Discussion
Paper

Discussion Paper No. 95-26

Neuronale Netze in der Ökonometrie

Die Entmythologisierung ihrer Anwendungen

von Ulrich Anders

ZEW
Zentrum für Europäische
Wirtschaftsforschung Gmb

International Finance Serie

Neuronale Netzwerke in der Ökonometrie
Die Entmythologisierung ihrer Anwendung

Ulrich Anders

Zentrum für Europäische
Wirtschaftsforschung (ZEW)

Kaiserring 14–16
Postfach 10 34 43
68034 Mannheim

Tel: 0621/1235-141
Fax: 0621/1235-223
Email: anders@zew.de

Abstrakt

Die Anwendung neuronaler Netzwerke ist ein extrem kontrovers diskutiertes Thema. In dem einen Extrem erhoffen die Verfechter neuronaler Netzwerke, daß diese als Produkt der 'künstlichen Intelligenz' in der Lage sind, Aufgaben zu lösen, die bislang als unlösbar oder nur schwer lösbar galten. In dem anderen Extrem sehen die Kritiker neuronaler Netzwerke deren Vorteile hinter dem *black box*–Ansatz verschwinden und behaupten, neuronale Netzwerke können nichts, was sich nicht schon mit den bekannten statistischen Methoden in Griff bekommen ließe. Unrecht haben beide Seiten. Tatsächlich sind neuronale Netzwerke nichts anderes als eine neue Klasse von statistischen Verfahren, eine Erkenntnis, die mittlerweile auch in der jüngeren wissenschaftlichen Literatur Platz greift. In der vorliegenden Arbeit werden neuronale Netzwerke zu den herkömmlichen Verfahren in Beziehung gesetzt. Darüber hinaus wird aufgezeigt, wie sich neuronale Netzwerke mit Hilfe von statistischen Methoden objektiv analysieren lassen. Die Anwendung statistischer Methoden auf neuronale Netzwerke soll mit 'Neurometrie' betitelt werden. Im Rahmen diese neurometrischen Ansatzes wird eine Vorgehensweise für die Spezifikation einer Netzwerkarchitektur vorgestellt, die bei der ökonometrische Modellierung eines funktionalen Zusammenhangs mit Hilfe von neuronalen Netzwerken zum Einsatz kommen sollte.

Inhaltsverzeichnis

1 Einleitung

Die Entwicklung neuronaler Netzwerke entsprang dem Versuch, die Leistungsfähigkeit biologischer Nervensysteme auszunutzen und die Schranken der sequentiell arbeitenden von-Neumann-Rechner zu durchbrechen. In Anlehnung an das biologische Vorbild Gehirn wurden dazu eine Vielzahl von einfachen Recheneinheiten miteinander verbunden, in der Hoffnung, so komplexe Phänomene wie 'Intelligenz' oder 'Lernfähigkeit' nachbilden zu können (Sarle, 1994). Solche Attribute wie 'Intelligenz' und 'Lernfähigkeit' sind mitverantwortlich für die kontroverse Auseinandersetzung, die um das Thema 'neuronale Netzwerke' geführt wird. Denn wirklich intelligentes Verhalten — wie z.B. das Erlernen und selbständige Abstrahieren eines Zusammenhangs — können neuronale Netzwerke schon allein wegen der dazu benötigten Komplexität bei weitem nicht erreichen. Selbst große Netzwerke bestehen zur Zeit aus nicht mehr als ein paar tausend Recheneinheiten, das menschliche Gehirn besitzt dagegen etwa 10^{12} Neuronen.

Die Bezeichnung 'neuronales Netzwerk' ist nicht eindeutig festgelegt. Zur Zeit wird sie in verschiedenen Anwendungsgebieten benutzt:

- als Simulationsmodell für die Informationsverarbeitung in biologischen Nervensystemen;
- als signalverarbeitendes Echtzeitsystem oder als Steuerungseinheit von Maschinen;
- als Methode zur Approximation funktionaler Zusammenhänge.

Im ökonometrischen Kontext der vorliegenden Arbeit werden ausschließlich neuronale Netzwerke betrachtet, die sich für die Approximation funktionaler Zusammenhänge eignen. Denn nur diese lassen sich für die Modellierung ökonomischer Zusammenhänge einsetzen.

Neuronale Netzwerke können vereinfachend als Baukastensysteme, deren Bausteine sich auf fast beliebige Art und Weise miteinander kombinieren lassen, interpretiert werden.

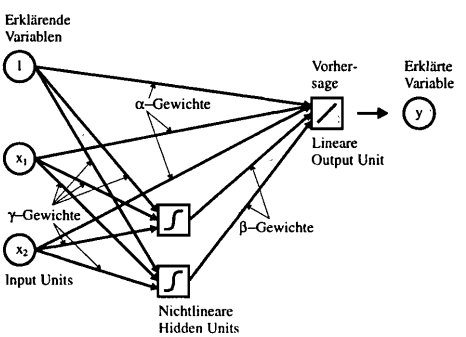

Abbildung 1: Standardnetzwerk mit drei Schichten.

Die Bausteine werden in Anlehnung an Sarle (1994) folgendermaßen visualisiert:

- Kreise entsprechen den beobachteten Variablen und beinhalten den entsprechenden Variablennamen.
- Rechtecke repräsentieren den errechneten Wert einer Funktion, die von einer oder mehreren Variablen abhängig ist. Die Symbole innerhalb des Rechtecks stellen die sogenannten Aktivierungsfunktionen dar, die alle in eine *unit* eingehenden Signale in ein Ausgangssignal transformiert. Den meisten Aktivierungsfunktionen ist ein sogenannter *Schwellwert-* bzw. affiner Parameter zugeordnet.
- Dünne Pfeile charakterisieren die Abhängigkeiten im Netzwerk. Der Wert am Beginn eines Pfeils geht in die Berechnung der Funktion am Ende des Pfeiles ein. Jedem Pfeil ist ein Gewicht bzw. Parameter zugeordnet, der im Trainingsprozeß des Netzwerks bestimmt wird. Die Gewichte der linearen Verbindungen werden α–Gewichte, die Gewichte zwischen *output* und *hidden units* β–Gewichte und die Gewichte zwischen *hidden* und *input units* γ–Gewichte genannt.
- Ein dicker Pfeil bedeutet, daß die Vorhersagen des Netzwerks an die tatsächlichen Beobachtungen der zu erklärenden Variable angepaßt werden sollen.

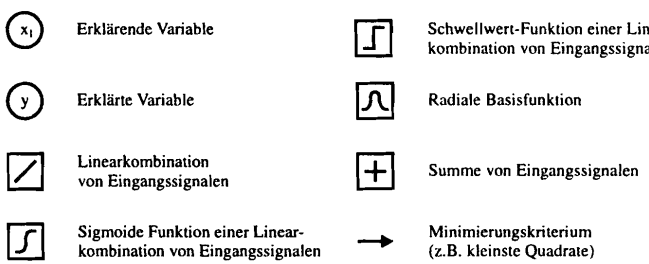

Abbildung 2: Die Bausteine neuronaler Netzwerke.

Ein Beispiel für ein einfaches neuronales Netzwerk, das die wesentlichen Bausteine verwendet, ist in Abbildung 1 dargestellt. Die einzelnen Bausteine sind in Abbildung 2 aufgelistet. Netzwerke dieser vorwärtsgerichteten Art finden in den meisten Untersuchungen Anwendung, und daher sollen in der vorliegenden Arbeit lediglich solche Netzwerktypen betrachtet werden. Das wesentliche Charakteristikum dieser Netzwerktypen besteht darin, daß sie auf einen im voraus bekannten Zieloutput hintrainiert werden (sog. *supervised learning*). Im Gegensatz dazu existieren Netzwerktypen, die darauf ausgerichtet sind, im voraus unbekannte Merkmale aus den zu analysierenden Daten (ähnlich der Faktorenanalyse) zu extrahieren (sog. *unsupervised learning*).

In den Rechtecken sind die in Netzwerken verwendeten Transformationsfunktionen dargestellt. Bei der Implementierung neuronaler Netzwerke verwendet man für die lineare Transformationsfunktion die identische Funktion $g(x) = x$. Die Schwellwert-Funktion

wird mit einer Signum–Funktion $g(x) = \text{sgn}(x)$ implementiert und bei der radialen Basisfunktion verwendet man in den meisten Fällen eine Gauss'sche Glockenkurve $g(x) = e^{-x/2}$. Als sigmoide Transformationsfunktion benutzt man die hyperbolische Tangensfunktion $g(x) = \tanh(x)$ bzw. die logistische Funktion $g(x) = (1 + e^{-x})^{-1} = (\tanh(x/2) + 1)/2$. Die tanh-Funktion bietet gegenüber der logistischen Funktion zwei Vorteile: erstens ist ihre Ableitung mit $\tanh(x)' = 1 - \tanh^2(x)$ leichter zu berechnen und zweitens ist sie symmetrisch zum Ursprung, so daß $\tanh(x) = -\tanh(-x)$ gilt.

Jedes neuronale Netzwerk läßt sich durch eine Funktion der erklärenden Variablen $X = [x_0, x_1, \ldots, x_I]$ und der Netzwerkgewichte $w = (\alpha', \beta', \gamma')'$ beschreiben. x_0 stellt ist konstant und wird für alle Beobachtungen als $x_0 \equiv 1$ definiert. I bezeichnet die Anzahl der nichtkonstanten erklärenden Variablen, H die Anzahl der verwendeten *hidden units*. Für das in Abbildung 1 dargestellt Netzwerk ergibt sich damit die folgende funktionale Form $f(X, w)$:

$$f(X, w) = X\alpha + \sum_{h=I+1}^{I+H} \beta_h g(\sum_{i=0}^{I} \gamma_{hi} x_i) \tag{1}$$

2 Neuronale Netzwerke in der Ökonometrie

Ziel jeder ökonometrischen Untersuchung ist die Erklärung oder Prognose von ökonomischen Variablen. Die Erklärung geschieht dabei unter Verwendung anderer ökonomischer Variablen, aber auch unter Zuhilfenahme externen Wissens, sofern dieses zur Verfügung steht.

Im einfachsten Fall soll die (funktionale) Beziehung einer gegebenen Variable y zu anderen Variablen $X = [x_0, x_1, \ldots, x_I]$ aufgedeckt werden, d.h.

$$y = F(X) + \varepsilon. \tag{2}$$

ε stellt einen Störterm dar, der unabhängig von X verteilt sein soll, und es wird $E[\varepsilon|X] = 0$ angenommen. Die Funktion F bezeichnet man als Regression von y auf X, denn es gilt: $E[y|X] = F(X)$.

In der Ökonometrie versucht man nun, die wahre Funktion $F(X)$ mit einer Funktion $f(X, \theta)$ zu approximieren. In den meisten Fällen nimmt man dazu an, daß die funktionale Form von f der Form der wahren Funktion F entspricht, so daß dann lediglich noch die Parameterwerte θ, die den Verlauf der Funktion f spezifizieren, mit Hilfe der Daten geschätzt werden müssen. Alternativ dazu ist es jedoch möglich, die wahre Funktion F auch ohne jede Annahme über ihre funktionale Form direkt mittels der Daten anzunähern. Die Funktion f ist dann das Resultat der Annäherung.

Um nun die Anwendbarkeit neuronaler Netzwerke für die Problemstellung der Funktionsapproximation zu erkennen, betrachte man noch einmal die Abbildung 1. Die unabhängigen Variablen X werden als Input für das neuronale Netzwerk verwendet, die abhängige

Variable y stellt den Zielwert des Netzwerks dar. Hornik/Stinchcombe/White (1989) haben bewiesen, daß ein Netzwerk dieser Form mit nur einem *hidden layer* in der Lage ist, jede beliebige (meßbare) Funktion mit jedem gewünschten Grad an Genauigkeit anzunähern, vorausgesetzt das Netzwerk besitzt eine hinreichend große Anzahl von *hidden units*. Insbesondere können neuronale Netzwerke also auch nichtlineare Funktionen ohne Schwierigkeiten abbilden. Mit Hilfe neuronaler Netzwerke lassen sich demgemäß beliebige Funktionen $F(X) = E[y|X]$ approximieren. Im Gegensatz zu vielen anderen statistischen Verfahren müssen bei der Anwendung neuronaler Netzwerke jedoch keine expliziten Annahmen über die funktionale Form der wahren Funktion F gemacht werden.

2.1 Parametrische und nichtparametrische Verfahren

Neuronale Netzwerke sind nichts anderes als statistische Verfahren zur Approximation von Regressionsfunktionen. Sie lassen sich daher nahtlos in den Kontext der herkömmlichen statistischen Verfahren eingliedern. In der Statistik und der Ökonometrie unterscheidet man zwischen drei Klassen von Verfahren:[1] den parametrischen, den semiparametrischen und den nichtparametrischen Verfahren. Letztere gehören zu den jüngeren Methoden in der Ökonometrie[2] und haben in den letzten zehn Jahren zunehmend an Aufmerksamkeit gewonnen. Die Verfahren aller drei Klassen sind für denselben Zweck geeignet, nämlich einen (ökonomischen) Zusammenhang zu modellieren. Sie unterscheiden sich jedoch darin, daß sie zur Modellierung von Zusammenhängen unterschiedlich starke Annahmen erfordern.

- Bei den parametrischen Verfahren unterstellt man dem zu modellierenden Zusammenhang eine bestimmte funktionale Form (z.B. $f(X,\theta) = X\theta$ oder $f(X,\theta) = \sum_k \theta_k x^k$ etc.), in dem lediglich noch die Parameter θ der unterstellten Funktion zu bestimmen sind.

- Nichtparametrische Verfahren hingegen erlauben die Modellierung eines Zusammenhangs, ohne im voraus Annahmen über die funktionale Form dieses Zusammenhangs treffen zu müssen. Die Form des Zusammenhangs ergibt sich (in den meisten Fällen durch Glättung) allein aus den zur Verfügung stehenden Beobachtungen. Parameter, die zur Beschreibung des funktionalen Zusammenhangs benötigt werden, haben keine theoretisch fundierbare Bedeutung mehr.

- Semiparametrische Verfahren sind Mischformen von parametrischen und nichtparametrischen Verfahren.

Die Vor- und Nachteile der verschiedenen Verfahren liegen auf der Hand: parametrische Verfahren eignen sich immer dann besonders gut, wenn man entweder die zugrundeliegende Struktur eines Zusammenhangs kennt oder wenn man hinreichend belegte Vermutungen über dieselbe aussprechen kann. Nichtparametrische Modelle finden vor allen Dingen

[1] Vgl. Granger/Teräsvirta (1993).
[2] Vgl. Härdle (1990).

dort Anwendung, wo keine gesicherten Erkenntnisse über die Modellstruktur vorliegen oder gewonnen werden können.

Im Sinne der obigen Definition sind neuronale Netzwerke rein parametrische Verfahren, denn sie unterstellen dem zu modellierenden Zusammenhang eine bestimmte Struktur, in der lediglich noch geeignete Parameter gewählt werden müssen. Um dies zu erkennen, betrachte man das bekannte einfache lineare Regressionsmodell in Vektornotation[3]

$$y = X\alpha + \varepsilon. \tag{3}$$

mit den Annahmen

$$E[\varepsilon] = 0 \text{ und } E[\varepsilon\varepsilon'] = \sigma^2 I. \tag{4}$$

Dieses Regressionsmodell ließe sich alternativ zu OLS mittels eines neuronalen Netzwerks, das in Abbildung 3 dargestellt ist, schätzen. Dieses neuronale Netzwerk entspricht also dem einfachen linearen Regressionsmodell.

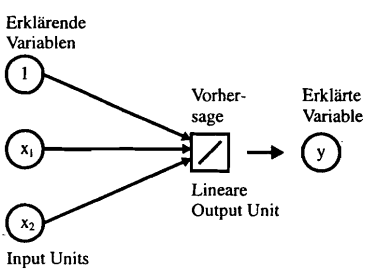

Abbildung 3: Das einfache lineare Regressionsmodell als neuronales Netz.

Die Architektur eines Netzwerks legt also eine funktionale Form fest und lediglich die Gewichte des Netzwerks müssen im sogenannten Trainingsprozeß bestimmt werden. Die sich ergebenden Gewichtswerte $\hat{w}$ entsprechen den geschätzten Parametern $\hat{\alpha}$ eines Regressionsmodells. Die Netzwerkfunktion f, deren Werte Vorhersagen $\hat{y}_t$ für die beobachteten Werte y_t der abhängigen Variable liefert, gestaltet sich entsprechend:

$$\hat{y}_t = f(X_t, \hat{w}) = X_t\hat{\alpha} \tag{5}$$

Neuronale Netzwerke sind also rein parametrische Verfahren. Unter gewissen Umständen können sie jedoch — wie im folgenden nach einem kurzen Exkurs über *Bias* und Varianz dargestellt werden soll — als nichtparametrische Verfahren aufgefaßt werden.

[3]Die Variablenvereinbarungen sind am Ende dieser Arbeit in der Nomenklatur getroffen.

Das bei parametrischen Verfahren am häufigsten verwendete Zielkriterium für die Bestimmung von Parametern θ einer Funktion f besteht in der Minimierung der durchschnittlichen quadratischen Abstände zwischen den tatsächlichen Werten der abhängigen Variablen y und den durch die Funktion $f(X, \hat{\theta})$ geschätzten Werten $\hat{y}$. Das ist der geschätzte *Mean Squared Error* (MSE) einer Regression:

$$\text{MSE} = \frac{1}{T}\sum_{t=1}^{T}(y_t - f(X_t, \hat{\theta}))^2. \tag{6}$$

Der durchschnittliche Fehler (MSE) jeder Modellschätzung $f(X, \hat{\theta})$ läßt sich in zwei Komponenten zerlegen: einen unsystematischen Fehler MSE_u und einen systematischen Fehler MSE_s, denn es gilt:

$$\begin{aligned}
\text{MSE} &= E[(y - f(X, \hat{\theta}))^2] \\
&= E[(y - F(X) + F(X) - f(X, \hat{\theta}))^2] \\
&= E[(y - F(X))^2] + E[(F(X) - f(X, \hat{\theta}))^2] + 2E[(y - F(X))(F(X) - f(X, \hat{\theta}))] \\
&= E[(y - F(X))^2] + E[(F(X) - f(X, \hat{\theta}))^2] + 0 \\
&= \text{MSE}_u + \text{MSE}_s
\end{aligned} \tag{7}$$

Der erwartete unsystematische Fehler MSE_u wird durch den Störterm $\varepsilon = y - F(X)$ erzeugt und läßt sich, da er rein zufälliger Natur ist, durch kein Modell approximieren. Er kann aber erheblich von den Residuen der Regression $u = y - f(X, \hat{\theta})$ abweichen, da sich die Residuen aus dem Störterm ε und dem Approximationsfehler $a(X, \hat{\theta}) = F(X) - f(X, \hat{\theta})$ der Regression zusammensetzen. Die Güte einer Regression läßt sich grundsätzlich nur mit dem systematischen Fehler messen, denn dieser gibt an, wie hoch die erwartete quadratische Abweichung zwischen der wahren Funktion F und ihrer Approximation f ist. Im günstigsten Fall ist die Abweichung zwischen diesen beiden Funktionen Null, so daß in diesem Fall $\text{MSE} = \text{MSE}_u$.

Interpretiert man nun die Funktion f als einen Funktionsschätzer für die Funktion F, dann läßt sich der systematische MSE_s der obigen Modellschätzung noch weiter aufteilen. Der MSE eines jeden Schätzers unterteilt sich nämlich in *Bias* und Varianz dieses Schätzers. Für einen Parameterschätzer $\hat{\theta}$ gilt beispielsweise:

$$\text{MSE}[\hat{\theta}] = \text{Bias}[\hat{\theta}]^2 + \text{Var}[\hat{\theta}] \tag{8}$$

Der *Bias* eines Schätzers ist als die Abweichung des Erwartungswertes des Schätzers von dem tatsächlichen Wert definiert. Die Varianz des Schätzers ist die durchschnittliche quadrierte Abweichung des Schätzers von seinem Erwartungswert. Für den *Bias* des Punktschätzers $\hat{\theta}$ gilt also beispielsweise

$$\text{Bias}[\hat{\theta}] = E[\hat{\theta}] - \theta \tag{9}$$

und für seine Varianz

$$\mathrm{Var}[\hat{\theta}] = \frac{1}{T}(\hat{\theta} - E[\hat{\theta}])^2. \tag{10}$$

In der obigen Regression ist der systematischen MSE_s die durchschnittliche Abweichung des Funktionsschätzers f von dem wahren Funktionsverlauf F. Es gilt also $\mathrm{MSE}_s = \mathrm{MSE}[f]$. Der Funktionsschätzer f läßt sich nun analog zu dem obigen Parameterschätzer in einen *Bias* und eine Varianz zerlegen:

$$\begin{aligned} \mathrm{MSE}[f] &= E[(F(X) - f(X,\hat{\theta}))^2] \\ &= \mathrm{Bias}[f(X,\hat{\theta})]^2 + \mathrm{Var}[f(X,\hat{\theta})] \end{aligned}$$

Der *Bias* von f gibt dabei an, wie weit der Funktionsverlauf von f im erwarteten Mittel von dem Funktionsverlauf von F entfernt ist. Er stellt somit die systematische Verzerrung der Approximation dar. Die Varianz von f gibt an, in welchem Bereich um den Funktionsverlauf von f herum die aus den Daten ableitbare Approximation der Funktion F auch hätte liegen können. Die folgende Abbildung 4 soll dies verdeutlichen.

