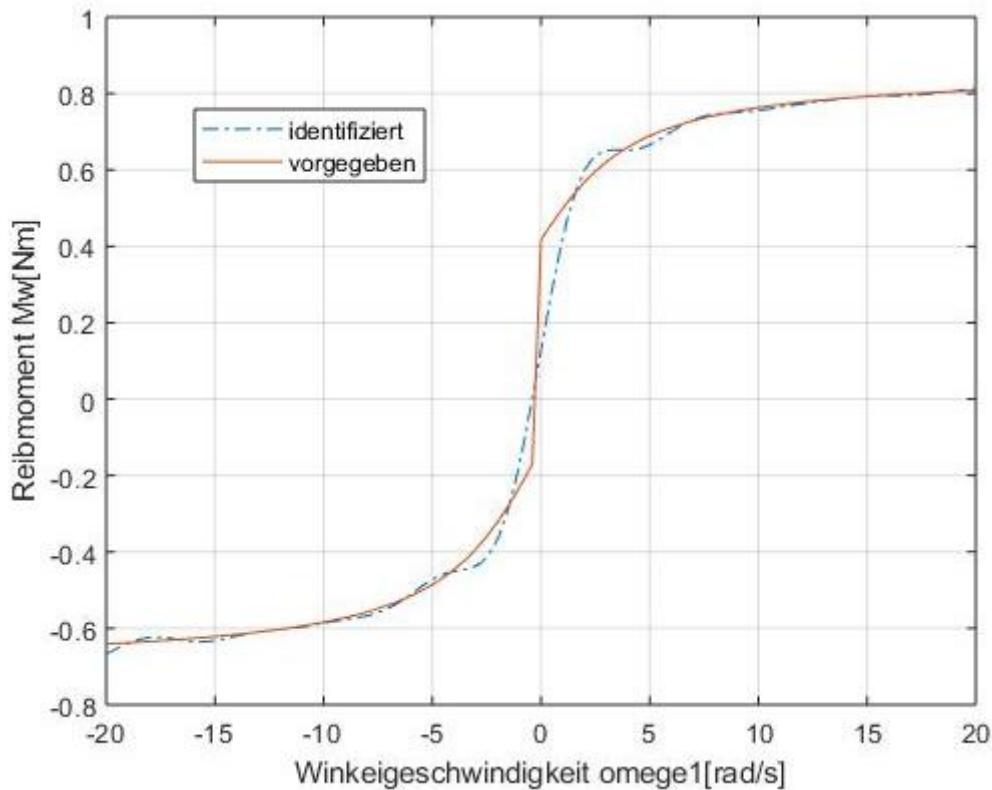
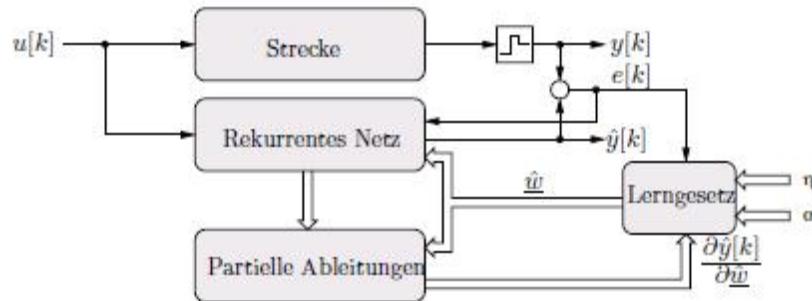


Abb. 4.12: Identifikationsergebnis der Reibungskennlinie $M_R(\Omega)$

4.5 Kurzzusammenfassung

Ausgehend von einer bekannten Systemstruktur wird entsprechend Abbildung 4.13, diese Struktur in ein rekurrentes Netz übertragen. Die statischen Nichtlinearitäten werden dabei durch statische Funktionsapproximatoren im rekurrenten Netz ersetzt. Die zu identifizierenden Parameter sind dort als Gewichte enthalten, werden durch das Lerngesetz erlernt und im Parametervektor zusammengefasst.

Abb. 4.13: Kombination von Strecke, Beobachter, Lerngesetz und Berechnung der partiellen Ableitungen zur Parameteradaption [S10]



Das rekurrente Netz wird zu einem Beobachter erweitert. Zur Auswertung des Lerngesetzes aus Abbildung 3.3 werden aus der Zustandsbeschreibung unter Berücksichtigung der Beobachterrückführungen die partiellen Ableitungen $\nabla\hat{y}$ berechnet. Strecke, rekurrentes Netz, Lerngesetz und die Berechnung der partiellen Ableitungen werden, wie in Abbildung 4.13 dargestellt, zur Adaption des Parametervektors kombiniert.

Vorteilhaft bei der vorgestellten Identifikationsmethode ist die physikalische Interpretierbarkeit der Identifikationsergebnisse. Die Ergebnisse können damit auf ihre Plausibilität überprüft werden, insbesondere die linearen Parameter. Außerdem handelt es sich um eine Methode, was Zeit und Arbeit sparen kann, das Problem der Nichtlinearität zu lösen. Ein weiterer Vorteil ist die Echtzeitfähigkeit des eingesetzten Gradientenverfahrens zur Parameteradaption. Die Implementierung des rekurrenten Netzes in einer Beobachterstruktur bewirkt zusätzlich zur Umgehung der Anfangswertproblematik.

Nachteilig wirkt sich die starke Abhängigkeit der Parameterkonvergenzgeschwindigkeit von den manuell zu bestimmenden Lernparametern η und α aus. Weitere Nachteile sind, dass die Startwerte der Parameter (lineare und nichtlineare Parameter) nicht zu ungünstig gewählt werden dürfen und dass aufgrund der Änderung der linearen Parameter auch die Beobachter-Rückführungen \tilde{l} während der Identifikation angepasst werden müssen. Außerdem die Auswahl von dem Parameter im GRNN, wie Eingangsbereich, Glättungsfaktor und die Anzahl der Stützstellen, ebenfalls wesentlich für die Implementierung des Systems. Der zu ungünstige Parameter kann nicht nur zu falschen Ergebnissen, sondern auch zu divergierenden Systemzustände führen.

Zusätzlich ist für eine erfolgreiche Identifikation des Parameters ein genaues und eindeutiges Systemmodell in Form eines Signalflussplans notwendig (Vorwissen). Stimmt die Struktur des Systemmodells nicht mit dem realen System überein, werden die im Signalflussplan fehlenden Teile soweit wie möglich über die vorhandenen Gewichte ausgeglichen.

5 Identifikation eines Duffing-Oszillators

In diesem Kapitel wird die oben gewählte Methodik auf ein komplexeres mechanisches Beispiel, nämlich Duffing-Oszillatoren, angewendet, um herauszufinden, wie das strukturierte rekurrente Netz bei der Lösung eines Problems mit zweiter Ordnung und mehr als einem linearen Parameter geändert werden soll.

5.1 Duffing-Oszillator

Der Duffing-Oszillator ist eines der Prototypsysteme nichtlinearer Dynamik. Es wurde zuerst für das Studium anharmonischer Schwingungen und später für chaotische nichtlineare Dynamik nach frühen Studien des Ingenieurs Georg Duffing [D18] populär. Das System wurde erfolgreich eingesetzt, um eine Vielzahl physikalischer Prozesse wie Versteifungsfedern, Strahlknicken, nichtlineare elektronische Schaltungen, supraleitende parametrische Josephson-Verstärker und Ionisationswellen in Plasmen zu modellieren. Trotz der Einfachheit des Duffing-Oszillators ist das dynamische Verhalten äußerst umfangreich und die Forschung wird noch heute durchgeführt.

Der Duffing-Oszillator wird durch die Differentialgleichung als

$$\ddot{x} + \delta \cdot \dot{x} + \alpha \cdot x + \beta \cdot x^3 = r \cdot \cos(\omega_0 \cdot t) \#(5 - 1)$$

einfach beschrieben, wobei die Dämpfungskonstante $\delta \geq 0$ gehorcht. Im Regelungsform kann die Gleichung (5-1) umgeschrieben werden als

$$\ddot{x} = -\delta \cdot \dot{x} - \alpha \cdot x - \beta \cdot x^3 + r \cdot \cos(\omega_0 \cdot t). \#(5 - 2)$$

In diesem Kapitel wird anhand dieser Gleichung ein strukturiertes rekurrentes Netz aufgestellt. Die Dämpfungskonstante δ und die Federkonstante α darin werden als lineare Parameter definiert. $\beta \cdot x^3$ gelten hierbei als eine Nichtlinearität. Sie werden durch einen Gewichtsvektor approximiert und nicht separat diskutiert. Gleicherweise ist der ganze Term $r \cdot \cos(\omega_0 \cdot t)$ der Systemeingang.

