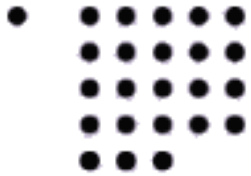


Fachhochschule Köln - Campus Gummersbach
Fakultät für Informatik und Ingenieurwissenschaften

University of Applied Sciences Cologne
Faculty of Computer and Engineering Science

Studiengang Allgemeine Informatik

Diplomarbeit

zur Erlangung des Diplomgrades
Diplom-Informatiker (FH)
in der Fachrichtung Allgemeine Informatik

Entwicklung einer Methodologie für ein hybrides neuronales Prognosesystem und Test an Anwendungsbeispielen

Von: Serhat Cinar

Matr.-Nr.: 11030409

Erstprüfer: Prof. Dr. Hartmut Westenberger

Zweitprüfer: Prof. Dr. Wolfgang Konen

vorgelegt: 22. Juli 2005

Korrekturblatt

zur Diplomarbeit
„Entwicklung einer Methodologie für ein hybrides neuronales Prognosesystem und Test an Anwendungsbeispielen“
von Serhat Cinar
AI 11030409

Auf Seite 62 wurde der Theilsche Ungleichheitskoeffizient falsch definiert. Die korrekte Formel lautet:

$$U = \sqrt{\frac{\frac{1}{N}\sum_{t=1}^{N}\left[(x_t - f_t)^2\right]}{\frac{1}{N}\sum_{1}^{N}\left[(x_t - x_{t-1})^2\right]}}$$

Die durchgeführten Berechnungen des Theilschen Koeffizienten wurden entsprechend ebenfalls falsch durchgeführt. Hier die korrekten Werte:

Versuch Prognose StromBRD

Modell	Theil U Trainingsmenge	Theil U Validierungsmenge
SDIF MLR ohne PCA	0.5020265324	0.5209473352
SDIF KNN 21x7x1 ohne PCA	0.4047708910	0.5139529405
SDIF KNN 7x10x1 (mit 6% PCA)	0.3311129951	0.5135571777
Hybrid (mit 6% PCA) KNN 7x15x1	0.4824079693	0.4877386061

Versuch Prognose DAX

Modell	Theil U Trainingsmenge	Theil U Testmenge
Lag 1 MLR	0.9797244224	0.9795564530
DIF Lags 1, 2, 3, 6, 12 MLR	0.9784204086	0.9508961425
DIF Lags 1, 2, 3, 6, 12 KNN 15x13x1	1.1374637150	0.9973190589
DIF Lags 1, 2, 3, 6, 12 PCA KNN 10x7x1	1.2983784770	0.9064465654
Hybrid 1, 2, 3, 6, 12 PCA KNN 10x11x1	0.4095976642	0.8577944079

Vorwort

Diese Arbeit entstand im Anschluss auf die zwei semestrige Vorlesung „Künstliche Neuronale Netze“ von Prof. Dr. H. Westenberger, in dem ich die Grundlagen und weiterführenden Themen im Bereich der Anwendung von künstlichen neuronalen Netzen lernte. Hier fasste ich auch meinen Entschluss meine Diplomarbeit im Bereich der künstlichen neuronalen Netze zu schreiben. Prof. Dr. Westenberger schlug mir verschiedene Themen vor, darunter das vorliegende Thema. Prof. Dr. Westenberger hatte sich bereits in einigen Projekten mit multivariater Prognose durch künstliche neuronale Netze auseinander gesetzt und schlug mir vor, klassische lineare Ansätze mit künstlichen neuronalen Netzen zu kombinieren. Die Zeitreihenprognose durch Kombination verschiedener Verfahren ist nicht neu, doch gibt es wenig Literatur, die eine konkrete Methodologie für Kombinationen von künstlichen neuronalen Netzen und klassischen Prognosemodellen aufzeigt. Eine solche Methodologie und ihre Anwendung soll die vorliegende Arbeit aufzeigen.

Mein Dank gilt Prof. Dr. Westenberger, der mir mit vielen Ratschlägen bei der Erstellung dieser Arbeit geholfen hat. Schließlich gilt mein besonderer Dank an meine Eltern, die mir das Studium erst ermöglichten und mich während meines Studiums unterstützten, sowie meinen Freunden, die mir geistige Unterstützung gaben. Außerdem danke ich allen freien Quellen im Internet, welche mir schnell eine umfangreiche Sicht auf das Thema aus mehreren Blickwinkeln gewährten. Entsprechend möchte ich diese Arbeit frei zugänglich machen für alle Wissensdurstigen, frei nach dem 14. Dalai Lama: „Teile Dein Wissen, so erlangst Du Unsterblichkeit.“

Abstract

Timeseries forecasting is an important tool for controling in many areas. Several authors have combined traditional univariate and linear forecasting methods like ARIMA with nonlinear forecasting methods like artificial neural nets to improve forecasting performance. This thesis points a methodology out for dealing with timeseries and using a hybrid methodology by combining the multiple linear regression and artificial neural nets for multivariate timeseries forecasting. With some sample timeseries it is shown, how the performance of the hybrid method can improve the forecasting performance in comparison to plain artificial neural nets and plain multiple linear regression methods.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungen und Symbole

Abb.	Abbildung
ABSE	Summe der absoluten Fehlerbeträge, sum of absolute errors
Absch.	Abschnitt
ACF	Autokorrelationsfunktion
AO	Additive outlier, lokaler Ausreißer
AR	Autoregression
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
avg.	Average, Durchschnitt, durchschnittliche
DFT	Diskrete Fourier Transformation
EWMA	Exponential weighted moving average, exponential smoothing exponentiell gewichteter gleitender Durchschnitt, exponentielle Glättung
FFT	Fast Fourier Transformation
IO	Innovative outlier, Ausreißer mit einem längeren temporären Effekt
IQR	Interquartile range
KNN	Künstliches neuronales Netz
LC	Level change, permanente Niveauänderung
LS	Level Shift, Plötzlicher Wechsel des mittleren Niveaus
MA	Moving average, gleitender Durchschnitt
MAD	Mittlerer absoluter Fehler, mean absolute error, mean absolute deviation
MAPE	Mittlerer absoluter prozentualer Fehler, mean absolute percent error
MCPN	McCulloch-Pitts Neuron
MKQ	Methode der kleinsten Quadrate
MLR	Multiple lineare Regression
MSE	Mittlerer quadratischer Fehler, mean squared error
PACF	Partielle Autokorrelationsfunktion
RMSE	Quadratwurzel des mittleren quadrierten Fehlers, root mean squared error
SARIMA	Seasonal Autoregressive Integrated Moving Average
SOM	Self organizing map, selbstorganisierende Karte, Kohonen-Karte
SSE	Summe der quadrierten Fehler, sum of squared errors
Tanh	Tangens hyperbolicus
TC	Transient level change , sanfte Niveauänderung
Theil U	Theil'scher Ungleichheitskoeffizient
VC	Variance change, stetige Änderung der Varianz
WMA	Wighted moving average, gewichteter gleitender Durchschnitt

t	Index der Beobachtungen (Zeitindex), erste Beobachtung $t = 1$
x_t	Wert zum Zeitpunkt t
f_t	Prognostizierter Wert für den Zeitpunkt t
N	Anzahl der Beobachtungen
∇	Differenzfilter
Δ	Delta, Steigung
r	Autokorrelationskoeffizient

r_k	Autokorrelationskoeffizient mit lag k (Autokorrelationsfunktion)
σ_{xy}	Kovarianz der Werte x und y
T	Trendfunktion für den globalen Trend
T_t	Wert der Trendfunktion für den globalen Trend zum Zeitpunkt t
S	Trendfunktion für den saisonalen Trend
S_t	Wert der Trendfunktion für den saisonalen Trend zum Zeitpunkt t
$\hat{x}_t$	Trend bereinigter Wert zum Zeitpunkt t
e	Eulersche Konstante
ε_t	Nicht prognostizierter Fehler zum Zeitpunkt t

1 Einleitung

1.1 Motivation

„My interest is in the future...
because I'm going to spend the rest of my life there."
Charles Kettering[1]

Nach den antiken Mythen aus Griechenland (um das 6. und 5. Jahrhundert v. Chr.) verfügte das Orakel von Delphi die Fähigkeit, die Zukunft vorauszusagen. „Apollon, der Gott der Weisheit, sprach durch seine Priesterin, Pythia genannt, und erfüllte sie mit seiner Weisheit, so dass sie den richtigen Rat geben konnte. Die Ratschläge bestanden aus Sprüchen oder anderen Zeichen."[2]
Diese Art der Zukunftsvoraussage blieb lange Zeit erhalten. Bei den Römern sagten Pontifices (sakrale Beamte) und Flämines (Priester einer bestimmten Gottheit) die Zukunft aus himmlischen Zeichen, wie Blitz und Donner, oder dem Flug der Vögel voraus.[3] Weitere verbreitete Rituale waren lesen aus den Eingeweiden von Opfertieren, Befragung von Tier und Menschenknochen. Dies zeigt, dass die Menschen seit ältester Zeit versucht haben, die Zukunft voraus zu sagen, mit mehr oder weniger Erfolg.
Die modernen Methoden der Prognose sind gegenüber den antiken Weissagungen mathematisch-wissenschaftlich begründet, empirisch und deterministisch. Zu den Hauptanwendungsgebieten der Zeitreihenanalyse gehören[4]:

- Deskription der Daten
- Modellierung des Prozesses, der den Daten zugrunde liegt
- Prognose des weiteren Verlaufs der Daten in der Zukunft
- Kontrolle der zugrunde liegenden Prozesse, welches jedoch meist wegen der Komplexität der Prozesse kaum möglich ist

Der Bedarf an Prognosen entstammt vor allem der Entscheidungsfindung. Hier einige Beispiele aus verschiedenen Anwendungsgebieten, die teilweise später auch genauer diskutiert werden:

- Wirtschaft
  - Prognose von Wertpapierkursen, um fundierte Entscheidungen für den An- / Verkauf zu treffen
  - Prognose von Absatzmengen zur Bestimmung der zu erwartenden Absätze
  - Bedarfsschätzung von Ressourcen zur Planung (Operational Research)
- Volkswirtschaft
  - Prognose von Kennziffern eines Landes, Volkswachstum, Arbeitslosenzahlen, etc., welche auch in Gesetze einfließen
- Ökologie
  - Wetterprognosen zur Vorbereitung für Landwirtschaften oder einfach zur Planung des Wochenendes

[1] Charles F. Kettering (*1876 †1958), amerikanischer Ingenieur.
[2] Lindenthal, Friedemann; http://www.geschi.de/artikel/orkdelph.shtml (Abruf: 19.04.05).
[3] http://de.wikipedia.org/wiki/Orakel (Abruf: 19.04.05).
[4] Nach [Schlittgen 01] Vorwort

    - Schadstoffemissionsprognosen zur Prävention von Umweltverschmutzung
    - Erdbebenprognosen zur Vorbereitung und Warnung
- Biologie
    - Prognosen zu Populationswachstümen
- Technische Kontrollen
    - Prognose von Schadstoffkonzentrationen zur präventiven Reglerkontrolle

Die Popularität und auch Effektivität moderner Prognosemethoden liegt vor allem in zwei Faktoren begründet:

1. Lange Entwicklung der Prognoseverfahren mit immer ausgefeilteren Methoden, welche recht gute Ergebnisse liefern.
2. Fortschritt der Computertechnik, die es erst ermöglicht den immensen Rechenaufwand, der zur Analyse von großen Datenmengen benötigt wird, zu bewältigen.

Daraus ergeben sich aber auch die Nachteile moderner Prognoseverfahren: man benötigt Expertenwisssen sowohl über die zu prognostizierende Domäne als auch über die Prognoseverfahren. Dennoch haben sich viele Programme am Markt etabliert, welche das Expertenwisssen aufbringen und in automatisierte Verfahren umsetzen. Dennoch sind auch solche mächtigen Werkzeuge durch einen Experten auszuwerten und zu validieren und berücksichtigen kaum besondere Modelle, wie hybride Methoden aus Kombinationen mit künstlichen neuronalen Netzen.

1.2 Aufgabenstellung und Zielsetzung

„The best way to predict the future is to invent it."
Alan Curtis Kay[5]

Ziel dieser Arbeit ist es eine Methodologie zu entwickeln um lineare Methoden der Zeitreihenanalyse und –prognose mit Prognosemodellen künstlicher neuronaler Netze zu vereinen, um ein effizientes und effektives Modell der Prognose zu erhalten. Hierzu sollen zuerst verschiedene Methoden der Zeitreihenanalyse und -prognose vorgestellt werden. Anschließend sollen diese Methoden der linearen Zeitreihenanalyse an verschiedene Zeitreihen angepasst und als Vorverarbeitung einer Prognose mit künstlichen neuronalen Netzen (*KNN*) eingesetzt werden.
Im Bereich der Zeitreihenanalyse gibt es bereits sehr viele automatisierte Verfahren, die gute Prognoseergebnisse liefern. Beispielsweise kann das Programm Demetra[6] alle für ein komplexes SARIMA-Modell benötigten Parameter automatisch ermitteln und optimieren. Ziel dieser Arbeit soll es daher nicht sein, ein automatisches Verfahren für Zeitreihenprognosen zu entwickeln, sondern vielmehr einen Baukasten an Methoden zu untersuchen, mit deren Hilfe die Leistungsfähigkeit neuronaler Prognosen gesteigert werden können.

[5] Alan Curtis Kay (*1940), US-amerikanischer Computerpionier.
[6] Demetra: http://forum.europa.eu.int/irc/dsis/eurosam/info/data/demetra.htm

1.3 Gliederung

Das Kapitel 2 beschäftigt sich hauptsächlich mit der Definition von Zeitreihen, sowie mit Charakteristiken von Zeitreihen, welche als Grundlage der Zeitreihenanalyse dienen.
In Kapitel 3 wird anschließend ein Vorgehensmodell zur Zeitreihenprognose aufgezeigt, welche die Darstellung, Vorverarbeitung, Zerlegung und Prognose der Zeitreihe umfaßt. Hierbei werden primär Methoden der klassischen linearen Zeitreihenanalyse und –prognose vorgestellt und untersucht. Ein Abschnitt beschäftigt sich mit den Grundlagen und Anwendungen von künstlichen neuronalen Netzen im besonderen auf die Zeitreihenprognose.
Kapitel 4 gibt einen Überblick über Maßzahlen, welche es erlauben die Prognosegüte verschiedener Modelle miteinander zu vergleichen.
Im Kapitel 5 werden Maßzahlen vorgestellt, welche es erlauben die Prognosegüte verschiedener Modelle miteinander zu vergleichen.
Schließlich wird in Kapitel 6 experimentell die Performanz der multiplen linearen Regression mit einem künstlichen neuronalen Netz für multivariate Prognosen verglichen und anschließend zur Verbesserung der Performanz ein hybrides Modell aus beiden Methoden entwickelt.
Das letzte Kapitel enthält die obligatorische Zusammenfassung und diskutiert im Ausblick zusätzliche Ideen für Verbesserungen der Prognosemodelle.

2 Zeitreihenanalyse und –prognose

„It is said that the present is pregnant with the future."
Voltaire[7] in „The Portable Voltaire"

Dieses Kapitel führt Grundbegriffe, Konzepte und die Methodologie der üblichen Zeitreihenanalyse und –prognose ein und erklärt diese an Beispielen. Dabei wird auch das von Thiesing vorgestellte Modell der Zeitreihenprognose mit KNN vorgestellt. Die in diesem Kapitel vorgestellten methodischen Werkzeuge zur Zeitreihenanalyse und –prognose werden im nächsten Kapitel auf ihre Tauglichkeit als Vorverarbeitung einer KNN-Prognose im Vergleich zu derselben benutzt.

2.1 Definition einer Zeitreihe

„Eine Zeitreihe ist eine nach dem Zeitindex geordnete Menge von Beobachtungen x_t einer Zufallsvariablen X_t mit t = 1, ..., T, d. h. es liegen T Beobachtungen vor."[8] Dabei gehen die meisten Verfahren zur Analyse von Zeitreihen von einem äquidistanten Zeitraster aus, also von Beobachtungen, die in gleichmäßigen Abständen gewonnen wurden.

2.2 Univariate und multivariate Zeitreihen

Bei einer ***univariaten*** Zeitreihe liegt für jeden Zeitpunkt eine einzige Beobachtung vor, es wird also nur ein Merkmal betrachtet. Die Prognose zukünftiger Beobachtungen wird anhand der vorliegenden (vergangenen und aktuellen) Beobachtungen ermittelt. Dem liegt die Annahme zugrunde, dass die vorliegenden Beobachtungen über die Zeit hinweg eine interne Struktur bzw. Regelmäßigkeit besitzen, welche durch die Analyse herausgearbeitet und durch die Prognose benutzt werden soll.[9] Üblicherweise werden die letzten n Beobachtungen als ***Stützbereich*** (englisch: ***Lag***) benutzt, um die nächste, nicht mehr vorliegende Beobachtung zu prognostizieren. Thiesing spricht hierbei auch von einer „***horizontalen***" Prognose.[10]

Bei ***multivariaten*** Zeitreihen betrachtet man nicht eine Variable isoliert, sondern das Zusammenwirken mehrerer Variablen zugleich und damit ihre Abhängigkeitsstruktur. Damit liegen für jeden Zeitpunkt mehrere Beobachtungen oder verschiedene Merkmale vor: Die zu prognostizierende ***endogene*** Zeitreihe und die erklärenden ***exogenen*** Zeitreihen. Thiesing nennt die multivariate Prognose auch die „***vertikale***" Prognose. Der Ansatz der multivariaten Zeitreihe versucht auch der Tatsache gerecht zu werden, dass die meisten Prozesse nicht isoliert auftreten und auch nicht isoliert prognostizierbar sind, sondern nur als Menge mehrere miteinander in Abhängigkeit stehender Prozesse.

---

[7] Voltaire, eigentlich François-Marie Arouet (*1694 †1778), französischer Schriftsteller und Philosoph.
[8] [Leiner 82] S. 2.
[9] Vergl. [PrinsNIST] Absch. 6.4.
[10] Vergl. [Thiesing 98] S. 88.

2.3 Charakteristische Merkmale von Zeitreihen

Arithmetischer Mittelwert

Das arithmetische Mittel, auch als Durchschnitt bezeichnet, ist der Mittelpunkt der Beobachtungen. In der Physik wird er auch als Massezentrum bezeichnet. Er wird im univariaten Fall als Quotient der Summe der einzelnen Beobachtungen und der Anzahl der Beobachtungen berechnet.

$$\overline{x} = \frac{1}{N}\sum_{t=1}^{N}[x_t]$$

Bei einer multivariaten Zeitreihe aus M Reihen $X_1 ... X_m$ mit m=1...M besteht der Mittelwert aus dem M-dimensionalen Vektor $(\overline{x}_1,...,\overline{x}_m)$. Der Mittelpunkt wird also als Vektor der Mittelpunkte der einzelnen Reihen verstanden.

Varianz und Standardabweichung

Die Varianz bezeichnet die Stärke der Streuung der Werte um den Mittelwert und wird im univariaten Fall wie folgt berechnet:

$$s^2 = \frac{1}{N}\sum_{t=1}^{N}[(x_t - \overline{x})^2]$$

Auf der Varianz aufbauend wird die ***Standardabweichung*** definiert, welche im Gegensatz zur Varianz dieselbe Maßeinheit hat wie die zugrunde liegenden Werte:

$$s = \sqrt{s^2}$$

Empirische Kovarianz

Die Kovarianz beschreibt, wie stark zwei verschiedene Reihen X und Y linear voneinander abhängig sind. Linear bedeutet hier, dass eine Gerade das Verhältnis zwischen beiden Reihen beschreibt. Dies ist auch die Schwäche dieser Kenngröße, da sie nichts über nichtlineare (z.B. quadratische, logarithmische oder andere) Zusammenhänge beider Reihen aussagt.

$$\sigma_{XY} = \frac{1}{N}\sum_{t=1}^{N}[(x_t - \overline{x})(y_t - \overline{y})]$$ [11]

Liegt lediglich eine Zeitreihe vor, also im univariaten Fall, kann die Kovarianz auch die lineare Abhängigkeit zwischen Werten mit verschiedenen zeitlichen Abständen (***Lag***) bestimmen und wird dann als ***Autokovarianz*** bezeichnet. Dabei wird y_t durch x_{t+lag} ersetzt.

Korrelationskoeffizient

Der Korrelationskoeffizient (Ko-Relation = Mit-Beziehung), hier die Definition von Bravais-Pearson[12], auch als Produkt-Moment-Korrelation bekannt, ist eine

[11] [Heiler 94] S. 222., [Schlittgen 01] S. 4.

[12] Benannt nach den Mathematikern Auguste Bravais (*1811 †1863) und Karl Pearson (*1857 †1936), siehe

Normierung der Kovarianz auf das Intervall [-1;+1] und gibt Richtung und Stärke des linearen Zusammenhanges an. Der Wert des Korrelationskoeffizienten r liegt immer im Bereich –1 bis +1, wobei $-1 \le r < 0$ für einen umgekehrt proportionalen Zusammenhang, $r = 0$ für keinen linearen Zusammenhang und $0 < r \le 1$ für einen proportionalen Zusammenhang stehen. Je größer der Betrag von r ist, desto stärker ist die gemeinsame Korrelation.

$$r_{XY} = \frac{\sum_{t=1}^{N}[(x_t - \bar{x}) \cdot (y_t - \bar{y})]}{\sqrt{\sum_{t=1}^{N}[(x_t - \bar{x})^2] \cdot \sum_{t=1}^{N}[(y_t - \bar{y})^2]}}$$ [13]

Bei mehreren Zeitreihen bildet man üblicherweise die Korrelationsmatrix, deren Reihen und Spalten die Korrelationen der verschiedenen Zeitreihen zueinander enthalten. Da die Korrelation einer Zeitreihe zu sich selbst stets 1 ist, besteht die Diagonale der Korrelationsmatrix aus Einsen. Da der Korrelationskoeffizient auch reflexiv ist, d.h. $r_{XY} = r_{YX}$ gilt, ist die Korrelationsmatrix symmetrisch zu ihrer Diagonalen. Im folgenden die Korrelationsmatrix der Strom-Zeitreihen, welche in der Abb. 5 auch graphisch dargestellt werden.

	DE	FR	NL	UK	ES
DE	1.	.9156	.868	.7426	.7687
FR	.9156	1.	.8351	.814	.7723
NL	.868	.8351	1.	.6811	.9486
UK	.7426	.814	.6811	1.	.6247
ES	.7687	.7723	.9486	.6247	1.

Zu erkennen sind die stärksten Korrelationen zwischen dem Stromverbrauch in Deutschland und Frankreich und zwischen den Niederlanden und Spanien. Auch die anderen Länder haben hinsichtlich des Stromverbrauches große Ähnlichkeiten.
Bei der Auswahl exogener Reihen für multivariate Prognosen ist eine Vorbetrachtung der Korrelationen zwischen den Reihen von Vorteil. So kann im Vorfeld erkannt werden, wie stark zwei Reihen überhaupt zueinander in Beziehung stehen.

Autokorrelation

Im univariaten Fall interessiert besonders die Selbstähnlichkeit, bzw. der lineare Zusammenhang der Werte einer Reihe mit einem festen Abstand (Lag) zueinander. Man bedenke, dass meist zyklische Schwankungen, wie etwa Verkaufszahlen für Tannenbäume im Monat Dezember, regelmäßig ein hoch haben (Weihnachtsverkäufe) und damit die Dezember-Werte verschiedener Jahre einen starken linearen Zusammenhang haben. Daher wird bei Betrachtung univariater Zeitreihen den Autokorrelationskoeffizienten mit verschiedenen

Baur, Franz: Korrelationsrechnung. Mathematisch-Physikalische Bibliothek. Leipzig: Teubner, 1928.

[13] [Chatfield 89] S. 19., [Schlittgen 01] S. 4, [Heiler 94] S. 255.

Abständen besondere Aufmerksamkeit geschenkt, welche solche Zusammenhänge erkennen lassen.

Der Autokorrelationskoeffizient r_k mit Lag k ist definiert als:

$$r_k = \frac{\sum_{t=1}^{N-k}[(x_t - \bar{x})(x_{t+k} - \bar{x})]}{\sum_{t=1}^{N}(x_t - \bar{x})^2}$$ [14]

Bildet man von einer Reihe alle Autokorrelationskoeffizienten mit den möglichen Laglängen (k=1..N-k), so spricht man von der ***Autokorrelationsfunktion*** (kurz ***ACF***), welche durch verschiedene Formeln auch abgeschätzt werden kann. Graphisch wird diese Funktion dargestellt, indem die Koeffizienten im Verhältnis zur Laglänge gezeichnet werden. Wie auch im weiteren Verlauf dieser Arbeit ersichtlich wird, gibt die grafische Darstellung der ACF sehr viel Aufschluss über verschiedene Eigenschaften und Komponenten der Zeitreihe und ist damit ein unverzichtbares Werkzeug zur Bestimmung von Saisonalität und Trend. Die Beispiel ACF der Reihe ArbeitslosBRD in Abb. 1 zeigt eine saisonale Korrelation im Abstand von 12 Monaten (sichtbar an den regelmäßigen Spitzen) sowie eine starke Korrelation über mehrere Lags (langsamer An-/Abstieg des Graphen), welche für einen starken Trend spricht.

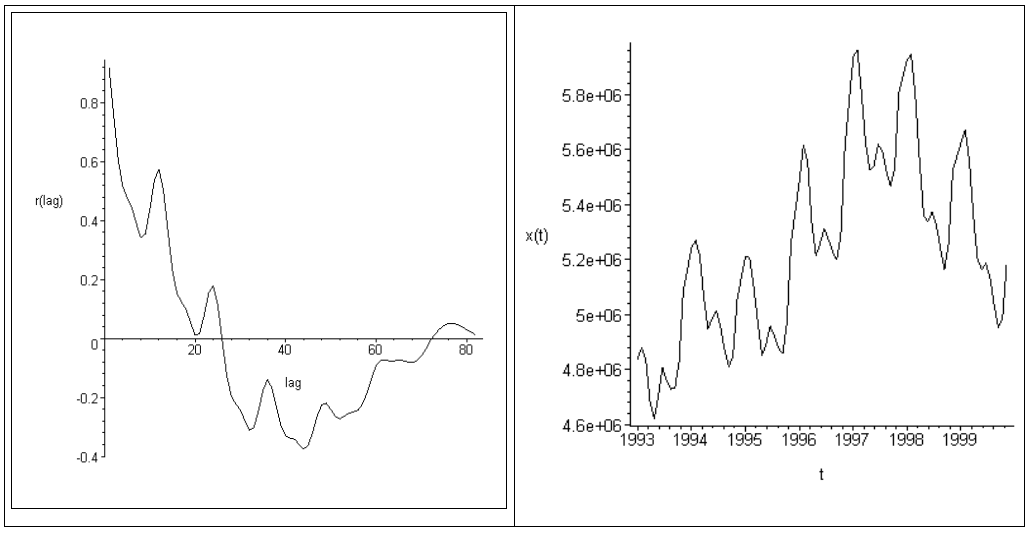


Abb. 1: ACF der Reihe ArbeitslosBRD mit Laggrößen bis 80 (links) sowie die Originalwerte von 1993-2000.

2.4 Stationäre und Nicht-Stationäre Zeitreihen

Von ***stationären Zeitreihen*** spricht man, wenn die Zeitreihe keine systematische Änderung des Mittelwertes (***Trend***), keine systematische Änderung der Varianz und keine zyklischen Schwankungen (***Saisonalität***) über die Zeit hinweg besitzt[15]. Dies bedeutet auch, dass verschiedene, gleichlange Segmente einer Zeitreihe keinen signifikanten Unterschied im Erwartungswert, in der Varianz und der Autokovarianz aufweisen und damit diese Charakteristiken unabhängig vom Zeitindex sind.[16]

[14] [Chatfield 89] S. 20., [Schlittgen 01] S. 5.
[15] Vergl. [Chatfield 89] S.10.
[16] [Walter 99] http://user.uni-frankfurt.de/~andreas/rekur/node3.html

Entsprechend spricht man von ***nicht-stationären Zeitreihen***, wenn diese Eigenschaften nicht erfüllt werden, weil z.B. die Zeitreihe mit der Zeit zunehmend divergiert. Die meisten Methoden der Zeitreihenprognose bauen auf stationäre Reihen auf. Die Prognose stationärer Reihen ist der einfachste Fall der Prognose: als zukünftige Werte wird einfach der Mittelwert der Reihe genommen, der stabil ist. Die verbreitetsten Verfahren versuchen durch Transformationen eine Reihe in eine stationäre zu überführen, um diese anschließend durch ihren Mittelwert oder einem Regressionsverfahren zu approximieren.

3 Methodologie der Zeitreihenprognose

Im folgenden soll das Vorgehen zur Prognose einer Zeitreihe dargestellt werden. Das Flussdiagramm in Abb. 2 zeigt den schematischen Verlauf.

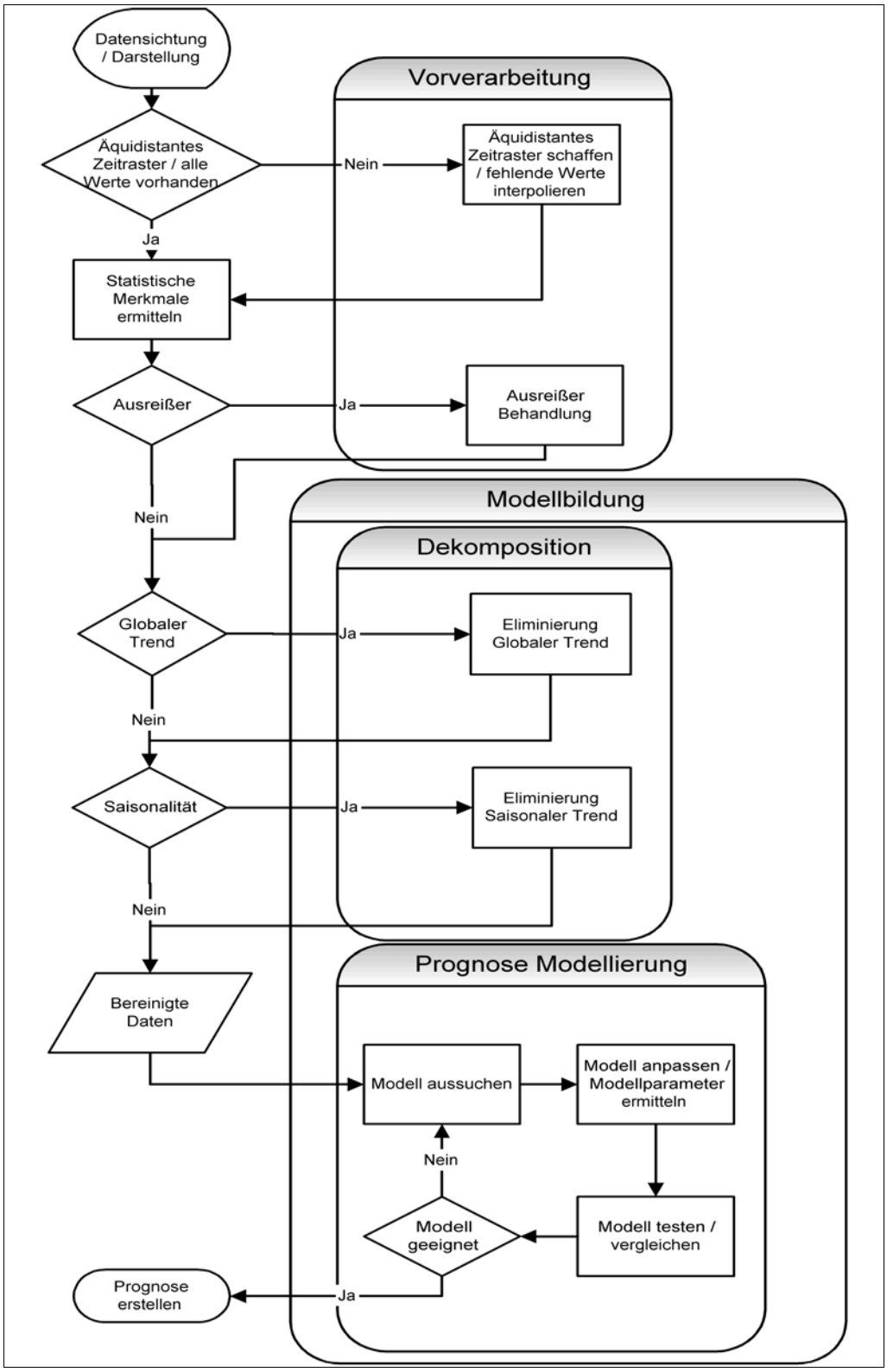


Abb. 2: Flussdiagramm einer Zeitreihenprognose.

3.1 Datensichtung und Darstellung

Der erste Schritt in der Analyse und Prognose von Zeitreihen ist die Darstellung derselben. Die erste Form der Datendarstellung ist meist der zweidimensionale Scatterplot, der in der horizontalen Achse die Zeit und in der vertikalen Achse den Wert zum entsprechenden Zeitpunkt als einfachen Punkt darstellt. Abb. 3. stellt den Scatterplot der Reihe StromBRD dar. Der Nachteil dieser Darstellungsart ist, das man mit dem bloßen Auge den zeitlichen Verlauf der vielen Punkte kaum erkennen kann.

