Strategie zur Lösung von Erosionsproblemen unter Verwendung von Luft- und Bodendaten

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften der Universität Dortmund

vorgelegt von

Lars Tschiersch

aus Wanne-Eickel

Dortmund 2002

Erstgutachter: Prof. Dr. W. Urfer
Zweitgutachter: Prof. Dr. W. Krämer

Tag der mündlichen Prüfung: 17. Juli 2002

Danksagung

Ich danke Herrn Prof. Dr. Walter Krämer, der mir im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter an seinem Institut die Promotion ermöglichte. Er begleitete die Arbeit intensiv und gab wertvolle Anregungen.

Herrn Prof. Dr. Wolfgang Urfer danke für viele fruchtbare Diskussionen sowie die herzliche, intensive Betreuung. Durch sein Engagement sind zahlreiche internationale Kontakte entstanden, die Teile der Arbeit erst ermöglichten.

Ich danke Herrn Prof. Dr. Andreas Bürkert, der mir mit wertvollen Ratschlägen über Bodenkunde zur Seite stand und durch dessen Projekte im Niger die in dieser Arbeit verwendeten Daten erst entstanden sind. Ferner danke ich ihm für die Bereitstellung dieser Daten.

Meinem besonderen Dank gilt meinem Kollegen und Freund Herrn Dr. Matthias Zerbst, der mir stets ein sehr guter Partner bei der Arbeit war. Er half mir über manche Durststrecke durch seine Aufmunterung hinweg. Die Grundlagen für diese Arbeit entstanden im Rahmen seiner Dissertation.

Herrn Marco Stolpe danke ich für exzellente Hilfe bei der Implementierung der vorgestellten Verfahren in ein Rechnerprogramm.

Ich danke Herrn Dipl.-Stat. Olaf Schoffer für kritische Anmerkungen, sinnvolle Korrekturvorschläge und hilfreichen Diskussionen.

Ich bedanke mich bei Herrn Peter Reimpell für die hilfreichen Kommentare und Anmerkungen sowie intensives Korrekturlesen.

Mein größter Dank gilt Frau Claudia Reimpell für ihre Geduld und Aufmunterung, die mich über die gesamte Dauer der Arbeit begleiteten. Ebenso für ihre kritischen Anmerkungen und Kommentare.

Inhaltsverzeichnis

Symbolverzeichnis

n	-	Anzahl Bildpunkte des zugrundeliegenden Bildes
q	-	Anzahl Cluster für Clusterung / Dimension der Clusterung
r	-	Anzahl unterschiedlicher Bildpunkte bzgl. ihres Farbwertes des zugrundeliegenden Bildes
P	-	Wert eines Bildpunkts, hier im allgemeinen vektorieller RGB-Farbwert
p_i	-	Wert der i-ten Komponente des Bildpunkts i
C	-	Menge der Cluster der Clusterung
c_i	-	i-ter Cluster der Clusterung
z_i	-	i-ter (Cluster-) Zentroid
Ω	-	Parameterraum, hier im allgemeinen $(0,...,255)^3$
n_i	-	Anzahl (unterschiedlicher) Bildpunkte im Cluster c_i
δ	-	Distanz beim Maximum Linkage
Δ	-	Distanzmatrix beim Maximum Linkage
Q	-	Menge der Indizes der Bildpunkte in der Menge der Zentroide
R	-	Indexmenge der Bildpunkte außerhalb des Clusters der Zentroide
m	-	Dimension der Komponenten eines Bildpunktes P, hier im allgemeinen 3 wegen des RGB-Modells
s	-	aktueller Lernschritt beim Training der Neuronalen Netze
d	-	Nachbarschaftsfunktion für das Training der Neuronalen Netze
$\eta(s)$	-	monoton fallende Lernrate beim Lernschritt s
α	-	Faktor der Konvergenzgeschwindigkeit beim Training der Neuronalen Netze
h_{ij}	-	Gewicht, berechnet aus der Summe der absoluten Häufigkeiten der (unterschiedlichen) Bildpunkte P_i und P_j beim Maximum Linkage
κ	-	Varianzbasierter Parameter beim Maximum Linkage
b	-	Konstante aus dem Intervall $(0,1)$, die die Lernrate beim Training neuronaler Netze definiert
s_{max}	-	Maximale Anzahl an Lernschritten beim Training neuronaler Netze
Z	-	Menge der Zentroide beim Maximum Linkage
k	-	Index der Komponente eines Bildpunktes
ρ	-	Parameter zur prozentualen Berücksichtigung der Häufigkeitsstruktur beim erweiterten Maximum Linkage

Einleitung

In der vorliegenden Arbeit werden Clusterungsverfahren vorgestellt, die speziell für die Clusterung von Luftbildern geeignet sind. Neben der Clusterung wird aufgezeigt, wie Bodendaten aus der Erosionsforschung mit den aus der Bildclusterung gewonnen Informationen vereint werden können. Die in Luftbildern enthaltene Information ist zu umfangreich, um in statistischen Analysen benutzt werden zu können. Durch die Clusterung werden die Informationen auf ein handhabbares Maß reduziert. Der Gedanke zur Clusterung von Bilddaten und der Verbindung mit Bodendaten entstand aus der Analyse von Erosionsdaten aus Marokko (Tschiersch, 1998). Dort wurden nach Regenereignissen der Verlust an fruchtbarem Boden gemessen. Die Meßmethode war allerdings sehr störanfällig, was die Qualität der Daten stark negativ beeinflußte. Auch die Quantität war für eine fundierte Analyse zu gering. Da seitens der marokkanischen Behörden dennoch großes Interesse daran bestand, herauszufinden, wie sich der Bodenverlust beschränken läßt, entstand ein Prozeß aus Forschung und internationaler Kommunikation, der ein vorläufiges Ende mit dieser Arbeit findet. Es sollte die Möglichkeit entstehen, aufgrund von Sekundärinformationen, wie Bewuchs und Bodenbeschaffenheit, die Ausbreitung von Erosion zu verlangsamen. Diese Informationen sollten ergänzt werden um Informationen, die aus Luftaufnahmen gewonnen werden können. Dabei ist ein Untersuchungsgebiet in einem gewissen Zeitraum zu überfliegen, um Luftaufnahmen zu gewinnen. Diese werden durch eine Clusterung auf den zentralen Informationsgehalt reduziert. Als Clusterung bezeichnet man die Einteilung einer Menge von Elementen in homogene Gruppen (Cluster). Die Informationen werden als zusätzliche Variablen in ein Modell mit aufgenommen, welches die lokale Situation bzgl. der Erosion beschreibt. Der Extrapolation von Informationen aus Luftbildern muß dabei besonderes Augenmerk verliehen werden. Eine Luftaufnahme besteht leicht aus mehreren tausend Farben. Diese Farben müssen so in Flächen umgewandelt werden, daß verschiedene Cluster entstehen, die jeweils einen gewissen Erosionsgrad beschreiben. Wünschenswert ist zudem die Möglichkeit, das Ergebnis einer Clusterung visuell darzustellen.

Ausgehend von den klassischen Clusterungsverfahren, die auf dem k-Mittelwert-Algorithmus basieren, sei auch ein neu entwickelter Algorithmus, Maximum Linkage, vorgestellt. Das von Zerbst et al. (2000) entwickelte Verfahren ist dabei besonders zu beachten. Es weist die wesentlichen Nachteile der klassischen Verfahren nicht mehr auf. Unter anderem ist die Laufzeit stark beschleunigt worden und die im Zusammenhang mit der Erosionsforschung wichtigen seltenen Elemente in einem Bild (etwa eine Mauer oder Strauch in der Wüste) verschwinden nicht mehr. Ferner ist die Trennung zwischen den gefundenen Clustern hinreichend groß. Die Weiterentwicklung dieses Verfahrens dahin, daß jetzt auch die Häufigkeitsstruktur der im Bild vorhandenen Farben berücksichtigt wird, ist ein zentrales

Thema dieser Arbeit. Diese Weiterentwicklung hat von mir den Namen Erweitertes Maximum Linkage bekommen. Desweiteren wird ein neues Kriterium zur Beurteilung von Clusterungen vorgestellt, das viele Nachteile bisheriger Kriterien überwindet.
Zum Einordnen der Maximum-Linkage-Verfahren werden darüberhinaus auch neuronale Netze vorgestellt. Diese werden ebenfalls zur Berechnung von Clusterungen herangezogen. Neben den in der Literatur vorgeschlagenen Varianten, werden auch selbst abgestimmte neuronale Netze vorgestellt (vgl. Zerbst, 2001).
Die erhobenen Bodendaten werden mit zeitreihenanalytischen Methoden ausgewertet. Insbesondere in der Modellierung durch ein Zustandsraummodell können zukünftig schnell Online-Prognosen durchgeführt werden. Diese können bei der Planung für eine weitere Datengewinnung hilfreich sein. Die Komplexität des Zeitreihenmodells nimmt durch Hinzunahme von Informationen aus Luftbildern zu. Aus diesen Überlegungen heraus erscheint ein strukturelles Zeitreihenmodell angebracht. Die schlußendliche Zusammenführung von Boden- und Luftdaten ist jedoch noch vorzunehmen. Gleichwohl wird eine Möglichkeit eröffnet, wie dies geschehen kann.

Bevor das Augenmerk auf die statistische Herangehensweise an das Thema Erosion gerichtet wird, sollte dem Leser klar sein, was Erosion bedeutet. Im nachfolgenden Abschnitt sei daher ein kurzer Abriß über die zugrundeliegende Problematik gegeben. Zudem wird auf die Verhältnisse im Untersuchungsgebiet eingegangen.

Erosion

Nicht nur in den dicht besiedelten und industriell entwickelten Ländern hat sich der landwirtschaftliche Produktionsfaktor „Boden“ längst zu einem vielseitig beanspruchten Naturgut entwickelt. Mit steigender Intensität seiner Nutzung kommt dem Schutz des Bodens vor Degradation im allgemeinen, Erosion und Desertifikation im Speziellen, eine besondere Bedeutung zu (Kuntze, 1992). Im folgenden wird das Augenmerk auf Erosion und Desertifikation gelegt. Im Hinblick auf das besondere Untersuchungsgebiet ist die Desertifikation und deren Eingrenzung von essentieller Bedeutung.
Unter den vielen verschiedenen „Arten“ von Erosion hat im Untersuchungsgebiet (wir betrachten eine Studie aus dem Niger) die Winderosion den größten Einfluß, während Erosion durch Niederschlag, ähnlich wie in einer Studie in Algerien (Gomer, 1994), keinen großen Einfluß hat.
Unter Winderosion versteht man das Abtragen und Verfrachten lockerer Bodenteile der Erdoberfläche durch Wind. Erosion ist ein natürliches Phänomen, das aber in zuvor landwirtschaftlich intensiv genutzten Gebieten durch menschliche Einflüsse enorm verstärkt wird. Nach Düwel (1995) ist der Schaden für den Boden durch den

Verlust von Mineralien und Nährstoffen sowie durch eine physikalische Verschlechterung der Oberfläche bedingt. Gerade die physikalische Veränderung sollte besonders berücksichtigt werden, wenn es um die Modellierung von Erosion geht. Der physikalische Prozeß der durch Wind bedingten Erosion ist in mehrere Teilprozesse zu gliedern. Diese sind Ablösung der Bodenpartikel, Transport dieser Partikel sowie die Sedimentation. Die Einzelheiten dieser Prozesse sind in Düwel (1995), Pye (1987) und Breburda (1983) nachzulesen. An dieser Stelle sei nur angemerkt, daß die Oberflächenbeschaffenheit einen großen Einfluß auf die Erosion hat. In seiner Dissertation weist Düwel (1995) auf die Rauheit des Bodens hin und auf die essentielle Bedeutung der Rauhigkeit auf Erosion durch Wind. Dabei ist die Rauhigkeit des Bodens in Anlehung an die internationale Norm ISO 4287/1 (1984) definiert als die Oberflächenunregelmäßigkeit mit relativ kleinen Abständen. Je nach Ausrichtung der Untersuchung spielen dann weitere Faktoren eine Rolle. So ist z. B. zu unterscheiden, ob die Betrachtung des Erosionseinflusses im bodenkundlichen-ackerbaulichen Umfeld oder im meteorologischen Umfeld erfolgt. Eine Klassifikation von verschiedenen Rauhigkeitsstufen ist etwa Davenport (1960) zu entnehmen.
Ein weitere wichtiger Bestandteil der Bodendegradation ist die durch die Erosion mitbedingte Desertifikation. Diese ist nicht nur in warmen südlichen Ländern ein zunehmendes Problem, sondern breitet sich auch nach Europa aus. Spätestens seit der großen Hungerkatastrophe im Sahel Mitte der 70er und Anfang der 80er Jahre steht die Ausbreitung von Trockengebieten im Mittelpunkt öffentlichen Interesses. In Anlehnung an Beugler-Bell (1996) seien hier kurz die Hauptursachen der Ausbreitung von Trockengebieten vorgestellt. Diese sind extreme Klimavariabilität, hohe Erosionsgefährdung, Bevölkerungswachstum, Armut, steigender Nutzungsdruck auf die natürlichen Ressourcen, Vegetationsdegradierung, Bodenerosion, langfristige Abnahme der Regenerationsfähigkeit und Produktivität von Ökosystemen.
Bei dem Begriff der Desertifikation muß unterschieden werden zwischen kurzfristigen Auswirkungen „natürlicher“ Dürreperioden und der irreversiblen Degradation von Trockengebieten unter Einfluß des Menschen (Beugler-Bell, 1996). Desertifikation wird nach Beugler-Bell definiert als:

> *„...langfristige (über mehrere Generationen) irreversible Degradation von ariden, semiariden und subhumiden Ökosystemen über die Auflösung bzw. Beschleunigung der Prozesse der Vegetationsdegradation, Bodenerosion, Sedimentation, Bodenversalzung hervorgerufen durch direkte und indirekte Eingriffe des Menschen in den Landschaftshaushalt.“*

Neben dem komplexen ökologischen Prozeß- und Wirkungsgefüge Mensch/Umwelt kommt dem Boden eine weitere Schlüsselrolle im Landschaftshaushalt zu. Der Boden als Wasserspeicher steuert die Verteilung und Produktivität der Pflanzen. Daher wirken sich Veränderungen in Folge von Bodenerosion und Bodendegradation um so verheerender aus. Das im Untersuchungsgebiet vorherrschende aride Klima, die anhaltende Erosion und die extremen Niederschlagsereignisse tun ein übriges, um einen Selbststärkungseffekt im Hinblick auf die Desertifikation zu unterstützen.

Beugler-Bell (1996) weißt darauf hin, daß trotz wachsender Bemühungen in Industrie- und Entwicklungsländern die meisten bisherigen Lösungsstrategien und Aktionspläne zur Bekämpfung der Desertifikation aus den unterschiedlichsten Gründen gescheitert sind. Unter Desertifikation wird eine Verschlechterung oder Degradierung des Bodens mit damit einhergehender Verringerung der Wasserressourcen verstanden. Dadurch bedingt breiten sich wüstenähnliche Fläche weiter aus oder entstehen neu. Als Ursache für die Desertifikation sind Eingriffe des Menschen in das Ökosystem ausgemacht worden. Neben der Überweidung (zu viele Nutztiere auf zu wenig Bodenfläche) und Abholzung sind aber auch klimatische Veränderungen zu nennen. Beispielsweise hat sich die Wüste Gobi in der Mongolei in den letzten elf Jahren um 350 km nach Süden ausgebreitet. Dies ist eine zunehmende Gefahr auch für das angrenzende China, wo erstmals Beeinträchtigungen für die Hauptstadt Peking festgestellt wurde (Kolonko, 2002). Ferner gilt weiterhin der Aufruf der UN, im Kampf gegen die Desertifikation zusammenzuarbeiten und gemeinsame Anstrengungen zu unternehmen, um dem Problem Herr zu werden. Dabei sei das Augenmerk, gemäß Teil II, Artikel 7 des UNCCD 241/27 (1994)-Aufrufs, auf Afrika zu legen. (UNCCD = United Nations Convention to Combat Desertification; Konvention zur Bekämpfung der Wüstenbildung) Wissenschaftliche Forschungstätigkeit wird aufgrund dieser Konvention durch die UNCCD gefördert.

Auch andere UN-Organisationen weisen auf die Dringlichkeit hin, daß wachsende Problem der Erosion in den Griff zu bekommen. Neueste Schätzungen der UNFPA (United Nation Population Fund; Bevölkerungsentwicklung) prognostizieren für das Jahr 2050 eine Weltbevölkerung von rund 9 Milliarden Menschen. (Tab. 1 zeigt die prognostizierte Bevölkerung in ausgesuchten Ländern.)

Land	Bevölkerung (in Tausend / Jahr)		geschätzte Bevölkerung im Jahr 2050 (in Tausend)
Algerien	27.990	1995	57.731
Äthopien	55.050	1995	169.446
China	1.213.100	1997	1.477.730
Deutschland	82.100	1998	73.303
Indien	935.700	1995	1.528.853
Italien	57.500	1997	41.197
Marokko	27.030	1995	45.434
Niger	9.151	1995	32.029
Nigeria	111.721	1995	244.311
Russland	146.300	1999	121.256
Tunesien	8.896	1995	14.983
Vereinigte Staaten	267.637	1997	349.318

Tab. 1: Prognostizierte Bevölkerung in ausgesuchten Ländern
Quellen: UNFPA (1999) (Prognose) und Meyers (1999) (Bevölkerung heute)

Der Großteil der Bevölkerung wird in nicht entwickelten Ländern leben und enorme Probleme haben an Trinkwasser zu gelangen, weil der Boden nicht mehr als Wasserspeicher dienen kann. Nach Schätzungen der UNFPA (2001) gehen pro Jahr zwischen 5 und 7 Millionen Hektar Ackerland durch Erosion verloren. Dazu kommt in diesen Ländern das exorbitante Wachstum der Bevölkerung. Während nach Schätzungen der UNFPA in den Industrieländern eine Geburtenrate von 0,3 % zu erwarten sei, liegt diese in Asien und Afrika um ein Vielfaches höher. So wird prognostiziert, daß Nordafrika ein Bevölkerungswachstum von 2,0 % haben wird. Das ist neben Westasien der größte Zuwachs (UNFPA (2001)).

Klimatische Bedingungen, Flora und Fauna sowie Wirtschaft in Niger

Niger läßt sich in drei landwirtschaftliche Zonen einteilen. Der Norden des Landes, der fast die Hälfte der Gesamtfläche von Niger einnimmt, befindet sich in der Sahara. Dort herrschen Hochplateaus und Berge vor. Die Vegetation ist spärlich, zumal es hier äußerst selten regnet. Das Zentrum des Landes, besser bekannt unter dem Gebiet Sahel, ist klimatisch semiarid. Ferner wachsen dort nur wenige Gehölze. Der Süden hingegen ist ein fruchtbares, bewaldetes Gebiet. Es profitiert von den ergiebigen Regenfällen (bis zu 800 Millimeter pro Jahr). Die Temperatur in diesem Gebiet liegt bei durchschnittlich 30 °C.

Trotz der großen Trockenheit im Niger wird der Großteil des Einkommens über die Landwirtschaft erzielt. Rund 41 % des Bruttosozialprodukts stammt aus der Landwirtschaft.

Das Kapitel 1 stellt die klassischen Clusterverfahren wie k-Mittelwert-Algorithmus, ISODATA-Methode und die Weiterentwicklungen bis zum PHASE-Ansatz vor. Dazu wird der Maximum-Linkage-Algorithmus eingeführt und die Erweiterung des Ansatzes aufgezeigt. Kapitel 2 gibt eine allgemeine Einführung in die neuronalen Netze und berücksichtigt speziell die im Kontext dieser Arbeit benötigten Verfahren. Die zur Analyse der Bodendaten notwendigen statistischen Methoden der Zeitreihenanalyse, insbesondere das Zustandsraummodell und der Kalman-Filter, werden im Kapitel 3 beschrieben. Die Umsetzung der theoretischen Überlegungen in ein Rechnerprogramm ist Gegenstand von Kapitel 4. Ferner werden dort die Kriterien zur Beurteilung von Clusterungen vorgestellt. Neben dem bekannten Kriterium der Intra-Class-Varianz, wird auch das neue Kriterium der gewichteten Minimalabstände zwischen den Zentroiden einer Clusterung eingeführt. In Kapitel 5 wird die Auswertung der Bodendaten und die gewonnen Clusterungen erläutert. Die verschiedenen Methoden werden gegenübergestellt und die Ergebnisse verglichen. Kapitel 6 gibt schließlich einen zusammenfassenden Überblick über die Ergebnisse und einen Ausblick, welchen Aspekten in Zukunft nachgegangen werden sollte.

1 Clusterverfahren

Explorative Prozeduren können oftmals zum Verständnis von komplexen Beziehungen zwischen multivariaten Daten beitragen. Das Durchsuchen der Daten nach einer Struktur einer „natürlichen“ Einteilung (Clusterung) ist eine explorative Technik. Die Clusterbildung kann bei der Feststellung von Ausreißern und Beziehungen in den Daten hilfreich sein.
Zunächst soll jedoch der Unterschied zwischen den häufig synonym verwendeten Begriffen Clusterung und Klassifikation verdeutlicht werden. Eine Clusterung ist ein einfaches Verfahren, welches keine Annahmen bezüglich der Clusteranzahl oder der Struktur macht. Eine Clusterung erfolgt über Ähnlichkeitsstrukturen. Das bedeutet, daß ein Distanzmaß auf dieser Datenmenge berechnet werden kann. Bei den meisten Verfahren wird die Clusteranzahl vom Anwender spezifiziert. Bei einer Klassifikation hingegen sind die Clusteranzahl sowie die Struktur von vornherein bekannt. Neue Beobachtungen werden einem bestehenden Cluster zugeordnet.
Um die Schwierigkeit einer Clusterung zu demonstrieren, sei das folgende Beispiel betrachtet:

Beispiel
Gegeben seien die 16 Bildkarten eines Skatspiels. Diese Bildkarten sollen in ähnliche Cluster eingeteilt werden. Es ist sofort klar, daß die Unterteilung dieser Karten abhängig ist von der Definition des Begriffs „ähnlich“.
In vielen praktischen Anwendungen des Clusterns kann der Anwender aus Erfahrungswerten die Clusteranzahl vorgeben, um somit zwischen einer „guten“ und einer „schlechten“ Clusterung zu unterscheiden. Warum sollte man dann nicht alle möglichen Clusterungen durchnumerieren und anschließend die Beste auswählen?
Für das Kartenbeispiel, und ganz allgemein, gibt es nur eine Möglichkeit, die Karten in einen Cluster einzuteilen. Für die Einteilung in zwei Cluster gibt es gemäß der Stirling’schen Formel, die die Einteilung von n Elementen in k nicht-leere Cluster berechnet,

$$\frac{1}{k!}\sum_{j=0}^{k}(-1)^{k-j}\binom{k}{j}j^n ,$$

bereits 32.767 Möglichkeiten. Für drei Cluster sind schon $7.141.686$ Möglichkeiten vorhanden (vgl. Johnson und Wichern, 1992).

Wie das Beispiel aufzeigt, ist es aufgrund des Wunsches nach einer Rechenzeitbegrenzung unmöglich, die „beste“ Clusterung (im Sinne eines vorgegeben Kriteriums) aus einer Liste mit allen möglichen Clusterungen auszuwählen. Selbst moderne und schnelle Rechner stoßen dabei schnell an ihre Grenzen. Somit suchen wir nach Algorithmen, welche uns eine gute, aber nicht notwendigerweise die beste Clusterung liefern.
In diesem Kapitel werden im Abschnitt 1.1 die klassischen Verfahren sowie in Abschnitt 1.2 ein neues Verfahren für die Clusterung vorgestellt. Die praktische Umsetzung der Algorithmen erfolgt dann in Kapitel 4.

1.1 Klassischer Ansatz

Ein klassischer Clusterungsansatz beruht auf dem k-Mittelwert-Algorithmus (MacQueen, 1967). Von ihm abgeleitet sind der ISODATA-Ansatz (Tou und Gonzales, 1974), die Weiterentwicklung zur Hyperklumpen-Strategie (Kelly und White, 1993) und der PHASE-Ansatz von Myers, Patil und Taillie (1997a). Diese Ansätze werden im folgenden vorgestellt:
Zunächst führen wir einige Notationen für eine Clusterung ein. Seien $\Omega \subseteq \mathbb{R}^m$ der Merkmalsraum und $O = \{P_1,\ldots,P_n\}$ eine Menge mit $P_i \in \Omega$, $i = 1,\ldots,n$, vektorwertiger Elemente gegeben. Bevor eine Clusterung beginnen kann, hat der Anwender die Clusteranzahl q zu bestimmen. In diese Cluster werden die Daten aufgeteilt; für die Clusteranzahl gilt: $q \ll n$.
Eine Clusterung $C(O)$ ergibt sich als Zerlegung der Menge der Elemente $P_1,\ldots,P_n$, mit $P_i \in \Omega$. Wir bestimmen also:

$$C(O) = \{c_1,\ldots,c_q\} \quad \text{mit } c_i = \{P_{i1},\ldots,P_{in_i}\}, \quad i = 1,\ldots,q \quad , \tag{1.1}$$

wobei die $n_i > 0$, $i = 1,\ldots,n$, damit alle Cluster besetzt sind.
Ferner seien $z_1,\ldots,z_q \in \mathbb{R}^m$ als sogenannte Zentroide definiert. Diese dienen im weiteren Verlauf dazu, die gefundenen Cluster durch ein Element zu beschreiben. Die algorithmische Berechnung der Clusterung erfolgt oftmals mit rekursiven Verfahren. Im Zuge der Rekursion werden vorläufige Zentroide bestimmt. Nach Ablauf der Rekursion werden die letzten Zentroide als Klassenrepräsentanten bezeichnet.
Clusterungen unterliegen gewissen Gütekriterien. So sollen in einem Cluster möglichst ähnliche Elemente zusammengefaßt werden. Man sagt, daß die Cluster homogen sein sollen. Der Unterschied zwischen den Clustern soll jedoch möglichst groß sein. Bock (1998) hat beschrieben, welches Kriterium für eine gute Clusterung hinreichend ist. Dieses wird als Intra-Class-Varianz-Clusterungskriterium bezeichnet und ist nachfolgend dargestellt:

$$g_n(C) := \frac{1}{n} \sum_{i=1}^{q} \sum_{k \in c_i} \left\| P_k - \overline{P}_{c_i} \right\|_2 \to \min_{C \in \Gamma} . \qquad (1.2)$$

In (1.2) bezeichne Γ die Menge aller möglichen Clusterungen der Elemente $P_1,...,P_n \in \Omega$ und

$$\overline{P}_{c_i} = \begin{pmatrix} \overline{P}_{i1} \\ \vdots \\ \overline{P}_{im} \end{pmatrix} = \begin{pmatrix} \frac{1}{n_i} \sum_{j=1}^{n_i} P_{ij1} \\ \vdots \\ \frac{1}{n_i} \sum_{j=1}^{n_i} P_{ijm} \end{pmatrix}, \quad i = 1,...,q , \qquad (1.3)$$

den komponentenweisen Mittelwert über alle Elemente des Clusters c_i. Die für die Praxis häufig gebräuchlichen Darstellungen sind in Bock (1998) oder Zerbst (2001) nachzulesen.