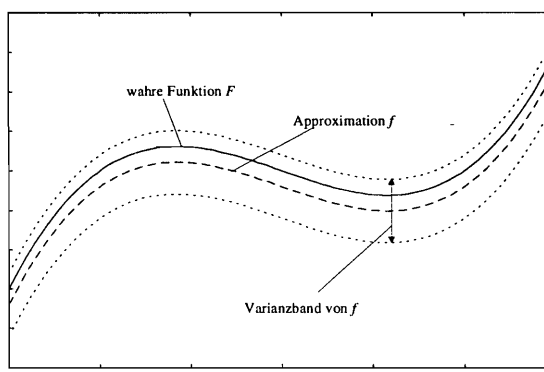

Abbildung 4: *Bias* und Standardabweichung eines Funktionsschätzers f.

Die Approximation der Funktion F durch die Funktion f in der Abbildung 4 ist offensichtlich verzerrt, da der Kurvenverlauf von f systematisch neben dem Kurvenverlauf von F liegt. Alle im Varianzband liegenden Funktionsverläufe sind aufgrund der gegebenen Daten für f möglich. Eine offensichtlich optimale Funktionsschätzung wäre unverzerrt im Erwartungswert und hätte keine Varianz. Die Kurven für f und F würden sich in diesem Fall also decken und Abweichungen wären aufgrund der Beobachtungen sowie der Modellierung nicht möglich.

Aus diesen Vorüberlegungen läßt sich nun ableiten, daß die Abweichung jeder Approximation f von der zu approximierenden Funktion F entweder durch eine Verzerrung im

Erwartungswert, durch die Varianz des Funktionsverlaufs oder durch eine Kombination von *Bias* und Varianz erzeugt werden. Mit den Annahmen über die Modellierung von F legt man implizit fest, ob letztendlich *Bias* oder Varianz für den realistischerweise zu erwartenden Approximationsfehler verantwortlich sind. In der empirischen Anwendung einen Fehler sowohl bezüglich der Varianz als auch bezüglich des *Bias* auszuschließen, ist unmöglich. Dieses Dilemma nennen Geman/Bienenstock (1992) das *Bias*-Varianz-Dilemma.

Grundsätzlich ist es so, daß fehlspezifizierte (parametrische) Modelle einen *Bias* verursachen; modellfreie Schätzungen sind hingegen in der Regel unverzerrt, leiden jedoch an einer hohen Varianz, die sich lediglich mit Hilfe ungewöhnlich großer Datenmengen reduzieren läßt, d.h. sie konvergieren in Bezug auf die Größe der Datenmenge nur extrem langsam zu einer konsistenten Schätzung[4]. Die einzige Möglichkeit die Varianz zu kontrollieren besteht deshalb in der Vorgabe einer Modellstruktur, worin sich das Dilemma deutlich macht: die wahre Modellstruktur ist nämlich häufig komplex und daher nur selten exakt zu identifizieren. Mit einer möglicherweise inkorrekten Annahme über die Modellstruktur erhält jedoch dann wiederum ein *Bias* Eingang in die Schätzung.

Wie sind nun neuronale Netzwerke in das Schema von parametrischen und nichtparametrischen Verfahren einzuordnen? Neuronale Netzwerke gehören — wie oben festgestellt — zu den parametrischen Verfahren. Beim Entwurf einer Netzwerkarchitektur werden jedoch in der Regel keine expliziten Annahmen über die funktionale Form des zu modellierenden Zusammenhangs getroffen. Mit der Festlegung einer bestimmten Netzwerkkomplexität trifft man dennoch die implizite Annahme, daß der zu modellierende Zusammenhang durch das verwendete Netzwerk approximiert werden kann. Soll auch diese Annahme fallengelassen werden, muß ein Netzwerk gewählt werden, daß jede beliebige Funktion approximieren kann, im Extremfall also ein Netzwerk, das unendlich viele *hidden units* besitzt. Damit entsprechen hinreichend dimensionierte Netzwerkmodelle nichtparametrischen Verfahren dahingehend, daß sie erstens unverzerrt sind und zweitens (wegen ihrer hohen Parameterzahl) unter einer großen Varianz der Modellschätzung leiden.

Zusammenfassend kann man sich neuronale Netzwerke auf einem Kontinuum vorstellen, an dessen einem Ende parametrische Verfahren, an dessen anderem Ende nichtparametrische Verfahren stehen (siehe Abbildung 5).

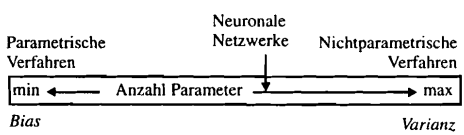

Abbildung 5: Kontinuum zwischen parametrischen und nichtparametrischen Verfahren.

In den Extremen ist ein Netzwerk ohne *hidden units* — wie in Gleichung (5) dargestellt — als ein rein parametrisches Verfahren, ein neuronales Netzwerk mit unendlich vielen *hidden*

[4]Ein Schätzer ist konsistent, wenn er asymptotisch unverzerrt ist und seine Varianz asymptotisch verschwindet.

Neuronales Netzwerk	Parametrisches Modell
Netzwerkarchitektur	Modellspezifikation
Ungeeignete Netzwerkarchitektur	Über- bzw. Unterparametrisierung
Aktivierungsfunktion	Transformationsfunktion (z.B. logistische)
Inputs	Erklärende (unabhängige) Variablen
Output	Erklärte (abhängige) Variable
Gewichte	Parameter
Training	Parameterschätzung
Konvergenz	*in sample*–Qualität
Generalisierung	*out of sample*–Qualität

Tabelle 1: Vergleich der Terminologien

units dagegen als ein nichtparametrisches Verfahren zu interpretieren. Mit der Wahl einer bestimmten Netzwerkkomplexität wird ein Punkt auf dem Kontinuum angenommen, ein Kompromiß zwischen *Bias* und Varianz. Ein Netzwerk niedriger Dimension ist mit großer Wahrscheinlichkeit verzerrt, da nur eine geringe Anzahl von Funktionen approximiert werden kann, ein Netzwerk von großer Dimension hat einen geringen *Bias*, leidet dafür jedoch an einer hohen Varianz des Modells. Ziel jedes Modellbildungsprozesses mit neuronalen Netzwerken muß sein, eine Architektur zu finden, die den besten Kompromiß zwischen *Bias* und Varianz des Modells eingeht. Optimal ist ein neuronales Netzwerk genau dann gewählt, wenn es gerade groß genug ist, die wahre Funktion F zu approximieren. Dann nämlich ist der erwartete *Bias* Null und die erwartete Varianz bleibt wegen der minimal möglichen Parameterzahl so gering wie möglich.

2.2 Glossar

Die im bisherigen Verlauf dieser Arbeit dargestellten Analogien zwischen neuronalen Netzwerken und anderen statistischen Verfahren sollen an dieser Stelle kurz zusammengefaßt werden. Dabei sticht die Verwandtschaft zwischen neuronalen Netzwerken und statistischen Verfahren deutlich hervor. Es zeigt sich einmal mehr, daß neuronale Netzwerke nichts anderes sind als eine neue Klasse von statistischen Verfahren, die lediglich mit einer unterschiedlichen Terminologie ausgestattet sind.

Vor dem Hintergrund der offensichtlichen Verwandtschaft von neuronalen Netzwerken und statistischen Verfahren läßt sich eine jeweils exklusive Betrachtung nicht mehr rechtfertigen. Im Gegenteil, White (1992, S. 123) schreibt:

> *„[...] neural network models provide a novel, elegant, extremely rich class of mathematical tools for data analysis. Application of neural network models to new and existing datasets holds the potential for fundamental advances in empirical understanding across a broad spectrum of the sciences. To realize these advances, statistics and neural network modelling must work together, hand in hand.“*

2.3 Neuronale Netzwerke und statistische Modelle

In den letzten Abschnitten ist deutlich geworden, daß neuronale Netzwerke nichts anderes sind als Regressionsmodelle. Um diese Feststellung etwas anschaulicher zu gestalten, sollen im folgenden einige Regressionsmodelle in Form von Netzwerkarchitekturen dargestellt werden.[5] Dabei wird u.a. die große Flexibiliät neuronaler Netzwerke deutlich werden, selbst die verschiedensten Modelltypen mittels einer vereinheitlichten Terminologie zu implementieren. In Abbildung 3 wurde bereits das einfache lineare Regressionsmodell als neuronales Netzwerk dargestellt. Abbildung 1 zeigt die Umsetzung eines nichtlinearen Regressionsmodells mittels eines Netzwerks.

Ein vorwärtsgerichtetes Netzwerk, das — wie in Abbildung 6 dargestellt — eine logistische Transformationsfunktion für die *output unit* besitzt, ist ein logistisches Regressionsmodell (Hosmer/Lemeshow, 1989).

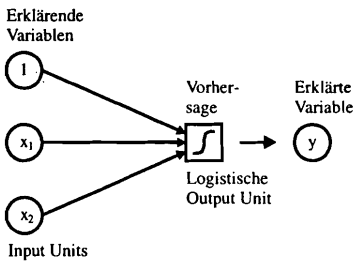

Abbildung 6: Logistisches Regressionsmodell als neuronales Netz.

Ein Netzwerk mit einer Schwellwertfunktion als *output unit* läßt sich als Diskriminanzfunktion interpretieren. In einem Netzwerk ohne *hidden units* hat die diskriminierende Funktion dabei einen linearen Verlauf, in einem Netzwerk, das *hidden units* verwendet (Abbildung 7), kann sie einen beliebigen nichtlinearen (z.B. quadratischen) Verlauf annehmen (Flury et al., 1983). Sollen die Beobachtungen in mehr als nur zwei Klassen aufgeteilt werden, wählt man üblicherweise ein Netzwerk mit entsprechend vielen logistischen *output units*. Der Wert einer *output unit* sollte Eins sein, wenn die Beobachtung in der entsprechenden Klasse liegt und Null anderenfalls. Die Werte, die eine *output unit* tatsächlich liefert (üblicherweise kleiner als Eins), lassen sich dann als Wahrscheinlichkeiten auffassen, mit der eine Beobachtung der entsprechenden Klasse angehört. Das für diese Modellierung beste Zielkriterium ist die *Cross Entropy*, die sich aus der *Loglikelihood*-Funktion einer Bernoulli-Verteilung der Variablen y ergibt.[6]

[5] Vgl. Sarle (1994).
[6] Vgl. Michie/Spiegelhalter/Taylor (1994), S. 89.

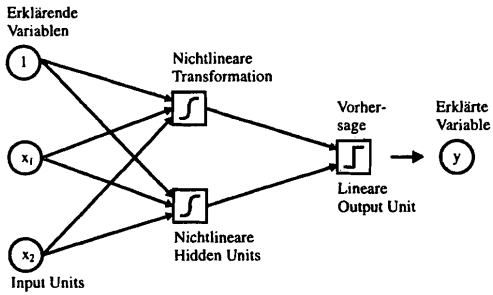

Abbildung 7: Nichtlineare Diskriminanzanalyse als neuronales Netz.

Wie sich aus der Darstellung dieser beiden Modelle bereits erkennen läßt, sind viele parametrische Regressionsmodelle auch als Netzwerk darstellbar. Es ist nun mit neuronalen Netzwerken leicht möglich, bestehende Regressionsmodelle in ein Netzwerk einzubetten und um eine nichtlineare Komponente zu erweitern. Das in Abbildung 8 dargestellten Netzwerk stellt die Erweiterung des einfachen linearen Regressionsmodells um einen nichtlinearen Teil dar. Das Netzwerk kann mit den Parametern einer OLS-Schätzung initialisiert werden und ist anschließend in der Lage, auch nichtlineare Zusammenhänge abzubilden, sofern diese vorhanden sind. Ein Netzwerk dieser Art ist Grundlage des von White (1989) entwickelten *Neural Network*-Test, der der Entdeckung von vernachlässigten Nichtlineritäten im Erwartungswert eines Modells dient.

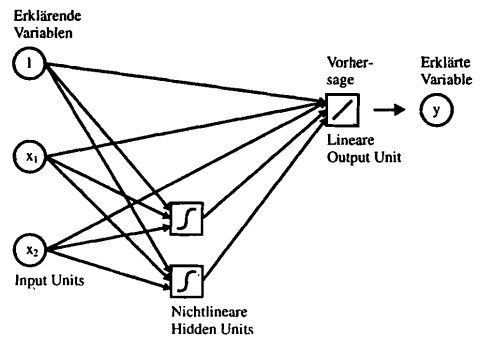

Abbildung 8: In einem neuronalen Netzwerk eingebettetes lineares Regressionsmodell.

Eines der jüngeren multiplen nichtparametrischen Verfahren heißt *Projection Pursuit*-Regression (Friedman/Stuetzle, 1981). Bereits Kuan und White (1994) haben die Ähnlichkeit zwischen neuronalen Netzwerken und diesem Verfahren festgestellt. *Projection Pursuit*-Regression modelliert den Regressionsverlauf als Summe von m glättenden Funktionen g, deren jeweiliger Verlauf von unterschiedlichen Linearkombinationen $X\alpha_m$ der erklärenden Variablen abhängig ist. Die Regression ergibt sich also durch:

$$f_{\mathrm{PP}}(X,\alpha) = \sum_{m=1}^{M} g_{\alpha_m}(X\alpha_m). \tag{11}$$

Die Analogie zur Vorgehensweise neuronaler Netzwerke ist unverkennbar. Der einzige Unterschied besteht darin, daß die glättenden Funktionen im Falle neuronaler Netzwerke durch Teilnetzwerke nachgebildet werden (Abbildung 9).

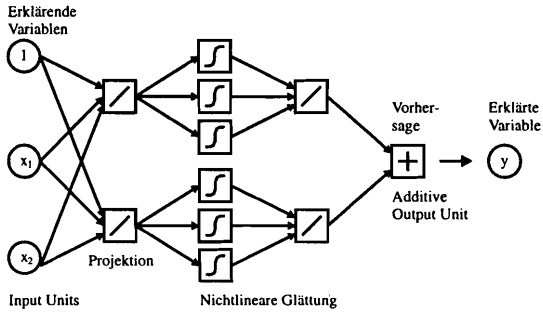

Abbildung 9: *Projection Pursuit*-Regression als neuronales Netz.

Am Beispiel des *Projection Pursuit*-Netzwerks läßt sich deutlich erkennen, daß jedes statistische Verfahren, das zur Approximation von Funktionen Linearkombinationen von Basisfunktionen (z.B. Polynome, *Splines*, trigonometrische Funktionen etc.) verwendet, ebenso als neuronales Netzwerk implementiert werden kann.

Ein Netzwerk, das gänzlich anders funktioniert als die bisher betrachteten, ist das in Abbildung 10 dargestellte *Radial Basis Function*-Netzwerk.[7]

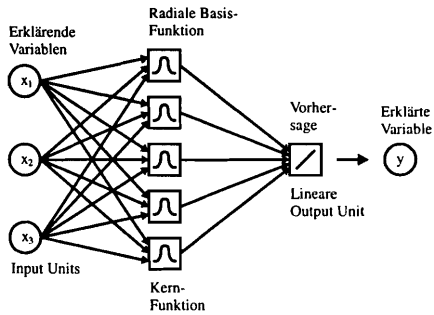

Abbildung 10: *Radial Basis Function*-Netzwerk.

[7]Vgl. Poggio/Girosi (1990).

Anstelle der logistischen Transformationsfunktionen besitzen die *hidden units* radiale Basisfunktionen (RBF), die den in Kern-Regressionen[8] benutzten Kernfunktionen k gleichen. Als solche verwendet man beispielsweise die Gauss'sche Glockenkurve $k(x) = (2\pi)^{-1/2}\,\mathrm{e}^{-x^2/2}$. Die Netzwerkgewichte γ zum *hidden layer* nennt man Stützstellen (*Centers*). Sie werden ohne Training ex ante bestimmt und entsprechen ausgewählten Beobachtungen, die den Regressionsverlauf möglichst gut charakterisieren. Im Extremfall lassen sich alle Beobachtungen als Stützstellen definieren. Im Netzwerktraining werden also lediglich noch die Gewichte β des *hidden layer* bestimmt, die sich unter Umständen sogar analytisch errechnen lassen.

Der Vektor der Eingangssignale r_h, die eine *hidden unit* in einem RBF–Netzwerk erhält, errechnet sich als normierte euklidische Distanz zwischen dem Gewichtsvektor γ_h und dem Inputvektor x_i:

$$r_{t,h} = \left[\sum_{i=1}^{I} \frac{(\gamma_{hi} - x_{t,i})^2}{b_h}\right]^{1/2}. \tag{12}$$

Die Normierung geschieht durch die sogenannte Bandbreite b. Die Bandbreite regelt, wie stark die Bestimmung des Regressionsverlaufs im Bereich der Stützstelle von benachbarten Beobachtungen beeinflußt werden soll. Je größer b gewählt wird, desto weniger stark werden die Nachbarwerte der Stützstelle für die Bestimmung des lokalen Regressionsverlaufs berücksichtigt. In dieser Hinsicht ist die Bandbreite für die Steuerung von *Bias* und Varianz der Schätzung verantwortlich: kleine Bandbreiten sorgen tendenziell für einen niedrigen *Bias* und eine hohe Varianz, große Bandbreiten tendenziell für einen hohen *Bias* und eine kleine Varianz.

Der von den *hidden units* gelieferte Ausgangssignalvektor s_h ergibt sich unter Anwendung der oben beispielhaft genannten Kernfunktion durch

$$s_h = \mathrm{e}^{-r_h^2/2}. \tag{13}$$

Die Netzwerkgewichte β von den *hidden units* zur *output unit* mit Zieloutput y werden analog einer Parameterschätzung im linearen Regressionsmodell bestimmt, d.h.

$$\beta = (S_h' S_h)^{-1} S_h' y, \tag{14}$$

wobei $S_h = [s_1, \ldots, s_H]$.

Damit ergibt sich der Schätzer f für den Funktionsverlauf der wahren Funktion F an der Stelle $X_t = X_\tau$ mittels eines RBF–Netzwerks durch:

$$f(X_\tau, \gamma, \beta) = \sum_{h=1}^{H} \beta_h s_h(X_\tau, \gamma). \tag{15}$$

[8] Vgl. Härdle (1990).

Im Prinzip ist das RBF–Netzwerk nichts anderes als eine Modifikation der schon bekannten Nadaraya–Watson–Kernschätzung[9]. In dieser errechnet sich der Funktionsverlauf des Schätzers f durch:

$$f(X_\tau) = \sum_{t=1}^{T} \frac{y_t}{\sum_{t=1}^{T} k(||X_\tau - X_t||/\sqrt{b})} k(||X_\tau - X_t||/\sqrt{b}). \tag{16}$$

Offensichtlich entspricht diese Regressionsgleichung der eines RBF–Netzwerks, wenn die Anzahl H der *hidden units* genau mit der Anzahl T der Beobachtungen übereinstimmt. Denn in diesem Fall gleichen die Werte der Kernfunktion k den Ausgangssignalen s_h der *hidden units* und die Faktoren der Kernfunktion k entsprechen den β_h, d.h. also:

$$\beta_h = \frac{y_t}{\sum_{t=1}^{T} k(||X_\tau - X_t||/\sqrt{b})}. \tag{17}$$

Es ist deutlich, daß das beschriebene Modell sehr stark überparametrisiert ist. Eine Modifikation der RBF-Netzwerke besteht nun darin, daß diese meist eine geringere Anzahl an *hidden units* verwenden als Beobachtungen vorliegen. Problematisch ist dabei jedoch die geeignete Auswahl der Stützstellen, die den Regressionsverlauf charakterisieren sollen. Aus diesem Grund faßt man die Beobachtungen häufig zu *Clustern* zusammen und benutzt die *Cluster*–Mittelwert als Stützstellen. Die *Cluster*–Varianzen können dann als die Bandbreiten b_h dienen.