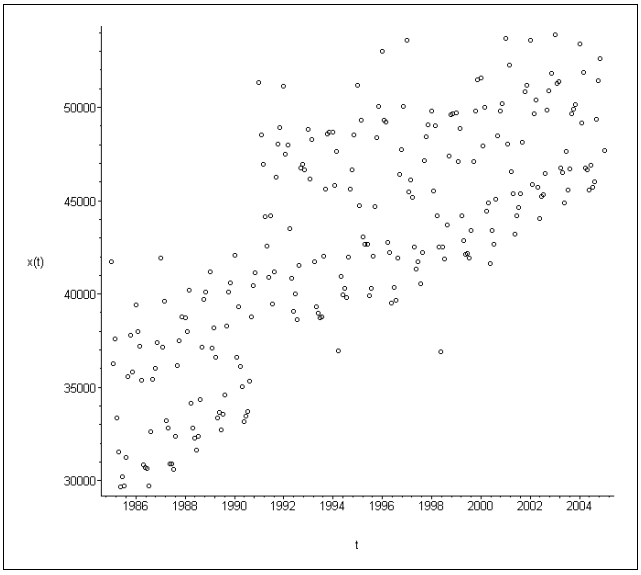


Abb. 3: Scatterplot der Reihe StromBRD.

Da bei Zeitreihen meist von stetigen Prozessen ausgegangen wird, die lediglich zu diskreten Zeitpunkten gemessen wurden, werden die einzelnen Punkte zu einem Graphen verbunden, wie in Abb. 4 zu sehen ist.

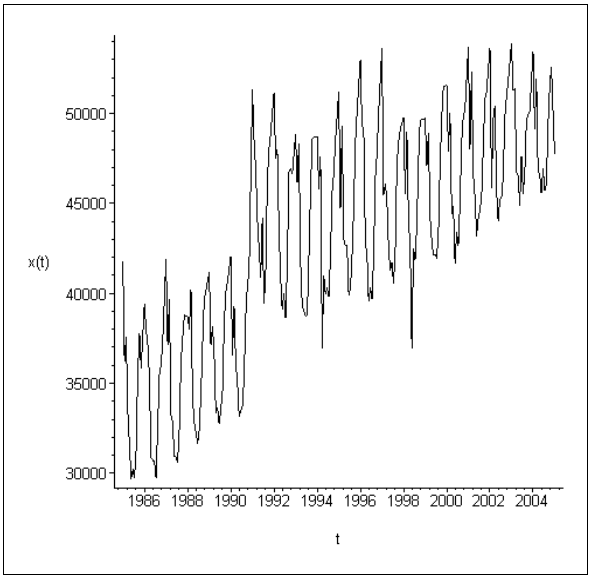


Abb. 4: Graph der Reihe StromBRD.

Bei multivariaten Zeitreihen können die verschiedenen Reihen auf einem gemeinsamen Graph gezeichnet werden (siehe Abb. 5). Dies setzt jedoch voraus,

dass die verschiedenen Reihen eine gemeinsame Maßeinheit haben. Außerdem dürfen ihre Höhen nicht zu stark voneinander unterschiedlich sein, da sonst die Y-Achsenskalierung die Darstellung zu sehr verzerrt. In einem solchen Fall können die Reihen zuerst Mittelwert bereinigt werden, so dass ihre Graphen annähernd gleiche Höhen haben.

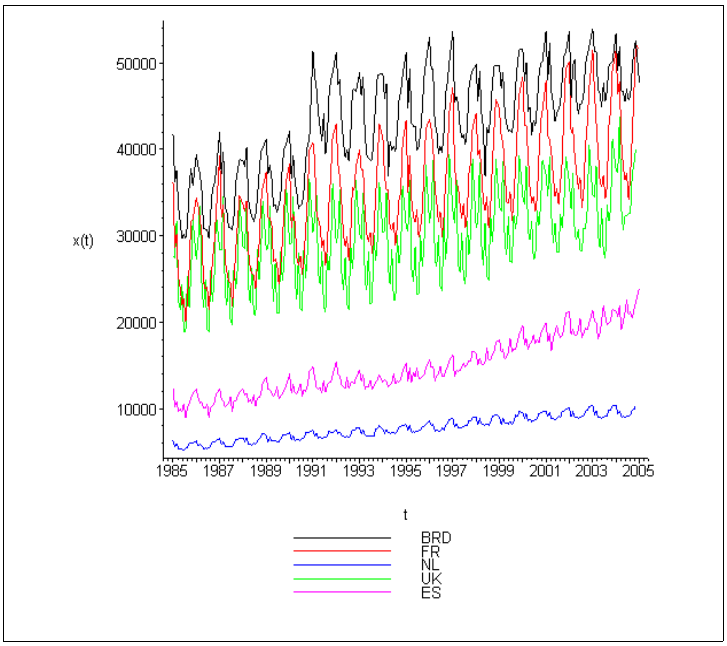


Abb. 5: Plot der Originalreihen StromBRD, StromFR, StromNL, StromUK, StromES.

Weitere Beispiele für grafische Darstellungen werden an gegebenen Stellen ergänzt.

3.2 Vorverarbeitung der Daten

3.2.1 Gleichmäßiges Zeitraster schaffen

Die meisten Methoden der Zeitreihenanalyse und –prognose setzen voraus, dass die Beobachtungen in einem regelmäßigen Zeitintervall (***äquidistantes Zeitraster***) gewonnen wurden. Diese Voraussetzung liegt hauptsächlich in der Vergleichbarkeit der Werte und der Vereinfachung des Zeitfaktors begründet. Werden Beobachtungen verschiedener Intervalle vermischt, z.B. einige Beobachtungen in täglichen, andere in monatlichen Intervallen, so sind die Werte zuerst an ein gemeinsames Zeitraster anzupassen. Besonders Messungen physikalischer Werte werden meist nicht zu vorgegebenen Zeitpunkten getätigt. Durch Aufteilen der Messwerte auf die kleinste gemeinsame Zeitskala können die Originalwerte auf äquidistante Zeitpunkte interpoliert werden. Auch tritt in der praktischen Anwendung häufig das Problem von fehlenden Werten durch Ausfall einer Messstation auf. Auch diese Werte können durch Interpolation abgeschätzt und ergänzt werden.

Zur Interpolation stehen eine Vielzahl von Möglichkeiten bereit, von denen hier lediglich die einfachsten dargestellt werden sollen:

- Lineare Interpolation
  Mit Hilfe der benachbarten Punkte $(t-1/x_{t-1})$ und $(t+1/x_{t+1})$ wird eine Geradengleichung g bestimmt und für den Punkt x_t an der Stelle t ausgewertet. Hierbei sind:
  $$g(t) = m \cdot t + b$$
  $m = \frac{x_{t+1} - x_{t-1}}{(t+1)-(t-1)}$ die Steigung der Geraden und
  $b = x_{t-1} - m \cdot (t-1)$ der y-Achsen Schnittpunkt der Geraden.
- Quadratische Interpolation
  Mit Hilfe der benachbarten Punkte $(t-2/x_{t-2})$, $(t-1/x_{t-1})$ und $(t+1/x_{t+1})$ wird eine Quadratische Gleichung g bestimmt und für den Punkt x_t an der Stelle t ausgewertet. Hierbei sind:
  $$g(t) = a \cdot t^2 + b \cdot t + c$$
  Da alle drei gegebenen Punkte der Gleichung g genügen müssen, erhält man drei lineare Gleichungen mit drei unbekannten, die exakt lösbar sind.
  (1) $x_{t-2} = a \cdot (t-2)^2 + b \cdot (t-2) + c$
  (2) $x_{t-1} = a \cdot (t-1)^2 + b \cdot (t-1) + c$
  (3) $x_{t+1} = a \cdot (t+1)^2 + b \cdot (t+1) + c$
- Polynomiale Interpolation mit Polynomen p-ten Grades, deren Spezialfälle p=1 die lineare und p=2 die quadratische Interpolation sind.
- Splineinterpolationen, die hier jedoch nicht genauer erläutert werden.

3.2.2 Ausreißer Identifikation und Behandlung

Als Ausreißer werden diejenigen Werte bezeichnet, die mit der Masse der übrigen Werte unvereinbar erscheinen.[17] Ausreißer können durch Mess-, Übertragungs-, Berichts- oder Rechenfehler verursacht werden.[18] Aber auch korrekte, ungewöhnliche Beobachtungen können als Ausreißer in den Werten auftauchen. So wäre in einer Datenmenge mit den Einwohnerzahlen deutscher Städte Berlin mit 3.382.169 Einwohnern[19] sicherlich ein Ausreißer (die nächst kleinere Stadt ist Hamburg mit 1.715.392 Einwohnern, bei drei Städten mit mehr als einer Million Einwohnern). Dies liegt aber offensichtlich nicht an einem Fehler.[20]
Die Wichtigkeit der Behandlung von Ausreißern wird vor allem dann deutlich, wenn man sich die für die Zeitreihenanalyse benutzten Maßzahlen und Methoden der Statistik vor Augen führt: Bei der Bildung der Varianz fließen die Abweichungen (oder Fehler) der Ausreißer quadriert ein. Somit verfälschen sie die Annahmen über die zugrundeliegenden Eigenschaften einer Menge sehr stark. Auch die Methode der kleinsten Quadrate bei der Regression reagiert auf Ausreißer sehr empfindlich, da auch hier die Quadrate der Abweichungen benutzt werden, wobei die größte Abweichung auch die größte Gewichtung erhält.[21]

[17] Vergl. Barnett, V., Lewis, T., Outliers in Statistical Data, Wiley John + Sons, 1994 zitiert nach [Buttler 03] S. 4.
[18] Rönz, B.; Strohe, H. G.: Lexikon Statistik, Gabler-Verlag Wiesbaden, 1994 zitiert nach [Buttler 03] S. 3.
[19] Bundesagentur für Arbeit, http://www.arbeitsagentur.de, Stand März 2005.
[20] Vergl. [Buttler 03] S. 4.
[21] Vergl. [Buttler 03] S. 6.

Ausreißer werden häufig nach ihrer Art kategorisiert:

- Additive Outlier (***AO***)
  Additive Ausreißer sind Ereignisse mit einem (relativ) großen aber temporären Effekt auf die Reihe.[22] Ein Beispiel für einen solchen Ausreißer ist der Dow-Jones Index ab dem 11.09.01 (Anschläge auf die Twin-Tower und das Pentagon, siehe Abb. 12). Bei einer feineren Unterteilung dieses Ausreißertyps wird der Begriff AO nur für einzelne Ausreißerwerte und Innovative Outlier (***IO***) für Ausreißer mit einem längeren temporären Effekt benutzt[23].
- Level Shift (***LS***)
  Level Shifts sind Ereignisse, die das Niveau der Zeitreihe ab einem bestimmten Zeitpunkt permanent verändern. Vor allem bei technischen Messungen werden durch Neukalibrierungen das Niveau der Zeitreihe verändert. Ein Beispiel hierfür bietet die Reihe StromBRD (siehe Abb. 4). Im Jahr 1991 fand die Wiedervereinigung statt, ab dem die Verbrauchswerte der früheren DDR und der alten BRD zusammengelegt wurden. Dies erklärt den plötzlichen Sprung des Verbrauchs in diesem Jahr, welcher natürlich konstant weiter erhalten blieb. Level Shifts werden weiterhin in permanente Niveauänderungen (level change, ***LC***) und sanfte Niveauänderungen (transient level change, ***TC***) unterteilt, welche einen langsamen und stetigen Wechsel zweier Niveaus bezeichnen.[24]
- Variance Change (***VC***)
  Varianzänderungen sind stetige Änderungen der Varianz. Dabei hat die Reihe die Form eines Kegels, der mit zunehmender Zeit eine größere Streuung zeigt.

Additive Ausreißer spielen auch eine wichtige Rolle im Bereich Data Mining, wo außergewöhnliche Werte besondere Aufmerksamkeit erhalten. Beispiele hierfür sind ungewöhnliche Transaktionen bei Kreditkartenmißbrauch oder Leistungen außergewöhnlicher Sportler. [25]

Zur Bestimmung von Ausreißern gibt es zahlreiche Verfahren, die meistens eine statistische Verteilung der zugrunde liegenden Menge annähern (wie z.B. eine Normalverteilung) und anhand eines Hypothesentests entscheiden, ob ein Wert mit großer Wahrscheinlichkeit ein Ausreißer (daher, außerhalb eines bestimmten Intervalls) ist oder nicht. Weitere Verfahren zur Bestimmung von AOs, IOs und LCs in univariaten Reihen betrachten die Differenzen erster Ordnung $\nabla^1 = x_t - x_{t-1}$, welche im Absch. 3.4.5 S.42 noch genauer betrachtet werden. Hier soll es ausreichen, die ersten Differenzen als Abstände zwischen benachbarten Punkten zu interpretieren. Werte, die einen außergewöhnlich großen Abstand zu ihren Nachbarwerten haben, deuten auf einen Ausreißer hin und sind durch besonders hohe Werte in den ersten Differenzen erkennbar. Nimmt man für die Differenzen eine Normalverteilung an (welches meist gegeben ist), so liegen 95% der Werte innerhalb der 2-fachen Standardabweichung um den Mittelwert (bzw. 99% der Werte innerhalb eines $3 \cdot s$ Intervalls). Die Intervallgrenzen helfen Ausreißer außerhalb der Intervallgrenzen zu erkennen. In der Zeitreihe

[22] Vergl. [Arranz 03] S. 2.
[23] Vergl. [Tsay 86] nach [Juntilla 01].
[24] Vergl. [Tsay 86] nach [Juntilla 01].
[25] Vergl. [Knorr 00].

StromBRD sind bei 1991 und 1994 zwei solcher Ausreißer außerhalb des $3 \cdot s$ Intervalls zu erkennen (siehe Abb. 6).

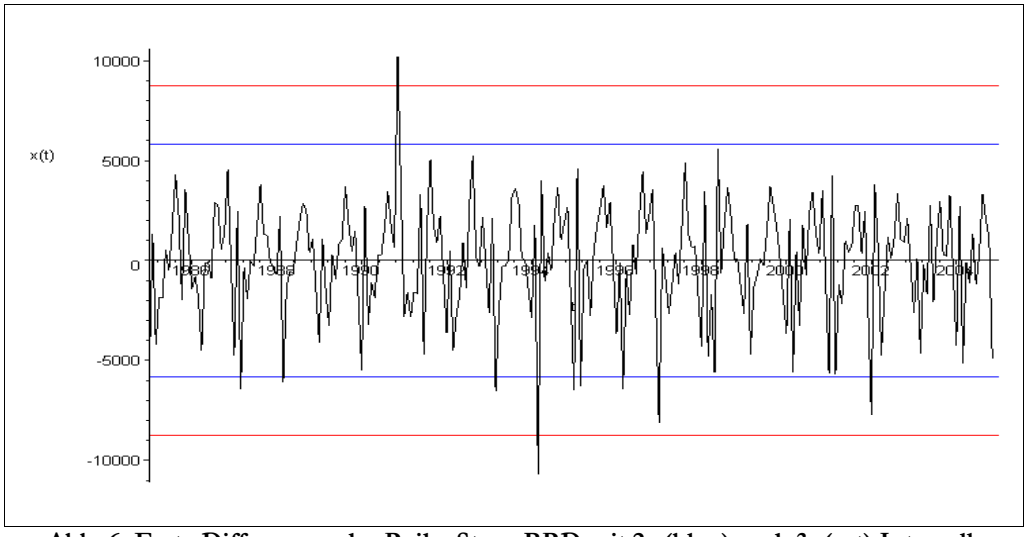


Abb. 6: Erste Differenzen der Reihe StromBRD mit 2s (blau) und 3s (rot) Intervall.

Eine weitere sehr einfache Möglichkeit der Ausreißerbestimmung sind die so genannten Box-Whisker-Plots (kurz ***Boxplot***).[26] Hierbei wird die Reihe nach ihren Werten aufsteigend sortiert und in zwei gleichgroße Teilmengen unterteilt.

Der Wert in der Mitte (***Median***) der Teilmenge mit den kleineren Werten ist das erste ***Quartil*** (unterhalb dessen 25% der Werte liegen), der entsprechende Median der Teilmenge mit den größeren Werten ist das dritte Quartil (unterhalb dessen 75% der Werte liegen). Der Median der gesamten Menge wird als Strich in einem Kästchen gezeichnet, dessen obere und untere Kante das dritte bzw. erste Quartil kennzeichnen (wodurch 50% der Werte in diesem Bereich liegen). Der Interquartilbereich (engl. interquartile range, kurz ***IQR***) wird als Differenz des ersten und dritten Quartils gebildet. Anschließend werden die Whiskers mit 1.5-fachem IQR oberhalb des dritten Quartils gezeichnet. Werte ober- und unterhalb der Whisker werden als Ausreißer betrachtet.[27] Üblicherweise wendet man die Boxplots auf die Originaldaten an. Haben die Originaldaten jedoch starke Trends, so ist es hilfreich die Boxplots der ersten Differenzen zu betrachten.

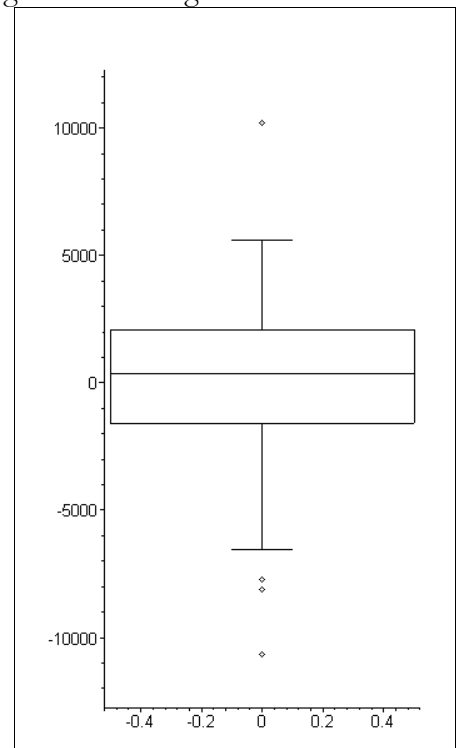


Abb. 7: Boxplot der ersten Differenzen der Reihe StromBRD.

Im Beispiel der ersten Differenzen der Reihe StromBRD werden so vier Werte als Ausreißer klassifiziert (siehe Abb. 8).

[26] Erstmals in [Tukey 83] vorgestellt. Der Name Box-Whiskers (Kästchen-Schnurrhaare) bezieht sich auf das grafische Kästchen mit den Linien ober- und unterhalb.
[27] Vergl. [EduQu 97], [Heiler 94] Absch. 4.4.

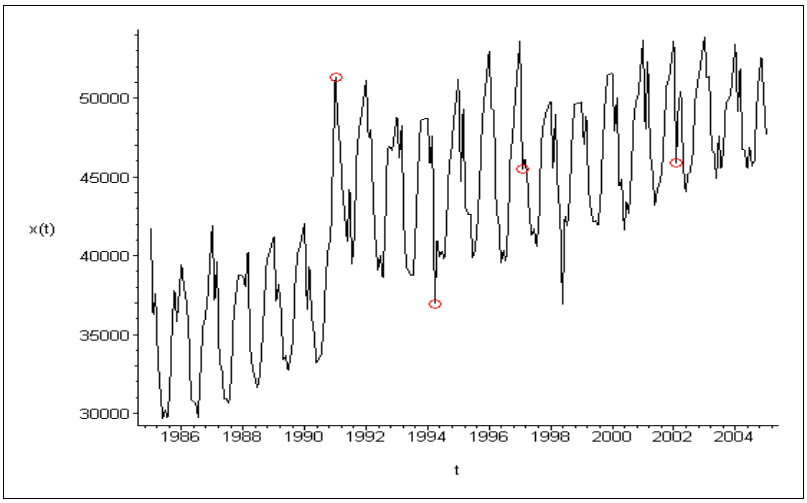


Abb. 8: Boxplot-Ausreißer der Reihe StromBRD.

Wurden Ausreißer entdeckt, ist zuerst einmal zu bestimmen, was die Fehlerquelle war, also ob es sich beim Ausreißer um einen Fehler oder lediglich um eine richtige, jedoch außergewöhnliche Beobachtung handelt. Hierzu muss Hintergrundwissen zur Beobachtungsreihe herangeholt werden. Bei der Reihe StromBRD ist zumindest für den Zeitpunkt 1991 der Grund bekannt. Um den LC zu diesem Zeitpunkt zu beheben, wird die Reihe in zwei Teilreihen $t < 1991$ und $t \geq 1991$ aufgeteilt und jeweils eine Gerade durch Regression angepasst. Der Höhenunterschied (Differenz beider Geraden am Ausreißerpunkt) zwischen beiden Geraden gibt die Höhe des LC an und ist zu den Werten mit $t < 1991$ hinzu zu addieren (siehe Abb. 10) , um diese dem Niveau der aktuelleren Werte $t \geq 1991$ anzupassen (siehe Abb. 11). In diesem konkreten Fall wäre es auch einfach möglich die Werte $t < 1991$ ganz zu verwerfen, da ab 1991 genügend Werte vorliegen. Damit die Regression der Gerade im rechten Teil nicht durch die Ausreißer verfälscht wird, wird zuerst eine quadratische Interpolation dieser Werte durchgeführt (siehe Abb. 9).

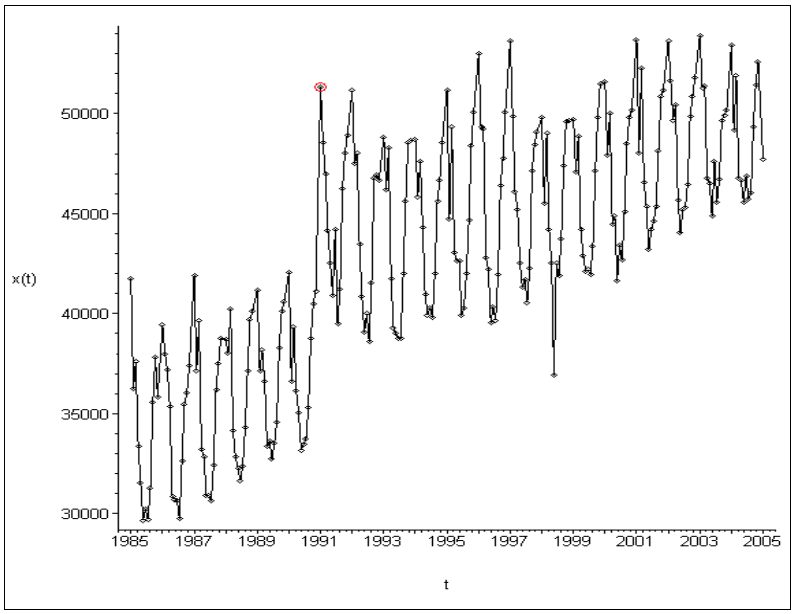


Abb. 9: Restliche Ausreißer nach der Interpolation der Ausreißer mit t>1991 der Reihe StromBRD.

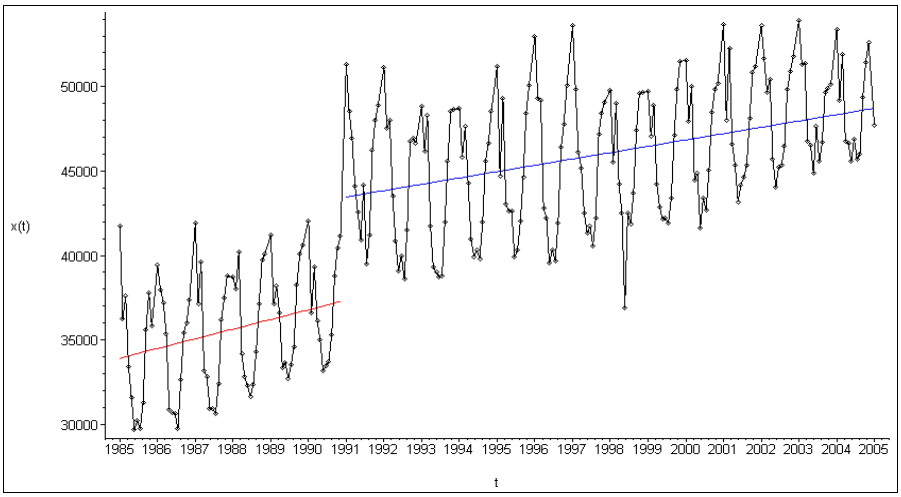


Abb. 10: Regressionsgeraden für beide Hälften am Punkt t=1991 der Reihe StromBRD.

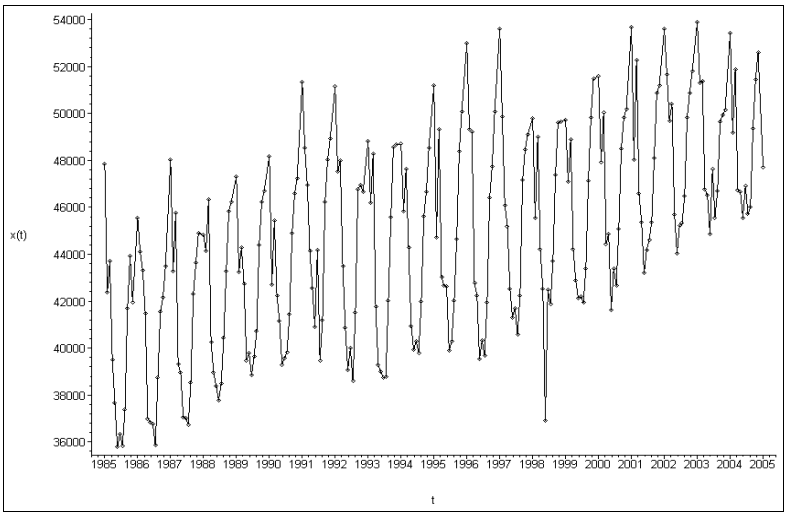


Abb. 11: Reihe StromBRD nach der Korrektur des LS.

Nach der Korrektur des LS erscheinen keine Werte mehr als Ausreißer.

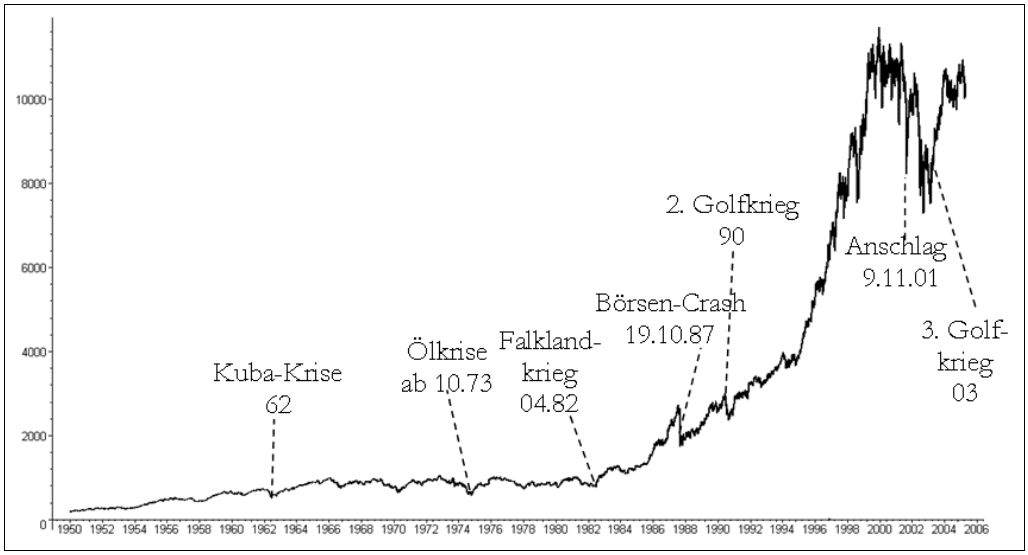


Abb. 12: Tägliche Dow-Jones Industrial Average Index Schlussnotierung 1950-2005.[28]

[28] Reihe DJ, Geschichtliche Eckdaten: http://www.weltchronik.de, http://de.wikipedia.org, http://www.quarks.de/boerse/0602.htm (Alle Abrufe 09.05.05).

Eine weitere Möglichkeit Ausreißer zu behandeln besteht darin, die Werte der Zeitreihe derart zu Skalieren, dass extremere Werte an Gewicht verlieren. Hierzu ist die Skalierung[29]

$$\hat{x}_t = \frac{x_t - \overline{x}}{s}$$

mit der Zurücktransformation

$$x_t = \hat{x}_t \cdot s + \overline{x}$$

besonders gut geeignet, da sie die Werte auf ihren Kern „zusammenschrumpft".

3.3 Klassisches Komponentenmodell für Zeitreihen

Meistens sind die zugrunde liegenden Zeitreihen nicht stationär. Vor allem bei ökonomischen Zeitreihen treten regelmäßige, Saison abhängige Höhen und Tiefen auf. Beispielsweise hat die Reihe ArbeitslosBRD im 12ten Monat regelmäßig ein Jahreshoch. Gleichzeitig erscheinen die Werte für den Monat Februar in den meisten Reihen gering, relativ zu den anderen Monaten. Dies ist u.a. darauf zurück zu führen, dass der Februar stets weniger Tage hat als die anderen Monate. Auch unter den anderen Monaten gibt es diesen Unterschied, da einige Monate 30, andere 31 Tage haben.
Gleichermaßen kann der Durchschnitt entlang der Zeitachse wachsen oder sinken. Beispiel hierfür ist der jährliche Stromverbrauch, der scheinbar konstant zunimmt (siehe Abb. 11).
Meist treten beide Varianten eines Trends zusammen auf. Um diesen Gegebenheiten gerecht zu werden, wurde das klassische Komponentenmodell aufgestellt[30]. In diesem Modell wird die zugrundeliegende Zeitreihe in einzelne Komponenten, dem ***globalen Trend*** (G_t), der ***Konjunkturkomponente*** (K_t), dem ***zyklischen Trend*** (S_t) und der ***Restkomponente*** (R_t) zerlegt (***Dekomposition***). Die Konjunkturkomponente repräsentiert mehrjährige, nicht notwendig regelmäßige Schwankungen und wird meist auf den globalen und den zyklischen Trend verteilt. Die Restkomponente spiegelt die nicht erklärbaren Einflüsse oder Störungen wieder, wobei es sich um den Teil der Zeitreihe handelt, der durch die anderen Komponenten nicht erklärt werden kann. Bei einem guten Modell wird angenommen, das die Restkomponente einem zufälligen Prozess, einem weißen Rauschen, entspricht.[31] Der globale und der zyklische Trend werden in den folgenden Abschnitten gesondert behandelt, da das Augenmerk der Trendbereinigung, bei Verteilung der Konjunkturkomponente auf diese beiden Komponenten, besonders auf ihnen liegt.
Die Zerlegung der Beobachtungen kann additiv oder multiplikativ geschehen:[32]

[29] [Thiesing 98] S. 83.
[30] Vergl. [Schlittgen 01] S. 9-12, [Leiner 82] S. 5-6.
[31] Vergl. [Schlittgen 01] S. 9., [Leiner 82] S. 5.-6.
[32] Vergl. [Leiner 82] S. 7, [Schlittgen 01] S. 9ff, [Heiler 94] S. 332ff

Additives Komponentenmodell:

$$x_t = G_t + K_t + S_t + R_t$$

Multiplikatives Komponentenmodell:

$$x_t = G_t \cdot K_t \cdot S_t \cdot R_t$$

Das multiplikative Komponentenmodell ist besonders dann angebracht, wenn die Zeitreihe aus Werten besteht, die durch eine Wachstumsrate miteinander verbunden sind.[33] Dies erkennt man u.a. daran, dass der Trend der Reihe aufwärts (oder abwärts) verläuft und die lokalen Fluktuationen dabei mit der Zeit proportional größer (oder kleiner) werden.[34]

3.3.1 Globaler Trend

Als globalen Trend oder auch ***glatte Komponente***, bezeichnet man die „langfristige systematische Veränderung des mittleren Niveaus der Zeitreihe."[35] Ist ein globaler Trend vorhanden und evtl. wachsend oder fallend, so ist die Zeitreihe nicht stationär. Durch Trendbereinigung kann die Zeitreihe jedoch in eine stationäre Reihe überführt werden. Die üblichsten Methoden hierfür sind Transformationen, Regression einer Trendfunktion oder auch Glättungsverfahren, wie der gleitende Durchschnitt.
Auch durch die Bildung der ersten (bzw. der n-ten) Differenzen benachbarter Werte kann eine Trendbereinigung durchgeführt werden. Die Differenzierung, auch ***Differenzfilter*** genannt, wird vor allem in Integrierten Modellen wie ARIMA oder SARIMA zusammen mit dem gleitenden Durchschnitt benutzt. Im Abschnitt 3.4.5 wird auf diese Technik noch genauer eingegangen.
Einen globalen Trend kann man sowohl an dem Graphen der Reihe als auch an dem ACF erkennen: Da benachbarte Werte einen gemeinsamen Trend haben, wirkt dieser stark korrelierend, wodurch der ACF einer Reihe mit großer Trendkomponente sehr langsam sinkt.
Die Abb. 13 zeigt das schematische Vorgehen zur Eliminierung des Trends bei einer Reihe mit reinem globalen Trend, also ohne saisonalen Trend. In Abb. 16 ist auch ein Vorgehensmodell bei kombiniertem globalen und saisonalem Trend dargestellt.

[33] Vergl. [Heiler 94] S.93.
[34] Vergl. [Jansen 03] S. 3.
[35] [Schlittgen 01] S. 9.