Mit diesen Notationen und der Zusammenfassung für die Berechnungen einer Clusterung betrachten wir kurz die oben angekündigten klassischen Verfahren.

1.1.1 k - Mittelwert - Algorithmus

Der k-Mittelwert-Algorithmus ist iterativ und teilt die Daten in q Cluster dergestalt ein, daß die Cluster möglichst gleiche Varianz aufweisen. Dabei wird die Minimierung der Intra-Class-Varianz nach Bock (1998) berücksichtigt. Mathematisch läßt sich der k-Mittelwert-Algorithmus wie folgt darstellen (vgl. auch Hartigan, 1975):

Sei $\Omega \subseteq \mathbb{R}^m$ der Merkmalsraum der Datenvektoren $P_j, j = 1,...,n$. Eine Clusterung C ist, analog zu Abschnitt 1.1, gegeben als $C = \{c_1,...,c_q\}$. Ferner seien $z_1,...,z_q \in \Omega$ die Zentroide oder Klassenrepräsentanten.

Die Zentroide $z_i^{(0)}, i = 1,...,q$ werden zu Beginn zufällig aus dem Merkmalsraum initialisiert. Die im Superskript angegebene Null steht für die Anzahl der Iterationsschritte. Dieser Wert verändert sich im Laufe des Algorithmus.

Im ersten Schritt des Algorithmus werden die Ausgangsdaten $P_j, j = 1,...,n$, gemäß einer vorgegeben Norm den Zentroiden $z_1,...,z_q$ zugeordnet. Dadurch wird eine erste Einteilung in q Cluster $c_1,...,c_q$ erreicht. Als Norm wird bei vielen praktischen Anwendungen die euklidische Norm gewählt (Hoffman, 1975). Daher wird auch hier auf die euklidische Norm zurückgegriffen. Die Daten werden somit den Clustern gemäß

$$P \in c_{k^*} \Leftrightarrow k^* = \underset{k=1,\ldots,q}{\arg\min} \left\| P - z_k^{(0)} \right\|_2 \tag{1.4}$$

zugeordnet. Innerhalb dieser Cluster c_i befinden sich jetzt jeweils n_i Elemente $P_{i1},\ldots,P_{in_i}$, $i = 1,\ldots,q$. Im zweiten Schritt wird der Mittelwert über die n_i Elemente in den jeweiligen Clustern gebildet. Dieser ergibt sich gemäß Gleichung (1.3). Die gebildeten Mittelwerte werden im nächsten Iterationsschritt als neue Zentroidwerte $z_i^{(1)} := \overline{P}_{c_i}$, $i = 1,\ldots,q$ benutzt. Damit werden die Zentroide durch den multidimensionalen Raum in die Richtung des Datenschwerpunkts bewegt. An dieser Stelle beginnt ein neuer Iterationsschritt und die Daten werden, beginnend mit dem ersten Schritt, erneut klassifiziert.
Dieses Verfahren terminiert, sobald die aktuellen Mittelwerte aus (1.3) mit den Zentroiden, die zum Start des Iterationsschritts benutzt wurden, identisch sind. Die Laufzeit des Algorithmus ist unter Umständen sehr lang, da die Iterationsschrittanzahl hoch sein kann. Die Geschwindigkeit des Algorithmus hängt von der Initialisierung der Startzentroide ab. Nur in Ausnahmefällen terminiert dieses Verfahren nicht. Die dann anzuwendenden Abbruchbedingungen werden bei Bock (1974) vorgestellt.
In der Praxis weist dieser Algorithmus noch einen anderen Schwachpunkt auf. Seltene Elemente, also Elemente mit geringer Häufigkeit, die je nach Anwendung extrem wichtig sein können, werden nicht in eigene Cluster eingeteilt. Diese Elemente werden oft benachbarten Clustern zugeordnet. Nach der Mittelwertbildung zur Bestimmung der neuen Zentroide ist der Einfluß der seltenen Elemente äußerst gering.

1.1.2 Der ISODATA-Ansatz

Der von Tou und Gonzales (1974) eingeführte Ansatz der Iterative Self-Organizing Data Techique A (kurz ISODATA) basiert im wesentlichen auf dem k-Mittelwert-Algorithmus. Ihre Überlegung bestand darin, den k-Mittelwert-Algorithmus nicht bis zur vollständigen Konvergenz durchlaufen zu lassen, sondern, um Rechenzeit einzusparen, diesen vorzeitig abzubrechen. Neben der a priori festgelegten Iterationsanzahl werden in jedem Rechenschritt eine Vielzahl von Parametern überprüft, um eine Abbruchbedingung zu erreichen. Durch diese Parameter kann ein Abbruch vor der maximalen Iterationsanzahl erreicht werden. Die bis zum Abbruch berechnete Clusterung wird als Endergebnis betrachtet.

Mit der Rechnerumsetzung steht 1974 erstmals ein Verfahren zur Verfügung, das in vielen Clusterungsproblemen eine Lösung findet. Neben dem weiterhin bestehenden Problem der extrem langen Laufzeit werden Elemente mit geringer Häufigkeit ebenfalls nicht durch eigene Cluster repräsentiert.

1.1.3 Hyperklumpen-Ansatz

Bei der theoretischen Betrachtung einer Klassifikation kann es vorkommen, daß Werte, die theoretisch einem bestimmten Cluster zuzuordnen sind, praktisch nicht diesem Cluster zugeordnet werden. In diesem Fall spricht man von Mißklassifikation. Diese gilt es möglichst klein zu halten, um eine „gute“ Einteilung der Werte zu erhalten. Eine Mißklassifikation kann u. a. durch äußere Einflüsse hervorgerufen werden. Man stelle sich etwa einen Hügel mit Waldbewuchs vor, der auf einer Seite von der Sonne beschienen wird und dessen andere Seite im Schatten liegt. Aufgrund der durch die Sonneneinstrahlung hervorgerufenen unterschiedlichen Helligkeiten bzw. Farben könnte der Wald in zwei verschiedene Cluster eingeteilt werden. Mittels einer unbeaufsichtigten Klassifikation (Abschnitt 2.2.1) ist dieses Problem nicht unmittelbar zu lösen. Abhilfe kann hier der von Kelly und White (1993) entwickelte Hyperklumpen-Ansatz schaffen.

Der Hyperklumpen-Ansatz verfolgt die Strategie, deutlich mehr Cluster zu bilden als tatsächlich benötigt werden. Kelly und White (1993) schlagen eine Größenordnung von drei- bis viermal soviel Cluster vor. Diese bezeichnen sie als Hyperklumpen. Dabei besteht die Idee des Hyperklumpen-Ansatzes darin, daß die unbeaufsichtigte Clusterung nach einer gewissen Anzahl von Iterationsschritten durch eine beaufsichtigte Clusterung ersetzt wird. Mit andern Worten: der Algorithmus läuft ab wie der k-Mittelwert-Algorithmus, nur daß die gefundenen Cluster erneut geclustert werden. Der zusätzliche Clusterschritt erfolgt nicht mehr unbeaufsichtigt, sondern nach den Vorgaben des Anwenders mit einem beaufsichtigten Verfahren.

Bezogen auf das Waldbeispiel von oben, wird der Wald mittels des beaufsichtigen Clusterungsschritts wieder zu einem Cluster zusammengefaßt.

1.1.4 Der PHASE-Ansatz

Der PHASE-Ansatz (Pixel Hypercluster Approximating Spatial Ensembles) verbessert wiederum den k-Mittelwert-Algorithmus unter Einbeziehung der ISODATA und Hyperklumpenstrategie. Der PHASE-Ansatz maximiert die minimalen euklidischen Abstände zwischen den Zentroiden dergestalt, daß unterscheidbare Cluster gefunden werden. Der von Myers, Patil und Taillie (1997a, 1997b) vorgestellte

Ansatz weist im Vergleich zu der Hyperklumpenstrategie vor allem technische Neuerungen auf, die die praktische Arbeit erleichtern sollen. So stellen die Autoren Methoden der Vor- und Nachverarbeitung der gefunden Clusterung vor, die zudem eine bessere Umsetzung innerhalb eines Rechnerprogramms ermöglichen.
Die erste Innovation besteht darin, daß der Algorithmus nicht mehr direkt auf den Originaldaten arbeitet, sondern daß alle Verarbeitungsschritte auf vom Algorithmus angelegten relationalen Datenbanktabellen durchgeführt werden können. Relationale Datenbanktabelle bedeutet hierbei, daß die Information in eine Haupt- und mehrere, über ein Schlüsselelement verknüpfte Untertabellen zerlegt werden. Über diesen Schlüssel ist die Gesamtinformation immer herstellbar (Schmidt, 1983). Die Form der Informationsspeicherung in relationale Tabellen hat vor allem den Zweck, den Speicherplatz zu minimieren. Jede Information wird nur einmal gespeichert. Betrachtet man ein einfarbiges Rechteck, so muß nur einmal die Farbinformation sowie die Größe der Fläche gespeichert werden. Durch die verbesserte Speicherplatzverwaltung wird zudem der Algorithmus schneller.
Ein weiterer Innovationsschritt besteht in der Einführung eines Startwertzerstreuungsalgorithmus für die Zentroide $z_i^{(0)}$, $i = 1,\ldots,q$. Dabei werden die Zentroide so angeordnet, daß der minimale euklidische Abstand zwischen den Startwerten möglichst groß ist (Myers et al., 1997a). Dadurch wird die Laufzeit bis zur Konvergenz des Algorithmus wesentlich verkürzt. Es wird eine kleinere Anzahl an Iterationsschritten benötigt.
Der dritte Innovationsschritt beschäftigt sich mit der Nachbehandlung der Clusterung. Diese Nachbehandlung soll vor allem zu „kleine“ Cluster vermeiden. Eine Kontrollmöglichkeit die Clustergröße minimal zu halten, ohne die Anzahl der Cluster zu reduzieren, besteht in der dynamischen Clusteraufteilung. Damit eine Mindestgröße an Elementen in einem Cluster gewährleistet ist, werden zu kleine Cluster aufgelöst. Das bedeutet zunächst, daß deren Zentroidwerte gelöscht werden. Die Werte der aufgelösten Cluster, werden klassifiziert und zu den bestehenden Clustern hinzugefügt. Dann werden die gelöschten Zentroidwerte herangezogen, um die Cluster mit den größten Intra-Class-Varianzen in jeweils zwei gleich große Cluster aufzuteilen.

1.1.5 Kritik der vorgestellten Verfahren

Die vorgestellten Verfahren weisen für die in dieser Arbeit vorgestellten Bildverarbeitung Schwächen auf. Das größte Problem betrifft die Laufzeit des entsprechenden Algorithmus bei einer Implementierung in einen Rechner. Eine Luftaufnahme wird z. B. in einzelne Bildpunkte „zerlegt“. Ein Bild kann aus mehr als 100.000 Bildpunkten bestehen. Zu einem Bildpunkt werden neben den Koordinaten auch die

Farbinformationen gespeichert. Diese Farbinformationen werden nicht als einzelner Farbwert gespeichert, sondern immer in der RGB-Darstellung der Farbe. Als RGB-Darstellung bezeichnet man die Aufspaltung des Farbspektrums in die Farben Rot, Grün und Blau. Mit diesen Farben lassen sich alle anderen Farben erzeugen (vgl. Haberäcker, 1995; Anhang B).
Idealerweise sollte ein Algorithmus nur einmal über ein Bild laufen, was bei den obigen Verfahren nicht gegeben ist. Nur durch dieses einmalige Durchlaufen kann die Rechenzeit verringert werden.
Ein weiterer Nachteil der Verfahren besteht in dem Verlust seltener Elemente durch die Mittelwertbildung. Wie bereits erwähnt, sind gerade diese seltenen Elemente für die Erosionsproblematik wichtig. Ein Weg oder eine Mauer in der Wüste kann die Verwüstung stark beeinflussen. Eine Mauer kann z. B. die Winderosion hemmen. Durch die Anwendung des k-Mittelwert-Algorithmus wird die Mauer oftmals nicht erkannt. Auch Vor- oder Nachbearbeitung der Daten, wie beispielsweise im PHASE-Ansatz, können den Verlust der seltenen Elemente nicht verhindern.
Um alle diese Probleme in den Griff zu bekommen, sollte ein geeigneter Algorithmus die angesprochenen Schwachstellen nicht aufweisen. Die Laufzeit des Algorithmus ist so zu gestaltet, daß für die Klassifikation nur wenig Zeit aufgewandt werden muß. Daher darf die Iterationsschrittanzahl nicht zu groß werden. Ebenso sollten die für viele Anwendungen wichtigen seltenen Elemente nicht unterdrückt werden, sondern diese Elemente müssen sogar erkannt werden. Aus diesem Grund ist die Mittelwertbildungen gering zu halten. Idealerweise ist nur eine Mittelwertbildung von Nöten, diese aber erst nach der Clusterung und nicht zur Bildung von Clustern. In diesem Zusammenhang ist es auch wichtig, daß die Zentroide geeignet gewählt werden. Ferner ist eine deutliche Trennung zwischen den Clustern wünschenswert. Das heißt, ein notwendiges Kriterium ist eine möglichst hohe Inter-Class-Varianz. Ein hinreichendes Kriterium wird im Abschnitt 4.3.2 vorgestellt.

1.2 Maximum Linkage

Rückblickend auf Abschnitt 1.1.5 muß ein Algorithmus entwickelt werden, der sowohl schnell zu einer Clusterung kommt als auch hinreichend trennscharf zwischen den gebildeten Clustern ist. Dieser Algorithmus wird nun vorgestellt. Die Idee bei diesem Verfahren – Maximum Linkage – liegt in der für die Problemstellung geeigneten Zentroidauswahl, so daß die Datenwerte mit geringem Zeitaufwand klassifiziert werden können.

1.2.1 Single Linkage

Das Maximum Linkage weist eine gewisse Verwandtschaft mit dem aus der multivariaten Statistik bekannten Verfahren des Single Linkage auf. Das Single Linkage (Johnson und Wichern, 1992) gehört zu der Gruppe der agglomerativen, hierarchischen Clustermethoden. Methoden dieser Art beginnen damit, jeden einzelnen Wert aus der zu clusternden Datenmenge als eigenständiges Cluster anzusehen. Die beiden Datenpunkte, die die größte Ähnlichkeit aufweisen, werden zu einem neuen Cluster zusammengefaßt. Dieses Zusammenfassen kann solange fortgesetzt werden, bis die erforderliche Anzahl an Cluster gebildet ist oder bis nur noch ein Cluster übrig bleibt. Die Resultate der agglomerativen Methode können graphisch in Form eines zweidimensionalen Diagramms dargestellt werden, dem sogenannten Dendrogramm. Mit den „Linkage-Verfahren" können sowohl Datenpunkte als auch Variablen in Cluster eingeteilt werden (Johnson und Wichern, 1992).

Das Single-Linkage-Verfahren betrachtet zunächst alle Datenpunkte als eigenständige Cluster und faßt die beiden Werte, die bezüglich eines Distanzmaßes am nächsten beieinander liegen, zu einem neuen Cluster zusammen. Als Distanzmaß wird in der Regel die euklidische Norm gewählt. Anschließend werden die Abstände bezüglich der euklidischen Norm von den verbliebenen Datenpunkten zu dem ersten entstandenen Cluster gebildet. Dabei berechnet sich der Abstand eines Datenpunkts zu einem Cluster als die Minimaldistanz des Datenpunktes zu den Elementen im Cluster. Erneut werden die Datenpunkte vereinigt, die den kürzesten Abstand zueinander aufweisen. Dabei kann entweder ein weiterer Datenpunkt zu dem bestehenden Cluster hinzugefügt oder aber ein komplett neues Cluster gebildet werden.

Das Single-Linkage-Verfahren läßt sich algorithmisch wie folgt darstellen:

Analog zu Abschnitt 1.1.1 seien der Merkmalsraum mit $\Omega \subseteq \mathbb{R}^m$ und die verschiedenen Werte des Merkmalsraums mit $\boldsymbol{P}_1,\ldots,\boldsymbol{P}_r$ bezeichnet. Seien ferner die zu bildenden Cluster mit $\boldsymbol{C}_j$ bezeichnet und sei durch Δ eine Distanzmatrix definiert, die die Abstände der Datenpunkte enthält. Der Abstand zwischen zwei Cluster $\boldsymbol{C}_i$ und $\boldsymbol{C}_j$ sei mit δ gekennzeichnet.

Es werden zunächst die Abstände zwischen allen Punkten $\boldsymbol{P}_1,\ldots,\boldsymbol{P}_r$ berechnet und diese in die Matrix

$$\Delta = \left(\delta_{ij}\right), \quad \delta_{ij} = \left\|\boldsymbol{P}_i - \boldsymbol{P}_j\right\|_2, \quad i,j = 1,\ldots,r$$

eingetragen.

Das erste zu bildende Cluster C_1 besteht aus den Werten P_{i^*} und P_{j^*} gemäß der Bedingung:

$$P_{i^*}, P_{j^*} \in C_1 \Leftrightarrow \{i^*, j^*\} = \underset{i,j \in \{1,\ldots,r\}, i \neq j}{\arg\min} \delta_{ij} \; .$$

Ist das Minimum nicht eindeutig, wird zufällig einer der existierenden Minimalwerte ausgewählt.
Für die nicht verwendeten Werte gilt:

$$C_j := P_i, \quad j \in \{2,\ldots,r-1\}, \quad i \in \{1,\ldots,r\} \setminus \{i^*, j^*\},$$

d. h. daß diese Werte einelementige Cluster bilden.
An dieser Stelle wird erneut eine Distanzmatrix Δ berechnet. Diese enthält sämtliche Distanzen δ_{ij} zwischen den Mengen C_i und C_j.
Es gilt:

$$\delta_{ij} := d(C_i, C_j) := \min_{P_{i'} \in C_i, P_{j'} \in C_j} \left\| P_{i'} - P_{j'} \right\|_2 .$$

Die Cluster, die das minimale δ_{ij} aufweisen, werden gemäß

$$C_{i^*} \cup C_{j^*} \Leftrightarrow \{i^*, j^*\} = \underset{i,j \in \{1,\ldots,r-2\}}{\arg\min} \delta_{ij}$$

vereinigt.
Die Berechnung der Distanzmatrix für die Cluster und Vereinigung der Mengen wird so lange wiederholt, bis alle Cluster zu einem einzigen Cluster C zusammengefaßt sind. Es ist jedoch möglich, den Algorithmus vorher zu beenden. Die maximale Iterationsanzahl beträgt demnach $r-1$, denn in jedem Iterationsschritt wird die Clusteranzahl um eins verringert.
Zur Verdeutlichung dieses Algorithmus betrachten wir das folgende Beispiel.

1.2.2 Beispiel Single Linkage

Betrachten wir die hypothetischen Abstände zwischen den vier Punkten A, B, C und D. Gegeben sei die Distanzmatrix

$$\Delta_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & & & \\ 9 & 0 & & \\ 3 & 7 & 0 & \\ 6 & 5 & 9 & 0 \end{bmatrix} \end{array}.$$

Hierbei gibt der Subskript von Δ den Schritt im Algorithmus an. Aufgrund der verwandten euklidischen Norm ist es ausreichend, die untere Dreiecksmatrix zu betrachten. Die Hauptdiagonalelemente sind ebenfalls unwichtig, da Elemente nicht mit sich selbst zusammengefaßt werden dürfen. In der obigen Distanzmatrix Δ entspricht der Eintrag δ_{12} dem Abstand zwischen Punkt A und B, δ_{14} dem Abstand zwischen Punkt A und D usw. Mit Anwendung des Single-Linkage- Verfahrens werden die Punkte A und C zusammengefaßt, da zwischen ihnen der kürzeste Abstand (3) besteht. Nach der Neuberechnung der Distanzmatrix hat diese die folgende Gestalt

$$\Delta_2 = \begin{array}{c} \\ AC \\ B \\ D \end{array} \begin{array}{c} \begin{array}{ccc} AC & B & D \end{array} \\ \begin{bmatrix} 0 & & \\ 7 & 0 & \\ 6 & 5 & 0 \end{bmatrix} \end{array}.$$

Dabei entspricht die erste Spalte und die erste Zeile dem neuen Cluster, welches aus den Punkten A und C gebildet wurde. Im nächsten Schritt werden jetzt die Punkte B und D zusammengefaßt, so daß zwei Cluster entstehen. Die Distanzmatrix hat die Form

$$\Delta_3 = \begin{array}{c} \\ AC \\ BD \end{array} \begin{array}{c} \begin{array}{cc} AC & BD \end{array} \\ \begin{bmatrix} 0 & \\ 6 & 0 \end{bmatrix} \end{array}.$$

An dieser Stelle empfiehlt es sich, den Algorithmus abzubrechen, damit nicht alle Punkte in einem Cluster liegen. Das entsprechende Dendrogramm der Clusterung ist Abb. 1 zu entnehmen. Die gestrichelten Linien deuten an, wie das Dendrogramm für eine Clusterung mit einem Cluster weiter zusammengefaßt würde.

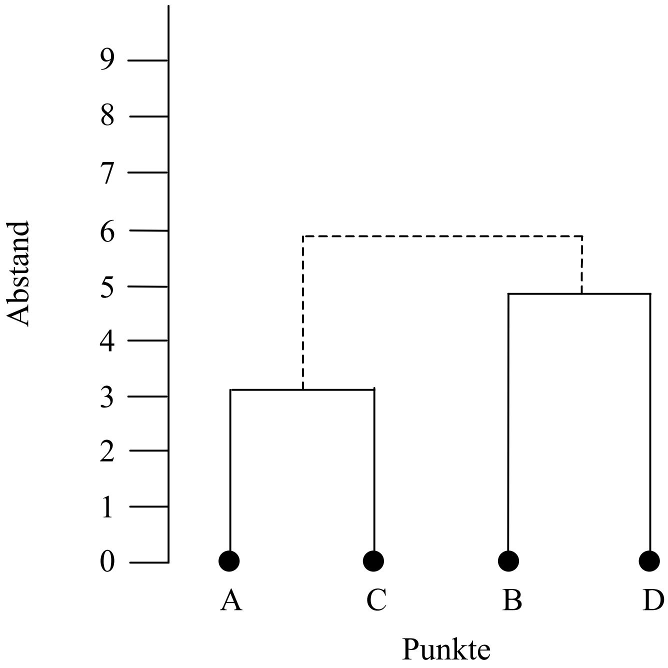


Abb. 1: Single Linkage Dendrogramm (2 Cluster) für die Abstände zwischen den Punkten. Die gestrichelten Linien dienen zur Verdeutlichung wie die Cluster weiter zusammengefaßt werden würden.

1.2.3 Der Maximum-Linkage-Algorithmus

Der Maximum-Linkage-Algorithmus (MLA) weist durch die Berechnungsstruktur eine Verwandtschaft zum Single Linkage auf. Aufgrund der hierarchischen Struktur beim Single Linkage entsteht eine Ordnungsrelation (Anderberg, 1973). Diese ist im Dendrogramm darstellbar. Maximum Linkage hingegen teilt die zu clusternden Werte direkt in eine a priori vorgegebene Clusteranzahl ein.

Der MLA sucht sich aus einer Datenmenge einzelne Werte sukzessive so heraus, daß der Abstand zwischen diesen Werten hinsichtlich einer gegebenen Norm maximal ist. Mit Hilfe dieser Berechnungsgrundlage werden Zentroide gebildet. Diese sind so beschaffen, daß bei einer Klassifikation der übrigen Werte bezüglich dieser Zentroide eine Clusterung mit hoher Trennschärfe entsteht.

Zur späteren Formulierung des Algorithmus seien die folgenden Bezeichnungen eingeführt:

Zusätzlich zu dem aus 1.2.1 bekannten Annahmen über Ω und $P_1,...,P_r$ sei q die Anzahl der zu bildenden Cluster, mit der Bedingung das $q \ll r$, so daß eine Reduktion der unterschiedlichen Werte eintritt. Ferner sei $\{P_{1^*},...,P_{q^*}\}$, $\{1^*,...,q^*\} \subset \{1,...,r\}$ die Menge der ausgewählten Zentroide. Diese Menge dient zur Klassifikation der übrigen Datenpunkte. Die Idee des Verfahrens besteht darin, die Werte von $\{P_{1^*},...,P_{q^*}\}$ so auszuwählen, daß sie der Bedingung

$$\min_{i,j\in\{1^*,...,q^*\},\, i\neq j} \left\| P_i - P_j \right\|_2 \to \max_{\{1^*,...,q^*\}\subset\{1,...,r\}} \tag{1.5}$$

genügen. Hier wird die euklidische Norm verwendet. Im Unterschied zum Single Linkage, wo sukzessive verschiedene Cluster gebildet werden, wird beim Maximum Linkage zunächst eine Menge von Zentroiden berechnet. Anschließend wird aus dieser Menge durch Klassifikation der verbliebenen Datenpunkte die Clusterung gebildet.