Im Unterschied zum Beispiel im Kapitel 4 über die Mechanik einer leerlaufenden elektrischen Maschine beinhaltet der Duffing-Oszillator eine Differentialgleichung mit zweiter Ordnung und zwei Zurückführungen von x und \dot{x} . Aus diesem Grund soll das strukturierte rekurrente Netz geändert werden, um die neue Gleichung anzupassen und durch einen Lernprozess ein akzeptables Ergebnis zu erhalten. Im weiteren Verlauf werden zu Beginn die erforderlichen Gleichungen abgeleitet und dann das strukturierte rekurrente Netz entsprechend erneuert.

5.2 Durchführung der Identifikation

Weil die Differentialgleichung hierbei zweiter Ordnung ist, werden in der Regelungsstrecke zwei Integratoren eingesetzt. Die Zustandsparameter vor und nach

dem Integrator sind nicht in gleicher Größenordnung, deshalb bei der Einführung der Beobachterstruktur soll vom Ausgangsfehler $e = \dot{x} - \hat{\dot{x}}$ zum Punkt vor jedem Integrator eine Zurückführung (hierbei insgesamt zweimal) mit manuell gewählten und angepassten Beobachterkoeffizienten \tilde{l}_1 und \tilde{l}_2 erstellt werden.

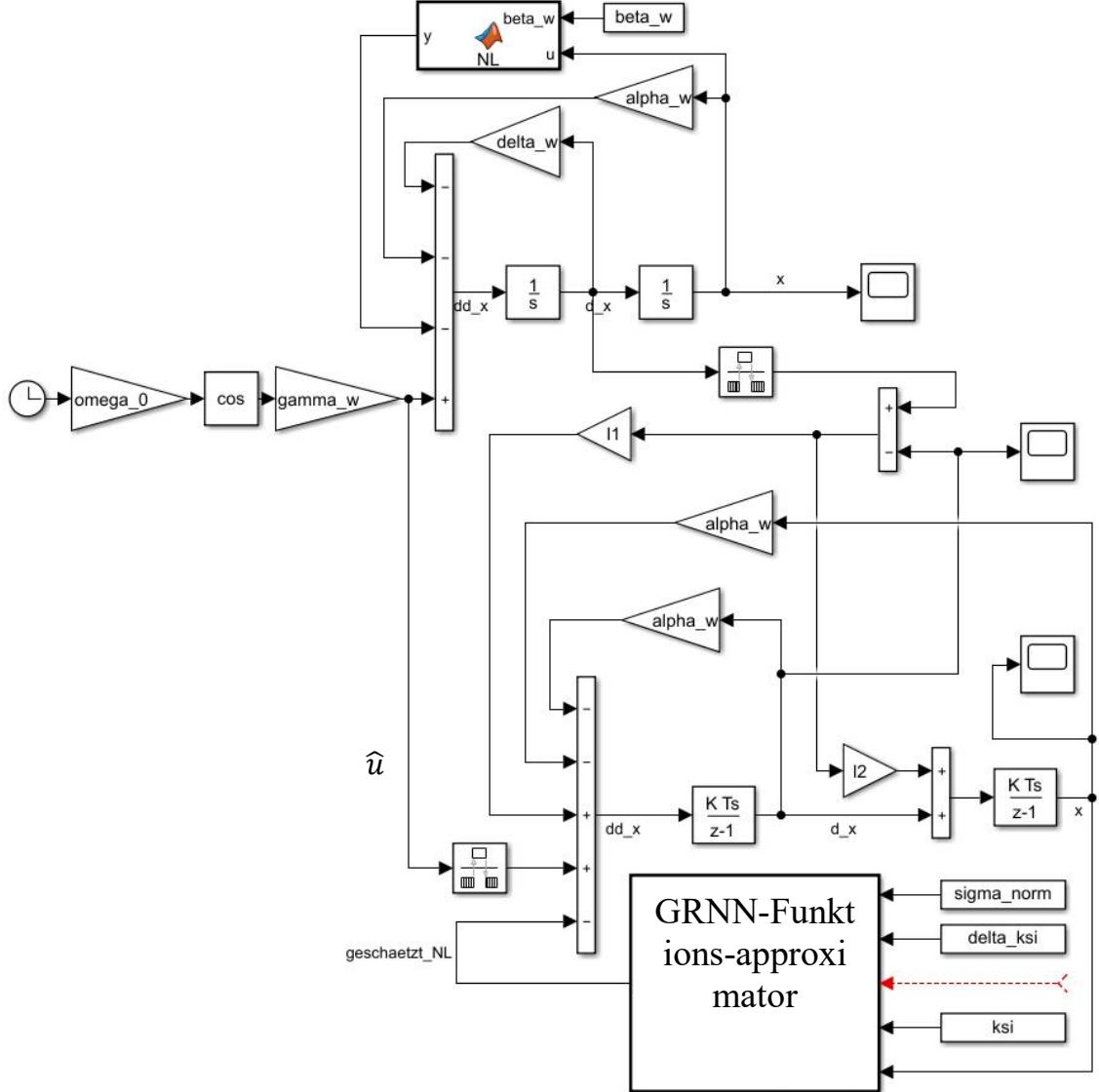


Abb. 5.1: Rekurrentes Netz zur Identifikation der Mechanik eines Duffing-Oszillators als Beobachter anhand der Gleichung (5-2)

Aus Abbildung 5.1 können die Differenzengleichungen

$$\hat{x}[k+1] = \hat{x}[k] + h \cdot \left(\hat{u} - \hat{\varphi}_\delta \cdot \hat{\Omega}[k] - \hat{\varphi}_\alpha \cdot \hat{\Omega}[k] - \hat{y}_{GRNN}(\hat{\Omega}[k]) + l_1 \cdot e[k] \right) \#(5-3)$$

und

$$\hat{x}[k+1] = \hat{x}[k] + h \cdot (\hat{x}[k+1] + l_2 \cdot e[k]) \#(5-4)$$

abgelesen werden. Mit

$$e = \underline{\dot{x}} - \hat{\underline{x}} \#(5 - 5)$$

können die Gleichungen (5-3) und (5-4) als

$$\hat{x}[k + 1] = (1 - h \cdot \hat{\varphi}_\delta - h \cdot l_1) \cdot \hat{x}[k]$$

$$+ h \cdot \hat{u}$$

$$- h \cdot \hat{y}_{GRNN}(\hat{x}[k])$$

$$- h \cdot \hat{\varphi}_\alpha \cdot \hat{x}[k]$$

$$+ h \cdot l_1 \cdot \dot{x}[k] \#(5 - 6)$$

und

$$\hat{x}[k + 1] = - h \cdot l_2 \cdot \hat{x}[k]$$

$$+ h \cdot l_2 \cdot \dot{x}[k]$$

$$+ h \cdot \hat{x}[k + 1]$$

$$+ \hat{x}[k] \#(5 - 7)$$

umgeschrieben werden. Nach der Ableitung beider Seiten von den Gleichungen (5-6) und (5-7) nach dem Gewichtsvektor $\underline{\hat{\omega}}$ werden die Gleichungen

$$\begin{aligned} \frac{\partial \hat{x}[k + 1]}{\partial \underline{\hat{\omega}}} &= \frac{\partial ((1 - h \cdot \hat{\varphi}_\delta - h \cdot l_1) \cdot \hat{x}[k])}{\partial \underline{\hat{\omega}}} \\ &\quad + \frac{\partial (h \cdot \hat{u})}{\partial \underline{\hat{\omega}}} \\ &\quad - \frac{\partial (h \cdot \hat{y}_{GRNN}(\hat{x}[k]))}{\partial \underline{\hat{\omega}}} \\ &\quad - \frac{\partial (h \cdot \hat{\varphi}_\alpha \cdot \hat{x}[k])}{\partial \underline{\hat{\omega}}} \\ &\quad + \frac{\partial (h \cdot l_1 \cdot \dot{x}[k])}{\partial \underline{\hat{\omega}}} \#(5 - 8) \end{aligned}$$

und

$$\frac{\partial \hat{x}[k+1]}{\partial \underline{\omega}} = \frac{\partial(-h \cdot l_2 \cdot \hat{x}[k])}{\partial \underline{\omega}}$$

$$+ \frac{\partial(h \cdot l_2 \cdot \dot{x}[k])}{\partial \underline{\omega}}$$

$$+ \frac{\partial(h \cdot \hat{x}[k+1])}{\partial \underline{\omega}}$$

$$+ \frac{\partial \hat{x}[k]}{\partial \underline{\omega}} \#(5-9)$$

erhalten. Mit $\frac{\partial h}{\partial \underline{\omega}} = 0$, $\frac{\partial \hat{u}}{\partial \underline{\omega}} = 0$, $\frac{\partial l_1}{\partial \underline{\omega}} = 0$, $\frac{\partial l_2}{\partial \underline{\omega}} = 0$ und $\frac{\partial \dot{x}[k]}{\partial \underline{\omega}} = 0$ ergibt sich dann