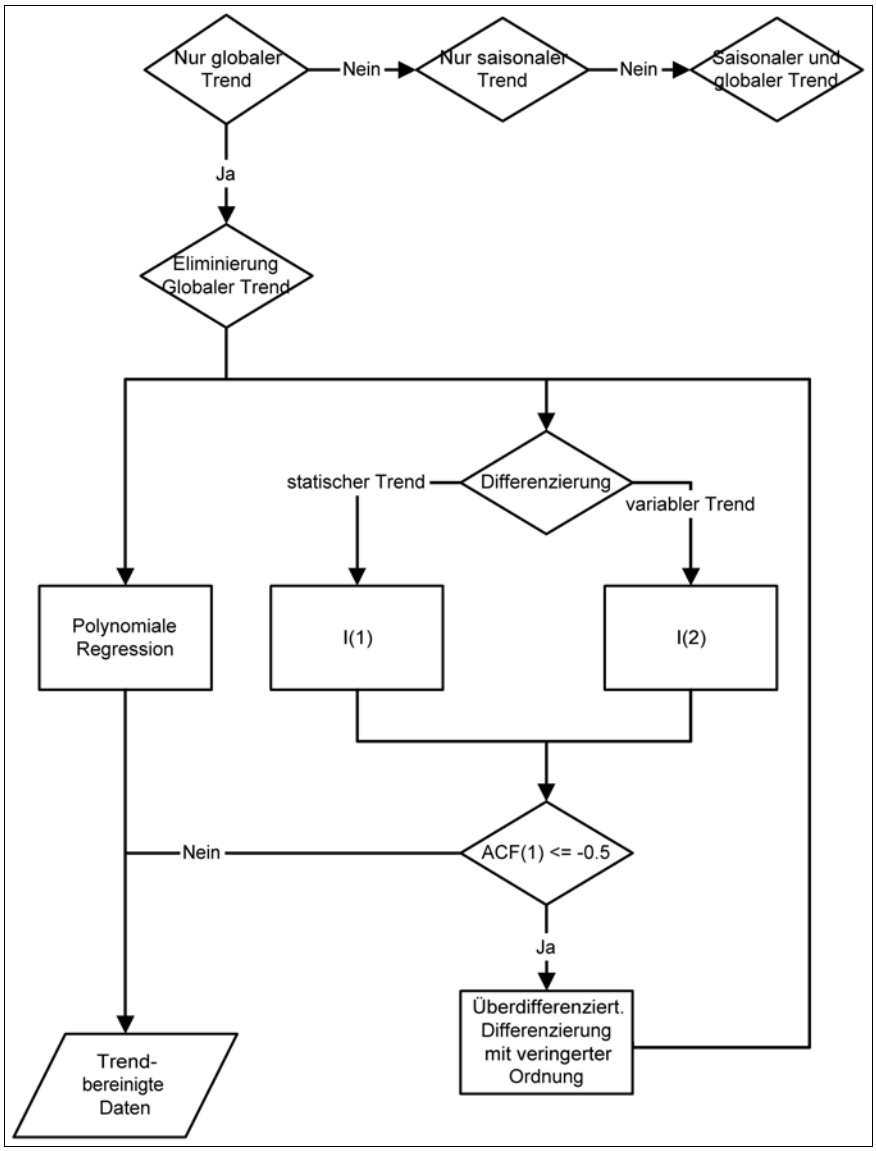


Abb. 13: Vorgehen bei einem reinen globalen Trend.

3.3.2 Zyklischer Trend (Saisonalität)

Der zyklische Trend einer Zeitreihe ist eine wiederkehrende, meist saisonbedingte Änderung des mittleren Niveaus. Bei Zeitreihen mit Beobachtungen in Zeitabständen, die kleiner als ein Jahr sind, treten wiederholt jahreszeitliche (oder auch tageszeitliche) Schwankungen auf. Man kann sich leicht vorstellen, dass die Verkaufszahlen von Speiseeis im Sommer regelmäßig höher sind als im Winter. Aber auch bei jährlichen Beobachtungen können Saisonalitäten auftreten. Hierzu ist die Zeitreihe Lynx[36] ein gutes Beispiel: die Anzahl jährlich gefangener Luchse in einem Gebiet in Kanada hat in Abständen von 10 Jahren ein Hoch (welches u.a. auf Populationsschübe zurückzuführen ist).
Der zyklische Trend kann durch entsprechende Verfahren, wie dem Phasendurchschnittsverfahren, Regression zyklischer Funktionen, wie Fourierreihen, sowie durch saisonale Differenzen eliminiert werden.
Einen zyklischen Trend kann man, wie auch einen globalen Trend, am Graphen oder an der ACF der Zeitreihe erkennen. Der ACF einer Zeitreihe mit starker zyklischer Komponente zeigt beim Lag mit der Saisonlänge einen hohen Wert.

[36] Siehe [Schlittgen 01] S. 531.

Dies liegt daran, dass die Werte mit den Abständen der Saison und deren mehrfachen eine gemeinsame Korrelation haben, nämlich den zyklischen Trend. So können auch noch unbekannte Saisonalitäten mit Hilfe des ACF aufgespürt werden.
Eine weitere Visualisierungsmöglichkeit bei vermuteter Periodenlänge der Saisonalität sind mehrere übereinander gezeichnete Graphen, die jeweils nur Werte einer Saison enthalten. Einen solchen Graphen mit einer Saisonlänge von 12 Monaten der Reihe ArbeitslosBRD aus monatlichen Beobachtungen zeigt die Abb. 14. Deutlich ist zu erkennen, dass in den Monaten Dezember und Januar das Jahreshoch liegt, und dies in allen Jahren. Außerdem ist zu erkennen, dass diese Reihe einen globalen Trend hat, da die älteren Saisons relativ zu den neueren Saisons kleinere Werte haben.

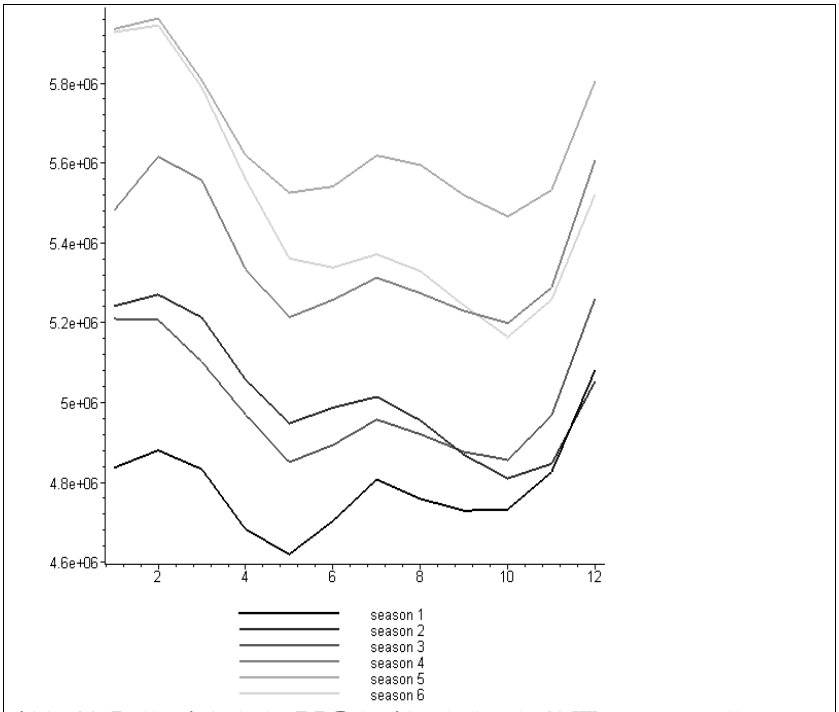


Abb. 14: Reihe ArbeitslosBRD in Abschnitte je 12 Werte unterteilt.

Eine weitere Visualisierungsmöglichkeit bildet der saisonale Indexplot, der im Abschnitt 3.4.3 im Zusammenhang mit dem Phasendurchschnittsverfahern vorgestellt wird.
Das schematische Vorgehen zur Eliminierung der saisonalen Komponente bei einer Reihe mit reinem saisonalen Trend ist in Abb. 15 dargestellt.

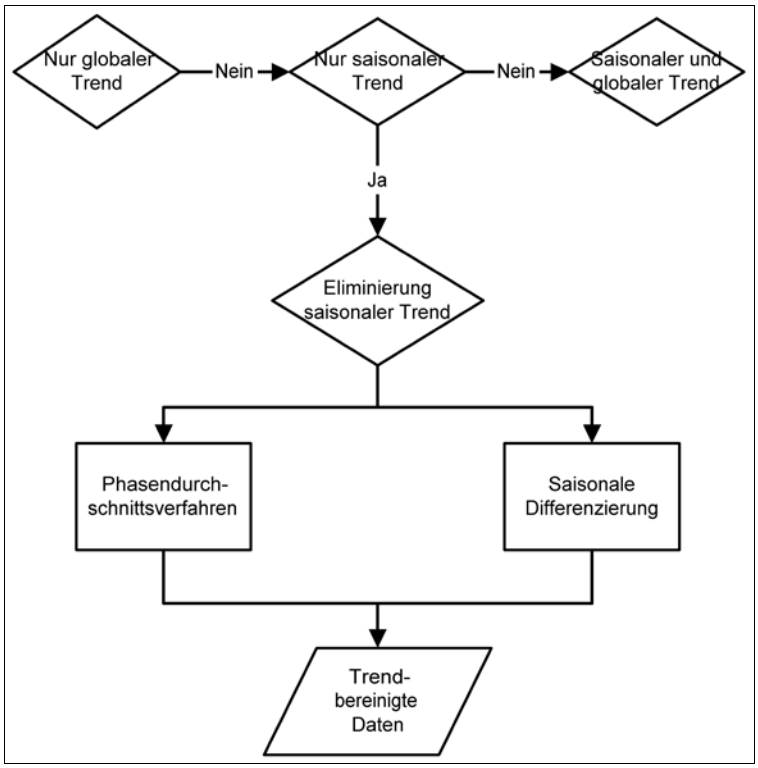


Abb. 15: Vorgehen bei einem reinen saisonalen Trend.

Liegen beide Trendvarianten vor, so müssen auch beide behandelt werden. Die Differenzierung bietet die Möglichkeit, beide Trendvarianten in einem zu eliminieren, während bei der Alternative zur Differenzierung die beiden Trendkomponenten hintereinander ausgeschaltet werden. Die Abb. 16 stellt dies schematisch dar.

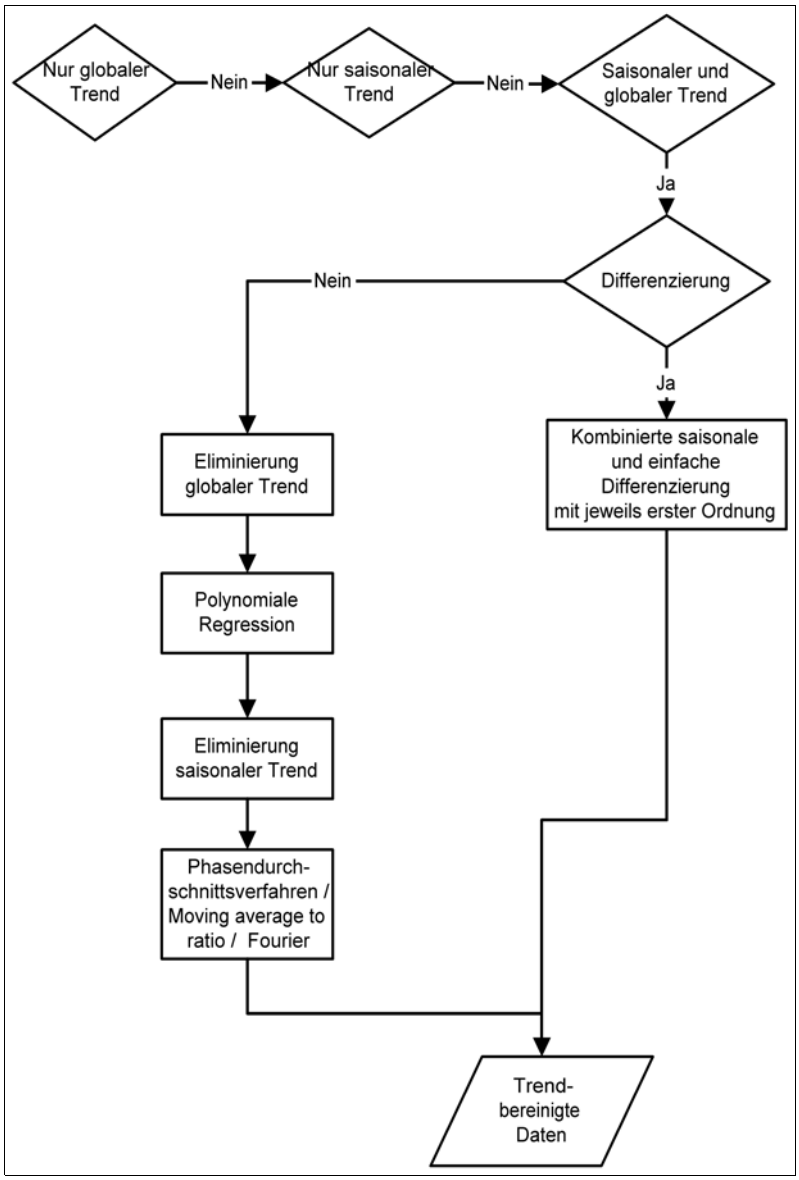


Abb. 16: Vorgehen bei einem kombinierten globalen und saisonalen Trend.

3.4 Trendbereinigung

Nach der Eliminierung des globalen und zyklischen Trends erhält man durch bilden der Differenz zwischen dem Trendwert (T_t) und der Beobachtung (x_t) die lokalen Fluktuationen der Reihe (R_t, ***Residuen***).[37]

$$R_t = x_t - T_t$$

Die Bereinigung des globalen und zyklischen Trends entspricht dem herausfiltern langsamer Schwingungen, welches vor allem in der Signalverarbeitung als ***Hochpassfilter*** bekannt ist. Diese Art Filter lassen nur Schwingungen hoher Frequenzen durch und blockieren damit die tiefen (langsamen) Frequenzen. Äquivalent wird das Bilden der Werte des Trends selber als das Filtern der schnellen Schwingungen interpretiert. Hierbei spricht man vom ***Tiefpassfilter***, der nur langsame und damit tiefe Frequenzen, wie den globalen oder zyklischen Trend, passieren lässt.

[37] [Chatfield 89] S. 13.

Die Trend bereinigten Werte werden durch das eigentliche Prognoseverfahren (z.B. AR, Exponential Smoothing) prognostiziert. Die Prognoseergebnisse müssen anschließend wieder mit dem Trend versehen werden, um mit den Originalwerten der Reihe vergleichbar zu sein.

3.4.1 Transformationen

Eine Transformation überführt die Zeitreihe in eine andere Zeitreihe, wobei wesentliche Eigenschaften der Zeitreihe erhalten bleiben sollen. Außerdem muss die Transformation invertierbar sein, es muss also möglich sein, die Originalwerte durch Anwendung einer inversen Transformation in ihre ursprüngliche Form zurück zu bringen. Im folgenden werden die am meisten verbreiteten Transformationen für Zeitreihen vorgestellt:[38]

- **Cox-Box**[39] **Power Transformation**
  Diese Transformation beschreibt eigentlich eine Schar von Transformationen, welche über den Transformationsparameter λ gesteuert werden.[40] Mit $\lambda = 1$ wird keine Transformation, für $\lambda = 0$ wird die Log-Transformation, für $\lambda = 0.5$ die Wurzeltransformation und für $\lambda = -1$ die Inverse Transformation durchgeführt. Sie ist definiert durch:

$$y_t = \begin{cases} (x_t^{\lambda} - 1)/\lambda & , \lambda \neq 0 \\ \log(x_t) & , \lambda = 0 \end{cases}$$

- **Log-Transformation**
  Durch die Log-Transformation kann ein multiplikatives Komponentenmodell in ein lineares Komponentenmodell überführt werden:[41]

$$x_t = G_t \cdot K_t \cdot S_t \cdot R_t$$
$$\Leftrightarrow \log(x_t) = \log(G_t) + \log(K_t) + \log(S_t) + \log(R_t)$$

  Besonders bei Zeitreihen mit exponentiellem Wachstum, wie dem Dow-Jones Index (Abb. 12 S.25) ist die Anwendung dieser Transformation sinnvoll. Bei Log-transformierten Werten wird für alle weiteren Verarbeitungsschritte die additiven Modelle benutzt (z.B. das additive Phasendurchschnittsmodell für die Saisonalität).

  $y_t = \log(x_t)$, $x_t > 0$

[38] Vergl. [BoxCox 64] S. 211-252 nach [Census 02].
[39] Benannt nach dem englischen Mathematiker George P.E. Box (*1919) und D.R. Cox.
[40] Vergl. [Chatfield 89] S. 12.
[41] Vergl. [Heiler 94] S. 332, [Leiner 82] S. 6.

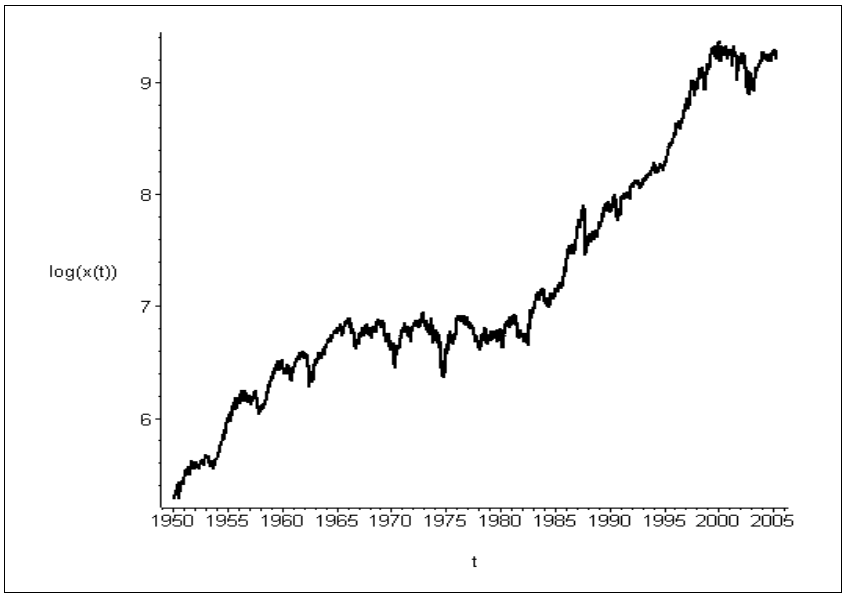


Abb. 17: Reihe DJ daily: Tägliche Dow-Jones Industial Average Index Schlussnotierung 1950-2005 nach Log-Transformation.[42]

- **Wurzeltransformation**

  $$y_t = \frac{1}{4} + 2 \cdot \left(\sqrt{x_t} - 1 \right), \; x_t \geq 0$$

- **Inverse Transformation**

  $$y_t = 2 - \left(\frac{1}{x_t} \right), \; x_t \neq 0$$

- **Logistische Transformation**

  $$y_t = \log \left(\frac{x_t}{1 - x_t} \right), \quad 0 < x_t < 1$$

  Für die logistische Transformation gibt es kein Cox-Box äquivalent.

3.4.2 Regression

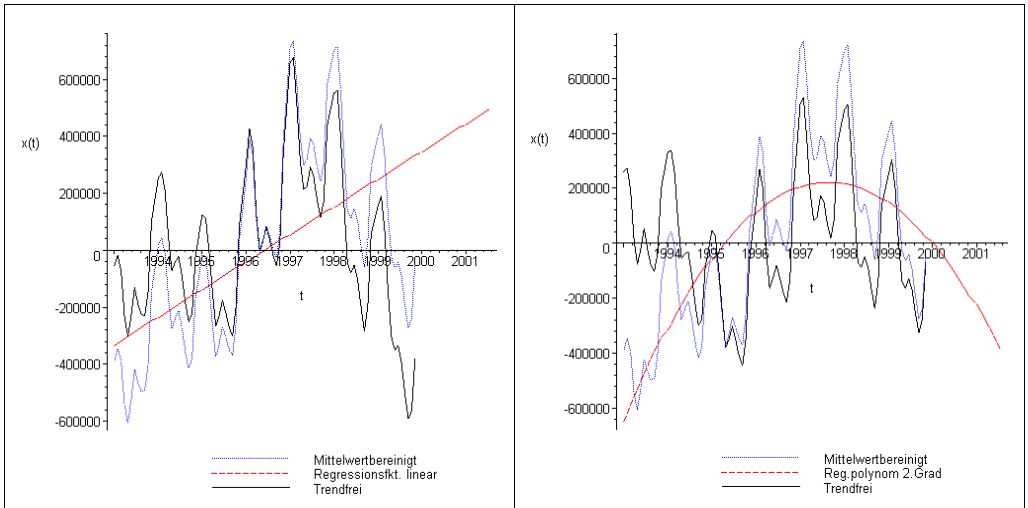
Durch Regression kann eine Trendfunktion T an die Daten angepasst und die Trend bereinigten Werte $\hat{x}_t$ durch $\hat{x}_t = x_t - T_t$ berechnet werden, um die Zeitreihe vom globalen Trend zu bereinigen. Hierbei agiert die Trendfunktion als Hochpassfilter. Die Trendfunktion kann, im einfachsten Fall, eine Gerade sein, wobei man von einem ***linearen Trend*** spricht. Mit Hilfe der Gauß'schen Methode der kleinsten Quadrate (kurz ***MKQ,*** engl. ordinary least squares, kurz OLS) kann die Gerade an die Daten angepasst werden.[43] Weitere häufig benutzte Trendfunktionen sind[44]:

[42] Dow-Jones Werte: http://www.handelsblatt.com, Geschichtliche Eckdaten um Dow-Jones: http://www.weltchronik.de, http://de.wikipedia.org, http://www.quarks.de/boerse/0602.htm (Alle Abrufe 09.05.05).
[43] Vergl. [Leiner 82] S. 11-12.
[44] Vergl. [Leiner 82] S. 9.

- **Polynome p-ten Grades** $x_t = \sum_{i=0}^{p} \left[a_i \cdot x^i \right]$

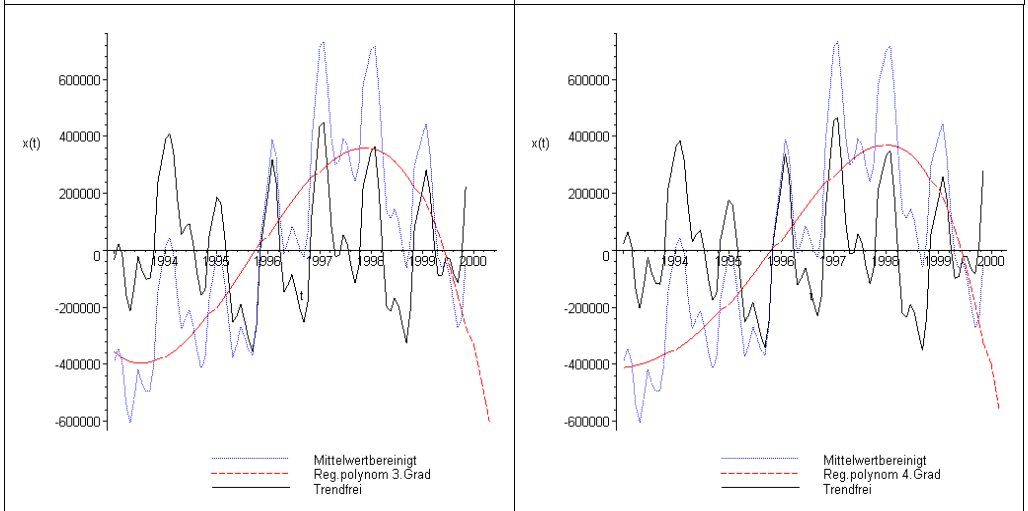
  (von denen die Gerade der Fall p=1 ist)

  Dabei werden üblicherweise Polynome mit $p \leq 3$ benutzt, da höhergradige Polynome (auch bei p=2 oder 3) an den Randbereichen sehr schnell divergieren und daher für den eigentlichen Prognosezeitraum unbrauchbar werden. Als Beispiel wurden Polynome ersten bis 10. Grades an die Zeitreihe Arbeitslosenzahlen in der BRD angeglichen (siehe Abb. 18). Die schwarz gezeichnete bereinigte Reihe zeigt für höhere Polynomgrade eine sehr gute Trendbereinigung, welches an der gleichmäßigen Verteilung der Werte um den Mittelwert 0 erkennbar ist. Jedoch zeigen die Trendpolynome ein divergentes Verhalten für den zukünftigen Bereich t>2000 und sind für die Zurücktransformation von prognostizierten Werten in diesen Bereichen unbrauchbar. Weiterhin ist zu erkennen, dass die meisten Regressionspolynome nach unten $(-\infty)$ divergieren, was schließen läßt, dass sich die Reihe gegen t=2000 bei einem Abwärtstrend befindet (siehe Abb. 18 B, C, D, F).



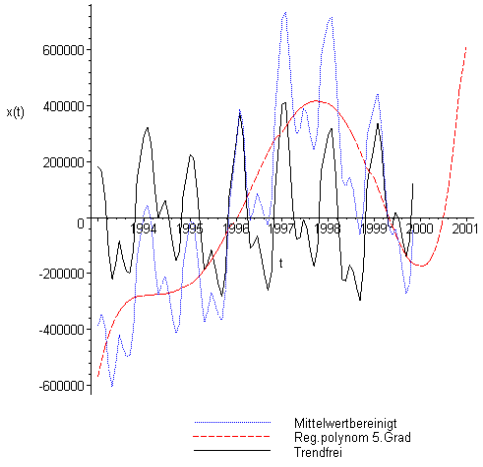
(A) Regressionsgerade

(B) Polynom 2. Grades

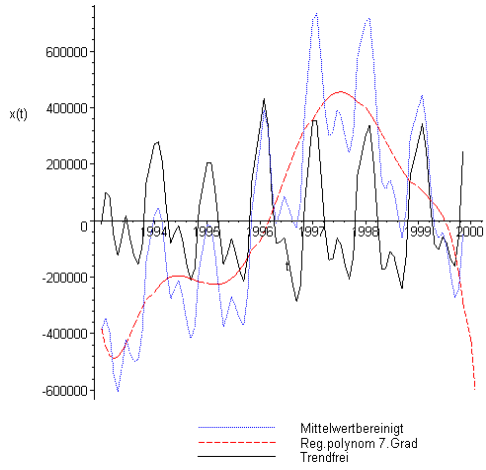


(C) Polynom 3. Grades

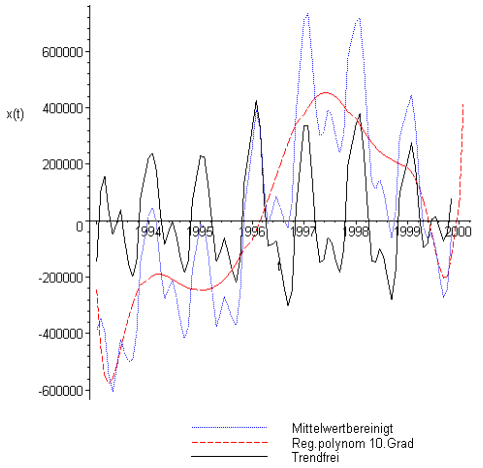
(D) Polynom 4. Grades



(E) Polynom 5. Grades



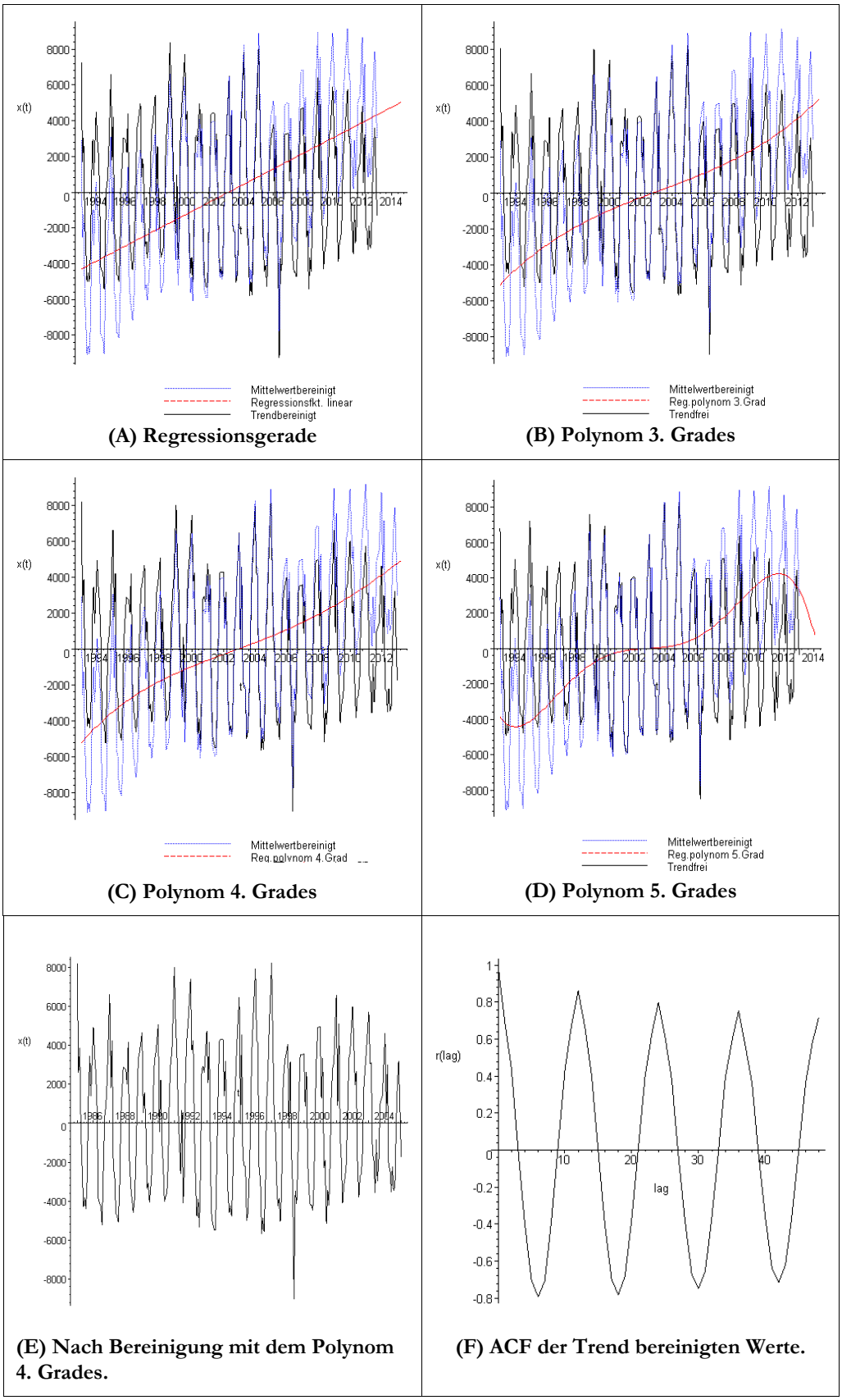
(F) Polynom 7. Grades



(G) Polynom 10. Grades

Abb. 18 A-G: Die Mittelwert bereinigte Reihe ArbeitslosBRD (blau), Regressionspolynom (rot) und Trend bereinigte Werte (schwarz) mit Polynomen verschiedener Grade.

Hingegen zeigt die Regression eines Polynoms auf die Mittelwert bereinigte Reihe StromBRD an den Randbereichen bei Polynomen bis zum 4. Grad keine Divergenz (siehe Abb. 19 A-C). Erst das Polynom 5. Grades divergiert an den Randbereichen (siehe Abb. 19 D). Für diese Reihe ist eine Polynomregression 4. Grades gut geeignet. Die ACF der Trend bereinigten Werte zeigt nun eine sehr starke saisonale Korrelation auf. Werte mit einem Abstand von 12 und mehrfachen von 12 Monaten sind stark positiv, Werte mit einem Abstand von 6 und mehrfachen Monaten sind stark negativ linear korreliert.



(A) Regressionsgerade

(B) Polynom 3. Grades

(C) Polynom 4. Grades

(D) Polynom 5. Grades

(E) Nach Bereinigung mit dem Polynom 4. Grades.

(F) ACF der Trend bereinigten Werte.

Abb. 19: (A)-(D): Die Mittelwert bereinigte Reihe StromBRD (blau), Regressionspolynom (rot) und Trend bereinigte Werte (schwarz) mit Polynomen verschiedener Grade, (E): Trend bereinigte Werte, (F): ACF der Trend bereinigten Werte.

Ein weiterer Nachteil der Polynomregression ist der Prognosebereich. Das Regressionspolynom besitzt für den zukünftigen Bereich der Prognosen

keine Freiheitsgrade mehr. Zukünftige Trendschwankungen können hierbei nicht mehr erfaßt werden.