Die Bedingung (1.5) wird im weiteren Verlauf Maximum-Linkage-Bedingung genannt. Sie fordert, daß der minimale Abstand zwischen den Zentroiden möglichst groß ist.

Der Ablauf des iterativen Algorithmus läßt sich formal wie folgt beschreiben:

Im ersten Schritt werden die Distanzen δ_{ij} zwischen allen Punkten $P_1,...,P_r$ berechnet. Diese werden in die Distanzmatrix Δ eingetragen:

$$\Delta = (\delta_{ij}),\ \delta_{ij} = \left\| P_i - P_j \right\|_2,\ \ i,j = 1,...,r\ . \tag{1.6}$$

Wie beim Single Linkage ist es auch beim Maximum Linkage ausreichend, die untere Dreiecksmatrix zu betrachten.

Der nächste Schritt besteht im Auffinden des Maximums über alle Distanzen δ_{ij} aus Δ. Die zugehörigen Punkte werden zu der Menge der Zentroidwerte Z zusammengefaßt. Durch diesen Schritt sind bereits zwei der gesuchten Zentroide gefunden. Die eingetragenen Zentroide erfüllen die Bedingung:

$$P_{i^*}, P_{j^*} \in Z \quad \Leftrightarrow \quad \{i^*, j^*\} = \arg\max_{i,j\in\{1,...,r\}} \delta_{ij}\ .$$

Dabei entsprechen P_{i^*}, P_{j^*} den gefundenen ersten beiden Zentroidwerten, die genau dann in den Cluster der Zentroide Z eingetragen werden, wenn der Abstand zwischen ihnen am größten ist. Ist das Maximum nicht eindeutig, so ist eines der möglichen Maxima zufällig auszuwählen und die korrespondierenden Datenpunkte in die Zentroidmenge aufzunehmen.

Anschließend werden alle Distanzen zwischen den Werten in der Zentroidmenge Z und den verbleibenden Werten berechnet und in eine neue Distanzmatrix Δ eingetragen. Diese Matrix hat aufgrund der Konstruktion des Algorithmus die Dimension $(r - 2) \times 2$. Die Dimensionsreduktion geschieht aufgrund der Tatsache, daß an dieser Stelle ausschließlich die Distanzen zwischen den verbliebenen Werten und

dem gebildeten Cluster berechnet werden müssen. Die Anzahl der zu betrachtenden Distanzen verringert sich dadurch von $(r - 1)\cdot(r / 2)$ auf $(r - |Z|)\cdot|Z|$.
Sei somit Q als Indexmenge der Werte definiert, die zur Zentroidmenge gehören, d. h. Q beinhaltet die Werte P_{i^*}, $i^* = 1,...,|Z|$. Weiterhin sei R die Indexmenge der übrigen Werte P_i, so daß gilt $R = \{1,...,r\} \setminus Q$. Die Distanzen δ_{iQ} sind damit gegeben durch den minimalen Abstand zwischen den Punkten:

$$\delta_{iQ} = \min_{k \in Q} \left\| P_i - P_k \right\|_2 , \quad i \in R . \tag{1.7}$$

Das Maximum dieser Distanzen δ_{iQ} wird aufgesucht und der korrespondierende Wert wird zur Zentroidmenge Z hinzugefügt, gemäß

$$Z_{neu} = P_{i^*} \cup Z_{alt} \quad \Leftrightarrow \quad i^* = \arg\max_{i \in R} \delta_{iQ} . \tag{1.8}$$

Es wird also der Index i^* gesucht, welcher den maximalen Distanzwert aufweist.
Die Schritte, beginnend bei der Abstandsberechnung zwischen den Zentroidwerten und den verbleibenden Werten – gemäß Gleichung (1.7) – werden solange wiederholt, bis $|Z| = q$ gilt.
Für die praktische Umsetzung des Algorithmus ist es ferner wichtig, die Häufigkeiten der auftretenden Elemente zu berücksichtigen. Obgleich diese Berücksichtigung Bestandteil des Algorithmus ist, wird diese erst im Abschnitt 4.1 vorgestellt. An dieser Stelle ist es ausreichend, die Funktionsweise des Algorithmus zu betrachten. Zur Veranschaulichung des Algorithmus dient das nachstehende Beispiel.

1.2.4 Beispiel Maximum Linkage

Gegeben sei eine Menge von 10 Punkten, aus denen vier als Zentroide ausgewählt werden sollen. In Tab. 2 sind die Punkte sowie ihre Distanzen gemäß (1.6) zusammengestellt. Die Zentroidmenge Z ist zu Beginn leer.

	A	B	C	D	E	F	G	H	I	J
A	0									
B	3	0								
C	8	3	0							
D	11	6	18	0						
E	4	10	9	21	0					
F	8	7	1	22	16	0				
G	4	11	10	10	19	2	0			
H	9	14	11	12	26	30	12	0		
I	3	4	9	11	13	14	22	28	0	
J	1	4	10	8	19	8	7	2	12	0

Tab. 2: Distanzmatrix zwischen den Punkten A - J. Berechnet gemäß (1.6)

Das Maximum der Abstände ist Tab. 2 zu entnehmen. Es liegt zwischen den Punkten ***H*** und ***F*** und besitzt den Wert **30**. Somit werden die Punkte ***H*** und ***F*** der Zentroidmenge hinzugefügt, also $\boldsymbol{Z} = \{\boldsymbol{H}\,;\boldsymbol{F}\}$.
Der neue Rekursionsschritt beginnt mit der Berechnung der Distanzen zwischen den Elementen der Zentroidmenge und den verbliebenen Werten. Diese gemäß (1.7) berechneten Distanzen werden in die Distanzmatrix eingetragen. Das heißt, das Minimum zwischen den verbliebenen Punkten und den Punkten der Zentroidmenge wird aufgesucht.

Punkte P_i	Z	
	F	H
$Z=\{H, F\}$	-	-
A	8	9
B	7	14
C	1	11
D	22	12
E	16	26
G	2	12
I	14	28
J	8	2

Tab. 3: Distanzvektor
Aufsuchen des Minimums zwischen den Punkten *A,B,C,D,E,G,I,J* und den in der Zentroidmenge befindlichen Punkten *H* und *F*

Die grau hinterlegten Zellen in Tab. 3 geben die Minima gemäß (1.7) an. Das maximale Minimum liegt hier gemäß (1.8) bei $\mathbf{16}$. Der korrespondierende Punkt $\boldsymbol{E}$ wird in die Zentroidmenge eingefügt. Diese lautet somit $\boldsymbol{Z} = \{\boldsymbol{H}\,;\boldsymbol{F}\,;\boldsymbol{E}\}$.
Da vier Zentroide spezifiziert wurden, muß erneut ein Rekursionsschritt durchgeführt werden. Dazu wird abermals die Distanzmatrix zwischen den Punkten der Zentroidmenge und den übrigen Punkten berechnet. Die Matrix hat die Gestalt:

Punkte	Z		
P_i	E	F	H
$Z=\{H,F,E\}$	-	-	-
A	4	8	9
B	10	7	14
C	9	1	11
D	21	22	12
G	19	2	12
I	$\boxed{13}$	14	28
J	19	8	2

Tab. 4: Distanzvektor
Aufsuchen des Minimums zwischen den Punkten *A,B,C, D,G,I,J* und den in der Zentroidmenge befindlichen Punkten *H,F* und *E*

Auch hier sind die Minima wieder grau hinterlegt und das Maximum der Minima eingerahmt dargestellt. Der letzte Punkt, der zur Zentroidmenge hinzugefügt wird, ist $\boldsymbol{I}$. Somit hat die Zentroidmenge die folgende Gestalt:

$$\boldsymbol{Z} = \{\boldsymbol{H}\,;\boldsymbol{F}\,;\boldsymbol{E}\,;\boldsymbol{I}\}.$$

In der Praxis kann es vorkommen, daß das Maximum nicht eindeutig festzulegen ist. Dieses kann insbesondere bei einem beschränkten und vor allem diskreten Parameterraum auftreten. Ist kein Wert eindeutig als Maximum festzulegen, wird zufällig einer der Maximalwerte ausgewählt. Die Auswahl erfolgt mit gleicher Wahrscheinlichkeit. Wie diese Auswahl in der Praxis umzusetzen ist, wird in Abschnitt 4.1.2 beschrieben.

1.2.5 Vorteile des Maximum-Linkage-Algorithmus

Im Vergleich zu den Clustermethoden aus Abschnitt 1.1 sind die Vorteile der Maximum-Linkage-Methode sofort zu erkennen. Um die Anzahl q der Zentroide zu bestimmen, bedarf es nur $(q - 1)$ Iterationsschritte. Im ersten Schritt werden auf Anhieb zwei Zentroide bestimmt. In jedem weiteren Schritt wird ein neuer Zentroid berechnet. Ferner beschränkt sich der Parameterraum auf die tatsächlich vorliegenden Werte, so daß die möglichen Alternativen stark beschränkt werden können. Durch diese Vereinfachung in der Berechnung wird gleichzeitig eine erhebliche Laufzeitreduktion erreicht.
Aufgrund der erwähnten Zentroidberechnung sind die Abweichungen zwischen den Zentroiden groß. Ist dieses nicht der Fall, deutet das auf eine zu große Clusteranzahl q hin. Ein Vergleich der Abweichungen läßt Rückschlüsse zu, ob die Clusteranzahl zu groß gewählt wurde.
Die Art der Berechnung hat einen nützlichen Nebeneffekt. So sind die zur Klassifikation berechneten Zentroide einer Clusterung mit $(q - 1)$ zu bildenden Clustern identisch mit den ersten $(q - 1)$ Zentroiden einer Clusterung mit q zu bildenden Clustern. Das hat zur Folge, daß nach der Berechnung von einer Zentroidmenge nur ein Rechenschritt mehr benötigt wird, um die Zentroide für eine Clusterung mit einem zusätzlichen Cluster zu erhalten. Die bekannten Zentroide können weiterhin benutzt werden. Diesen Vorteil in der Laufzeit besitzen weder ein auf dem k-Mittelwert-Algorithmus basierendes Verfahren noch die neuronalen Netze (Kapitel 2). Mit Hilfe dieser Tatsache ist es denkbar, einen Automatismus für die Clusterung bereitzustellen, der dem Anwender hilft, die richtige Anzahl der Zentroide zu bestimmen. Dieser Ansatz wird in Abschnitt 4.3 verfolgt.

1.3 Erweitertes Maximum Linkage

Bei der Weiterentwicklung des in Abschnitt 1.2.3 beschriebenen Verfahrens wird die Häufigkeitsstruktur der zu clusternden Daten berücksichtigt. Dadurch werden mehr Zentroide aus dicht besetzten Bereichen ausgewählt. Die dünn besetzten Bereiche dürfen auf der anderen Seite jedoch nicht vollständig vernachlässigt werden. In diesen Bereichen liegen häufig Elemente, die aufgrund ihrer exponierten Lage unbedingt eigene Cluster bilden sollten. Unter exponierter Lage versteht man, daß diese Elemente eine große Distanz zu den übrigen Elementen aufweisen.

1.3.1 Der erweiterte Maximum-Linkage-Algorithmus

Der Ablauf des nachstehenden, rekursiven Algorithmus kann kurz wie folgt umrissen werden: Nach einer Initialisierung wird eine Distanzmatrix zwischen allen un-

terschiedlichen zu clusternden Werten berechnet, das Maximum aufgesucht und die resultierenden beiden Werte als Zentroide geführt. Anschließend werden Distanzen zwischen allen bisher als Zentroide berücksichtigten Elementen und allen übrigen Elementen berechnet. Gemäß einer vorgegebenen Gewichtsfunktion werden die jeweiligen Minima bestimmt. Das Maximum dieser Minima wird aufgesucht und der korrespondierende Wert als Zentroid geführt. Dieser Schritt wird solange wiederholt, bis die gewünschte Anzahl an Zentroiden bestimmt ist.
Für die formale Einführung des erweiterten Maximum-Linkage-Algorithmus (EMLA) seien zusätzlich zu den in 1.2.3 gemachten Annahmen die absoluten Häufigkeiten $h_1,...,h_r$ zu den $P_1,...,P_r$ definiert. Somit sind von den n zu clusternden Werten r verschieden, $r < n$. Sei ρ ein vom Anwender zu spezifizierender Parameter. Dieser gibt den berücksichtigten prozentualen Anteil ($\rho \cdot 100\%$) der Häufigkeitsverteilung der Ausgangsdaten an. Für diesen Parameter gilt: $\rho \in [0,1]$. Dabei wird die Häufigkeitsverteilung bei $\rho = 0$ nicht und bei $\rho = 1$ maximal berücksichtigt. Ferner bezeichne s den aktuellen Iterationsschritt.
Weiter sei die Häufigkeit für das Auftreten von P_i und P_j definiert als λ_{ij} gemäß

$$\lambda_{ij} = \frac{h_i \cdot h_j}{n^2},$$

mit $i, j = 1,...,r,\ i \neq j$.

Sei weiter die Häufigkeit für das Auftreten von P_i gegeben durch:

$$\lambda_i = \frac{h_i}{n}, \quad i \in \{1,...,r\}.$$

Zur Berücksichtigung der Häufigkeitsverteilung sei die Gewichtsfunktion f gemäß

$$f(\rho, \lambda) = \rho\ \lambda + (1 - \rho), \tag{1.9}$$

mit $\rho \in [0, 1]$ und

$$\lambda = \begin{cases} \lambda_{ij}, & \text{falls} \quad s = 1 \\ \lambda_i, & \text{falls} \quad s > 1 \end{cases}$$

definiert. Dabei gibt der Parameter ρ den Grad der berücksichtigten Häufigkeiten an. Der Parameter λ hingegen beinhaltet je nach Iterationsschritt s des Algorithmus den Wert λ_{ij} oder λ_i. Der Grund für diese Einteilung ist Abschnitt 1.3.2 zu entnehmen.

Das Verhalten der Funktion f kann exemplarisch der Tab. 5 entnommen werden. Dabei ist zu sehen, daß sich (1.9) zur Gewichtung mit der Häufigkeit in Abhängigkeit von Parameter ρ eignet. Je größer ρ gewählt wird, desto stärker fällt die Gewichtung für die λ_{ij} , $i, j \in \{1,...,r\}$ aus.

ρ	$f(\rho, \lambda_{ij})$	Intervall für f
0	1	[1 , 1]
¼	¼ λ_{ij} + ¾	[¾ , 1]
½	½ λ_{ij} + ½	[½ , 1]
¾	¾ λ_{ij} + ¼	[¼ , 1]
1	λ_{ij}	$\left[\min\limits_{i,j=1,...,r} \lambda_{ij} \;, 1\right]$

Tab. 5: Wertetabelle der Funktion f für ausgewählte Werte von ρ im Iterationsschritt $s=1$.

Mit diesen Definitionen gestaltet sich der Algorithmus wie folgt:
Der Anwender hat zur Initialisierung die Parameterwerte ρ und q festzulegen. Im ersten Berechnungsschritt wird die Matrix Δ berechnet. Diese enthält die mit der Funktion f aus (1.9) gewichteten Distanzen zwischen allen r unterschiedlichen Werten $P_1,...,P_r$. Die Matrix Δ ergibt sich zu:

$$\Delta = (\delta_{ij})_{i,j=1,...,r}, \quad \text{mit } \delta_{ij} = f(\rho, \lambda_{ij}) \cdot \| P_i - P_j \|_2 . \tag{1.10}$$

Das Maximum aus (1.10) wird aufgesucht. Die korrespondierenden zwei Ausgangswerte werden als die ersten zwei Zentroide ausgewählt. Die Menge der als Zentroide ausgezeichneten Werte sei mit $Z^{(s)}$ bezeichnet, wobei der Superskriptindex sich auf den Iterationsschritt bezieht. Somit ergibt sich die Menge der Zentroide des ersten Iterationsschritt gemäß

$$Z^{(1)} = \{ P_{i^*}, P_{j^*} \} \quad \Leftrightarrow \quad \{ i^*, j^* \} = \underset{i,j \in \{1,...,r\}}{\arg\max}\, \delta_{ij} . \tag{1.11}$$

Im zweiten Schritt werden erneut Distanzen berechnet. Dazu bezeichnen $R^{(s)}$ und $Q^{(s)}$ vom Iterationsschritt s abhängige Indexmengen. Die Indizes aller im Iterationsschritt s bereits als Zentroide ausgezeichneten Werte P_{i^*} , $i^* \in \{1^*,...,(s+1)^*\}$, seien in $Q^{(s)}$ enthalten: $i \in Q^{(s)} \quad \Leftrightarrow \quad P_i \in Z^{(s)}$.
Die Indizes der Werte P_j, $j \in \{1,...,r\} \setminus \{1^*,...,(s+1)^*\}$, die nicht als Zentroide feststehen, seien in $R^{(s)}$ enthalten: $R^{(s)} = \{1,...,r\} \setminus Q^{(s)}$. Die Distanzberechnung wird

nun zwischen allen Werten mit Index aus $\boldsymbol{R}^{(s)}$ und allen Werten mit Index aus $\boldsymbol{Q}^{(s)}$ durchgeführt. Diese Distanzen werden anschließend mit (1.9) gewichtet und das jeweilige Minimum für alle $i \in \boldsymbol{R}^{(s)}$ in einen Distanzvektor $\delta_{iQ}, i \in \boldsymbol{R}^{(s)}$ eingetragen. Es gilt:

$$\delta_{iQ} = f(\rho,\lambda_i)\min_{k\in Q}\|P_i - P_k\|_2, \quad i \in \boldsymbol{R}^{(s)} . \tag{1.12}$$

Das Maximum aus (1.12) wird aufgesucht und der zugehörige Ausgangswert in die Zentroidmenge aufgenommen, so das gilt:

$$Z^{(s+1)} = Z^{(s)} \cup \{ P_{i^*} \} \quad \Leftrightarrow \quad i^* = \arg\max_{i \in R^{(s)}} \delta_{iQ} . \tag{1.13}$$

Der zweite Schritt wird solange wiederholt, bis $|Z| = q$. Insgesamt wird dieser Schritt $(q-2)$-mal durchlaufen.
Im dritten Schritt werden die unterschiedlichen zu clusternden Werte $P_1,...,P_r$ aufgrund der ermittelten Zentroide P_{i^*}, $i^* = 1,...,q$ klassifiziert. Es entsteht eine Clusterung. Anschließend werden, unter Berücksichtigung der Häufigkeit, in den einzelnen Clustern Mittelwerte über die enthaltenen Werte gebildet. Diese so erhaltenen Werte werden fortan nicht mehr als Zentroide, sondern als Klassenrepräsentanten bezeichnet. Die Klassenrepräsentanten werden zur Visualisierung verwendet, vgl. Abschnitt 5.2.3.

1.3.2 Art der Gewichtung

Warum wird in (1.10) λ_{ij} und in (1.12) λ_i zur Beschreibung der Häufigkeitsstruktur verwendet? Generell werden die Häufigkeiten der Zentroidskandidaten berücksichtigt. Da in (1.10) zwei Kandidaten beurteilt werden, wird λ_{ij} verwendet. Durch (1.12) wird jeweils nur ein Kandidat beurteilt, so daß aus diesem Grund die Häufigkeit λ_i berücksichtigt wird. Darüber hinaus gibt es einen weiteren entscheidenden Grund, in (1.12) nicht λ_{ij} zu verwenden. Dazu sei folgendes Beispiel betrachtet.

Beispiel

Gegeben seien vier Punkte mit unterschiedlichen Häufigkeiten. Gesucht seien drei Zentroide. Die nachstehende Abb. 2 verdeutlicht die Sachlage. Zwei der gesuchten

Zentroide seien bereits bestimmt (Z_1 und Z_2). Entscheidend ist das Auffinden des dritten Zentroids. Dazu stehen zwei Kandidaten (K_1 und K_2) bereit.

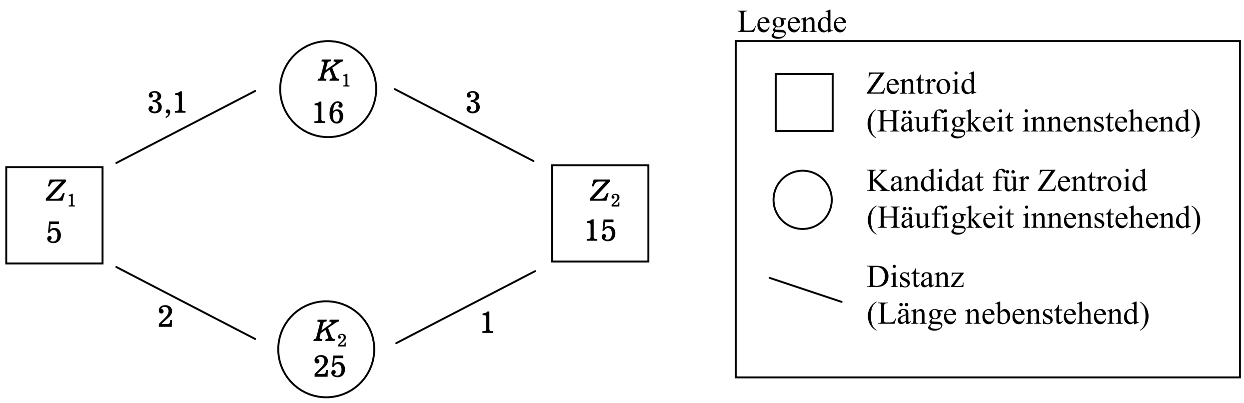


Abb. 2: Darstellung zweier ausgewählter Zentroide und zweier als Zentroid geeignete Kandidaten für den EMLA.

Die Gesamtsumme aller angegebenen Häufigkeiten beträgt 61. Für den Distanzvektor δ_{iQ} ergibt sich, unter Verwendung von (1.10):

	Z₁	**Z₂**
K_1	$16/61 \cdot 5/61 \cdot 3{,}1 = 0{,}0666$	$16/61 \cdot 15/61 \cdot 3 = 0{,}1935$
K_2	$5/61 \cdot 25/61 \cdot 2 = 0{,}0672$	$25/61 \cdot 15/61 \cdot 1 = 0{,}1008$

Tab. 6: Berechnung des Abstandes zwischen den Zentroiden und den Kandidaten unter Verwendung von (1.10).

In der Spalte Z_1 befinden sich die Minima der jeweiligen Zeile. Bei der anschließenden Maximumsbildung wird K_2 aus Spalte Z_1 ausgewählt. Dieser Kandidat ist jedoch nicht geeignet, denn selbst bei der Berücksichtigung der Häufigkeiten bleibt eine maximierte Minimaldistanz eines der wesentlichen Kriterien bei der Clusterung. Aufgrund der Verwendung von λ_{ij} wird diese maximierte Minimaldistanz nicht hinreichend beachtet. Betrachtet man die Minimaldistanz von K_2, so beträgt diese 1. Der Kandidat K_1 hingegen weist eine Minimaldistanz von 3 auf. Dadurch scheidet K_2 als Zentroid aus.
Bei der Verwendung von λ_i tritt diese Problem nicht mehr auf. Denn in (1.12) wird die Häufigkeit des schon feststehenden Zentroids nicht mehr verwendet. Tab. 7 zeigt die Ergebnisse unter Verwendung von (1.12).

	Z_1	Z_2
K_1	$16/61 \cdot 3{,}1 = 0{,}8131$	$25/61 \cdot 2 = 0{,}8197$
K_2	$16/61 \cdot 3 = 0{,}7869$	$25/61 \cdot 1 = 0{,}4098$

Tab. 7: Berechnung des Abstandes zwischen den Zentroiden und den Kandidaten unter Verwendung von (1.12).

Bildet man das Maximum über die zeilenweisen Minimalwerte der Distanzen, also $\max\{0{,}8131;\, 0{,}4098\}$, wird K_1 ausgewählt. Dabei wird zunächst das Minimum unabhängig von den Häufigkeiten bestimmt. Erst danach ist es sinnvoll, die Häufigkeiten zu berücksichtigen. Dahinter steht die Idee, gegebenenfalls nicht das absolute Maximum zu wählen, wenn dadurch eine erheblich größere Wertmenge (größere Häufigkeit) repräsentiert wird. Somit kann bei diesem Vorgehen die Häufigkeit der bereits als Zentroide ausgewählten Werte nicht berücksichtigt werden.

1.3.3 Beispiel Erweiterter Maximum-Linkage-Algorithmus

Zur Verdeutlichung des Algorithmus sei nachstehendes Beispiel betrachtet. Gegeben seien die RGB-Vektoren

$$a = \begin{pmatrix} 1 \\ 3 \\ 4 \end{pmatrix} \quad b = \begin{pmatrix} 8 \\ 20 \\ 15 \end{pmatrix} \quad c = \begin{pmatrix} 70 \\ 10 \\ 90 \end{pmatrix} \quad d = \begin{pmatrix} 7 \\ 100 \\ 4 \end{pmatrix} \quad e = \begin{pmatrix} 100 \\ 90 \\ 0 \end{pmatrix}.$$

Die Vektoren a) bis e) treten mit den Häufigkeiten

$$h_a = h_1 = 3,\ h_b = h_2 = 5,\ h_c = h_3 = 1,\ h_d = h_4 = 10,\ h_e = h_5 = 1$$

auf. Somit gibt es $r = 5$ unterschiedliche Vektoren. Ferner sei: $\rho = 0{,}5$. Gesucht sind $q = 3$ Cluster.