$$\frac{\partial \hat{x}[k+1]}{\partial \underline{\omega}} = (1 - h \cdot \hat{\varphi}_\delta - h \cdot l_1) \cdot \frac{\partial \hat{x}[k]}{\partial \underline{\omega}}$$

$$-h \cdot \hat{x}[k] \cdot \frac{\partial \hat{\varphi}_\delta}{\partial \underline{\omega}}$$

$$-h \cdot \frac{\partial \hat{y}_{GRNN}(\hat{x}[k])}{\partial \underline{\omega}}$$

$$-h \cdot \hat{\varphi}_\alpha \cdot \frac{\partial \hat{x}[k]}{\partial \underline{\omega}}$$

$$-h \cdot \hat{x}[k] \cdot \frac{\partial \hat{\varphi}_\alpha}{\partial \underline{\omega}} \#(5-10)$$

und

$$\frac{\partial \hat{x}[k+1]}{\partial \underline{\omega}} = -h \cdot l_2 \cdot \frac{\partial \hat{x}[k]}{\partial \underline{\omega}}$$

$$+ h \cdot \frac{\partial \hat{x}[k+1]}{\partial \underline{\omega}}$$

$$+ \frac{\partial \hat{x}[k]}{\partial \underline{\omega}} \cdot \#(5-11)$$

Danach wird die im Abschnitt 3.5 vorgestellte Jacobi-Matrix in den Gleichungen (5-10) und (5-11) eingesetzt. Da es im System zwei Zustandsparameter gibt, werden zwei Jacobi-Matrix erstellt.

$$\hat{J}_{\hat{x}[k]} = \frac{\partial \hat{x}[k]}{\partial \underline{\omega}} \quad \text{und} \quad \hat{J}_{\hat{x}[k]} = \frac{\partial \hat{x}[k]}{\partial \underline{\omega}} \#(5 - 12)$$

Beim Einsetzen der Gleichung (5-12) in den Gleichungen (5-10) und (5-11) ergibt sich

$$\hat{J}_{\hat{x}[k+1]} = (1 - h \cdot \hat{\varphi}_\delta - h \cdot l_1) \cdot \hat{J}_{\hat{x}[k]}$$

$$- h \cdot \hat{x}[k] \cdot \frac{\partial \hat{\varphi}_\delta}{\partial \underline{\omega}}$$

$$- h \cdot \frac{\partial \hat{y}_{GRNN}(\hat{x}[k])}{\partial \underline{\omega}}$$

$$- h \cdot \hat{\varphi}_\alpha \cdot \hat{J}_{\hat{x}[k]}$$

$$- h \cdot \hat{x}[k] \cdot \frac{\partial \hat{\varphi}_\alpha}{\partial \underline{\omega}} \#(5 - 13)$$

und

$$\hat{J}_{\hat{x}[k+1]} = - h \cdot l_2 \cdot \hat{J}_{\hat{x}[k]} + h \cdot \hat{J}_{\hat{x}[k+1]} + \hat{J}_{\hat{x}[k]} \#(5 - 14)$$

Gleich wie im Abschnitt 3.5 gilt für die Jacobimatrix $\hat{J}_{\hat{x}[0]} = 0$ und $\hat{J}_{\hat{x}[0]} = 0$.

Mit den Elementen der diskreten Zustandsdarstellung und dem Parametervektor

$$\underline{\omega} = [\hat{\varphi}_\alpha \ \hat{\varphi}_\delta \ \hat{\theta}_1 \ \dots \ \hat{\theta}_r]^T \#(5 - 15)$$

können die partiellen Ableitungen $\nabla \hat{\Omega}$ unter Berücksichtigung der Beobachterrückführung nach Gleichung (3-26) und (3-27) bestimmt werden. Die Ergebnisse dieser Berechnungen sind wie folgt:

$$\hat{J}_{\hat{x}[k+1]} = (1 - h \cdot \hat{\varphi}_\delta - h \cdot l_1) \cdot \hat{J}_{\hat{x}[k]} + \hat{F} \#(5 - 16)$$

und gleich die Gleichung (5-14)

$$\hat{J}_{\hat{x}[k+1]} = - h \cdot l_2 \cdot \hat{J}_{\hat{x}[k]} + h \cdot \hat{J}_{\hat{x}[k+1]} + \hat{J}_{\hat{x}[k]} \cdot$$

mit $i = 1$ (nämlich $\underline{\omega}_1 = \hat{\varphi}_\alpha$)

$$\underline{f}_1 = - h \cdot \frac{\partial \hat{y}_{GRNN}(\hat{x}[k])}{\partial \underline{\omega}_1}$$

$$- h \cdot \hat{\varphi}_\alpha \cdot \hat{J}_{\hat{x}[k]}$$

$$- h \cdot \hat{x}[k]$$

für $i = 2$ (nämlich $\underline{\hat{\omega}}_2 = \hat{\varphi}_\delta$)

$$\underline{\hat{f}}_2 = -h \cdot \frac{\partial \hat{y}_{GRNN}(\hat{x}[k])}{\partial \underline{\hat{\omega}}_2}$$

$$- h \cdot \hat{\varphi}_\alpha \cdot \hat{J}_{\hat{x}[k]}$$

$$- h \cdot \hat{x}[k]$$

und für $3 \leq i \leq p$ (nämlich $\underline{\hat{\omega}}_i = \hat{\theta}_{i-2}$)

$$\underline{\hat{f}}_i = -h \cdot \frac{\partial \hat{y}_{GRNN}(\hat{x}[k])}{\partial \underline{\hat{\omega}}_i}$$

$$- h \cdot \hat{\varphi}_\alpha \cdot \hat{J}_{\hat{x}[k]}$$

Zur Lösung von $\frac{\partial \hat{y}_{GRNN}(\hat{x}[k])}{\partial \underline{\hat{\omega}}}$ werden die Gleichungen vom GRNN in der Tabelle 3.1 herangezogen.

5.3 Modellbildung des strukturierten rekurrenten Netzes

Die gesamte Struktur des vorstrukturierten rekurrenten Netzes eines Duffing-Oszillators in Matlab/Simulink ist in Abbildung 5.2 dargestellt. Gleich wie Abbildung 4.4 besteht es auch aus vier Hauptteilen: Originalstrecke, Lerngesetz, GRNN und partiellen Ableitungen. In diesem Abschnitt werden hauptsächlich die Unterschiede zwischen den Strukturen beider Beispiele diskutiert.

Der größte Unterschied besteht darin, dass dieses Beispiel eine Differentialgleichung zweiter Ordnung hat und in der identifizierten Strecke entsprechend zweimal Integratoren liegen. Der Ausgangsfehler, der in der Gleichung 3-12 verwendet wird, soll in diesem Beispiel anhand des Zustandsparameters nach dem ersten Integrator (hierbei als $e = \dot{x} - \hat{x}$) definiert werden. Wenn der Ausgangsfehler anhand des Zustandsparameters nach dem zweiten Integrator als $e = x - \hat{x}$, definiert werden würde, kann das System einen falschen, aber auch stabilen Zustand erreichen, indem statt des Ausgangsfehlers e der Zwischenparameter \hat{x} zu null und der Ausgang \hat{x} zu einer Konstante wird. Der Grund dafür ist, dass in diesem Beispiel die linearen Parameter auf die Zurückführung liegen und die Zurückführungswerte von e , \dot{x} , \hat{x} und GRNN und die Anregung u beim Lernprozess miteinander ausgleichen können.

Danach sollen auf die Zurückführungslinien vom Ausgangsfehler zum ersten und zum zweiten Integrator die Beobachterrückführungen unterschiedliche Werte l_1 und l_2 eingestellt werden, weil die Eingangswerte zum ersten und zweiten Integratoren nicht in gleicher Ordnung sind. Außerdem gilt es zu betrachten, dass es zwei

Differentialgleichungen und zwei Jacobi-Matrix gibt. Beide sollen definiert und im rekurrenten Teil („Partielle Ableitung“) nach jeder Iteration sich akkumuliert werden.