- **Exponentialfunktionen der Form** $x_t = a \cdot e^{b \cdot t}$

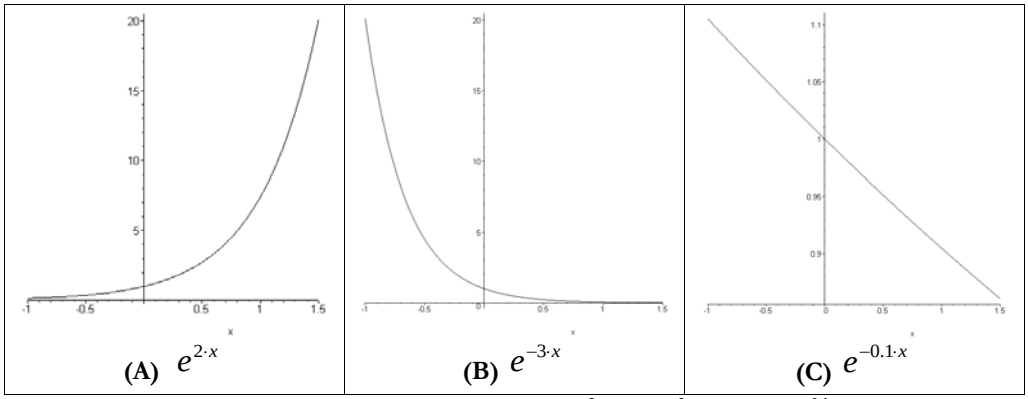


Abb. 20: Die Graphen der Funktionen (A) e^{2x}, (B) e^{-3x} und (C) $e^{-0.1x}$.

Durch Anwenden des natürlichen Logarithmus auf die ursprüngliche Zeitreihe kann ein exponentieller Trend in einen linearen Trend transformiert werden (***log-Transformation***).
Beispiel für einen exponentiellen Trend ist der Verlauf des Dow-Jones Industrial Average Index (Abb. 12 S.25). Dabei ist jedoch immer zu bedenken, dass ökonomisches oder Populationswachstum nicht unendlich fortschreiten kann. Daher empfiehlt es sich statt der Regression einer logarithmischen Funktion die log-Transformation anzuwenden.[45]

- **Logistische Funktionen**

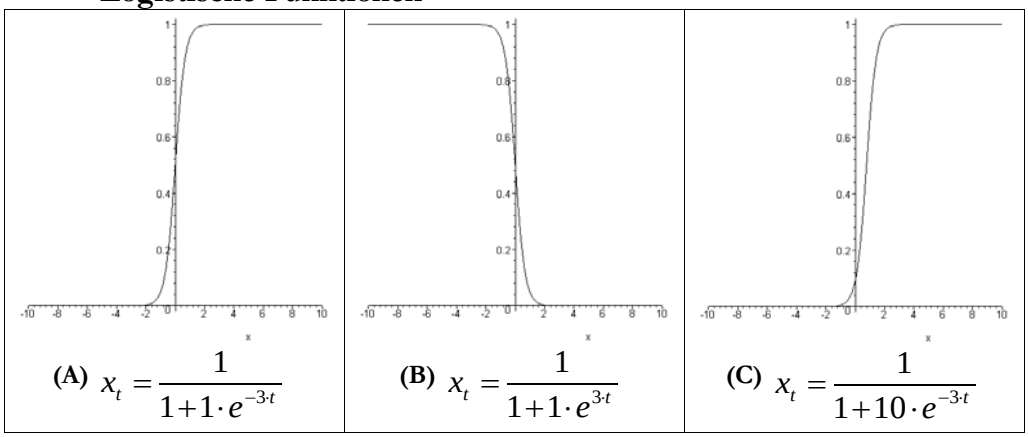


Abb. 21: Die Graphen verschiedener logistischer Funktionen.

Diese Funktionen können besonders gut auf Wachstumsprozesse angewendet werden, bei denen ein Sättigungseffekt auftritt, da diese für $t \to \pm\infty$ konvergieren (S-Form). Ein Beispiel für solche Funktionen ist $x_t = \frac{a}{1+b \cdot e^{-c \cdot t}}$.

- **Gompertz[46]-Kurven**
  Diese Funktionen haben, wie die logistischen Funktionen, ebenfalls eine S-Form und leiten sich aus der Gleichung $\log x_t = a - b \cdot r^t$ ab.

[45] Vergl. [Chatfield 89] S. 84.
[46] Benannt nach dem englischen Mathematiker Benjamin Gompertz (*1779 †1865).

Der Auswahl an Trendfunktionen sind im Prinzip keine Grenzen gesetzt. In einigen Fällen kann die Gesamtheit der Zeitreihe in Abschnitte unterteilt werden, für die jeweils verschiedene Trend-Funktionen angepasst werden können (***lokale Modelle***). Dabei müssen jedoch die Teilfunktionen durch eine Nebenbedingung an ihren Anschlussstellen stetig gemacht werden.[47] Ein Beispiel für zusammengesetzte Trendfunktionen sind ***Splines***, welche hier jedoch nicht genauer behandelt werden sollen.
Bei zyklischen Komponenten kann auch eine ***diskrete Fourier Transformation***[48] (kurz ***DFT***) durchgeführt werden. Die DFT zerlegt ein Signal (in diesem Fall die Zeitreihe) auf Basis der Fourier-Analyse (***harmonische Analyse***) in eine Summe von Sinus- und Kosinusfunktionen[49] :

$$x_t = \sum_{i=1}^{N-1} \left[a_i \cdot \cos(\omega_i \cdot t) + b_i \cdot \sin(\omega_i \cdot t) \right]$$

Äquivalent hierzu ist die komplexe Darstellung mit j als imaginäre Zahl und f_i als Fourier-Amplitude:

$$x_t = \frac{1}{N} \sum_{i=1}^{N-1} \left[f_i \cdot e^{2\pi i j k / n} \right]$$

In mathematischen Anwendungen wird die DFT auf Basis der Fast Fourier Transformation (FFT) berechnet, welche die Rechenzeit auf O(n log n) senkt. Die Fourier-Transformation liefert auch ein weiteres wichtiges Werkzeug der Analyse, welche auf den Amplituden der verschiedenen Frequenzen basiert, das ***Spektrogramm***. Im Spektrogramm werden die Amplituden (Signalstärken) gegen die verschiedenen Frequenzen aufgetragen. Eine Spitze im Spektrogramm zeigt, dass eine bestimmte Frequenz einen hohen Anteil am Gesamtsignal hat. Eine solche Frequenz deutet auf die Länge einer starken zyklischen Komponente. Das Spektrogramm ist eine äquivalente Darstellung zur ACF. Wurde die Reihe in eine Fourierreihe transformiert, können bestimmte Frequenzen benutzt werden, um entsprechende Komponenten der Reihe zu filtern.

3.4.3 Phasendurchschnittsverfahren

Das Phasendurchschnittsverfahren ist ein Saisonbereinigungsverfahren für univariate Zeitreihen ohne globalen Trend[50](welcher evtl. in einem vorangegangenen Arbeitsschritt bereinigt wurde). Die Vorgehensweise soll beispielhaft an monatlichen Daten erklärt werden und ist äquivalent auf Daten anderer Saisontypen übertragbar. Zuerst wird für jeden Monat der Durchschnitt aller Werte des gleichen Monats gebildet. Z.B. gilt für den Monat Januar (mit T=Anzahl des Vorkommens vom Monat Januar, $x_{1,Januar}$ = Januarwert des ersten Jahres, $x_{2,Januar}$ = Januarwert des zweiten Jahres,...):

$$\overline{x}_{Januar} = \frac{1}{T} \cdot \sum_{t=1}^{T} x_{t,Januar}$$

[47] Vergl. [Läuter 89] Absch. 5.1.2.
[48] Benannt nach dem französischen Mathematiker Jean Baptiste Joseph Fourier (*1768 †1830).
[49] [Chatfield 89] S. 94.
[50] Vergl. [Leiner 82] S. 54.

Allgemein gilt damit für jeden verschiedenen Monat i=1,...,12[51]

$$\overline{x}_i = \frac{1}{T} \cdot \sum_{t=1}^{T} x_{t,i}$$

Abb. 23 zeigt einen Graphen, in dem die Werte eines Monats für sukzessive Jahre zusammen gefügt wurden. Die Werte zwischen 1 und 2 repräsentieren die Januarwerte aufeinander folgender Jahre. Die roten Linien entsprechen den $\overline{x}_i$ Werten, also den Mittelwerten derjenigen Werte, die zum gleichen Monat gehören. Es ist deutlich zu erkennen, dass die Januarwerte durchgehend die höchsten Werte sind, während die Juni- und Juliwerte die kleinsten Werte sind. Entsprechendes ist auch dem Graphen in Abb. 22 zu entnehmen, in dem die Werte einer Saison einen Graphen darstellen.

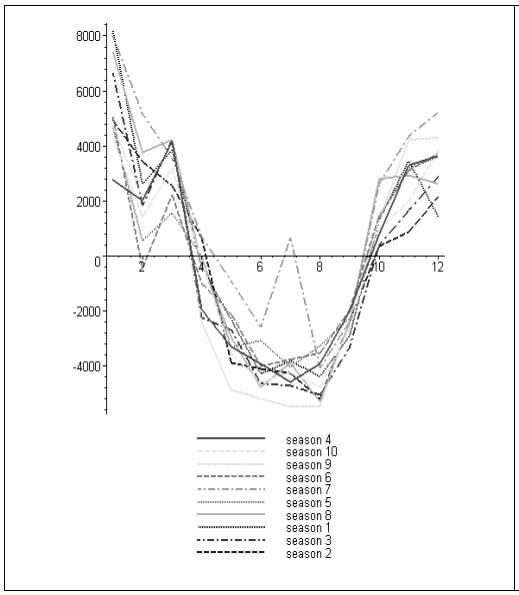


Abb. 22: Die globaler Trend bereinigten Werte der Reihe StromBRD, aufgeteilt in Abschnitte je 12 Werte.

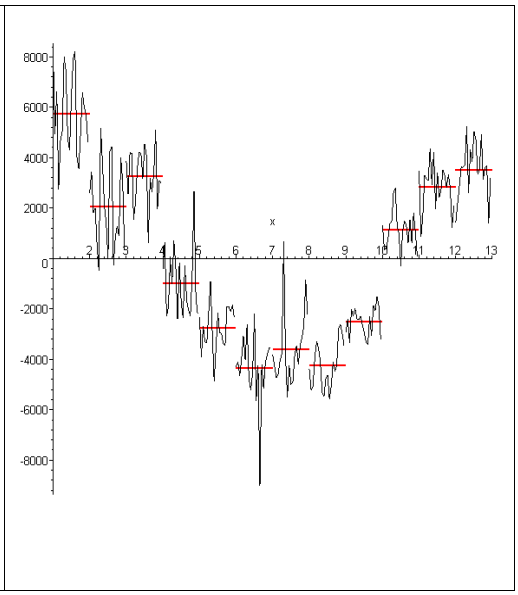


Abb. 23: Die globaler Trend bereinigten Werte der Reihe StromBRD, aufgeteilt nach ihrer Zugehörigkeit zu einem Monat (schwarz), sowie ihre Mittelwerte (rot).

Anschließend wird jeder Monatsdurchschnitt durch den Gesamtdurchschnitt aller Werte relativiert. Dies ergibt für jeden Monat i die ***Saisonmeßziffer*** s_i (auch Saisonindexziffer oder Saisonfaktor genannt):

Im multiplikativen Modell[52]: $s_i = \frac{\overline{x}_i}{\overline{x}}$

(Hierbei ist zu beachten, dass die Werte keinen Durchschnitt von 0 haben dürfen.)

Im additiven Modell[53]: $s_i = \overline{x}_i - \overline{x}$

[51] [Leiner 82] S. 55.
[52] [Leiner 82] S. 55.
[53] [Leiner 82] S. 56.

Mit Hilfe der Saisonmeßziffern s_i können nun die bereinigten Werte $\hat{x}_{t,i}$ errechnet werden:

Im multiplikativen Modell[54]: $\hat{x}_{t,i} = \frac{x_{t,i}}{s_i}$

Im additiven Modell[55]: $\hat{x}_{t,i} = x_{t,i} - s_i$

Die Abb. 24 (A) zeigt die in Saisons aufgeteilte Reihe StromBRD nach dem Phasendurchschnittsverfahren. Die Werte der verschiedenen Monate sind nun relativ zueinander gleichmäßiger verteilt. Obwohl der Januar immer noch die größte Varianz zeigt, so ist diese Varianz in beide Richtungen gegeben, wodurch diese sich im Durchschnitt aufheben. Die ACF der bereinigten Werte zeigt keine auffälligen linearen Korrelationen mehr zwischen Werten mit einem kleineren Lag als 3.

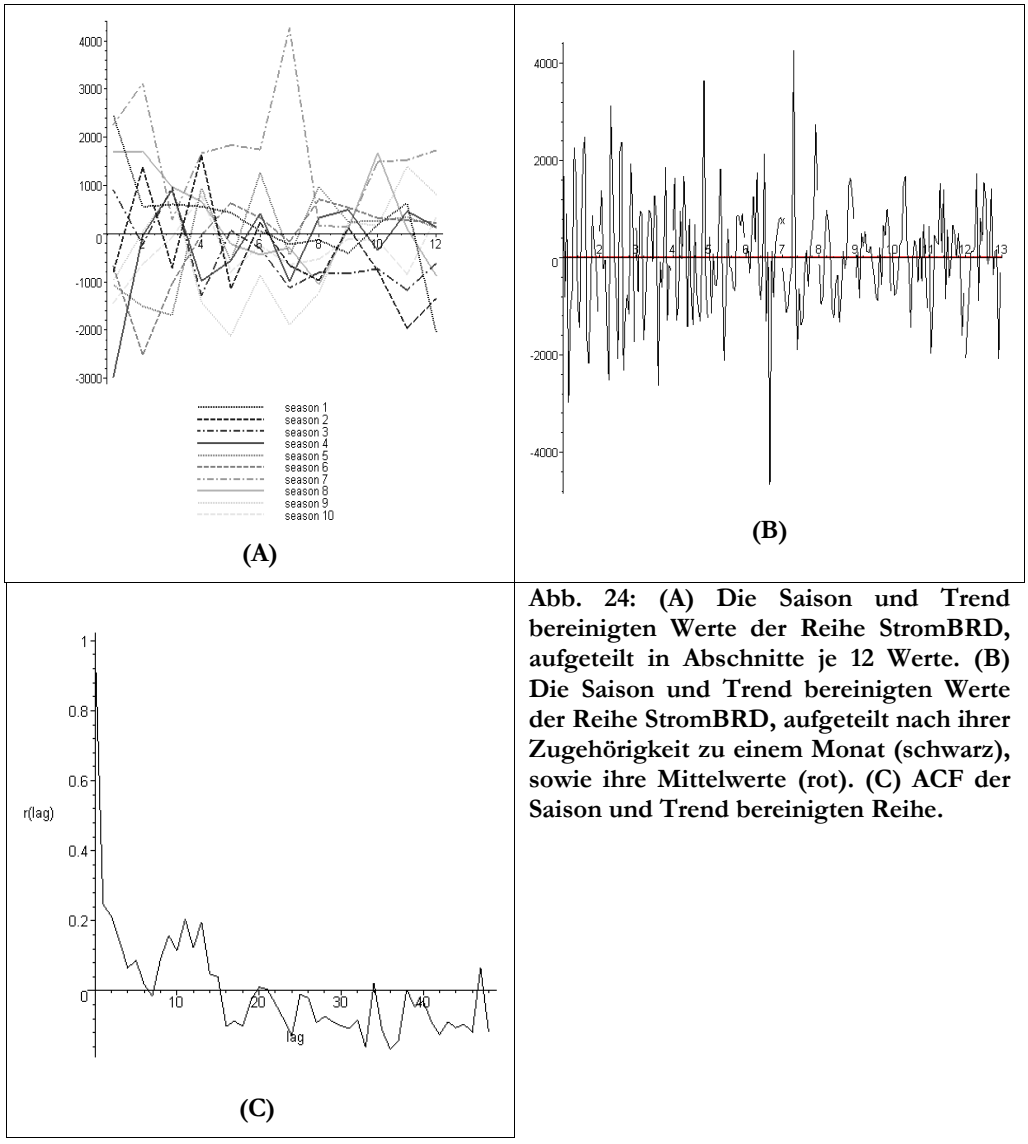


Abb. 24: (A) Die Saison und Trend bereinigten Werte der Reihe StromBRD, aufgeteilt in Abschnitte je 12 Werte. (B) Die Saison und Trend bereinigten Werte der Reihe StromBRD, aufgeteilt nach ihrer Zugehörigkeit zu einem Monat (schwarz), sowie ihre Mittelwerte (rot). (C) ACF der Saison und Trend bereinigten Reihe.

[54] [Leiner 82] S. 56.
[55] [Leiner 82] S. 56.

3.4.4 Gleitender Durchschnitt (moving average)

Der gleitende Durchschnitt (engl. moving average, kurz ***MA***) basiert auf der Bildung eines Durchschnitts in einem Stützbereich mit fester Länge n, welcher über die Zeitreihenwerte wandert. Dabei wird der Wert in der Mitte des Stützbereiches durch Einfließen der benachbarten Werte geglättet und damit die lokalen Fluktuationen der Reihe eliminiert. Ein gleitender Durchschnitt mit einem Stützbereich der Länge n wird wie folgt berechnet:

$$x_t = \frac{1}{n} \cdot \sum_{i=t-n/2}^{t+n/2} [x_i] = \frac{1}{n} \cdot x_{t-n/2} + \ldots + \frac{1}{n} \cdot x_{t-1} + \frac{1}{n} \cdot x_t + \frac{1}{n} \cdot x_{t+1} + \ldots + \frac{1}{n} \cdot x_{t+n/2}$$

Man beachte, dass die Summe der Koeffizienten stets 1 ergeben muss, da sonst der Mittelwert verfälscht wird. Damit der Stützbereich eine Mitte hat, muss die Länge des Stützbereiches n ungerade sein. Damit können die n/2 Werte am Anfang und am Ende der Zeitreihe auch nicht geglättet werden, da für diese kein entsprechender Stützbereich möglich ist. Dies ist auch der größte Nachteil des einfachen MA, da für die aktuellsten n/2 Werte keine geglätteten Werte vorliegen.
Eine Erweiterung des MA ist der ***gewichtete gleitende Durchschnitt*** (engl. weighted moving average, kurz ***WMA***). Hierbei werden für verschiedene Stellen des Stützbereiches verschiedene Koeffizienten benutzt, deren Summe stets 1 ergeben muss. Ein einfacher Vertreter ist der WMA mit der Stützbereichgröße 13, der im Census X11-Verfahren zur Trendbereinigung benutzt wird[56]. Die Reihe (monatlicher Werte) wird durch den gleitenden Durchschnitt

$$x_t = \frac{1}{24} \cdot x_{t-6} + \frac{1}{12} \cdot x_{t-5} + \frac{1}{12} \cdot x_{t-4} + \ldots + \frac{1}{12} \cdot x_{t+4} + \frac{1}{12} \cdot x_{t+5} + \frac{1}{24} \cdot x_{t+6}$$

geglättet. Die Gewichtung wird hier benutzt, um mit einem WMA mit einem Stützbereich von 13 Werten einen MA mit einem Stützbereich von 12 Werten zu simulieren. Deswegen werden die beiden Werte an den Rändern des Stützbereiches, die für den gleichen Monat stehen, jeweils halb so stark gewichtet wie die anderen Werte.
Die folgende Abb. 25 zeigt für einen Ausschnitt der Reihe StromBRD drei verschiedene gleitende Durchschnitte. Es ist deutlich erkennbar, dass größere Stützbereiche zu einer stärkeren Glättung führen.

[56] Vergl. [Census 02] S. 9.

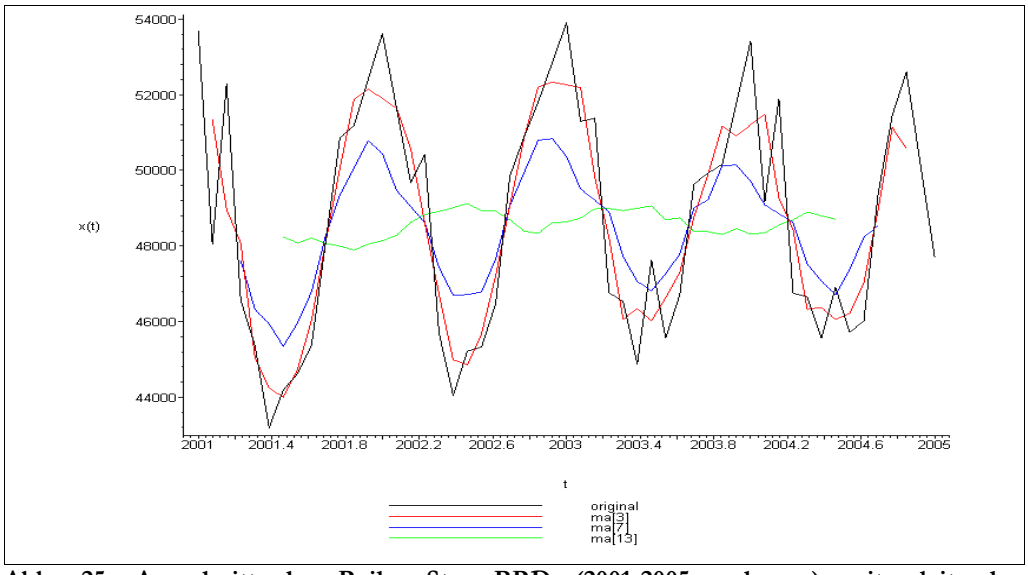


Abb. 25: Ausschnitt der Reihe StromBRD (2001-2005, schwarz) mit gleitenden Durchschnitten der Ordnung 3 (rot), 7 (blau) und 13 (grün).

Die bis hierhin genannten gleitenden Durchschnitte stellen ***symmetrische*** Filter dar, da ihre Koeffizienten symmetrisch zum Mittelpunkt des Stützbereiches sind. Durch eine ***asymmetrische*** Gewichtung kann eine Schätzung für den Wert am rechten Ende des Stützbereiches erstellt werden. Hierzu nimmt man an, dass die Vorgängerwerte weniger Einfluss auf den zu schätzenden Wert haben, je weiter sie zeitlich von dem zu schätzenden Wert entfernt liegen. Unter der Grundbedingung, dass die Summe der Koeffizienten 1 ergibt, können beliebige Werte benutzt werden, wie z.B. $\left[\frac{1}{6},\frac{2}{6},\frac{3}{6}\right]$

Einen besonderen Fall der WMA bildet das ***exponentielle Glätten***, welches im Absch. 3.5.4 genauer erläutert wird.

3.4.5 Differenzfilter

Der Differenzoperator[57] ∇^d wird definiert als die Differenz der d-ten Ordnung benachbarter Werte. Es gilt daher $\nabla^1 = x_t - x_{t-1}$ und entspricht der einfachen Differenzierung der Funktion, welche die Werte der Reihe produziert. Entsprechend werden die Differenzen 2. Ordnung als $\nabla^2 = x_t - 2 \cdot x_{t-1} + x_{t-2}$ und allgemein die Differenzen der d-ten Ordnung als

$$\nabla^d = \sum_{k=0}^{d} \binom{d}{k} \cdot x_{t-k} \cdot (-1)^k$$

definiert.[58] Der Differenzoperator ist ein wichtiger Bestandteil der integrierten Modelle, wie dem ARIMA oder SARIMA-Modell von Box-Jenkins, und dient der Eliminierung eines globalen Trends der Ordnung d.
Eine Erweiterung des Differenzoperators bildet der saisonale Differenzoperator[59] ∇_S^D, welcher Differenzen über ein Lag von S Werten (einer Saison) bildet. Ein saisonaler Differenzoperator der Ordnung D=1 für monatliche Daten (S=12)

[57] [Box-Jenkins 76] nach [Thiesing 98] S. 84, [Chatfield 89] S. 60.
[58] Vergl. [Schlittgen 01] S. 39.
[59] [Box-Jenkins 76] nach [Chatfield 89] S. 60., [Schlittgen 01] S. 39ff.

wird gebildet als $\nabla_{12}^{1} = x_t - x_{t-12}$. Entsprechend gilt für einen saisonalen Differenzoperator 2. Ordnung $\nabla_{S}^{2} = x_t - 2 \cdot x_{t-S} + x_{t-2S}$ und allgemein

$$\nabla_{S}^{D} = \sum_{k=0}^{D} \binom{D}{k} \cdot x_{t-k \cdot S} \cdot (-1)^k$$

Der saisonale Differenzoperator dient vor allem der Eliminierung der saisonalen Komponente und wird im SARIMA-Modell von Box-Jenkins zusammen mit dem normalen Differenzoperator benutzt. Dabei gilt für den Gesamt-Differenzoperator $\nabla^{d} \nabla_{S}^{D}$ im Fall d=1, D=1 und S=12 $\nabla^1 \nabla_{12}^1 = \nabla_{12}^1 x_t - \nabla_{12}^1 x_{t-1} = (x_t - x_{t-12}) - (x_{t-1} - x_{t-13})$.[60]
Die saisonale Differenzbildung wird also vor der normalen Differenzbildung angewendet. Der kombinierte Differenzoperator bildet die d-te Ableitung der Reihe unter Berücksichtigung der D-ten Ableitung der saisonalen Komponente und kann damit beide Trendkomponenten ausschalten.

Mit jeder Differenzierung geht auch Information über die eigentlichen Daten verloren. Eine Überdifferenzierung sollte vermieden werden, da sie keinen starken Bezug mehr zur undifferenzierten Reihe aufweist. Bei jeder Differenzierung ist zu beobachten, dass die ACF der differenzierten Reihe bei Lag 1 zunehmend negativ wird. Zusätzlich steigt die Standardabweichung der differenzierten Reihe an dem Punkt, bei dem die Überdifferenzierung stattfindet. Eine optimale Ordnung der Differenzierung kann daher mit Hilfe folgender Regeln ermittelt werden[61]:

1. Ist die Reihe stationär, so bedarf sie keiner Differenzierung.
2. Hat die Reihe einen linearen Trend, so sollte sie einfach differenziert werden.
3. Eine Reihe mit einem variablen Trend sollte zweifach differenziert werden.
4. Wenn die ACF der differenzierten Reihe bei Lag 1 kleiner oder gleich -0.5 wird, so könnte die Reihe über differenziert sein.
5. Die optimale Ordnung der Differenzierung ist oft diejenige, bei der die Standardabweichung minimal wird.

Am Beispiel der Reihe StromBRD werden diese Regeln verdeutlicht: In Abb. 26 kann man links den Wert der ACF bei Lag 1 für verschiedene Differenzierungsordnungen erkennen. Auf der rechten Seite ist das Minimum der Standardabweichungen für die verschiedene Differenzierungsordnungen bei der Ordnung 1 deutlich erkennbar. Die 5. Regel, welche besagt, dass die Differenzierungsordnung mit der kleinsten Standardabweichung bzw. Varianz die am besten geeignete ist, kann auch auf die saisonale Differenzierung angewendet werden. Reihen mit einem globalen oder saisonalen Trend haben eine starke Streuung um ihren Mittelwert, letztendlich verursacht durch die Trendkomponente. Mit der Differenzierung wird eine Trendkomponente (teilweise) ausgeschaltet, wodurch die Werte der Reihe weniger um den Mittelwert verstreut liegen.

[60] [Chatflied 89] S. 60.
[61] Vergl. [Nau 03] Absch. „Identifying the order of differencing“.

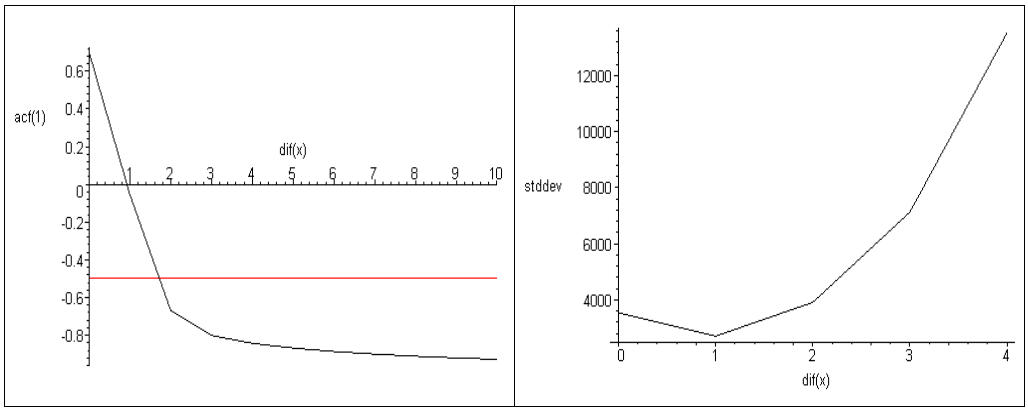


Abb. 26: Differenzierungen der Reihe StromBRD bis zu einer Ordnung von 10. Links: ACF bei Lag 1 (schwarz) und Grenzwert bei -0.5 (rot). Rechts: Standardabweichung der differenzierten Reihen.

Die im Abschnitt 3.4.3 Phasendurchschnittsverfahren vorgestellten Plots mit Werten und ihrer Zugehörigkeit zu einem Monat verdeutlichen die Wirkung der beiden Differenzoperatoren auf die Saisonkomponente der Reihe StromBRD in der folgenden Abb. 27. Bei der einfachen Differenzierung bleibt die Saisonkomponente erhalten (Abb. 27 links). Bei der saisonalen Differenzierung wird die Saisonkomponente eliminiert (Abb. 27 rechts).

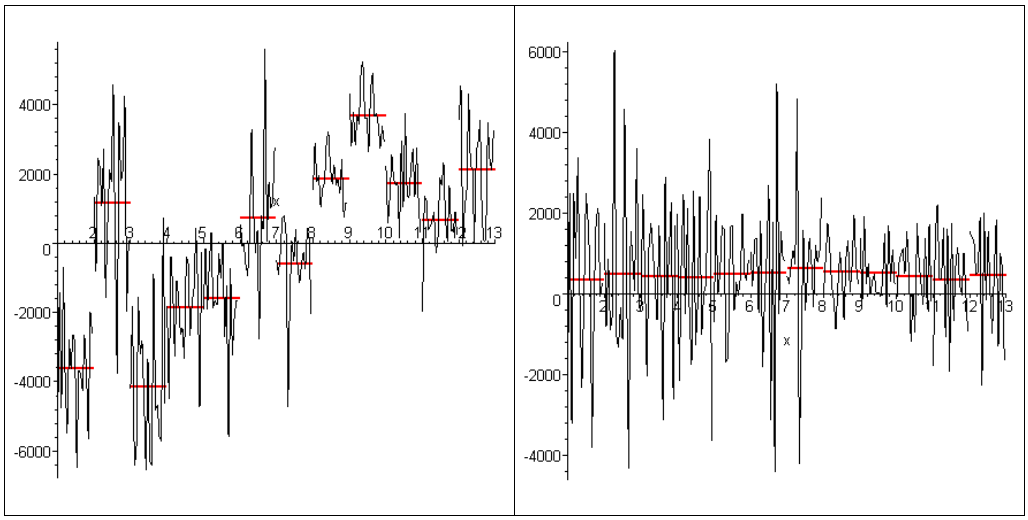


Abb. 27: Links: Die einfach differenzierten Werte der Reihe StromBRD, aufgeteilt nach ihrer Zugehörigkeit zu einem Monat (schwarz), sowie ihre Mittelwerte (rot). Rechts: Die einfach saisonal differenzierten Werte der Reihe StromBRD, aufgeteilt nach ihrer Zugehörigkeit zu einem Monat (schwarz), sowie ihre Mittelwerte (rot).

Zur Zurücktransformation der Daten von ihrer differenzierten Form in die ursprüngliche Form werden die ersten Originalwerte der Reihe benötigt. Die Ordnung der Differenzierung gibt vor, wieviele Originaldaten benötigt werden. Bei einer Differenzierung der Ordnung 2 ergibt sich der differenzierte Wert d als:

$$d=x_t-2x_{t-1}+x_{t-2}$$

Löst man die Gleichung nach x_t auf, so erhält man:

$$x_t=d+2x_{t-1}-x_{t-2}$$

Es werden also genauso viele Originalwerte benötigt, wie die Ordnung der Differenzierung ist, um aus den Differenzen wieder die Originalwerte zu bestimmen.

3.4.6 Hauptkomponentenanalyse (PCA)

Die Hauptkomponentenanalyse ist eine Analyse der Kovarianzen mehrerer Reihen, die bei multivariaten und univariaten Zeitreihenanalyse zur Reduzierung der Datendimensionalität sowie zur Eliminierung eines linearen globalen Trends eingesetzt werden kann. Die PCA dekorreliert die Vektormerkmale und liefert die Eigenwerte sowie die Eigenvektoren. Die Eigenvektoren (Hauptachsen) geben an, in welche Richtungen die größten Varianzen verlaufen. Die Eigenwerte geben an, welchen Anteil eine Dimension der Daten bezogen auf die Hauptachsen an der Varianz der gesamten Daten haben. Dimensionen, die eine kleine Varianz haben (daher recht flach verlaufen), haben entsprechend einen kleineren Eigenwert und enthalten wenig Information. Die Besonderheit, dass die Eigenvektoren orthogonal zueinander sind, ermöglicht es, diese als neues Koordinatensystem zu benutzen. Dadurch können die Daten so gedreht werden, dass die Dimension mit der größten Varianz der x-Achse entspricht. Einige einfache Beispiele sind in [Smith 02] zu finden.
Bei der Hauptachsentransformation mit Hilfe der Eigenvektoren können einige der Eigenvektoren, vor allem jene mit einem kleinen Eigenwert, ausgelassen werden. Mit dem Entfernen jedes Eigenvektors wird bei der Transformation auch eine Datendimension entfernt. Diese Kompression ist daher zwar verlustbehaftet, doch kann über die Eigenwerte genau kontrolliert werden, wieviel Information verloren geht.
Die Durchführung der PCA besteht aus folgenden Schritten:

1. Mittelwert Bereinigung der Datenspalten
2. Ermitteln der Kovarianzmatrix der Datenspalten
3. Berechnen der Eigenwerte und Eigenvektoren mit Hilfe der Kovarianzmatrix
4. Berechnen des prozentualen Anteils jedes Eigenvektors an der Summe aller Eigenvektoren
5. Entfernen der Eigenvektoren, welche einen geringen Anteil haben (z.B. <1%)
6. Multiplikation der restlichen Eigenvektoren mit den Originaldaten

Das Resultat des letzten Schrittes sind die transformierten, dekorrelierten und reduzierten Daten. Statistisches Rauschen in Daten ist meist hochfrequent, welches heißt, dass das Rauschen sich mit jedem Schritt ändert. Die eigentlichen Muster hingegen haben eine längerfristige Struktur und sind daher meist niederfrequent, wie das beispielsweise bei der saisonalen Komponente der Fall ist. Durch die PCA werden die eigentlichen Muster, welche auch die höchste Varianz ausmachen, übernommen, das Rauschen, welches die geringere Varianz hat, wird geglättet. Durch die PCA wird also auch Rauschen entfernt.

