

# Adaptive Systemidentifikation mit Neuronalen Netzen zur Profilsteuerung in Walzwerken<sup>1</sup>

Kontinuierliches Lernen in der Prozeßautomatisierung

*Dr.-Ing. Peter Protzel, Erlangen*

*Dipl.-Phys. Lars Kindermann, Erlangen*

*Dipl.-Inform. Michael Tagscherer, Erlangen*

*Dr. rer. pol. Achim Lewandowski, Erlangen*

*Bayerisches Forschungszentrum für Wissensbasierte Systeme (FORWISS)  
Am Weichselgarten 7, 91058 Erlangen*

## Zusammenfassung

Viele industrielle Prozesse, insbesondere in der Stahl- und Aluminiumproduktion, haben sowohl stark nichtlinearen als auch zeitvarianten („Tagesform“) Charakter. Werden Neuronale Netze zur Modellierung solcher Prozesse eingesetzt, erfordert dies im Gegensatz zur üblichen Anwendung Neuronaler Netze ein zeitlich unbegrenztes oder *kontinuierliches* Lernen. In diesem Beitrag werden die Aufgabenstellung und die Anforderungen an ein kontinuierliches Lernen mit Neuronalen Netzen diskutiert und verschiedene Lösungsansätze aufgezeigt. Am Beispiel der Profilsteuerung in Walzwerken wird eine industrielle Anwendung vorgestellt, bei der ein Neuronales Netz durch die Kombination mit einem konventionellen, mathematischen Modell die Vorteile beider Verfahren auf elegante Weise miteinander verknüpft.

## 1. Einleitung

Neuronale Netze werden in der Prozeßsteuerung und -regelung typischerweise dann eingesetzt, wenn kein oder kein ausreichend genaues Streckenmodell existiert, z.B. weil die Nichtlinearitäten des Prozesses nicht genau bekannt und daher nicht analytisch faßbar sind. Eine der Grundaufgaben Neuronaler Netze in der Prozeßautomatisierung ist daher die Identifikation von nichtlinearen Prozessen. Als Resultat erhält man in der Regel die Modellierung eines Kennfeldes, das z.B. eine Ausgangsgröße  $y$  des Prozesses als Funktion des Eingangsvektors  $x$  beschreibt. Fragen der Dynamik von Prozessen, d.h. die zeitliche Abhängigkeit des Ausgangs nicht nur von aktuellen, sondern auch von vergangenen Eingangswerten, werden im folgenden ausgeklammert, da sie eine zusätzliche Problem-Dimension darstellen, die den Rahmen dieses Beitrags sprengt. Die Aufgabe der Identifikation nichtlinearer Prozesse wird erheblich erschwert, wenn der Prozeß zeitvariant ist, d.h. wenn sein Kennfeld auch von der Zeit  $t$  abhängt [1]. Um das Prinzip zu verdeutlichen, betrachten wir im folgenden nur eine Eingangsgröße  $x$ . Bezüglich der Ausgangsgröße eines Prozesses  $y(x, t)$  überlagern sich dann mehrere Effekte:

---

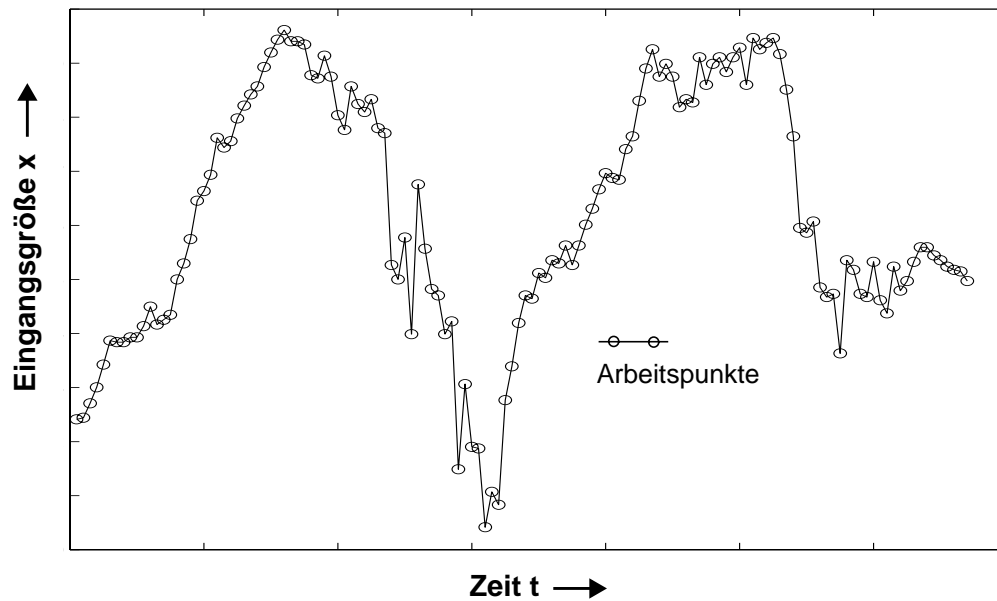
<sup>1</sup>. Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung, Wissenschaft, Forschung und Technologie unter dem Förderkennzeichen 01IN505B gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

- a) Nichtlinearität des Kennfeldes  $y(x) \neq ax + b$ ,
- b) Zeitvariantes Prozeßverhalten  $y(x, t_1) \neq y(x, t_2)$ ,
- c) Arbeitspunktverschiebungen der Eingangsgrößen  $x = x(t)$ .