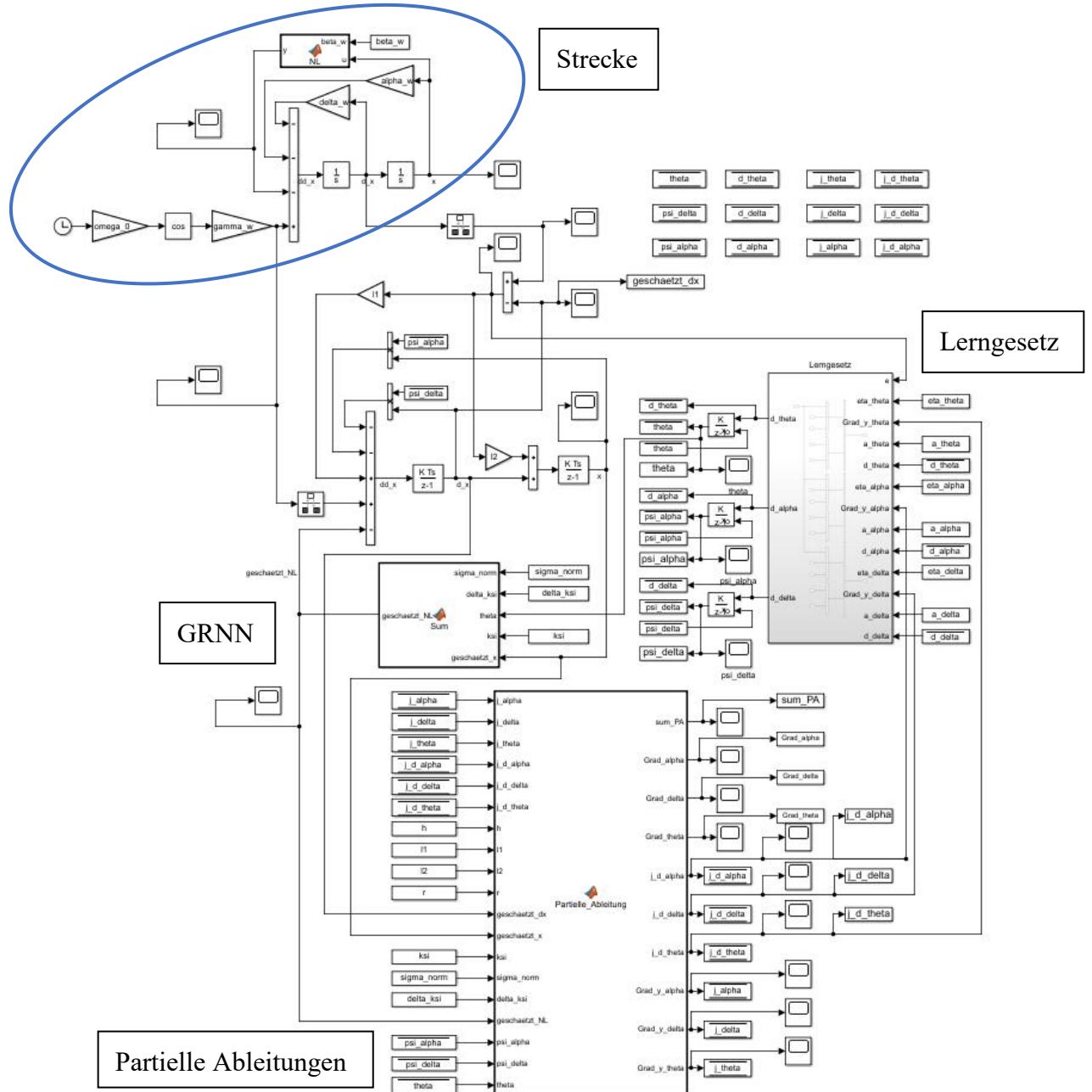


Abb. 5.2: Vorstrukturiertes rekurrentes Netz zur Identifikation der Mechanik eines Duffing-Oszillators als Beobachter in Matlab/Simulink

5.4 Bestimmung vom Parameter

Nach der Modellbildung soll wie im Abschnitt 4.5 gesagt noch geeignete Parameter bestimmt werden, wie die Parameter in vorgegebener Strecke, die Parameter von

GRNN, die Beobachterrückführung u.s.w.. In diesem Abschnitt wird dann über die Erfahrungen von der Auswahl der Werte von Parameter diskutiert.

Zuerst wird über die Parameter in vorgegebener Strecke diskutiert. Die vorgegebenen Eingangs- und Ausgangswerte sollen bestimmte Eigenschaften erfüllen, damit der Lernprozess von strukturierten rekurrenten Netz gut funktioniert. Diese Eigenschaften umfassen:

- Ein periodisches oder fast periodisches Ausgangswertkennlinie: Damit alle Gewichte vom GRNN vielmals gelernt werden können, sollen die Ausgangswerte der vorgegebenen Strecke, bzw. die Eingangswerte des GRNNs, am besten periodisch mit dem Zeitlauf sich verhalten. In jedem Zyklus soll bei jeder Stützstelle mindestens einmal der Lernprozess durchgeführt werden. Ein Gegenbeispiel ist wie in der Abbildung 5.3 zu sehen. Der Eingang vom GRNN in diesem Beispiel ist x . Der Eingangsbereich von GRNN, nämlich der Wertebereich von x , ist von -1 bis 1. Aber es gibt mit dem Zeitlauf nur zwei Mals, der Wert von x um 1 und -1 zu erreichen. Das heißt, die Gewichte an den Stützstellen -1 und 1 nur zweimal gelernt werden können. Es ist für den ganzen Lernprozess ungünstig. Eine Idee zur Lösung dieses Problems unter dem Beispiel in der Abbildung 5.3 ist, dass der Lernprozess vom Zeitpunkt $t=60s$ anfängt. Außerdem wenn der Anfangswert des Ausgangs veränderlich sind, kann durch die Änderung des Anfangswertes der Ausgangsverlauf verbessert werden.

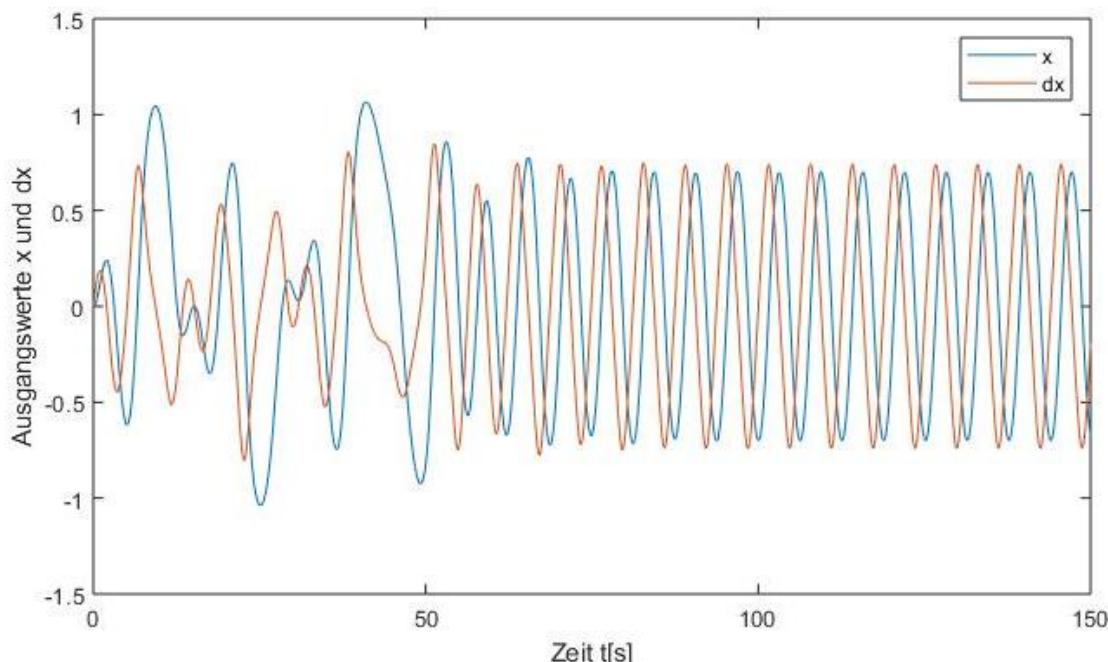


Abb. 5.3: Gegenbeispiel 1 mit $\omega_0 = 1$, $r = 0.3$, $\alpha = 1$, $\beta = -1$, $\delta = 0.2$ und $x_0 = 0$

- Die angemessene Frequenz: Wie die Abbildung 5.4 darstellt, ein Zyklus dauert fast

5 Sekunde und in ersten drei Zyklen erreicht die Amplitude das Maximum nicht. Das heißt, in je 5 Sekunden können alle Gewichte unter einem bestimmten Eingangswert nur ein- oder zweimal gelernt werden. Und die Approximation von den Gewichten, die um Stützstelle -4 und 4 liegen, können nur nach c.a. 15 Sekunden anfangen. Es ist für den ganzen Lernprozess ineffizient. Um dieses Problem zu lösen, kann entweder die Frequenz des Eingangsverlauf oder die Abtastzeit h sich vergrößert werden.