3.5 Prognose

„Ein Blick in die Vergangenheit hat nur Sinn, wenn er der Zukunft dient."
Konrad Adenauer[62]

Nach der Eliminierung der Trendkomponenten können die Residuen nun durch eine Prognosemethode modelliert werden. Im folgenden werden klassische lineare Prognosemethoden vorgestellt und erläutert.

3.5.1 Multiple lineare Regression

Bei multivariaten Zeitreihen mit mehreren exogenen Variablen ($X_1 \ldots X_N$) ist es möglich, die endogene Variable (Y) anhand der exogenen Daten zu beschrieben. Dazu wird eine Hyperebenengleichung gebildet[63]:

$$Y = a_0 + a_1 \cdot X_1 + a_2 \cdot X_2 + \ldots + a_N \cdot X_N = \sum_{i=0}^{N} a_i X_i$$

Die Bestimmung der Koeffizienten a_i geschieht anhand mehrerer linearer Gleichungssysteme, die aus den einzelnen Werten für die exogenen und der endogenen Variablen gebildet werden. a_0 ist mit dem y-Achsenabschnitt der zweidimensionalen Regression einer Geraden vergleichbar. Die Lösung der Gleichungssysteme wird mit Hilfe der MKQ bestimmt. Hierfür werden die Werte der N-Reihen mit jeweils M-Werten wie folgt als Matrize geschrieben:

$$X = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & \ldots & x_{N,1} \\ 1 & x_{1,2} & x_{2,2} & \ldots & x_{N,2} \\ 1 & \ldots & \ldots & \ldots & \ldots \\ 1 & x_{1,M} & x_{2,M} & \ldots & x_{N,M} \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \ldots \\ y_M \end{bmatrix}, \quad A = \begin{bmatrix} a_1 \\ a_2 \\ \ldots \\ a_M \end{bmatrix}$$

Die erste Spalte der Matrix X spiegelt die Werte der Zeitreihe wieder, welche den Koeffizienten a_0 haben (und daher konstant den Wert 1 besitzen). Die zweite Spalte enthält alle Werte der exogenen Reihe X_1 (daher der erste Index 1).
Die MKQ wird dann wie folgt definiert[64]:

$$A = (X^T \cdot X)^{-1} (X^T \cdot Y)$$

Mit M^T als die Transponierte und M^{-1} als die Inverse der Matrix M.

[62] Konrad Adenauer (*1876 †1967), erster Bundeskanzler der BRD.
[63] Vergl. [Pokropp 94] Kapitel 2.
[64] Nach [Pokropp 94] S. 22.

3.5.2 Autoregression

Ein univariates Verfahren zur Prognose von Zeitreihen sind die autoregressiven Modelle (kurz ***AR***), welche von Yule vorgestellt wurden[65]. AR-Modelle bestehen auf der Annahme, das die Werte einer Reihe einen linearen Zusammenhang zu den vorangegangenen Werten haben, unter der Berücksichtigung dass durch zufällige Einflüsse (Shocks, Rauschen) der lineare Zusammenhang verändert wird. Die Ordnung p eines AR-Modelles AR(p) ist die Größe des Lags, welche festlegt, wie weit beeinflussende Werte in der Vergangenheit liegen dürfen. Die rekursive Gleichung

$$f_t = \theta_1 \cdot x_{t-1} + \theta_2 \cdot x_{t-2} + \ldots + \theta_p \cdot x_{t-p} = \sum_{i=1}^{p} [\theta_i \cdot x_{t-i}] \quad [66]$$

beschriebt das allgemeine AR-Modell. Für ein AR(p)-Modell gilt es die Koeffizienten $\theta_1 \ldots \theta_p$ der rekursiven Gleichung zu bestimmen. Dies entspricht genau der MLR, wobei hier statt exogener Variablen die Werte aus der Vergangenheit der Reihe selbst benutzt werden. Ein sinnvoller Wert für die Ordnung p eines AR-Modells kann durch Ermitteln der Koeffizienten für verschiedene Laggrößen bestimmt werden. Dabei ist besonders der letzte Koeffizient θ_p interessant. Sein Wert gibt darüber Aufschluß, wie groß der Beitrag des letzten Lagabschnittes zum gesamten Modell ist. Ist der Wert unsignifikant klein (z.B. $\theta_p < 0.2$), so ist der Beitrag von x_{t-p} zur Schätzung sehr gering. Die (diskrete) Funktion der Koeffizienten θ_p bei verschiedenen Laggrößen wird als ***partielle Autokorrelationsfunktion***, kurz ***PACF*** bezeichnet.

Die PACF der Saison und Trend bereinigten Reihe StromBRD zeigt beim Lag 1 den letzten signifikanten Wert (siehe Abb. 28, der PACF bei Lag 2 beträgt etwa 0.15). Hier ist also ein AR(1) Modell angebracht.

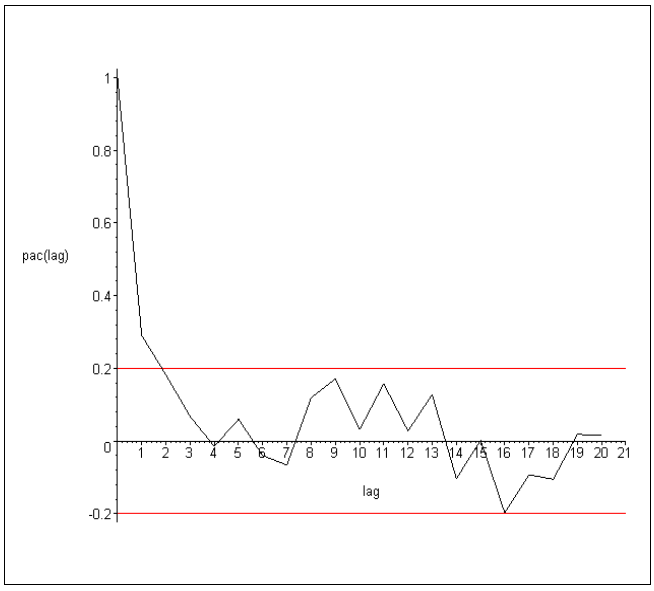


Abb. 28: PACF der Saison und Trend bereinigten Reihe StromBRD (schwarz) mit Signifikanzintervall bei 0.2 und -0.2 (rot).

[65] [Yule 27] nach [Pollock 96] S. 7.
[66] Vergl. [Pollock 96] S. 7, [Schlittgen 01] Absch. 2.3.4.

Die einschritt Prognosen des AR(1)-Modells (siehe Abb. 29) sowie die Residuen (siehe Abb. 30 links) und die ACF der Residuen (siehe Abb. 30 rechts) bestätigen die Qualität des AR(1)-Modells, denn der ACF kann entnommen werden, dass keine signifikanten kurzzeitigen Korrelationen mehr vorhanden sind.

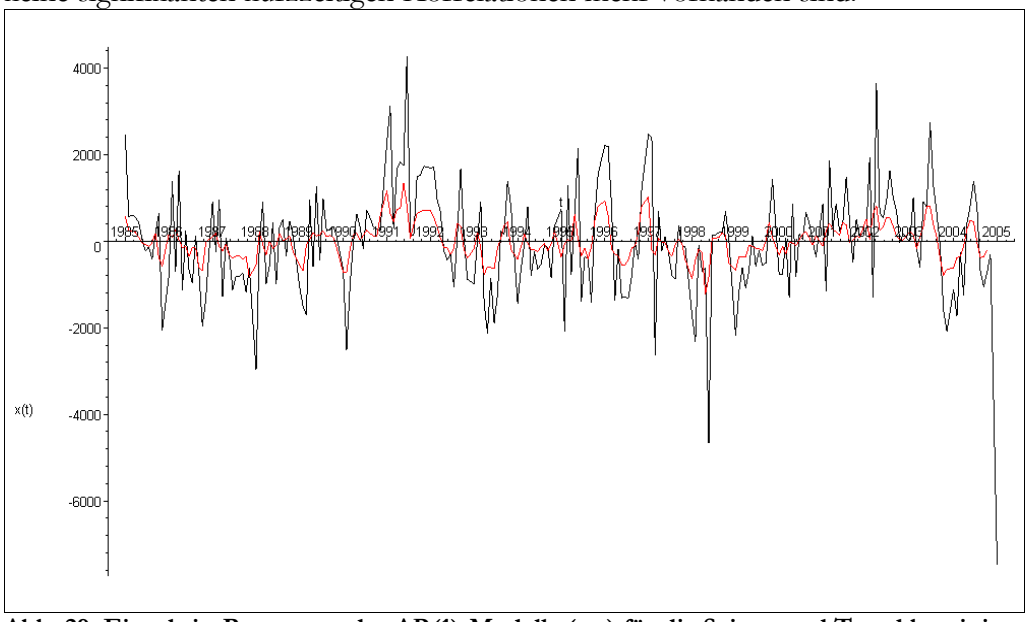


Abb. 29: Einschritt Prognosen des AR(1)-Modells (rot) für die Saison und Trend bereinigte Reihe StromBRD (schwarz).

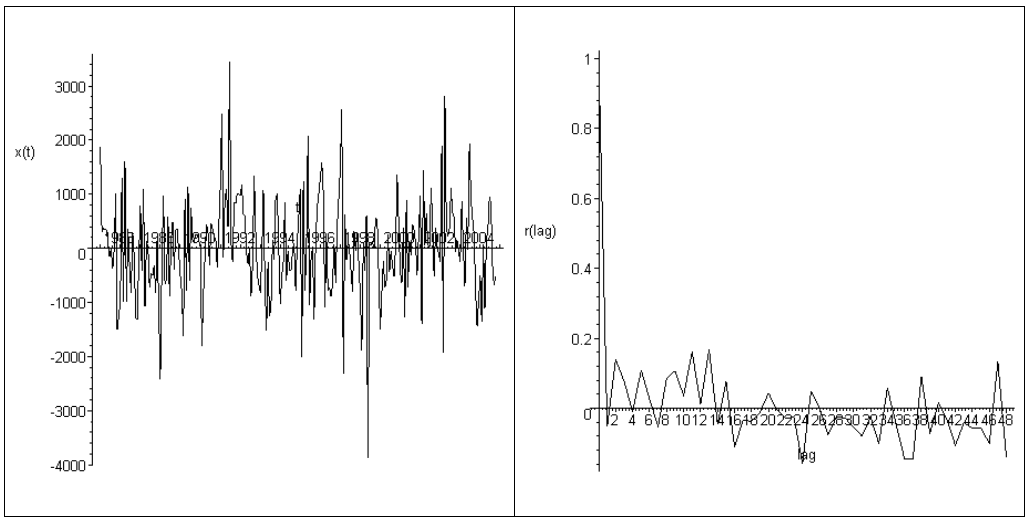


Abb. 30: Residuen und ACF der Residuen der einschritt Prognosen des AR(1)-Modells für die saisonbereinigte Reihe StromBRD.

Nach dem Hinzufügen der saisonalen und der Trendkomponente der einschritt Prognosen ist erkennbar, dass das Modell die grundlegende Form der Reihe recht gut erfasst hat (siehe Abb. 31).

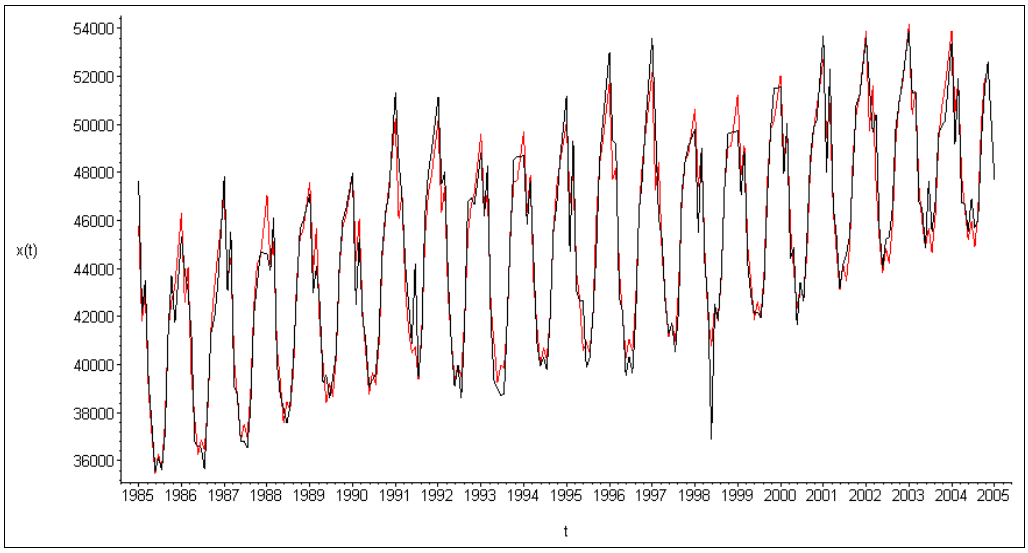


Abb. 31: Einschritt Prognosen des AR(1)-Modells (rot) nach Zurücktransformation der Saison- und Trendbereinigung für die Originalreihe StromBRD (schwarz).

Wegen des hohen Rechenaufwands bei der Bestimmung der Koeffizienten über die MLR wird üblicherweise der rekursive Levinson-Durbin-Algorithmus[67] benutzt. Das Anpassen eines AR-Modells sollte jedoch erst nach einer in Betracht gezogenen Differenzierung (siehe Absch. 3.4.5 S.42) durchgeführt werden. Zur Notwendigkeit einer Differenzierung gibt auch eine Einheitswurzel (engl. Unit-root) Hinweise. Ist in einem AR(p)-Modell die Summe der Koeffizienten annähernd 1, so haben diese eine Einheitswurzel. (Die Bezeichnung Einheitswurzel bezieht sich auf die n Lösungen einer komplexen Gleichung der Art $c^n = 1$). Ist dies der Fall, so simuliert das AR-Modell eine Differenzierung. Es empfiehlt sich dann die Ordnung der Differenzierung um eins zu erhöhen, während die Ordnung des AR-Modells um eins verkleinert werden sollte.[68]

Mit Hilfe der MLR können auch Kombinationen von horizontalen und vertikalen Prognosemodellen erstellt werden. Dabei können verschiedene Lags exogener und endogener Variablen miteinander benutzt werden.

3.5.3 Prognose mit gleitenden Durchschnitten

Eine einfache Prognose besteht in der Annahme eines konstanten Modells[69], bei dem sich die Werte der Zeitreihe bis auf einen kleinen zufälligen Anteil nicht ändern. Mit a als Konstante und ε_t als zufälliger Fehler am Zeitpunkt t (mit dem Mittelwert 0 für alle Fehler) gilt dann:

$$f_t = a + \varepsilon_t$$

Eine Annäherung für eine Prognose eines solchen Modells ist dann der Mittelwert der Reihe[70]:

$$a = \overline{x} \Rightarrow f_t = \overline{x} + \varepsilon_t$$

Eine erste Verbesserung dieses Modells ist die Annahme eines konstanten Modells zumindest für einen lokalen Bereich. So kann a über den lokalen

[67] Vergl. [Pollock 96] S. 537 - 538.
[68] Vergl. [Nau 03] Absch. „Identifying the numbers of AR or MA terms“.
[69] Vergl. [Brown 63] S. 97.
[70] Vergl. [Brown 63] S. 98.

Mittelwert der letzten m-Werte, also dem gleitenden Durchschnitt angenähert werden:

$$a = \frac{x_{t-1} + \ldots + x_{t-m}}{m} \Rightarrow f_t = \frac{x_{t-1} + \ldots + x_{t-m}}{m} + \varepsilon_t$$

Da der gleitende Durchschnitt eigentlich eine Schätzung für den Wert in der Mitte des Lags gibt, erscheint die Schätzung etwas Zeit verzögert. Im Prinzip werden die geglätteten Werte lediglich um die hälfte des Lags nach rechts verschoben, wodurch bei größeren Laggrößen eine stärke Verschiebung zu erkennen ist als bei kleinen Laggrößen (siehe Abb. 32).

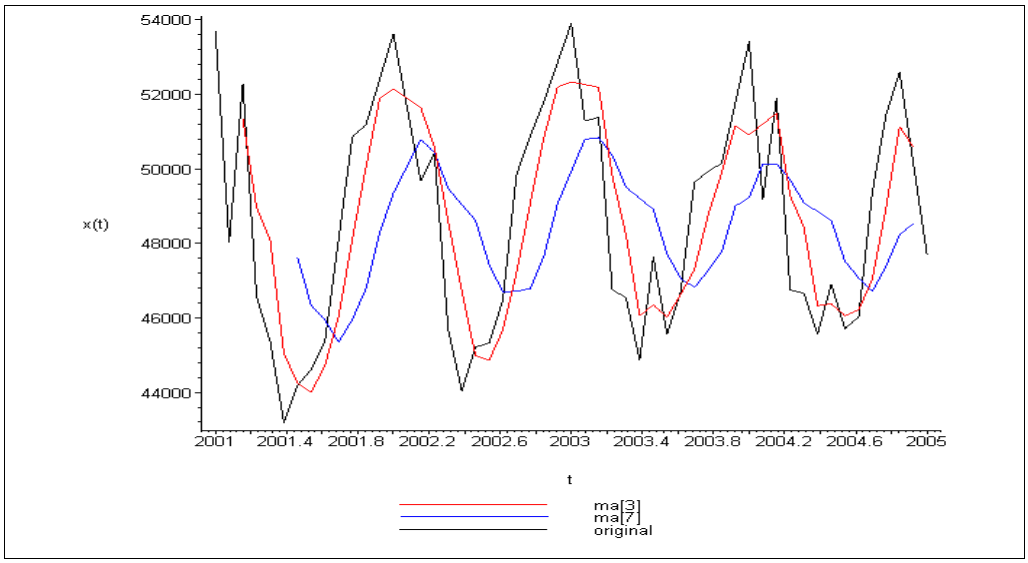


Abb. 32: Einschritt Prognosen durch gleitende Durchschnitte mit Laggrößen 3 (rot) und 7 (blau) der Originalreihe StromBRD (schwarz).

3.5.4 Exponentielle Glättung (exponential smoothing)

In einem weiteren Verbesserungsschritt wird die Prognose mit Hilfe des gleitenden Durchschnitts auf WMA erweitert, wobei die Koeffizienten der vergangenen Werte exponentiell abfallend Gewichtet werden (engl. exponential smoothing), welches auch exponentiell gewichteter gleitender Durchschnitt (engl. exponential weighted moving average, kurz ***EWMA***) genannt wird.

Der nächste Wert f_{t+1} wird anhand des letzten geschätzten und des aktuellsten Wertes mit der folgenden rekursiven Formel geschätzt[71]:

$$f_{t+1} = a \cdot x_t + (1-a) \cdot f_t$$

Mit

f_{t+1} = abzuschätzender Wert für Zeitpunkt $t+1$

f_t = bereits geschätzter Wert für Zeitpunkt t

x_t = Beobachtung zum Zeitpunkt t

a = Gewicht mit $0 \le a \le 1$

[71] Vergl. [Brown 63] S. 101.

Der letzte geschätzte Wert f_t spiegelt dabei alle Werte der Vergangenheit wieder. Die letzte Beobachtung erhält das Gewicht a, während die weiter zurückliegenden Werte (Schätzung für t) mit dem Gewicht $(1-a)$ einfließen. Entsprechend erhält bei $a=1$ die aktuellste Beobachtung das volle Gewicht, bei $a=0$ hingegen erhält die letzte Schätzung das volle Gewicht.[72] Äquivalent erhalten bei einem kleinen Wert ($a<0.5$) ältere Beobachtungen mehr Gewicht, bei einem großen Wert ($a>0.5$) der aktuellste Wert mehr Gewicht.
Entwickelt man die Schätzung für f_t rekursiv zurück, so erhält man:

$$
\begin{aligned}
& f_{t+1} = a \cdot x_t + (1-a) \cdot f_t && \mid \text{Ersetzen von } f_t \\
& = a \cdot x_t + a \cdot (1-a) \cdot x_{t-1} + (1-a)^2 \cdot f_{t-1} && \mid \text{Ersetzen von } f_{t-1} \\
& = a \cdot x_t + a \cdot (1-a) \cdot x_{t-1} + a \cdot (1-a)^2 \cdot x_{t-2} + (1-a)^3 \cdot f_{t-2} && \mid \ldots
\end{aligned}
$$

Für den Startwert kann $f_1 = x_1$ benutzt werden. Alternativ können auch die ersten Werte zu einem Mittelwert zusammen gerechnet und als Startwert benutzt werden[73]. In der praktischen Anwendung ist das Verfahren recht schnell und benötigt wenig Speicherplatz, da lediglich der letzte geschätzte Wert f_t benötigt wird. Eine äquivalente Darstellung des EWMA ist $f_{t+1} = a \cdot x_t + a \cdot e_{t-1}$, mit $e_{t-1} = x_{t-1} - f_{t-1}$ als der Fehler der letzten Schätzung[74]. Diese Form wird in den Darstellungen des ARIMA-Modells benutzt, der aus einem AR-Term, der Differenzierung (I) und einem EWMA-Term (MA) zusammengesetzt wird.
Für die Trend und Saison bereinigte Reihe StromBRD wurde mit Hilfe einer Intervallschachtelung ein geeigneter Wert für das Gewicht ermittelt. Die Abb. 33 zeigt die bereinigte Reihe sowie das angepaßte EWMA. Auch das für auf Durchschnitte basierenden Verfahren typische "hinterher hinken" bezüglich der eigentlichen Reihe ist hier erkennbar, aber nicht so extrem wie bei den einfachen MA. Die Residuen (Abb. 34 links) sowie die ACF (Abb. 34 rechts) zeigen, dass das Verfahren Korrelationen heraus filtert.

[72] Vergl. [Leiner 82] Kapitel 8
[73] Vergl. [Brown 63] S. 102.
[74] Vergl. [Nau 03] Absch. „Averaging and Exponential Smoothing Models“.

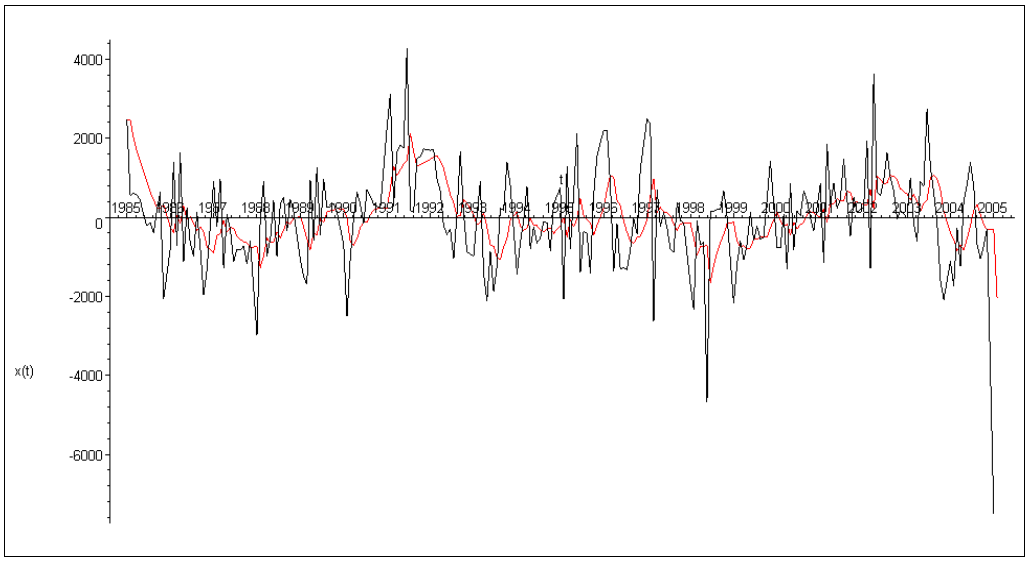


Abb. 33: Einschritt Prognosen des EWMA-Modells (rot) für die Saison und Trend bereinigte Reihe StromBRD (schwarz).

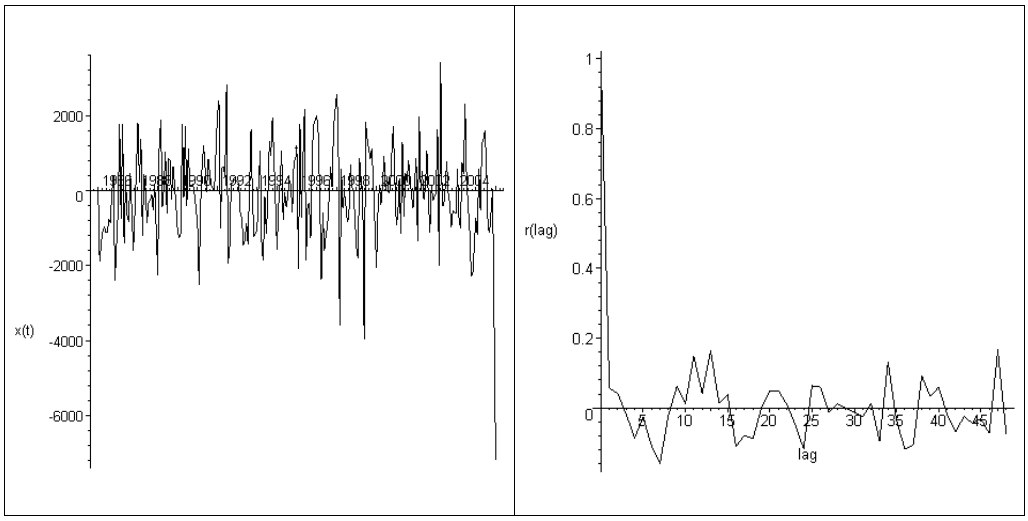


Abb. 34: Die Residuen der EWMA-Prognose (links) sowie die ACF der Residuen (rechts) für die Saison und Trend bereinigte Reihe StromBRD.

3.5.5 ARMA

Es liegt auf der Hand verschiedene Prognosemodelle zu kombinieren. Eine solche Kombination ist das univariate ARMA-Modell (AutoRegressive Moving Average), in dem das AR-Modell mit dem gleitenden Durchschnitt, speziell dem EWMA kombiniert wird. Hierzu werden beide Modelle hintereinander angewendet. Abb. 35 zeigt das EWMA-Modell, welches auf die Residuen des AR(1)-Modells angepaßt wurde. Die Residuen dieses kombinierten Modells (Abb. 36 links) haben kaum noch Korrelationen und ähneln der ACF eines Zufallsprozesses. Dennoch kann an dem SSE des kombinierten Modells im Vergleich zum nur AR-Modell erkannt werden, dass der zusätzliche EWMA Teil eher zu einer Verschlechterung des Modells geführt hat.

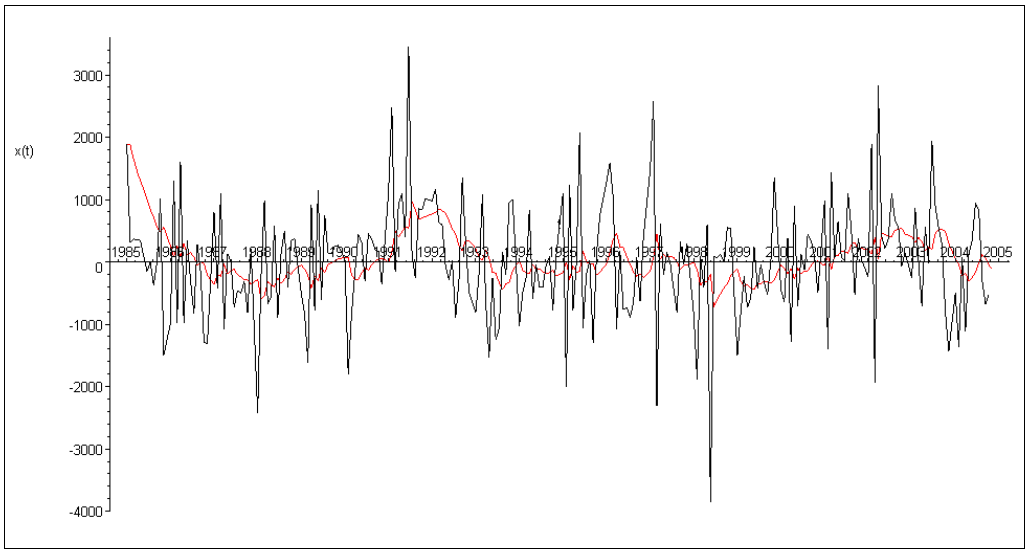


Abb. 35: Einschritt Prognosen des EWMA-Modells (rot) für die Residuen des AR(1)-Modells (schwarz) für die Reihe StromBRD.

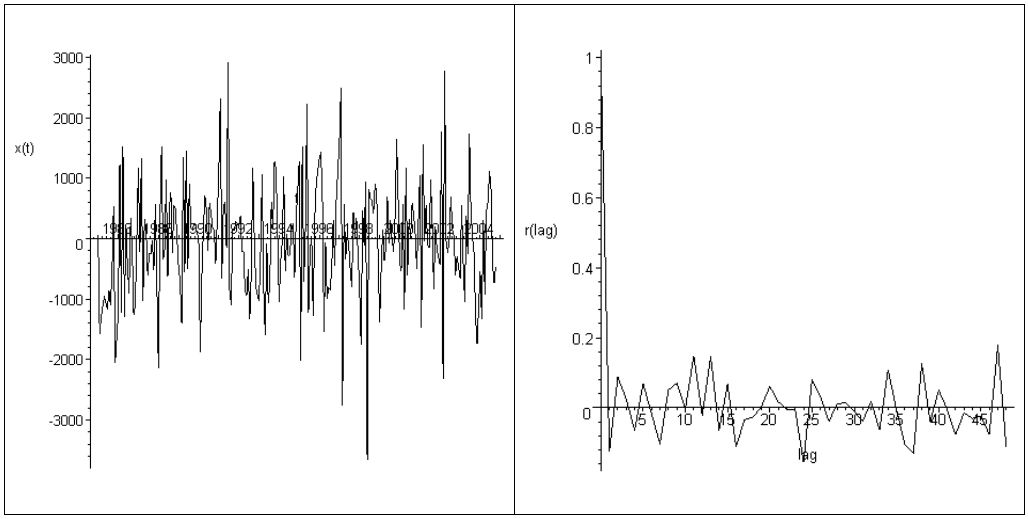


Abb. 36: Die Residuen der EWMA-Prognose (links) sowie die ACF der Residuen (rechts) für die Residuen des AR(1)-Modells für die Reihe StromBRD.

Wird das ARMA-Modell mit differenzierten Daten durchgeführt, so erhält man das ARIMA-Modell. Das „I“ steht hierbei für "Integrated" und bezeichnet die Differenzierung.

4 Künstliche Neuronale Netze

4.1 Grundlagen der künstlichen neuronalen Netze

„Die Erforschung künstlicher neuronaler Netze (kurz ***KNN***, im engl. artifical neural net, kurz ANN) begann bereits um 1940 und war durch das Interesse an den neuro-physiologischen Grundlagen des menschlichen Gehirns motiviert. Man wusste, dass das Gehirn aus Nervenzellen – den Neuronen – besteht, die untereinander verbunden sind und sich gegenseitig über elektrische Signale beeinflussen. Ein Neuron leitet seine Signale über sein Axon weiter und empfängt Signale von anderen Neuronen über die Kopplungsstellen zwischen deren Axonen und seinen Dendriten. Diese Kopplungsstellen werden Synapsen oder synaptische Spalte genannt.“[75]
Aus dem biologischen Vorbild des Neurons wurde von McCulloch und Pitts 1943 ein mathematisches Modell, das McCulloch-Pitts-Neuron (kurz ***MCPN***), entwickelt.