2.4 Die Methode der nichtlinearen kleinsten Quadrate

Alle im letzten Abschnitt dargestellten Netzwerkarchitekturen sind Regressionsmodelle. Um nun eine Regression durchführen zu können, benötigt man ein Zielfunktionskriterium. In der Statistik verwendet man dabei verschiedene Kriterien, u.a. die absolute Abweichung $\sum |y - \hat{y}|$ oder die sogenannte *Cross Entropy* $\sum y \cdot \ln(\hat{y}) + (1-y) \cdot \ln(1-\hat{y})$, für $0 < y < 1$. Das jedoch am häufigstes eingesetzte Kriterium ist die Summe der kleinsten Quadrate (*Sum of Squared Errors*, SSE).

Man betrachte dazu das einfache nichtlineare Regressionsmodell

$$y = f(X, \theta) + \varepsilon, \tag{18}$$

für das folgende Annahmen zutreffen sollen. Der Vektor ε sei ein Vektor von i.i.d.-Zufallszahlen (*independent and identically distributed*) mit Erwartungswert $E[\varepsilon] = 0$ und Varianz $\mathrm{Var}[\varepsilon] = \sigma_\varepsilon$.

Da zur Bestimmung der Gewichte bzw. Parameter eines Netzwerks in den meisten Fällen die quadrierten Residuen der Regression minimiert werden, kann unmittelbar die Theorie der nichtlinearen kleinsten Quadrate (NLS) angewendet werden.[10]

[9] Vgl. Härdle (1994).
[10] Vgl. Amemiya (1985).

Der nichtlineare kleinste–Quadrate–Schätzer minimiert die Summe der quadratischen Fehler (SSE):

$$\mathrm{SSE}(\theta) = [y - f(X,\theta)]'[y - f(X,\theta)] \rightarrow \mathrm{Min!} \tag{19}$$

Mit den Annahmen für das nichtlineare Regressionsmodell und einigen unter gewissen Voraussetzungen erfüllten Regularitätsbedingungen für f (White, 1992) kann nun bewiesen werden (Amemiya, 1983), daß der Parameterschätzer $\hat{\theta}$ konsistent ist und eine asymptotische Normalverteilung besitzt

$$\sqrt{T}(\hat{\theta} - \theta) \sim \mathcal{N}(0, C), \tag{20}$$

wobei C einen Kovarianzmatrixterm[11] und T die Anzahl der Beobachtungen bezeichnet. Über die Konsistenz–Eigenschaft hinaus hat der NLS–Schätzer für θ keine weiteren wünschenswerten Eigenschaften (z.B. Effizienz), so daß im Falle einer bekannten Verteilung der Residuen die *Maximum Likelihood*–Methode verwendet werden sollte, die zu einem effizienteren Schätzer $\hat{\theta}_{\mathrm{ML}}$ führt.[12]

Ein konsistenter Schätzer für die Varianz σ^2 der Residuen ist

$$\hat{\sigma}^2_{\mathrm{NLS}} = \frac{\mathrm{SSE}(\hat{\theta}_{\mathrm{NLS}})}{T - K}. \tag{21}$$

Der Wert K steht für die Anzahl der zu bestimmenden freien Parameter in einem Modell bzw. neuronalen Netzwerk.[13]

Gilt für die Kovarianzmatrix der Residuen

$$E[\varepsilon\varepsilon'] = \Omega \neq \sigma^2 I \tag{22}$$

(z.B. bei Heteroskedastizität und/oder serieller Korrelation), dann bleibt unter relativ allgemeingültigen Annahmen der NLS–Schätzer $\hat{\theta}$ konsistent und asymptotisch normalverteilt, jedoch wird der asymptotische Kovarianzmatrixterm noch wesentlich komplexer (Amemyia, 1983). Analog zu einer Schätzung mit *Generalized Least Squares* (GLS) ist es jedoch möglich, eine Schätzung in zwei Stufen auch bei NLS anzuwenden (GNLS), so daß die in Ω enthaltene Information zu einer effizienteren Bestimmung von $\hat{\theta}$ verwendet werden kann.[14]

[11] Vgl. Gleichung (79).

[12] Ein Schätzer ist effizient, wenn er erwartungstreu ist und die kleinste Varianz unter allen alternativen erwartungstreuen Schätzern aufweist.

[13] Da $\tilde{\sigma}^2_{\mathrm{NLS}}$ lediglich einen asymptotischen Schätzer für die Varianz der Residuen darstellt, ist die Bereinigung um die Anzahl der freien Modellparameter K im Nenner der Gleichung (21) nicht notwendig.

[14] Die Vorgehensweise bei einer Zwei–Stufen–Schätzung wird in Abschnitt 3.4 vorgestellt.

2.5 Die Maximum Likelihood Methode

Maximum Likelihood ist ein allgemein einsetzbares Verfahren zur Schätzung von Parametern in den unterschiedlichsten Modellen. Die Grundidee der *Maximum Likelihood*-Methode besteht darin, zunächst eine Annahme über die Wahrscheinlichkeitsverteilung von beobachteten Daten zu machen und dann die Wahrscheinlichkeit zu errechnen, mit der die beobachteten Daten aufgetreten sind. Die Beobachtungswahrscheinlichkeit der Daten hängt üblicherweise von unbekannten Parametern ab, die dann mit Hilfe der *Maximum Likelihood*-Schätzung so bestimmt werden, daß die Wahrscheinlichkeit für das Auftreten der Daten in der beobachteten Weise maximal ist.

Angenommen es liegt ein Menge von Beobachtungen $(z_1, z_2, \ldots, z_T)$ vor, die unabhängig voneinander entsprechend einer identischen Wahrscheinlichkeitsverteilung $P(Z_t|\vartheta)$ der Zufallsvariablen Z_t gezogen wurden. Dabei bezeichnet ϑ einen Parametervektor, der die Wahrscheinlichkeitsverteilung P eindeutig festlegen soll. Unter diesen Annahmen ergibt sich, daß die gemeinsame Wahrscheinlichkeitsverteilung der Menge $(z_1, z_2, \ldots, z_T)$ wie folgt gegeben ist:

$$P[z_1, z_2, \ldots, z_T] = \prod_{t=1}^{T} P[Z_t = z_t|\vartheta]. \tag{23}$$

Das Ziel der *Maximum Likelihood*-Schätzung besteht nun darin, die Parametermenge $\hat{\vartheta}$ zu bestimmen, die die Wahrscheinlichkeit für das Auftreten der z_t in der beobachteten Art und Weise maximiert. Zur einfacheren Berechnung arbeitet man üblicherweise anstatt mit der *Likelihood*-Funktion

$$L(\theta) = \prod_{t=1}^{T} P[Z_t = z_t|\vartheta] \tag{24}$$

mit der logarithmierten *Likelihood*, also mit:

$$\mathcal{L}(\theta) = \ln L(\theta) = \sum_{t=1}^{T} \ln P[Z_t = z_t|\vartheta]. \tag{25}$$

Der Vorteil des *Maximum Likelihood*-Verfahrens ist seine allgemeingültige Einsetzbarkeit. *Maximum Likelihood*-Schätzungen sind normalerweise konsistent und asymptotisch effizient (Judge et al., 1988).

Betrachtet sei nun das einfache nichtlineare Regressionsmodell aus Gleichung (18) mit den dort gemachten Annahmen. Unter der zusätzlichen Annahme, daß die Residuen multivariat normalverteilt[15] sind mit Erwartungswert Null und Kovarianz Ω, kann folgende *Loglikelihood*-Funktion aufgestellt werden, wobei $\vartheta = (\theta, \Omega)$:

$$\mathcal{L}(\theta, \Omega) = -\frac{1}{2}(T \ln(2\pi) + \ln|\Omega| + [y - f(X,\theta)]'\Omega^{-1}[y - f(X,\theta)]) \tag{26}$$

[15] Die Dichte der multivariate Normalverteilung lautet $\phi(x) = (2\pi)^{-T/2}|\Omega|^{-1/2}\,\mathrm{e}^{-\frac{(x-\mu)'\Omega^{-1}(x-\mu)}{2}}$.

Die Maximierung dieser Funktion führt zu Schätzwerten für die Parameterwerte von θ und Ω. (Es ist offensichtlich, daß der Kovarianzmatrix Ω einige Restriktionen auferlegt werden müssen, da die Anzahl der zu bestimmenden Parameter des Modells kleiner sein muß als die Anzahl der zur Verfügung stehenden Beobachtungen.) Der *Maximum Likelihood*-Schätzer ist konsistent, er hat eine asymptotische Normalverteilung und er ist asymptotisch effizient (Greene, 1993).

In dem Spezialfall, daß $\Omega = \sigma^2 I$ ist und die gemeinsame Verteilung der Residuen der multivariaten Normalverteilung entspricht, gleichen sich die Methoden NLS und *Maximum Likelihood*. Wendet man also unter den Annahmen normalverteilter Residuen und einer konstanten Varianz zur Schätzung der Parameter eines neuronalen Netzwerks die Methode der kleinsten Quadrate an, dann entsprechen die geschätzten Parameterwerte einer *Maximum Likelihood*-Schätzung. Andersherum bedeutet dies, daß bei Verwendung der Methode der kleinsten Quadrate implizit immer die Annahme normalverteilter Residuen getätigt wird. Trifft diese Annahme nicht zu, dann nennt man die mit der kleinsten-Quadrate-Methode bestimmten Parameterwerte *Pseudo Maximum Likelihood*-Schätzer. Grundsätzlich spricht man von einer *Pseudo Maximum Likelihood*-Schätzung[16], wenn die bei *Maximum Likelihood* verwendete Verteilung nicht der tatsächlichen Verteilung entspricht.

Zum Abschluß der Betrachtung der beiden Verfahren soll noch auf ein Argument hingewiesen werden, das häufig fälschlicherweise für die Verwendung neuronaler Netzwerke vorgebracht wird. Diese Argument lautet, daß die Verwendung neuronaler Netzwerke immer dann geschehen sollte, wenn man eine Funktion approximieren möchte, ohne dabei eine Annahme über die Verteilung der Störterme machen zu müssen. Wie aber oben dargestellt wurde, ist diese Argument falsch. Implizit unterstellt man nämlich — auch bei neuronalen Netzwerken — durch die Verwendung der kleinsten-Quadrate-Methode — normalverteilte Störterme. Hingegen trifft man bei neuronalen Netzwerken keine Annahme hinsichtlich der funktionalen Form der zu approximierenden Funktion. Dies ist der wesentliche Unterschied zwischen neuronalen Netzwerken und anderen parametrischen Verfahren.

2.6 Numerische Schätzverfahren

Im allgemeinen lassen sich die Parameter eines nichtlinearen Regressionsmodells nicht oder nur sehr schwer analytisch bestimmen. Deshalb ist es sowohl bei der *Maximum Likelihood*- als auch bei der kleinsten-Quadrate-Methode in den meisten Fällen notwendig, ein numerisches Verfahren einzusetzen, das die optimalen Parameter der Regressionsfunktion mit Hilfe eines Näherungsverfahrens bestimmen soll.

Ohne Beschränkung der Allgemeinheit sei für die folgende Darstellung der verschiedenen Näherungsverfahren eine Zielfunktion $q(\theta)$ zu minimieren.[17]

[16] Vgl. Gourieroux/Monfort/Trognon (1984).

[17] Im Fall der kleinsten-Quadrate-Methode gilt $q(\theta) = \mathrm{SSE}(\theta)$, im Fall der *Maximum Likelihood* gilt $q(\theta) = -\ln L(\theta) = -\mathcal{L}(\theta)$.

Die Gruppe der Abstiegsverfahren läßt sich in Algorithmen unterteilen, die bei der Optimierung den Gradienten der Zielfunktion verwenden, und solche, die dies nicht tun. Verfahren der ersten Gruppe (Gradientenmethoden) sind die mit Abstand wichtigeren und werden auch in den meisten Fällen verwendet (z.B. Newton–Raphson, Davidon–Fletcher–Powell, steilster Abstieg, etc). Verfahren der zweiten Gruppe werden meist eingesetzt, wenn die Zielfunktion in einem hohen Maße irregulär verläuft. Zu diesen Verfahren zählen Gittersuche, Simulated Annealing oder die in letzter Zeit vielfach untersuchten genetischen Algorithmen. Die Funktionalität der Verfahren in der zweiten Gruppe für das hier gestellte Problem wird wegen ihres enormen Rechenaufwands jedoch stark bezweifelt, so daß sie im folgenden nicht weiter betrachtet werden. Für eine umfangreiche Darstellung dieser Verfahren sei auf Davis (1987) verwiesen.

Die zur numerischen Lösung eines Minimierungsproblems eingesetzten Gradientenverfahren besitzen eine verwandte Vorgehensweise. Sie alle starten mit einer Initialschätzung für θ, z.B. $\theta_{(0)}$, und konstruieren eine endliche Sequenz von Approximationen $\theta_{(0)}, \theta_{(1)}, \ldots, \theta_{(N)}$, mit dem Ziel, daß $\theta_{(N)} \to \theta^*$, wenn $N \to \infty$.

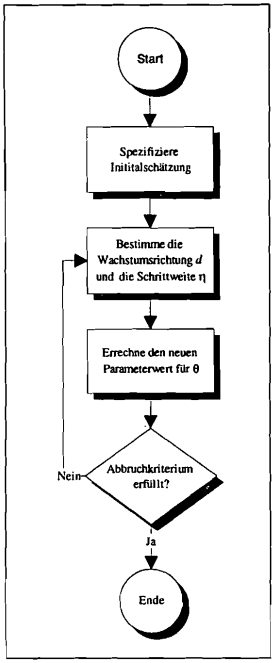

Abbildung 11: Iterationsschritte numerischer Gradientenverfahren.

Wie in Abbildung 11 gezeigt lassen sich die Gradientenverfahren in vier wesentlich Schritte gliedern:

1. Wähle einen geeigneten (eventuell zufälligen) Startwert $\theta_{(0)}$.
2. Bestimme eine Abstiegsrichtung d für $\theta_{(n)}$, die den Wert der Zielfunktion $q(\theta_{(n)})$ verkleinert.
3. Bestimme die Schrittlänge η in der Abstiegsrichtung und damit den neuen Parameterwerte $\theta_{(n+1)}$.
4. Prüfe, ob die festgelegten Abbruchkriterien erfüllt sind. Wenn ja, breche ab, wenn nein, wiederhole Schritt 2.

Als Abbruchkriterien sollten folgende Bedingungen erfüllt sein, wobei δ_1 und δ_2 zwei kleine Tolaranzgrenzen bezeichnen und $\dot{q}$ den Gradienten der Funktion q hinsichtlich $\theta_{(n)}$ darstellt.

- $||\theta_{(n+1)} - \theta_{(n)}|| < \delta_1$
- $q(\theta_{(n+1)}) - q(\theta_{(n)}) < \delta_2$
- $\dot{q}(\theta_{(n+1)}) \approx 0$

Nach Abbruch muß zusätzlich die Bedingung zweiter Ordnung überprüft werden, d.h. die Hessematrix von $q(\theta_{(n)})$ muß positiv definit sein, um auszuschließen, daß $\theta_{(n)}$ einen Sattelpunkt der Funktion q darstellt.

Bevor die Gradientenverfahren im einzelnen betrachtet werden, sollen noch einige Bezeichnungen vereinbart werden:

- $\dot{q}(\theta)$ bezeichnet den Gradienten der Funktion $q(\theta)$, d.h. $\dot{q}(\theta) = \frac{\partial q(\theta)}{\partial \theta}$; $\dot{q}_{(n)}$ bezeichnet den Gradienten der Funktion $q(\theta)$ an der Stelle $\theta = \theta_{(n)}$, d.h. $\dot{q}_{(n)} = \frac{\partial q(\theta)}{\partial \theta}|_{\theta=\theta_{(n)}}$.
- $H(\theta)$ bezeichne die Hessematrix der Funktion $q(\theta)$, d.h. $H(\theta) = \frac{\partial^2 q(\theta)}{\partial \theta \partial \theta'}$; $H_{(n)}$ bezeichnet die Hessematrix der Funktion $q(\theta)$ an der Stelle $\theta = \theta_{(n)}$.
- $Q_{(n)}$ bezeichnet die Richtungsmatrix, mit deren Hilfe sich die Abstiegsrichtung d im n-ten Iterationsschritt bestimmen läßt. $Q_{(n)}$ charakterisiert das verwendete Gradientenverfahren eindeutig.
- η bezeichne die skalare Schrittweite in der Abstiegsrichtung $d(\theta)$.

Alle Gradientenverfahren erzeugen theoretisch eine Folge $q(\theta_{(0)}), q(\theta_{(1)}), \ldots, q(\theta_{(n)})$ mit der Eigenschaft, daß

$$q(\theta_{(n+1)}) < q(\theta_{(n)}), \tag{27}$$

wobei das Bildungsgesetz für $\theta_{(n+1)}$

$$\theta_{(n+1)} = \theta_{(n)} - \eta_{(n)} \cdot d_{(n)}(\theta) = \theta_{(n)} - \eta_{(n)} \cdot Q_{(n)} \dot{q}_{(n)} \tag{28}$$

allen Gradientenverfahren gemein ist. Die Matrix $Q_{(n)}$ muß in jedem Iterationsschritt positiv definit sein, um zu gewährleisten, daß der Iterationsschritt von $\theta_{(n)} \rightarrow \theta_{(n+1)}$ auch tatsächlich zu einer Verkleinerung des Zielfunktionswerts führt.[18] Verfahren, die in ihren Iterationsschritten lediglich den Gradienten der Zielfunktion benötigen, haben die Ordnung eins, Verfahren, die darüber hinaus die Hessematrix der Zielfunktion oder eine entsprechende Approximation verwenden, die Ordnung zwei.

2.6.1 Newton–Verfahren

Das Newton–Verfahren (auch Newton–Raphson genannt) ist das fundamentale und gleichzeitig anspruchvollste der Gradientenverfahren. Es basiert auf einer quadratischen Taylorreihenapproximation der Zielfunktion q durch $\tilde{q}$ an der Stelle $\theta_{(n)}$:

$$q(\theta) \approx \tilde{q}(\theta) = q(\theta_{(n)}) + \dot{q}_{(n)} \cdot (\theta - \theta_{(n)}) + \frac{1}{2}(\theta - \theta_{(n)})' H_{(n)} (\theta - \theta_{(n)}) \tag{29}$$

Aus der Bedingung erster Ordnung für ein Optimum von $\tilde{q}$ ergibt sich

$$\frac{\partial \tilde{q}(\theta)}{\partial \theta} = \dot{q}_{(n)} + H_{(n)}(\theta - \theta_{(n)}) = 0 \tag{30}$$

[18] Vgl. Judge/Griffiths/Hill/Lütkepohl/Lee (1985).

und daraus

$$\theta = \theta_{(n)} - H_{(n)}^{-1}\dot{q}_{(n)}. \tag{31}$$

Für die Berechnung des aktuellen Iterationsschritts im Newton–Verfahren gilt für die Matrix $Q_{(n)}$ aus Gleichung (28) daher:

$$Q_{(n)} = H_{(n)}^{-1}. \tag{32}$$

Hätte die Funktion q eine quadratische Form, würde das Newton–Verfahren das Optimum in einem Schritt erreichen. In allen anderen Fällen besteht die Funktionsweise des Verfahrens in einer Reihe von lokalen Approximationen entsprechend der Gleichung (28).

Das Newton–Verfahren konvergiert unter „normalen Unständen" in wesentlich weniger Schritten als alle anderen Verfahren. Jedoch dauert die Berechnung der einzelnen Iterationsschritte erheblich länger, da jedesmal sowohl der Gradient als auch die Hessematrix von q an der aktuellen Stelle $\theta_{(n)}$ ermittelt werden müssen. Je höher die Anzahl der Parameter eines Modells ist, desto aufwendiger ist die Berechnung der Hessematrix, so daß das Newton–Verfahren für die Schätzung in großen Modellen weniger geeignet ist. Ist die die Hessematrix darüber hinaus analytisch nicht spezifiziert und muß deshalb numerisch berechnet werden, reduziert sich der Vorteil des Newton–Verfahrens gegenüber den Quasi–Newton–Verfahren noch weiter. Das Newton–Verfahren ist zudem sehr fehleranfällig gegen Datenprobleme[19] und konvergiert in solchen Fällen nur unzureichend. Darüber hinaus ist die Hessematrix üblicherweise nur in einer kleinen Region um das Minimum positiv definit, so daß eventuell auch ein Sattelpunkt oder ein Maximum erreicht werden kann. Die Überprüfung der Hessematrix H im erreichten Endpunkt der Iteration ist also extrem wichtig.

2.6.2 Quasi–Newton–Verfahren

Quasi–Newton–Verfahren basieren auf dem Newton–Verfahren. Anstatt jedoch die Hessematrix in jedem Iterationsschritt neu zu berechnen, approximieren sie die Hessematrix, indem diese einmalig berechnet und in allen anschließenden Schritten lediglich um eine Matrix $K_{(n)}$ korrigiert wird. Dabei wird die Matrix $K_{(n)}$ so gewählt, daß $Q_{(n+1)}$ positiv definit bleibt.

$$Q_{(n+1)} = Q_{(n)} + K_{(n)} \tag{33}$$

Mit Fortschreiten des Iterationsprozesses wird die Approximation der wahren Hessematrix immer besser. Die bekanntesten Quasi–Newton–Verfahren heißen BFGS nach Broyden, Fletcher, Goldfarb und Shanno und DFP nach Davidon, Fletcher und Powell.