Zur Verdeutlichung seien die RGB-Vektoren in Abb. 3 dargestellt.

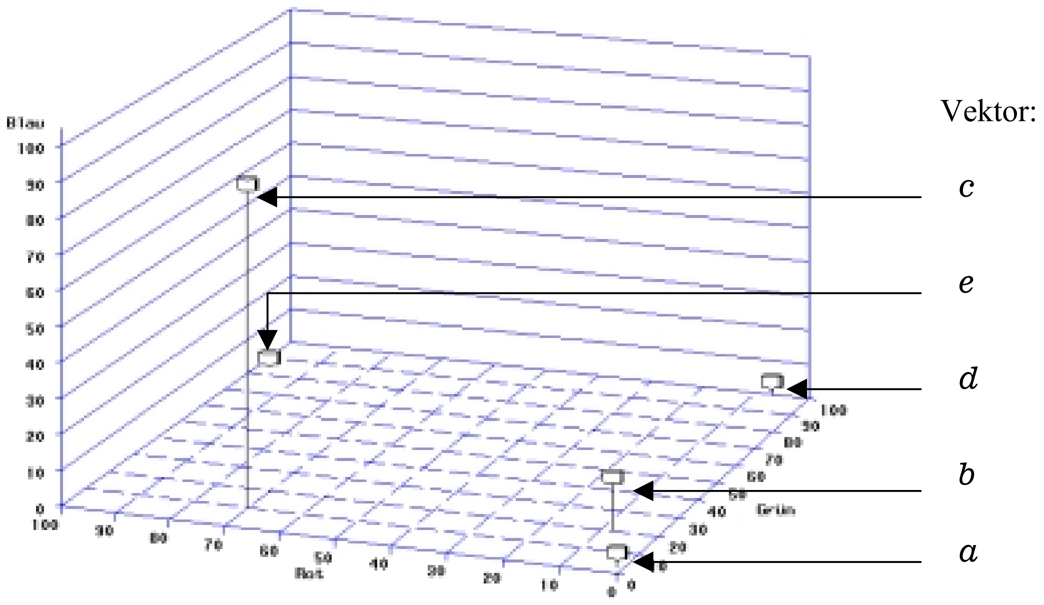


Abb. 3: Visualisierung der RGB-Vektoren a bis e.

Für die Häufigkeiten λ_{ij} bzw. für die Gewichtsfunktion f ergeben sich:

$i \backslash j$	**1**	**2**	**3**	**4**	**5**
1	0,0225				
2	0,0375	0,0625			
3	0,0075	0,0125	0,0025		
4	0,0750	0,1250	0,0250	0,250	
5	0,0075	0,0125	0,0025	0,025	0,0025

Tab. 8: Werte für λ_{ij} .

i \ j	**1**	**2**	**3**	**4**	**5**
1	0,51125				
2	0,51875	053125			
3	0,50375	0,50625	0,50125		
4	0,53750	0,56250	0,51250	0,6250	
5	0,50375	0,50625	0,50125	0,5125	0,50125

Tab. 9: Werte für die Gewichtsfunktion f aus (1.9).

Im ersten Algorithmusschritt wird die Distanzmatrix Δ berechnet. Gemäß (1.10) ergibt sich die Matrix

i \ j	1	2	3	4	5
1	0,0				
2	11,114	0,0			
3	55,655	49,522	0,0		
4	52,237	45,427	71,502	0,0	
5	66,422	59,015	62,203	47,981	0,0

Tab. 10: Distanzmatrix Δ aus (1.10).

Das Maximum der Einträge aus Δ liegt bei $71{,}502$. Die korrespondierenden Punkte sind die Vektoren c und d. Diese beiden Vektoren sind die ersten beiden Zentroide, d. h.

$$Z^{(1)} = \{c,\, d\}.$$

Somit ergibt sich für $R^{(1)}$ bzw. $Q^{(1)}$:

$$R^{(1)} = \{3, 4\} \text{ bzw. } Q^{(1)} = \{1, 2, 5\}.$$

Für den Distanzvektor δ_{iQ} aus (1.12) ergibt sich:

$$\delta_{iQ} = \begin{pmatrix} 14{,}578 \\ 20{,}190 \\ 4{,}038 \end{pmatrix}.$$

Das Maximum der Distanzen liegt bei $20{,}190$. Diese ist der gewichtete Abstand von Vektor b) zum Zentroid $Z^{(1)}$. Somit ergibt sich die schlußendlich bestimmte Zentroidmenge als:

$$Z = \{b, c, d\}.$$

2 Künstliche neuronale Netze

Künstliche neuronale Netze sind in Anlehnung an das biologische Vorbild entwikkelt worden. Bevor man sich mit künstlichen neuronalen Netzen beschäftigt, ist es interessant, dieses biologische Vorbild zu betrachten. Bei Lebewesen besteht dieses Vorbild aus Nervenzellen, den sogenannten *Neuronen*, und Zellverbänden. Die Neuronen haben Eingänge (Dendriten) und Ausgänge (Axonen), mit denen sie untereinander verknüpft sind. Die Information, die ein Neuron von seinen Eingängen bekommt, wird auf eine bestimmte Art verarbeitet. Bei der Information handelt es sich um einen elektrischen Impuls, aus dem der Zustand des Neurons bestimmt wird. Das Neuron befindet sich entweder im angeregten (aktiven) oder im nicht angeregten (passiven) Zustand. Mittels der Ausgänge wird dieser Zustand an die anderen Neuronen weitergegeben.
Abb. 4 zeigt zwei miteinander verbundene Nervenzellen der Hirnrinde (Neokortex). Die Dendriten summieren die Ausgabesignale der umgebenen Neuronen und leiten diese dem Zellkörper (Soma) zu. Überschreitet die Summe der elektrischen Impulse einen Schwellenwert, erzeugt der Soma einen Impuls (die Zelle „feuert“). Die Kontaktstellen eines Axons befinden sich auf dem Soma des Zielneurons und werden als Synapsen (später: Gewichte) bezeichnet.

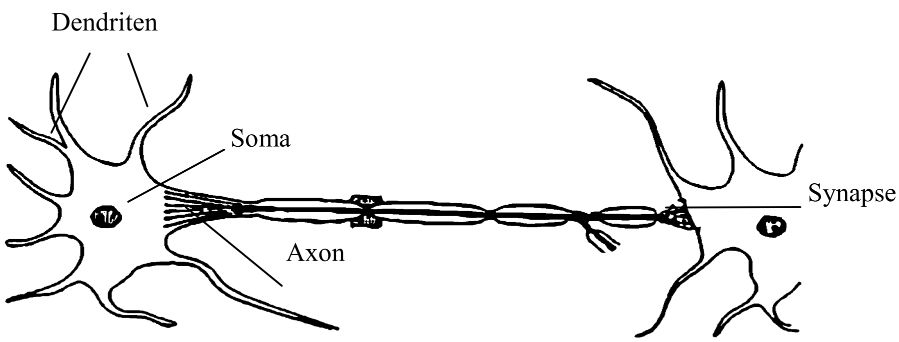


Abb. 4: Schematische Darstellung zweier verbundener Neuronen. Die Eingänge (Dendriten) sind über der Soma mit den Ausgängen (Axonen) verbunden. Am Soma des zweiten Neurons (rechts) befindet sich die Kontaktstelle der beiden Neuronen, die Synapse.

Das biologische Vorbild für die künstlichen neuronalen Netze ist sehr kompliziert. Die technische Umsetzung muß Vieles stark vereinfachen. Nur diese vereinfachten Modelle sind derzeit einer genaueren theoretischen Analyse zugänglich.
Künstliche neuronale Netze (KNN) sind informationsverarbeitende Systeme, die aus einer großen Zahl einfacher Einheiten (Neuronen) bestehen und die Informationen massiv parallel verarbeiten. Nach Zell (2000) ist ein wesentliches Element der KNN ihre Lernfähigkeit. Darunter versteht man die Fähigkeit selbständig aus Trainingsbeispielen zu lernen, ohne daß das neuronale Netz dazu explizit program-

miert werden muß. Die Lösung eines Klassifikationsproblems ist somit mit einem neuronalen Netz möglich. Die wesentlichen positiven Eigenschaften neuronaler Netze sind die Lernfähigkeit und die massive Parallelität. Darüber hinaus sind bei fast allen neuronalen Modellen die Informationen in den Gewichten verteilt. Zum einen wird erst damit die parallele Verarbeitung ermöglicht, zum anderen bewirkt es eine höhere Fehlertoleranz des Gesamtsystems gegenüber dem Ausfall einzelner Neuronen. Die höhere Fehlertoleranz gilt jedoch nur, wenn diese beim Entwurf des Systems mit berücksichtigt wurde (z. B. bei der Dimensionierung des Netzes). Ferner reagieren neuronale Netze bei richtigem Training bei verrauschten Daten oder Störungen in den Eingabemustern in der Regel weniger empfindlich als konventionelle Algorithmen.

Zu den Nachteilen neuronaler Netze zählen, daß der Wissenserwerb ausschließlich durch Training möglich ist sowie die langsame Geschwindigkeit des Lernens.

Bei einer verteilten Repräsentation ist es schwer, einem neuronalen Netz ein gewisses Basiswissen mitzugeben. Bis auf wenige Ausnahmen geschieht Wissenserwerb ausschließlich über Lernen. Die Geschwindigkeit, mit der dieses Lernen umgesetzt werden kann, ist besonders bei vollständig ebenenweise verbunden Netzwerken sehr langsam. Ebenenweise bedeutet hier, daß jedes Neuron einer Ebene mit allen Neuronen der nächsten Ebene verbunden ist. Das bekannteste derartige Verfahren ist der Backpropagation-Algorithmus.

2.1 Kurze geschichtliche Darstellung der neuronalen Netze

Die Erforschung künstlicher neuronaler Netze ist beinahe ebenso alt wie die elektronische Datenverarbeitung. Bereits im Jahre 1943 beschrieben Warren McCulloch und Walter Pitts neuronale Netzwerke basiertend auf dem McCulloch-Pitts-Neuron (McCulloch und Pitts, 1943). Sie zeigten, daß einfache neuronale Netze jede arithmetische oder logische Funktion berechnen können. Sechs Jahre später entwicklete Donald Hebb die klassische Hebb'sche Lernregel als einfaches Lernkonzept individueller Neuronen. In ihrer allgemeinen Form ist die Hebb'sche Lernregel bis heute Basis fast aller neuronalen Lernverfahren. Hebb (1949) erkannte, daß, wenn ein aktives Neuron eine Eingabe von einem anderen aktiven Neuron erhält, sich die Verbindung zwischen den Neuronen stärkt. Mit anderen Worten weisen aktive Neuronenverbindungen ein höheres Gewicht auf.

Der erste erfolgreiche Neurocomputer (Mark I Perceptron) wurde in den Jahren 1957-1959 am Massachusetts Institute of Technology (MIT) entwickelt und für Mustererkennungsprobleme eingesetzt (Rosenblatt, 1958).

Bernard Widrow und Marcian E. Hoff stellten in ihrer Monographie das Adaline vor. Das Adaline ist ein adaptives System, das schnell und genau lernen kann. Ähn-

lich wie das Perzeptron ist es ein binäres Schwellenwert-Neuron. (Widow und Hoff, 1960).
Im Jahre 1969 zeigten Marvin Minsky und Seymour Papert, daß das Perzeptron viele wichtige Probleme nicht lösen kann. Mittels des einfachen XOR-Problems (siehe unten) konnten sie zeigen, daß das ursprüngliche Perzeptron aus prinzipiellen Gründen gewisse einfache Funktionsklassen nicht darstellen konnte. Ihre Schlußfolgerung, daß damit auch komplexere Modelle als das Perzeptron die gleichen Probleme aufweisen, konnte nicht nachgewiesen werden.
Erst ab Mitte der achtziger Jahre nahm das Interesse wieder zu. Mit der Veröffentlichung des Backpropagationansatzes für mehrlagige Netze durch Rumelhart (1986) setzte eine Forschungsentwicklung ein, die bis heute anhält. Dieser Ansatz konnte die von Minsky und Papert beschriebenen Einschränkungen überwinden.

XOR-Problem
Binäre Verknüpfungen spielen eine wichtige Rolle zum algorithmischen Lösen von Rechenproblemen. Neben den bekannten Verknüpfungstypen UND und ODER gibt es auch eine sogenannte „Exclusive-Oder"-Verknüpfung (XOR).
Gegeben seien zwei Eingänge und ein Ausgang. Dann läßt sich das XOR-Problem wie folgt beschreiben:
An zwei Eingängen liegen je ein binärer Wert 0 oder 1 an. Die Ausgabe einer XOR-Verknüpfung liefert eins, falls genau einer der Eingabewerte eins gesetzt ist, sonst null. Die Wahrheitstabelle für eine XOR-Verknüpfung sowie für eine UND- und ODER-Verknüpfung ist in Tab. 11 gegeben.

x_1	x_2	x_1 UND x_2	x_1ODER x_2	x_1 XOR x_2
0	0	0	0	0
1	0	0	1	1
0	1	0	1	1
1	1	1	1	0

Tab. 11: Wahrheitstabelle der UND-, ODER- und XOR-Verknüpfung

Ein Perzeptron (vgl. Abschnitt 2.3.1) kann alle Funktionen berechnen, bei denen die Zweiteilung des Eingaberaumes durch eine Gerade erreicht werden kann. Damit wird die Menge der Ausgaben mit dem Wert eins von denen mit dem Wert null getrennt. Bei der UND- und ODER-Verknüpfung ist diese Trennung möglich, wie Abb. 5 und Abb. 6 verdeutlichen.

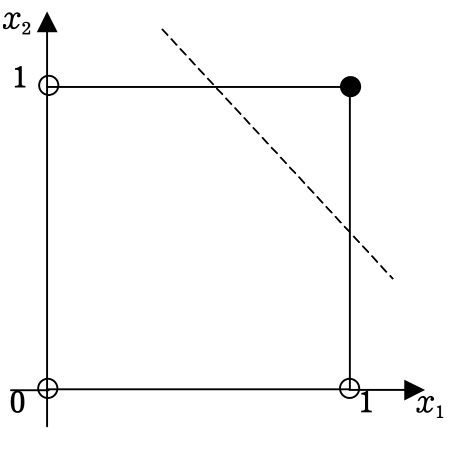


Abb. 5: UND-Verknüpfung. Die schwarzen Kreise repräsentieren eine Ausgabe von eins; die weißen eine Ausgabe von null.

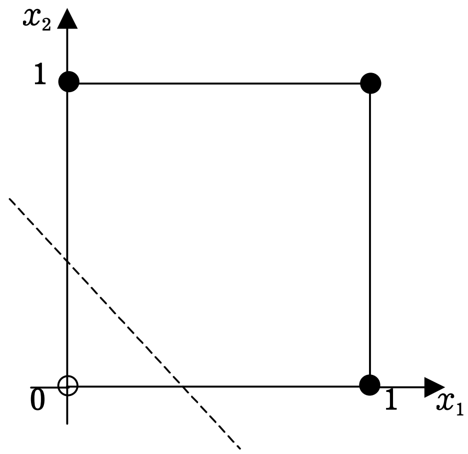


Abb. 6: ODER-Verknüpfung. Die schwarzen Kreise repräsentieren eine Ausgabe von eins; die weißen eine Ausgabe von null.

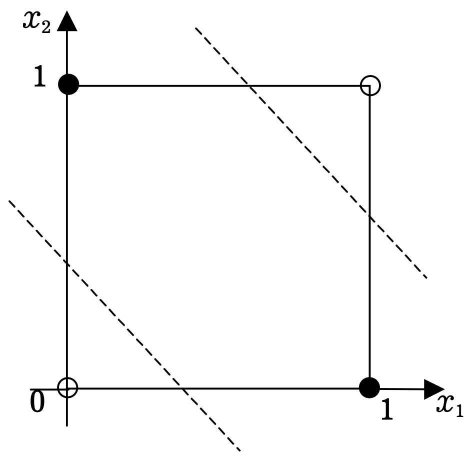


Abb. 7: XOR-Verknüpfung. Die schwarzen Kreise repräsentieren eine Ausgabe von eins; die weißen eine Ausgabe von null.

Abb. 7 hingegen zeigt das XOR-Problem. Das Perzeptron kann keine einzelne Gerade finden, die die Ausgabe „1“ (schwarze Kreise) von der Ausgabe „0“ (weiße Kreise) trennt.

Behauptung:
Die XOR-Funktion von zwei boolschen Variablen x_1, x_2 kann nicht mit einem Perzeptron berechnet werden.

Begründung: Es seien die Gewichte W_1 und W_2 der zwei Eingabeleitungen zu einem Perzeptron gegeben. Ferner sei θ der Schwellenwert, bei dem das Perzeptron „schaltet". Damit das XOR-Problem zu lösen ist, müssen die folgenden Ungleichungen erfüllt sein:

$$x_1 = 0, x_2 = 0: \; W_1 x_1 + W_2 x_2 = 0 < \theta \tag{2.1}$$

$$x_1 = 1, x_2 = 0: \; W_1 x_1 + W_2 x_2 = W_1 \geq \theta \tag{2.2}$$

$$x_1 = 0, x_2 = 1: \; W_1 x_1 + W_2 x_2 = W_2 \geq \theta \tag{2.3}$$

$$x_1 = 1, x_2 = 1: \; W_1 x_1 + W_2 x_2 = W_1 + W_2 < \theta \tag{2.4}$$

Nach (2.1) ist θ positiv. Daraus folgt, daß nach (2.2) und (2.3) die Gewichte W_1 und W_2 ebenfalls positiv sind. Die Ungleichung (2.4) kann dann nicht erfüllt werden. Dadurch ergibt sich, daß ein Perzeptron eine solche Berechnung nicht durchführen kann (vgl. Rojas, 1996).

Für eine Trennung bedarf es mindestens einer verdeckten Schicht.

2.2 Lernverfahren in neuronalen Netzen

Künstlich neuronale Netze „lernen" stets durch Modifikation von Gewichten. Wird ein neuronales Netz als Menge von Funktionen aufgefaßt, so läßt sich die Lernphase durch wiederholtes Aufrufen einer vorgegebenen Lernfunktion beschreiben. Diese Lernfähigkeit ist die interessanteste Komponente; sie erlaubt, daß ein Netz eine gegebene Aufgabe selbständig aus Beispielen löst. Gute Lernalgorithmen erkennen aus der „Erfahrung" notwendige Gewichtsveränderungen, um das anstehende Problem schrittweise einer Lösung zuzuführen. Es gibt unterschiedliche Verfahren, wie das Lernen ablaufen kann. Allgemein unterscheidet man zwischen unüberwachtem und überwachtem Lernen.

2.2.1 Unüberwachtes Lernen

Beim unüberwachten Lernen (unsupervised learning) gibt es keinen externen „Lehrer". Das Lernen geschieht durch Selbstorganisation. Dabei klassifiziert das Netz durch das Lernverfahren die angelegten Eingabemuster. Die Klassifikation vollzieht sich, indem ähnliche Eingabemuster in ähnliche Cluster eingeteilt werden.

Klassifizieren bedeutet in diesem Zusammenhang, daß bei ähnlichen Eingaben gleiche oder räumlich benachbarte Neuronen aktiviert werden. Der bekannteste Vertreter der unüberwachten Lernverfahren sind die im Abschnitt 2.5.1 vorgestellten selbstorganisierenden Karten nach Kohonen. Bei dieser Art des Lernens, die dem biologischen Lernen am nächsten kommt, werden besonders gut die statistischen Eigenschaften der Eingabemuster extrahiert (Kohonen, 1990).

2.2.2 Überwachtes Lernen

Beim überwachten Lernen gibt ein externer „Lehrer" zu jedem Eingabemuster der Trainingsmenge das korrekte bzw. beste Ausgabemuster an. Dem neuronalen Netz liegen immer gleichzeitig ein vollständig spezifiziertes Eingabemuster und das korrekte Ausgabemuster vor. Aufgabe des Lernverfahrens ist es, die Gewichte des Netzes so zu ändern, daß dieses nach wiederholter Präsentation von Eingabe- und Ausgabemustern diese Assoziation selbständig auch für unbekannte ähnliche Eingabemuster vornehmen kann.

2.3 Klassische Netze bei überwachtem Lernen

Ein künstliches neuronales Netz besteht aus einer Vielzahl von elektronischen Schaltelementen (= Neuronen). Diese Neuronen besitzen mehrere Eingänge und Ausgänge, die so miteinander verbunden sind, daß jedes Neuron eine gewichtete Ausgabe seiner Vorgänger als Eingangssignale erhält. Man spricht von der Eingabe- und der Ausgabeschicht eines Netzes.
Das von McCulloch und Pits vorgestellte Modell geht von einem Neuron als ein logisches Schwellenwertelement aus. Ein Neuron kann die Zustände aktiv (Zustand = 1) oder passiv (Zustand = 0) annehmen. Der Zustand des Schwellenwertelements entspricht der Summe der Eingangssignale im Vergleich mit einem vorgegeben Schwellenwert. Wird diese Vorgabe überschritten, wird das Neuron als aktiv bezeichnet, andernfalls nicht. Aktivierte Eingangssignale werden durch die „Synapsenstärke" oder Gewichte $w_i = +1$ beschrieben. Eine einfache schematische Darstellung ist in Abb. 8 zu sehen. Dort entsprechen P_i, $i = 1,...,m$ den Eingabewerten für das Netz, w_i, $i = 1,...,m$ den Gewichten. Die sogenannte Netzaktivität ergibt sich aus der gewichteten Summe der Eingaben, verringert um einen Schwellenwert θ an jedem Neuron i. Formal:

$$net = \sum_{i=1}^{m} P_i w_i - \theta \,. \tag{2.5}$$

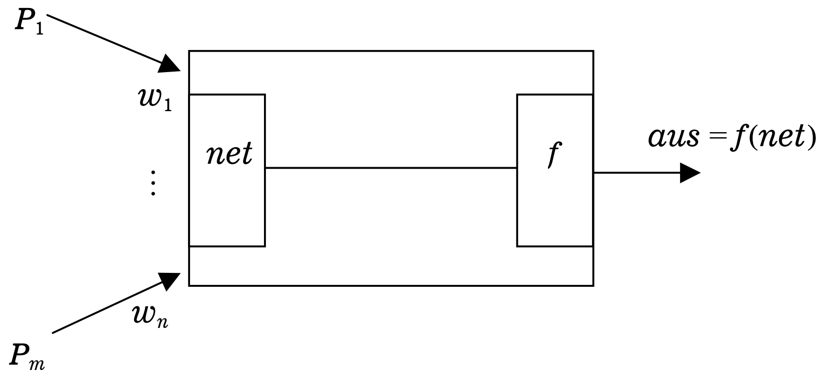


Abb. 8: Grundstruktur eines neuronalen Netzes. Die Eingabe P_i, mit Gewicht w_i wird an das Netz angelegt. Die Netzaktivität wird gemäß Formel (2.5) berechnet. Mittels der Aktivierungsfunktion f wird der Ausgangswert aus des Netzes berechnet.

Die Ausgabe des Netzes wird mittels einer Aktivierungsfunktion f gemäß

$$aus = f(net) \tag{2.6}$$

berechnet. Für die Aktivierungsfunktion aus (2.6) gilt $f(x) = 1$ für $x \geq 0$ und $f(x) = 0$ für $x < 0$. Je nach Netztyp wird die Aktivierungsfunktion unterschiedlich gewählt. Mögliche Funktionen sind lineare Funktionen, die den Merkmalsraum durch Einziehen von Hyperebenen in Halbräume zerlegen. Diese Art der Funktion hat sich in der Praxis jedoch nicht bewährt. Für diskrete Aktivierungen bieten sich Treppenfunktionen, wie die Heaviside-Funktion an. Am gebräuchlichsten sind jedoch die Sigmoidfunktionen. Bei Perzeptronen-Netzen (siehe unten) zählen Funktionen wie

$$f(x) = \frac{1}{1+e^{-x}},$$

oder logistische, sigmoide Verteilungsfunktionen wie

$$f(x) = \frac{1}{1+e^{-net}}$$

dazu.

Zu den bekannten Schwachstellen des Ansatzes gehört zum einen die Tatsache, daß es nicht möglich ist, zu erkennen, wie Neuronen verschaltet werden können. Das führt dazu, daß das Lernen der Netze nicht nachvollziehbar ist. Zum anderen müssen bei diesem Modell alle Komponenten funktionieren, um einen reibungslosen Ablauf zu gewährleisten. Die erste Schwachstelle ist durch die Einführung der Hebb'schen Lernregel gelöst worden (Hebb, 1949). Dieser zufolge ändert sich die Verschaltung zweier Neuronen proportional zur Aktivität vor und hinter der Synapse.
Durch die Entwicklung der Rechner ist man dann erstmals in der Lage, durch Simulationen die Lernfähigkeit von neuronalen Netzen zu erproben. Die für die Simulation notwendige Erweiterung wurde von Rosenblatt (1958) vorgenommen. Sein Ansatz soll im folgenden beschrieben werden.