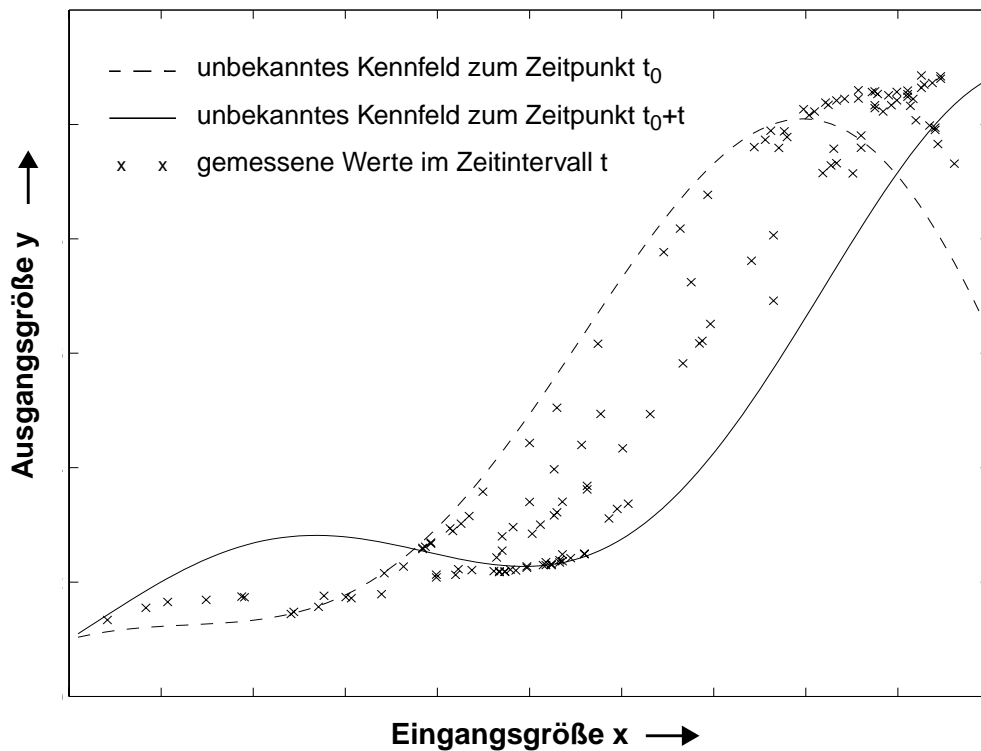
Punkt a) entspricht der Approximation einer nichtlinearen, zeitinvarianten Funktion und stellt die „klassische“ Anwendung für Neuronale Netze dar. Die Netze können das nichtlineare Kennfeld  $y(x)$  im Prinzip beliebig genau nachbilden, falls ausreichend Trainingsdaten zur Verfügung stehen, was wiederum durch eine entsprechende Arbeitspunktvariation  $x(t)$  (Punkt c) gewährleistet werden muß. Typischerweise wird das Netz in diesem Fall einmalig trainiert und im anschließenden Einsatz nicht mehr verändert. Ist der Prozeß zusätzlich zeitvariant (Punkt b), ist eine schritthaltende Adaption des Neuronalen Netzes erforderlich. D.h., ein einmaliges Training vor dem „Einsatz“ des Neuronalen Netzes ist nicht mehr ausreichend, sondern es muß während des Einsatzes kontinuierlich „nachtrainiert“ werden. Im Gegensatz zum zeitlich begrenzten Training bezeichnen wir dies als zeitlich unbegrenztes oder *kontinuierliches Lernen*. Andere Bezeichnungen wie on-line Lernen, Adaption, sequentielles oder inkrementelles Lernen sind ebenfalls geläufig, aber nicht eindeutig definiert und können je nach Kontext zu Fehldeutungen führen.

Beim kontinuierlichen Lernen treten eine Reihe von bekannten Problemen auf, z.B. wie kann sichergestellt werden, daß das Neuronale Netz relevante neue Daten möglichst schnell lernt, ohne alte, immer noch relevante Informationen zu vergessen? (Stabilitäts- vs. Plastizitätsdilemma, [2]). Dies ist insbesondere dann problematisch, wenn die Effekte b) und c) auf ähnlichen Zeitskalen auftreten. Leider kommt dies in industriellen Anwendungen nicht selten vor; man spricht auch von der „Tagesform“ einer Anlage, die ein Bediener manchmal erkennen kann, der ein Prozeßmodell jedoch automatisch folgen muß.

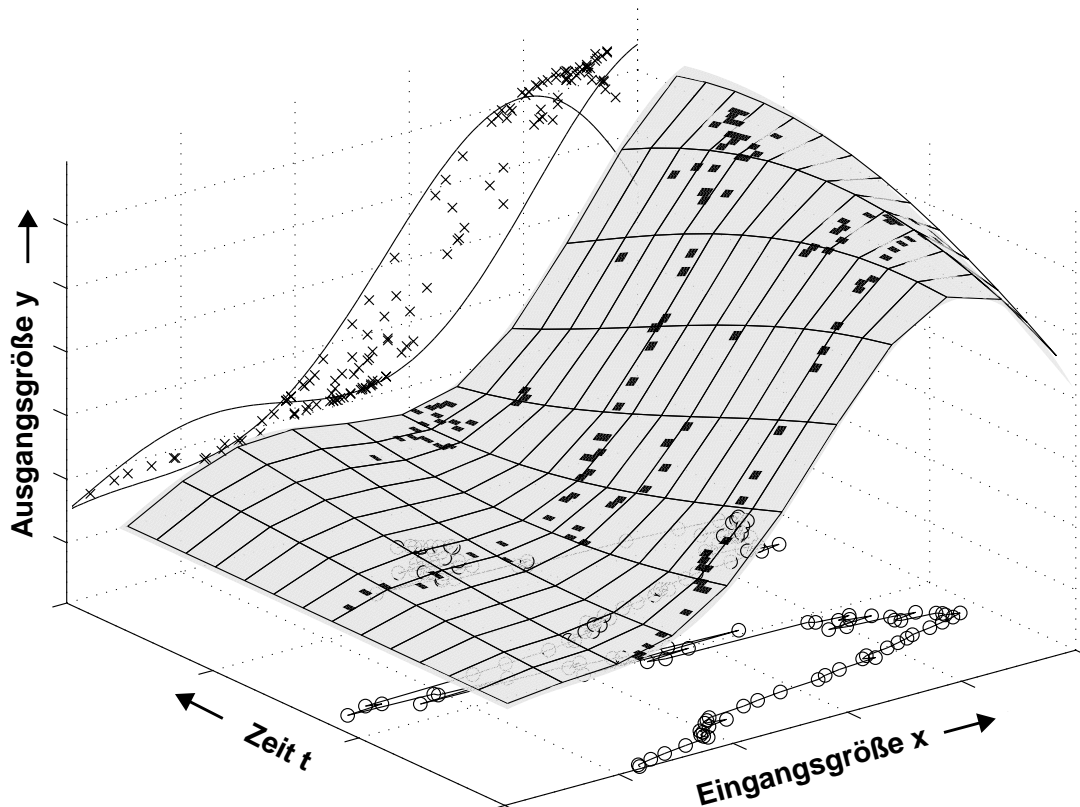
Dieses Grundproblem des kontinuierlichen Lernens einer nichtlinearen, zeitvarianten Funktion soll anhand der folgenden Bilder veranschaulicht werden. Bild 1 zeigt zunächst einen angenommen zeitlichen Verlauf der Eingangsgröße  $x(t)$ , die z.B. die Arbeitspunktverschiebung eines Prozesses beschreibt. In diesem Beispiel wird der gesamte Wertebereich von  $x$  im Laufe der Zeit mehrfach „durchfahren“. Bild 2 zeigt als Beispiel die zeitliche Veränderung eines nichtlinearen Zusammenhang zwischen Eingangs- und Ausgangsgröße eines Prozesses. Dies ist dargestellt anhand von zwei „Schnappschüssen“ des Kennfeldes zum Zeitpunkt  $t_0$  und zum Zeitpunkt  $t_0+t$ . In der Praxis sind dieses Kennfelder nicht bekannt, sondern es liegen lediglich Meßwerte vor, die zusätzlich mit Rauschen behaftet sein können. In Bild 2 sind die Meßwerte eingetragen, die sich entsprechend des zeitvarianten Kennfeldes und des Arbeitspunktverlaufs nach Bild 1 ergeben. Die dreidimensionale Darstellung des zeitvarianten Kennfeldes  $y(x, t)$  in Bild 3 verdeutlicht das Gesamtproblem: Die zur Modellierung verfügbaren Daten sind als dunkle Flächen auf dem (unbekannten) Kennfeld markiert. Bild 1 und 2 finden sich entsprechend als Projektionen auf die Grund- bzw. Seitenfläche wieder. Nur aus den bis zu einem Zeitpunkt  $t$  gesehenen Daten soll ein Modell des Kennfelds zum Zeitpunkt  $t+1$  „erlernt“ werden. Die zeitliche Struktur der Daten kann und muß dabei berücksichtigt werden. Wenn alle Daten „gleichberechtigt“, d.h. ohne Berücksichtigung der Zeitstruktur, in die Modellierung eingehen, entsteht ein „Durchschnittsmodell“, das über die zeitliche Veränderung des Kennfeldes mittelt. Dieses Modell entspräche dann einem gemitt-