Quasi–Newton–Verfahren benötigen durchschnittlich mehr Iterationsschritte bis zur Konvergenz als das Newton–Verfahren, wegen des geringeren Rechenaufwands pro Iterationsschritt sind sie in der Regel dennoch schneller. Dadurch, daß die Hessematrix lediglich

[19] Z.B. Multikollinearität; in diesem Fall ist die Hessematrix fast singulär und die inverse sehr irregulär.

approximiert wird, sind Quasi-Newton-Verfahren zudem in der Regel weniger sensitiv gegenüber den Bedingungen des Modells sowie der Daten als das Newton-Verfahren.

2.6.3 Gauss-Newton

Ein weiteres Verfahren, das die Hessematrix der Zielfunktion lediglich approximiert, ist das Gauss-Newton-Verfahren. Es basiert auf der Verwendung der kleinsten-Quadrate-Methode dargestellt in Gleichung (19). Der Gradient der Zielfunktion $\mathrm{SSE}(\theta)$ lautet

$$\frac{\partial \mathrm{SSE}(\theta)}{\partial \theta} = -2\dot{f}(X,\theta)'[y - f(X,\theta)], \tag{34}$$

wobei $\dot{f}$ den Gradienten der Funktion f bezeichnet. Die sich weiterhin ergebende Hessematrix läßt sich unter Vernachlässigung eines Rests durch

$$\frac{\partial^2 \mathrm{SSE}(\theta)}{\partial \theta \partial \theta'} \approx 2\dot{f}(X,\theta)'\dot{f}(X,\theta) \tag{35}$$

approximieren. Nach obiger Iterationsformel (28) gilt damit

$$\theta_{(n+1)} = \theta_{(n)} + [\dot{f}(X,\theta_{(n)})'\dot{f}(X,\theta_{(n)})]^{-1}\dot{f}(X,\theta_{(n)})'[y - f(X,\theta_{(n)})], \tag{36}$$

wobei

$$Q_{(n)} = [\dot{f}(X,\theta_{(n)})\dot{f}(X,\theta_{(n)})']^{-1} \tag{37}$$

entspricht. Die Schrittweite η ist in diesem Fall 1. Die Intuition dieser Methode wird klar, wenn man die Gleichung (36) zur Veranschaulichung lediglich in ihrem Gerüst aufschreibt. Dann ergibt sich nämlich die jeweilige Änderung des Parameters $\theta_{(n)}$ durch

$$\theta_{(n+1)} - \theta_{(n)} = [\dot{f}'\dot{f}]^{-1}\dot{f}u_{(n)}. \tag{38}$$

Die Differenz der θ-Werte ist damit nichts anderes als der OLS-Schätzer, wenn man die Residuen $u_{(n)} = y - f(X,\theta_{(n)})$ in jedem Iterationsschritt auf den Gradienten der Funktion f regressiert.

2.6.4 Scoring-Verfahren

Im Gegensatz zum Gauss-Newton-Verfahren basiert das *Scoring*-Verfahren auf der *Maximum Likelihood*-Methode, d.h. $q(\theta) = -\ln L(\theta)$. Im *Scoring*-Verfahren wird die lokale Hessematrix der *Loglikelihood*-Funktion durch ihren Erwartungswert approximiert, d.h. in Gleichung (28) ist

$$Q_{(n)} = -E\left[\frac{\partial^2 q(\theta)}{\partial \theta \partial \theta'}|_{\theta_{(n)}}\right]^{-1} = \mathcal{I}_{(n)}(\theta). \tag{39}$$

Die Matrix $\mathcal{I}(\theta)$ ist unter dem Namen Fisher–Informationsmatrix bekannt. Sie läßt sich alternativ durch

$$\mathcal{I}(\theta) = E\left[\frac{\partial \ln L(\theta)}{\partial \theta}\frac{\partial \ln L(\theta)}{\partial \theta'}\right] \tag{40}$$

errechnen. Das *Scoring*–Verfahren hat seinen Namen von der ersten Ableitung $\partial \ln L(\theta)/\partial\theta$ der *Loglikelihood*–Funktion, die auch *Score*–Funktion genannt wird. Eine Weiterentwicklung des *Scoring*–Algorithmus wurde von Berndt, Hall, Hall und Hausman (BHHH) durchgeführt.

Die Vorteile des *Scoring*–Verfahrens bestehen darin, daß erstens die Fisher–Informationsmatrix, und damit die Matrix Q, in einem identifizierten Modell immer positiv definit ist und zweitens, daß lediglich die ersten Ableitungen für die Berechnung eines Iterationsschritts benötigt werden. Nachteilig ist jedoch, daß zur Schätzung der Informationsmatrix eine Gradientenmatrix in der Größe der gesamten Datenmenge errechnet werden muß, da diese eine Erwartungswertematrix darstellt. Dies ist ein Unterfangen, das besonders bei großen Datenmengen sehr aufwendig ist. Das *Scoring*–Verfahren (bzw. BHHH) sollte daher nicht die erste Wahl für eine Optimierung sein.

2.6.5 Steilster Abstieg und konjugierter Gradientenabstieg

Der steilste Abstieg an einer beliebigen Stelle der Zielfunktion wird durch ihren Gradienten an dieser Stelle gegeben. In diesem Fall entspricht die Matrix $Q_{(n)}$ aus Gleichung (28) in jedem Iterationsschritt der Einheitsmatrix. Die Methode des steilsten Abstiegs verwendet also weder die Hessematrix der Zielfunktion noch eine ihrer Approximationen und benötigt deshalb einen vergleichsweise geringen Rechenaufwand. Zudem gelangt diese Methode üblicherweise schneller in die Nähe des lokalen Minimums als die Algorithmen zweiter Ordnung, wenn der Startwert für die Iteration schlecht gewählt wurde. In der Nähe des lokalen Optimums konvergieren die Algorithmen erster Ordnung jedoch sehr schlecht, da sie in jedem Iterationsschritt dieselbe Richtungsmatrix Q benutzen und dadurch keine flexible Anpassung an die unterschiedlichen Formen der Zielfunktion zugelassen wird.

Eine deutliche Verbesserung des einfachen Gradientenabstiegs erreichen sogenannte konjugierte Gradientenabstiegsverfahren, die keinen höheren Rechen- oder Speicheranforderungen stellen als die einfachen Gradientenabstiegsverfahren. Die Idee des konjugierten Gradientenabstiegs besteht darin, nicht in Richtung des aktuellen Gradienten abzusteigen, sondern in eine Richtung, die sich unter Einbeziehung der vergangenen Abstiegsrichtungen ergibt. Beim konjugierten Gradientenabstieg ergibt sich die Abstiegsrichtung d in Gleichung (28) durch

$$d_{(n+1)} = \dot{q}_{(n+1)} + \kappa_{(n)} \cdot d_{(n)} \tag{41}$$

wobei sich der Skalar $\kappa_{(n)}$ beispielsweise beim konjugierten Gradientenabstieg nach Polak und Ribiere durch

$$\kappa_{(n)} = \frac{(\dot{q}_{(n+1)} - \dot{q}_{(n)})'\dot{q}_{(n+1)}}{\dot{q}'_{(n)}\dot{q}_{(n)}} \tag{42}$$

errechnet. Der Polak–Ribiere–Gradientenabstieg ist anderen konjugierten Verfahren (z.B. Fletcher–Reeves) im Schnitt überlegen.[20]

2.6.6 Levenberg–Marquardt

Der Levenberg–Marquardt–Algorithmus ist kein eigenständiges Optimierungsverfahren, sondern kann dazu benutzt werden, solche Verfahren zu erweitern, die keine semidefinit positive Richtungsmatrix Q gewährleisten. Der Algorithmus nutzt dabei aus, daß

$$Q_{(n)} + \rho_{(n)}\tilde{Q}_{(n)} \tag{43}$$

immer positiv definit ist, wenn $\tilde{Q}_{(n)}$ positiv definit und der Parameter $\rho_{(n)}$ hinreichend groß gewählt ist. Für $\tilde{Q}_{(n)}$ wird häufig die Identitätsmatrix benutzt. Eine Levenberg–Marquardt–Modifikation eignet sich vor allen Dingen für das Gauss–Newton oder das Newton–Raphson–Verfahren. Insbesondere wenn sich die Zielfunktion nur schlecht quadratisch approximieren läßt, z.B. in einiger Entfernung zum lokalen Optimum, kann man mit einer Levenberg–Marquardt–Modifikation eine erhebliche Verbesserung der Konvergenz des entsprechenden Optimierungsverfahrens erreichen.

2.6.7 Line Search

Die Wahl der skalaren Schrittweite η in der Abstiegsrichtung d in Gleichung (28) ist ein eigenständiges Optimierungsproblem. Diese Suche bezeichnet man als *Line Search*, da lediglich in einer Richtung, nämlich der Abstiegsrichtung, das Optimum gefunden werden muß. Zur Auswahl für die Suche nach dem lokalen Optimum in der Abstiegsrichtung stehen sowohl iterative Algorithmen zur Verfügung (z.B. *Golden Section* oder Brents Methode) als auch Algorithmen, die die Wachstumrichtung mit einem Polynom approximieren und dieses dann minimieren (z.B. *Stepbt*).

Die *Line Search*–Verfahren bieten sich als natürliche Erweiterung der Gradientenverfahren zweiter Ordnung an. In diesen Methoden ergibt sich bei der Herleitung üblicherweise eine Schrittweise von $\eta = 1$.

2.6.8 Backpropagation

In den letzten Abschnitten sind die Standardverfahren zur Bestimmung der Optima von nichtlinearen Funktionen dargestellt worden. Diesen Verfahren soll nun der bekannteste Optimierungsalgorithmus, der bei neuronalen Netzwerken Verwendung findet, gegenübergestellt werden. Der Algorithmus heißt *Backpropagation* und wurde von Rumel-

[20] Vgl. Press/Flannery/Teukolsky/Vetterling (1992).

hart/McClelland (1986) vorgestellt. Er leitet sich aus der Minimierung der quadratischen Abweichungen

$$\text{SSE} = \sum_{t=1}^{T} \text{SE}_t \rightarrow \text{Min} \tag{44}$$

her, wobei

$$\text{SE}_t = [y_t - f(X_t, w)]^2 \tag{45}$$

und entspricht im Prinzip einem einfachen Gradientenabstiegsverfahren. Die Iterationsregel für die Parameter w, i.e. die Gewichte eines Netzwerks, lautet:

$$\begin{aligned} w_{(n+1),t} &= w_{(n),t} - \Delta w_{(n),t} \\ &= w_{(n),t} - \eta \cdot \frac{\partial \text{SE}_t(w)}{\partial w_{(n),t}}. \end{aligned} \tag{46}$$

Der Unterschied zu den Standard–Gradientenabstiegsverfahren besteht darin, daß die Parameterwerte sofort in Abhängigkeit jeder Beobachtung verändert werden. Die Gesamtänderung $\Delta w_{(n)}$ der Parameter, die sich nach einer Epoche (eine Präsentation aller T Beobachtungen) ergibt, errechnet sich dabei nicht mehr aus dem Gradienten der Fehlerfunktion SSE, d.h.

$$\frac{\partial \text{SSE}(w)}{\partial w_{(n)}} = \sum_{t=1}^{T} \frac{\partial \text{SE}_t(w)}{\partial w_{(n)}}, \tag{47}$$

sondern durch eine Annäherung

$$\frac{\partial \text{SSE}(w)}{\partial w_{(n)}} \simeq \sum_{t=1}^{T} \frac{\partial \text{SE}_t(w)}{\partial w_{(n),t}}. \tag{48}$$

Diese Verfahrensweise ist grundsätzlich jedoch nur dann notwendig, wenn zum Zeitpunkt des Trainings nicht alle Beobachtungen vorliegen, sondern das Netzwerk auf neu hinzukommende Beobachtungen in Echtzeit angepaßt werden soll. Im einem statistischen bzw. ökonometrischen Kontext ist dies jedoch nicht der Fall, denn dort werden nur solche Beobachtungen verwendet, die zum Zeitpunkt der Untersuchung zur Verfügung stehen. Im Gegenteil, bei der Verwendung von *Backpropagation* oder verwandten Algorithmen nimmt man wegen der Approximation in (48) von vorneherein eine nicht notwendige leichte Ineffizienz in Kauf. Um diese Ineffizienz zu vermeiden, sollte grundsätzlich der sogenannte *Batch Backpropagation*–Algorithmus[21] Verwendung finden. Er verzichtet auf die Approximation in Gleichung (48) und nimmt die Änderung der Gewichte immer erst am Ende einer Epoche vor.

[21] Vgl. Rojas (1993).

Eine weitere Schwäche des *Backpropagation*-Algorithmus besteht darin, daß er standardmäßig eine konstante (oder abnehmende) Lernrate anstatt — wie das bei Gradientenverfahren üblich ist — ein *Line Search*-Verfahren zur Bestimmung der optimalen Schrittweite verwendet.

Eine Erweiterung des *Backpropagation* führt zu folgender modifizierten Iterationsregel

$$\Delta w_{(n+1),t} = \eta \cdot \frac{\partial \mathrm{SE}_t(w)}{\partial w_{(n),t}} + \mu \Delta w_{(n),t}, \tag{49}$$

wobei man den zweiten Summanden als *Momentum*-Term mit einem *Momentum*-Parameter μ bezeichnet. Diese Erweiterung entspricht von der Intention dem konjugierten Gradientenabstieg in Gleichung (41), jedoch mit dem Unterschied, daß der Parameter μ wiederum ex ante mittels heuristischer Regeln gewählt wird, anstatt wie in Gleichung (42) optimal bestimmt zu werden.

Neben diesen speziellen Schwächen der *Backpropagation*-Algorithmen im Vergleich mit den herkömmlichen Optimierungsverfahren, besitzen sie darüber hinaus alle Nachteile der bereits oben dargestellten Gradientenabstiegsverfahren erster Ordnung. Unter den in dieser Arbeit beschriebenen Algorithmen sind die *Backpropagation*-Algorithmen also die mit Abstand leistungsschwächsten. Zur Schätzung der Parameter eines Netzwerks sollten deshalb an Stelle der *Backpropagation*-Algorithmen eher die oben beschriebenen Gradientenabstiegsverfahren, insbesondere die Quasi-Newton-Verfahren[22], Verwendung finden. Sie beschleunigen die Bestimmung der Parameter in neuronalen Netzwerken um Größenordnungen.

2.6.9 Probleme

Die Probleme, die bei numerischen Verfahren auftreten können, sind vielfältig:

- Es ist möglich, daß der Iterationsprozeß nicht konvergiert, wenn die Initialschätzung zu Beginn des Verfahrens 'zu weit entfernt' von dem lokalen Optimum liegt.
- Die Funktion q kann mehrere lokale Optima aufweisen und man kann nicht sicher sein, daß die Iteration im globalen Optimum endet.
- Das Iterationsverfahren muß nicht unbedingt in einer vorher festgelegten Anzahl von Schritten konvergieren, sondern kann sehr langsam sein.
- Die Verfahren sind teilweise sensitiv gegenüber Datenproblemen (Hessematrix ist bei Multikollinearität fast singulär) und weisen in solchen Fällen ein anormales Verhalten auf.

[22] Das für neuronale Netzwerke entwickelte *Quickprop*-Verfahren (Fahlmann, 1989) ist im Prinzip ein Quasi-Newton-Verfahren, das jedoch mit einigen heuristischen Komponenten ausgestattet wurde.

Trotz all dieser Schwierigkeiten arbeiten die hier vorgestellten Methoden unter 'normalen Umständen' relativ gut und liefern hinreichend gute Schätzergebnisse. Eventuell lassen sich die erzielten Ergebnisse durch die Kombination verschiedener Verfahren noch verbessern. Eine gute Strategie für eine Optimierung kann sein, zunächst mit einem (konjugierten) Gradientenabstiegsverfahren zu beginnen, das von der gewählten Startlösung vergleichsweise schnell in die Nähe des lokalen Optimums führt, und dieses dann nach einer Anzahl von Iterationsschritten durch ein Verfahren zweiter Ordnung abzulösen, das die Bestimmung des lokalen Optimums in einer näheren Umgebung von diesem wesentlich beschleunigt. In allen Fällen ist es ratsam, die optimale Schrittweite jeweils mit einem *Line Search* zu bestimmen.

3 Modellbildung mit neuronalen Netzwerken

Im Abschnitt 2 wurden neuronale Netzwerke in den Kontext der herkömmlichen ökomometrischen bzw. statistischen Verfahren gestellt. In diesem Abschnitt soll nun eine Vorgehensweise dargestellt werden, wie der praktische Einsatz von neuronalen Netzwerken unter einer statistischen Perspektive erfolgen sollte. Die Anwendung statistischer Methoden auf neuronale Netzwerke soll mit Neurometrie bezeichnet werden.

Der wichtigste aber auch schwierigste Schritt bei der Anwendung neuronaler Netzwerke ist die Wahl einer geeigneten Netzwerkarchitektur, d.h. die Bestimmung der Anzahl freier Parameter in einem Netzwerk. Bei der Analyse in Abschnitt 2.1 hat sich ergeben, daß sich neuronale Netzwerke als Verfahren interpretieren lassen, die auf einem Kontinuum zwischen parametrischen und nichtparametrischen Verfahren liegen. Die Stelle, die ein neuronales Netz auf diesem Kontinuum einnimmt, wird dabei durch seine Parameterzahl bestimmt. Die zu erwartenden Residuen werden bei einem kleinen Netz tendenziell eher durch eine Verzerrung im Erwartungswert (*Bias*) erzeugt, bei einem großen Netz durch eine hohe Varianz der Schätzgröße. Um nun den kleinstmöglichen Schätzfehler zu erzeugen, sollten sowohl *Bias* als auch Varianz der Schätzung so klein wie möglich gehalten werden. Im Prinzip ist dies dann der Fall, wenn das Netzwerk gerade groß genug ist, die gewünschte Funktion approximieren zu können. Dann nämlich ist die Schätzung unverzerrt und die Varianz aufgrund der im Netzwerk so klein wie möglich gehaltenen Parameterzahl vergleichsweise gering. Die beste Stelle auf dem Kontinuum (Abbildung 5) ist bestimmt. Mit anderen Worten: es soll eine Netzwerkarchitektur bestimmt werden, die so klein ist, daß das Rauschen in den Beobachtungen nicht approximiert, also kein *overfitting* betrieben wird, und die groß genug ist, den funktionalen Anteil der Beobachtungen abzubilden, also ein *underfitting* vermieden wird. Ein Netzwerk dieser Art ist um irrelevante *input* und *hidden units* bereinigt.

Die neurometrische Modellierung von Netzwerken unterscheidet sich grundsätzlich nicht von dem Vorgehen, das üblicherweise in der Ökonometrie Anwendung findet. Eine klassische Vorgehensweise wurde von Box und Jenkins (1976) vorgeschlagen. Die vorliegende Arbeit orientiert sich an diesem Ansatz. Er besteht aus vier Schritten:

1. Identifikation der Daten.
2. Spezifikation des Modells.
3. Schätzung der Parameter des Modells.
4. Diagnose der Parameter und des Modells.

Die Bearbeitung dieser vier Schritte ist — wie in Abbildung 12 dargestellt — ein rekursiver Prozeß und wird solange fortgeführt, bis die resultierende Netzwerkarchitektur für die zu lösende Aufgabe geeignet ist.

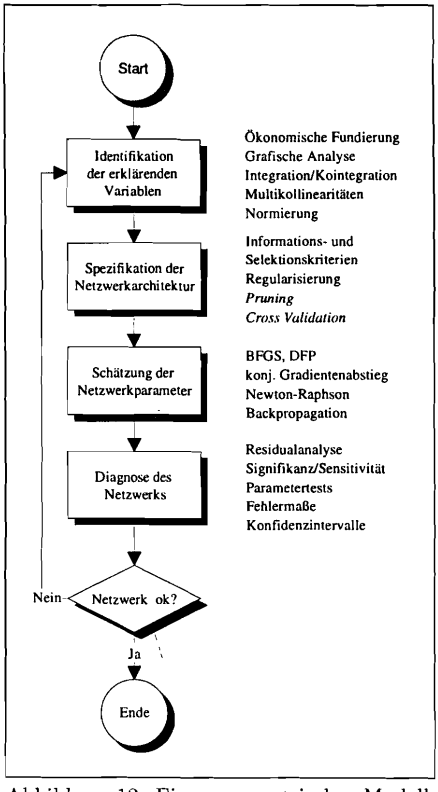

Abbildung 12: Ein neurometrischer Modellbildungsprozeß.