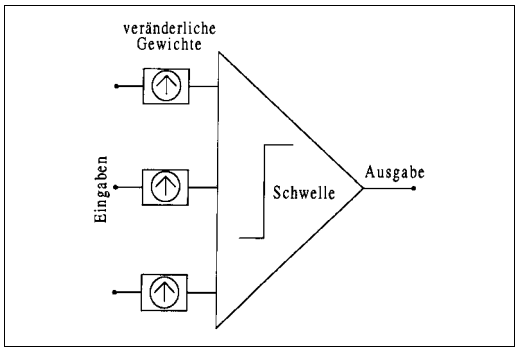


Abb. 37: Schematische Darstellung eines McCulloch-Pitts-Neurons.[76]

Die über die Dendriten eingehenden Signale werden in einem Neuron akkumuliert und falls diese einen Reizschwellwert übersteigen (das Neuron feuert) an die Synapsen weitergeleitet. Entscheidend hierbei ist vor allem die Synapse, welche über die Stärke des Signales entscheidet. Die Dendriten eines Neurons sind natürlich mit den Synapsen der vorgeschalteten Neuronen verbunden. Beim MCPN werden eingehende Signale als Zahlenwerte interpretiert. Diese werden durch veränderliche Gewichte, welche die Leitfähigkeit der Synapsen repräsentieren, gewichtet. Eine Transferfunktion repräsentiert die Schwellwerteigenschaft des biologischen Neurons, welche nach der Akkumulation der eingehenden Werte eine Ausgabe erzeugt oder das Signal hemmt. Ein einfaches MCPN mit einer linearen Transferfunktion $f(x) = x$ und drei eingehenden Eingaben $x_1...x_3$ und den entsprechenden Gewichten $w_1...w_3$ hat als Ausgabe somit die Funktion[77] $f(x) = x_1*w_1+x_2*w_2+x_3*w_3$. Die biologischen Vorbilder haben jedoch meist keine lineare Schwellwertfunktion. Daher werden für die Transferfunktion üblicherweise sigmoide Funktionen, wie z.B. der tangens hyperbolicus (kurz ***Tanh***) benutzt. Mit Tanh als Transferfunktion ergibt sich für das oben genannte Beispiel des MCPN die Ausgabefunktion $\text{Tanh}(x_1*w_1+x_2*w_2+x_3*w_3)$.
Im biologischen Vorbild basiert das eigentliche Lernen auf der Veränderung der Leitfähigkeit der Synapsen und wird im Modell durch die veränderlichen Gewichte dargestellt. Um 1949 postulierte Hebb, dass ein Lernvorgang die

[75] Aus [NKK 96] S. 11.
[76] Aus [NKK 96] S. 12.
[77] Vergl. [Rojas 93] S. 24.

Verbindung zwischen zwei Neuronen verstärkt, wenn die postsynaptische und präsynaptische Zelle gleichzeitig aktiv sind. Aufbauend auf der Hebb'schen Lernregel wurden erste künstliche neuronale Netze erstellt. Diese konnten jedoch zuerst nur auf einschichtige Netze (***Perceptron***) angewendet werden. Den Durchbruch erzielte später das auf der Deltaregel von Widrwos und Hoffs aufbauende Backpropagationverfahren von Rumelhart, Hinton und Willliams um 1986.

4.2 Lernverfahren Backpropagation

Das heute noch (in verschiedenen Variationen) übliche Backpropagationverfahren ermöglicht es mehrschichtige KNN zu trainieren. Dazu wird für jedes Ausgangsmuster und jedes Ausgabeneuron ein Fehlermaß berechnet (Differenz zwischen Ist- und Sollwert), das durch das Netz „Rückwärts", also von den Ausgabeneuronen zu den Eingabeneuronen hin, zurückgeleitet wird (***Backpropagation***). Das so weitergeleitete Fehlermaß hilft dann bei der Entscheidung, ob ein Gewicht verstärkt (vergrößert) oder gehemmt (verkleinert) werden soll. Anschaulich versucht das Lernverfahren auf einer Fehlerhyperlandschaft (Unterschied zwischen Ist-Zielgrößen und Soll-Zielgrößen) ein Minimum zu lokalisieren und damit den Fehler zu minimieren.

Hierzu werden die Gradienten der Fehlerhyperlandschaft in Abhängigkeit zu jedem einzelnen Gewicht berechnet und entsprechend entschieden, ob die Veränderung eines Gewichtes einem Ab- oder Aufstieg in der Landschaft entspricht. Veränderungen, die zu einem Abstieg führen, werden bevorzugt.

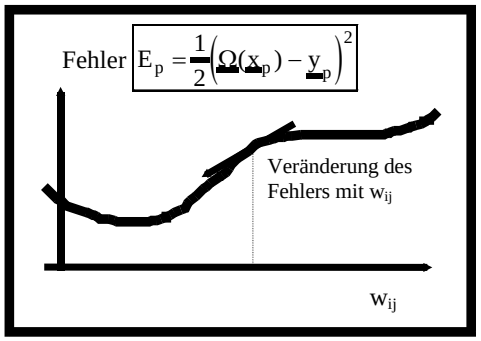


Abb. 38: Abstieg an einer Fehlerfunktion (mittlerer quadratischer Fehler).[78]

Üblicherweise wird das Backpropagationverfahren durch die Formel in Abb. 41 definiert. Dabei stehen die Bezeichnungen ∂E_p für den Gradienten des Fehlersignales, ∂w_{ij} für den Gradienten zwischen den Neuronen i und j,

α für die Lernschrittweite (der Schrittweite beim Abstiegsverfahren), w_{ij}^{alt} für das Kantengewicht zwischen den Neuronen i und j vor dem Lernschritt und w_{ij}^{neu} für das Kantengewicht zwischen Neuron i und j nach dem Lernschritt.

$$w_{ij}^{neu} = w_{ij}^{alt} - \alpha \cdot \frac{\partial E_p}{\partial w_{ij}}$$

Abb. 39: Berechnungsformel für die Gewichtsaktualisierung beim Standard Backpropagation.[79]

Das Standard Backpropagationverfahren wurde bereits durch mehrere Erweiterungen zur schnelleren und robusteren Konvergenz zu einem möglichen

[78] Aus [Westenberger 04] S. 2.
[79] Aus [Westenberger 04] S. 2.

Minimum verbessert. Nennenswert ist hier das 1992 von Riedmiller und Braun[80] vorgestellte Lernverfahren Resilient Backpropagation, kurz auch ***Rprop*** genannt. Das RProp-Verfahren vereint das SuperSAB und das QuickProp Verfahren (verallgemeinertes Newtonsches Verfahren zur Minimasuche) und benutzt nicht die Gradienten selber zur Berechnung eines Änderungswerts für das entsprechende Gewicht, sondern benutzt den Gradienten lediglich zur Bestimmung, ob die Änderung einen Auf- oder Abstieg zur Folge hatte. Die Änderung jedes Gewichtes wird durch einen eigenen Wert, einer Art individueller Lernschrittweite, definiert, die für jedes Gewicht separat verwaltet wird. Dadurch können die Kantengewichtsänderungen individueller verlaufen.[81] Man spricht auch von einem „lokal adaptiven Lernverfahren", da sich das Lernverfahren lokalen Gegebenheiten der Fehlerkurve besser anpasst.

4.3 KNN als multiple nichtlineare Regressoren

Die MLR kombiniert die Koeffizienten der Anpassungsfunktion linear miteinander (einfache Summe der Koeffizienten), um eine Näherungsgleichung zu erhalten. Die allgemeine Form einer MLR hat daher die Form:

$$y=w_0+w_1*x_1+w_2*x_2+w_3*x_3$$

Ein KNN mit einer Eingabeschicht und einer Ausgabeschicht mit linearer Schwellwertfunktion besitzt die gleiche Ausgabefunktion. In diesem Fall entspricht das Trainieren des Netzes genau der MLR.[82] Wird eine nicht-lineare Schwellwertfunktion für das zwei schichtige Netz gewählt, so entspricht dies der MLR auf die durch die Schwellwertfunktion transformierten Zieldaten. Im Falle der Schwellwertfunktion Tanh entspricht die Ausgabefunktion des Netzes

$$y = Tanh\ (w_0+w_1*x_1+w_2*x_2+w_3*x_3)$$

Werden zusätzlich verborgene Schichten mit Neuronen benutzt, so erweitert sich die Ausgabefunktion zu einer nichtlinearen Funktion, wie z.B.

$$\begin{aligned} y=&Tanh(\\ &w_{14}*Tanh(w_{01}+w_{11}*x_1+w_{21}*x_2+w_{31}*x_3)+\\ &w_{24}*Tanh(w_{02}+w_{12}*x_1+w_{22}*x_2+w_{32}*x_3)+\\ &w_{34}*Tanh(w_{03}+w_{13}*x_1+w_{23}*x_2+w_{33}*x_3)) \end{aligned}$$

In diesem Fall setzen KNN die in ihnen verwendeten Koeffizienten, die Gewichte (w_{ij}), als eine nichtlineare Funktion zusammen**.** Die KNN werden im Zusammenhang mit der Regression daher auch als ein nichtlineares Regressionsmodell interpretiert, da ihre Koeffizienten nicht einfach linear aufaddiert, sondern durch die Transferfunktion nichtlinear kombiniert werden. Das Lernverfahren kann hierbei als eine MKQ für nichtlineare Funktionen angesehen werden.
Solche Funktionsnetze können mit einer differenzierbaren Schwellwertfunktion beliebige stetige Funktionen durch eine endliche Anzahl von Elementen (Neuronen) in der verborgenen Schicht approximieren.[83]
Eine weitere Eigenschaft von mehrschichtigen KNN ist ihre Fähigkeit eine PCA durchzuführen. Bei KNN mit weniger Neuronen in der ersten verborgenen Schicht als in der Eingabeschicht muss das Netz zwischen diesen beiden Schichten eine geeignete Kodierung finden, um die Eingabedaten auf die Größe

[80] Siehe [Riedmiller 92].
[81] Vergl. [Riedmiller 94].
[82] Vergl. [Rojas 93] S. 180-181.
[83] Vergl. [Kolgomorov 57] und [IrieMijake 88] nach [Rojas 93] S. 204-207.

der ersten verborgenen Schicht zu komprimieren.[84] Rojas bezeichnet in diesem Zusammenhang die Funktion zwischen Eingabe- und erster verborgener Schicht als Definition von Attraktionsbecken. Zwischen erster verborgener Schicht und Ausgabeschicht wird anschließend eine MLR durchgeführt. Dies wird im Gegensatz zu einer Hintereinanderausführung der klassischen PCA und MLR durch das Netz gleichzeitig gelöst.[85]
Die nichtlineare Eigenschaft von KNN erlaubt es ihnen, nicht nur die Ausgangsgrößen mit den Zielgrößen in Zusammenhang zu bringen, sondern darüber hinaus die einzelnen Ausgangsgrößen und Zielgrößen sowie die einzelnen Koeffizienten untereinander in Bezug zu setzen und so auch nichtlineare Korrelationen zwischen diesen zu modellieren.

4.4 Modellparameter für KNN

Für den Einsatz von KNN sind viele Parameter zu bestimmen. Dazu gehören: Anzahl der Eingabe-, Hiddenlayer- und Ausgabeneuronen, Vernetzungsstruktur, Lernverfahren, Abbruchkriterium sowie die Kodierung und Vorverarbeitung der Daten. Im folgenden sollen diese Modellparameter diskutiert werden.

Netzkomplexität - Generalisierung versus Überanpassung

KNN können, wie auch die polynomiale Regression, zu sehr an die vorgegebenen Daten angepasst werden, wobei man von ***Overfitting*** (Überanpassung) spricht. In diesem Fall speichert das Netz die Lernmuster komplett in seinen Gewichten, es lernt die Daten „auswendig". Es sei daran erinnert, dass bei gegebenen Daten ein genügend komplexes Netz alle Daten speichern kann, auch statistisches Rauschen[86]. Dadurch kann das Netz zwar die Lerndaten sehr gut wiedergeben, liefert aber bei out-of-sample Daten eine schlechtere Performanz. Dem gegenüber steht die Generalisierungsfähigkeit des Netzes, bei der das Netz lediglich wiederkehrende Regelmäßigkeiten speichert und damit auch bei unbekannten Daten eine gute Performanz liefert. Beide Eigenschaften – Generalisierung und Überanpassung stehen meist im Gegensatz zueinander. Daher ist es Ziel des Trainings, nicht nur eine gute Performanz auf den Lerndaten zu erzielen, sondern die Generalisierungsfähigkeit des Netzes für unbekannte Daten aus der Testmenge optimal auszunutzen. Hierzu werden die Lerndaten in zwei Mengen unterteilt, (üblicherweise werden 10-30% der Daten für die Testmenge benutzt) von denen lediglich die Trainingsmenge dem Netz während des Trainings präsentiert wird. Der zweite Teil der Lerndaten wird als Testmenge vorenthalten und nach jedem Lernschritt zur Überprüfung durch das Netz evaluiert. Somit wird die optimale Generalisierungsfähigkeit des Netzes an dem Punkt ermittelt, bei dem der Fehler des Netzes für beide Mengen, vor allem aber für die Testmenge, ein Minimum hat.
Ein sinnvoller Anfang für die Bestimmung der Netzkomplexität ist daher mit einem KNN ohne verborgene Schicht zu beginnen. Dies entspricht dann einer einfachen MLR. Schritt für Schritt kann dann die Anzahl von Neuronen in der verborgenen Schicht erhöht werden. Dabei ist jeweils das Minimum des Fehlers auf der Testmenge zu beobachten. Diejenige Netzkomplexität, bei welcher das Minimum des Fehlers auf der Testmenge am kleinsten ist, ist die optimal

[84] Vergl. [Callan 99] S. 114, S. 160. sowie [Rojas 93] S. 194-195.
[85] Vergl. [Rojas 93] S. 196.
[86] Vergl. [Rojas 93] S. 175 ff..

generalisierende Netzstruktur. Ein Beispiel hierfür findet sich im Kapitel hybride Modelle.
Eine weitere Faustregel gibt ebenfalls einen Anhaltspunkt zur Komplexität des Netzes bei einer gegebenen Anzahl an Lerndatensätzen: Für jedes Gewicht des Netzes sollten mindestens 10 Lerndatensätze vorhanden sein.[87] Diese Faustregel ist eine Evaluierung der Formel[88]

$$Trainingsmuster > \frac{Netzgewichte}{Fehlerproportion}$$

für die Fehlerproportion von 10%. Die Praxis zeigt jedoch, dass durch die Bestimmung der Generalisierungsfähigkeit des Netzes genauere und bessere Ergebnisse erzielt werden. Diese Formel ist also lediglich ein Anhaltspunkt für eine Startkomplexität, welche vor allem bei großen Datenmengen und großen Netzkomplexitäten hilfreich ist.
In einigen Fällen kann es sein, dass das Netz ohne verborgene Schicht die beste Generalisierungsfähigkeit zeigt. Dies spricht dann dafür, dass das zugrunde liegende Problem am besten durch ein lineares Modell, wie die MLR zu lösen ist. Die Daten enthalten dann außer einer erklärbaren linearen Komponente lediglich Rauschen, welches auch durch ein KNN nicht zu erklären ist.
Ist ein Netz mit einer verborgenen Schicht die optimale Lösung, so kann anhand der Größe der verborgenen Schicht eine Vorverarbeitung durch die PCA in Betracht gezogen werden. Wie bereits genannt, führt das Netz zwischen Eingabeschicht und verborgener Schicht eine Art PCA durch. In einem Beispiel im Kapitel 6 wird eine 21-dimensionale Eingabe am besten durch ein KNN mit drei Neuronen in der verborgenen Schicht verarbeitet. Dies deutet darauf hin, dass hier durch eine PCA die Eingabedaten auf 3 Dimensionen reduziert werden. Daher kann es hier sinnvoll sein, die Eingabedaten zuerst durch eine PCA zu reduzieren, um diese Arbeit dem KNN abzunehmen.

Anzahl der Eingabeneuronen

Damit das KNN Zusammenhänge im zeitlichen Verlauf der Reihe ermitteln kann, ist es notwendig dem Netz einen Ausschnitt der Daten mit einem zeitlichen Kontext zu präsentieren. Hierzu gibt die ACF einen ersten Anhaltspunkt: Die Kontextgröße sollte zumindest größer als die korrelierten Lags sein. Dennoch kann es sinnvoll sein, auch größere Kontexte zu benutzen. Hierzu ist es sinnvoll, dem Netz eine übergroße Lagstruktur anzubieten und zu beobachten, welche Eingabeneuronen kleine Gewichte zur Neuronen in der verborgenen Schicht haben. Diese sind dann redundant. Einen weiteren Hinweis geben die Eigenwerte der Eingabedaten. Solche Eingabedaten mit einem hohen Eigenwert sind für die Prognose am relevantesten.

Anzahl der Neuronen in der verborgenen Schicht

Eine Vielzahl von Faustregeln zur Ermittlung der Größe der verborgenen Schicht wurden bereits aufgestellt[89], die sich im Bereich von 0.5 mal Anzahl Eingabeneuronen bis 3 mal Anzahl Eingabeneuronen bewegen[90]. Letztendlich bleibt nur das Ausprobieren. Hierzu kann, wie bereits erwähnt, ohne verborgene

[87] Vergl. [Thiesing 98] S. 91.
[88] Vergl. [BaumHassler 89] nach [Callan 99] S. 58.
[89] Vergl. [Kaastra 94] liefern eine Auflistung in Absch. 3.5.2.
[90] Vergl. [Katz 92] nach [Kaastra 94].

Schicht begonnen, und bei Bedarf die Anzahl der Neuronen in der verborgenen Schicht erhöht werden. Die beste Generalisierungsfähigkeit gibt dann an, welche Anzahl am besten geeignet ist.

Anzahl Ausgabeneuronen

Die Anzahl der Ausgabeneuronen wird auf ein einzelnes Neuron festgelegt. Es mag zwar naheliegend sein, für einer Langzeitprognose mehrere Werte (x_{t+1}, x_{t+2}, etc.) zugleich zu prognostizieren, jedoch ist dabei zu bedenken, dass der Ausgabefehler der weiter in der Zukunft liegenden Werte stets größer sein wird als die des nächsten Wertes. Dadurch würde das Lernverfahren mehr Aufwand in den größeren Fehler investieren und dadurch das Gesamtergebnis verschlechtern[91].

Vernetzungsstruktur

Bei der Vernetzungsstruktur stehen verschiedene Modelle zur Verfügung. Grundsätzlich ist zwischen einer einfachen Vollvernetzung, bei der alle Neuronen einer Schicht mit allen Neuronen der folgenden Schicht verbunden werden, und der Vollvernetzung mit Shortcuts, bei der zusätzlich alle Neuronen einer Schicht mit den Neuronen aller nachfolgenden Schichten verbunden werden, zu wählen. Die Vernetzung mit Shortcuts besitzt mehr Gewichte, welches sich in einem langsameren Training auswirkt. Dafür benötigt das Shortcut-Modell weniger Neuronen in der verborgenen Schicht. Zusätzlich können die direkten Verbindungen der Eingabeschicht zur Ausgabeschicht das Lernverhalten verbessern.

Lernverfahren

Als Lernverfahren wird das schnell konvergierende Rprop-Verfahren benutzt. Es ist weit aus schneller als das Standard-Backpropagationverfahren und vereint auch die Vorzüge der Lernverfahren QuickProp sowie SuperSAB. Es ist ein addaptives Verfahren und paßt seine Parameter automatisch an.

Kodierung und Vorverarbeitung der Daten

Durch die Transferfunktion ist der sensitive Wertebereich für das Netz vorgegeben. Während der Wertebereich der Eingabedaten unbegrenzt ist, ist die Asugabe des Netzes bei der Transferfunktion Tanh auf den Wertebereich [-1, +1] festgelegt.
Daher müssen die Ausgabemuster auf diesen Bereich skaliert werden. Genauer sollte die Skalierung im Bereich [-0.9, +0.9] benutzt werden, da die Tanh Transferfunktion im Bereich der Ränder weniger sensitiv ist.
Um Ausreißern die Gewichtung zu nehmen ist es ebenfalls notwendig, die Eingabedaten von diesen zu befreien oder zumindest eine Skalierung zu wählen, durch welche die Ausreißer an Gewicht verlieren.
Um die vom Netz produzierten Fehler weiter zu verkleinern kann die Prognose nicht direkt auf den nächsten Wert selbst, sondern lediglich die Differenz zum Vorgängerwert prognostiziert werden. So kann eine Abweichung von 10% auf dem Originalwert auf eine Abweichung von 10% (oder mehr) auf die Differenz

[91] Vergl. [Kaastra 94] Absch. 3.5.3.

reduziert werden. Dies sollte jedoch nur dann gemacht werden, wenn durch die einfache Differenzierung der Zieldaten keine Überdifferenzierung auftritt.
Thiesing beschreibt, dass die Eingabe der Rohdaten bei Reihen mit hohem Trendanteil zu schlechter Performanz führt, weswegen zumindest eine Trendbereinigung als Vorverarbeitung der Eingabedaten anzuwenden ist[92]. Wird die Differenzierung angewendet, erreicht das KNN Modell bereits eine Vorstufe eines hybriden Modells. Man kann hierbei entsprechend dem integrierten ARMA von einem integrierten KNN-Modell sprechen.

[92] Vergl. [Thiesing 98] S. 169-170.

5 Qualitäts- und Fehlermaße zum Vergleich verschiedener Prognosen

"Qualität ist das beste Rezept."
Dr. Oetker[93]

Um die Qualität einer Prognose zu bewerten, ist es notwendig, einen Teil der Werte bei der Modellbildung außen vor zu lassen. Üblicherweise werden etwa die letzten 10%-30% der Werte bei der Modellbildung nicht verwendet (***Validierungsmenge***). Im Fall der Zeitreihen empfiehlt sich natürlich die daten der letzten Saison als Validierungsmenge zu benutzen. Diese Menge ist dem Modell nicht bekannt (engl. ***out of sample***) und dient nach Bildung des Modells dazu, die erstellte Prognose auf ihre Güte zu testen. Die folgenden Fehlermaße sind daher auf die Validierungsmenge anzuwenden.

Fehlermaße zum Vergleich von Prognosen und der Originalreihe werden üblicherweise aus der Differenz zwischen den Prognosewerten (Ist) und den Originalwerten (Soll) gebildet (Fehler). Das einfachste Fehlermaß ist dementsprechend die Summe der Fehlerbeträge:

$$\boldsymbol{ABSE} = \sum_{t=1}^{N} \left[|x_t - f_t| \right]$$

Die Bildung des Betrages ist wichtig, da sich sonst positive und negative Fehler gegenseitig aufheben könnten. Entsprechend der MKQ kann statt der Betragsbildung auch das Quadrat der einzelnen Fehler aufsummiert werden, woraus sich die Summe der quadrierten Fehler (engl. sum of squared errors) ergibt:

$$\boldsymbol{SSE} = \sum_{t=1}^{N} \left[(x_t - f_t)^2 \right]$$

Die beiden oberen Fehlersummen werden üblicherweise noch durch die Anzahl der Reihenwerte normiert, damit verschieden mächtige Mengen miteinander verglichen werden können. Für den ABSE wird damit der mittlere absolute Fehler (engl. mean absolute error, mean absolute deviation) gebildet:

$$\boldsymbol{MAD} = \frac{ABSE}{N} = \frac{1}{N} \cdot \sum_{t=1}^{N} \left[|x_t - f_t| \right]$$

Entsprechend wird mit dem SSE der mittlere quadratische Fehler (engl. mean squared error) gebildet:

$$\boldsymbol{MSE} = \frac{SSE}{N} = \frac{1}{N} \sum_{t=1}^{N} \left[(x_t - f_t)^2 \right]$$

Aus dem MSE wird meistens noch die Wurzel gebildet, um die Quadrierung der Fehlersummen teilweise aufzuheben. Die Quadratwurzel des mittleren quadrierten Fehlers (engl. root mean squared error) hat daher dieselbe Einheit wie die Daten und ist wie folgt definiert:

$$\boldsymbol{RMSE} = \sqrt{\frac{SSE}{N}} = \sqrt{\frac{1}{N} \sum_{t=1}^{N} \left[(x_t - f_t)^2 \right]}$$

[93] Werbespruch der Firma Dr. Oetker, 1980.

Dabei gehen jedoch individuelle Abweichungswerte unter. Daher ist auch die maximale Abweichung der Prognose von der Originalreihe interessant, da neben einem durchschnittlichen Fehlerverhalten auch die extremen Fehler sehr wichtig sind:

$$\boldsymbol{MaxE} = Max(|x_t - f_t|), \text{ t=1...N}$$

Ein weiteres weit verbreitetes Fehlermaß ist der mittlere absolute prozentuale Fehler (engl. mean absolute percent error), welcher die Differenz zwischen Prognose- und Originalwerten prozentual ausdrückt:

$$\boldsymbol{MAPE} = \sum_{t=1}^{N}\left[\left|\frac{x_t - f_t}{x_t}\right|\right] * \frac{100}{N}$$

Der Nachteil dieses Fehlermaßes ist jedoch, dass er voraussetzt, dass alle Werte der Reihe ungleich 0 sind.
Der Ungleichheitskoeffizient von Theil gibt nicht nur ein Fehlermaß, sondern vergleicht die Prognose direkt mit einer naiven Prognose. Ist U kleiner als 1, so ist die Prognose besser als eine naive Prognose. Je näher der Wert bei 0 liegt, desto besser ist die Prognose.

$$\boldsymbol{U} = \frac{\sqrt{\frac{1}{N}\sum_{t=1}^{N}\left[(x_t - f_t)^2\right]}}{\sqrt{\frac{1}{N}\sum_{1}^{N}\left[x_t^{\,2}\right]} + \sqrt{\frac{1}{N}\sum_{1}^{N}\left[f_t^{\,2}\right]}}$$

Ein sinnvoller Vergleich verschiedener Prognoseverfahren kann nur aufgrund mehrerer Fehlermaße erfolgen. Dabei sind alle Prognoseverfahren über derselben Reihe zu testen. Auch ist zu berücksichtigen, dass Fehlervergleiche nur auf Daten derselben Skala und Dimension sinnvoll sind.

6 Hybride Modelle

"All Models are Wrong, but some models are useful"
George .E.P. Box

KNN können bei entsprechender Komplexität beliebige Funktionen approximieren und damit auch die Arbeit linearer Modelle übernehmen. Da das Trainieren von KNN im Vergleich zu linearen Modellen recht aufwändig ist, liegt einem hybriden linearen und KNN Modell die Idee zugrunde, den linearen Anteil der Prognose durch schnellere lineare Modelle abzudecken, um die nichtlinearen Anteile durch das KNN zu modellieren. Erste in diesem Sinne bereits durchgeführte Arbeiten stammen von Thiesing und Zhang und behandeln primär univariate Prognosen.
Thiesing[94] benutzte den Differenzfilter als Vorverarbeitung eines KNN-Modells, um dessen Leistungsfähigkeit zu erhöhen. Dabei konnten die Trendkomponenten der Reihen ausgeschaltet werden, wodurch sich die Prognosefähigkeit des KNN erheblich erhöhte. In einigen Fällen schnitt das KNN sogar besser ab als das klassische ARIMA-Verfahren. Zhang[95] kombinierte das ARIMA-Modell mit KNN und erreichte signifikante Verbesserungen relativ zum reinen KNN oder ARIMA Ansatz. Zhang beschreibt ein allgemeines hybrides Modell mit kombiniertem linearen (L_t) und nichtlinearen (NL_t) Modell wie folgt[96]:

$$Y_t = L_t + NL_t + \varepsilon_t$$

Weiterhin beschreibt Zhang die Vorgehensweise wie folgt: Zuerst wird ein lineares Modell auf die vor verarbeiteten Werte der Reihe angewendet. Anschließend wird ein KNN angewendet, dass die Ausgaben des linearen Modells sowie die vor verarbeiteten Werte der Reihe selbst als Eingabe erhält. Als Ausgabemuster dienen die Residuen des linearen Modells, welche noch nichtlineare Zusammenhänge enthalten sollten. Zum Schluss werden die beiden Modelle, die Prognose des linearen Modells sowie die vom KNN prognostizierten Fehler kombiniert.

Für eine multivariate Prognose wird ein ähnliches Modell benutzt. Hierbei wird zuerst eine lineare Methode der multivariaten Prognose, in diesem Fall die MLR benutzt, um die Reihe zu prognostizieren. Anschließend wird versucht, den durch die MLR nicht erfassten Anteil durch KNN zu modellieren.

Die im folgenden durchgeführten Versuche wurden mit Maple und der mscNeuralNet-API in Java durchgeführt, welche ich im Verlauf meines Wahlpflichtprojekts erstellte.

[94] [Thiesing 98]
[95] [Zhang 99], [Zhang 04]
[96] Vergl. [Zhang 04] Kapitel 11.

6.1 MLR

Um eine geeignete Lagstruktur sowie geeignete Vorverarbeitungsschritte für die MLR zu finden, werden die endogene und die exogenen Variablen in eine Trainingsmenge und eine Validierungsmenge (die letzten beiden Jahre) unterteilt. Die Validierungsmenge dient dem out-of-sample Performanztest und wird beim Bestimmen der MLR-Koeffizienten nicht benutzt.
Um eine geeignete Lagstruktur bei verschiedenen Vorverarbeitungen zu finden, wurden 18 verschiedene Lagstrukturen festgelegt, die MLR mit diesen Lags durchgeführt und die Theil Koeffizienten für die Trainings- und Validierungsmenge bestimmt. Im folgenden wurden die Reihen StromBRD, StromFR und StromNL benutzt, um die Reihe StromBRD zu prognostizieren. Der Lag n bezeichnet dabei den Wert zum Zeitpunkt x_{t-n}.

Differenzierung für die MLR

Die Differenzierung in einfacher und saisonaler Form wurde bereits als Trendbereinigungsverfahren vorgestellt.

Saisonale Differenzierung

Da beide untersuchten Reihen eine starke Saisonkomponente besitzen, wird die Eignung die saisonalen Differenzierung erster Ordnung ∇_{12}^{1} untersucht.

Versuchsnr.	Lags	Theil U Trainingsmenge	Theil U Testmenge
1	1	0.001320554695	0.002169832609
2	1,2	0.001306451153	0.002037138101
3	1,2,3	0.001299617509	0.002075282929
4	1,2,3,4	0.001291349539	0.002159468468
5	1,2,3,4,5	0.001259022091	0.002042188977
6	1,2,3,4,5,6	0.001261318323	0.002002203652
7	1,2,3,4,5,6,7	0.001251910909	0.001957917157
8	1,2,3,4,5,6,7,8	0.001247403451	0.001963991658
9	1,2,3,4,5,6,7,8,9	0.001236411159	0.002001636130
10	1,2,3,4,5,6,7,8,9,10	0.001209114604	0.002007886407
11	1,2,3,4,5,6,7,8,9,10,11	0.001167766216	0.002125180351
12	1,2,3,4,5,6,7,8,9,10,11,12	0.001063120217	0.001956042064
13	1, 2, 3, 4, 12	0.001223451663	0.001969656552
14	1, 2, 3, 6, 12	0.001227179628	0.001902756201
15	1, 2, 3, 4, 5, 6, 12	0.001188336560	0.001843675204
16	1, 2, 3, 6, 10, 11, 12	0.001154213514	0.002058495500
17	1, 2, 3, 6, 11, 12	0.001182161617	0.002011789100

Versuchsnr.	Lags	Theil U Trainingsmenge	Theil U Testmenge
18	1, 2, 6, 11, 12	0.001187722976	0.001985215624

Tabelle 1: Fehlerwerte der MLR nach einfacher saisonaler Differenzierung der Daten bei verschiedenen Lagstrukturen für die Reihe StromBRD.

6.2 Vergleich MLR und KNN

Zum Vergleich wurde die für die MLR benutzte Datenmatrix als Grundlage für einen KNN Ansatz benutzt. Die Datenreihen wurden Mittelwert bereinigt und durch die Standardabweichung skaliert. Weiterhin wurde durch eine einfache Skalierung der Wertebereich auf [-0.9;+0.9] eingeschränkt.
Die trainierte Netzstruktur hat 21 Eingabeneuronen (entsprechend den Lags 1, 2, 3, 4, 5, 6, 12 von drei Reihen), eine verborgene Schicht und ein Ausgabeneuron mit einer Vollvernetzung ohne Shortcuts. Die letzten 48 Daten (die letzten vier Jahre) wurden als Testmenge beim Training außen vor gelassen und mit ihnen nach jedem Lernschritt ein Validierungsschritt durchgeführt, um die Generalisierungsfähigkeit des Netzes zu testen. Als Lernverfahren wurde das RProp-Verfahren eingesetzt.
Die getesteten Netze wurden auf ihre Konvergenzgeschwindigkeit, ihrer Genauigkeit sowie ihrer Fähigkeit zum Generalisieren verglichen.
Es wurden jeweils 20 Testdurchläufe gemacht und die kleinsten Fehlerwerte für die Testmenge sowie die durchschnittlichen Fehlerwerte der 20 Versuche auf der Testmenge bestimmt.
Das ermittelte KNN zeigte auf der Testmenge eine Verbesserung, war jedoch auf der Validierungsmenge schlechter. Die Größe der verborgenen Schicht gibt den wertvollen Hinweis, dass die Eingabedaten recht viel Redundanz enthalten und eigentlich mit Hilfe der PCA auf 7 oder 8 Werte reduziert werden könnten. Dies soll im nächsten Schritt durchgeführt werden.

Modell	Theil U Trainingsmenge	Theil U Validierungsmenge
SDIF MLR	0.0011883365600	0.001843675204
SDIF KNN 21x7x1	0.0009565832362	0.001923904365

Tabelle 2: Fehlerwerte der KNN und MLR Prognosen für die Reihe StromBRD.