2.3.1 Das Perzeptron und Backpropagation

Bei der Netzwerktopologie unterscheidet man in vorwärtsgerichtete (azyklische) und rückgekoppelte Netze. Der bekannteste Vertreter der azyklischen Topologie ist das Perzeptron oder „Multilayer-Perceptron" (Vielschicht-Perzeptron). Die grundlegende Änderung dieser Netztypen im Vergleich zu den von McCulloch und Pits vorgestellten Netzen sind die Einführung von Gewichten für die Neuronen. Diese Entwicklung geht auf Rosenblatt (1958) zurück. Er bezeichnete sein Netz als *Perzeptron*.
Ein Perzeptron besteht aus einer festen Anzahl N an Neuronen. Diesen Neuronen werden sogenannte Eingabemuster über m Leitungen angelegt. Ein Eingabemuster entspricht daher einem m-dimensionalen Vektor und gibt eine mögliche Eingabe aus der Menge aller möglichen Eingaben für das Netz an. Hinzu kommt, daß ausschließlich gewichtete Netze aufgebaut werden. Im Gegensatz dazu sind in den Modellen von McCulloch und Pits keine Gewichte vorgesehen. Die Gewichte sind an den Verbindungen (Kanten) zwischen den Neuronen angebracht. Mit Hilfe eines Algorithmus werden die Kantengewichte verändert. Diese Gewichtsveränderung wird als Lernen bezeichnet. Während der Lernphase (= die Dauer des Lernens) paßt jedes Neuron seine Gewichte an. Diese Anpassung vollzieht sich dergestalt, daß es nur auf Eingabemuster reagiert, die aus „seinem" Cluster kommen. Das Anlernen des Netzes bezeichnet man als Training.
Neben der Einführung von Gewichten gab es als weitere Entwicklung die Parallelisierung der Informationsverarbeitung. Diese wurde durch Minsky und Papert initiiert, indem sie das Rosenblatt-Modell bis auf das minimal Notwendige vereinfachten. Dieses Modell wies jedoch erhebliche Unzulänglichkeiten in der Berechnung von logischen Funktionen auf (siehe Abschnitt 2.1 sowie Minsky und Papert, 1969). Daher betrachtet man zur Lösungen solcher Probleme Vielschicht-Per-

zeptronen-Modelle. Für das Netz aus Abb. 8 bedeutet das, daß zwischen der Eingabeschicht und der Ausgabeschicht weitere, verdeckte Neuronenschichten eingesetzt werden.

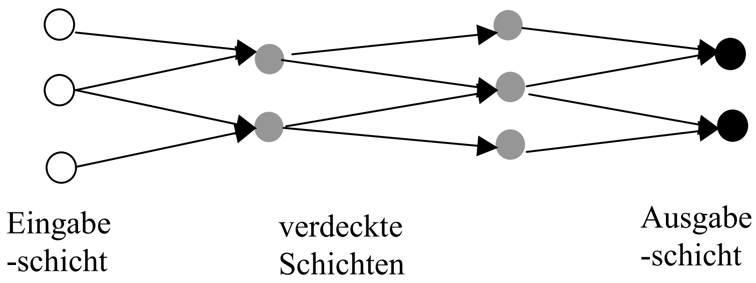


Abb. 9: Vielschicht-Perceptron: Zwischen Eingabeschicht und Ausgabeschicht werden weitere, verdeckte Neuronenschichten eingesetzt.

Die Ausgabe eines Netztes wie in Abb. 9 hängt nicht mehr nur von der Eingabe an das Netz ab, sondern auch vom Verhalten der inneren, verdeckten Schichten. Durch die Adaption der gewichteten Verbindungen zwischen den Neuronen wird ein solches Netz trainiert. Der Algorithmus zur Gewichtsadaption (Training) beim einfachen Perzeptron verläuft wie folgt:
Die Gewichte w_{ij} (Gewicht zwischen dem i-ten und j-ten Neuron) werden mit zufälligen Werten initialisiert. Ein Eingabewert $P=(P_1,\ldots,P_n)'$ wird zufällig ausgewählt. Da es sich hierbei um überwachtes Lernen handelt, ist zu jeder Eingabe die Ausgabe aus des Netzes bekannt. Wird für die an das Netz angelegte Eingabe eine korrekte Ausgabe berechnet, so werden die Gewichte nicht korrigiert. Weicht die berechnete Ausgabe von der bekannten Ausgabe ab, wird ein Fehlermaß δ dergestalt berechnet, daß die berechnete Ausgabe von der bekannten Ausgabe subtrahiert wird. Die Gewichte w_{ij} werden korrigiert mit:

$$w_{ij_{neu}} = w_{ij_{alt}} + \eta P_i \delta . \qquad (2.7)$$

Hierbei ist η, $\eta \in [0,1]$, die Lernrate. Bei verdeckten Schichten sind die Ausgaben dieser Schichten nicht mehr bekannt. Dieses wird mittels des Backpropagations-Algorithmus gelöst. An dieser Stelle werden die günstigen Eigenschaften der sigmoiden Funktion ausgenutzt. Neben den Eigenschaften der Differenzierbarkeit, strenger Monotonie (steigend) und der Stetigkeit gilt für die Ableitung f' von f:

$$f'(net) = f(net)(1-f(net)).$$

Genau diese Eigenschaft wird bei der Adaption der Gewichte im sogenannten Gradientenabstiegsverfahren verwendet. Die Grundidee des Gradientenabstiegsverfahren besteht in der Minimierung des Fehlermaßes δ, indem die Gewichte w_{ij} entgegen dem Gradienten mit Schrittweite $\Delta(t)$ verändert werden (Braun, 1997). Dabei bezeichnet t die Sollausgabe des Netzes. Als geeignetes Fehlermaß dient z. B. der quadratische Fehler als Funktion des Gewichtsvektors. Bezeichne W den Gewichtsvektor und E den Fehler, so wird der Gradient gemäß

$$\Delta W = -\eta \nabla E \, .$$

berechnet. Das heißt, es wird in Richtung des Fehler-Gradienten solange abgestiegen, bis ein lokales Minimum von E erreicht wird. Für die genaue Beschreibung dieses Verfahrens siehe Braun (1997) oder Matecki (1999). Eine Skizze der Herleitung des Backpropagationalgorithmus ist in Anhang A nachzulesen.

2.4 Lernende Vektorquantisierung

Die lernende Vektorquantisierung (LVQ) (Kohonen et al., 1996) und die selbstorganisierenden Karten (SOM) (Kohonen et al., 1996; Kohonen, 2001; siehe Abschnitt 2.5.1) sind eng verwandt. SOM's sind Weiterentwicklungen der LVQ unter Einbeziehung einer Nachbarschaftsbeziehung zwischen den Neuronen. Der Unterschied zwischen diesen Methoden liegt in den verschiedenen Lernverfahren. Während die LVQ ein überwachter Lernvorgang ist, bei dem zu jedem Eingabevektor die Clusterzugehörigkeit bekannt sein muß, sind SOM's unüberwachte Lernverfahren, bei denen diese Clusterinformation nicht bekannt ist.

Das Prinzip der lernenden Vektorquantisierung beruht auf einer ungeordneten Menge von Gewichtsvektoren (sogenannte Codebook-Vektoren), die so über den Eingaberaum verteilt werden sollen, daß sie diesen möglichst hinreichend repräsentieren. Für die zu bildenden Cluster gibt es mehrere Gewichtsvektoren. Ein neuer oder trainierter Eingabevektor wird gleichzeitig mit allen Referenzvektoren verglichen und der ähnlichste dieser Referenzvektoren gibt dann den Cluster an, zu welcher die neue Eingabe zählt. Kohonen führte von dieser Methode drei Varianten ein, die in der Literatur als LVQ1, LVQ2.1 und LVQ3 bezeichnet werden. Auch eine Optimierung des LVQ1 wurde vorgestellt.

2.4.1 LVQ1

Die Gewichtsvektoren $W_1,\ldots,W_q$ werden bei der LVQ1 Methode parallel mit dem Eingabevektor P verglichen. Das Neuron j^*, dessen Gewichtsvektor W_{j^*} der Eingabe am ähnlichsten ist, wird als Gewinner bezeichnet. Dazu wird bezüglich der L_2-Norm der folgende Ausdruck minimiert:

$$j^* = \underset{j=1,\ldots,q}{\arg\min} \left\| P - W_j \right\|_2 .$$

LVQ1 ist prinzipiell ein Nächster-Nachbar-Klassifikator. Jedoch werden nicht alle Trainingsmuster gespeichert. Ferner werden mittels eines Lernverfahrens die Gewichtsvektoren W_j modifiziert. Die den Klassifikationsfehler minimierenden Gewichtsvektoren können bei diesem Verfahren wie folgt gefunden werden (Zell, 2000):

$$W_i(s+1) = \begin{cases} W_i(s) + \alpha(s)\,(P - W_i(s)) & \text{falls } W_i, P \text{ in gleichem Cluster} \\ W_i(s) - \alpha(s)\,(P - W_i(s)) & \text{falls } W_i, P \text{ nicht in gleichem Cluster} \end{cases} .$$

Der Gewichtsvektor W_i, der dem Eingabevektor am ähnlichsten ist, wird diesem noch ähnlicher gemacht, sofern der Eingabevektor P und der Gewichtsvektor in dem selben Cluster liegen. Andernfalls wird der Gewichtsvektor unähnlicher. Diese Veränderung der Ähnlichkeit geschieht über die Addition oder Subtraktion des gewichteten Abstandes von P zu W_i. Dabei kann das Gewicht α entweder konstant oder monoton fallend sein (Zell, 2000). Ferner gilt $0 < \alpha < 1$.

2.4.2 LVQ2.1

Die Methode LVQ2.1 ist der Methode LVQ1 sehr ähnlich. Bei gleicher Eingabe-Klassifikation wird allerdings die Modifiaktion der Gewichtsvektoren verändert. Dieses geschieht dergestalt, daß das LVQ2.1 den nächsten und übernächsten Gewichtsvektor aussucht und die Gewichtsadaption nur dann durchführt, wenn entweder die beiden Gewichtsvektorencluster sich unterscheiden, die Eingabe P einer der Gewichtsvektorencluster angehört oder die Eingabe in einem Fenster entlang der Mittelsenkrechten zwischen den beiden Clustern liegt. Ein Hinweis auf die Breite des Fensters liefert Kohonen et al. (1992). So soll das Minimum des

Verhältnisses vom euklidischen Abstand der Eingabe von den Gewichtsvektoren nicht über eine vorgegeben Schranke von $0{,}2$ oder $0{,}3$ steigen. Grafisch kann dieser Sachverhalt wie in Abb. 10 veranschaulicht werden.
Die Formel der Gewichtsveränderung bei LVQ2.1 ist analog zu LVQ1. Die Besonderheit dieses Verfahrens besteht in der Grenzflächenveränderung der Cluster durch Verschiebung der Vektoren. Daher kann es vorkommen, daß die Dichte der Vektoren sich der Verteilungsdichte der Eingabevektoren annähert.
Diesen Mißstand umgeht das LVQ3-Verfahren.

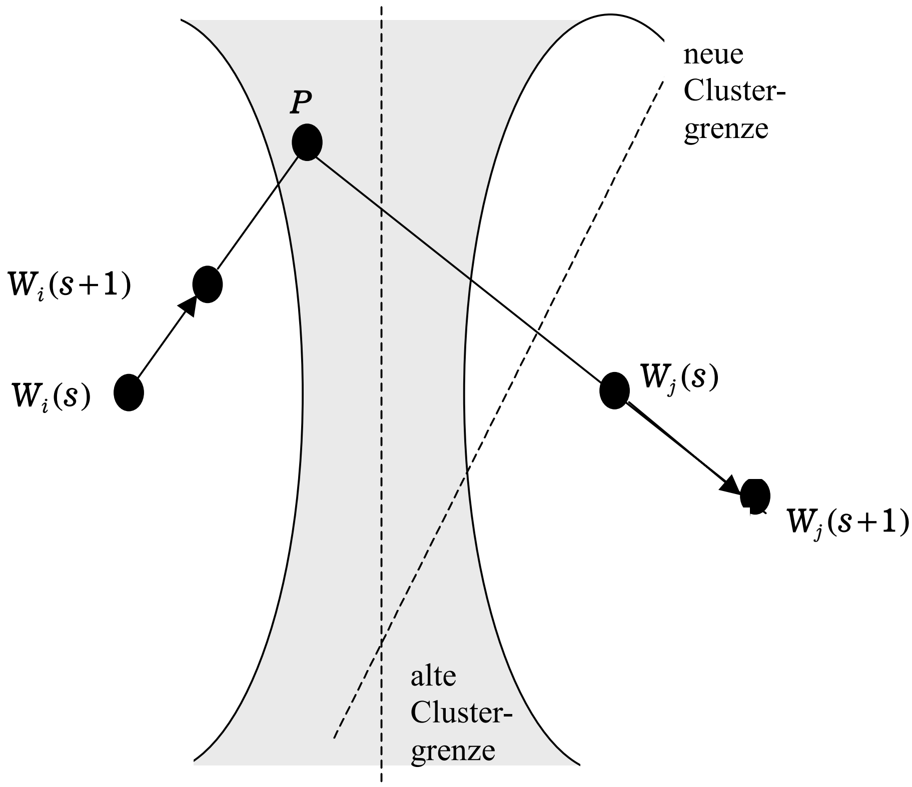


Abb. 10: Prinzip des LVQ2.1. Das Gewicht $W_i(s)$, dessen Cluster mit der von P übereinstimmt, wird in Richtung der Eingabe P verschoben. Das Gewicht $W_j(s)$, dessen Cluster nicht P übereinstimmt, wird weggeschoben. Dadurch dreht sich die Clustergrenze. Das Fenster ist kein Rechteck (vgl. Zell, 2000).

2.4.3 LVQ3

Als Erweiterung des LVQ2.1-Verfahrens werden bei LVQ3 zusätzlich zur Änderung der Gewichtsvektoren einer Clustergrenze die Gewichtsvektoren W_i und W_j verändert. Dieses aber nur dann, wenn beide Gewichtsvektoren zum gleichen Cluster gehören.

Dieser Ansatz hat den Vorteil, daß im Gegensatz zu LVQ2.1 optimale Gewichtsvektoren nicht mehr verändert werden. Dagegen wird bei LVQ2.1 nur die relative Entfernungen der Gewichtsvektoren von der Grenzlinie zu den Clustern optimiert, wohingegen bei den Verfahren LVQ1 und LVQ3 die Lage der Gewichtsvektoren der Verteilung der Eingabevektoren angepaßt wird. Das impliziert, daß bei LVQ2.1-Verfahren eine schärfere Clustertrennung mit einer kleineren Lernrate α verwendet wird.

2.5 Netze bei unüberwachtem Lernen: Winner-takes-all-Netze

Die Klasse der Winner-takes-all-Netze (WTAN) ist ein neuronales Netzmodell bei unüberwachtem Lernen. Die im Abschnitt 2.5.1 vorgestellten selbsorganisierenden Karten nach Kohonen sind eine Variante dieses Netztypes, vgl. Kohonen (1990), Kohonen (2001).

Ein WTAN besteht aus einer Eingabe- und Ausgabeschicht; verdeckte Schichten existieren nicht. Ferner weisen die Schichten keine Strukturen zwischen den Neuronen auf. Die Anzahl der Neuronen der Eingabeschicht entspricht der Anzahl m der Merkmale der verarbeitenden Daten. Als Aktivierungsfunktion der Eingabeschicht ist die Identität $f(x) = x$ zu wählen. Die Ausgabeschicht besteht aus q Neuronen. Die Neuronen der Ausgabeschicht sind über einen m-dimensionalen Gewichtsvektor mit der Eingabeschicht verbunden. Bei Eingabe an das Netz wird genau ein Neuron der Ausgabeschicht aktiviert. Ein WTAN ist aufgrund seiner Neuronenverbindungen kein Feed-Forward-Netz, bei denen die Neuronenverbindung nur von der Eingabeschicht zur Ausgabeschicht verläuft. Da bei den WTANs genau ein Neuron der Ausgabeschicht aktiviert werden soll, bedarf es einer Rückkoppelung zwischen den Neuronen der Ausgabeschicht. Hierbei erhält jedes Neuron hemmende Verbindungen zu den anderen Neuronen der Schicht. Nur das Neuron mit der stärksten Aktivierung setzt sich durch. Veranschaulicht wird dieser Sachverhalt durch die nachstehende Abb. 11. Die weißen Neuronen repräsentieren die Eingabeschicht, die grauen die nichtaktivierten Neuronen der Ausgabeschicht und das schwarze das aktivierte Neuron. Durch die Pfeile ist der Verbindungsverlauf zwischen den Neuronen gekennzeichnet. Pfeile mit einem durchgehenden Strich deuten auf das Gewinnerneuron. Gestrichelte Pfeillinien weisen auf die nichtaktivierten Neuronenverbindungen hin. Ein Doppelpfeil weist auf die Rückkoppelung hin.

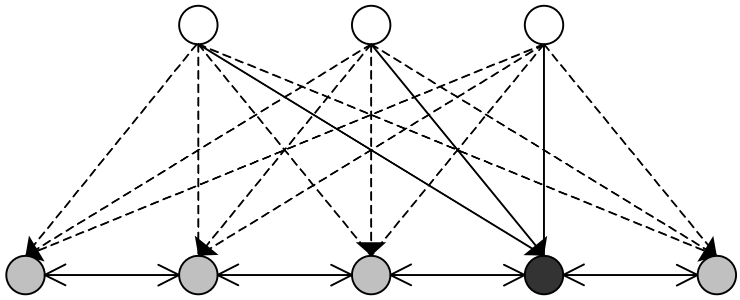


Abb. 11: Winner-takes-all-Netz mit drei Eingabeneuronen und fünf Ausgabeneuronen.

Die Aktivierungsfunktion der Ausgabeschicht eines WTAN für die Klassifikation der Eingabe $\boldsymbol{P}$ ist wie folgt definiert:

$$f_{WTAN}(N_j) = \begin{cases} 1 & , \quad \| P - W_j \|_2 = \min\limits_{i=1,\ldots,q} \{ \| P - W_i \|_2 \} \\ 0 & , \quad \| P - W_j \|_2 \neq \min\limits_{i=1,\ldots,q} \{ \| P - W_i \|_2 \} \end{cases}, \tag{2.8}$$

mit N_j als Neuron, dessen Aktivierung gesucht wird, und W_i die Gewichte der Ausgabeneuronen N_i, $i = 1,..,q$.
Gemäß (2.8) wird das Ausgabeneuron aktiviert dessen Gewichtsvektor den geringsten Abstand zur Eingabe aufweist. Die Abstandsberechnung erfolgt im Sinne einer geeigneten Norm. Aus der Tatsache, daß genau ein Neuron aktiviert wird, hat diese Netzart seinen Namen. Der Gewinner erhält „alles".
Ein Nachteil dieser Netzart besteht in der geeigneten Initialisierung der Gewichtsvektoren. Bei einer zufälligen Initialisierung ist es möglich, daß ein Großteil der Neuronen niemals Gewinner wird und daher unverändert bleiben. Betrachten wir dazu folgendes Beispiel (vgl. Braun, 1997):

Beispiel
Gegeben sei ein eindimensionaler Eingaberaum mit Lernbeispielen, die ausschließlich aus dem Intervall $[2, 3]$ stammen. Seien die Neuronen zufällig nahe Null initialisiert, dann wird nur das Neuron Gewinner, welches das größte Gewicht aufweist. Nur dieses Neuron wird verändert. Abschwächen läßt sich diese Tatsache, indem man die Neuronen zufällig um den Mittelwert aller Lernbeispiele initialisiert. Bei einer kleinen Lernrate verteilen sich die Neuronen sternförmig aus dem Zentrum der Lernbeispiele heraus.

Damit dieser Ansatz als Methode des unüberwachten Lernens aufgefaßt werden kann, sei auf den k-Mittelwert Klassifizierungsalgorithmus verwiesen (vgl. Abschnitt 1.1.1).
Mit Hilfe dieses Algorithmus kann eine unüberwachte Klassifikation gefunden werden. Ist im voraus schon eine Klassifikation der Elemente vorhanden, ist die mittels unüberwachter Klassifikation gefundene Einteilung gewöhnlich nicht mit der vorgegebenen konsistent. Bei neuronalen Netzen, die sich diesem Algorithmus bedienen (z. B. WTAN), wird diese Inkonsistenz dadurch hervorgerufen, daß ein Neuron Gewinner für zwei Lernbeispiele sein kann. Darüberhinaus ist es aufgrund der Verwendung eines einfachen euklidischen Abstandsmaßes nicht möglich, daß ein Cluster genau einem Neuron entspricht. Ein Cluster wird im allgemeinen durch mehrere Neuronen repräsentiert.

2.5.1 Selbstorganisierende Karten

Selbstorganisierende Karten (engl. self-organizing map oder kurz SOM) sind in ihrer klassischen Form ein unüberwacht lernendes, zweischichtges Winner-takes-all-Netz. Sie bestehen aus einer Eingabeschicht und k Ausgabeschichten. Das von Kohonen (1982) beschriebene Modell berücksichtigt die räumliche Anordnung der Neuronen, indem nicht nur das Gewinner-Neuron j^* angepaßt wird, sondern auch seine Nachbarn. Diese Nachbarn werden durch eine Nachbarschaftsfunktion d_{ij^*} beeinflußt. Die Veränderung eines Neurons und seiner Nachbarn, zusammen als Nachbarschaftsstruktur bezeichnet, ist der wesentliche Unterschied zu den WTANs. Die räumliche Struktur ist z. B. eine zweidimensionale reguläre Gitterstruktur. Wie beim LVQ gibt es eine Menge von Neuronen, deren Gewichtsvektoren im Raum der Eingabevektoren so verteilt werden, daß sie ihn möglichst gut abdecken. Im Unterschied zur LVQ-Methode existiert zusätzlich eine lokale Nachbarschaftsfunktion zwischen den Neuronen. Durch das Lernen des Netzes mittels eines iterativen Algorithmus (siehe unten) wird eine Kartierung des Merkmalsraumes erreicht, so daß jedes Neuron für einen zusammenhängenden Bereich im Merkmalsraum zuständig ist. Nach Braun (1997) bewirkt das Anpassen der Nachbarn, daß die Gewichtsvektoren benachbarter Neuronen ähnlich sind und damit benachbarte Neuronen von einem Eingabemuster ähnlich aktiviert werden. Die Neuronen bilden eine topologische Merkmalskarte. Die Besonderheit der SOM's ist, daß diese topologische Struktur während des Lernens nicht verletzt wird. Man spricht auch von topologieerhaltender Struktur. Durch diese Eigenschaft ist es mit einer SOM möglich, hochdimensionale Daten auf den zweidimensionalen Raum des Netzes zu projizieren. Eine Anwendung und Ausschöpfung dieser Tatsache ist beispielsweise in Kraaijveld et al. (1995) nachzulesen. Eine typische SOM ist in nachstehender Abb. 12 dargestellt:

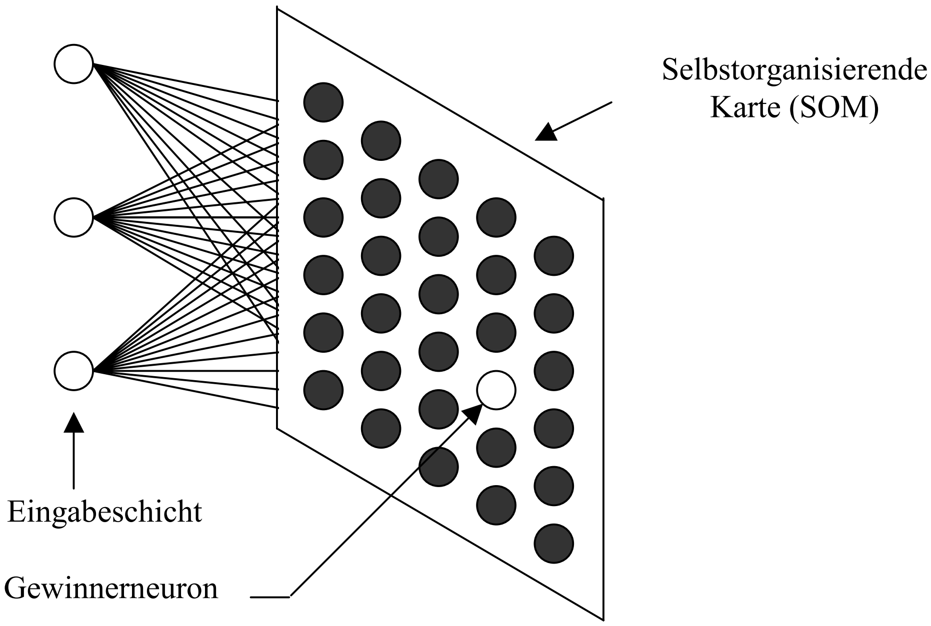


Abb. 12: Schematische Darstellung einer 5 x 6 SOM. In der Eingabeschicht befinden sich 3 Eingabeneuronen. Die Ausgabeschicht wird von 30 Ausgabeneuronen gebildet. (Adaptiert von Matecki, 1999)

Während bei vielen neuronalen Netzmodellen die Neuronenlage im Netz nicht entscheidend ist (hier interessieren mehr die Verbindungen der Neuronen und die dazugehörigen Gewichte), ist diese Tatsache bei selbstorganisierenden Karten von Bedeutung. Diese räumliche Struktur soll bei einer SOM so angelegt werden, daß der Lernprozeß das sogenannte Ansprechverhalten (Reaktion auf eine Eingabe) eines Neurons optimiert. Ziel der Optimierung ist die Reaktion benachbarter Neuronen auf ähnliche Eingaben. Dabei werden die Gewichtsvektoren so trainiert, daß sie die Eingabevektoren räumlich geordnet repräsentieren. Ist die Gitterstruktur des Netzes zweidimensional und die Eingabe von hoher Dimensionalität, erreicht die topologieerhaltende SOM eine starke Kompression des Merkmalraumes (niedere Dimensionalität) (vgl. Braun, 1997). Mit andern Worten, das Gitter der Gewichtsvektoren definiert eine Art nichtlineare Regression der Gewichtsvektoren auf die Datenpunkte der Eingabevektoren. Gleichzeitig wird dabei eine topologieerhaltende Abbildung der m-dimensionalen Eingabevektoren aus $\mathbb{R}^m$ auf das q-dimensionale Gitter der Neuronen definiert.
Da eine SOM zur Klasse der unüberwachten Verfahren gehört, gibt es keine bekannten Eingabe-Ausgabe-Muster, sondern eine SOM richtet sich nach wieder-

holtem Ausführen einer Aufgabe im Lernverfahren aus. Die Aktivierungsfunktion ist analog zum WTAN zu wählen. Sie entspricht somit der Form (2.8). Der Unterschied liegt in der Neuronenanordnung der Ausgabeschicht. Die Visualisierung der Nachbarschaftsstruktur verdeutlicht diesen Sachverhalt. Diese Methode der Visualisierung wird als U-Matrix bezeichnet (vgl. Kohonen, 2001; Ultsch und Siemon, 1989). In dieser Darstellung werden geringe Abstände zwischen den Neuronen durch helle Grauwerte und große Abstände mit dunklen Grauwerten markiert. Die Struktur dieser U-Matrix und mithin die Anordnung der Ausgabeneuronen sowie deren Auswirkung auf die Darstellungsform ist bei Zerbst (2001) nachzulesen.