**Bild 1.** Arbeitspunktverlauf  $x(t)$  eines beispielhaften Prozesses.



**Bild 2.** Darstellung der Daten, die sich aus der zeitlichen Veränderung eines nichtlinearen Kennfelds und aus dem Arbeitspunktverlauf nach Bild 1 ergeben. Ohne Berücksichtigung der Zeitstruktur scheinen die Daten verrauscht zu sein.



**Bild 3.** Beispiel einer nichtlinearen, zeitvarianten Funktion und zur Modellierung dieser Funktion verfügbarer Daten (dunkle Flächen) entsprechend eines gegebenen Arbeitspunktverlaufs. Nur aus den bis zu einem Zeitpunkt  $t$  gesehenen Daten soll ein Modell des Kennfelds zum Zeitpunkt  $t+1$  „erlernt“ werden. Die zeitliche Struktur der Daten kann und muß dabei berücksichtigt werden.

teltem Kennfeld aus allen Datenpunkten in Bild 2, wobei die unverrauschten Daten durch das Ignorieren der Zeitstruktur verrauscht erscheinen.

## 2. Kontinuierliches Lernen mit Neuronalen Netzen

Vor dem Einsatz neuronaler Modelle empfiehlt es sich stets, eine lineare Modellierung vorzunehmen. Oft leistet eine lineare Regression, evtl. mit höheren Potenzen, schon eine recht gute Annäherung an die Daten. Im Hinblick auf das kontinuierliche Lernen empfiehlt sich insbesondere die rekursive Berechnung der Regressionskoeffizienten [3] mit minimalem Rechenaufwand. Falls eine lineare Modellierung nicht die erforderliche Genauigkeit liefert, kommen neuronale Modelle in Betracht, wobei die meist für zeitinvariante Aufgaben entwickelten Lernverfahren geeignet erweitert werden müssen.

Wie beim Lernen von zeitinvarianten Funktionen muß auch beim kontinuierlichen Lernen zu Beginn die Topologie des Neuronalen Netzes festgelegt werden. Dabei ist jedoch zu beachten, daß bei zeitvarianten Prozessen die Komplexität der zu lernenden Funktion zunehmen kann. Das bedeutet, daß auch die Netztopologie ständig an diese neuen Anforderungen angepaßt werden sollte. Im nächsten Schritt muß entschieden werden, in welcher Form die Trainingsdaten dem Netz präsentiert werden. Im Gegensatz zum zeitlich begrenzten Lernen

besteht beim kontinuierlichen Lernen keine scharfe Trennung zwischen Trainings- und Anwendungsphase. In der Regel wird ständig zwischen Trainings- und Anwendungsphase hin- und hergeschaltet. Dabei stellt sich die Frage, wann das Netztraining angestoßen wird und welche Daten hierzu verwendet werden.

### **Gleitendes Fenster**

Ein naheliegende Methode besteht darin, ein Fenster auf den Daten zu definieren und dieses mit einer festen Schrittweite über die Daten zu schieben. Für das Netztraining werden jeweils die Daten innerhalb des gleitenden Fensters verwendet. Abbruchkriterien für das Training können beispielsweise eine feste Epochenzahl, eine vorgegebene Fehlerschwelle oder ein Signal sein, welches anzeigt, daß das Modell nun angewendet werden muß, um z.B. eine neue Prognose zu liefern.

Die Wahl der Parameter (Fenstergröße, Schrittweite, Abbruchkriterium usw.) hat einen entscheidenden Einfluß auf die Modellierungsgüte und die Stabilität des Modells und kann in der Regel nur durch Experimente an das jeweilige Problem angepaßt werden. Wird beispielsweise das Modell durch eine zu kleine Fenstergröße und Fehlerschranke zu intensiv mit einem nicht repräsentativen Datensatz nachtrainiert, kann bereits vorhandenes „Wissen“ durch eine zu starke Adaption völlig verloren gehen. Insbesondere in dem Grenzfall, bei dem nur mit dem letzten, aktuellen Muster nachtrainiert wird, besteht die Gefahr, daß das Netz die eigentliche Funktion nicht identifiziert, sondern lediglich eine Offset-Anpassung vornimmt, um den Fehler jeweils auf dem einen Datenpunkt zu minimieren.

### **Periodisches Netztraining**

Im Gegensatz zum gleitenden Fenster werden beim periodischen Netztraining die anfallenden Trainingsdaten über einen längeren Zeitraum gesammelt, um anschließend mit diesem „großen“ Datensatz das Netz nachzutrainieren. Feste Zeitschranken, das Erreichen einer vorgegebenen Datenmenge, Maschinenpausen oder Teilewechsel können Signale bzw. Situationen sein, durch die ein Netztraining gestartet wird.

Die größere Datenmenge soll sicherstellen, daß das Training mit repräsentativen Daten erfolgt und somit zu einer hohen Modellgüte führt. Wird jedoch das Intervall zwischen zwei Trainingseinheiten zu groß gewählt, können sich auch hier Probleme ergeben, wenn der Prozeß zeitlich stark schwankt. Der Fehler, den das Netz als Prozeßmodell bei der Anwendung liefert, steigt infolge der Zeitvarianz des Prozesses bis zum nächsten Nachtraining an, da das Netz erst dann wieder an die Veränderungen angepaßt wird. Werden anschließend alle gesammelten Daten gleichberechtigt, d.h. ohne Berücksichtigung der Zeitstruktur, zum Nachtraining verwendet, wird ein über die zeitliche Veränderung des Prozesses gemitteltes Modell gelernt, wie bereits im ersten Abschnitt (Bild 2 u. 3) erläutert wurde.

Um eine höhere Plastizität zwischen zwei Trainingsabschnitten zu erreichen, kann periodisches Netztraining z.B. mit einem gleitenden Fensteransatz kombiniert werden. Durch die permanente Präsentation aktueller Daten (gleitendes Fenster) erfährt das Netz eine Anpassung an den aktuellen Zustand des Prozesses. Die oben beschriebenen Probleme der Parametereinstellung und des periodischen Nachtrainings bleiben jedoch weitgehend erhalten.