An dieser Stelle soll noch einmal explizit darauf hingewiesen werden, daß ein erheblicher Unterschied zwischen der Spezifikation eines Modells und der Schätzung der Parameter dieses Modells existiert. In der Literatur zu neuronalen Netzwerken wird nämlich häufig keine Unterscheidung zwischen diesen Schritten getroffen. Bereits daraus erkennt man, daß dem wichtigen Modellierungsschritt der Netzwerkspezifikation in der Vergangenheit viel zu wenig Aufmerksamkeit geschenkt wurde. Das Resultat davon ist, daß eine große Anzahl von Untersuchungen auf einem überparametrisierten Modell basiert bzw. viel zu viele Inputvariablen verwendet.[23] Grundsätzlich muß die Anzahl der Beobachtungen die Anzahl der freien Modellparameter übersteigen. Wirklich zuverlässige Aussagen über ein Modell können nur getroffen werden, wenn die Anzahl der Beobachtungen die Parameterzahl um einen Faktor von mindestens zehn übertrifft (White, 1992).

Im folgenden soll ein Vorschlag zur Vereinheitlichung des Modellbildungsprozesses mit neuronalen Netzwerken gemacht werden.

[23] Vgl. Weigend (1990) oder Rehkugler/Poddig (1993).

3.1 Identifikation

In den letzten Abschnitten ist immer wieder betont worden, daß neuronale Netzwerke nichts anderes sind als eine bestimmte Klasse von statistischen Verfahren. Der Glaube also, daß neuronale Netzwerke aufgrund irgendeines 'intelligenten' Verhaltens selbständig in der Lage sein könnten, aus einer bestimmten Menge von unabhängigen Variablen diejenigen zu identifizieren, die für die Erkärung oder Prognose einer abhängigen Variable notwendig sind, ist trügerisch. Das Resultat dieses Irrglaubens ist, daß in einigen Fällen neuronale Netzwerke mit enormen Datenmengen gefüttert wurden, in der Annahme, das Netz werde schon die relevanten von den irrelevanten Informationen trennen. Das heißt nun nicht, daß sich mit neuronalen Netzwerken keine gegenseitigen Abhängigkeiten zwischen verschiedenen Größen identifizieren ließen, sondern nur, daß auch neuronale Netzwerke anfällig gegen Datenprobleme wie z.B. Multikollinearitäten sind. Aus diesem Grund muß auch bei der Modellierung mit neuronalen Netzwerken dem Schritt, geeignete Inputvariablen zu identifizieren, große Bedeutung beigemessen werden.

Auswahl der Daten

Die Auswahl der Daten erfolgt bei der Modellierung mit neuronalen Netzwerken genau wie bei herkömmlichen Modellierungsansätzen aufgrund verschiedener Überlegungen. Zunächst sollten aus der Menge der verfügbaren Daten diejenigen Größen gewählt werden, die in einem wie auch immer gearteten Zusammenhang mit der zu erklärenden Variable stehen. Die dazu wichtigsten Methoden seien kurz genannt:

- Die Auswahl von Variablen aufgrund eines oder mehrerer theoretischer Modelle.
- Die intuitive Auswahl von Variablen, die einem Fachmann relevant erscheinen.
- Die Auswahl von Variablen, die bei einer grafischen Analyse (z.B. Scatter-Plot, Box-Plot etc.) augenscheinlich einen Zusammenhang zu der abhängigen Variable aufweisen.
- Die Auswahl von Variablen mit Hilfe von statistischen Verfahren (z.B. Korrelationsanalyse o.ä.).

Wurde nun ein Pool von verschiedenen Größen ausgewählt, besteht der nächste Schritt in der Untersuchung der einzelnen Größen. Eine Darstellung der dazu notwendigen Methoden würde den Rahmen der vorliegenden Arbeit sprengen; sie sind in verschiedenen ökonometrischen Lehrbüchern dargestellt (z.B. Greene (1993), Judge et al. (1988), Davidson et al. (1993)). Stattdessen soll das Augenmerk auf einige wichtige Punkte gelenkt werden, die bei der Modellierung mit neuronalen Netzwerken auf keinen Fall außer acht gelassen werden dürfen.

Der erste Schritt für jeden Modellbilder sollte darin bestehen, sich über die Qualität seiner Daten zu informieren. Obwohl dies eine triviale Feststellung scheint, wird dieser Schritt in vielen Fällen unterlassen. Die beste Möglichkeit, sich mit den zur Verfügung stehenden Daten vertraut zu machen, besteht in einer grafischen Analyse. Dabei geben vor allen Dingen

Zeitreihenplots Auskunft über technische Datenprobleme. Durch sie lassen sich beispielsweise externe Schocks oder Strukturbrüche in den Beobachtungen erkennen, die auch bei neuronalen Netzwerken mit Hilfe von *Dummies* modelliert werden müssen. Darüber hinaus können Zeitreihenplots erste Anzeichen von zu erwartenden Schwierigkeiten bei der Modellbildung (z.B. Heteroskedastizität) offenbaren.

Integration/Kointegration

Handelt es sich bei den verwendeten Größen um nicht-stationäre Zeitreihen, müssen diese erst entsprechend differenziert werden,[24] bevor sie als Input- bzw. Zielgrößen für ein neuronales Netzwerk benutzt werden können. Anderenfalls lassen sich gute Approximationen schon allein aufgrund eines gemeinsamen Zeittrends erreichen, i.e. eine *spurious regression* (Granger/Newbold (1974)). Eine Ausnahme besteht für den Fall, daß man mit einem Netzwerk eine (nichtlineare) Kointegrationsbeziehung modellieren will.[25]

Multikollinearitäten

Eine weitere Schwierigkeit besteht in eventuell vorhandenen gegenseitigen Abhängigkeiten unter den erklärenden Variablen. Im einfachsten Fall sind die Beobachtungen der erklärenden Größen linear abhängig. Diese sollten dann um die redundanten Größen vermindert werden.[26] Weit öfter kommt jedoch der Fall vor, in dem die erkärenden Variablen nicht exakt linear abhängig sind, sondern hohe Multikollinearitäten untereinander aufweisen. Zur Aufdeckung dieser Multikollinearitäten existiert keine vereinheitlichte Vorgehensweise. Die verläßlichste Methode (Judge et al., 1988) besteht jedoch in einer Varianzanalyse der Beobachtungsmatrix X durch die Berechung der Eigenwerte der Matrix $X'X$ bzw. durch die Eigenwertzerlegung der Matrix $X'X$ in

$$X'X = A\Lambda A'. \tag{50}$$

Λ ist eine Diagonalmatrix, die auf ihrer Diagonale die Eigenwerte λ_k der Matrix $X'X$ enthält und A ist die Matrix der zugehörigen orthonormalen Eigenvektoren. Die Anzahl der relativ kleinen Eigenwerte deutet auf die Anzahl der fast linearen Abhängigkeiten unter den Variablen hin. Ist ein Eigenwert $\lambda_k = 0$, dann ist die Matrix $X'X$ singulär und es existieren exakte lineare Abhängigkeiten unter den Beobachtungen. Im linearen Modell wirken sich Multikollinearitäten besonders dann sehr negativ aus, wenn sie eine hohe Varianz bei der Koeffizientenschätzung einer Variable verursachen. Diese ergibt sich nämlich durch

$$\mathrm{Var}[\beta] = \sigma^2(X'X)^{-1} \tag{51}$$

bzw. für einen einzelnen Parameter durch

$$\mathrm{Var}[\hat{\beta}_i^{\mathrm{OLS}}] = \sigma^2 \left(\frac{a_{i1}^2}{\lambda_1} + \ldots + \frac{a_{iK}^2}{\lambda_K} \right) \tag{52}$$

[24] Vgl. Mills (1993).
[25] Vgl. Steurer (1994).
[26] Aus der Perspektive eines theoretischen Modells ist diese Vorgehensweise natürlich abzulehnen.

und wird immer dann sehr hoch, wenn einer der Brüche in Gleichung (52) sehr groß wird. Die a_{ik} stellen dabei die Elemente der Matrix A dar. Dieser Fall des linearen Regressionsmodells läßt sich jedoch nicht ohne weiteres auf neuronale Netzwerke übertragen, da die Parameter des Netzwerks nicht eindeutig einer bestimmten unabhängigen Variablen zugeordnet werden können. In neuronalen Netzwerken können Multikollinearitäten dazu führen, daß die Schätzverfahren zweiter Ordnung nicht mehr konvergieren, da sich die Hessematrix nur dann invertieren läßt, wenn sie regulär ist. Dies ist aber bei multikollinearen Beobachtungen nicht gewährleistet. Die einfachste Vorgehensweise, der Multikollinearität in Daten zu begegnen, besteht darin, die verantwortliche Variable aus der Regression herauszunehmen. Unter der Annahme, daß das ursprüngliche Modell korrekt spezifiziert war, verursacht dies jedoch sowohl im linearen als auch in nichtlinearen Modell, einen *omitted variable bias* in der Schätzung.

Nicht beachtete Variablen

Ein Modell-*Bias* existiert auch dann, wenn man eine relevante Variable in einer beliebigen Regression versehentlich nicht berücksichtigt hat (*omitted variable*) oder nicht beobachten konnte (latente Variable). Im letzteren Fall kann man versuchen, den Effekt der latenten Variable mittels einer beobachtbaren Variable[27] (*Proxy*) wiederzugeben.

Lineares Modell als Ausgangsbasis

Ein guter Ansatz, die Modellierung eines unbekannten Zusammenhangs mit einem neuronalen Netzwerk zu beginnen, besteht darin, zunächst ein einfaches lineares Regressionsmodell aufzustellen. Die Schätzung dieses linearen Modells hat als Ausgangsbasis folgende Vorteile:

- Die Analyse des linearen Regressionsmodells und der zugehörigen Residuen läßt erste Schlüsse über mögliche Heteroskedastizität oder Autokorrelation zu. Außerdem lassen sich eventuelle Modellinstabilitäten beispielsweise aufgrund eines oder mehrerer Strukturbrüche erkennen.

- Das lineare Regressionsmodell ermöglicht Tests auf vernachlässigte Nichtlinearitäten im Mittelwert[28], d.h. auf eine allgemeine Fehlspezifikation, die den Einsatz eines nichtlinearen Regressionsmodells bzw. eines neuronalen Netzwerks rechtfertigen.

- neuronale Netzwerke stellen als nichtlineare Regressionsmodelle eine Obermenge der linearen Regressionsmodelle dar. Die Schätzergebnisse des linearen Falls lassen sich beispielsweise bei der Initialisierung der Netzwerkgewichte weiterverwenden. Als Alternative läßt sich das lineare Regressionsmodell für eine erweiterte Regression, wie in Abbildung 8 dargestellt, benutzen.

In jedem Fall ist zunächst also eine Modellierung des betrachteten Zusammenhangs mittels eines linearen Regressionsmodells zu empfehlen, das darüber hinaus auch als Referenzmodell dienen kann, um die Qualität des Netzwerkmodells zu beurteilen.

[27] Diese Weg beschreitet man beispielsweise, wenn man Erwartungen von Marktteilnehmern verwenden möchte, die sich nicht direkt beobachten lassen.

[28] Vgl. Lee/White/Granger (1993).

Normierung

Im Prinzip ist damit die Identifikationsphase abgeschlossen. Bevor jedoch der nächste Schritt, die Modellspezifikation, betrachtet wird, soll noch eine kurze Anmerkung zu einem unter Netzwerkern weit verbreiteten Mythos gemacht werden. Dieser lautet, daß alle Größen, die in einem Netzwerk Verwendung finden, normiert werden müssen. Diese Aussage ist jedoch nicht grundsätzlich richtig. Lediglich wenn der Wertebereich der Aktivierungsfunktion der *output unit* beschränkt ist, muß eine Transformation der abhängigen Variable vorgenommen werden. Es scheint sich jedoch in der Literatur durchzusetzen, daß Nichtlinearitäten — sofern dies möglich ist — ausschließlich in den *hidden layers* eines Netzwerks modelliert werden, so daß die *output unit* in den meisten Fällen eine lineare Transformationsfunktion erhält.[29] In diesem Fall ist keine Normierung der verwendeten Variablen notwendig. Aus zwei Gründen bietet sich trotzdem eine Normierung der Daten an:

1. Die Konvergenz des numerischen Schätzverfahrens ist bei normierten Variablen im allgemeinen besser.
2. Es ergeben sich bestimmte Vorteile bei der Sensitivitäts- und Relevanzanalyse eines Netzwerks, auf die im Abschnitt 3.4 eingegangen wird.

Die in der Literatur am häufigsten vorgenommene Normierung der verwendeten Variablen ist eine lineare Transformation auf ein bestimmtes Intervall, also beispielsweise auf das Intervall $[-1;1]$:

$$\xi_t^{[-1;1]} = 2\frac{x_t - x_{\min}}{x_{\max} - x_{\min}} - 1. \tag{53}$$

Eine nach Ansicht des Verfassers wesentlich geeignetere Transformation liegt jedoch in der Bereinigung der Variablen um Mittelwert und Varianz, d.h.:

$$\xi_t = \frac{(x_t - \bar{x})}{\sigma_x}. \tag{54}$$

Die Mittelwert–Varianz–Bereinigung hat den Vorteil, daß sie Korrelationsstrukturen unter den Variablen unverändert läßt. Darüber hinaus liegt der Mittelwert aller Variablen bei Null, ein wesentlicher Vorteil für die noch zu betrachtende Relevanz- und Sensitivitätsanalyse.

3.2 Netzwerkspezifikation

Die Suche nach der bestmöglichen Netzwerkarchitektur ist mit Sicherheit der schwierigste der vier Modellierungsschritte. Die Netzwerkarchitektur ist letztendlich dafür verantwortlich, wie gut eine unbekannte Funktion approximiert werden kann und wie gut die daraus

[29] Eine Ausnahme besteht für den Fall, daß die abhängige Variable beschränkt ist, z.B. wenn Wahrscheinlichkeiten modelliert werden sollen. In diesem Fall wählt man als Aktivierungsfunktion der *output unit* die logistische Funktion.

resultierenden Vorhersagen sind. Dementsprechend viel Zeit sollte in die Spezifikation einer geeigneten Architektur investiert werden. Diese Aufgabe ist jedoch extrem schwierig. Obwohl sie den zentralen Schritt im Modellierungsprozeß darstellt, wird ihr in der Literatur neuronaler Netzwerke auch nicht ansatzweise soviel Bedeutung beigemessen, wie sie verdient hätte. Stattdessen wurden eine Reihe von Algorithmen entwickelt, die die beste Architektur automatisch bestimmen sollen, um die Suchkosten gering zu halten (z.B. *Cascade Correlation* (Fahlman, 1990)). Viele dieser Algorithmen führen die Suche nach einer geeigneten Netzwerkarchitektur dabei simultan zu der Bestimmung der Parameter des entsprechenden Netzwerks aus. Je nach verwendeter Methode kann eine solche Strategie unter Umständen zu einer Verzerrung der Modellschätzung führen bzw. zu einer schlecht parametrisierten Netzwerkarchitektur. Dies sollte vor der Anwendung einer solchen Methode beachtet werden.

Grundsätzlich müssen zur Bestimmung einer Netzwerkarchitektur zwei Fragen beantwortet werden:[30]

- Wieviele *hidden units* in wievielen *hidden layers* sind für die Approximation der gesuchten Funktion notwendig?
- Wie sollen die *units* des Netzwerks untereinander vernetzt werden?

Die beiden Fragen lassen sich nicht unabhängig voneinander beantworten, denn letztendlich sind beide Antworten für die Bestimmung der freien Parameter des Netzwerks verantwortlich. Bei der Beantwortung der Fragen muß man vor allen Dingen die Erkenntnis der linearen Regressionstheorie berücksichtigen: diese besagt, daß sich eine beliebig gute Approximation erreichen läßt, wenn man zum einen ein Modell mit entsprechend vielen frei variierbaren Parametern ausstattet, zum anderen eine beliebig hohe Zahl an erklärenden Größen zuläßt.

Daumenregeln

Weniger eine Spezifikationsstrategie als vielmehr ein Hilfsmittel für eine erste Netzwerkspezifikation sind die sogenannten Daumenregeln. Eine häufig zitierte Daumenregel schlägt z.B. vor, als Anzahl von *hidden units* den (geometrischen) Mittelwert von *input* und *output units* zu wählen. Jedoch tragen Daumenregeln nicht der Tatsache Rechnung, daß zur Approximation von verschieden komplexen Funktionen verschieden viele *hidden units* benötigt werden. Darüber hinaus hängt die benötigte Anzahl der *hidden units* zusätzlich von der Anzahl der zur Verfügung stehenden Beobachtungen, der relativen Stärke des Rauschens in den Beobachtungen und dem Typ der verwendeten Aktivierungsfunktion ab. Daumenregeln liefern also im allgemeinen nur einen sehr vagen Anhaltspunkt für die Bestimmung der Anzahl an *hidden units* und werden deshalb an dieser Stelle keine weitere Betrachtung finden.

Informations- und Selektionskriterien

Wesentlich objektiver sind die sogenannten Informationskriterien. Sie setzen in der Regel die quadrierten Residuen eines Modells zu der Anzahl der freien Parameter dieses Modell

[30] Es wird angenommen, daß alle Nichtlinearitäten innerhalb der *hidden layers* modelliert werden und daß die Transformationsfunktion der *output unit* linear ist.

in Beziehung. Die Intention besteht darin, den Fehler eines Modells gegen die Zahl seiner Parameter abzuwägen. Ein zusätzlicher Parameter sollte nur dann in ein Modell aufgenommen werden, wenn dadurch das entsprechende Informationskriterium einen geringeren Wert annimmt, d.h. die Summe der quadrierten Residuen entsprechend stark reduziert wird. Unter verschiedenen miteinander konkurrierenden Modellen wird dasjenige gewählt, das den kleinsten Wert des entsprechenden Informationskriteriums besitzt.

Die beiden bekanntesten Informationskriterien sind das Akaike (1973) Informationskriterium (AIC)

$$\text{AIC} = \ln(\frac{u'u}{T}) + \frac{2K}{T} \tag{55}$$

und das Schwarz (1978) Informationskriterium (SIC)

$$\text{SIC} = \ln(\frac{u'u}{T}) + \frac{K\ln(T)}{T} \tag{56}$$

Die beiden Kriterien ähneln sich sehr, jedoch bestraft das SIC — wie sich aus den beiden Gleichungen erkennen läßt — zusätzliche Parameter in Abhängigkeit der Anzahl an Beobachtungen wesentlich stärker. Das SIC bevorzugt im Vergleich mit dem AIC also sparsamer parametrisierte Modelle.

Neben diesen beiden Kriterien scheint sich im Netzwerkbereich ein neues Kriterium zu etablieren, das Netzwerk-Informationskriterium (NIC) (Murata/Yoshizawa/Amari, 1994). Das NIC ist eine Verallgemeinerung des AIC für den Fall, daß sich die wahre Funktion F mit dem gewählten Modell bzw. Netzwerk nicht exakt abbilden läßt. In diesem Fall spricht man von fehlspezifizierten Modellen.[31] Das NIC lautet nun für den Fall der kleinsten-Quadrate-Schätzung 26:[32]

$$\text{NIC} = \text{MSE} + \frac{1}{T} \cdot \text{tr}[BA^{-1}], \tag{57}$$

wobei die Matrizen A und B folgendermaßen definiert sind

$$A = E\left[\frac{\partial^2 \text{SE}(w)}{\partial w \partial w'}\right] \text{ und } B = \text{Var}\left[\frac{\partial \text{SE}(w)}{\partial w}\right] \tag{58}$$

Mit den Formeln in Gleichung (80) können die Matrizen A und B konsistent geschätzt werden. Für den Fall, daß das Netzwerk die wahre Funktion F exakt abbilden kann, also nicht fehlspezifiziert ist, läßt sich zwischen den Matrizen die asymptotische Beziehung $B = 2\sigma^2 A$ herleiten, so daß $\text{tr}[BA^{-1}] = 2\sigma^2 \text{tr}[I] = 2\sigma^2 K$ gilt. Damit reduziert sich das NIC auf die von Amemiya (1980) angegebene alternative Berechnungsformel des AIC:

$$\text{AIC} = \text{MSE} + \sigma^2 \frac{2K}{T} \tag{59}$$

[31] Vgl. White (1994).

[32] Für den Fall der Anwendung von *Maximum Likelihood* vgl. Murata/Yoshizawa/Amari (1994)

An dieser Stelle soll darauf hingewiesen werden, daß die Anwendung von Informationskriterien auf neuronale Netzwerke theoretisch nur dann möglich ist, wenn die Netzwerke keine irrelevanten *hidden units* besitzen. Leider ist dies ein Widerspruch in sich, denn mittels der Informationskriterien versucht man ja gerade irrelevante *hidden units* zu identifizieren, um sie aus dem Netzwerk zu entfernen. In der Literatur wird dieses Problem in den meisten Fällen jedoch ignoriert und die Informationskriterien dennoch benutzt.[33] Ein Möglichkeit zur statistisch sauberen Anwendung von Informationskriterien stellen Anders/Korn (1996) dar.