Der von Kohonen (1982) vorgestellte Algorithmus sei im folgenden beschrieben: Das Training eines Netzes wird zunächst durch die zufällige Initialisierung des Gewichtsvektors W gestartet. Nach der Auswahl eines Teils der Daten als Lerneingaben für das Netz beginnt der Lernprozeß. In der Lernphase wird jeder Wert des Trainingsdatensatzes mindestens einmal als Lerneingabe geschaltet. Dieser Vorgang sei im weiteren als Lernzyklus definiert. Das häufige Hintereinanderschalten des Trainigsdatensatzes wird als eine Reihe von Lernzyklen bezeichnet. Dabei gilt: Je mehr Lernzyklen, desto besser die Anpassung des Netzes an die Ausgangsdaten. Mit dem Anlegen der Lerneingabe an das Netz beginnt die Anpassung und das Gewinner-Neuron wird bestimmt. Die Anpassung wird als Adaption bezeichnet. In jeder Iteration (einem Lernzyklus) des Lernprozesses werden drei Schritte ausgeführt. Die Auswahl eines zufälligen Eingabevektors P aus dem Merkmalsraum (z. B. $I\!R$) ist der erste Schritt. Anschließend wird dieser Eingabevektor durch das Netz propagiert und im letzten Schritt der Gewinner bestimmt. Die Gewichte werden entsprechend einer geeigneten Funktion angepaßt.

Sei $P = (p_1,...,p_m)$, $p_k \in I\!R$, der m-dimensionale Eingabevektor für ein neuronales Netz. Sei $W_i = (w_{i1},...,w_{im})$, $i = 1,...,q$, eine Menge von Gewichtsvektoren, wobei das Gewicht W_i zum Neuron N_i, $i = 1,...,q$, gehört. Sei ferner q die Anzahl der Neuronen der Ausgabenschicht.

Um das Gewinnerneuron N_{j*} mit dem minimalen euklidischen Abstand zwischen dem Eingabe- und dem Gewichtsvektor zu bestimmen, betrachtet man

$$N_{j*} :\ j^* = \arg\min_{j=1,\dots,q}\left(\left\|P - W_j\right\|_2\right).$$

Die Gewichtsvektoren in der Umgebung des Gewinnerneurons werden nun in Richtung des Eingabevektors verschoben. Dabei wird der Abstand dieser benachbarten Neuronen zum Gewinner vom s-ten zum $s+1$-ten Iterationsschritt gemäß

$$W_i(s+1) = W_i(s) + \eta(s) \cdot d_{ij*}(s) \cdot (P - W_i(s)), \quad \forall\, i = 1,\dots,q \tag{2.9}$$

ermittelt. Die Lernrate $\eta \in [0,1]$ in (2.9) ist eine monoton fallende Funktion $\eta(s+1) = \alpha\,\eta(s)$, $\alpha \in (0,1)$, mit den Eigenschaften

$$\forall s : \eta(s) \in [0,1]$$

und

$$\forall s < \tilde{s} : \eta(s) > \eta(\tilde{s}).$$

Der Faktor α wird als Konvergenzgeschwindigkeit des Algorithmus bezeichnet. In der ersten Phase des Lernes einer SOM sollte die Lernrate möglichst groß, also nahe an eins, sein und dann monoton fallen. Nach Kohonen (2001) ist es unerheblich, ob η linear, exponentiell oder umgekehrt proportional zur Zeit fällt. Kohonen (2001) stellt auch die optimale Lernrate vor. Allerdings ist diese „optimale" Lernrate ein theoretischer Wert, der in vielen praktischen Anwendungen nicht einsetzbar ist, weil zu viele verschiedene Werte der Lernrate bestimmt werden müssen. Anhand der rekursiven Berechnung des Mittelwertes kann die „optimale" Lernrate hergeleitet werden. Betrachte dazu einen Satz Vektoren $\{v(t), v(t) \in I\!R^m\}$ mit t = $0,1,\ldots T$; T Indexmenge. Durch die rekursive Berechnung des Mittelwertes E von $v(t)$ ist es offensichtlich, daß dieser zu allen Zeitpunkten t korrekt ist, sofern die Anzahl der verwendeten Vektoren berücksichtigt wird (Kohonen, 2001). Formal ergibt sich daher:

$$E(t) = \frac{1}{T}\sum_{t=1}^{T} v(t)\ .$$

$$\begin{aligned} E(t+1) &= \frac{t}{t+1}E(t) + \frac{1}{t+1}v(t+1) \\ &= E(t) + \frac{1}{t+1}\left(v(t+1) - E(t)\right). \end{aligned}$$

Die Struktur der Formel ist analog zur Formel (2.9). Mit der Funktion d_{ij*}, der Nachbarschaftsfunktion, ergibt sich die optimale Lernrate für SOM's als:

$$\eta(s+1) = \frac{\eta(s)}{1 + d_{ij}(s)\eta(s)}$$

(vgl. Kohonen, 2001).

Die Nachbarschaftsfunktion d_{ij*} in (2.9) ist eine Distanzfunktion, welche die Veränderung der Gewichtsvektoren aller Neuronen im Hinblick auf den Gewinner beschreibt. Für die Formulierung der Nachbarschaftsfunktion verwendet man statt des Zeitparameters s häufig einen Distanzparameter u, der die Größe der Umgebung angibt. Für $s \to \infty$ gilt $u \to 0$.

Es sind viele Funktionen d_{ij^*} denkbar, wie z. B. die häufig benutzte Gauss-Funktion

$$d_{ij^*}(s) = e^{-\frac{\|W_i(s)-W_{j^*}(s)\|^2}{\sigma^2(s)}},$$

wie sie auch Kohonen (1990) vorgeschlagen hat. Diese Funktion „überwacht“ die paarweisen Distanzen der Gewichtsvektoren. Dabei ist $\sigma^2(s)$ eine geeignet gewählte in Abhängigkeit vom Iterationsschritt monoton fallende Varianzfunktion. Die Varianzfunktion wird in Abhängigkeit vom Iterationschritt gewählt, damit sich die SOM im Laufe des Lernes fein und detailliert ausbildet. Zu Beginn werden die Gewichtsvektoren global und mit großen Schritten verändert. Durch die Abnahme von $\sigma^2(s)$ werden die Gewichtsvektoren nur noch lokal angepaßt. Schließlich wird die Varianzfunktion so klein, daß nur noch der Gewinner bewegt wird. Nach Kraaijveld et al. (1995) ist der Algorithmus unempfindlich gegenüber der Wahl von $\sigma^2(s)$.
Die Gauss-Funktion ist nicht immer geeignet. In unserem Zusammenhang erscheint eine andere Funktion eher angebracht. Auf diese wird in Kapitel 4 eingegangen.
Der Gewichtsvektor-Anpassungsschritt aus (2.9) stellt die Verbindung zum oben vorgestellten k-Mittelwert-Verfahren dar (vgl. Abschnitt 1.1.1). Der Gewinner wird auch hier ein kleines Stück zum Eingabevektor verschoben, jedoch werden bei einer SOM neben dem Gewinner auch noch seine Nachbarn mitverschoben.
Damit eine SOM für Klassifikationsaufgaben heranzuziehen ist, muß diese im allgemeinen nach dem Lernen kalibriert werden. Dazu wird eine Variante von LVQ benutzt. Wie Matecki (1999) vorstellt, wird bei der Kalibrierung ein Satz an Eingabevektoren, deren Clusterzuordnung bekannt ist, durch die Karte propagiert. Dazu sollte ein Trainingsdatensatz verwendet werden. Jede Ausgabe wird anschließend einem Cluster zugeordnet. Das ist leicht möglich, da die Clusterzugehörigkeit der Eingabe bekannt ist. Danach kann das Cluster zur Klassifiktion eingesetzt werden. Natürlich können immer noch Mißklassifikationen auftreten, wodurch ein Fehlermaß notwendig ist. Ein geeignetes Fehlermaß kann bei Matecki (1999) nachgelesen werden.

2.5.2 Hinweise zur Verwendung von SOM’s

Kohonen (2001) empfiehlt aus Gründen der besseren visuellen Darstellung ein hexagonales Gitter, auch wenn sich ein quadratisches Gitter einfacher in einem Rechner implementieren läßt. Auch auf die Form der Karte geht Kohonen ein. So rät er, die äußere Form nicht quadratisch, sondern rechteckig zu wählen, damit es für die Karte weniger Symmetrien in der Orientierung gibt. Bei einer runden Karte beispielsweise gibt es keine stabile Orientierung, während bei länglichen Formen

die Orientierung für die Karte einfacher ist. Eine andere Kartenform wird im Verlauf dieses Abschnittes kurz angerissen.
Leider gibt es bei SOM’s keine Regel für die Anzahl an benötigten Neuronen. Diese Wahl bleibt dem Anwender überlassen. Ferner kann es bei ungeeigneter Initialisierung des Netzes, etwa durch eine Änderung der Nachbarschaftsfunktion d_{ij^*} oder der Lernrate η, dazu kommen, daß eine SOM zu früh konvergiert. Dabei können topologische Defekte auftreten, wie sie in Abb. 13 zu sehen sind. Defekt bedeutet in diesem Zusammenhang, daß sich die Karte nur auf Teilgebieten korrekt entfaltet. Kohonen spricht davon, daß die Karte dann zu „steif“ sei. Häufig entsteht diese Steifheit durch den Distanzparameters u. Ist dieser am Anfang zu klein oder sinkt er zu schnell, können sich entfernte Neuronen nicht mehr gegenseitig beeinflussen. In einer solchen Karte ordnen sich dann zwar Teilbereiche des Netzes korrekt an, die globale Ordnung kann aber gestört sein (Zell, 2000).
Bei gegebener Steifheit kann die Qualität einer solchen SOM über einen Quantisierungsfehler gemessen werden.

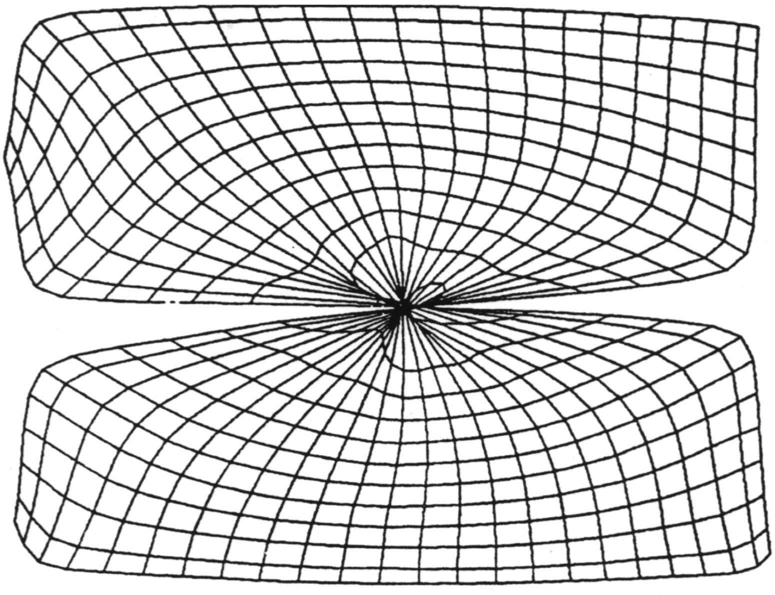


Abb. 13: Eine defekte SOM

Im Laufe der Jahre wurden viele verschiedene SOM-Algorithmen entwickelt. Um die topologischen Eigenschaften zu verbessern, betrachten einige dieser Varianten flexiblere Kartenstrukturen anstelle eines festen Gitters (Fritzke, 1994). Diese flexiblen Strukturen sind aber für Visualisierungsmaßnahmen nicht geeignet.
Andere Ansätze verfolgen die Reduktion der Rechenkomplexität der selbstorganisierenden Karten. Die Rechengeschwindigkeit ist ein wichtiger Aspekt der Datenverarbeitung, insbesondere in der Klassifikation von Luftaufnahmen und Satellitenbildern. So kann z. B. die Geschwindigkeit beim Auffinden des Gewinners verbes-

sert werden, wenn baumartige SOM's benutzt werden (Koikkalainen, 1994). Dabei entspricht jede Ebene des Baumes einer eigenen übergroßen SOM. Die Suche nach dem Gewinner wird dann Ebene für Ebene durchgeführt. Es wird dabei ausschließlich in Bereichen gesucht, die unterhalb des zuletzt gefundenen Gewinners liegen. Der Baum wird von den Blättern her durchlaufen. Auch hierarchische Baumstrukturen sind denkbar (Für die Definition eines Baumes aus Sicht eines Informatikers siehe Güting (1992)).
Viele dieser Geschwindigkeitsverbesserungen haben auch Nachteile. Einige sind suboptimal in dem Sinne, daß sie die original SOM nur approximieren.
Bishop et al. (1996) führten das Generative Topographic Mapping (GTM) ein, welches sich grob an einer SOM orientiert. Die Idee hinter dieser Art von Modellen ist, daß ein Datensatz hinreichend gut durch wenige latente Variablen beschrieben werden kann. Während der Raum, aus dem diese latenten Variablen kommen, kontinuierlich sein kann, ist ein GTM, ähnlich wie bei einer SOM, eine diskrete Gitterstruktur. Eine Wahrscheinlichkeitsverteilung auf diesem Gitter wird postuliert, um eine Wahrscheinlichkeitsverteilung auf dem Eingaberaum (input space) mittels des parametrisierten Modells zu erzeugen. Gegeben ein Eingaberaum, ist es das Ziel einer GTM ein Modell mittels Maximum-Likelihood-Schätzung an die Daten anzupassen. Dieses geschieht iterativ mit dem EM-Algorithmus (Dempster et al., 1977), indem zunächst die Wahrscheinlichkeitsverteilung des Gitters geschätzt wird und dann die Likelihood der Eingabedaten maximiert wird. Detaillierteres über GTM, seine Vor- und Nachteile sind in Bishop et al. (1996) nachzulesen.

Im folgenden wird ein kurzer, allgemeiner Überblick über den mathematischen Ansatz der selbstorganisierenden Karten gegeben. Dieser kann abgehoben vom Rest dieses Kapitels gelesen werden.

2.6 Mathematischer Ansatz des Kohonen-Algorithmus

Im allgemeinen Fall, daß ein m-dimensionaler Definitionsbereich auf ein q-dimensionales Netz abgebildet wird, ist der Algorithmus von Kohonen schwierig zu analysieren. Daher sei an dieser Stelle der eindimensionale Fall ohne und mit Nachbarschaftbeziehung betrachtet.

2.6.1 Der eindimensional Fall ohne Nachbarschaftsbeziehung

Gegeben sei ein Netz mit einem Neuron. Die Eingabe an das Netz habe das geschlossene Intervall $[a, b]$ als Definitionsbereich. Ist die Funktionsweise des Algorithmus korrekt, so wird sich das Neuron, genauer das Gewicht dieses Neurons, im

Verlauf des Lernens in die Mitte des Intervalls bewegen. Das dieses wirklich so ist, kann wie folgt veranschaulicht werden:
Sei W_0 das Ausgangsgewicht des Neurons. Während des Lernens wird das Gewicht sukzessive verändert, wie in Abb. 14 dargestellt:

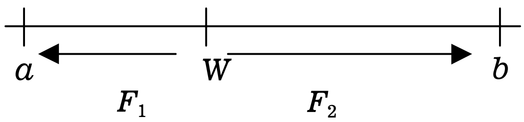


Abb. 14: Intervall a, b mit Gewicht W im eindimensionalen Fall.

Die im Segment $[a, W]$ ausgeübte „Anziehungskraft“ F_1 auf W ist proportional zur halben Länge dieses Segments, da der Erwartungswert eines zufällig ausgewählten Punkts in diesem Segment in der Mitte des Segments liegt. Analoges gilt für den Bereich $[W, b]$. Hier kehren sich jedoch die Vorzeichen um. Daraus folgt:

$$E\left(\frac{\partial W}{\partial t}\right) = \eta(b - W) + \frac{\eta}{2}(a - W) = \eta\left(\frac{a+b}{2} - W\right), \tag{2.10}$$

wobei t ein Zeitindex sei. Der Erwartungswert der Veränderung von W nach $W(0)$ ist folglich

$$E(\Delta W(0)) = \eta\left(\frac{a+b}{2} - W(0)\right). \tag{2.11}$$

Mit $y = W - (a+b) / 2$, ist (2.11) gegeben durch:

$$E(\Delta W(0)) = -\eta y_0, \tag{2.12}$$

wobei $y_0 = W(0) - (a+b) / 2$.

Der Erwartungswert $E(W(s))$ des Neuronengewichts im nächsten Lernschritt ist:

$$\begin{aligned} E(W(1)) &= W(0) + E(\Delta W(0)) \\ &= \frac{a+b}{2} + y_0 - \eta y_0 \\ &= \frac{a+b}{2} + y_0(1-\eta). \end{aligned} \tag{2.13}$$

Die erwartete Veränderung des Gewichts $W(1)$ folgt aus (2.10):

$$\begin{aligned} E(\Delta W(1)) &= -\eta y_1 \\ &= -\eta\left(E(W(1)) - \frac{a+b}{2}\right) \\ &= -\eta(1-\eta)y_0 \ . \end{aligned} \tag{2.14}$$

Für den Erwartungswert von $W(2)$ des Gewichts im zweiten Lernschritt gilt:

$$\begin{aligned} E(W(2)) &= E(W(1)) + E(\Delta W(1)) \\ &= \frac{a+b}{2} + y_0(1-\eta) - \eta(1-\eta)y_0 \\ &= \frac{a+b}{2} + y_0(1-\eta)^2 \ . \end{aligned} \tag{2.15}$$

Für den s-ten Schritt gilt daher:

$$E(W(s)) = \frac{a+b}{2} + y_0(1-\eta)^s \ . \tag{2.16}$$

Wegen $0 < \eta \leq 1$ ist der Grenzwert von (2.16) für $s \to \infty$ gerade $(a+b)/2$. Also bewegt sich das Neuronengewicht zur Mitte der Intervalls.
Analog kann man sich auch folgendes veranschaulichen.
Mit der Lernvorschrift

$$W_i(s+1) = W_i(s) + \eta\, d_{ij^*}(s)(P - W_i(s)) \ ,$$

gilt dann, daß die Punkte $W(1)$, $W(2)$,...,$W(s)$ das Intervall $[a, b]$ nicht verlassen können. Dabei bezeichne $W_i(s)$ das Gewicht aus dem s-ten Lernschritt, P die Lerneingabe an das Netz, in diesem Fall eine Zufallszahl aus $[a, b]$, und es gilt: $0 < \eta \leq 1$. Somit ist der Erwartungswert von W beschränkt und der Erwartungswert der Veränderung nach der Zeit ist Null (Rojas, 1996):

$$E\left(\frac{\partial W}{\partial t}\right) = 0 \ .$$

Wegen

$$E\left(\frac{\partial W}{\partial t}\right) = \eta\left(\frac{a+b}{2} - E(W)\right),$$

gilt aber

$$E(W) = \frac{a+b}{2}.$$

Mit diesen Überlegungen ist es jetzt möglich, den allgemeinen eindimensionalen Fall zu lösen. Seien m Neuronen in einer eindimensionalen Kette mit Gewichten W_1, W_2,..., W_n gegeben. Diese sollen nach Kohonen auf dem Intervall $[a, b]$ kartiert werden. Ferner gelte $a < W_1 < W_2 < ... < W_n < b$ und die W_i seien gleichmäßig verteilt, vgl. Abb. 15.

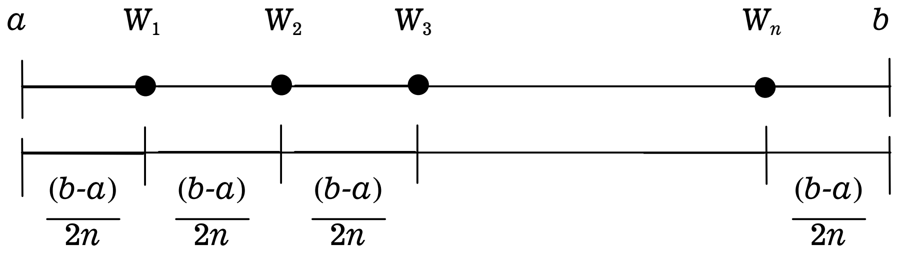


Abb. 15: Neuronenverteilung im Intervall $[a, b]$

Weil die Verteilung der Netzgewichte stabil ist, werden sich die Erwartungswerte der Gewichte im Intervall $[a, b]$ einpendeln (Rojas, 1996). Kohonen (2001) zeigt, daß dieses Einpendeln fast sicher eintritt. Dabei wird die Markov-Eigenschaft des Lernprozeß von Kohonen ausgenutzt. Insbesondere die Eigenschaft der absorbierenden Zustände, für welche der Übergang zu sich selbst mit Wahrscheinlichkeit eins eintritt. Nach Orey (1971) wird ein Zustand, beginnend von einem zufällig initialisierten Zustand, durch beliebige Eingaben mit positiver Wahrscheinlichkeit erreicht. Der absorbierende Zustand wird für $t \to \infty$ fast sicher erreicht.
Da die Erwartunsgswerte $E(W_i)$ beschränkt sind, und das Intervall nicht verlassen wird, gilt für $i = 1,....,m$:

$$E\left(\frac{\partial W_i}{\partial t}\right) = 0. \qquad (2.17)$$

Der Lernalgorithmus verändert die Lage der Gewichte im Intervall nicht.
Sind die Werte von $E(W_i)$ dergestalt gleichmäßig im Intervall verteilt, daß die Anziehung von links und rechts in Waage ist, gilt (2.17). Dieses ist dadurch bedingt, daß sich die Erwartungswerte $E(W_i)$ der Gewichte ausdrücken lassen gemäß

$$E(W_i) = a + (2i-1)\frac{b-a}{2} \; . \tag{2.18}$$

An dieser Stelle sei ausdrücklich darauf hingewiesen, daß die Nachbarschaft der Neuronen in der Kette keine Rolle spielt. Nur die nachfolgenden Bedingungen müssen erfüllt sein:

i) Die Netzgewichte müssen während der Lernphase im Intervall $[a, b]$ bleiben.
ii) Die Monotonie der zum Start initialisierten Gewichte darf nicht verletzt werden.

2.6.2 Einbeziehen der Nachbarschaft der Neuronen

Anders als bisher wird im weiteren die Nachbarschaft der Gewichte berücksichtigt. Auch hier wird der eindimensionale Fall betrachtet. Jedes Neuron besitze den Radius 1. Zur besseren Veranschaulichung sei ein Netz mit zwei Neuronen und Gewichten W_1 und W_2 aus $[-1 , 1]$ betrachtet. Es gilt: $-1 < W_1 < W_2 < 1$. Die Nachbarschaftsfunktion d_{ij*} sei gegeben durch:

$$d_{ij*} = \begin{cases} 1/2 & \text{falls } i = j-1 \quad \text{oder} \quad i = j+1 \\ 0 & \text{sonst} \end{cases} \; .$$

Mit obigen Annahmen, und weil die Gewichte W_1 und W_2 das Intervall $[-1,1]$ nicht verlassen können, gilt:

$$E(W_1) < E(W_2) \; .$$

Die Ungleichung besitzt nur Gültigkeit, wenn die Lernrate η klein genug ist.
Es gilt:

$$\begin{aligned} E\left(\frac{\partial W_1}{\partial t}\right) &= \frac{\eta}{2}\left(\left(0 - E(W_1)\right) - \left(E(W_1)+1\right)\right) + \frac{\eta}{2}\frac{1}{2}(1-0) \\ &= -\eta E(W_1) - \frac{\eta}{4} \end{aligned},$$

mit $E\left(\frac{\partial W_1}{\partial t}\right) = 0$ gilt dann $W_1 = 1/4$. Die Anziehung des zweiten Neurons auf seinen Nachbarn ist mit dem Term $(\eta/4)(1-0)$ gegeben. Die Anziehung ist mit der Lernrate η und der Nachbarschaftsfunktion d_{ij*} multipliziert. Ferner ist sie proportional zur halben Länge des Intervalls $[0,1]$. Eine analoge Berechnung gilt für W_2.
Welche Schlußfolgerung läßt sich aus diesem Sachverhalt ziehen? Die gleichmäßige Verteilung des Netzes mit nichtgekoppelten Neuronen bei einer Nachbarschaft mit Radius 1 geht verloren. Die Gewichte der Neuronen bewegen sich aufeinander zu und verteilen sich ungleichmäßig im Definitionsbereich. Die Parameter, die dies beeinflussen, sind der Radius und die Nachbarschaftsfunktion d_{ij*}. Ein größerer Radius zieht die Neuronen stärker zum Intervallzentrum. Somit läßt sich zusammenfassend sagen, daß zunächst eine breite Nachbarschaft (großer Radius) gewählt werden muß. Der Radius muß im Lernverlauf sukzessive so lange kleiner werden, bis schließlich nur noch einzelne Neuronen bewegt, also nur einzelne Gewichte verändert werden.
Im Zweidimensionalen läßt sich das durch Abb. 16 veranschaulichen. Dort ist zu erkennen, daß Neuronen am Rand des Gitters weniger Nachbarn besitzen und daher eine nach außen gerichtete „Anziehung“ auf sie wirkt. Bei einer zu starken Kopplung der Neuronen verhalten sich diese wie ein Punkt. Das komplette Netz wird zum Zentrum bewegt. Bei einer graduellen Verringerung der Kopplung kann die äußere Anziehung wirken und das Netz sich entfalten (Rojas, 1996).