## Aufteilung des Arbeitsraums

Bei zeitvarianten Prozessen kommt es häufig vor, daß sich die Prozeßcharakteristik nur lokal, d.h. in bestimmten Bereichen des Raums der Eingangsgrößen (Arbeitsraum) ändert. Für die Stabilität beim kontinuierlichen Lernen ist es daher günstig, wenn mehrere Modelle für die verschiedenen Arbeitsraumbereiche existieren und bei lokalen Veränderungen dann nur das jeweils „zuständige“ Modell angepaßt werden muß. Neuronale Netze kommen dieser Anforderung nur bedingt entgegen; es gibt sowohl Netze mit globalem Verhalten (z.B. Multilayer-Perceptrons - MLPs) als auch Netze mit lokalen Eigenschaften (z.B. Netze mit radialen Basisfunktionen - RBF Netze [4]).

Multilayer-Perzeptrons (MLPs) mit sigmoider Aktivierungsfunktion zeigen globales Verhalten in dem Sinne, daß sich die Aktivierung eines Neurons und damit auch die Veränderung von Gewichten im gesamten Raum der Eingangsgrößen (Arbeitsraum) auswirkt. Beim kontinuierlichen Lernen hat das zur Folge, daß eine (erwünschte) Anpassung des Netzes an eine lokale Prozeßänderung auch (unerwünschte) Auswirkungen auf die gelernte Funktion in allen anderen Raumbereichen nach sich ziehen kann. Die Aktivität der Neuronen von RBF-Netzen ist im Gegensatz dazu nur in einem definierten radialsymmetrischen Bereich um ihr Zentrum signifikant größer als Null. Daher wirken sich lokale Anpassungen beim Nachtraining auch nur lokal aus. Das Problem bei RBF-Netzen ist die große Zahl erforderlicher Neuronen, die mit der Dimension des Arbeitsraums exponentiell ansteigt.

Eine Alternative sind Hybrid-Verfahren, bei denen unterschiedliche Ansätze mit ihren Vorteilen kombiniert werden. Es gibt zahlreiche Ansätze mit zwei Verfahren, bei denen die erste Stufe eine Aufteilung des Arbeitsraums durchführt, um anschließend in der zweiten Stufe in den einzelnen Arbeitsraumregionen „lokale Modelle“ zu plazieren. So wird beispielsweise bei Counterpropagation-Netzen über ein Kohonen-Netz eine Aufteilung des Eingaberaums gemäß der Dichteverteilung der Eingangsdaten durchgeführt und über einen linearen Assoziator werden einfache lineare Modelle für die entsprechenden Regionen angepaßt [5]. Dabei wird die Kohonen-Schicht unüberwacht und der Lineare Assoziator überwacht trainiert, d.h. das Training des Assoziators ist vom Training der Kohonen-Schicht abgekoppelt. Da der Assoziator warten muß, bis das Kohonen-Netz eine sinnvolle Raumaufteilung gefunden hat, kann sich diese schwache Bindung zwischen den beiden Ansätzen nachteilig auswirken. So werden z.B. Raumbereiche mit relativ wenigen Datenpunkten durch den Kohonen Algorithmus nur grob oder gar nicht aufgeteilt, jedoch kann gerade in solchen Bereichen eine feine Modellierung mit mehreren Modellen aufgrund starker Nichtlinearitäten erforderlich sein.

Aus der bisherigen Diskussion können bereits einige Anforderungen an Verfahren des kontinuierlichen Lernens abgeleitet werden:

1. Das Verfahren muß auch schnellen Änderungen der Prozeßcharakteristik folgen und dabei die Nichtlinearitäten des Prozesses modellieren. Dies erfordert mehr als nur eine Offset-Anpassung an einzelne Datenpunkte, die zwar schnell ist, aber zu keinem Modell führt. Eine Möglichkeit, dies zu erreichen, ist eine Kombination von Modellen mit verschiedener Adaptionsgeschwindigkeit, z.B. eine Kurzzeit- und eine Langzeitadaption.
2. Die Stabilität muß in der Regel durch eine Arbeitsraumaufteilung und lokal adaptive Modelle gewährleistet werden. Die Arbeitsraumaufteilung sollte sich bei Bedarf ebenfalls

an Prozeßveränderungen anpassen und sie sollte optimal sein bezüglich der Approximationsgüte der lokalen Modelle und nicht bezüglich der Dichte der Eingangsdaten. Dies erfordert eine enge Kopplung zwischen dem Verfahren zur Raumaufteilung und den lokalen Modellen.

3. Das Verfahren sollte bereits nach wenigen Trainingspunkten plausible Ergebnisse liefern und Netztopologie und Parameter im weiteren Verlauf weitgehend selbständig anpassen und mit zunehmender Datenmenge entsprechend verfeinern. Verfahren, die vor ihrem Einsatz zunächst eine umfangreiche manuelle Parameteroptimierung auf 10.000 Datensätzen erfordern, erreichen zwar u.U. eine höhere Leistung, scheiden aber in der Praxis häufig aus.

Diese Fragen und Probleme des kontinuierlichen Lernens, die insbesondere in industriellen Anwendungen bisher nur ansatzweise gelöst sind, werden als aktuelles Forschungsthema im BMBF Projekt AENEAS (Anwendung und Entwicklung Neuronaler Verfahren zur autonomen Prozeß-Steuerung) bearbeitet. Im nächsten Abschnitt werden zwei Ansätze zum kontinuierlichen Lernen beispielhaft vorgestellt, um im vierten Abschnitt auf die konkrete Anwendung der Profilsteuerung in Walzwerken einzugehen.

### **3. Kontinuierliches Lernen - Beispiele aktueller Ansätze**

#### **Kurzzeitadaption durch Fehlerextrapolation**

Bei den folgenden Betrachtungen gehen wir von dem Szenario aus, daß ein neuronales Modell im Rahmen einer Prozeßführung zur Prognose des aktuellen Prozeßzustands eingesetzt wird. Dazu wird ein Eingangsvektor (Stellgrößen, Parameter etc.) an das Netz übergeben und dieses liefert gemäß der gelernten funktionalen Abbildung den Wert für eine oder mehrere Ausgangsgrößen als Prognose des Prozeßzustands. Diese Prognose wird zur Prozeßführung verwendet und nach einer bestimmten Zeit stehen Meßwerte der „tatsächlichen“ Ausgangsgrößen zur Verfügung. Diese Meßwerte können anschließend mit der Netzprognose verglichen und der resultierende Fehler kann im Rahmen eines kontinuierlichen Lernvorgangs zur Anpassung des neuronalen Modells verwendet werden.