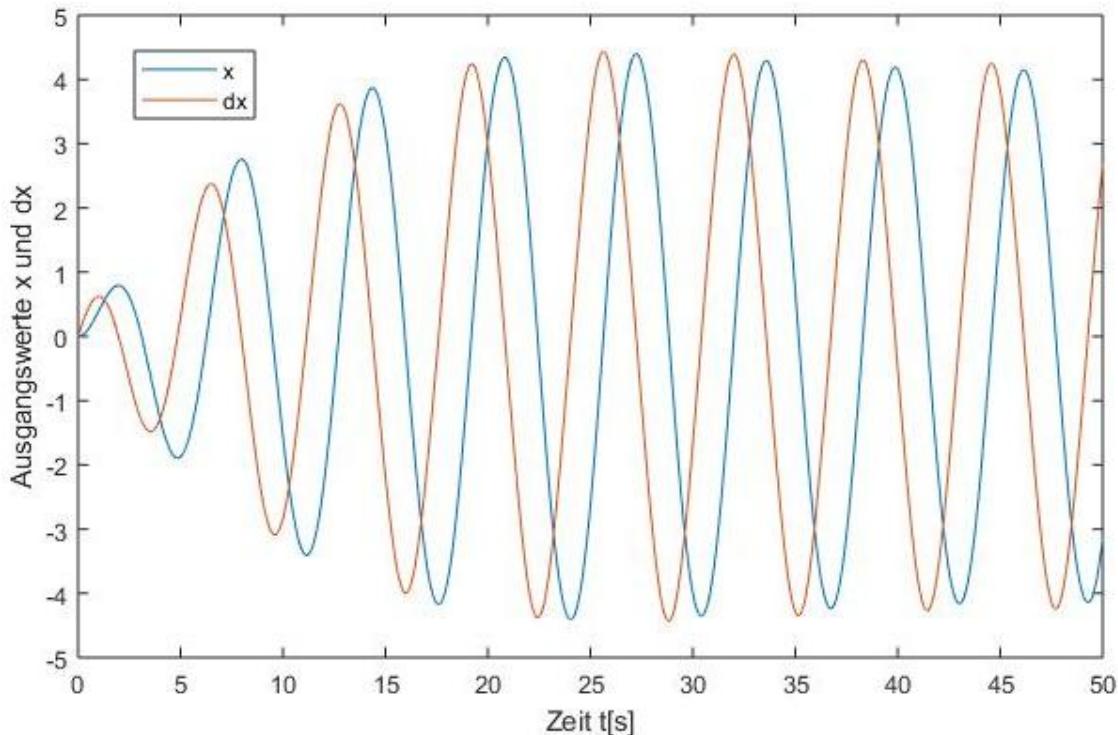


Abb. 5.4: Gegenbeispiel 2 mit $\omega_0 = 1$, $r = 1$, $\alpha = 1$, $\beta = -0.01$, $\delta = 0.2$ und $x_0 = 0$

- Der nicht zu große Ausgangsbereich: In der Abbildung 5.5 ist der Wertebereich von x von -150 bis 150. Mit so groß einem Eingangsbereich sollen im GRNN sehr viel Gewichte und Aktivierungsfunktionen eingestellt werden und entsprechend kann die Simulationszeit sehr groß sein.

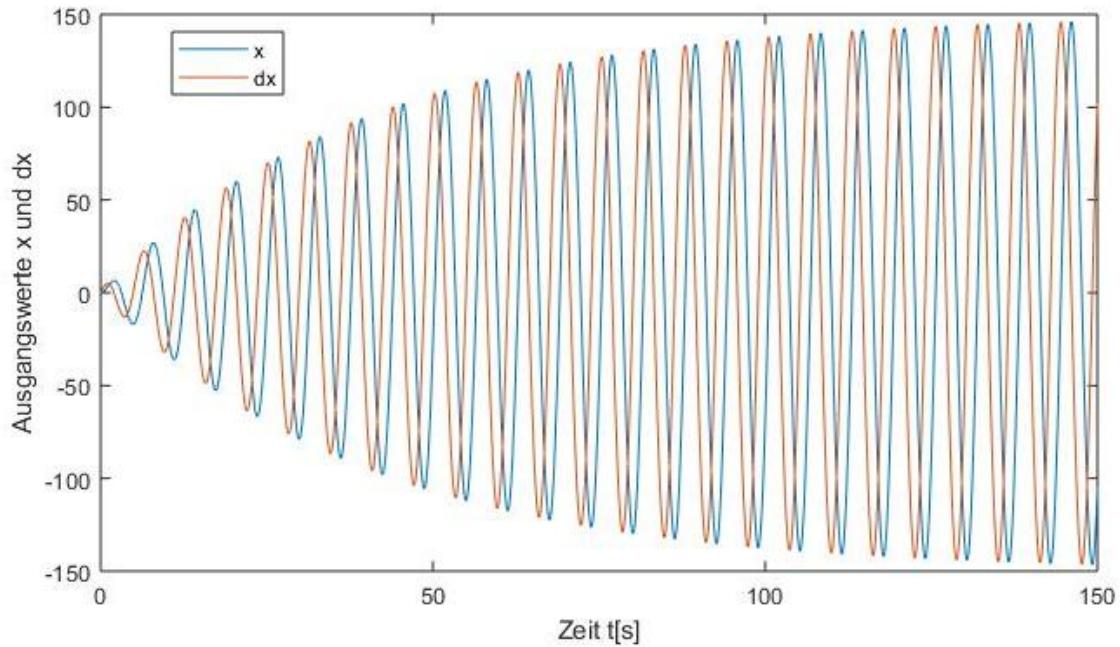


Abb. 5.5: Gegenbeispiel 3 mit $\omega_0 = 1$, $r = 7.5$, $\alpha = 1$, $\beta = 0$, $\delta = 0.05$ und $x_0 = 0$

Das ideale Bild vom Ausgangswert, das die obigen Bedingungen erfüllt, ist in der folgenden Abbildung 5.6 dargestellt:

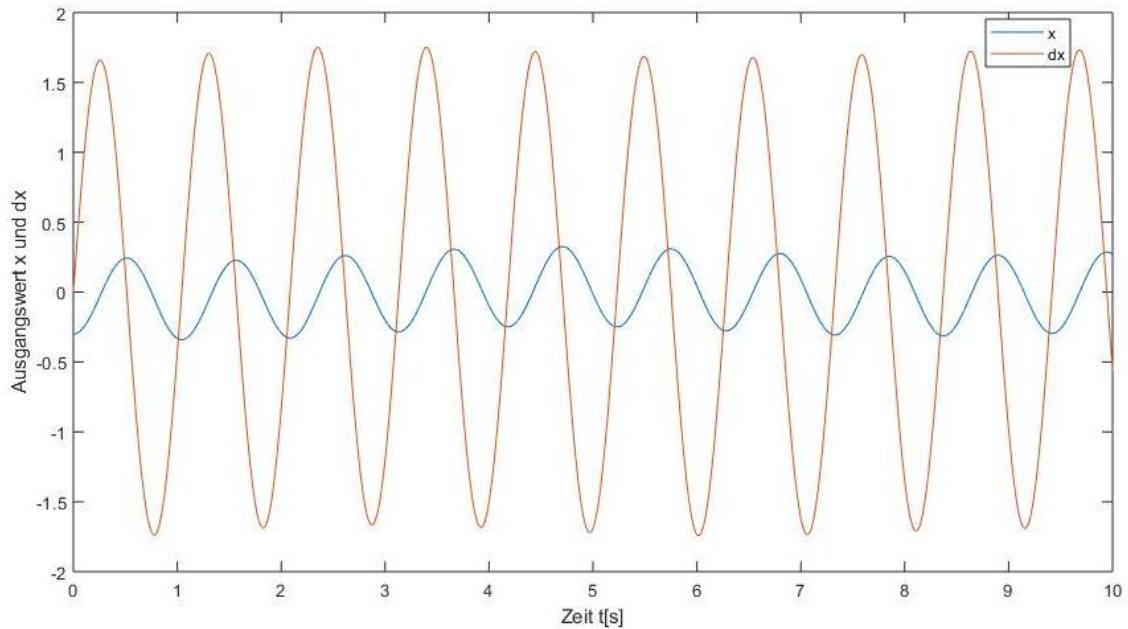


Abb. 5.6: geeignete Ausgangswerte mit $\omega_0 = 6$, $r = 10$, $\alpha = 1$, $\beta = -1$, $\delta = 0.2$ und $x_0 = -0.3$

Die Kennlinien des Ausgangswertes sind periodisch, stabil und halten sich in einem festen Wertebereich in jedem Zyklus (für x von -0.25 bis 0.25 und für dx von -1.6 bis 1.6). Die Frequenz ist fast 1Hz.