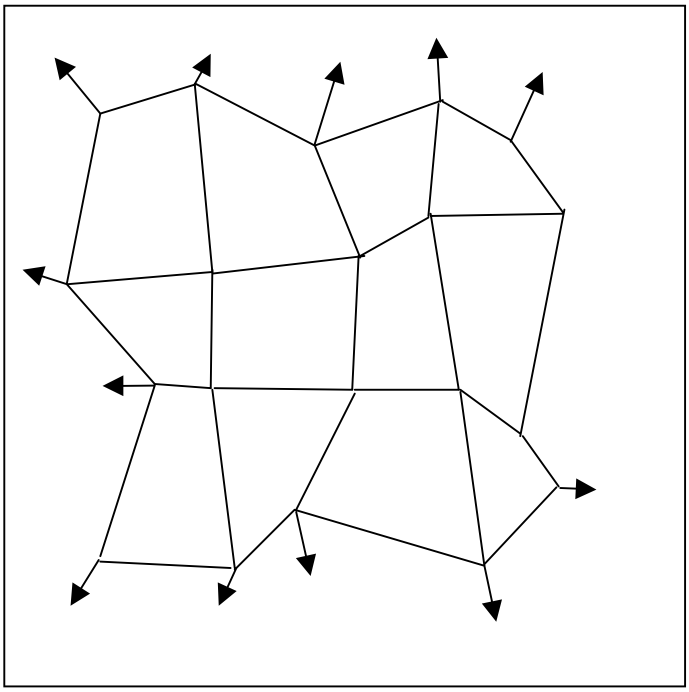


Abb. 16: Zweidimensionales Gitter mit möglicher Entfaltung (vgl. Rojas, 1996).

3 Zustandsraummodelle, der Kalman-Filter und strukturelle Zeitreihenmodelle

Dieses Kapitel führt die theoretischen Grundlagen des in Kapitel 5 auf das Ausgangsproblem spezifizierten Zeitreihenmodells ein. Das grundlegende Element zur Modellierung von strukturellen Zeitreihendaten sind die im ersten Abschnitt vorgestellten Zustandsraummodelle. Läßt sich ein Zeitreihenmodell in eine solche Form bringen, kann der Kalman-Filter angewandt werden. Dieser im zweiten Abschnitt eingeführte Algorithmus läßt sich zur Prognose einsetzen. Die folgenden Abschnitte befassen sich mit der Schätzung von Parametern des postulierten Modells. Abschnitt 3.4 stellt die strukturellen Zeitreihenmodelle und ihre Darstellung in Zustandsraumform vor.

3.1 Zustandsraummodelle

Damit Zeitreihenmodelle als Zustandsraummodelle geschrieben werden können, bedarf es der Einführung der sogenannten Zustandsraumform (ZRF). Die ZRF ist ein Instrument zur Handhabung einer weiten Spanne von Zeitreihenmodellen. Sobald ein Modell in die ZRF überführt wurde, kann der Kalman-Filter (vgl. Abschnitt 3.2) zur Prognose oder Glättung von Zeitreihen eingesetzt werden. Sind die Störgrößen des Modells normalverteilt, erlaubt der Kalman-Filter darüberhinaus eine nützliche Zerlegung der Likelihoodfunktion (Harvey, 1995).
Zur Definition der Zustandsraumform sei y_t ein $N \times 1$ Vektor der beobachteten Variablen zum Zeitpunkt t. Diese Variablen sind mit einem $m \times 1$ Vektor α_t, dem sogenannten Zustandsvektor, der den aktuellen Zustand des Modells beschreibt, durch die Beobachtungsgleichung

$$y_t = Z_t \alpha_t + d_t + \varepsilon_t, \quad t = 1,...,T^* \tag{3.1}$$

verbunden. Darin bezeichne Z_t eine nichtstochastische $N \times m$ Matrix, d_t einen nichtstochastischen $N \times 1$ Vektor und ε_t ein $N \times 1$ Vektor seriell unkorrellierter Störgrößen mit

$$E(\varepsilon_t) = 0 \quad \text{und} \quad Var(\varepsilon_t) = H_t. \tag{3.2}$$

Im allgemeinen sind die Elemente des Zustandsvektor α_t nicht beobachtbar. Jedoch wird unterstellt, daß sie durch einen Markov-Prozeß erster Ordnung beschrieben werden können. Die Markov-Eigenschaft besagt, daß die gesamte Information für die Zukunft bereits in der Gegenwart enthalten ist. Wie es zum gegenwärtigen Zustand kam, ist dabei nicht wesentlich (vgl. Schlittgen und Schreitberg, 1997). Formal:

$$\alpha_t = T_t \alpha_{t-1} + c_t + R_t \eta_t \,, \qquad t = 1,...,T^* \,, \tag{3.3}$$

wobei T_t eine $m \times m$ Matrix, c_t ein $m \times 1$ Vektor, R_t eine $m \times g$ Matrix und η_t ein $g \times 1$ Vektor unkorrellierter Störgrößen ist. T_t, R_t und c_t sind nichtstochastisch. Für die Störgröße η_t gilt:

$$E(\eta_t) = 0 \quad \text{und} \quad Var(\eta_t) = Q_t \,. \tag{3.4}$$

Die Gleichung (3.3) wird als Übergangsgleichung bezeichnet. Daß die Matrix R_t vor dem Störterm ins Modell aufgenommen wird, ist bis zu einem gewissen Grad arbiträr. Der Störterm η_t kann alternativ als $\eta_t{}^* = R_t \eta_t$ definiert werden, so daß sich für die Kovarianzmatrix ergibt: $Var(\eta_t{}^*) = R_t Q_t R_t{}'$.
Die Definition des Zustandsraumsystems wird, neben den oben beschriebenen Annahmen an die Störgrößen und Matrizen, durch zwei weitere Annahmen vervollständigt:

(i) Der Zustandsvektor α_t zum Zeitpunkt $t = 0$ hat den Erwartungswert a_0 und die Kovarianzmatrix P_0. Formal:

$$E(\alpha_0) = a_0 \quad \text{und} \quad Var(\alpha_0) = P_0 \,. \tag{3.5}$$

(ii) Die Störgrößen ε_t und η_t sind in allen Zeitperioden miteinander sowie mit α_0 unkorrelliert, d. h.

$$E(\varepsilon_t \eta_s') = 0 \quad \forall s,t = 1,...,T^* \tag{3.6}$$

und

$$E(\varepsilon_t \alpha_0') = 0 \text{ und } E(\eta_t \alpha_0') = 0 \,, \ \forall\, t = 1,...,T^*. \tag{3.7}$$

Die Matrizen Z_t, d_t und H_t der Beobachtungsgleichung und die Matrizen T_t, c_t, R_t und Q_t der Übergangsgleichung bezeichnet man als Systemmatrizen. Diese werden als nichtstochastisch vorausgesetzt. Die Veränderung dieser Matrizen über die Zeit ist dabei deterministisch. Das System ist also linear und y_t kann als Linearkombination der aktuellen und vergangenen Werte von ε_t und η_t sowie dem Zustandsvektor α_0 geschrieben werden ($\forall\, t = 1,...,T^*$). Ändern sich die Systemmatrizen über

die Zeit nicht, so spricht man von einem zeitinvarianten Modell. Wie in Harvey (1995) beschrieben, ist die Übergangsgleichung in einem solchen zeitinvarianten Modell ein vektorautoregressiver Prozeß 1. Ordnung. So ist beispielsweise ein AR(1) Prozeß mit weißem Rauschen (WR) ein zeitinvariantes Zustandsraummodell. Der Zustandsvektor α_t ist für jedes Modell durch die Konstruktion des Modells determiniert. Technisch betrachtet besteht das Ziel der Formulierung im Zustandsraum darin, den Zustandsvektor so zu formulieren, daß er alle relevanten Informationen des Systems zum Zeitpunkt t enthält. Dabei ist auf die geringst mögliche Zahl an Elementen zu achten.
Die Zustandsraumform ist nicht eindeutig.

Beispiel
Gegeben sei der AR(2) Prozeß $y_t = \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \xi_t$. Dann ist die ZRF zum einen gegeben durch:

$$y_t = (1 \quad 0)\alpha_t ,$$

$$\alpha_t = \begin{bmatrix} y_t \\ \varphi_2 y_{t-1} \end{bmatrix} = \begin{bmatrix} \varphi_1 & 1 \\ \varphi_2 & 0 \end{bmatrix} \alpha_{t-1} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \xi_t .$$

Zum anderen läßt sich der Prozeß aber auch darstellen gemäß

$$y_t = (1 \quad 0)\alpha_t^* ,$$

$$\alpha_t^* = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix} = \begin{bmatrix} \varphi_1 & \varphi_2 \\ 1 & 0 \end{bmatrix} \alpha_{t-1}^* + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \xi_t .$$

Harvey (1995) zeigt, daß sich alle ARMA-Modelle in ZRF überführen lassen.

3.2 Der Kalman-Filter

Der Kalman-Filter ist ein rekursiver Algorithmus zur Bestimmung des optimalen Schätzers des Zustandsvektors unter Verwendung der zum Zeitpunkt t zur Verfügung stehenden Information (Kalman, 1960). Der Filter kann ausschließlich für Modelle angewendet werden, die eine ZRF besitzen. Die Systemmatrizen a_0 und P_0 sind für alle Zeitpunkte t bekannt. Insbesondere in den Ingenieurwissenschaften ist der Kalman-Filter weit verbreitet, weil er „Online“-Schätzungen erlaubt.
Auf den ersten Blick erscheint sich beispielsweise der Nutzen einer derartigen Prozedur in ökonomischen Anwendungen als beschränkt zu erweisen. Neue Beobachtungen erscheinen eher in unregelmäßigen Abständen. Der Schwerpunkt liegt auf der Vorhersage von zukünftigen Beobachtungen, basierend auf einer gegebenen Stichprobe. Der Zustandsvektor hat nicht immer eine ökonomische Interpretation und in Fällen, in denen er es doch hat, ist es günstiger, seinen Wert zu einem bestimmten Zeitpunkt zu schätzen und dabei alle in der Stichprobe vorhandene Information zu verwenden als nur einen Teil davon. Wie Kalman (1960) zeigt, erlaubt der Kalman-Filter sowohl eine Prognose, als auch die als Glättung bezeichnete Berechnung des Zustandsvektors aus der gesamten Stichprobe.
Eine weitere Begründung für die zentrale Rolle des Kalman-Filters liegt in der Berechenbarkeit der Likelihood-Funktion über die sogenannte Prädiktionsfehlerzerlegung. Voraussetzung dafür sind normalverteilte Störgrößen sowie die Normalverteilung von α_0 (vgl. Harvey, 1990).
Dann kann eine Standardeigenschaft der multivariaten Normalverteilung ausgenutzt werden, um die Verteilung von α_t bedingt auf die Information zum Zeitpunkt t, $\forall\, t = 1,...,T^*$, rekursiv zu berechnen. Nachdem der Filter berechnet worden ist, kann ebenfalls gezeigt werden, daß der Erwartungswert der bedingten Verteilung von α_t ein optimaler Schätzer für α_t im Sinne des MSE-Kriteriums ist. Ohne Normalverteilungsannahme ist nicht mehr gewährleistet, daß der Kalman-Filter den bedingten Erwartungswert des Zustandsvektor berechnen kann.

3.2.1 Allgemeine Form des Kalman-Filters

Im Zustandsraummodell (3.1) - (3.4) sei a_{t-1} der optimale Schätzer von α_{t-1}, basierend auf den Beobachtungen bis einschließlich y_{t-1}. Ferner bezeichne P_{t-1} die $m \times m$ Kovarianzmatrix des Schätzfehlers. Damit ist:

$$P_{t-1} = E\left[(\alpha_{t-1} - a_{t-1})(\alpha_{t-1} - a_{t-1})'\right] . \qquad (3.8)$$

Ist a_{t-1} und P_{t-1} gegeben, so ist der optimale Schätzer von α_t gegeben durch:

$$a_{t|t-1} = T_t a_{t-1} + c_t \,. \tag{3.9}$$

Die Kovarianzmatrix des Schätzfehlers wird demnach folgendermaßen bestimmt:

$$P_{t|t-1} = T_t P_{t-1} T_t' + R_t Q_t R_t', \quad \text{mit } t = 1,\dots,T^*. \tag{3.10}$$

Die Gleichungen (3.9) und (3.10) werden Prognosegleichungen genannt. Die Prognoseschritte liegen dabei zeitlich vor der Beobachtung von y_t. Sobald eine neue Beobachtung eintritt, kann der Schätzer $a_{t|t-1}$ für α_t aktualisiert werden. Die Korrekturgleichung für (3.9) und (3.10) lautet:

$$a_t = a_{t|t-1} + P_{t|t-1} Z_t' F_t^{-1} \left(y_t - Z_t a_{t|t-1} - d_t \right) \tag{3.11}$$

und

$$P_t = P_{t|t-1} - P_{t|t-1} Z_t' F_t^{-1} Z_t P_{t|t-1} \,. \tag{3.12}$$

Es wird unterstellt, daß die Inverse von F_t existiert. Andernfalls wird diese durch die Pseudo-Inverse ersetzt. Nach Harvey (1990) wird allgemein die positive Definitheit von F_t unterstellt. F_t bestimmt sich gemäß

$$F_t = Z_t P_{t|t-1} Z_t' + H_t, \quad t = 1,\dots,T^*. \tag{3.13}$$

Der gesamte Kalman-Filter ist gegeben durch die Gleichungen (3.9) und (3.10) des Prognose- oder Prädiktionsschritts sowie durch die Gleichungen des Korrekturschritts (3.11) und (3.12).
In der Literatur wird der Korrekturschritt mittels des sogenannten Kalman-Gains K_t dargestellt. Für den Schätzer a_t aus (3.11) ergibt sich:

$$a_t = a_{t|t-1} + K_t \left(y_t - Z_z a_{t|t-1} - d_t \right) \,, \tag{3.14}$$

mit

$$K_t = P_{t|t-1} Z_t' F_t^{-1} \,. \tag{3.15}$$

Die korrigierte Kovarianzmatrix für den Zustandsvektor ist dann gegeben durch:

$$P_t = (I - K_t Z_t) P_{t|t-1} + R_{t+1} Q_{t+1} R'_{t+1} \,. \tag{3.16}$$

(vgl. Harvey (1990), Schlittgen und Streitberg (1997) oder Stier (2001)).
Eine Prognose für den Zustandsvektor zum Zeitpunkt $T^* + 1$ läßt sich aus den Beziehungen

$$a_{T^*+1|T^*} = T_{T^*+1} a_{T^*} + c_{T^*+1} \,, \tag{3.17}$$

bzw.

$$y_{T^*+1} = Z_{T^*+1} a_{T^*+1|T^*} + d_{T^*+1} \,. \tag{3.18}$$

ableiten.
Mehr-Schritt-Prognosen ergeben sich durch wiederholte Anwendung der Ein-Schritt-Prognose (3.17) und (3.18) unter Auslassung der Korrekturschritte. Die Kovarianzmatrix dieser Mehr-Schritt-Prognose lautet:

$$P_t = T_{T^*+1} (I - K_t Z_t) P_{t|t-1} T'_{T^*+1} + R_{t+1} Q_{t+1} R'_{t+1} \,. \tag{3.19}$$

Zusammenfassend läßt sich sagen, daß bei gegebenen Startbedingungen a_0 und P_0 der Kalman-Filter optimale Schätzwerte für den Zustandsvektor liefert, wenn jeweils eine neue Beobachtung verfügbar ist. Wenn alle T^* Beobachtungen bearbeitet worden sind, enthält der Schätzwert a_{T^*} alle notwendigen Informationen für Prognosen zukünftiger Beobachtungen.

Beispiel (vgl. Schlittgen und Streitberg, 1997)
Gegeben sei ein MA(1)-Prozeß $y_t = \varepsilon_t - \theta\varepsilon_{t-1}$. Dieser läßt sich in ZRF ausdrücken gemäß

$$\alpha_t = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_{t-1} \\ y_{t-1} \end{bmatrix} + \begin{bmatrix} 1 \\ -\theta \end{bmatrix} \varepsilon_t \,.$$

Die Beobachtungsgleichung lautet:

$$y_t = [1 \quad 0]\alpha_t .$$

Alle Systemmatrizen sind zeitunabhängig. Damit ergibt sich:

$$T_t = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \; R_t = \begin{bmatrix} 1 \\ -\theta \end{bmatrix}, \; Z_t = [1 \quad 0], \; H_t = \sigma^2 , \; Q_t = 0 \quad \forall t.$$

Der Prädiktionsschritt lautet:

$$a_{t+1|t} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_t \\ y_{t+1} \end{bmatrix},$$

$$\begin{aligned} P_{t+1|t} &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} P_{t|t-1} \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} 1 \\ -\theta \end{bmatrix} \sigma^2 [1 \quad -\theta] \\ &= \begin{bmatrix} p_{22}(t) & 0 \\ 0 & 0 \end{bmatrix} + \sigma^2 \begin{bmatrix} 1 & -\theta \\ -\theta & \theta^2 \end{bmatrix} \quad , \end{aligned}$$

wobei $P_{t|t-1} = [p_{ij}(t)]$ gesetzt wurde.
Die Prognose von y_{t+1} ist demnach die letzte Schätzung von $-\theta\varepsilon_t$. ε_{t+1} wird durch seinen Erwartungswert Null geschätzt.
Für den Korrekturschritt ergibt sich:

$$\begin{aligned} K_{t+1} &= P_{t+1|t} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \left([1 \quad 0] P_{t+1|t} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 0 \right)^{-1} \\ &= \begin{bmatrix} 1 \\ -\sigma^2 \, \theta / \left(p_{22}(t) + \sigma^2 \right) \end{bmatrix} \end{aligned}$$

und

$$\begin{aligned} P_{t+1|t} &= \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 \\ -\sigma^2 \, \theta \Big/ \left(p_{22}(t) + \sigma^2 \right) \end{bmatrix} [1 \quad 0] \right) \begin{bmatrix} p_{22}(t) + \sigma^2 & -\sigma^2\theta \\ -\sigma^2\theta & \sigma^2\theta^2 \end{bmatrix} \\ &= \frac{\sigma^2\theta^2 p_{22}(t)}{p_{22}(t) + \sigma^2} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}. \end{aligned}$$

Rekursives Auflösen von $p_{22}(t+1) = \sigma^2\theta^2 p_{22}(t)/(p_{22}(t)+\sigma^2)$ ergibt mit dem Startwert $p_{22}(0) = p_0$:

$$p_{22}(t+1) = \sigma^2 \frac{p_0 \theta^{2(t+1)}}{p_0\left(1+\theta^2+\cdots+\theta^{2t}\right)+\sigma^2} .$$

Der resultierende Kalman-Gain hat die Gestalt:

$$K_{t+1} = \begin{bmatrix} 1 \\ -\theta \Big/ \left(\sum_{u=0}^{t+1} \theta^{2u} + \sigma^2\right) \end{bmatrix} .$$

Die korrigierte Schätzung von y_{t+1} lautet dann:

$$a_{t+1|t} = \begin{bmatrix} a_{t+1|t} \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ -\theta \Big/ \left(\sum_{u=}^{t+1} \theta^{2u} + \sigma^2\right) \end{bmatrix} \left(y_{t+1} - a_{t+1|t}\right).$$

Bevor auf die Initialisierung und die Interpretation des Filters eingegangen wird, wird an dieser Stelle, zum besseren mathematischen Verständnis, eine Beweisskizze der Herleitung des Kalman-Filters eingeschoben.

3.2.2 Herleitung des Kalman-Filters

Sei der Zustandsvektor α_0 normalverteilt mit Erwartungswert a_0 und Kovarianzmatrix P_0. Seien die Störgrößen ε_t und η_t multivariat normalverteilt für $t = 1,...,T^*$. Ferner seien ε_t, η_t und α_0 stochastisch unabhängig. Der Zustandsvektor zum Zeitpunkt $t = 1$ ist gegeben durch:

$$\alpha_1 = T_1\alpha_0 + c_1 + R_1\eta_1 .$$

α_1 läßt sich also als Linearkombination zweier vektorieller, normalverteilter Zufallsvariablen und einem Vektor von Konstanten beschreiben. Also ist α_1 selbst normalverteilt mit Erwartungswert

$$a_{1|0} = T_1 a_0 + c_1 \tag{3.20}$$

und Kovarianzmatrix

$$P_{1|0} = T_1 P_0 T_1' + R_1 Q_1 R_1'. \tag{3.21}$$

Dabei gibt die Notation $a_{1|0}$ den Erwartungswert der Verteilung von α_0, bedingt auf die Information zum Zeitpunkt $t = 0$ an.
Um die Verteilung von α_1 bedingt auf y_1 zu erhalten, sind folgende Beziehungen zu bestimmen:

$$\alpha_1 = a_{1|0} + (\alpha_1 - a_{1|0}), \tag{3.22}$$

$$y_1 = Z_1 a_{1|0} + d_1 + Z_1(\alpha_1 - a_{1|0}) + \varepsilon_1. \tag{3.23}$$

(3.23) ergibt sich dabei durch Umstellen der Systemgleichung. Mit (3.22) und (3.23) folgt, daß $[\alpha_1' \quad y_1']'$ eine multivariate Normalverteilung mit Erwartungswert

$$\begin{bmatrix} a_{1|0} \\ Z_1 a_{1|0} + d_1 \end{bmatrix}$$

und Kovarianzmatrix

$$\begin{bmatrix} P_{1|0} & P_{1|0} Z_1' \\ Z_1 P_{1|0} & Z_1 P_{1|0} Z_1' + H_1 \end{bmatrix}$$

besitzt.
Aus den Eigenschaften der multivariaten Normalverteilung folgt, daß die bedingte Verteilung von α_1 gegeben y_1 ebenfalls multivariat normal ist mit Erwartungswert

$$a_1 = a_{1|0} + P_{1|0} Z_1' F_1^{-1} (y_1 - Z_1 a_{1|0} - d_1) \tag{3.24}$$

und Kovarianzmatrix

$$P_1 = P_{1|0} - P_{1|0} Z_1' F_1^{-1} Z_1 P_{1|0}, \tag{3.25}$$

wobei F_1 gegeben ist gemäß (3.13) mit $t = 1$. Die wiederholte Anwendung von (3.22) bis (3.25) für $t = 2,...,T^*$ ergibt die Kalman-Filter-Rekursion gemäß (3.9) und (3.10) bzw. (3.11) - (3.13).
Diese Herleitung ermöglicht es, a_t und P_t als den Erwartungswert bzw. die Kovarianzmatrix der bedingten Verteilung von α_t aufzufassen. D. h. es gilt:

$$a_t = E(\alpha_t | y_t) \tag{3.26}$$

und

$$P_t = E\left[(\alpha_t - E(\alpha_t))(\alpha_t - E(\alpha_t))'\right],$$

wobei der Erwartungswert der bedingten Verteilung von α_t zum Zeitpunkt t zu bilden ist. Der bedingte Erwartungswert ist der Schätzwert für α_t mit minimalem mittleren quadratischen Fehler. Diese Eigenschaft läßt sich wie folgt zeigen:

Für jeden Prognosewert $a_{T+\ell|T^*}$, der mit den Informationen zum Zeitpunkt T^* berechnet wird, läßt sich der Schätzfehler zerlegen in:

$$\alpha_{T^*+\ell} - a_{T+\ell|T^*} = \left(\alpha_{T^*+\ell} - E\left(\alpha_{T^*+\ell} | Y_{T^*}\right)\right) + \left(E\left(\alpha_{T^*+\ell} | Y_{T^*}\right) - a_{T^*+\ell}\right),$$

wobei Y_{T^*} die Informationsmenge $y_{T^*}, y_{T^*-1},...$ kennzeichnet.
Da der zweite Term auf der rechten Seite zum Zeitpunkt T^* fest ist, folgt durch Quadrieren der gesamten Gleichung und Erwartungswertbildung zum Zeitpunkt T, daß der Kreuzproduktterm entfällt. Somit verbleibt:

$$MSE\left(a_{T^*+\ell|T^*}\right) = Var\left(\alpha_{T^*+\ell|T^*} | Y_{T^*}\right) + \left(a_{T^*+\ell|T^*} - E\left(\alpha_{T^*+\ell|T^*} | Y_{T^*}\right)\right)^2.$$

Die bedingte Varianz von $\alpha_{T^*+\ell}$ hängt nicht von $a_{T^*+\ell|T^*}$ ab. Folglich ist der Schätzwert mit minimalen mittleren quadratischem Fehler (MMSE) von $\alpha_{T^*+\ell}$ durch den bedingten Erwartungswert (3.26) gegeben und eindeutig (vgl. auch Harvey, 1995).
Ferner kann gezeigt werden, daß der gesamte bedingte Erwartungswertvektor der Schätzer mit minimalem mittleren quadriertem Fehler für α_t ist. Die MSE Matrix eines beliebigen anderen Schätzeres kann als bedingte Kovarianzmatrix von α_t plus einer positiv semi-definiten Matrix geschrieben werden. Darüberhinaus ist der Schätzwert a_t unverzerrt. Weitere Eigenschaften sind bei Harvey (1990) nachzulesen.

Sind die Störgrößen im Zustandsraummodell nicht normalverteilt, führt der Kalman-Filter im allgemeinen nicht zu dem bedingten Erwartungswert des Zustandsvektors. Beschränkt man sich jedoch auf lineare Schätzfunktionen, dann minimiert der Schätzwert a_t den MSE auch weiterhin (Anderson und Moore, 1979).