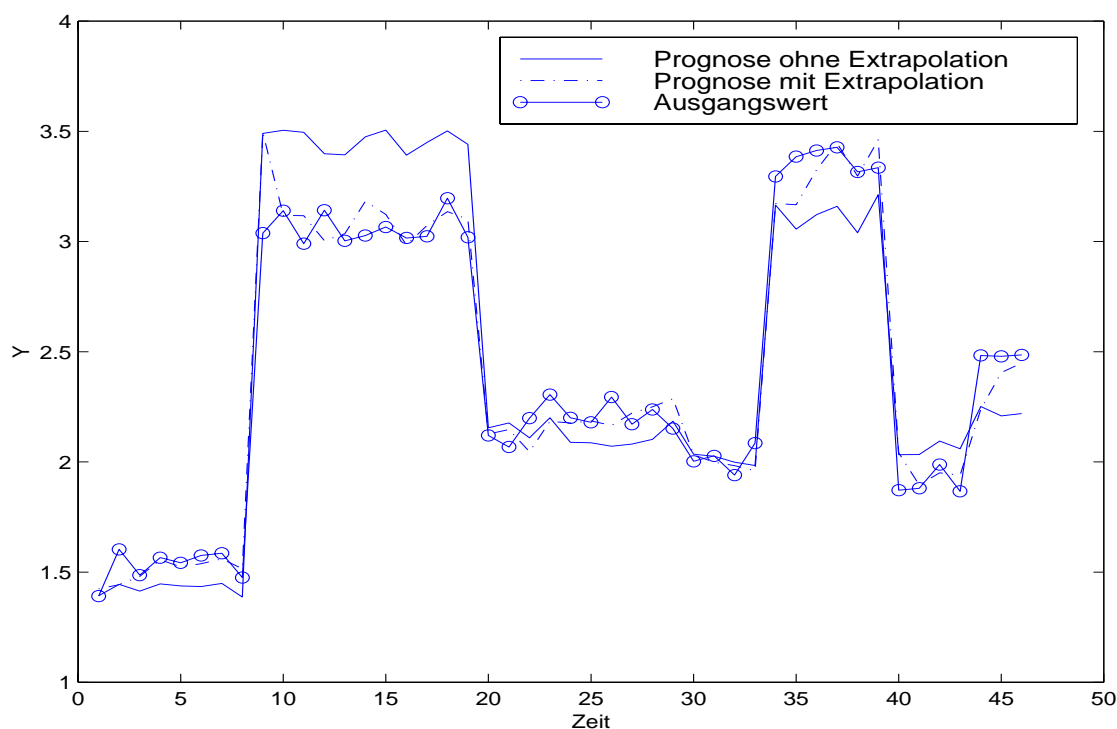
Zur Realisierung von unterschiedlichen Adaptionsgeschwindigkeiten (s. Punkt 1 im letzten Abschnitt) gibt es eine einfache Möglichkeit, zwei Verfahren zu kombinieren. Ein neuronales Modell liefert eine „Primärprognose“ und adaptiert relativ langsam, um die Nichtlinearitäten des Prozesses ausreichend modellieren zu können. Um trotzdem schnellen Änderungen des Prozesses folgen zu können, liefert ein zweites Verfahren eine Kurzzeitprognose, die darauf basiert, daß aufeinanderfolgende Fehler bei der Prognose statistisch korreliert sind. Eine solche Kurzzeitprognose kann beispielsweise durch eine „Fehlerextrapolation“ mit Hilfe sogenannter ARMA-Modelle nach Box-Jenkins erfolgen [6]. Über die Autokorrelationsfunktion und die partielle Autokorrelationsfunktion können geeignete Modelle aus den Daten geschätzt werden. Ein Autoregressionsmodell 2. Ordnung (AR(2)-Modell) nimmt beispielsweise an, daß der Fehler  $e(t) = p(t) - y(t)$  zwischen Prognose und tatsächlichem Ausgangswert (Meßwert) zum Zeitpunkt  $t$  durch

$$e(t) = \Phi_1 \cdot e(t-1) + \Phi_2 \cdot e(t-2) + f(t)$$

dargestellt werden kann, mit  $f(t)$  als weiterem Störterm. Der aktuelle Fehler hängt von den

letzten beiden Fehlern ab. Die Koeffizienten  $\Phi_1$  und  $\Phi_2$  können aus den Daten geschätzt werden. Für AR-Modelle lassen sich rekursive Gleichungen verwenden, wodurch die Berechnung der aktuellen Parameterschätzer sehr schnell wird.

Sind die Prognosen und (tatsächlichen) Ausgangswerte bis zum Zeitpunkt  $t - 1$  bekannt und liegt die Primärprognose  $p(t)$  vor, so ergibt sich die endgültige Voraussage zu  $\hat{y}(t) = p(t) - \hat{e}(t)$ , mit  $\hat{e}(t)$  als prognostiziertem Fehler. Die Verwendung der Fehlerextrapolation hat in allen bisher von uns betrachteten realen Datensätzen aus industriellen Prozessen die Prognosegüte gegenüber einzelnen neuronalen Modellen um etwa 10-15 Prozent gesteigert. Die Abbildung 4 zeigt das typische Bild bei Anwendung der Fehlerextrapolation. Während das primäre, langfristig adaptierende Prognoseverfahren deutliche Abweichungen zum Output aufweist, liegen die mittels Fehlerextrapolation korrigierten, kurzfristig angepaßten Prognosen sehr nahe an den tatsächlichen Werten.



**Bild 4.** Verbesserung der Prognosegüte durch Fehlerextrapolation mit einem AR(3)-Modell.

### Inkrementelles Clustern und Evaluieren (ICE)

Ein Ansatz, der die Punkte 2 und 3 des vorherigen Abschnitts adressiert, wird im folgenden kurz vorgestellt. Der ICE-Algorithmus ist in der Lage, seine Netztopologie sowie die optimale Anzahl der Modelle selbständig zu ermitteln und liefert bereits während der Initiallernphase plausible Ausgabewerte. Dieser Ansatz basiert auf einem Konzept konkurrierender Modelle, die ihre Einflußbereiche gegenseitig einschränken. Die Modelle selbst werden über eine Schicht von RBF-Neuronen aufgespannt, die sowohl in ihrer Anzahl und Position als auch in der Größe ihrer rezeptiven Felder variabel sind. Die RBF-Neuronen dienen zur Festlegung

der Cluster- bzw. Modellgrenzen und als Stützstellen für unterschiedliche Modelle, die linear oder auch beliebig nichtlinear sein können.

Der ICE-Algorithmus ist, ähnlich wie ART-Netze [7], ein inkrementelles Verfahren und baut die Netztopologie entsprechend den jeweiligen Anforderungen selbständig auf. Stark vereinfacht läßt sich das Verfahren nach Bild 5 beschreiben.