Danach sollen die Parameter im vorstrukturierten rekurrenten Netz eingestellt werden. Die Abtastzeit h ist gleich wie im letzten Kapitel als 1ms eingestellt. Die Beobachterrückführungen l_1 und l_2 sind als 10 und 1.75 definiert, nämlich das Maximum der Anregung und dx , damit die Werte nach beiden Rückführungen genug groß sind, die Konvergenz der Ergebnisse zu beibehalten.

Anhand des Wertbereich von x wird der Eingangsbereich des GRNNs, bzw. der Wertbereich von ξ , als (-0.3, 0.3) definiert. Unter realer Situation ist die Funktion von der Nichtlinearität unbekannt, deshalb soll die Anzahl der Aktivierungsfunktion in Bezug auf dem Eingangsbereich am Anfang ein bisschen größer und hierbei als $r=21$ eingestellt werden. Der Glättungsfaktor kann entsprechend klein ($\sigma_{norm} = 0.3$) definiert.

Weil der Eingangsbereich des GRNNs klein ist, kann auch die Lernschrittweite η von nichtlinearem Parameter klein sein und als 0.005 definiert. Die Lernschrittweite η von linearem Parameter kann anhand des Verhältnisses der Größenordnung zwischen vorgegebenem Parameter α , δ und dem Maximum von x . Hierbei ist $\eta_\alpha = 0.05$ und $\eta_\delta = 0.003$. Anhand der Identifikationsgeschwindigkeit vom Parameter können die Lernschrittweite weiter verändert werden. Der Momentumterm von allem Parameter werden gleich wie sie im Beispiel im Kapitel 4 als 0.95 definiert.

Die Anfangswerte von linearem und nichtlinearem Parameter können in erster Durchführung des vorstrukturierten rekurrenten Netzes einfach als 0, 0 und [0 0 0 0 0] eingestellt. Der Fuzzy-Bereich des Parameterwerts kann durch Analyse des Identifikationsverlaufs bestimmt werden. Danach können die genaueren Anfangswerte von linearem und nichtlinearem Parameter eingestellt werden. Hierbei sind $\alpha_0 = 1.4$, $\delta_0 = 0.3$, und $\underline{\omega}_0 = [0 0 0 0 0]$.