Neben den Informationskriterien gibt es eine Reihe weiterer Kriterien, die zur Modellselektion eingesetzt werden. Das justierte Bestimmtheitsmaß $\bar{R}^2$ versucht den *Goodness of Fit* der Regression zu messen:

$$\bar{R}^2 = 1 - \frac{T-1}{T-K} \sum_{t=1}^{T} \frac{(y_t - \hat{y}_t)^2}{(y_t - \bar{y}_t)^2}. \tag{60}$$

Es bestraft die Freiheitsgrade K des Modells jedoch nicht so stark wie beispielsweise das SIC.[34]

Als weitere Selektionskriterien werden häufig auch Amemiyas *Prediction Criterion* (PC)

$$\text{PC} = 1 - \frac{u'u}{T-K}\left(1 + \frac{K}{T}\right) \tag{61}$$

und Akaikes *Final Prediction Error* (FPE)

$$\text{FPE} = \frac{T+K}{T-K}\,\text{MSE} \tag{62}$$

herangezogen, die beide versuchen, den zu erwartenden Vorhersagefehler zu bestimmen.

Ein unbestreitbarer Vorteil der Informations- und Selektionskriterien besteht darin, daß sie sich sehr schnell berechnen lassen. Für eine zuverlässige Aussage benötigen sie üblicherweise jedoch große Stichproben. Daher soll darauf hingewiesen werden, daß die Informationskriterien in finiten Stichproben lediglich heuristische Ansätze darstellen, und ihre Nützlichkeit im praktischen Einsatz im wesentlichen mit Hilfe von Monte-Carlo-Simulationen demonstriert wurde (Judge et al., 1988). Grundsätzlich sollten die Informations- und Selektionskriterien also nicht alleinige Grundlage für die Wahl einer Netzwerkarchitektur bleiben.

Regularisierung

Eine zu den Informationskriterien verwandte Möglichkeit, eine geeignete Netzwerkarchitektur zu wählen, besteht in der Regularisierung neuronaler Netzwerke. Anstatt — wie bei den Informationskriterien — am Ende des Schätzprozesses die quadrierten Residuen zu der Anzahl der freien Parameter des Modells in Beziehung zu setzen, minimiert man bei der Regularisierung simultan die Summe der quadrierten Fehler sowie einen Wert, der die

[33] Vgl. Swanson/White (1995).

[34] Vgl. Judge/Griffiths/Hill/Lütkepohl/Lee (1988).

Komplexität des Netzwerk bestraft. Man minimiert also nicht mehr nur die herkömmliche Zielfunktion (z.B. SSE), sondern die Summe aus der Zielfunktion und einer Funktion, die die Komplexität des Netzwerks bestraft. Letztere soll im folgenden *Network Complexity Penalty* (NCP) genannt werden soll. Die NCP–Funktion eines Netzwerks stellt üblicherweise eine Funktion der Netzwerkgewichte dar und entspricht damit einem informativen bayesianischen Prior.[35] Die regularisierte Zielfunktion lautet im Fall der quadratischen Abweichungen also

$$q(w) = \mathrm{SSE}(w) + \lambda \cdot \mathrm{NCP}(w). \tag{63}$$

Der Parameter λ stellt eine Gewichtung der Straffunktion dar. Die beiden bekanntesten Regularisierungstechniken heißen *Weight Decay* und *Weight Elimination*. Die NCP–Funktion des *Weight Decay* bestraft die Existenz absolut hoher Gewichte in einem Netzwerk und lautet

$$\mathrm{NCP}_{\mathrm{decay}} = \sum w_{kj}^2. \tag{64}$$

Die *Weight Decay*–Funktion ist in der statistischen Literatur als *Ridge Regression* bekannt. Eine Alternative zu dieser NCP-Funktion ist die des *Weight Elimination*.[36] Diese bestraft die Anzahl der Gewichte eines Netzwerks, indem sie kleinen Gewichten eine Tendenz gibt, auf Null hin abzunehmen:

$$\mathrm{NCP}_{\mathrm{elim}} = \sum \frac{w_{kj}^2}{1 + w_{kj}^2} \tag{65}$$

Es ist offensichtlich, daß mit der erweiterten Zielfunktionen (63) ein unnötiger *Bias* in das System eingeführt wird. Aus diesem Grund sollte das Netzwerk, das am Ende der Regularisierung resultiert, nur als Grundlage für eine Architekturwahl verstanden werden. Die durch das Regularisierungsverfahren als irrelevant identifizierten Gewichte bzw. *hidden units* müssen aus dem Netzwerk entfernt und das reduzierte Netzwerk dann mit Hilfe der nicht regularisierten Zielfunktionen neu trainiert werden.

Pruning

Eine alternative Verfahrensweise zur Entfernung von Gewichten eines neuronalen Netzwerks sind die sogenannten *Pruning*–Techniken. Sie versuchen redundante Gewichte dadurch zu identifizieren, daß sie die Änderungen im *Mean Squared Error* errechnen, die sich bei Herausnahme eines oder mehrerer Gewichte in einem bereits trainierten Netzwerk ergeben. Ändert sich der *Mean Squared Error* nicht deutlich, dann wird das Gewicht aus dem Netzwerk enfernt. Die beiden bekanntesten Verfahren heißen *Optimal Brain Damage* und *Optimal Brain Surgeon*. Sie unterscheiden sich lediglich in der Komplexität ihrer Berechnung. Im Prinzip ähneln sie der Vorgehensweise eines Wald-Tests, jedoch basieren sie eher auf heuristischen Überlegungen als auf einer statistischen Verteilungstheorie.

[35] Vgl. Neal (1985).
[36] Vgl. Hertz/Krogh/Palmer (1991).

Ihr Einsatz stellt dennoch ein geeignetes Hilfsmittel dar, um Netzwerke um redundante Parameter zu bereinigen. Für eine weiterführende Diskussion sei auf Zell (1994) verwiesen.

Cross Validation

Ein häufig für neuronale Netzwerke und nichtparametrische Modelle eingesetztes Verfahren ist das *Cross Validation* (CV) (z.B. *v-leave-out*).[37] Mit Hilfe dieses Verfahrens versucht man, den zu erwartenden Fehler bezüglich unbekannter Daten zu prognostizieren. Unter mehreren konkurrierenden Modellen wird dasjenige gewählt, das den kleinsten Vorhersagefehler erwarten läßt. Zur Schätzung dieses Fehlers unterteilt man die Menge aller Beobachtungen in eine Anzahl von M disjunkten Teilmengen mit jeweils v Beobachtungen, wobei für den Parameter v häufig $v = 1$ gewählt wird. Man betrachtet nun sukzessive jeweils eine dieser Teilmengen als Validierungsmenge und schätzt die Parameter eines Modells, ohne die Beobachtungen dieser Menge zu benutzen. Die durchschnittlichen quadratischen Fehler MSPE_m, die sich jeweils bezüglich der einzelnen Validierungsmengen ergeben, werden gemittelt und dienen als Schätzer für den erwarteten Fehler einer Vorhersage. Der *Cross Validation*-Fehler berechnet sich also durch

$$\mathrm{CV} = \frac{1}{M} \sum_{m=1}^{M} \mathrm{MSPE}_m \,. \tag{66}$$

Im Gegensatz zu den Informationskriterien ist das *Cross Validation*-Verfahren sehr rechenintensiv, denn die Parameterschätzung des Modells müssen jeweils ohne die disjunkten Mengen wiederholt werden. Die Eignung von *Cross Validation* zur Modellselektion neuronaler Netzwerke wird von Anders/Korn (1996) untersucht.

Stopped Training

In vielen Fällen wird eine in der Literatur der neuronalen Netzwerke weit verbreitet Methode eingesetzt, die dem *Cross Validation* sehr ähnlich ist. Auch diese Methode wird häufig mit *Cross Validation* bezeichnet, soll aber aus Gründen der Verwechslungsmöglichkeit mit dem Verfahren des *v-leave-out Cross Validation* in dieser Arbeit mit *Stopped Training* bezeichnet werden. Bei diesem Verfahren wird die zur Verfügung stehende Datenmenge in zwei Mengen, eine Trainingsmenge und eine Validierungsmenge, aufgespaltet. Die in der Literatur dargestellte Verwendung dieser Methode besteht darin, in dem iterativen Schätzprozeß simultan die quadratischen Fehler der Abweichungen sowohl bezüglich der Trainings- als auch bezüglich der Validierungsmenge zu berechnen. Die Iteration wird abgebrochen, wenn die Summe der quadratischen Fehler bzgl. der Validierungsmenge ihr Minimum erreicht hat (Abbildung 13).

[37] Vgl. Craven/Wahba (1979), Härdle (1990).

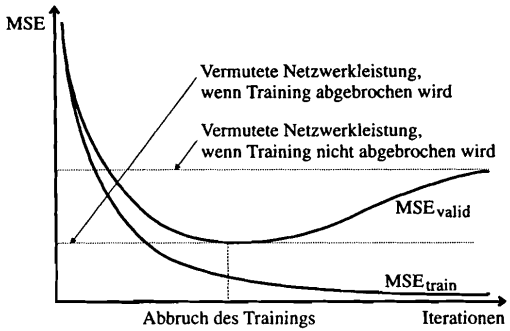

Abbildung 13: Die Methode des *Stopped Training*.

Die Intention, die hinter *Stopped Training* steckt, besteht darin, das Training dann abzubrechen, wenn das Netzwerk offensichtlich beginnt, die Störterme zu approximieren, d.h. das Rauschen erlernen zu wollen (*overfitting*). Obwohl diese Vorgehensweise auf den ersten Blick einsichtig erscheint, hat sie zwei entscheidende Nachteile und ist daher wenig empfehlenswert. Zunächst ist klar, daß ein Netzwerk, das in der Lage ist, Rauschen zu approximieren, überparametrisiert ist. Um also die gewünschte Funktion approximieren zu können, sind vergleichsweise weniger Parameter notwendig als in dem Netzwerk verwendet wurden. Die Konsequenz daraus ist, die Anzahl der freien Parameter des Netzwerks zu reduzieren und nicht, den Schätzprozeß zu einem früheren Zeitpunkt, also an einem suboptimalen Punkt der Fehlerfunktion, abzubrechen. Der zweite erhebliche Nachteil besteht darin, daß der Zeitpunkt des Abbruchs sehr stark von der Wahl der Trainings- und Validierungsmenge abhängt, vor allen Dingen dann, wenn Trainings- und Validierungsmenge durch einen Zufallsmechanismus bestimmt worden sind.

Die Verwendung der *Stopped Training*-Methode findet dagegen auf eine andere Weise Berechtigung. Mittels ihrer Anwendung kann man nämlich eine heuristische Aussage treffen, ob ein neuronales Netzwerk über- oder unterparametrisiert ist. In einem auf Grundlage der Trainigsmenge geschätzten Netzwerk sollten sich nämlich kleine Variationen der optimalen Gewichte des Netzwerks in der gleichen Weise auf die quadrierten Fehler bezüglich der Trainings- und der Validierungsmenge auswirken. Die Variation der Gewichte entspricht nämlich einer kleinen Änderung des Verlaufs der Netzwerkfunktion. Wird also die wahre Funktion nach Variation der optimalen Gewichte nicht mehr so gut approximiert wie bei Verwendung der optimalen Gewichte, sollte sich dies bezüglich der quadrierten Fehlersummen beider Mengen dokumentieren. Die Voraussetzung für das Funktionieren dieser Heuristik besteht darin, zwei Teilmenge zu schaffen, die beide die Grundgesamtheit repräsentieren. Weigend et al. (1990) schlagen dazu vor, zwei Mengen mit gleichem Mittelwert und gleicher Varianz zu generieren. De Groot (1993) entwickelt einen *Clustering*-Algorithmus, der soviele *Cluster* generiert wie Beobachtungen für die Validierungsmenge vorgesehen sind. Die *Cluster*-Zentren stellen dann die Validierungsmenge dar. Snee (1977) schlägt mit seinem DUPLEX-Algorithmus ein Verfahren vor, das die Grundgesamtheit so aufteilt, daß beide Teilmengen denselben Raum überdecken.

3.3 Schätzung

Ist nun eine erste Netzwerkarchitektur mittels eines oder mehrerer Verfahren des letzten Abschnitts spezifiziert worden, müssen im dritten Schritt die Parameter bzw. Gewichte des Netzwerks bestimmt werden. Dazu bedient man sich der bekannten und in Abschnitt 2.6 dargestellten numerischen Schätzverfahren für nichtlineare Regressionsmodelle. Der sogenannte Trainingsprozeß eines neuronalen Netzwerks stellt also nichts anderes dar als die Anwendung eines nichtlinearen Optimierungsverfahrens auf die entsprechende Zielfunktion. Die potentiellen Probleme, die bei Anwendung solcher Verfahren auftreten können, wurden bereits in Abschnitt 2.6.9 beschrieben. Deshalb soll an dieser Stelle nur noch die für Netzwerke spezifische Vorgehensweise dargestellt werden.

Durchführung der Iteration

Es hat sich mittlerweile als Konvention eingebürgert, den Iterationsprozeß mit einer zufälligen Initialisierung der Gewichte zu beginnen, wobei die Gewichtswerte aus einem kleinen Intervall um Null gezogen werden. Dies kann dazu führen, daß der Iterationsprozeß entweder nicht konvergiert oder aber in einem lokalen Minimum stecken bleibt, das auf einem wesentlich höheren Niveau liegt als das globale Minimum. Aus diesem Grund sollte der Iterationsprozeß von verschiedenen Startwerten begonnen und die Ergebnisse miteinander verglichen werden. Die Parameter der Netzwerkarchitektur, die den kleinsten *Mean Squared Error* des Modells ergeben, sind zu bevorzugen.

Die verschiedenen Abstiegsverfahren haben unterschiedliche Konvergenzeigenschaften, die sowohl von der Größe des gewählten Netzwerks abhängen als auch von der Qualität der Daten. Es wird also auch hier angeraten, entweder verschiedene Algorithmen für das Training des Netzwerks zu verwenden oder eine Kombinationen von verschiedenen Verfahren, wie sie beispielsweise in Abschnitt 2.6.9 vorgeschlagen wurde. In jedem Fall sollte die Konvergenz des benutzten Verfahrens nach seinem Abbruch verifiziert werden. Im lokalen Minimium muß der Gradient der Netzwerkfunktion dem Nullvektor entsprechen und die Hessematrix positiv definit sein.

Gradient und Hessematrix

Ein Vorteil neuronaler Netzwerke besteht darin, daß sich Gradient und Hessematrix der Netzwerkfunktion für alle *Feedforward*-Netzwerkarchitekturen im Prinzip gleichen und daher im vorhinein analytisch bestimmt werden können. Gradient und Hessematrix stehen damit den Optimierungsverfahren der ersten und zweiten Ordnung zur Verfügung. Es entfällt also die jeweilige analytische Bestimmung bzw. numerische Approximation von Gradient und Hessematrix, die bei allen Zielfunktionen spezifischer Problemstellungen notwendig ist.

Für das Netzwerk aus Abbildung 1 soll der Gradient beispielhaft bestimmt werden. Dabei wird der Übersichtlichkeit halber der Parameter t, der die beobachteten Variablen indexieren müßte, unterdrückt. Mit der Netzwerkfunktion

$$f(X,w) = X\alpha + \sum_{h=I+1}^{I+H} \beta_h g(\sum_{i=0}^{I} \gamma_{hi} x_i) \tag{67}$$

ergibt sich die partielle erste Ableitungen der Fehlerfunktion SSE nach einem Gewicht β_h des *hidden layers* durch

$$\frac{\partial \mathrm{SSE}}{\partial \beta_h} = \sum_t -2[y - f(X, w)]s_h, \tag{68}$$

wobei für s_h gilt:

$$s_h = g_h(\sum_{i=0}^{I} \gamma_{hi} x_i). \tag{69}$$

Die partielle erste Ableitung der Fehlerfunktion SSE nach einem Gewicht γ_{hi} des *input layers* ergibt sich aus:

$$\frac{\partial \mathrm{SSE}}{\partial \gamma_{hi}} = \sum_t -2[y - f(X, w)]\beta_i g_h'(r_h)x_i, \tag{70}$$

wobei für r_h gilt:

$$r_h = \sum_{i=0}^{I} \gamma_{hi} x_i. \tag{71}$$

Die Berechnung der partiellen Ableitungen zweiter Ordnung kann entsprechend weitergeführt werden.

Monitoring

In der Literatur wird in vielen Anwendungen ein *Monitoring* des Iterationsverfahrens durchgeführt.[38] Es besteht darin, die Entwicklung des *Mean Squared Error*, der Gewichte sowie der Korrelationsstrukturen zwischen den Ausgangssignalen der *hidden units* während des Iterationsprozesses mitzuverfolgen. Mit Hilfe des *Monitoring* lassen sich jedoch ausschließlich qualitative Aussagen machen, denn die Beobachtungen während des Iterationsprozesses ergeben sich alle an suboptimalen Punkten im Fehlergebirge des neuronalen Netzwerks. Ihre Relevanz für den optimalen Punkt im Fehlergebirge muß daher in Frage gestellt werden. Dennoch läßt sich unter Umständen erkennen, daß beispielsweise ein Algorithmus nur unzureichend konvergiert oder daß die Parameterwerte eines Netzwerks sehr stark oszillieren. Die erste Konsequenz daraus ist, den Iterationsprozeß mit anderen Startwerten neu zu beginnen bzw. ein anderes Optimierungsverfahren zu wählen. Die zweite Konsequenz besteht in der Wahl einer anderen Netzwerkarchitektur.

3.4 Diagnose

Wesentlich wichtiger als die Beobachtung des Iterationsverfahrens ist die eingehende Diagnose des Netzwerks, nachdem die Parameter des Netzwerks geschätzt und das lokale Optimum verifiziert worden ist.

[38] Vgl. beispielsweise de Groot (1993), Weigend et al. (1990), Zimmermann et al. (1991).

Im Vergleich zu einem linearen Modell ist die Durchführung von Signifikanztests einzelner Parameter in einem neuronalen Netzwerk wesentlich aufwendiger. Darüber hinaus kann ihnen im Fall von neuronalen Netzwerken auch nicht annähernd die Bedeutung beigemessen werden, die sie im linearen Modell haben. Ist nämlich in einem linearen Modell ein Parameter nicht signifikant von Null verschieden, dann geht man davon aus, daß die zugehörige erklärende Variable keinen Einfluß auf die zu erklärende Variable ausübt. In einem neuronalen Netzwerk bedeutet ein nicht signifikant von Null verschiedener Parameter lediglich, daß die zugehörige Verbindung in der spezifizierten Netzwerkstruktur unnötig ist. Eine Aussage über die Wirkungszusammenhänge der Variablen läßt sich damit also nicht treffen. Aus diesem Grund gewinnt bei der Diagnose neuronaler Netzwerke die Residualanalyse sowie die Untersuchung der Input–Output–Beziehungen ein ungleich stärkeres Gewicht als Parametertests.

Residualanalyse

Die Residualanalyse beginnt mit einem Plot der Residuen. Damit lassen sich meist unmittelbar Ausreißer in den Beobachtungen sowie eventuell vorhandene Strukturen (z.B. Heteroskedastie, Autokorrelation) in den Störtermen erkennen. Der Grund für die Untersuchung der Residuen besteht darin, festzustellen, ob das Modell grundsätzlich eine gute Erklärung für die Vergangenheit liefert und daher potentiell gute Vorhersagen für die Zukunft ermöglicht. Es ist klar, daß Vorhersagen nur dann bestmöglich werden, wenn alle relevanten Informationen in einem Modell berücksichtigt wurden und keine unbeachteten Informationen in den Residuen zurückbleiben.

Die grafische Analyse der Residuen läßt sich mit verschiedenen Tests untermauern. Die wichtigsten Tests sollen kurz genannt werden. Für eine nähere Erläuterung ihrer Verwendung und Funktion muß jedoch auf ein Ökonometrie–Lehrbuch verwiesen werden (z.B. Judge et al., 1988). Tests auf Autokorrelation sind der Durbin–Watson–Test, der Ljung–Box– bzw. der Box–Pierce–Test sowie der LM–Test von Breusch–Godfrey. Heteroskedastie läßt sich u.a. mit dem White–Test oder dem LM–Test von Breusch–Pagan erkennen. Daneben existieren noch eine Reihe weiterer genereller Spezifikationstests, mit denen sich häufig jedoch nur feststellen läßt, daß in den Residuen noch nicht erklärte Struktur vorhanden ist (z.B. der BDS–Test).