### **ICE-Algorithmus**

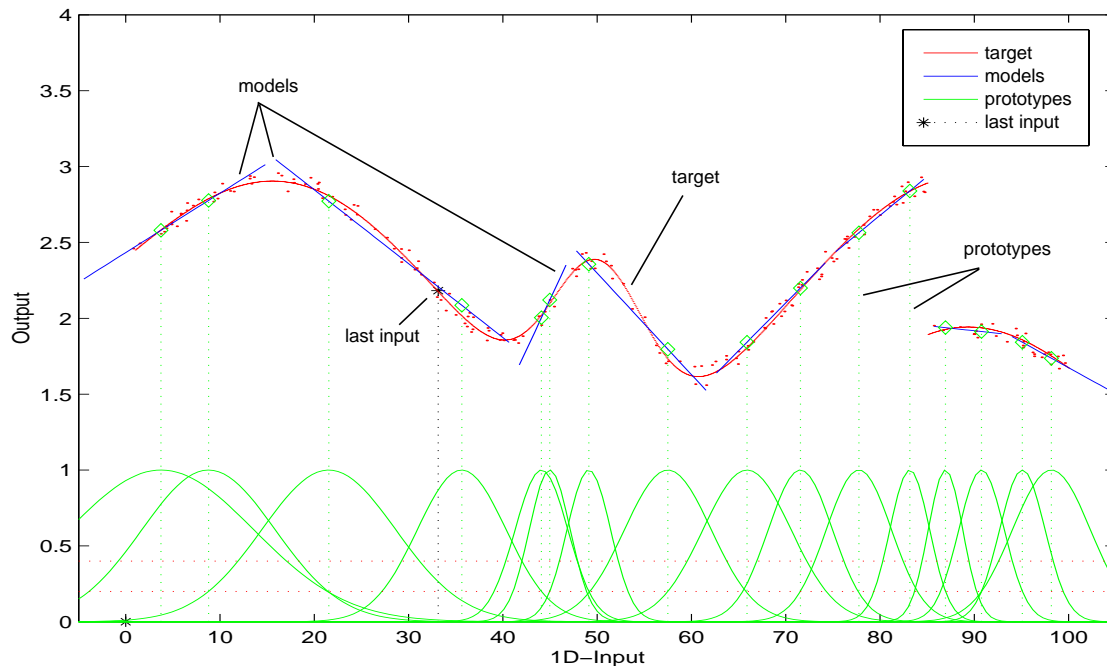
1. *Ermittle aktives Modell (=Modell mit höchster Aktivierung).*
  2. *Falls in aktivem Modell genügend Prototypen vorhanden sind und die Modellprognose ungenügend ist*  
*=> 2.1 erzeuge neues Modell.*  
*sonst*  
*=> 2.2 modifiziere aktives Modell.*
  3. *Berechne alle modifizierten Modelle neu.*
- 
- 2.1 *Erzeuge neues Modell:*
    - 2.1.1 *Splitte aktives Modell (Prototyp: potentielle Stützstelle für neues Modell).*
    - 2.1.2 *Erzeuge neuen Prototypen an der Position des aktuellen Eingabemusters.*
    - 2.1.3 *Beschränke die rezeptiven Felder aller Prototypen konkurrierender Modelle.*
    - 2.1.4 *Fasse alle Prototypen zu einem Modell zusammen, die nicht in Konkurrenz zueinander stehen.*
- 
- 2.2 *Modifiziere aktives Modell:*
    - 2.2.1 *Falls zu wenig Prototypen*  
*=> Erzeuge neuen Prototypen an der Position des aktuellen Eingabemusters.*  
*sonst*  
*=> Verschiebe aktiven Prototyp in Richtung des aktuellen Eingabemusters.*
    - 2.2.2 *Beschränke die rezeptiven Felder aller Prototypen konkurrierender Modelle.*

**Bild 5.** ICE-Algorithmus (vereinfachte Darstellung): Falls das aktive Modell den Forderungen entspricht, wird das Modell an das aktuelle Eingabemuster angepaßt. Ist dies nicht der Fall, wird ein neues Modell erzeugt.

Die Hauptunterschiede zu Verfahren wie Counterpropagation sind der sukzessive Aufbau der Modelle inkl. ihrer Prototypen, sowie die höhere Anzahl der Prototypen für jedes Modell. Durch die Prototypen wird zum einen eine feinere Auflösung der Modellgrenzen sowie eine bessere Repräsentation der Trainingsdaten für die einzelnen Modelle erreicht.

Das Begrenzen der rezeptiven Felder ist angelehnt an den DDA-Algorithmus [8]. Modelle dürfen an der Position von Prototypen konkurrierender Modelle lediglich eine maximale Aktivierung erreichen. Der Schwellwert wird von außen vorgegeben und ist einer der wenigen vorzugebenden Parameter. Im Gegensatz zum DDA besitzen beim ICE die Prototypen die Möglichkeit, ihre Position in Abhängigkeit der Daten zu variieren. Die Feinpositionierung erfolgt vergleichbar zu Kohonen-Netzen. Darüber hinaus dienen die Prototypen als Stützstellen für lokal lineare bzw. nichtlineare Modelle und nicht als Repräsentanten unterschiedlicher Klassen. Wieviele Modelle erforderlich sind, wird automatisch in Abhängigkeit der Modellprognosegüte ermittelt. Eine obere Grenze der Modellanzahl kann jedoch vorgegeben werden.

Die einzelnen Modelle werden aus Gründen der Stabilität mit der Zeit „eingefroren“. Wenn die Möglichkeit besteht, ein intensives Netztraining durchzuführen, kann, wie bei Simulated Annealing [9], die Modell- bzw. Prototypbeweglichkeit erhöht werden. Die Funktionsweise des Verfahrens wird in Bild 6 anhand des Beispiel einer nichtlinearen und un stetigen Funktionsapproximationsaufgabe erläutert.



**Bild 6.** Ergebnis einer Funktionsapproximation nach einmaliger Präsentation von 200 Trainingsmustern. Erzeugt wurden 8 lineare Modell mit je 2 RBF-Prototypen als Modellstützstelle.

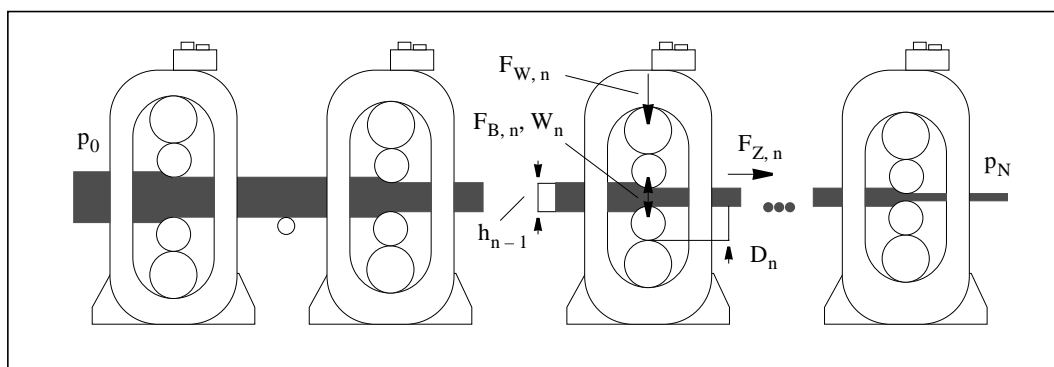
Bei diesem Versuch wurden dem Netz 200 Trainingmuster einer nichtlinearen, un stetigen Funktion einmalig präsentiert. Die Daten waren mit 10% Rauschen behaftet. Den einzelnen Modellen wurde eine maximale Abweichung von 0.2 (Absolutfehler) zugestanden. Wurde diese überschritten, wurde ein neues Modell erzeugt. Insgesamt wurden 8 Modelle generiert. Auch bei der Präsentation von 500, 1000, 5000 und 10000 Datenpunkten wurden max. 12 Modelle erzeugt (bei gleicher Parametereinstellung). Da das Erzeugen der Modelle nur von der Prognosegüte der vorhandenen Modelle abhängt, ist auch die Modellanzahl zum größten Teil vom genannten Schwellwert abhängig.