5.5 Ergebnisse und Beurteilung

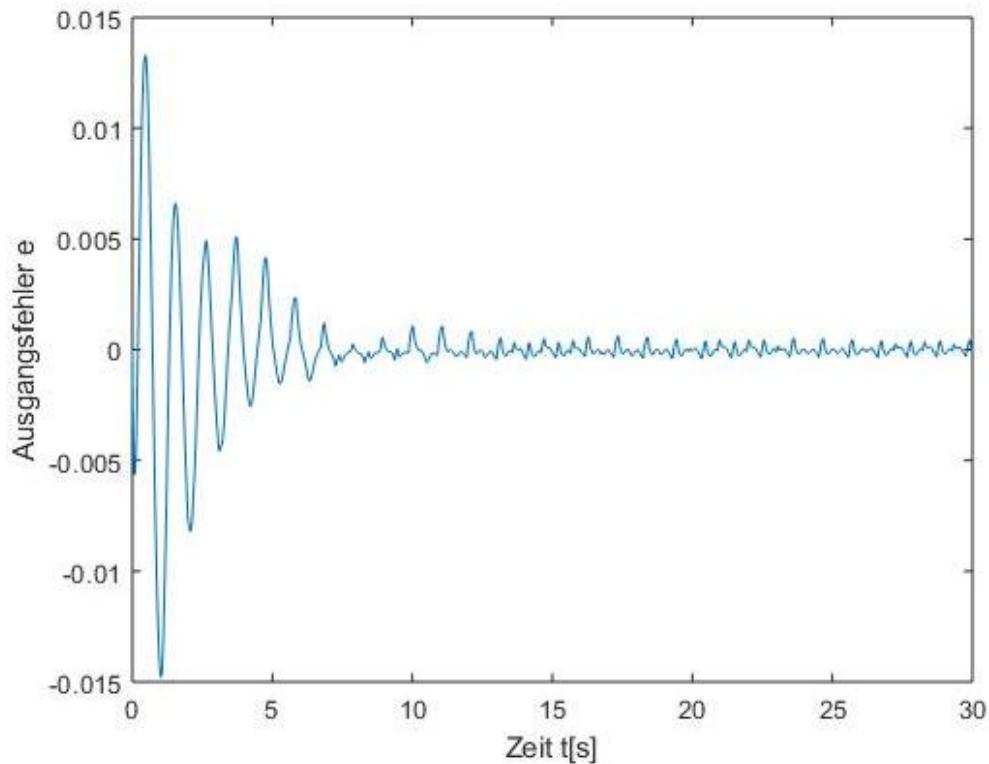


Abb. 5.7: Ausgangsfehlerverlauf während der Identifikation

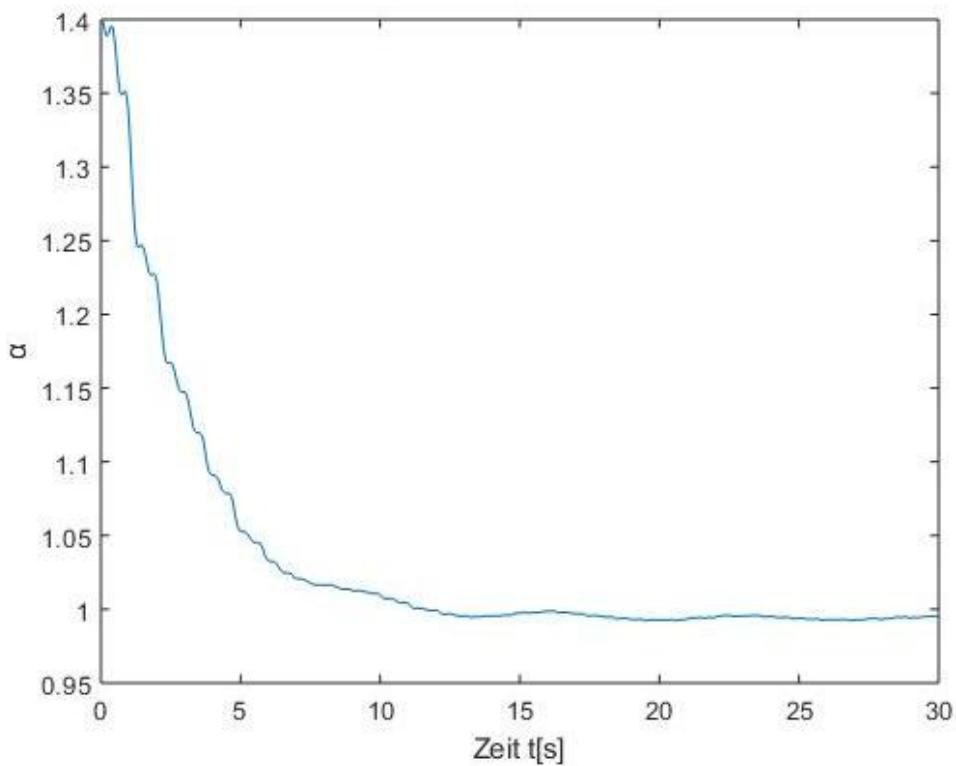


Abb. 5.8: Identifikationsverlauf von α

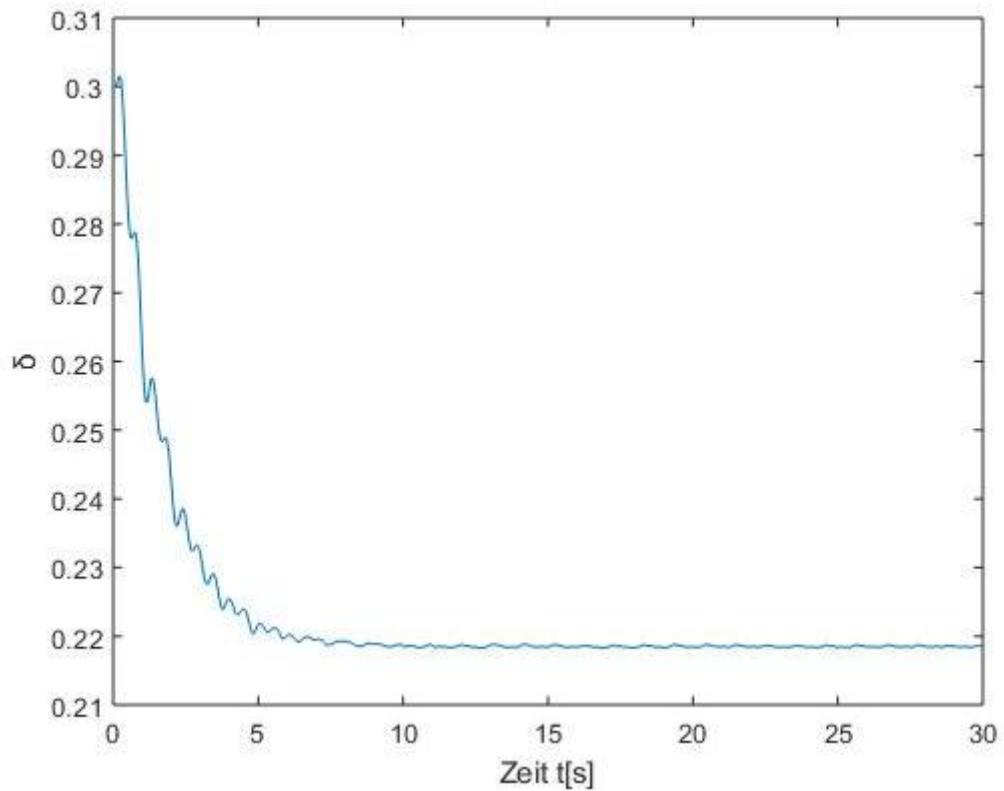
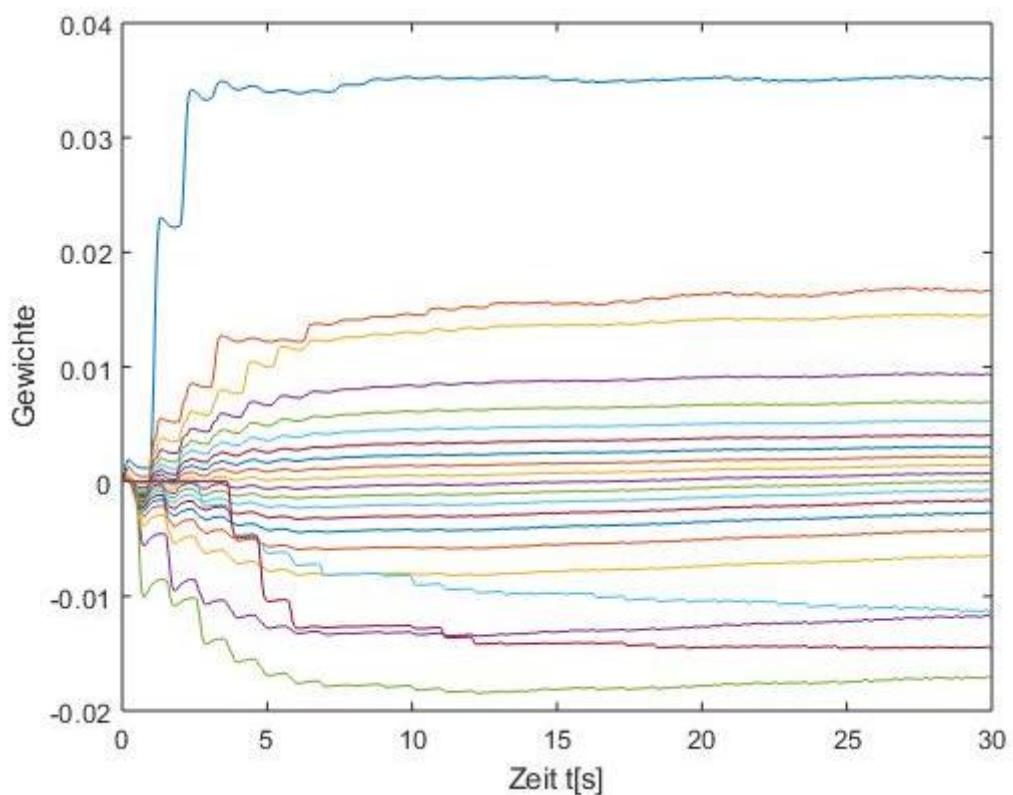
Abb. 5.9: Identifikationsverlauf von δ 

Abb. 5.10: Identifikationsverlauf der Gewichte des Funktionsapproximators

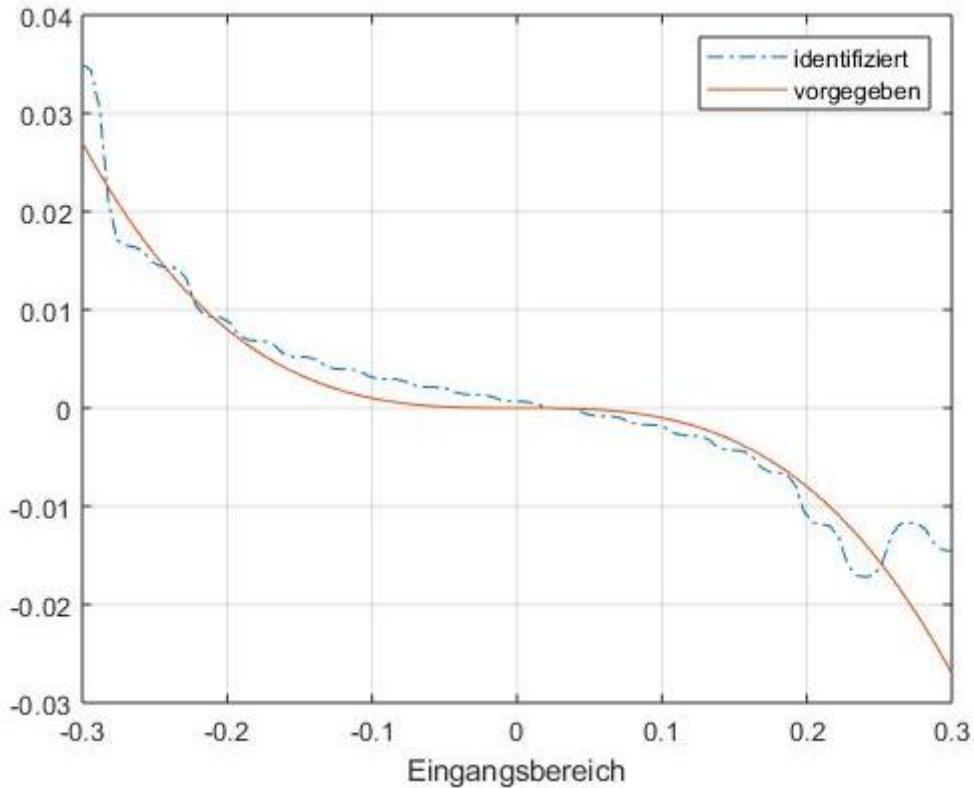


Abb. 5.11: Identifikationsergebnis der Nichtlinearität

Aus die Abbildungen 5.7 bis zu 5.11 ist es ersichtlich, dass die Ergebnisse von der Simulation nicht ideal sind. Der Ausgangsfehler in der Abbildung 5.7 und die linearen und nichtlinearen Parameter in der Abbildungen 5.8, 5.9 und 5.10 sind im Vergleich zu den Anfangswerten seit c.a. 8 Sekunden schon sehr klein. Darüber hinaus ist der identifizierte Wert von δ ($\hat{\delta} = 0.22$) nicht ganz gleich der von entsprechendem vorgegebenem Parameter ($\delta = 0.2$). Außerdem ist das Identifikationsergebnis der Nichtlinearität auch ungünstig. Besonders entspricht die identifizierte Kennlinie an den beiden Seiten die vorgegebene Kennlinie nicht.

Der Ausgangsfehler ist in der Lage, sich dem Punkt 0 anzunähern, deshalb kann die Grundfunktion von dieser vorstrukturierten rekurrenten Netz funktionieren. Nach 8 Sekunden erreichen die identifizierten Parameter stabile Werte, was bedeutet, die Auswahl von Lernparameter richtig ist.

Das Hauptproblem liegt dann beim GRNN. Beim GRNN ist die identifizierte Kennlinie bei der Extrapolation immer auf einer Konstante beibehalten. Diese Eigenschaft ist bei der Approximation von einigen Funktionen wie \arctan hilfreich, weil der Absolutwert der Ableitung von der Funktion \arctan bei der Entfernung des Eingangs vom Nullpunkt sich verkleinert. Aber in diesem Beispiel die zu approximierende Funktion ist eine kubische Gleichung. Je entfernter der Abstand zwischen dem Eingangspunkt und Nullpunkt ist, je schneller der Ausgangswert der kubischen Gleichung ansteigt. Deshalb ist das GRNN nicht in der Lage, die kubische Gleichung an beiden Seiten des

Eingangsbereiches zu approximieren. Außerdem ist der Eingangsbereich des GRNNs nicht ganz gleich (-0.3, 0.3) sondern ein bisschen kleiner, deshalb kann die Gewichte an beiden Seiten nicht genügend gelernt werden.