Da die Parameter neuronaler Netzwerke üblicherweise mit Hilfe der kleinsten–Quadrate–Methode bestimmt werden, sind sie sehr sensitiv gegen Ausreißer in den Beobachtungen. Wenige große Ausreißer können den Funktionsverlauf der Netzwerkregression vollständig verzerren, so daß bei der Schätzung mit neuronalen Netzwerken eine Bereinigung um Ausreißer in den Beobachtungen notwendig ist. Ein Indiz für Ausreißer läßt sich mit Hilfe des Normalverteilungstests von Bera–Jarque gewinnen. Existieren viele Ausreißer unter den Residuen, sind letzere üblicherweise nicht normalverteilt.

Sind keine Ausreißer in den Residuen zu erkennen, dafür aber bestimmte Strukturen, z.B. Autokorrelation oder Heteroskedastie, sind zwei Schlüsse zulässig:

1. Das Netzwerk ist unzureichend spezifiziert, d.h. die Netzwerkarchitektur ist schlecht gewählt, die Variablen–*lags* bei Zeitreihenmodellen wurden falsch gewählt, notwendige Variablen wurden nicht benutzt (*omitted variables*) oder durchgeführte Datentransformationen waren unzulässig.

2. Die Annahme unabhängiger Störterme des einfachen Regressionsmodells in Gleichung (18) ist verletzt, d.h. $E[\varepsilon\varepsilon'] = \Omega \neq \sigma^2 I$.

Zwei–Stufen–Schätzung

Führt die Überprüfung des ersten Falls zu keiner wesentlichen Änderung der Struktureffekte in den Residuen, sollte das Netzwerk neu geschätzt werden, wobei jedoch die Kovarianzen Ω in den Residuen zu berücksichtigen sind. Für diesen Fall wird in dieser Arbeit erstmals die Anwendung eines Zwei–Stufen–Schätzverfahrens für neuronale Netzwerke vorgeschlagen:

Im ersten Schritt werden die Gewichte $\hat{w}_{(1)}$ der Netzwerk–Regression

$$y = f(X, \hat{w}_{(1)}) + u_{(1)} \tag{72}$$

geschätzt. Besitzt das Netzwerk keine irrelevanten *units*, ist die Parameterschätzung der Gewichte $\hat{w}_{(1)}$ konsistent. Folglich sind auch die Residuen $u_{(1)}$ der Regression konsistent. Im zweiten Schritt erklärt man nun die durch die quadrierten Residuen $u^2_{(1)}$ approximierten Varianzen der Netzwerk–Regression mittels eines geeigneten Modells. Die Wahl dieses Modells hängt davon ab, welche Annahmen über die Struktur in den Residuen $u^2_{(1)}$ gemacht werden. Im Fall von autokorrelierten Residuen der ersten Ordnung läßt sich die Kovarianzmatrix Ω der Residuen z.B. mittels der Methoden von Cochrane–Orcutt oder von Theil bestimmen. Im Fall von Heteroskedastie ist es sinnvoll, die Art der Heteroskedastie (*groupwise*, *functional*, (G)ARCH) zu ermitteln und die beobachteten Varianzen unter Umständen mittels weiterer exogener Variablen entsprechend zu modellieren.[39] Will man keine Annahmen über die Eigenschaften der beobachteten Varianzen machen, lassen sich die Varianzen wiederum mit einem neuronalen Netzwerk modellieren, indem

$$u^2_{(1)} = \tilde{f}(Z, \tilde{w}) + v \tag{73}$$

angenommen wird. Die Matrix Z bezeichnet unabhängige Variablen, die sich durchaus mit den Variablen X überschneiden dürfen, $\tilde{w}$ den Gewichtsvektor der Netzwerkfunktion $\tilde{f}$, v die Residuen dieser Regression. Wie auch immer die Varianzen der ersten Regression modelliert werden, das Ziel besteht in einer konsistenten Schätzung der Kovarianzmatrix Ω, die sich aus den Schätzwerten $\hat{u}^2_{(1)}$ zusammensetzen läßt.[40] Zum Ende des zweiten Schritts verwendet man die geschätzte Kovarianzmatrix $\hat{\Omega}$ für die erneute Parameterschätzung der ursprünglichen Netzwerk–Regression $f(X, w)$ aus Gleichung (72), indem man die Methode der gewichteten kleinsten Quadrate anwendet

$$\mathrm{SSE}(w) = [y - f(X, w)]'\hat{\Omega}^{-1}[y - f(X, w)] \rightarrow \mathrm{Min}, \tag{74}$$

wobei die Komponenten der Matrix $\hat{\Omega}^{-1}$ die Gewichte der quadratischen Abstände darstellen. Als Resultat der Minimierung erhält man einen neuen Schätzer $\hat{w}_{(2)}$ für die optimalen

[39] Vgl. Greene (1993).

[40] Im Fall von unabhängig verteilten Residuen, stehen die geschätzten Residuen $\hat{u}^2_{(1)}$ beispielsweise auf der Hauptdiagonalen der Kovarianzmatrix Ω.

Gewichte, der nicht nur konsistent, sondern auch asymptotisch effizient ist.[41] Je effizienter ein Schätzers ist, desto kleiner ist seine Varianz, desto besser ist seine Vorhersagequalität.

Relevanz- und Sensitivitätsanalyse

Bei der Diagnose neuronaler Netzwerke ist neben der Residualanalyse vor allen Dingen die Untersuchung der Relevanz der erklärenden Variablen notwendig. Während im linearen Modell dies direkt durch einen t-Test ermöglicht wird, muß man sich im nichtlinearen Modell und insbesondere bei neuronalen Netzwerken einiger Heuristiken bedienen. Diese bestehen im wesentlichen in einer Relevanz- und Sensitivitätsanalyse.

Die Relevanz, die eine unabhängige Variable zur Erklärung einer abhängigen Variable in einem Modell besitzt, sei als der Faktor definiert, um den der MSE einer Schätzung steigt, wenn eine betrachtete Variable auf ihren Mittelwert (dieser ist Null bei einer vorgenommenen Mittelwertbereinigung) gesetzt wird, d.h.

$$\mathrm{Rel}_i = \frac{\mathrm{MSE}_{X_i=\bar{X}_i}}{\mathrm{MSE}}. \tag{75}$$

Je höher dieser Wert liegt, desto wesentlicher ist der Erklärungsbeitrag, den die Variable im geschätzten Modell liefert. Liegt der Faktor bei $\mathrm{Rel}_i = 1$ oder sogar noch darunter, liefert die Variable keinen Erklärungsbeitrag in der geschätzten Netzwerkarchitektur und kann entfernt werden. Anschließend muß das reduzierte Netzwerk neu trainiert werden.

Eine weitere Möglichkeit, die Beziehung zwischen abhängiger und unabhängiger Variable, zu analysieren, besteht in der Berechnung der partiellen Ableitung erster Ordnung der Netzwerkfunktion nach der betrachteten unabhängigen Variablen. Für den linearen Fall $\hat{y} = X\hat{\alpha}$ ergibt sich als partielle Ableitung genau der Parameter $\hat{\alpha}_i$:

$$\frac{\partial \hat{y}}{\partial x_i} = \hat{\alpha}_i. \tag{76}$$

Für den nichtlinearen Fall $\hat{y} = f(X, \hat{w})$ mit $\hat{w} = (\hat{\alpha}', \hat{\beta}', \hat{\gamma}')'$ ist die partielle Ableitung der Regressionsfunktion f hingegen nicht ein einzelner Parameter, sondern wiederum eine Funktion. Für den Fall der Standard-Netzwerkfunktion aus Gleichung (67) ergibt sich beispielsweise:[42]

$$\frac{\partial \hat{y}}{\partial x_i} = \hat{\alpha}_i + \sum_{h=I+1}^{I+H} \hat{\beta}_h g_h'(r_h)\hat{\gamma}_{hi}. \tag{77}$$

Die partiellen Ableitungen der Netzwerkfunktion sind jeweils von allen erklärenden Variablen abhängig, so daß keine eindeutige Beziehung zwischen einer betrachteten unabhängigen und der abhängigen Variable extrahiert werden kann. Man kann sich jedoch dadurch behelfen, daß man alle unabhängigen Variablen außer der betrachteten auf ihren Mittelwert setzt (dieser ist bei mittelwertbereinigten Variablen Null) und den Zusammenhang

[41] Ein Schätzer ist asymptotisch effizient, wenn er asymptotisch normalverteilt ist und asymptotisch eine kleinere Varianz hat als jeder alternative asymptotisch normalverteilte Schätzer.

[42] Der Vektor r_h ist in Gleichung (71) definiert.

zwischen der abhängigen und der unabhängigen Variable unter dieser Restriktion analysiert. Da die erklärenden Variablen nach dem Vorschlag in Abschnitt 3.1 um ihren Mittelwert und ihre Standardabweichung bereinigt sind, variieren sie ungefähr im Intervall [-3;3]. Man kann nun die Intervallwerte einer Inputvariable gegen die Werte der partiell abgeleiteten Netzwerkfunktion auftragen, wobei alle nicht betrachteten erklärenden Größen auf ihren Mittelwert (Null) gesetzt werden. Auf diese Weise gewinnt man einen Eindruck, wie sensitiv die abhängige Variable auf kleine Änderungen der unabhängigen Variable an verschiedenen Stellen reagiert. Dazu analog läßt sich verfahren, wenn man anstelle der Sensitivität, d.h. der partiellen Ableitung der Netzwerkfunktion, die zugrundeliegende funktionale Form der Beziehung zwischen einer abhängigen und der unabhängigen Variable untersuchen möchte.

Signifikanztests für Parameter

Obwohl der Signifikanz von Parametern bei neuronalen Netzwerken nicht — wie oben erwähnt — dieselbe Bedeutung zukommt wie in linearen Modellen, läßt sie sich dennoch überprüfen. Um einen Hypothesentest bezüglich eines Schätzers durchführen zu können, muß man eine Aussage über dessen Verteilung oder zumindest über dessen asymptotische Verteilung machen können. Die Voraussetzung dafür ist der Nachweis der Konsistenz des Schätzers, was soviel bedeutet, daß der Schätzer mit zunehmender Beobachtungszahl T zu einem Punkt kollabiert. Damit ist der Schätzer eindeutig und identifiziert. Es ist nun aber offensichtlich, daß der Gewichtsvektor $\hat{w}$ eines Netzwerks aus mindestens zwei Gründen nicht eindeutig ist:

1. aufgrund von Symmetrien im neuronalen Netzwerk existiert eine bestimmte Anzahl von Permutationen der Gewichtswerte, die exakt dieselbe Regression ergeben;
2. irrelevante *input* bzw. *hidden units* in einem Netzwerk führen aufgrund von gegenseitigen Abhängigkeiten nicht zu einer eindeutigen Lösung, sondern zu Lösungsräumen mit mehreren gleichwertigen Lösungen.

Es läßt sich dennoch eine Aussage über die asymptotische Verteilung des Gewichtsvektors $\hat{w}$ machen. Nach White (1992) reicht es bereits aus, wenn der optimale Gewichtsvektor lokal identifiziert ist, d.h. wenn der Gewichtsvektor in einer kleinen Umgebung im Parameterraum eindeutig bestimmt werden kann. Damit bereitet die Existenz von Permutationen des Gewichtsvektors keine Schwierigkeiten mehr. Es muß lediglich zusätzlich angenommen werden, daß alle irrelevanten *units* aus dem entsprechenden Netzwerk entfernt worden sind. Mit diesen Voraussetzungen leitet White (1992) ab, daß die Gewichte eines neuronalen Netzwerks multivariat normalverteilt sind. Auf dieser Grundlage läßt sich nun in der üblichen Art und Weise Parameterinferenz betreiben (Lagrange-Multiplier-Test, Wald-Test).

Besondere Beachtung verdienen die Parametertests, denen die folgenden beiden Hypothesen zugrunde liegen (White, 1992):

1. Die Hypothese einer irrelevanten *input unit* $H_0 : R\hat{w} = 0$, wobei R unter den Gewichten $\hat{\gamma}$ des *input layers* diejenigen selektiert, die mit der zu testenden *input unit* verbunden sind, und die Gegenhypothese $H_1 : R\hat{w} \neq 0$.

2. Die Hypothese einer irrelevanten *hidden unit* $H_0 : R\hat{w} = 0$, wobei R unter den Gewichten $\hat{\beta}$ des *hidden layers* diejenigen selektiert, die mit der zu testenden *hidden unit* verbunden sind, und die Gegenhypothese $H_1 : R\hat{w} \neq 0$.

Für das Testen der ersten Hypothese verwendet man einen Wald-Test. Zuvor muß man jedoch sicherstellen, daß die im Netzwerk verwendeten *hidden units* nicht irrelevant sind. Dies läßt sich u.a. mit dem unten angesprochenen LM-Test bewerkstelligen. Unter dieser Voraussetzung lautet die Teststatistik:

$$(R\hat{w}_T)'(R\hat{C}_T R')^{-1}(R\hat{w}_T) \sim \chi^2_q, \tag{78}$$

wobei $\hat{C}_T$ die Schätzung der Kovarianzmatrix der Parameter darstellt und q der Anzahl der Parameter entspricht, die unter der Nullhypothese Null sind. Die Schätzung $\hat{C}_T$ der Kovarianzmatrix geschieht mittels

$$\hat{C}_T = T^{-1}\hat{A}_T^{-1}\hat{B}_T\hat{A}_T^{-1}. \tag{79}$$

Für den Fall der kleinsten-Quadrate-Methode lassen sich die Matrizen A und B konsistent schätzen,[43] indem

$$\hat{A}_T = \frac{1}{T}\sum_{t=1}^{T}\frac{\partial^2 \mathrm{SE}_t}{\partial w \partial w'} \text{ und } \hat{B}_T = \frac{1}{T}\sum_{t=1}^{T} u_t^2 \left(\frac{\partial f(X_t, w)}{\partial w}\right)\left(\frac{\partial f(X_t, w)}{\partial w}\right)' \tag{80}$$

berechnet werden.

Das Testen der irrelevanten *hidden unit*-Hypothese ist etwas komplizierter und soll deshalb nur kurz angesprochen werden. Wenn die Nullhypothese zutrifft, sind die Input-Gewichte der getesteten *hidden unit* nämlich nicht lokal identifiziert. Denn in diesem Fall sind die Output-Gewichte der *hidden unit* Null und die Werte der entsprechenden Input-Gewichte gleichgültig. Man nennt Parameter, deren Werte gleichgültig sind, *nuisance parameter*. Hypothesentests in diesem Kontext wurden von Davies (1977, 1987) untersucht. Die Schwierigkeit besteht darin, daß im Fall von nicht identifizierten Gewichten die Verteilung der Gewichte nicht mehr multivariat normal ist, sondern einer viel allgemeineren Klasse von *Mixed Gaussian*-Verteilungen angehört (Phillips, 1989). Damit ist eine auf der Normalverteilung beruhende Inferenz nicht mehr möglich. Unter bestimmten Voraussetzungen läßt sich das Identifikationsproblem der Parameter jedoch zu umgehen und ein LM-Test auf die Relevanz einer *hidden unit* anwenden. Dies wird von Anders/Korn (1996) dargestellt.

Prognosequalität

Die Qualität jedes ökonometrischen Modells muß sich daran messen, wie gut es zukünftige Entwicklungen vorhersagt. Entsprechend läßt sich der Einsatz eines neuronalen Netzwerks für die Prognose ökonomischer Variablen auch nur dann rechtfertigen, wenn der zu erwartende Prognosefehler kleiner ist als der einfacherer Modelle. Um darüber eine Aussage

[43] Für den Fall der *Maximum Likelihood*-Methode vgl. Arminger (1993).

treffen zu können, ist die *out of sample*-Analyse ausgesprochen wichtig. Dennoch wird sie in der Literatur neuronaler Netzwerke ausgesprochen selten auf eine zulängliche Art und Weise durchgeführt. In Finanzmarktanwendungen findet man beispielsweise häufig, daß Trefferquoten ausgezählt werden oder daß D-Mark Beträge errechnet werden, die sich mit einer auf einem neuronalen Netzwerk aufbauenden Handelsstrategie hätten erwirtschaften lassen. Mögen diese Werte auch hoch sein, so stellen sie dennoch lediglich absolute Werte dar. Es läßt sich mit Hilfe dieser Maße also nicht feststellen, ob hohe Werte mehr oder minder nur „zufällig" erreicht worden sind, also durch eine günstige Entwicklung in der betrachteten *out of sample*-Periode. Aus diesem Grund sollten immer entweder alternative Modelle oder alternative Strategien betrachtet werden, um die relative Qualität eines Netzwerks beurteilen zu können.

Es existieren zwei Möglichkeiten der Prognose: die Vorhersage des Wertes, der in der Zukunft unter *ceteris paribus*-Annahmen am wahrscheinlichsten angenommen wird, und die Vorhersage eines Konfidenzintervalls, in dem der zukünftige Wert unter *ceteris paribus*-Annahmen mit großer Wahrscheinlichkeit liegen wird. Vorhersagen enthalten auch im korrekt spezifizierten und geschätzten Modell eine erhebliche Ungewißheit aufgrund der Störterme des Modells, der Varianz der Parameter sowie durch die Vorhersagefehler eventuell pronostizierter unabhängiger Variablen. Die Unsicherheit, die jedem Modell immanent ist, stellt die Grundlage für die Konstruktion des Konfidenzintervalls der Prognose dar. Dabei hängt die Breite des Konfidenzintervalls sehr stark von den Parameterschätzern des Modells ab. Je effizienter die Schätzer, desto verläßlicher die Vorhersage des Modells.

Um den Vorhersagefehler im Erwartungswert zu quantifizieren, verwendet man eine Reihe verschiedener Statistiken. Die wichtigsten sind

- der *Mean Squard Error* $\mathrm{MSE} = \frac{u'u}{T}$
- der *Root Mean Squard Error* $\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$
- Theils $U = \sqrt{(\frac{u'u}{y'y})}$
- der *Mean Error* $\mathrm{ME} = \frac{1}{T}\sum_t u_t$
- der *Mean Percentage Error* $\mathrm{MPE} = \frac{100}{T}\sum_t \frac{u_t}{y_t}$
- der *Mean Absolute Error* $\mathrm{MAE} = \frac{1}{T}\sum_t |u_t|$
- der *Mean Absolute Percentage Error* $\mathrm{MAPE} = \frac{100}{T}\sum_t |u_t|/|y_t|$

Diese Statistiken können dabei helfen, den *Goodness of Fit* des Modells zu evaluieren. Der RMSE ist der populärste unter den genannten Statistiken, da er mit dem Standardfehler der Regression, i.e. die Standardabweichung der Residuen, unmittelbar vergleichbar ist. Jedoch hat er den Nachteil aller absoluten Maße: er kann nur interpretiert werden, wenn die Größenordnung der Variable y bekannt ist. Der Vorteil der relativen Maße besteht in ihrer Unabhängigkeit von der Größe der Variable y. Jedoch können sie nicht interpretiert werden, wenn einige y_t nahe an Null liegen, und nicht errechnet werden, wenn mindestens ein $y_t = 0$ ist.

Die Konstruktion eines Vorhersageintervalls basiert auf der Varianz des Vorhersagefehlers. Im linearen Modell werden für die Berechnung des Konfidenzintervalls üblicherweise die Störterme der Regression sowie die Parameterunsicherheit berücksichtigt. Der Vorhersagefehler $u_{T+1} = y_{T+1} - \hat{y}_{T+1}$ setzt sich also aus

$$u_{T+1} = y_{T+1} - \hat{y}_{T+1} = X_{T+1}(\alpha - \hat{\alpha}) + \varepsilon_{T+1} \tag{81}$$

zusammen. Damit ergibt sich die Varianz des Vorhersagefehlers durch:

$$\begin{aligned} \sigma^2_{u_{T+1}} &= \mathrm{Var}[X_{T+1}(\alpha - \hat{\alpha})] + \sigma^2_\varepsilon \\ &= \mathrm{Var}[\hat{y}_{T+1}] + \sigma^2_\varepsilon \\ &= X'_{T+1}[\sigma^2_\varepsilon (X'X)^{-1}]X_{T+1} + \sigma^2_\varepsilon. \end{aligned} \tag{82}$$

Das $(1-\alpha)$–Vorhersageintervall der exogenen Variable errechnet sich jetzt durch

$$y_{T+1} \in [\hat{y}_{T+1} - \varphi_{\alpha/2}\sigma_{u_{T+1}}; \hat{y}_{T+1} + \varphi_{\alpha/2}\sigma_{u_{T+1}}], \tag{83}$$

wobei $\varphi_{\alpha/2}$ den Wert der Standardnormalverteilung am $\alpha/2$–Wahrscheinlichkeitsniveau bezeichnet.[44]

Im nichtlinearen Fall wird die Unsicherheit der Prognose aufgrund der Parameterunsicherheit in vielen Fällen ignoriert. Das dann angegebene Intervall unterschätzt also die wahre Intervallbreite. Eine Berechnung der Varianz von $\hat{y}_{T+1}$ ist jedoch mittels einer linearen Taylorreihenentwicklung um w möglich

$$f(X_{T+1}, \hat{w}) \approx f(X_{T+1}, w) - \dot{f}_{T+1} \cdot (\hat{w} - w), \tag{84}$$

wobei $\dot{f}$ den Gradienten der nichtlinearen Regressionsfunktion f bezeichnet.[45] Damit ergibt sich die Varianz des Vorhersagefehlers analog zu Gleichung (82) mit

$$\sigma^2_{u_{T+1}} = \dot{f}'_{T+1}[\sigma^2_\varepsilon (\dot{f}'\dot{f})^{-1}]\dot{f}_{T+1} + \sigma^2_\varepsilon \tag{85}$$

und das Prognoseintervall entsprechend der Gleichung (83). Anstelle des Terms $C = \sigma^2_\varepsilon(\dot{f}'\dot{f})^{-1}$ für die Berechung der Kovarianzmatrix der Parameter läßt sich alternativ Gleichung (79) verwenden.