Die weiteren Arbeiten sollen es ermöglichen, nicht nur verschiedene Modelle für verschiedene Raumbereiche zu generieren, sondern auch unterschiedliche Modelle für den gleichen Raumbereich zu erzeugen und zu verwalten. Die Nutzung dieser Modelle soll in Anlehnung an die Idee des Case-Based-Reasoning (CBR) [10] erfolgen, mit dem Unterschied, daß nicht nach der ähnlichsten Situation in der Vergangenheit gesucht wird, sondern nach dem Modell, welches am besten zur aktuellen Situation paßt. Anstatt also ein einzelnes Modell kontinuierlich an veränderte (und ggf. wiederkehrende) Situationen anzupassen, sollen dann verschiedene Modelle (ggf. auch als Expertenwissen) nebeneinander existieren, die z.B. frühere Anlagenzustände widerspiegeln und auf die bei Bedarf unverzüglich zurückgegriffen werden kann.

#### 4. Industrielle Anwendung: Profilsteuerung in Walzwerken

Eine der prominentesten industriellen Anwendungen Neuronaler Netze liegt im Bereich der Walzwerksautomatisierung [11], [12]. Neuronale Netze werden dort als ergänzende Modelle zur Prozeßführung eingesetzt. Die zentrale Aufgabe der Prozeßführung einer Walzstraße ist die möglichst genaue Ermittlung der Einstellung der Anlage für das jeweils nächste Band vor dessen Einlauf in die Straße. Diese Einstellungen werden als Führungsgrößen an die Basisautomatisierung übermittelt, die die Regelung auf der untersten Ebene übernimmt. Die ausgeprägte Nichtlinearität und Zeitvarianz („Tagesform“) des Walzprozesses erfordert ein kontinuierliches Lernen der neuronalen Modelle mit all den Problemen, die in den letzten Abschnitten diskutiert worden sind.

Neben der Breite und der Dicke ist das Profil ein wichtiges Gütekriterium für die Geometrie des Walzgutes nach Verlassen der Fertigstraße. Im einfachsten Fall ist das Profil definiert als der Dickenunterschied zwischen der Mitte und den Rändern eines Bandes. Das Ziel ist, das Profil während des Warmwalzens so klein wie möglich zu halten, da es in späteren Kaltwalzprozessen nicht mehr beeinflußt werden kann. Das Profil des Walzgutes nach dem Durchlaufen eines Gerüsts ergibt sich als Funktion der Eigenschaften des Bandes beim Eintritt in das Gerüst und dem aktuellen Zustand und den Einstellungen des Gerüsts. Bild 7 zeigt den schematischen Aufbau einer Fertigstraße mit einigen Einflußgrößen auf das Profil. Das dem Stahl aufgeprägte Walzspaltprofil resultiert aus einer mechanischen und thermischen Walzenverformung und spiegelt die Prozeßvorgang des Walzvorgangs wider. Die thermische Walzenverformung (thermischer Crown) hängt z.B. von den Temperaturen, Breiten und Pausenzeiten einer ganzen Reihe aufeinanderfolgender Bänder ab.

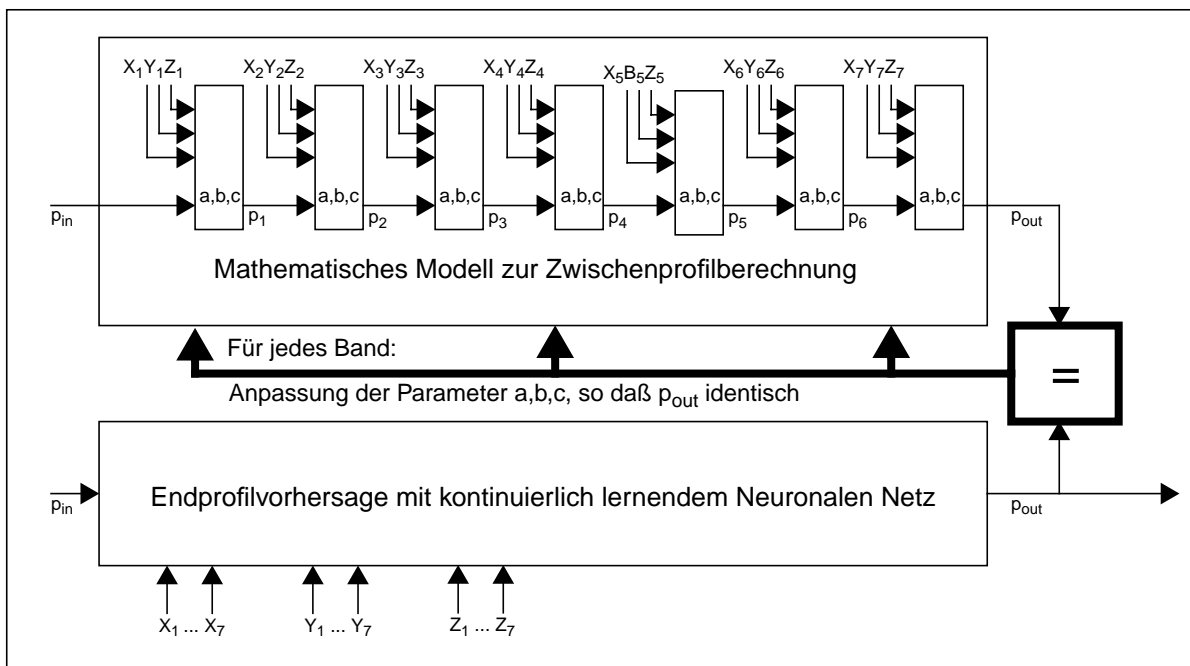


**Bild 7.** Schematische Darstellung einer Fertigstraße Jedes Gerüst  $n$  ändert das einlaufende Profil in Abhängigkeit von Parametern wie Banddicke  $h$ , Walzkraft  $F_W$ , Biegekraft  $F_B$ , Zugkraft  $F_Z$ , Walzspaltprofil  $W$  und Walzendurchmesser  $D$ .