Weil das GRNN die kubische Gleichung nicht gut approximieren kann und es zwischen der vorgegebenen und identifizierten Kennlinie immer Fehler gibt, um den gesamten Ausgangsfehler zu 0 zu halten, soll der Fehler im GRNN durch die Änderung des linearen Parameters kompensiert werden. Dieses verursacht das Resultat, dass die identifizierten linearen Parameter δ nicht gleich groß wie die vorgegebenen linearen Parameter sind.

5.6 Kurzzusammenfassung

In diesem Kapitel wird diskutiert, wie das vorstrukturierte rekurrente Netz aussieht, wenn es in einem mechatronischen System mit 2-ter Ordnung verwendet wird. Mit einer Gleichung mit 2-ter Ordnung gibt es zwei Differenzialgleichungen und entsprechend zwei Jacobi-Matrizen für jeden zu identifizierenden Parameter. Außerdem soll mit zwei Integratoren in der Strecke zwei Beobachterrückführungen eingestellt werden.

Danach wird diskutiert, wie die Parameter im strukturierten rekurrenten Netz bestimmt werden können und welche Eigenschaften die vorgegebenen Ein- und Ausgangswerte vom strukturierten rekurrenten Netz erfüllt werden soll, damit es gut funktionieren kann.

Am Ende wird beim Analysieren der Ergebnisse noch die Begrenztheit des GRNNs herausgefunden. Bei der Approximation der kubischen Gleichung kann statt des GRNNS anderes neuronale Netz, wie z.B. MLP, genutzt werden.

6 Zusammenfassung und Ausblick

Vorstrukturiertes rekurrentes Netz ist eine Struktur, die auf einer Beobachtungsstruktur und einem rekurrenten Teil basiert, um den unbekannten linearen und nichtlinearen Teil des mechatronischen Systems durch GRNN zu identifizieren, ohne die internen Parameter einzugreifen. Weil das vorstrukturierte rekurrente Netz anhand der Struktur des realen Systems aufgebaut wird, können die identifizierten Parameter auch physikalisch Interpretierbarkeit besitzen.

Der gesamte Prozess zur Lösung des Problems ist wie folgt: Zunächst müssen Lernfähiger Luenberger-Beobachter gemäß der bekannten Systemstruktur erstellt werden. Basierend auf Luenberger-Beobachter können die vorhandenen Differentialgleichungen aufgelistet, die partiellen Ableitungen des Ausgangsfehlers basierend auf linearen und nichtlinearen Parametern abgeleitet und Jacobi-Matrix bestimmt werden. Dann wird in Matlab/Simulink der vorstrukturiertrekurrenter Netz eingerichtet und die vorgegebenen Eingabe- und Ausgabewerte eingegeben. Danach müssen die Anfangswerte der entsprechenden linearen und nichtlinearen Parameter, die Parameter des neuronalen Netzwerks und den Wert von Beobachterrückführungen bestimmt werden, bis ein stabiles Ergebnis mit einem Fehler nahe 0 erhalten zu werden.

In dieser Arbeit werden die vorgegebenen Eingangs- und Ausgangswerte von vorstrukturiertem rekurrentem Netz über das in Simulink erstellte Regelungsmodell erhalten, da kein spezifisches mechatronisches Instrument hierbei beteiligt ist. In praktischen Anwendungen sollten die von vorstrukturiertem rekurrentem Netz geforderten Eingangs- und Ausgangswerte von realen mechatronischen Systemen bereitgestellt werden.

Der größte Vorteil von vorstrukturiertem rekurrentem Netz besteht darin, die zu identifizierenden Parameter sich durch den Lernprozess die realen Werte annähern können. Weil der ganze Prozess im Computer durchgeführt wird und nur einige Paare von vorgegebenen Ein- und Ausgangswerte benötigt, ist auch der erforderliche Zeit-, Personal- und Materialaufwand, um die Parameterwerte zu erhalten, viel kleiner als die direkte physikalische Ableitung des ursprünglichen Systems.

Der Nachteil von vorstrukturiertem rekurrentem Netz besteht zuerst darin, dass für komplexe mechatronische Systeme, die mehrere lineare Parameter oder mehrere nichtlineare Teile oder mehrere Ordnungen enthalten, die zu erstellende Struktur von rekurrentem Netz ebenfalls komplizierter ist, d. h. ein strukturiertes rekurrentes Netz mit bestimmter Struktur passt nicht allgemein zu allen mechatronischen Systems an. Außerdem um die Konvergenz des Simulationsprozesses sicherzustellen, müssen die Anfangswerte der Parameter von rekurrentem Netz und die Werte von Beobachterrückführung im Voraus bestimmt werden. Es gibt auch keine spezifische Regel dafür und sie sind für verschiedene Strukturen ebenfalls unterschiedlich und können nur anhand der Erfahrungen des Arbeiters und zahlreicher Versuche bestimmt

werden. Falsche Anfangsparameterwerte kann nicht zu der konvergierten Ergebnisse führen, und die Zeit, die zur Ermittlung der Parameterwerte benötigt wird, nimmt den größten Teil der gesamten Aufgabe ein. Und um die Anzahl der Aktivierungsfunktion r , den Glättungsfaktor σ_{norm} sowie Stützstelle ξ des GRNNs im Voraus zu bestimmen, soll der Eingangsbereich der Nichtlinearität als Vorwissen bereitgestellt werden.

Schließlich ist das neuronale Netzwerk selbst nicht für unregelmäßige zu lernende Eingabe- und -ausgabewerte geeignet. Beispielsweise liegen 100 vorgegebene Eingabewerte des neuronalen Netzwerks im Intervall (-1, 1) und 2 Mals in (2, 3). Die Lernqualität des GRNNs in (2, 3) ist dann relativ gering.

Im Allgemeinen eignet sich vorstrukturiertes rekurrentes Netz zur Simulation der linearen und nichtlinearen Parameter von Systemen mit bekannten spezifischen Strukturen, unbekannten internen Parametern und bekanntem Eingangs- und Ausgangsbereich mit gleichmäßig verteilten Werte.

Basierend auf bestehenden Problemen kann weiter optimiert werden. Um eine bestimmte Struktur von rekurrentem Netz für mehrere mechatronische Systeme, größeren Bereich der Parameterauswahl und größeren Bereich von Eingangs- und Ausgangswerten geeignet zu machen, können versucht werden, das General-Regression-Neural(GRN)-Netzwerk durch Multilayer Perceptron(MLP) zu ersetzen. Der Nachteil besteht darin, dass die physikalische Interpretierbarkeit von vorstrukturiertem rekurrentem Netz abnehmen und die Simulationszeit sowie die Komplexität der Modellierung zunehmen kann.

7 Literaturverzeichnis

- [C03] CHRISTIAN, H.: *Identifikation nichtlinearer mechatronischer Systeme mit strukturierten rekurrenten Netzen*, Lehrstuhl für Elektrische Antriebssysteme, Technische Universität München, 2003.
- [S10] SCHRÖDER, D.: *Intelligente Verfahren*, 2010.
- [MCP88] MCCULLOCH, W. S. und W. PITTS: *A logical calculus of the ideas immanent in nervous activity*. Bulletin of Mathematical Biophysics 5, 1988.
- [ROS88] ROSENBALTT, F.: *The perceptron: a probabilistic model for information storage and organization in the brain*. Psychological review 65, 1988.
- [MIP88] MINSKY, M. und S. PAPERT: Perceptrons - Expanded Edition: *An Introduction to Computational Geometry*. 1988.
- [W74] WERBOS, P. J.: *Beyond regression: new tools for prediction and analysis in the behavioral sciences*. Ph.D. thesis, Harvard University, Cambridge, MA, 1974.
- [RHW88] RUMELHART, D. E., G. E. HINTON und R. J. WILLIAMS: *Learning internal representations by error propagation*. 1988.
- [C06] CHRISLB, Wikipedia.
https://commons.wikimedia.org/wiki/File:ArtificialNeuronModel_deutsch.png
[2006]
- [P91] PAPAGEORGIOU, M.: *Optimierung. Statische, dynamische, stochastische Verfahren für die Anwendung*. 1991.
- [W16] Weibo,
<http://weibo.com/5066241201/EggmMg7ym?ref=collection&type=comment>
[01.2020]
- [S95] SCHRÖDER, D.: *Verfahren zur Beobachtung nicht messbarer Größen nichtlinearer dynamischer Systeme*. 1995.
- [D18] G. DUFFING: *Erzwungene Schwingung bei veränderlicher Eigenfrequenz und ihre technische Bedeutung*, Vieweg, Braunschweig, 1918.
- [KJH08] KORSCH, H. J., H.-J. JODL, T. HARTMANN: *Chaos*, 2008