An dieser Stelle läßt sich noch einmal leicht erkennen, daß es ein Ziel jeder Modellierung neuronaler Netzwerke sein sollte, diese so sparsam wie möglich zu parametrisieren. Redundante Parameter sorgen nämlich üblicherweise für eine Vergrößerung der Kovarianzmatrix, wodurch das Konfidenzintervall des Prognosewertes breiter als notwendig wird.

[44] Beim üblichen 95%–Wahrscheinlichkeitsniveau ist $\varphi = 1,96$.
[45] Vgl. Seber/Wild (1989).

Bootstrapping

Anstatt die Varianz eines Modells zu schätzen, kann sie auch simuliert werden. Ein dazu geeignetes Verfahren ist das *Bootstrapping* (Efron, 1982). Das Verfahren besteht darin, aus der zur Verfügung stehenden Stichprobe eine Anzahl von (beispielsweise 200) *Bootstrap*-Stichproben zu generieren. Die *Bootstrap*-Stichproben werden dabei mittels eines Zufallsgenerators aus der ursprünglichen Stichprobe „mit Zurücklegen" gezogen, bis sie dieselbe Größe wie die ursprüngliche Stichprobe erreicht haben. Anschließend wird das zu untersuchende Modell auf der Basis aller *Bootstrap*-Stichproben jeweils neu geschätzt und jeder interessierende Wert berechnet. Insbesondere lassen sich nun auch die verschiedenen Prognosewerte simulieren, woraus sich das Konfidenzintervall der Prognosen ergibt.[46]

Die Verteilung (bzw. das Histogramm) des entsprechenden Schätzwertes erlaubt eine Aussage über Erwartungswert und Varianz dieses Wertes. Damit besteht der große Vorteil des *Bootstrapping*-Verfahrens darin, daß es eine *Bias*- sowie Varianz-Korrektur von Schätzwerten ermöglicht. Leider ist *Bootstrapping* sehr rechenintensiv, da jeweils so viele Modelle neu geschätzt werden müssen wie *Bootstrap*-Stichproben vereinbart wurden.

4 Zusammenfassung

Neuronale Netzwerke sind eine neue Klasse von statistischen Verfahren in der Ökonometrie. In ihrer Anwendung unterscheiden sie sich nicht von herkömmlichen Regressionsmodellen. Im Gegenteil, neuronale Netzwerke sind so flexibel, daß viele herkömmliche Regressionsmodelle mit ihrer vereinheitlichenden Methodologie nachgebildet werden können.

Neuronale Netzwerke sind parametrische Verfahren. Der wesentliche Unterschied zwischen neuronalen Netzwerken und herkömmlichen parametrischen Regressionsmodellen besteht darin, daß neuronale Netzwerke keine Annahmen über die funktionale Form des zu approximierenden Zusammenhangs benötigen. Neuronale Netzwerke lassen sich darüber hinaus auch als nichtparametrische Verfahren interpretieren, da sie wie diese unverzerrte Schätzungen ermöglichen, dafür aber einer große Varianz der Schätzung in Kauf nehmen müssen. Wie bei anderen nichtparametrischen Verfahren haben die Gewichte eines neuronalen Netzwerks keine theoretisch fundierbare Bedeutung.

Die statistische Modellierung mit neuronalen Netzwerken wird in dieser Arbeit mit Neurometrie bezeichnet. Die neurometrische Modellierung ist ein iterativer Prozeß, der sich in vier Schritte unterteilt: Identifikation der Daten, Spezifikation des Modells, Schätzung der Modellparameter und Diagnose des Modells. Die Iteration dieser Schritte sollte erst abgebrochen werden, wenn die Diagnose des Modells zufriedenstellend ist.

Es wurde hervorgehoben, wie wichtig die Wahl einer geeigneten Netzwerkarchitektur ist, um für eine zu erklärende Variable die bestmöglichen Prognosen mit einem kleinstmöglichen Konfidenzintervall abgeben zu können. Die für die Modellselektion verwendbaren Verfahren sind: Informationskriterien, Fehlermaße, Regularisierungstechniken, *Pruning*-Techniken, *Cross Validation* sowie *Stopped Training*. In dieser Arbeit wurde dargestellt,

[46] Vgl. Efron/Tibishirani (1986).

daß die Methode des *Stopped Training* jedoch nicht sinnvoll ist, um als Methode für die Bestimmung von Netzwerkparametern zu dienen.

Das Training Neuronaler Netzwerke entspricht der Schätzung der Netzwerkgewichte. Als Schätzmethode kann sowohl die Methode der kleinsten Quadrate als auch die *Maximum Likelihood*-Methode verwendet werden. Im Fall von normalverteilten Störtermen sind die Parameterschätzungen, die man mit diesen Methoden erhält, äquivalent. Wendet man diese Methoden für die Schätzung neuronaler Netzwerke auch dann an, wenn die Störterm nicht normalverteilt sind, dann spricht man von einer *Pseudo Maximum Likelihood*-Schätzung mit neuronalen Netzwerken. In jedem Fall sind die geschätzten Parameter eines Netzwerks unter gewissen Einschränkungen konsistent.

Die für die Schätzung verwendbaren numerischen Verfahren entsprechen den Optimierungsverfahren für nichtlineare Funktionen aus der numerischen Mathematik. Für die Bestimmung der Parameter neuronaler Netzwerke mit diesen Schätzverfahren läßt sich ausnutzen, daß alle *Feedforward*-Netzwerke die gleiche Grundstruktur aufweisen, so daß sich Gradient und Hessematrix der Zielfunktion unabhängig von der Problemstellung, auf die ein Neuronales Netzwerk angewendet wird, ex ante fest in einer Software implementieren lassen.

Die umfangreiche Diagnose eines trainierten Neuronalen Netzwerkes ist außerordentlich wichtig, um festzustellen, ob die Netzwerkarchitektur geeignet ist, einen gesuchten funktionalen Zusammenhang erwartungstreu wiederzugeben. Dazu lassen sich sowohl die Diagnoseverfahren der Ökonometrie verwenden (z.B. die Residualanalyse) als auch die heuristischen Verfahren der Netzwerkdiagnose (z.B. Relevanz- und Sensitivitätsanalyse). Für den Fall, daß die Diagnose neuronaler Netzwerke eine Struktur in den Residuen offenbart, die sich nicht auf eine Fehlspezifikation des Netzwerks zurückführen läßt, schlägt diese Arbeit die Anwendung eines Zwei-Stufen-Schätzverfahrens vor. Dieses ergibt konsistente und asymptotisch effiziente Parameterwerte.

Obwohl Parametertests bei neuronalen Netzwerken nicht die gleiche Bedeutung besitzen wie bei linearen Modellen, kann es sehr hilfreich sein, sie durchzuführen. Es lassen sich beispielsweise Hypothesentests hinsichtlich der Relevanz einer exogenen Variable sowie der Eignung der spezifizierten Netzwerkarchitektur durchführen. Die Tests beruhen auf dem Resultat von White, der gezeigt hat, daß die Gewichte eines neuronalen Netzwerkes asymptotisch normalverteilt sind. Insbesondere lassen sich neuronale Netzwerke mit Hilfe dieser Tests dazu einsetzen, die Approximationsqualität anderer statistischer Modelle zu überprüfen, indem letztere in ein neuronales Netzwerk eingebettet werden.

Alles in allem sind Neuronale Netzwerke also eine leistungsfähige und sehr flexible Klasse von statistischen Modellen, die den Vergleich mit den herkömmlichen Verfahren nicht zu scheuen braucht. Leider läßt die Literatur jedoch umfangreiche Studien vermissen, die die Leistungsfähigkeit neuronaler Netzwerke vor einem theoretischen Hintergrund bestätigt. Um neuronaler Netzwerke als ernstzunehmende ökonometrische Modelle, wären dazu eine Reihe von Monte-Carlo-Simulationen notwendig. Aus solchen Studien ließen sich zudem auch Erkenntnisse hinsichtlich der *finite sample properties* neuronaler Netzwerke ableiten, die in der bisherigen Untersuchung der asymptotischen Eigenschaften neuronaler Netzwerke nicht berücksichtigt sind.

Nomenklatur

b	Bandbreite einer Kernfunktion
d	Abstiegsrichtung bei der numerischen Optimierung
f	Funktionsschätzer für F
g	Transformationsfunktion im Netzwerk
h	Index einer *hidden unit*
i	Index einer *input unit*
k	Kernfunktion
n	Index der Iterationsschritte
r	Vektor von Eingangssignalen einer *unit*
s	Vektor von Ausgangssignalen einer *unit*
t	Index einer Beobachtung
u	Residuen einer Regression
v	Residuen einer Regression
w	Gewichtsvektor
y	Abhängige Variable
q	Zielfunktion
A	Erwartungswert der Hessematrix der Zielfunktion
B	Varianz des Gradienten der Zielfunktion
C	Kovarianzmatrix
F	Zu approximierende Funktion
H	Hessematrix oder Anzahl *hidden units*
I	Einheitsmatrix oder Anzahl nichtkonstanter erklärender Variablen x
K	Anzahl der freien Modellparameter
L	*Likelihood*-Funktion
N	Anzahl von Iterationsschritten
P	Wahrscheinlichkeit
Q	charakteristische Richtungsmatrix der Optimierungsverfahren
R	Matrix von Restriktionen
S	Matrix der Ausgangssignale eines *hidden layers*
T	Anzahl der Beobachtungen
X	Matrix der erklärenden Variablen
α	OLS-Parametervektor bzw. lineare Gewichte eines Netzwerks
β	Parametervektor des *hidden layer*
γ	Parametervektor des *input layer*
δ	Umgebungsparameter
ε	*white noise*-Störterm
η	Schrittweite der Gradientenverfahren
κ	Parameter des konjugierten Gradientenabstiegs
λ	Parameter für die Netzwerk-Regularisierung
μ	Momentum-Parameter
ρ	Parameter beim Levenberg-Marquardt-Verfahren
σ	Standardabweichung
θ	allg. Parametervektor
ϑ	allg. Parametervektor
Ω	Kovarianzmatrix
ξ	normierte erklärende Variable
$\mathcal{I}$	Fisher-Informationsmatrix
$\mathcal{L}$	*Loglikelihood*-Funktion
$\mathcal{N}$	Normalverteilung

Literatur

Akaike H. (1973): *Information Theory and an Extension of the Maximum Likelihood Principle.* In Petrov B., Csáke F. (eds): *2nd International Symposium on Information Theory.* Budapest.

Amemiya T. (1980): *Selection of Regressors.* International Economic Review, 21, 331–354.

Amemiya T. (1983): *Non-linear Regression Models.* In Griliches Z., Intriligator M.D. (eds): *Handbook of Econometrics*, I.

Amemiya T. (1985): *Advanced Econometrics.* Basil Blackwell.

Anders U. (1993): *Multivariate Zeitreihenanalyse mit Neuronalen Netzwerken.* Diplomarbeit 1993, Universität Karlsruhe (TH).

Anders U. (1996) : *Was neuronale Netze wirklich leisten.* Die Bank (bevorstehend).

Anders U., Korn O. (1996) : *Statistical Model Selection in Neural Networks.* ZEW Discussion Paper (bevorstehend).

Arminger G. (1993): *Ökonometrische Schätzverfahren für neuronale Netze.* S. 25–39 in Bol G., Nakhaeizadeh G., Vollmer K.-H. (Hrsg): *Finanzmarktanwendungen Neuronaler Netze und ökonometrischer Verfahren.* Physica-Verlag.

Box G.E.P., Jenkins M. (1976): *Time Series Analysis, Forecasting and Control.* Holden Day.

Craven P., Wahba G. (1979): *Smoothing Noisy Data with Spline Functions: Estimating the Correct Degree of Smoothing by the Method of Generalized Cross-Validation.* Numerische Mathematik, 31, 377–403.

Davidson R., MacKinnon J.G. (1993): *Estimation and Inference in Econometrics.* Oxford University Press.

Davies R.B. (1977): *Hypothesis Testing when a Nuisance Parameter is Present only under the Alternativ.* Biometrika, 64, 247–254.

Davies R.B. (1987): *Hypothesis Testing when a Nuisance Parameter is Present only under the Alternativ.* Biometrika, 74, 33–43.

Davis L. (1987): *Genetic Algorithms and Simulated Annealing.* Pitman.

De Groot C. (1993): *Nonlinear Time Series Analysis with Connectionist Networks.* Dissertation, ETH-Zentrum. Diss. ETH no. 10038.

Dennis J.E., Schnabel R.B. (1983): *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall.

Efron B. (1982): *The Jackknife, the Bootstrap and Other Resampling Plans.* Siam.

Efron B., Tibshirani R. (1986): *Bootstrap Methods for Standard Errors Confidence Intervals, and Other Measures of Statistical Accuracy.* Statistical Science, 1(1), 54–77.

Efron B., Tibshirani R.J. (1993): *An Introduction to the Bootstrap.* Chapman & Hall.

Efron B., Gong G. (1983): *A Leisurely Look at the Bootstrap, the Jackknife, and Cross-Validation.* The American Statistician, 37(1), 36–48.

Fahlman S.E., Lebiere C. (1990): *The Cascade-Correlation Learning Architecture.* Pages 524–532 of Touretzky D.S. (ed): *Advances in Neural Information Processing Systems II (Denver 1989).* Morgan Kaufmann.

Fahlmann S.E. (1989): *An empirical study of learning speed in back-propagation networks.* Morgan Kaufmann.

Friedman J.H., Stuetzle W. (1981): *Projection Pursuit Regression.* Journal of the American Statistical Association, 76(376), 817–823.

Geman S., Bienenstock E. (1992): *Neural Networks and the Bias/Variance Dilemma.* Neural Computation, 4, 1–58.

Gourieroux C., Monfort A., Trognon A. (1984): *Pseudo maximum likelihood methods: Theory.* Econometrica, 52(3), 681–700.

Granger C.W.J., Teräsvirta T. (1993): *Modelling Nonlinear Economic Relationships.* Oxford University Press.

Granger C.W.J., Newbold P. (1974): *Spurious Regression in Econometrics.* Journal of Econometrics, 2, 111–120.

Greene W.H. (1993): *Econometric Analysis.* Macmillan.

Härdle W., Linton O. (1994): *Applied Nonparametric Methods.* In Griliches Z., Intriligator M.D. (eds): *Handbook of Econometrics*, IV, 2295–2339.

Härdle W. (1990): *Applied Nonparametric Regression.* Cambridge University Press.

Hecht-Nielsen R. (1989): *Neurocomputing.* Addison-Wesley.

Hertz J., Krogh A., Palmer R.G. (1991): *Introduction to the Theory of Neural Computation.* Addison-Wesley.

Hosmer D.W., Lemeshow S. (1989): *Applied Logistic Regression.* John Wiley & Sons.

Hutchinson, J.M. (1994): *A Radial Basis Function Approach to Financial Time Series Analysis.* Dissertation, Massachusetts Institute of Technology.

Judge G.G., Griffiths W.E., Hill R.C., Lütkepohl H., Lee T.-C. (1985): *The Theory and Practice of Econometrics (2nd edition).* John Wiley and Sons.

Judge G.G., Hill R.C., Griffiths W.E., Lütkepohl H., Lee T.-C. (1988): *Introduction to the Theory and Practice of Econometrics (2nd edition).* Wiley, New York.

Kuan C.-M., White H. (1994): *Artificial Neural Networks: An Econometric Perspective.* Econometric Reviews, 13(1), 1–91.

Lee T.-H., White H., Granger C.W.J. (1993): *Testing for Neglected Nonlinearity in Time Series Models.* Journal of Econometrics, 56, 269–290.

Neal R. M.(1995): *Bayesian Learning for Neural Networks.* PhD Thesis, University of Toronto.

Michie D., Spiegelhalter D.J., Taylor, C.C. (eds) (1994): *Machine Learning, Neural and Statistical Classification.* Ellis Horwood.

Mills T.C. (1993): *The Econometric Modelling of Financial Time Series.* Cambridge University Press.

Moody J.E. (1992): *The Effective Number of Parameters: An Analysis of Generalization and Regularization in Nonlinear Learning Systems.* Advances in Neural Information Processing Systems, 4, 847–854.

Müller B., Reinhardt J. (1990): *Neural Networks.* Springer.

Murata N., Yoshizawa S., Amari S. (1991) *A Criterion for Determining the Number of Parameters in an Artificial Neural Network Model.* In Kohonen T., Mäkisara K., Simula O., Kangas J. (eds): *Artificial Neural Networks.* North-Holland.

Murata N., Yoshizawa S., Amari S. (1994) *Network Information Criterion Determining the Number of Hidden Units for Artificial Neural Network Models.* IEEE Trans. Neural Networks, 5, 865–872.

Poggio T., Girosi F. (1990): *Networks for Approximation and Learning.* Proceedings of the IEEE, 78(9), 91-106.

Phillips P.C.B. (1989): *Partially Identified Econometric Models.* Econometric Theory, 5, 181–240.

Press W.H., Flannery B.P., Teukolsky S.A., Vetterling W.T. (1992): *Numerical Recipes in C.* Cambridge University Press.

Rehkugler H., Poddig, T. (1993): *Kurzfristige Wechselkursprognosen mit Künstlichen Neuronalen Netzwerken.* S. 1–24 in Bol G., Nakhaeizadeh G., Vollmer K.-H. (Hrsg): *Finanzmarktanwendungen neuronaler Netze und ökonometrischer Verfahren.*

Rehkugler H., Zimmermann H.G. (1994): *Neuronale Netze in der Ökonomie.* Vahlen.

Rojas, R. (1993): *Theorie der neuronalen Netze: eine systematische Einführung.* Springer.

Rumelhart D.E., McClelland J.L. (1986): *Parallel Distributed Processing.* Vol. I & II. MIT Press.

Sarle W.S. (1994): *Neural Networks and Statistical Models.* Proceedings of the Nineteenth Annual SAS Users Group International Conference.

Schwarz G. (1978): *Estimating the Dimension of a Model.* The Annals of Statistics, 6, 461–464.

Seber G.A.F., Wild C.J. (1989): *Nonlinear Regression.* John Wiley & Sons.

Steurer E. (1994): *DM/US-Dollar. Monatsprognosen mit ökonometrischen Verfahren und neuronalen Netzen.* Daimler Benz AG.

Swanson N.R., White H. (1995): *A Model-Selection Approach to Assessing the Information in the Term Structure Using Linear Models and Artificial Neural Networks.* Journal of Business & Economic Statistics, Vol 13, No 3, 265–275.

Teräsvirta T., Lin C.-F., Granger C.W.J. (1992): *Power of the Neural Network Linearity Test.* Journal of Time Series Analysis, 14 (2), 209–220.

Teräsvirta T., Tjostheim D., Granger C.W.J. (1994): *Aspects of Modelling Nonlinear Time Series.* In Griliches Z., Intriligator M.D. (eds): *Handbook of Econometrics,* IV, 2917–2957.

Weigend A.S., Hubermann B.A., Rummelhart D.E. (1990): *Predicting the Future: a Connectionist Approach.* International Journal of Neural Systems, 1, 193–209.

White H. (1989): *An Additional Hidden Unit Test for Neglected Nonlinearity in Multilayer Feedforward Networks.* Proceedings of the International Joint Conference on Neural Networks, Washington, DC. SOS Printing, II, 451-455.

White H. (1992): *Artificial Neural Networks: Approximation and Learning Theory.* Blackwell.

White H. (1994): *Estimation, Inference and Specification Analysis.* Cambridge University Press.

Zell A. (1994): *Simulation Neuronaler Netze.* Addison–Wesley.

Zimmermann H.G., Hergert F., Finnoff W. (1991) *Neuron Pruning and Merging Methods for Use in Conjunction with Weight Elimination.* Siemens AG, Corporate Research and Development, ZFE IS INF 23.