Die existierenden mathematischen Modelle, beispielsweise zur Berechnung des Walzspaltprofils, beruhen zwar auf der Modellierung der physikalischen Zusammenhänge, jedoch sind in der Praxis viele idealisierte Voraussetzungen für die Gültigkeit der Modelle nicht erfüllt. Dies führt zu der Verwendung von zusätzlichen, heuristischen Modellen mit einer Vielzahl unbekannter Parameter, deren optimale Einstellung gefunden werden muß. Ein besonderes Problem liegt dabei in der Struktur der Profilmodelle. Während des gesamten Walzprozesses ändert sich das Profil des einlaufenden Stahlbandes von Gerüst zu Gerüst. Erst am Ende der Walzstraße kann es gemessen werden und mit den Vorhersagen des Modells verglichen wer-

den. Das Profil des Bandes zwischen den Gerüsten ist aus technischen Gründen nicht direkt meßbar, jedoch benötigt man für die Stichplanberechnung eine Prognose für den Verlauf des Bandprofils über der gesamten Walzstraße. Dieser Profilverlauf kann durch das n-fache Hintereinanderschalten eines mathematischen Gerüstmodells erzeugt werden. Es hat sich jedoch gezeigt, daß kontinuierlich lernende Neuronale Netze in der Lage sind, das Endprofil insbesondere bei Produktumstellungen um bis zu 20% genauer vorherzusagen. Da das Neuronale Netz wiederum keine Zwischenprofile erzeugt, werden die Vorteile von mathematischem Modell und Neuronalem Netz in einem Hybridsystem verbunden.

Bild 8 zeigt das Schema dieses Hybridsystems, bei dem das mathematische Modell zur Zwischenprofilprognose und das Neuronale Netz zur Endprofilprognose parallel arbeiten. Da das Neuronale Netz das Endprofil in der Regel genauer prognostiziert, werden die Parameter des mathematischen Modells für jedes Band so modifiziert, daß die Endergebnisse beider Modelle übereinstimmen, wodurch sich wiederum genauere Zwischenprofilwerte ergeben.



**Bild 8.** Hybrides Modell zur Profilvorhersage bei einer 7-gerüstigen Walzstraße. Das Gerüstmodell berechnet aus einlaufendem Profil  $p_{in}$  und den Gerüstesteinstellungen  $X, Y, Z$  die Zwischenprofile  $p_n$  und das Endprofil  $p_{out}$ . Das Neuronale Netz berechnet ebenfalls das Endprofil  $p_{out}$ , allerdings in der Regel genauer. Daher werden die Parameter  $a, b, c$  des Gerüstmodells so adjustiert, daß es ein möglichst ähnliches Endprofil wie das NN erzeugt.

Diese Form der Parallelschaltung von Modellen hat insbesondere auch Vorteile bezüglich der Zuverlässigkeit und Sicherheit der Verfahren im industriellen Einsatz. Das mathematische Modell ist robust und kann bei Bedarf ohne die neuronale Optimierung verwendet werden, auch wenn dies zu suboptimalen Ergebnissen führt. Erst wenn das Netz durch ausreichendes Training verlässlich bessere Werte liefert, wird sein Ausgangswert zur Optimierung der Parameter des mathematischen Modells herangezogen. Dieses Hybridmodell wurde im Rahmen des BMBF Forschungsprojektes AENEAS in Zusammenarbeit mit der SIEMENS AG entwickelt und wird z. Zt. in einer Anlage implementiert.

## 5. Ausblick

Neuronale Netze werden in der Prozeßautomatisierung oft dann eingesetzt, wenn das Verbesserungspotential konventioneller Verfahren erschöpft ist, z.B. bei den hier beschriebenen nichtlinearen und zeitvarianten Problemstellungen. Dabei darf nicht übersehen werden, daß solche Anforderungen auch an die Grenzen des gegenwärtigen Stands der Technik der Neuronalen Netze stoßen. Wir haben in diesem Beitrag die Anforderungen speziell im Hinblick auf das kontinuierliche Lernen Neuronaler Netze diskutiert und beispielhafte Lösungsansätze aufgezeigt. Dabei sind die zahlreichen, noch ungelösten Probleme Gegenstand intensiver Forschung. Parallel zu diesen Arbeiten bewähren sich Neuronale Netze mittlerweile in vielen industriellen Anwendungen im täglichen Einsatz, insbesondere in der Walzwerksautomatisierung. Hier verdrängen Neuronale Netze nicht die konventionellen, mathematischen Modelle, sondern ergänzen sie vielmehr als spezielle Komponenten eines hybriden Systems dort wo es sinnvoll und nützlich ist.

## Literatur

- [1] White, D. & Sofge, D. (Eds.), (1992). „Handbook of Intelligent Control - Neural, Fuzzy, and Adaptive Approaches“, Van Nostrand Reinhold, New York.
- [2] McCloskey, M. & Cohen, N. (1989). „Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem“. *The Psychology of Learning and Motivation*, 24, pp. 109-165.
- [3] Isermann, R. (1992). „Identifikation dynamischer Systeme 1“. Berlin, Heidelberg, New York, Springer Verlag, S. 235-248.
- [4] Poggio, T., Girosi, F. (1990). „Regularization Algorithms for Learning That Are Equivalent to Multilayer Networks“. *Science* 247, pp. 978-982.
- [5] Hecht-Nielsen, R. (1987). „Counterpropagation networks“. *Proc. of the IEEE First International Conference on Neural Networks*. San Diego, CA, Vol. 2, pp. 19-32.
- [6] Box G.E.P., Jenkins G.M., (1970). „Time series analysis, forecasting and control“. , Holden Day, San Francisco.
- [7] Carpenter, G., Grossberg, S. (1987). „A massively parallel architecture for a selforganizing neural pattern recognition machine“. *Computer Vision, Graphics and Image Processing*, pp. 54-115.
- [8] Berthold, M., Diamond J. (1994). „Boosting the Performance of RBF Networks with Dynamic Decay Adjustment“. *Advances in Neural Information Processing Systems 7*, Morgan Kaufmann Publishers, San Mateo, CA.
- [9] Kirkpatrick, S., Gelatt, C. & Vecchi, M. (1983). „Optimization by simulated annealing“, *Science*, Vol. 220, pp. 671-680.
- [10] Kolodner, J. (1993), „Case-Based Reasoning“, Morgan Kaufmann Publishers, San Mateo, CA.
- [11] Lindhoff, D., Sörgel, G., Gramckow, O. & Klode, K.-D. (1994). „Erfahrungen beim Einsatz Neuronaler Netze in der Walzwerksautomatisierung“. *Stahl und Eisen*, 114, Heft 4, S. 49-53+208.
- [12] Martinetz, T., Gramckow, O., Protzel, P. & Sörgel, G. (1996). „Neuronale Netze zur Steuerung von Walzstraßen“, *atp - Automatisierungstechnische Praxis*, 38, Heft 10, S. 28-42.