Vergleich KNN nach PCA

Um die Redundanzen aus den Daten zu entfernen, werden als nächstes die vor verarbeiteten Daten mit Hilfe der PCA Hauptachsen transformiert und die Eigenvektoren mit geringen Eigenwerten entfernt. Als Schwellwert für die Entfernung von Spalten wird ein prozentualer Anteil der Eigenwerte an der Gesammtsumme der Eigenwerte von 6% festgelegt.
Durch die PCA wurden die 21-dimensionalen Eingabewerte auf eine Dimension von 7 reduziert. Um eine geeignete Netzstruktur für das KNN zu finden, wurden wieder jeweils 10 Durchläufe mit verschiedenen Netzstrukturen durchgeführt.

Modell	Theil U Trainingsmenge	Theil U Validierungsmenge
SDIF MLR ohne PCA	0.0011883365600	0.001843675204
SDIF KNN 21x7x1 ohne PCA	0.0009565832362	0.001923904365
SDIF KNN 7x10x1 (mit 6% PCA)	0.0007852735652	0.001914197909

Tabelle 3: Fehlerwerte der KNN und MLR Prognosen mit und ohne PCA für die Reihe StromBRD.

Durch die PCA konnte die Leistung der KNN-Prognose auf der Test- und Validierungsmenge verbessert werden. Dennoch ist das Modell mit der MLR auf der Validierungsmenge etwas besser.

6.3 Hybrides MLR und KNN Modell

Im folgenden sollen die Erkenntnisse aus den vorangegangenen Schritten dazu benutzt werden ein hybrides Modell MLR+KNN zu entwickeln. Hierzu werden in einem ersten Schritt äquivalent zum Modell von Zhang die Daten durch das lineare MLR-Modell prognostiziert. Die Fehler der MLR werden anschließend durch ein KNN modelliert. Das KNN soll dabei das durch das lineare MLR-Modell nicht erklärten Anteil analysieren und prognostizieren um dadurch die Prognosegüte des gesamten hybriden Modelles zu verbessern.

Für die MLR Prognose wird das Modell mit der besten Performanz benutzt. Dies war dasjenige ohne PCA. Es werden also alle 21 Datenspalten für die MLR benutzt. Anschließend werden die Fehler der MLR als Differenz zwischen Originalwerten und den MLR-Prognosen ermittelt. Die Prognosen der MLR werden zusammen mit der Datenmatrix durch die PCA wieder auf 7 Spalten komprimiert. Die Fehlerwerte der MLR werden als Ausgabewerte des Netzes benutzt. Die einfache Differenzierung der Ausgabewerte führt in diesem Fall zu einer Überdifferenzierung. Daher wird keine Differenzierung angewendet.

Modell	Theil U Trainingsmenge	Theil U Validierungsmenge
SDIF MLR ohne PCA	0.0011883365600	0.001843675204
SDIF KNN 21x7x1 ohne PCA	0.0009565832362	0.001923904365
SDIF KNN 7x10x1 (mit 6% PCA)	0.0007852735652	0.001914197909
Hybrid (mit 6% PCA) KNN 7x15x1	0.0010672858820	0.001837605779

Tabelle 4: Fehlerwerte der MLR und KNN Prognosen mit und ohne PCA sowie des hybriden Modells für die Reihe StromBRD.

Die Kombination aus MLR und KNN Modell lieferte eine kleine Verbesserung der Generalisierung und ist daher etwas besser als das MLR Modell. Die höhere Prognosegenauigkeit der KNN Modelle auf der Trainingsmenge spricht dafür, dass diese Modelle die Daten mehr Speichern.

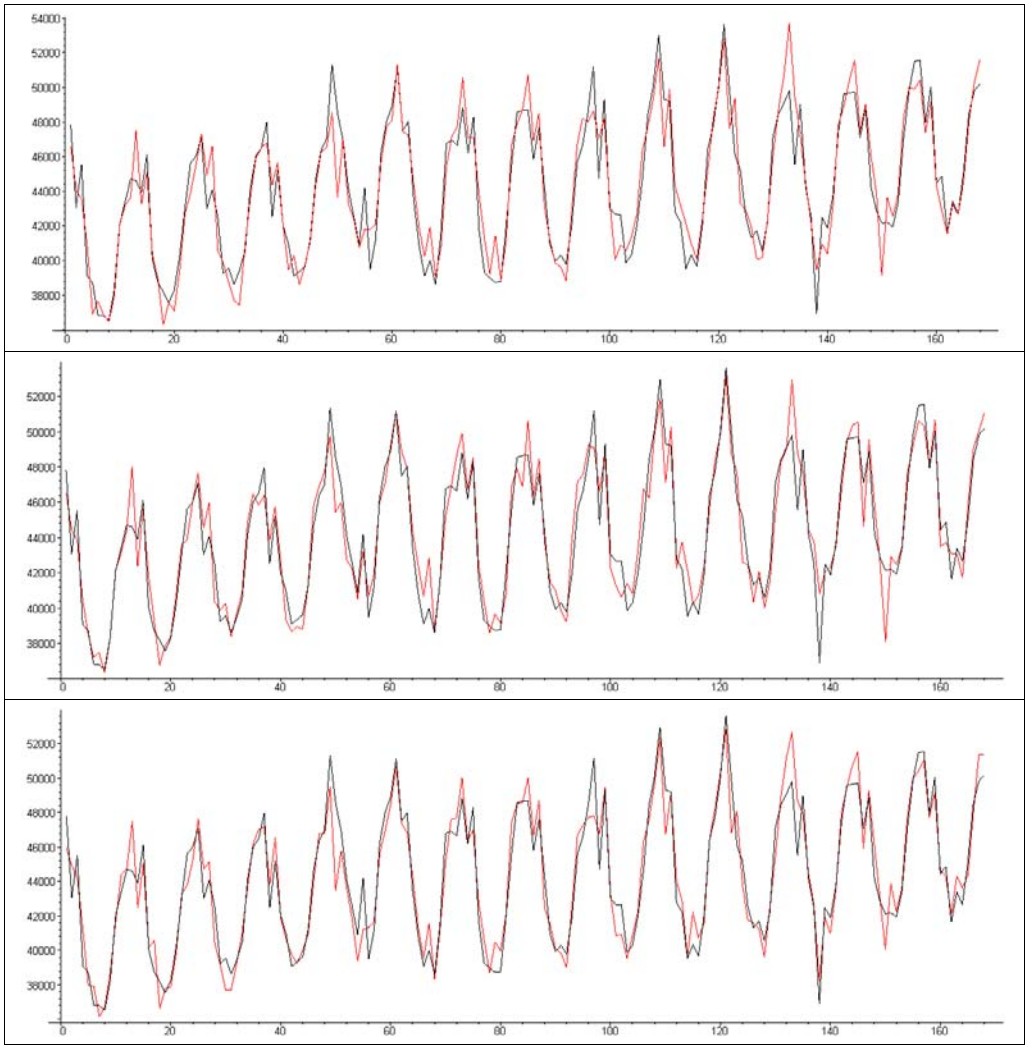


Abb. 40: Prognosen der Modelle SDIF MLR (oben), SDIF KNN mit PCA (mitte) und des hybriden Modells (unten).

6.4 Hybride Methodologie

Im folgenden wird das im Abschnitt 6.3 erstellte hybride Modell als Flußdiagramm dargestellt.

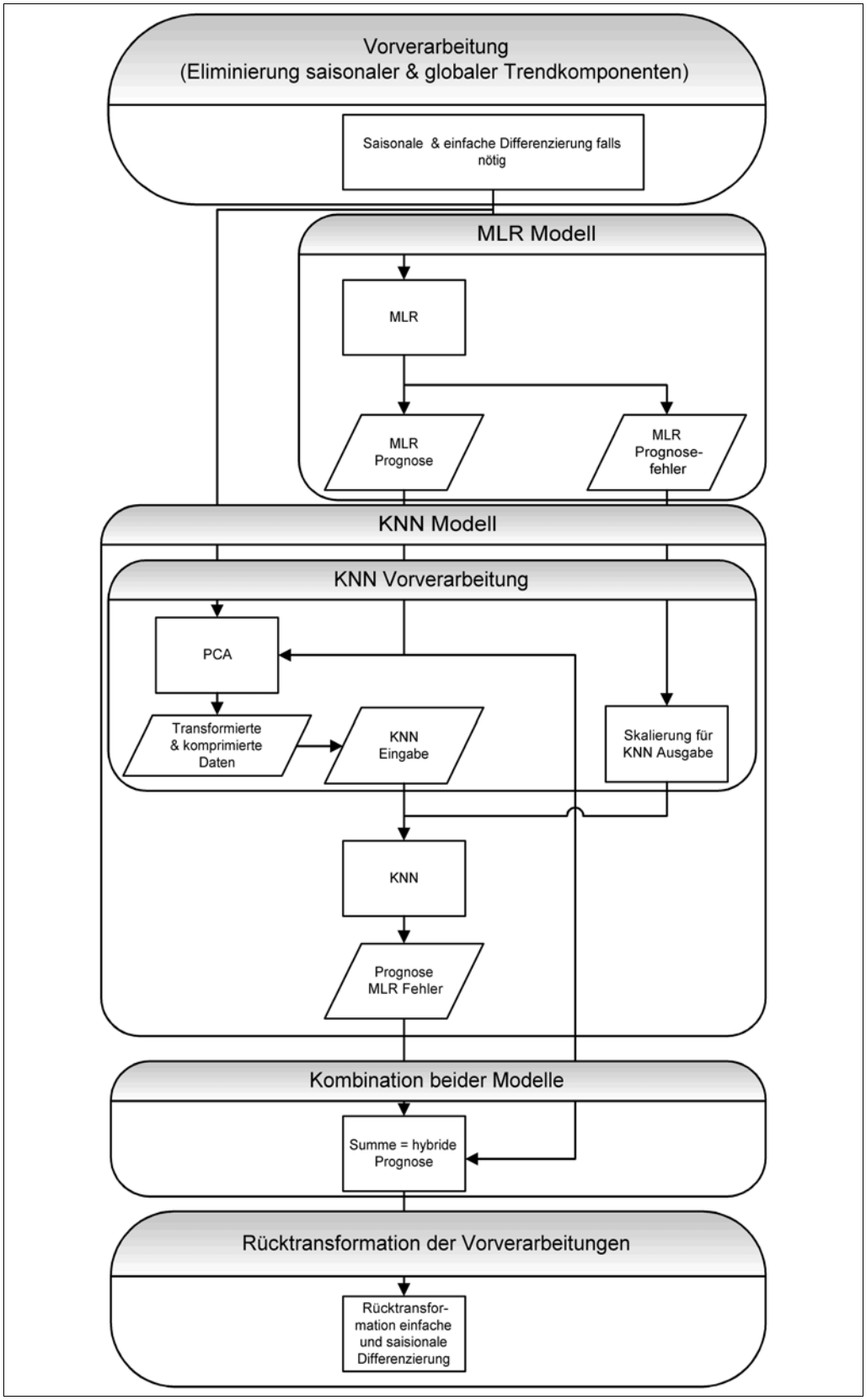


Abb. 41: Flußdiagramm zum Vorgehen beim hybriden Modell.

6.5 Verifizierung an einem weiteren Beispiel

Zur Verifizierung der Ergebnisse werden die gleichen Modelle auf anderen Datenreihen angewendet. Im folgenden wird die monatliche Schlussnotierung des Deutschen Aktienindex (DAX) prognostiziert. Als exogene Reihen stehen die monatlichen Schlussnotierungen der Indizes Paris CAC 40 (kurz CAC) aus Frankreich und der Dow-Jones (kurz DJ) aus den USA zur Verfügung. Alle Werte liegen im Zeitraum zwischen dem 01. 1989 und dem 12. 2004.

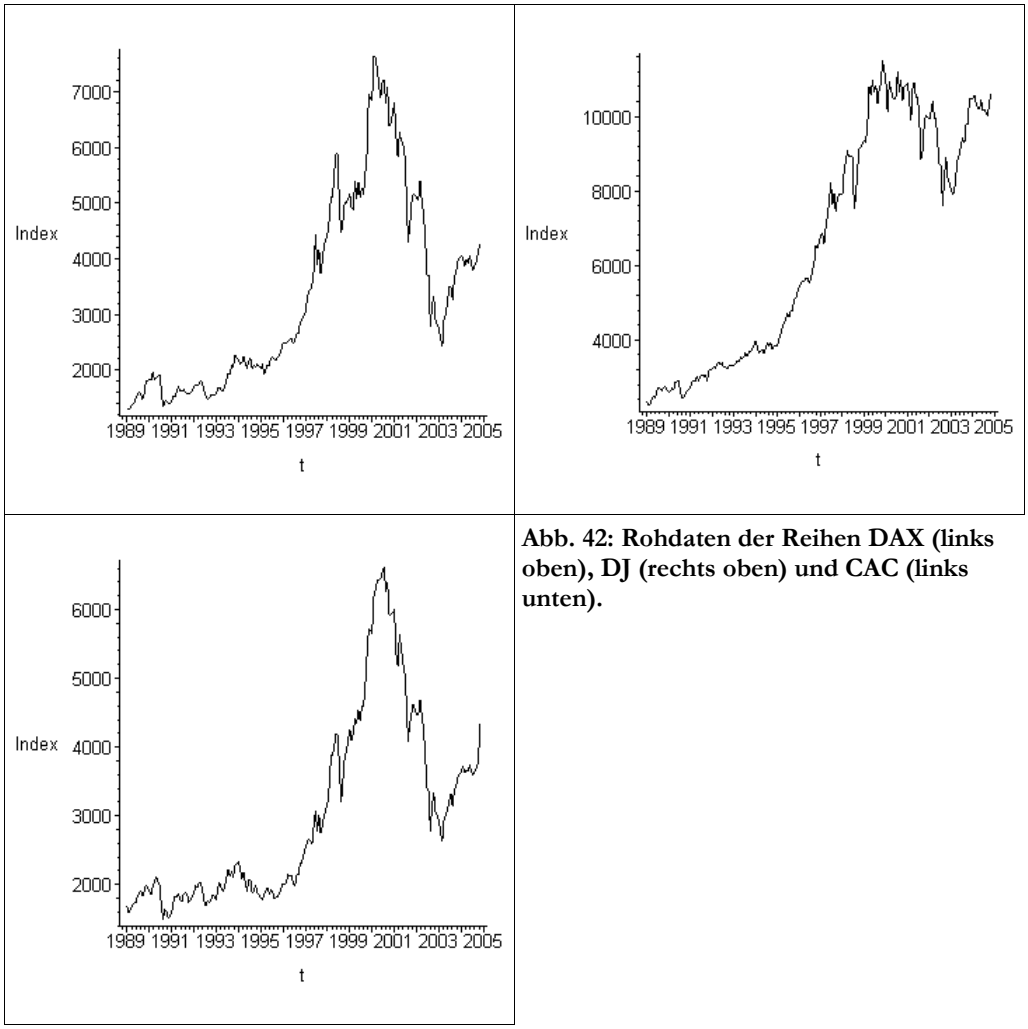


Abb. 42: Rohdaten der Reihen DAX (links oben), DJ (rechts oben) und CAC (links unten).

Die Korrelationsmatrix zeigt, dass eine starke positive lineare Korrelation zwischen den Reihen DAX, DJ und CAC besteht. Sowohl der CAC als auch der DJ haben ihre größte Korrelation zum DAX.

$$\begin{array}{c} \\ \text{DAX} \\ \text{DJ} \\ \text{CAC} \end{array} \begin{array}{c} \begin{array}{ccc} \text{DAX} & \text{DJ} & \text{CAC} \end{array} \\ \begin{bmatrix} 1. & .9156 & .7687 \\ .9156 & 1. & .7723 \\ .7687 & .7723 & 1. \end{bmatrix} \end{array}$$

Um den Anteil der globalen und saisonalen Komponenten zu ermitteln, wird die ACF der Reihe DAX ermittelt (siehe Abb. 43 links). Gleichzeitig wird die Reihe

durch einen saisonalen Indexplot dargestellt, um eine eventuell vorhandene saisonale Komponente zu identifizieren (siehe Abb. 43 rechts). Der ACF zeigt, dass eine sehr starke Korrelation über mehrere Lags existiert. Dies deutet auf einen starken globalen Trend. Der saisonale Indexplot hingegen zeigt, dass die Saisonkomponente sehr klein ist.

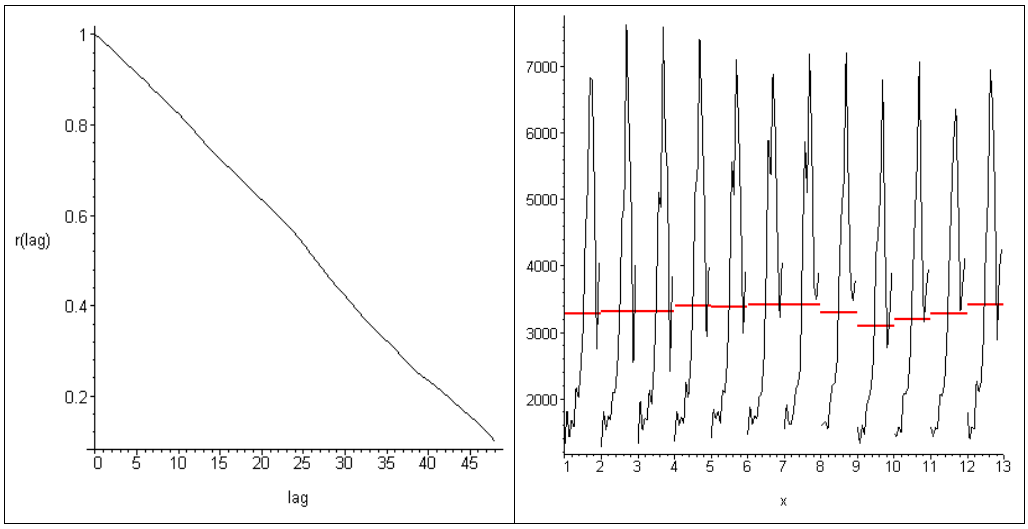


Abb. 43: ACF (links) sowie saisonaler Indexplot (rechts) der Reihe DAX.

Um eine geeignete Vorverarbeitung zu finden, werden wieder mehrfache Differenzierungen durchgeführt und das Minimum der Standardabweichung der differenzierten Reihen betrachtet. Die Standardabweichung der verschiedenen Differenzierungsordnungen (siehe Abb. 44 oben rechts) legt eine einfache Differenzierung mit der Ordnung eins nahe.

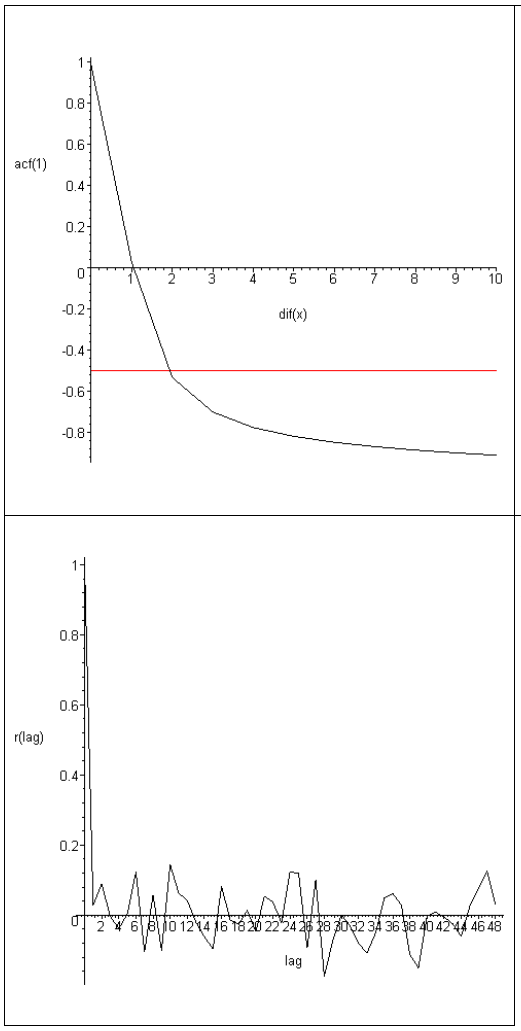


Abb. 44: ACF(1) (oben links) und Standardabweichung (oben rechts) bei verschiedenen Differenzierungsordnungen sowie ACF nach einfacher Differenzierung (unten links) der Reihe DAX.

Als Validierungsmenge wurden die Daten der letzten zwei Jahre bei der MLR und den KNN Modellen außen vor gelassen. Die Versuche zeigen, dass die einfache Differenzierung den Fehler auf der Validierungsmenge verbessert, den Fehler auf der Trainingsmenge jedoch etwas verschlechtert.
Bei den KNN, welche mit den differenzierten Werten der Lags 1, 2, 3, 6 und 12 trainiert wurden, zeigte das KNN mit der Netzstruktur 15x13x1 die beste Performanz auf der Validierungsmenge, welche jedoch schlechter waren als die der MLR der differenzierten Werte. Durch die PCA als Vorverarbeitung für das KNN wurde die Prognosequalität auf beiden Mengen verbessert. Bei der PCA wurden die 15 dimensionale Eingabewerte auf 10 Dimensionen reduziert. Als Schwellwert für die prozentualen Eigenwerte wurde 3% gewählt. Alle Eigenvektoren mit einem prozentualen Eigenwert weniger als 3% wurden entfernt.
Schließlich wurde das hybride Modell benutzt, welches als Eingabe die PCA transformierten Eingabewerte der MLR und die Prognosewerte der MLR erhielt. Als Ausgabe wurde der Fehler der MLR benutzt. Das hybride Modell zeigte die beste Performanz auf beiden Mengen. Die Ergebnisse der besten Modelle sind in der folgenden Tabelle zusammengefaßt.

Modell	Theil U Trainingsmenge	Theil U Testmenge
Lag 1 MLR	0.002746308290	0.005430276859
DIF Lags 1, 2, 3, 6, 12 MLR	0.002847842302	0.005240176221
DIF Lags 1, 2, 3, 6, 12 KNN 15x13x1	0.003298770650	0.005432461085
DIF Lags 1, 2, 3, 6, 12 PCA KNN 10x7x1	0.003689591626	0.004940966729
Hybrid 1, 2, 3, 6, 12 PCA KNN 10x11x1	0.001193527095	0.004742521471

Tabelle 5: Fehlerwerte der verschiedenen Modelle für die Prognose der Reihe DAX.

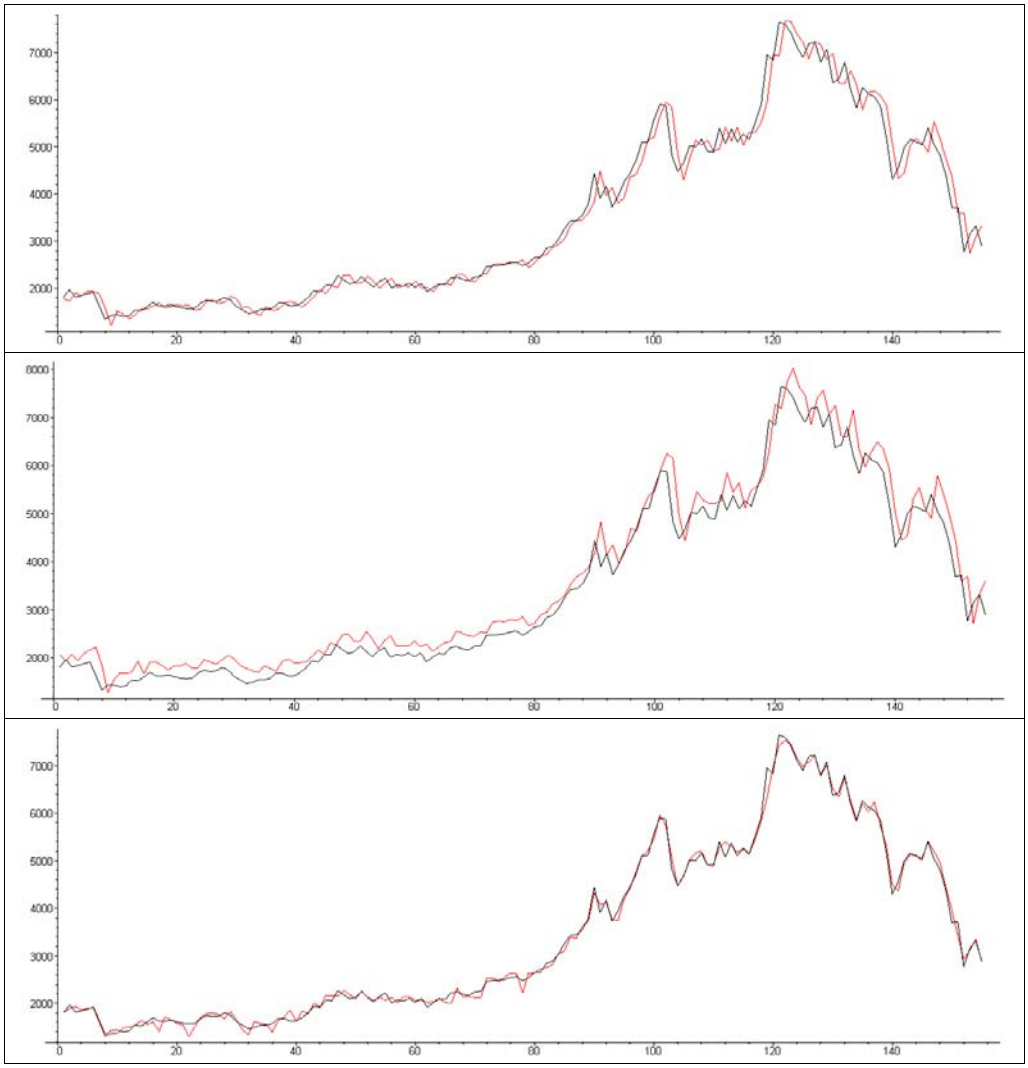


Abb. 45: Prognosen der Modelle DIF MLR (oben), DIF KNN mit PCA (mitte) und des hybriden Modells (unten).

6.6 Diskussion der Ergebnisse

Im Abschnitt 4.3 wurde die Fähigkeit dreischichtiger KNN zur gleichzeitigen Lösung einer PCA und MLR genannt und durch den Versuch des KNN Modells ohne PCA als Vorverarbeitung demonstriert. Dabei konnte das KNN die Eingabedaten in der verborgenen Schicht auf lediglich drei Datenwerte reduzieren und anschließend zwischen verborgener Schicht und Ausgabeschicht eine MLR durchführen. Da die linear erklärbaren Anteile der Daten dominierten, war die Dekorrelation der Daten durch die vom KNN durchgeführte PCA mit

anschließender MLR die effektivste Art, ein Minimum der Fehlerhyperfläche (Differenz der Hyperfläche der Daten und der vom Modell repräsentierten Hyperfläche) zu erreichen.
Die Vorverarbeitung der Lerndaten für das KNN durch die PCA verbesserte die Leistung des KNN, erhöhte jedoch auch die Anzahl der verborgenen Neuronen. Dies zeigt, dass das Netz nun weniger Leistung in die Datenkompression steckte. Die dominierenden linearen Strukturen wurden durch die PCA geglättet, die Fehlerhyperfläche bot damit kein großes Minimum mehr für die linear korrelierten Anteile, wodurch das Netz mehr Leistung zur Ermittlung nichtlinearer Strukturen und damit Minima für nichtlineare Korrelationen ansteuern konnte. Einen Hinweis hierzu gibt die größere verborgene Schicht als bei dem KNN Modell ohne PCA. Schließlich wurde im hybriden Modell der lineare Anteil sowohl durch die PCA als auch durch die MLR besonders gut abgedeckt. Die Fehlerhyperfläche enthielt somit primär Minima für nichtlineare Korrelationen, auf welche das Netz nun seine ganze Leistung konzentrieren konnte. Die große Anzahl der Neuronen, die für die verborgene Schicht nötig waren, deutet stark darauf hin, dass das Netz in diesem Modell keine lineare PCA mehr durchführen musste.

7 Zusammenfassung und Ausblick

„Prediction is very difficult, especially if it's about the future."
Niels Bohr[97]

7.1 Zusammenfassung

Mit Hilfe von graphischen Methoden, wie der Graph der ACF, der saisonale Plot oder der saisonale Indexplot können globale und saisonale Trendkomponenten einfach und visuell detektiert werden. Die Differenzierung in einfacher und saisonaler Form bietet eine einfache Möglichkeit, die Trendkomponenten einer Reihe auszuschalten. Die Trend bereinigte Reihe kann anschließend durch lineare Modelle prognostiziert werden. KNN können anschließend die Residuen des linearen Modells mit Hilfe der Ausgaben des linearen Modells und der Datenreihen verbessert werden. Dabei kann die PCA-Transformation der Eingabedaten für das KNN dessen Prognose weiter verbessern. Solche hybriden Modelle können Vorteile beider Methoden ausnutzen. Die Verbesserung der Prognosequalität hängt jedoch stark davon ab, ob die Reihen außer einem linear erklärbaren Mustern auch nichtlinear erklärbare Muster aufweisen. Die Prognose der Reihe StromBRD zeigte, dass diese Reihen sehr wenig nichtlineare Muster besitzen, was sich in einer sehr geringen Verbesserung der MLR Prognose durch das hybride Modell zeigte. Dagegen konnte die Prognosequalität der Reihe DAX stärker verbessert werden, was für etwas mehr nichtlineare Muster spricht.

7.2 Ausblick

Im folgenden werden Anregungen für die Weiterführung der in dieser Arbeit behandelten Themen genannt, deren Ausführungen die Grenzen dieser Arbeit sprengen würden.

7.2.1 Weitere hybride Modelle

In dieser Arbeit wurden primär auf MLR basierende Verfahren und deren Kombinationen mit KNN erläutert. Diese sind jedoch nicht die einzigen oder überhaupt die besten Prognosemodelle. Weitere Kombinationen von Prognosemodellen, wie Wiener-Filter oder Kalman-Filter, von denen letztere auch nichtlineare Korrelationen aus den Daten verwenden können, mit KNN sind denkbar.

7.2.2 Alternative Kodierungen

In vielen Fällen können alternative Kodierungen der Eingabewerte die Performanz des KNN verbessern. Hierfür sind Beispielsweise folgende Kodierungen denkbar: das KNN wird auf zwei Ausgabeneuronen erweitert, von denen eines lediglich das Vorzeichen der Steigung bestimmt und damit vorgibt, ob der nächste Wert steigend oder fallend ist. Das zweite Ausgabeneuron gibt dann lediglich den Betrag der Änderung an.

7.2.3 SOM

[97] Niels Bohr (*1885 †1962), dänischer Physik-Nobelpreisträger.

SOM (Self organizing maps), meist auch Kohonen[98]-Karten genannt, können durch überwachtes oder unüberwachtes Lernen Daten zu Clustern zusammenfassen. Hierdurch könnten Segmente von Zeitreihen daraufhin klassifiziert werden, ob ihnen ein Aufwärts- oder ein Abwärtstrend folgen wird. Desweiteren können durch die Clusteranalyse verlaufsrelevante Segmente oder auch Ausreißer identifiziert werden.

7.2.4 Komitees von KNN

Bereits in vielen Arbeiten wurden mehrere KNN gleichzeitig auf ein Problem angewendet. Hierbei könnten die Daten z.B. durch verschieden große KNN parallel klassifiziert bzw. prognostiziert werden, um dann die Ergebnisse gemeinsam zu evaluieren. Auf diese Weise wird eine bessere Generalisierung erreicht[99]. Ragg behandelt die Benutzung von Komitees in seiner Dissertation sehr ausführlich.
Eine weitere Einsatzmöglichkeit von KNN-Komitees wäre es, verschiedene Aspekte der zu prognostizierenden Werte, wie z.B. steigende / fallende Tendenzen und Betrag des Wertes durch verschiedene, spezialisierte KNN getrennt zu evaluieren.

7.2.5 Hybride KNN

Bereits in einigen Arbeiten wurden hybride KNN, welche z.B. ein Feedforward Netze mit einer SOM zu einer BP-SOM kombinieren untersucht.[100] Dabei werden die Neuronen der verborgenen Schicht mit einer SOM verbunden. Dadurch kann die Generalisierungsfähigkeit des Feedforward Netzes verbessert werden.
Auch könnten die multiple lineare Regression sowie die PCA durch entsprechende KNN ersetzt werden. Damit könnte ein erstes Netz einen Lagausschnitt von 12 Werten erhalten, die es dann äquivalent zur PCA und MLR prognostiziert. Ein zweites Netz könnte die Werte der verborgenen Schicht des ersten Netzes als PCA-transformierte Werte sowie die Ausgabe des ersten Netzes (PCA+MLR Prognose) als Eingabe erhalten um die Fehler des ersten Netzes zu prognostizieren.

7.2.6 Rekurrente KNN

Rekurrente KNN besitzen Verbindungen aus Neuronen der höheren Schichten zurück zur Eingabe. Sie besitzen die Fähigkeit zeitliche Zusammenhänge ohne Lag zu erkennen. Ihre Anwendung auf Zeitreihen ist daher prinzipiell geeignet. Durch rekurrente KNN fällt auch die Auswahl eines Kontextfensters weg.

7.2.7 Schnellere KNN

Ein Hauptproblem von KNN ist ihr recht aufwändiges Training, dass sehr viel Zeit in Anspruch nehmen kann (in Abhängigkeit der Anzahl der Lernmuster, ihrer Dimension sowie der Netzkomplexität). Im folgenden sollen Möglichkeiten gezeigt werden, KNN schneller zu evaluieren. Durch die schnellere Evaluierung von KNN werden diese auch praktisch anwendbarer.

GPUs

[98] Nach Teuvo Kohonen (*1934), finischer Ingenieur und Erfinder der SOM.
[99] Vergl. [Ragg 00] Kapitel 3.
[100] Vergl. [Callan 99] S. 151ff.

Während üblicherweise die KNN durch die CPU eines Rechners meist seriell evaluiert werden, sind GPUs (grafische Prozessoren von Grafikkarten) darauf ausgelegt mathematische Transformationen sehr schnell und auf mehrere Daten parallel anzuwenden (zur Berechnung von Texturen und Beleuchtungseffekten). Oh und Jung beschreiben in ihrer Veröffentlichung[101], wie die GPU so programmiert werden kann, dass sie KNN evaluieren. Hierzu werden die Daten der KNN als Texturen und die Evaluierung als Transformationen dieser Texturen interpretiert. Die in parallelen Berechnungen überlegenen GPUs konnten somit KNN mit 20-fach höherer Geschwindigkeit als entsprechende CPUs trainieren.

Neurochips

Die zunehmende Parallelisierung von Rechenvorgängen ermöglicht eine effizientere Evaluierung von KNN, welche sehr gut parallelisiert werden können. Es wurden bereits mehrere Neuro-Prozessoren entwickelt, welche das klassische serielle Rechnen eines Prozessors beim Trainieren von KNN ablösen können. Die Neurochips benutzen hierzu Vektor-CPUs, welche auf KNN Operationen optimiert wurden[102].

Biochips

Ein sehr interessanter Artikel[103] berichtete über den Einsatz biologischer Neuronen zur Lösung von Problemen. Hierbei gelang es Forschern der University of Florida Gehirnzellen einer Ratte in einer Petrischale zu züchten und diese zur Bedienung eines simplen Flugsimulators zu trainieren. Nichts liegt näher, als das Prinzip der KNN in ihrer Urform, den biologischen neuronalen Netzen, einzusetzen. Diese Technologie ist sicherlich noch experimentell, weist jedoch die Zukunft der Neuroinformatik. Schließlich besitzt ein biologisches neuronales Netz alle Vorzüge einer massiven parallelen Datenverarbeitung, die mit herkömmlichen Computern noch nicht erreicht wurde.

7.2.8 Anwendung der Modelle im Audiobereich

Die Zeitreihenprognose hat große Parallelen zu anderen Bereichen. Beispielsweise werden in der Audiobearbeitung ähnliche Verarbeitungsschritte benutzt, welche als Filter bestimmte Eigenschaften von Audiodaten herausfiltern und Regelmäßigkeiten approximieren. Vor allem in der Filmbranche müssen bei Nachvertonungen von Filmen die akustischen Eigenschaften des Originaltons nach gebaut werden, damit der nach vertonte Film die gleiche Atmosphäre wiederspiegelt wie im Originalton. Hierzu werden primär lineare Filter, wie etwa Fouriertransformationen benutzt, um einen Filter zu schaffen, der die rohen Audiodaten der Nachvertonung dem Originalton anpasst. Der Einsatz von nichtlinearen Modellen, wie den KNN und den hybriden Modellen würde sicherlich eine gute Alternative bieten.

[101] Vergl. [Oh-Jung 04].
[102] Vergl. [Rojas 93] Kap. 18.
[103] Vergl. [Rötzer 04].

Literaturverzeichnis

[Arsham 05] Arsham, Hossein: Time-Critical Decision Making for Economics and Finance.
University of Baltimore, 1994-2005.
http://home.ubalt.edu/ntsbarsh/stat-data/Forecast.htm#rhomave
(Abruf 22.04.05)

[Arranz 03] Arranz, Miguel Angel: Unit root tests with additive outliers.
Universität Carlos III de Madrid, 2003.
Seminarpapier an der Universität de Salamanca, Fakultät für Ökonomie und Geschichte der Ökonomie.
http://www3.usal.es/~ehe/Semin.htm
(Abruf 12.05.05)

[BaumHassler 89] Baum, E. B.; Hassler, D.: What size net gives valid generalisation?
In: Neural Computation, Ausg. I (1989), S. 151-160.

[BoxCox 64] Box, G.E.P.; Cox, D.R.: An analysis of transformations.
Journal of the Royal Statistical Society, 1964.

[BoxJenkins 76] Box, G.E.P.; Jenkins, G.M.: Time series analysis – forecasting and control.
Holden-Day Inc., Oakland California, 1976.

[Buttler 03] Buttler, Günter: Ein einfaches Verfahren zur Identifikation von Ausreißern bei multivariaten Daten.
Diskussionspapier, Friedrich-Alexander-Universität Erlangen-Nürnberg, 1996.
http://www.statistik.wiso.uni-erlangen.de/forschung/publikationen.html
(Abruf 20.04.2005)

[Brown 63] Brown, Robert Goodell: Smoothing, Forecasting, and Prediction.
Prentice Hall, Englewood Cliffs, 1963.

[Callan 99] Callan, Robert: Neuronale Netze im Klartext.
Prentice Hall Europe, 1999.

[Census 02] X-12-ARIMA Reference Manual V. 0.2.10
U.S. Census Bureau, 2002.

[Chatfield 89] Chatfield, Christopher: The analysis of time series: an introduction.
4th Edition. Chapman & Hall, 1989.

[EduQu 97] Ohne Authorenangabe: How to Draw a Boxplot.
Education Queensland, 1997.
http://exploringdata.cqu.edu.au/box_draw.htm
(Abruf 20.05.2005)

[Heiler 94] Heiler, Siegfried; Michels, Paul: Deskriptive und Explorative Datenanalyse.
R. Oldenbourg Verlag, 1994.

[IrieMijake 88] Irie, B.; Miyake, S.: Capabilities of Three-layered Perceptrons.
In: IEEE Vol. I (1988), S. 641-648.

[Jansen 03] Jansen, Michiel: Seasonal and working day adjustment for the Industry Production Index – Methodological Note.
Statistics Netherlands, 2003.
www.cbs.nl/en/publications/articles/macro-economics/business-cycle/explanation.pdf
(Abruf 04.05.2005)

[Juntilla 01] Juntilla, Juha: Structural breaks, ARIMA model and Finnish inflation forecasts.
In: International Journal of Forecasting Ausg. 17 (2001) S.203–230.

[Kaastra 94] Kaastra, Iebeling; Boyd, Milton: Designing a neural network for forecasting financial and econometric time series.
In: Neurocomputing Ausg. 10 (1996), S. 215-236.

[Katz 92] Katz, J.O.: Developing neural network forecasters for trading.
In: Technical Analysis of Stocks and Commodities Ausg. April (1992), S. 58-70.

[Kolmogorov 57] Kolmogorov, A. N.: On the Representation of Continuous Functions of Several Variables by Superposition of Continuous Functions of One Variable and Addition.
In: Dokl. Akad. Nauk SSSR, Vol. 114 (1957), S. 369--373.

[Knorr 00] Knorr, Edwin M.; Ng, Raymond T.; Tucakov, Vladimir: Distance-Based Outliers: Algorithms and Applications.
In: The VLDB Journal Ausgabe 8 (2000), S.237-253.

[Läuter 89] Läuter, Henning; Pincus, Richard: Mathematisch-statistische Datenanalyse.
R. Oldenbourg Verlag, 1989.

[Leiner 82] Leiner, Berndt: Einführung in die Zeitreihenanalyse.
R. Oldenbourg Verlag, 1982.

[Maple 03] Maple 9 Learning Guide.
Waterloo Maple Inc., 2003.

[Nau 03] Nau, Robert F.: Online-Skript Forecasting Course Decision 411.
Duke University - The Fuqua School of Busines (Durham, NC USA).
http://www.duke.edu/~rnau/Decision411CoursePage.htm
(Abruf 10.05.2005)

[Nauck 96] Nauck, Detlef; Klawonn, Frank; Kruse, Rudolf: Neuronale Netze und Fuzzy-Systeme.
Vieweg Verlag, 1994-1996.

[Oh-Jung 04] Oh, Kyoung-Su; Jung, Keechul: GPU implementation of neural networks.
In: Pattern Recognition 37 (2004) S. 1311 – 1314.

[Pokropp 94] Pokropp, Fritz: Lineare Regression und Varianzanalyse. Oldenbourg Verlag München, 1994.

[Pollock 96] Pollock, D. S. G. : A Handbook of Time-Series Analysis, Signal Processing and Dynamics. Academic Press, USA / UK, 1992.

[PrinsNIST] Prins, Jack: NIST/SEMATECH e-Handbook of Statistical Methods, Chapter 6 - Process or Product Monitoring and Control. http://www.itl.nist.gov/div898/handbook/ (Abruf 20.04.2005)

[Ragg 00] Ragg, Thomas: Problemlösung durch Komitees neuronaler Netze. Dissertation an der Universität Karlsruhe (Technische Hochschule) in der Fakultät für Informatik, 2000.

[Riedmiller 92] Riedmiller, Martin; Braun, Heinrich: RPROP – A Fast Adaptive Learning Algorithm. Institut für Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe, 1992

[Rojas 93] Rojas, Raul: Theorie der neuronalen Netze. Springer Verlag, 1993.

[Rötzer 04] Rötzer, Florian: Chip im Gehirn oder biologisches Gehirn in der Maschine. In: Telepolis, Heise.de (25.10.2004). http://www.heise.de/tp/r4/artikel/18/18644/1.html

[Schlittgen 01] Schlittgen, Rainer; Streitberg, Bernd: Zeitreihenanalyse. R. Oldenbourg Verlag, 2001.

[Smith 02] Smith, Lindsay I.: A tutorial on Principal Component Analysis. Ohne Veröffentlichung. http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf (Abruf 02.05.2005)

[Thiesing 98] Thiesing, Frank M.: Analyse und Prognose von Zeitreihen mit Neuronalen Netzen. Shaker Verlag, 1998.

[Tsay 51] Tsay, Ruey. S.: Analysis of Financial Time Series. John Wiley & Sons Inc., USA, 1951 (Auflage 2002).

[Tsay 86] Tsay, Ruey. S.: Time series model specification in the presence of outliers. In: Journal of the American Statistical Association Ausg. 81 (1986), S.132–141.

[Tukey 83] Paul, Tukey; Chambers, John; Cleveland, William; Kleiner, Beat: Graphical Methods for Data Analysis. Wadsworth, 1983.

[Walter 99] Walter, Andreas; Skript Rekurenzplots zur Untersuchung von Zeitreihen auf periodisches Verhalten
o.V., 1999
http://user.uni-frankfurt.de/~andreas/rekur/
(Abruf 02.05.2005)

[Westenberger 04] Westenberger, Hartmut: Skript Künstliche Neuronale Netze II.
Fachhochschule Köln Abteilung Gummersbach, Wintersemester 2004

[Yule 27] Yule, George Udny: On a Method of Investigating Periodicities in Disturbed Series with Special Reference to Wolfer's Sunspot Numbers,
In: Philosophical Transactions of the Royal Society, Series A (1927), S. 226, 267-298.

[Zhang 04] Zhang, G. Peter: Neural Networks in Business Forecasting.
Idea Group Publishing USA, 2004.

[Zhang 03] Zhang, G. Peter: Time series forecasting using a hybrid ARIMA and neural network model.
In: Neurocomputing Ausg. 50 (2003), S.159 – 175.

Anhang

A. Zeitreihen

Die meisten der verwendeten Zeitreihen stammen aus der Eurostat Onlinedatenbank, welche unter der URL
http://epp.eurostat.cec.eu.int/portal/page?_pageid=1996,45323734&_dad=portal&_schema=PORTAL&screen=welcomeref&open=/&product=EU_MAIN_TREE&depth=1
zu finden ist.

ArbeitslosBRD	Die Arbeitslosenzahlen in Deutschland (Bestand an Arbeitsuchenden, nichtarbeitslosen Arbeitsuchenden und Arbeitslosen, einschließlich Teilzeitarbeitsuchende, Ausländer, Jugendliche, Schwerbehinderte und Spätaussiedler) jeweils am Monatsende für die Monate 01/1993 (t=1) bis 12/1999 (t=84). Quelle: Eurostat Onlinedatenbank
CAC	Monatliche Paris CAC 40 Index Schlussnotierung vom 1.1.1989 bis zum 1.12.2004. Quelle: www.handelsblatt.com
DAX	Monatliche Deutscher Aktien Index Schlussnotierung vom 1.1.1989 bis zum 1.12.2004. Quelle: www.handelsblatt.com
DJ	Monatlichee Dow-Jones Industial Average Index Schlussnotierung vom 1.1.1989 bis zum 1.12.2004. Quelle: www.handelsblatt.com
DJ daily	Tägliche Dow-Jones Industial Average Index Schlussnotierung vom x.x.1950 bis zum x.x.2005. Quelle: www.handelsblatt.com
StromBRD	Bruttostromverbrauch der BRD in Gigawattstunden, monatlich für den Zeitraum 01.1985-01.2005. Quelle: Eurostat Onlinedatenbank
StromES	Bruttostromverbrauch von Spanien in Gigawattstunden, monatlich für den Zeitraum 01.1985-01.2005. Quelle: Eurostat Onlinedatenbank
StromFR	Bruttostromverbrauch von Frankreich in Gigawattstunden, monatlich für den Zeitraum 01.1985-01.2005. Quelle: Eurostat Onlinedatenbank
StromNL	Bruttostromverbrauch der Niederlande in Gigawattstunden, monatlich für den Zeitraum 01.1985-01.2005. Quelle: Eurostat Onlinedatenbank
StromUK	Bruttostromverbrauch des United Kingdom in Gigawattstunden, monatlich für den Zeitraum 01.1985-01.2005. Quelle: Eurostat Onlinedatenbank

B. Maple-Quellkode

Da für verschiedene Teilaufgaben (Zeitreihenanalyse, Prognose, Visualisierung etc) recht viele Methoden erstellt wurden, sollen hier lediglich Kurzbeschreibungen der wichtigsten Methoden aufgeführt werden. Die Maple-Worksheets sind auf der beiliegenden CD zu finden.

00 library.mws

Dieses Worksheet enthält die meisten Routinen zur Zeitreihenanalyse. In weiteren Worksheets wurden sie benutzt, um Zeitreihen und ihre Eigenschaften hauptsächlich Visuell darzustellen. Durch Ausführen dieses Worksheets wird eine Maple-Bibliothek erzeugt, welche alle Methoden für weitere Worksheets zur Verfügung stellen.

```
getData := proc(pFileName)
```
Liefert eine Sequenz der Daten aus einer Textdatei. Die Daten müssen jeweils in einer eigenen Zeile stehen. Die erste Zeile wird als Kommentar nicht ausgewertet.

```
storeData := proc(pFilename, pTitle, pData)
```
Speichert eine Sequenz von Daten in der angegebenen Datei. Jeder Wert wird in einer eigenen Zeile gespeichert, die erste Zeile enthält den Titel.

```
const_season_monthly := 12:
const_season_workdaily := 252:
```
Konstanten mit vordefinierten Werten pro Saison.

```
plotDataColLabDot := proc(pData, pFirstIndex, pLagSize, pColor,
pPlotDots, pXlabel, pYlabel)
```
Zeichnet einen Plot der Daten mit einer Jahresbeschriftung auf der X-Achse. pFirstLag beschreibt das Datum (z.B. Jahr) des ersten Datensatzes. LagSize definiert die Saisonlänge, PlotDots legt fest, ob zu dem durchgängige Graphen zusätzlich Punkte für jeden Wert gezeichnet werden sollen. Die Labelparameter geben die Beschriftung der X- und Y-Achsen an.

```
plotConvertScale := proc(pXValue, pFirstIndex, pLagSize)
```
Konvertiert einen Arrayindex in eine X-Koordinate für einen Plot um.

```
addConst := proc(pData, pConst)
```
Addiert eine Konstante zu jedem Wert einer Sequenz.

```
addData := proc(pData1, pData2)
```
Addiert Komponentenweise die Werte zweier Sequenzen zusammen.

```
subData := proc(pData1, pData2)
```
Subtrahier Komponentenweise die Werte der ersten Sequenz von den Werten der zweiten Sequenzen.

```
invertData := proc(pData)
```
Kehrt die Reihenfolge der Werte in einer Sequenz um.

```
correlation := proc(pData1, pData2)
```
Berechnet die lineare Korrelation zweier Reihen

`correlationMatrix := proc(pDatas)`
Berechnet die Korrelationsmatrix zu einer Sequenz von Reihen.

`laggedCorrelation := proc(pData1, pData2, pFirstlag, pLastlag)`
Berechnet die Korrelationen zweier Reihen mit den Lags FirstLag bis LastLag zueinander.

`plotLaggedCorrelation := proc(pData)`
Liefert einen Plot der gegebenen laggedCorrelation Werten.

`acf := proc(pData, pMaxLag)`
Berechnet die Werte der ACF einer Reihe. Der Acf(0) liegt bei Index 1.

`plotAcf := proc(pData)`
Liefert einen Plot der gegebenen Acf-Werte.

`polyReg := proc(pData, pDegree)`
Liefert die Gleichung einer Polynomregression auf die gegebene Reihe mit dem gegebenen Polynomgrad.

`seasonIndexSplit := proc(pData, pSeasonLength)`
Teilt die Werte einer Reihe in Gruppen zu Saisonindizes auf, z.B. in Gruppen für Januarwerte, Februarwerte etc.

`plotSeasonindex := proc(pData)`
Zeichnet einen saisonalen Indexpot mit den aus seasonIndexSplit gegebenen Werten. Die Saisondurchschnitte werden als rote Linien eingezeichnet.

`phasendurchschnittAddKennziffern := proc(pDataSis)`
Liefert die Saisonindexziffern anhand der gegebenen Werte aus der Methode seasonIndexSplit.

`phasendurchschnittAddClear := proc(pData, pSeasonKennziffern)`
Liefert die Phasendurchschnitt bereinigte Reihe zurück.

`phasendurchschnittAddUnaply := proc(pData, pSeasonKennziffern)`
Macht die Phasendurchschnittsbereinigung rückgängig.

`plotSeasonal := proc(pData, pSeasonLength)`
Zeichnet für jede Saison einen eigenen Graphen.

`autoregressGetAllMatrices := proc(pData, pOrder)`
Liefert die Matrizen X, Y und B (Koeffizienten) einer Autoregression der gegebenen Ordnung. Die Lags werden automatisch gebildet.

`autoregress := proc(pData, pOrder)`
Liefert lediglich die Koeffizienten einer Autoregression zurück.

`convertMatrixToSeq := proc(pMatrix)`
Liefert die Werte einer einspaltigen Matrix als Sequenz zurück.

`pacf := proc(pData)`
Extrahiert den letzten Koeffizienten aus einer Sequenz von Koeffizienten mehrerer Autoregressionen.

`plotPacf := proc(pData)`
Zeichnet anhand der Werte von pacf die partielle Autokorrelationsfunktion.

`getOptimalAutoregression := proc(pData, pMaxOrder, pSignificanceLevel)`
Führt Autogregressionen mit steigender Ordnung, bis der letzte Koeffizient kleiner ist als der gegebene Signifikanzwert.

`errorABSE := proc(pData1, pData2)`
Berechnet die Summe der Fehlerbeträge.

`errorSSE := proc(pData1, pData2)`
Berechnet die Summe der Fehlerquadrate.

`errorMAD := proc(pData1, pData2)`
Berechnet den mittleren Fehlerbetrag.

`errorMSE := proc(pData1, pData2)`
Berechnet den mittleren quadratischen Fehler.

`errorRMSE := proc(pData1, pData2)`
Berechnet die Wurzel des mittleren quadratischen Fehlers.

`errorMaxE := proc(pData1, pData2)`
Liefert den größten Fehlerbetrag.

`errorMAPE := proc(pData1, pData2)`
Berechnet den mittleren absoluten prozentualen Fehler.

`errorTheilU := proc(pData1, pData2)`
Berechnet den Theil'schen Ungleichheitskoeffizienten.

`simpleExponentialSmoothing := proc(pData, pWeightingFactor)`
Führt eine einfache exponentielle Glättung mit dem gegebenen Glättungsfaktor durch.

`findOptimalSimpleExponentialSmoothing := proc(pData, pMaxIter, pLeftInit, pRightInit)`
Liefert die optimale exponentielle Glättung per Intervallschachtelung.

`invertExponentialSmoothing := proc(pData, pWeightingFactor)`
Transformiert eine geglättete Reihe zurück zur Originalreihe.

`scaleStddev := proc(pData)`
Skaliert die Daten durch Subtraktion des Mittelwertes und Division durch Standardabweichung.

`unscaleStddev := proc(pScaledData, pMean, pStddev)`
Rücktransformation für die Skalierung scaleStddev.

`scaleLinear:=proc(pData, pMin, pMax)`
Skaliert die Daten linear zwischen Min und Max.

`unscaleLinear:=proc(pScaledData, pMin, pMax, pMinX, pMaxX)`
Rücktransformation für scaleLinear.

`dif := proc(pData, pOrder)`
Bildet die Differenzen der gegebene Ordnung.

`getOptimalDifOrder := proc(pData)`
Berechnet die optimale Differenzierungsordnung mit Hilfe der Standardabweichung und der ACF der intern steigend differenzierten Reihe.

`getDifOrderInfo := proc(pData, pMaxOrder)`
Liefert die Standardabweichungen und ACF bei Lag 1 bis zur gegebenen Ordnung der Differenzierung.

`undif := proc(pDiffs, pOriginalData, pOrder)`
Rücktransformation der Differenzierung gegebener Ordnung mit Hilfe der gegebenen Originalwerte.

`sdif := proc(pData, pOrder, pSeasonLen)`
Bildet die saisonalen Differenzen der gegebene Ordnung.

`unsdif := proc(pDiffs, pOriginalData, pOrder, pSeasonLen)`
Rücktransformation der saisonalen Differenzierung gegebener Ordnung mit Hilfe der gegebenen Originalwerte.

`getLowerAndUpperBounds := proc(pData)`
Liefert Unter- und Obergrenze der 1.5-fachen IQR.

`getOutrangedData := proc(pData, pRangeBounds)`
Liefert die X-Werte aller Punkte ausserhalb des angegebenen Bereiches wieder. RangeBounds=[Untergrenze, Obergrenze].

`getOutliers := proc(pData)`
Liefert die X-Werte von Ausreißern anhand der 1.5-fachen IQR.

`detectSimpleLevelShift := proc(pData, pOutlier)`
Liefert den Höhenunterschied zwischen zwei Regressionsgeraden auf beide Hälften der Daten, die am Punkt pOutlier getrennt werden sowie beide Regressionsgeraden.

`applySimpleLevelShift := proc(pData, pOutlier, pLeftreg, pRightreg)`
Entfernt einen einfachen Levelshift in den Daten anhand der Regressionsgeraden auf beiden Hälften.

`plotLevelShift := proc(pDataAmount, pOutlier, pLeftreg, pRightreg, pFirstIndex, pLagSize)`
Zeichnet den Graphen sowie die Regressionsgeraden eines Levelshifts als Plot.

`interpolate := proc(pData, pXValues, pPolynomOrder)`
Interpoliert die Werte, die in der Sequenz XValues angegeben wurden durch ein Polynom mit gegebenem Grad.

`ma := proc(pData, pOrder)`
Einfacher gleitender Durchschnitt.

`wma := proc(pData, pWeights)`
Gewichteter gleitender Durchschnitt. Die Anzahl der Gewichte geben die Laggröße an.

`mlr := proc(pMatrixX, pMatrixY)`
Führt eine multiple lineare Regression auf den gegebenen Matrizen durch. Die Daten werden in den Spalten erwartet.

`addOnesRowToMatrix := proc(pDataMatrix)`
Fügt der gegeben Matrix eine Spalte mit 1-en für die MLR hinzu.

`addRowToMatrix:= proc(pDataMatrix, pColToInsertAsSeq)`
Fügt einer Matrix eine Datenspalte hinzu.

`removeRowFromMatrix:=proc(pMatrix, pRowIndex)`
Entfernt eine Dateinzeile aus der gegebenen Matirx.

`covarianceMatrix := proc(pDataSeqs)`
Berechnet die Kovaranzmatrix der gegebenen Sequenz von Datensequenzen.

`laggedValueGenerator := proc(pDataSeqs, pLagSeqs, pTargetSeq)`
Liefert die Spalten einer Datenmatrix als Sequenz von Sequenzen sowie die Zieldatenspalte. Die gegebenen Lags werden als xt-lag interpretiert.

`meansSeqs := proc(pDataSeqs)`
Liefert die Mittelwerte mehrerer Reihen zurück.

`meanClearSeqs:=proc(pDataSeqs)`
Liefert die Mittelwertbereinigten Reihen sowie die Mittelwerte zurück.

`procentualEigenvals:=proc(pEigenVals)`
Konvertiert die Eigenwerte zu prozentualen Angaben.

`indexOf:=proc(pData, pValue)`
Liefert den ersten Index, bei dem der gegebene Wert in der Sequenz gefunden wird.

`orderEigenVecs := proc(pEigenVecs, pEigenVals)`
Ordnet die Eigenvektoren nach ihren Eigenwerten. Die Eigenvaktoren mit den höchsten Eigenwerten stehen in den ersten Spalten.

`reduceEigenvecs := proc(pEigenVecs, pEigenVals, pMinPercent)`
Entfernt die Eigenvektoren, deren prozentualer Anteil geringer ist als pMinPercent.

`pca := proc(pDataMatrix, pMinPercent)`
Führt eine PCA durch. Die Datenspalten werden Mittelwert bereinigt, die Kovarianzmatrix gebildet, die Eigenvektoren und -werte bestimmt und reduziert und die Daten transformiert und komprimiert.

`matrixDims:=proc(pMatrix)`
Liefert die Anzahl an Reihen und Spalten einer Matrix zurück.

`splitMatrix:=proc(pMatrix, pSplitIndex)`
Teilt eine matrix an der gegebenen Reihe in zwei Matrizen auf.

`writeSnnsPattern:=proc(pFileName, pInputMatrix, pOutputMatrix)`
Schreibt die gegebenen Matrizen als SNNS-kompatible Lerndaten ab.

Maple Sheets für die Versuche

Strom Reihen

b1 plain mlr.mws
Dieses Sheet führt die MLR mit verschiedenen Lagstrukturen durch und berechnet die Theil'schen Koeffizienten der verschiedenen Lagstrukturen. Die Daten werden zuerst in zwei disjunkte Mengen, der Trainings- und der Validierungsmenge, unterteilt. Die MLR wird nur auf der Trainingsmenge durchgeführt.

b2 sdif mlr.mws
Dieses Sheet führt die MLR äquivalent zum Sheet b1 plain mlr.mws aus, jedoch erst nach einer saisonalen Differenzierung.

b3 sdif knn.mws
In diesem Sheet werden die Daten saisonal differenziert, auf das Intervall [-0.9; +0.9] skaliert und anschließend als Lerndatensätze für KNN gespeichert.

b3-2 sdif knn re.mws
In diesem Sheet werden die Ausgaben des KNN eingeladen und zurück transformiert. Anschließend wird der Theil'sche Koeffizient berechnet.

b4 sdif pca knn.mws
In diesem Sheet werden die Daten saisonal differenziert, auf das Intervall [-0.9; +0.9] skaliert, durch die PCA transformiert und anschließend als Lerndatensätze für KNN gespeichert.

b4-2 sdif pca knn re.mws
In diesem Sheet werden die Ausgaben des KNN eingeladen und zurück transformiert. Anschließend wird der Theil'sche Koeffizient berechnet.

b5-6 hybrid allpca outliers.mws
In diesem Sheet wird das hybride Modell angewendet. Zuerst wird eine MLR durchgeführt, anschließend mit den Ausgaben der MLR sowie den Trainingsdaten der MLR eine PCA durchgeführt und anschließend als Lerndaten für das KNN gespeichert. Die Fehler der MLR werden auf das Intervall [-0.9;+0.9] skaliert und als Ausgabemuster für das KNN gespeichert.

b5-2 hybrid re outliers.mws
In diesem Sheet werden die Ausgaben des KNN aus dem hybriden Modell eingeladen und zurück transformiert. Anschließend wird der Theil'sche Koeffizient berechnet.

Börsen Indizes

z1 plain mlr.mws
Dieses Sheet führt die MLR mit verschiedenen Lagstrukturen durch und berechnet die Theil'schen Koeffizienten der verschiedenen Lagstrukturen. Die Daten werden zuerst in zwei disjunkte Mengen, der Trainings- und der Validierungsmenge, unterteilt. Die MLR wird nur auf der Trainingsmenge durchgeführt.

z2 dif mlr.mws
Dieses Sheet führt die MLR äquivalent zum Sheet z1 plain mlr.mws aus, jedoch erst nach einer einfachen Differenzierung.

z3 dif knn.mws
In diesem Sheet werden die Daten differenziert, auf das Intervall [-0.9;+0.9] skaliert und anschließend als Lerndatensätze für KNN gespeichert.

z3-2 dif knn re.mws
Hier werden die Ausgaben des KNN aus z3 dif knn.mws zurück transformiert und die Prognosefehler berechnet.

z4 dif pca knn.mws
In diesem Sheet werden die Daten differenziert, auf das Intervall [-0.9;+0.9] skaliert, durch die PCA transformiert und anschließend als Lerndatensätze für KNN gespeichert.

z4-2 dif pca knn re.mws
In diesem Sheet werden die Ausgaben des KNN aus dem Sheet z4 dif pca knn.mws eingeladen und zurück transformiert. Anschließend wird der Theil'sche Koeffizient berechnet.

z5 hybrid.mws
Dieses Sheet implementiert einen alternativen Versuch eines hybriden Modells. Die PCA wir hierbei nicht auf alle Daten angewendet, sondern nur auf diejenigen Daten, die bei der MLR als Eingabe benutzt wurden. Die Ausgaben der MLR werden ohne PCA eingefügt. Die Performanz ist jedoch schlechter als das Modell, bei dem alle Eingabedaten für das KNN PCA transformiert werden.

z5-2 hybrid re.mws
In diesem Sheet werden die Ausgaben des KNN aus dem hybriden Modell z5 hybrid.mws eingeladen und zurück transformiert. Anschließend wird der Theil'sche Koeffizient berechnet.

z5-3 hybrid.mws
Dies ist das hybride Modell, bei dem alle Daten PCA transformiert werden. Siehe auch das Sheet b5-6 hybrid allpca outliers.mws.

z5-4 hybrid re.mws
In diesem Sheet werden die Ausgaben des KNN aus dem hybriden Modell z5-3 hybrid.mws eingeladen und zurück transformiert. Anschließend wird der Theil'sche Koeffizient berechnet.

Maple Sheets für diverse Visualisierungen

01 datensichtung - stromverbrauch multi.mws
In diesem Worksheet werden verschiedene graphische Darstellungen der Stromreihen erzeugt.

01 datensichtung 0001-stromverbrauch brd.mws
In diesem Worksheet werden verschiedene graphische Darstellungen der Reihe StromBRD erzeugt.

01 datensichtung-indizes.mws
In diesem Worksheet werden verschiedene graphische Darstellungen der Börsen Indizes erzeugt.

01b acf 0001.mws
In diesem Sheet wird die ACF der Reihe Strom BRD aufbereitet.

01b korrelation - stromverbrauch multi.mws
In diesem Sheet wird die Korrelationsmatrix der Stromreihnen erzeugt.

02a interpolation 0001.mws
In diesem Sheet wird die Interpolation ausprobiert.

02b transformation 0001.mws
In diesem Sheet werden verschiedene Transformationen für die Reihe StromBRD getestet.

02c outlierdetection.mws
In diesem Sheet wird die Ausreißerbehandlung der Reihe StromBRD visualisiert und angewendet.

03a polyreg strom brd.mws
In diesem Sheet werden Polynome verschiedener Grade auf die Reihe StromBRD angewendet.

03b saisonalität dax.mws
In diesem Sheet werden die verschiedenen saisonalen Darstellungen der Reihe DAX durchgeführt.

03b saisonbereinigung strom brd.mws
In diesem Sheet wird das Phasendurchschnittsverfahren auf der Reihe StromBRD angewendet.

05a ar.mws
In diesem Sheet wird das AR-Modell auf die Reihe StromBRD angewendet.

05b exsmo.mws
In diesem Sheet wird das EWMA-Modell auf die Reihe StromBRD angewendet.

05c exsmo after ar.mws
In diesem Sheet wird das EWMA-Modell auf die Residuen des AR-Modells von der Reihe StromBRD angewendet.

C. Java-Quellkode

TrainNN.java
Um mehrere Tests mit verschiedenen Netzstrukturen zu machen, wurde die Klasse TrainNN.java erstellt, dass Lern- und Validierungsdaten lädt und diese mit verschiedenen Netzstrukturen durchtrainiert. Jede Netzstruktur wird dabei eine vorgegebene Anzahl male initialisiert und trainiert. Dabei werden diejenigen Netze mit den kleinsten Fehlern auf der Validierungsmenge gespeichert. Am Ende wird die Ausgabe des besten Netzes gespeichert.
Die benutzte KNN-Software ist die Java basierte mscJNeuralNet-API, die ich im Verlauf meines Wahlpflichtprojekts erstellte. Details der API können der Javadoc entnommen werden.

EvalNN.java
Mit dieser Klasse können gespeicherte Netze nachträglich mit Daten evaluiert und die Ausgabe gespeichert werden.

Ehrenwörtliche Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Köln, den 20.07.2005

_____

Serhat Cinar