3.3 Maximum-Likelihood-Schätzung

Der klassische Maximum-Likelihood-Ansatz unterstellt unabhängig und identisch verteilte Beobachtungen. Die gemeinsame Dichtefunktion ist dabei gegeben durch:

$$L(y;\Psi) = \prod_{t=1}^{T} p(y_t), \tag{3.27}$$

wobei $p(y_t)$ die Dichtefunktion des t-ten Beobachtungsvektor ist. Sobald die Beobachtungen getätigt wurden, wird (3.27) als Likelihoodfunktion aufgefaßt und der ML-Schätzer wird durch Maximierung dieser Funktion bezüglich Ψ ermittelt. Im Vektor Ψ sind die unbekannten Parameter der Systemmatrizen des Zustandsraummodells aufgeführt. Diese Parameter werden als Hyperparameter bezeichnet. Die ML-Schätzung der Hyperparameter kann durch die Anwendung des Kalman-Filters zur Konstruktion der Likelihoodfunktion und durch eine sich anschließende Maximierung dieser Funktion ausgeführt werden.

In einem Zeitreihenmodell sind die Beobachtungen jedoch im allgemeinen abhängig. Daher kann (3.27) nicht angewandt werden. Die gemeinsame Dichtefunktion ergibt sich demnach:

$$L(y;\Psi) = \prod_{t=1}^{T} p(y_t | Y_{t-1}). \tag{3.28}$$

In (3.28) beschreibt $p(y_t | Y_{t-1})$ die Verteilung von y_t bedingt auf die Informationsmenge zum Zeitpunkt t-1, also $Y_{t-1} = (y_{t-1}, y_{t-2}, \ldots, y_1)'$. Sind der Störterm und der ursprüngliche Zustandsvektor aus (3.1) und (3.2) normalverteilt, ist auch y_t bedingt auf Y_{t-1} normalverteilt. Der Erwartungswert und die Kovarianzmatrix des bedingten Modells sind direkt aus der Herleitung des Kalman-Filters zu bestimmen. So gilt für den Zustandsvektor α_t, daß dieser bedingt auf Y_{t-1} eine Normalverteilung mit Erwartungswert $a_{t|t-1}$ und Kovarianzmatrix $P_{t|t-1}$ besitzt. Wird die Systemgleichung des Zustandsraummodells geschrieben als

$$y_t = Z_t a_{t|t-1} + Z_t(\alpha_t - a_{t|t-1}) + d_t + \varepsilon_t,$$

so ist der Erwartungswert von y_t gegeben durch:

$$E(y_t) = \overline{y}_{t|t-1} = Z_t a_{t|t-1} + d_t \,.$$

Die Kovarianzmatrix ist durch (3.13) gegeben.
Mit diesen Voraussetzungen ist die Likelihoodfunktion von (3.28) nach Logarithmieren gegeben durch:

$$\log L(\psi) = -\frac{NT}{2}\log 2\pi - \frac{1}{2}\sum_{t=1}^{T}\log|F_t| - \frac{1}{2}\sum_{t=1}^{T} v_t' F_t^{-1} v_t \,, \tag{3.29}$$

mit $v_t = y_t - \overline{y}_{t|t-1}$, $t = 1,...,T^*$. v_t bezeichnet den Prognosefehler. Die obige Gleichung (3.29) wird auch als Zerlegung durch Prognosefehler-Form der Likelihoodfunktion bezeichnet.
Durch diese Kontruktion der Likelihoodfunktion wird das Problem der Initialisierung des Kalman-Filters auf die Schätzung der Hyperparameter übertragen. Unterschiedliche Methoden der Initialisierung des Filters führen zu Likelihoodfunktionen, die zwar asymptotisch gleich sind, bei kleinen Stichprobenumfängen jedoch stark voneinander abweichen. Im günstigsten Fall folgt der Zustandsvektor α_t einem stationären stochastischen Prozeß. Dadurch kann der Kalman-Filter mit dem Erwartungswert und der Varianz der unbedingten Verteilung von α_t initialisiert werden (Harvey, 1995). Zudem liefert die Zerlegung durch den Prognosefehler die exakte Likelihoodfunktion. Für nichtstationäre Zustandsvektoren kann eine Likelihoodfunktion unter Zuhilfenahme der Prognosefehler nur formuliert werden, wenn Vorinformationen in der Form verfügbar sind, daß der Zustandsvektor eine Verteilung mit bekanntem Erwartungswert und beschränkter Kovarianzmatrix besitzt.

Fehlende Beobachtungen

Fehlende Beobachten bereiten Zustandsraummodelle keine Schwierigkeiten. Wenn zu einem bestimmten Zeitpunkt $t = \tau$ ein Beobachtungswert fehlt, sind die Kalman-Filter-Korrekturgleichungen redundant. Es folgt:

$$a_\tau = a_{\tau|\tau-1} \quad \text{und} \quad P_\tau = P_{\tau|\tau-1} \,.$$

Die Prognosegleichungen führen zu einer zweistufigen Prognose. Die Korrekturgleichungen werden auf die beschriebene Weise angewendet, sobald die Beobachtung $y_{\tau+1}$ zur Verfügung steht.

Die Likelihoodfunktion wird mittels Prognosefehler aus dem Kalman-Filter berechnet. Mangels Daten können die Prognosefehler in τ nicht bestimmt werden. Ausführlich wird das Verhalten des Kalman-Filters bei Datenirregularitäten in Harvey (1990) behandelt.

3.4 Strukturelle Zeitreihenmodelle

Der klassische Ansatz in der Zeitreihenmodellierung unterstellt ein Modell mit vier Komponenten:

$$\begin{array}{lclclclcl} \textit{Zeitreihe} & = & \textit{Trend} & + & \textit{Zyklus} & + & \textit{Saison} & + & \textit{Störgröße} \\ y_t & = & T_t & + & z_t & + & s_t & + & u_t, \quad t = 1,\ldots,T^*\,; \end{array}$$

bzw.

$$\begin{array}{lclclcl} \textit{Zeitreihe} & = & \textit{Glatte-} & + & \textit{Saison-} & + & \textit{irreguläre Komponente} \\ y_t & = & g_t & + & s_t & + & u_t, \qquad t = 1,\ldots,T^*. \end{array}$$

Dieser Ansatz unterstellt, daß die einzelnen Komponenten entweder durch deterministische Funktionen der Zeit beschreibbar oder als Resultat eines bestimmten Algorithmus definierbar sind. Letzteres trifft beispielsweise auf die Störgöße (oder irreguläre Komponente) zu, die bei einigen Saisonbereinigungsverfahren rechnerisch als „Restkomponente“ bestimmt wird. Im vorliegenden Fall wird die Störgröße nicht modelliert.

Die strukturellen Zeitreihenmodelle, die hauptsächlich auf Harvey (1990) zurückgehen, basieren auf diesem traditionellen Komponentenmodell. Hinzu kommen jedoch die folgenden Eigenschaften und Erweiterungen (vgl. Stier, 2001):

- Es handelt sich grundsätzlich um stochastische, nicht um deskriptive Zeitreihenmodelle. Die einzelnen Komponenten sind stochastische Prozesse.
- Strukturelle Komponentenmodelle lassen sich jeweils als spezielle ARIMA-Modelle begreifen.

Ein strukturelles Modell ist dann vollständig gegeben, wenn seine einzelnen Komponenten stochastisch spezifiziert sind.

Das klassische Modell kann auch als Regression aufgefaßt werden. Hierbei sind die erklärenden Variablen Zeittrends und saisonale Dummys. Die notwendige Flexibilität des Ansatzes kann erreicht werden, indem den Regressionskoeffizienten erlaubt

wird, im zeitlichen Verlauf gemäß vorher definierten Gesetzmäßigkeiten zu variieren.
Der Schlüssel für die Behandlung struktureller Zeitreihenmodelle ist die Zustandsraumform. Ist das Modell linear, kann mittels des Kalman-Filters der Zustand aktualisiert werden, sobald eine neue Beobachtung verfügbar ist.

Den Ausgangspunkt eines strukturellen Zeitreihenmodells bildet ein Regressionsmodell, in dem die erklärenden Variablen Funktionen der Zeit sind. Ein Modell mit Trend- und Saisonkomponente kann formuliert werden gemäß

$$y_t = \alpha + \beta_t + \sum_{j=1}^{s-1} \gamma_j z_{jt} + \varepsilon_t, \tag{3.30}$$

wobei α und β_t die zum Trend gehörenden Koeffizienten und die γ_j's saisonale Koeffizienten sind. Die saisonalen Koeffizienten addieren sich für s aufeinanderfolgende Zeitpunkte zu Null. Die Restriktion wird durch die Definition der Dummy-Variablen z_{jt}, für $j = 1,...,s$,

$$z_{jt} = \begin{cases} 1 & , \quad t = j, j+s, j+2s,... \\ 0 & , \quad t \neq j, j+s, j+2s,..., \\ -1 & , \quad t = s,2s,3s,... \end{cases} \tag{3.31}$$

erfüllt.
Das Modell kann durch Aufnahme weiterer Komponenten, wie z. B. einen Zyklus sowie durch Trendpolynome höherer Ordnung, erweitert werden. Die Besonderheit des Modells liegt darin, daß ausschließlich der Störterm ε_t stochastisch ist. Dabei werden $\varepsilon_1,...,\varepsilon_{T^*}$ als unabhängige, identisch normalverteilte Zufallsvariablen mit Erwartungswert Null und Varianz σ_ε^2 angenommen.

Einfache strukturelle Zeitreihenmodelle bestehen aus einer Trendkomponente und einem stochastischen Störterm. Die Störgröße kann als irreguläre Komponente in der Zeitreihe oder als Meßfehler aufgefaßt werden. Das Modell lautet:

$$y_t = \mu_t + \varepsilon_t , \quad t = 1,...,T^*, \tag{3.32}$$

wobei μ_t der Trend und ε_t ein weißes Rauschen ist. Mit anderen Worten: ε_t hat den Erwartungswert Null, die Varianz σ_ε^2 und ist mit jedem stochastischen Element in μ_t unkorreliert. Kurz schreibt man $\varepsilon_t \sim WR$.

Der Trend kann verschiedene Formen annehmen. Die einfachste Form ist der Random-Walk:

$$\mu_t = \mu_{t-1} + \eta_t \,. \tag{3.33}$$

In Gleichung (3.33) entspricht η_t weißem Rauschen mit Erwartungswert Null und Varianz σ_η^2. Ferner sind η_t und ε_t unkorreliert. Bei einem Random-Walk entfernt sich die Reihe zu einem Zeitpunkt zufällig von der vorherigen Beobachtung. Daher hat das strukturelle Zeitreihenmodell (3.32) ein Niveau, das sich im Zeitverlauf auf- oder abwärts bewegt. Ein wichtiges Charakteristikum des Modells ist das Varianzverhältnis $\sigma_\eta^2 / \sigma_\varepsilon^2$. Dieses beschreibt, inwieweit vergangene Beobachtungen berücksichtigt werden. Je größer σ_η^2 im Vergleich zu σ_ε^2 ist, desto mehr Beobachtungen werden berücksichtigt (vgl. Harvey, 1990).
Durch die Einführung eines Steigungsparameters in die Trendkomponente kann das obige Modell erweitert werden. Ein deterministischer linearer Trend ist gegeben durch

$$\mu_t = \alpha + \beta t \,, \quad t = 1,\ldots,T^*. \tag{3.34}$$

Wenn α und β als Random-Walks aufgefaßt werden, kann der Trend als stochastisch unterstellt werden. Dadurch bedingt, verläuft μ_t i. a. nicht stetig. Ein geeigneteres Modell ergibt sich, wenn direkt das aktuelle Niveau μ_t und nicht Ordinatenabschnitt α verwendet wird. Aus der rekursiven Bestimmung von μ_t aus

$$\mu_t = \mu_{t-1} + \beta \,, \quad t = 1,\ldots,T^*, \tag{3.35}$$

mit $\mu_0 = \alpha$, lassen sich die stochastischen Größen gemäß

$$\mu_t = \mu_{t-1} + \beta_{t-1} + \eta_t \,, \tag{3.36}$$

$$\beta_t = \beta_{t-1} + \zeta_t \tag{3.37}$$

einführen. Dabei bezeichnen η_t und ζ_t jeweils weißes Rauschen mit Erwartungswert Null und Varianz σ_η^2 bzw. σ_ζ^2. Die Störgrößen sind sowohl wechselseitig als auch mit ε_t aus (3.32) unkorreliert. Verdichtet ausgedrückt bildet (3.33), (3.36) und (3.37) das lokal lineare Trendmodell. Dabei wird durch η_t dem Niveau des Trends ermöglicht, sich aufwärts oder abwärts zu bewegen. Die Steigung wird durch ζ_t beeinflußt. Je größer die Varianzen sind, desto größer sind tendenziell die stochastischen

Bewegungen im Trend. Im deterministischen Fall der Trendkomponente gilt: $\sigma_\eta^2 = \sigma_\zeta^2 = 0$.

Die Modelle, die bisher betrachtet wurden, bieten keine Möglichkeit, saisonale Effekte zu modellieren. Diese lassen sich analog zu (3.33) deterministisch festlegen. Dabei werden die saisonalen Effekte so restringiert, daß sie sich für s aufeinanderfolgende Zeitpunkte zu Null addieren. Mit (3.31) läßt sich der Saisoneffekt zum Zeitpunkt $t = s, 2s, 3s, \ldots$ schreiben als:

$$\sum_{j=1}^{s-1} \gamma_j z_{jt} = -\sum_{j=1}^{s-1} \gamma_j \,. \tag{3.38}$$

Bezeichnet man diesen Effekt mit γ_s, dann impliziert die Restriktion der saisonalen Effekte die Identität:

$$\sum_{j=1}^{s} \gamma_j = 0 \,. \tag{3.39}$$

Bezeichne γ_t den Saisoneffekt zum Zeitpunkt t, dann gilt wegen (3.39):

$$\sum_{j=0}^{s-1} \gamma_{t-j} = 0 \,. \tag{3.40}$$

Fügt man auf der rechten Seite der Gleichung (3.40) einen Zufallsterm ein, so verändert sich die Saisonkomponente über die Zeit:

$$\gamma_t = -\sum_{j=1}^{s-1} \gamma_{t-j} + \omega_t \,, \tag{3.41}$$

wobei ω_t ein weißes Rauschen mit Erwartungswert Null und Varianz σ_ω^2 ist.

Ein strukturelles Basismodell, das sowohl die langfristige Entwicklung als auch saisonale Schwankungen berücksichtigt, ist gegeben durch:

$$y_t = \mu_t + \gamma_t + \varepsilon_t \,, \quad t = 1,\ldots,T^* . \tag{3.42}$$

Der Trend μ_t in (3.42) ist gegeben durch (3.36) und (3.37) und γ_t ist gegeben durch (3.41). ε_t bezeichne ein weißes Rauschen. Ferner wird unterstellt, daß die Störgrößen der drei Komponenten unkorreliert sind. Das strukturelle Basismodell läßt sich dann als stochastische Verallgemeinerung des klassischen Komponentenmodells auffassen.

3.4.1 Pro und Kontra struktureller Zeitreihenmodelle

Zwischen dem klassischen ARIMA-Ansatz von Box-Jenkins (1976) und den strukturellen Zeitreihenmodellen, wie sie in den vorangegangenen Abschnitten behandelt wurden, besteht eine enge Beziehung. Überführt man die strukturellen Modelle in die sogenannte reduzierte Form (Harvey, 1990) ergibt sich sofort das korrespondierende ARIMA-Modell. Reduzierte Form bedeutet, daß sich die Modelle in Form einer einzigen Gleichung schreiben lassen. Dieser Sachverhalt gibt in der Literatur Anlaß zu kritischen Äußerungen bis hin zur provokanten Frage: Verwendet man ARIMA-Modelle oder strukturelle Modelle (Stier, 2001)? Hier ist nicht der Ort, alle Diskussionen zu behandeln, jedoch sollen die Kernpunkte kurz umrissen werden.
In der einführenden Arbeit von Harvey und Todd (1983) in die strukturellen Modelle führen sie aus, daß diese Modelle besonders im logischen Sinne ansprechend sind. Insbesondere führen diese Modelle zu ansprechenden Prognosefunktionen. ARIMA-Modelle werden von Harvey eher „abfällig" betrachtet. Das Identifikationsproblem bei ARIMA-Modellen steht zudem dabei im Mittelpunkt seiner Kritik. So schreibt er beispielsweise in Harvey (1990), Seite 98:

> *„the ARIMA class is not a particularly natural one. The components in a structural model on the other hand, are associated with particular features of the series, and therefore the chosen model is more likely to yield sensible predictions."*

P. Newbold erwidert darauf, daß die Selektion eines ARIMA-Modells sicherlich schwierig ist, oftmals wird diese Schwierigkeit jedoch stark übertrieben. In jedem Fall wird die Verwendung eines strukturellen Modells ebenfalls Schwierigkeiten in der gleichen Größenordnung aufweisen. Nur wenn ausschließlich das einfache strukturelle Modell verwendet wird, gilt dies nicht. Zitat:

> *„It is only when we stick to just the basic model that we have the attraction that it involves no model selection procedure whatsoever."*

In der Tat stellt sich nur dann kein Identifikationsproblem, wenn a priori ein einfaches strukturelles Modell verwendet wird (Stier, 2001).

Soweit zu den Diskussionen über die verschiedenen Ansätze. Weitere Stimmen sind in den Kommentaren von Ansley, Findley und Newbold zum Artikel von Harvey und Todd (1983) nachzulesen.

4 Implementierung

Nach der Vorstellung der Theorie in den Kapiteln 1 bis 3 wird in diesem Kapitel die praktische Umsetzung dieser Vorüberlegungen vorgenommen. Dabei sei zunächst das Maximum-Linkage-Verfahren und seine Erweiterung betrachtet und dann die einzelnen Umsetzungen der neuronalen Netze wie Winner-takes-all und selbstorganisierende Karten.
Die theoretischen Ansätze sind in ein rechnergestütztes Programm implementiert. Das in den Programmiersprachen JAVA und C++ geschriebene Programm läuft auf allen gängigen Betriebssystemen (Windows / Linux / Unix). Das Programm ist so konzipiert, daß es bequem als Hintergrundanwendung benutzt werden kann und dabei trotzdem eine kurze Laufzeit für eine Clusterung benötigt. Die Laufzeit für die Berechnung einer Clusterung ist dabei von der Größe des Hauptspeichers und der Prozessorleistung abhängig. Eine Größe von 128 MB für den Arbeitsspeicher und 400 MHz für die Prozessorleistung wird als untere Grenze vorausgesetzt. Je größer der Speicher ist, desto schneller erhält man das Ergebnis.
In der aktuellen Version sind die Eingangsdaten ausschließlich Bilddateien, die in den Formaten PNG/BMP/JPG (Holtorf, 1996) vorliegen können. Eine Ausweitung auf einen allgemeinen matrixorientierten Datensatz ist möglich.

4.1 Maximum Linkage

Für die programmiertechnische Umsetzung müssen die folgenden Überlegungen und Probleme bedacht und gelöst werden:
Für die korrekte Auswahl der Zentroidwerte muß die Verteilung der Daten im Ursprungsbild beachtet werden. Wie bei der Vorstellung des Algorithmus in Abschnitt 1.2.3 bereits erwähnt, ist dieses ein essentieller Bestandteil des Algorithmus. Desweiteren muß die Distanzberechnung in einer laufzeitgünstigen Variante implementiert werden.

4.1.1 Einbeziehung der Pixelverteilung im Ursprungsbild

Damit die Zentroide das eingelesene Bild hinreichend gut beschreiben, bedarf es der Berechnung der „besten“ Zentroidwerte. Der Maximum-Linkage-Ansatz kann dieses mit einer gewissen Modifikation gewährleisten. Dazu muß die Häufigkeitsverteilung der verschiedenen Pixel beachtet werden. Wird diese Häufigkeitsverteilung nicht beachtet, können zuviele Zentroide aus dem Randbereich der verschiedenen

Pixelfarbwerte berücksichtigt werden, während dichtbesetzte Gebiete unterrepräsentiert bleiben. Das bedeutet, daß bei der Klassifikation der Farbwerte zu den Zentroiden die Farbwerte den „falschen“ Clustern zugeordnet werden.
Die Einbeziehung der Verteilung spiegelt sich an den Abstände δ_{ij} aus Gleichung (1.6) wieder. Diese Abstände werden mit den entsprechenden Häufigkeiten der Pixel gewichtet. Weist beispielsweise ein Punkt A die Häufigkeit 7 auf und ein Punkt B die Häufigkeit 23, wird der Abstand mit der Summe der Häufigkeiten ($7+23=30$) gewichtet. Durch diese Modifikation wird die Pixelverteilung mit einbezogen. Gleichzeitig erlangt man damit eine Überrepräsentation der dichtbesetzten Gebiete, da diese höhere Häufigkeiten aufweisen. Die Konsequenz daraus ist, daß seltene, aber wichtige Elemente in unserem Ursprungsbild nicht gefunden werden. Es empfiehlt sich daher, zwei Mengen von Zentroiden zu berechnen: Eine gewichtete und eine ungewichtete Menge. Die Vereinigung der beiden Mengen ergibt die gewünschte Zentroidmenge. Durch diese Auswahl der zwei Mengen wird sichergestellt, daß sowohl Zentroide am Rand als auch im Zentrum gefunden werden. Im Sinne des SSW-Clusterungskriteriums von Bock (Abschnitt 4.3.1) entsteht damit eine gute Clusterung. Es stellt sich nun die Frage, wie groß die einzelnen Mengen zu wählen sind, genauer: wieviele Elemente die Mengen jeweils enthalten?
Zunächst ist klar, daß die Gesamtanzahl der Zentroide vom Anwender zuvor anzugeben ist. Dann allerdings ist nichts weiter über die Kardinalität der Mengen bekannt. Zur besseren Verständlichkeit, sei folgendes Beispiel betrachtet:

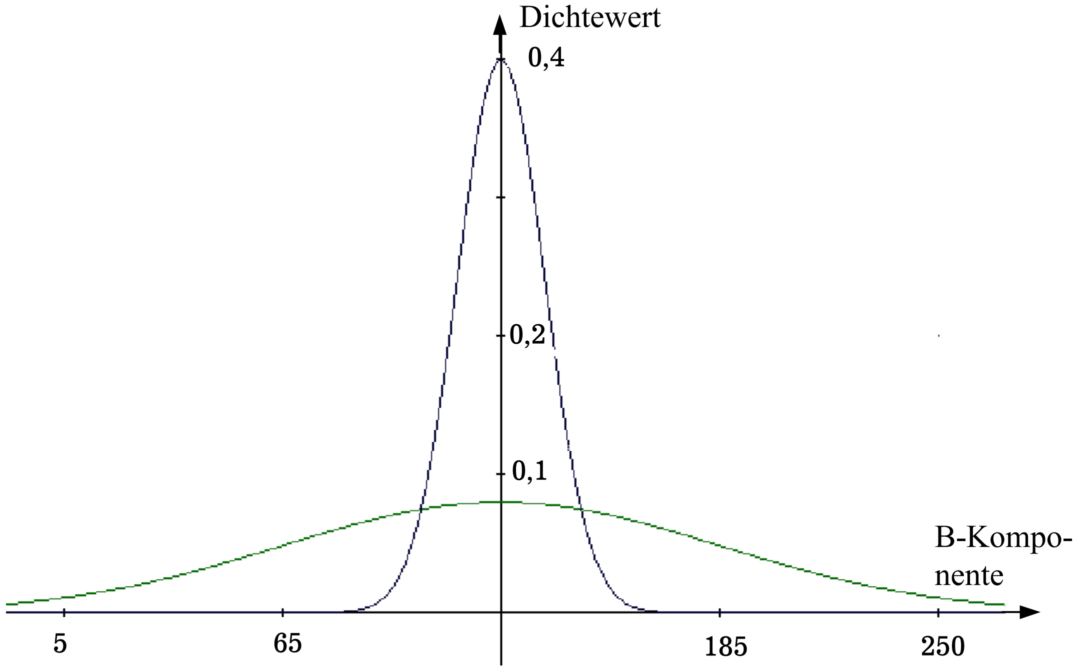


Abb. 17: Skizzierte Randdichte der RGB-Farbwerte zweier Bilder. Betrachtung der B-Komponente.

Unterstellen wir eine Farbverteilung gemäß der steileren Kurve aus Abb. 17. In diesem Fall werden mehr ungewichtete Zentoide gewählt, da es nur wenige Werte

mit großer Häufigkeit gibt. Aus diesem Grund werden nur wenige Zentroide zur Präsentation dieses Bereichs benötigt.
Bei einer Farbhäufigkeitsverteilung gemäß der flachen Kurve aus Abb. 17 werden viele gewichtete Zentroide benötigt. Zunächst werden mehr Zentroide dort gefunden, wo viele Werte liegen, also dort, wo die Kurve höher ist. Das Verhältnis der gewichteten und ungewichteten Zentroide entspricht dabei dem Verhältnis der Kurvenenden (flach) zum mittleren Kurvensegment (hoch). Aufgrund der gleichmäßigen Verteilung der Farbwerte wird die Einbeziehung der Häufigkeiten die Zentroidauswahl verbessern. Verschiebt sich das Verhältnis von flachen und höhrem Anteil stark, können zwei Probleme auftreten. Erstens werden zuviele Werte im „hohen" Bereich gefunden, die zudem räumlich dicht beieinander liegen. Dadurch geht die Trennschärfe zwischen Clustern verloren. Zweitens werden zuwenig Zentroide in den Bereichen mit niederer Kurve gefunden. Wodurch diese Bereiche unterrepräsentiert sind, weil die Abstände zwischen den Werten so groß sind, daß sehr viele unterschiedliche Werte in einem Cluster zusammengefaßt werden. Es kann dadurch vorkommen, daß seltene Elemente, die aufgrund ihres großen Abstandes zu andern Werten in einen eigenen Cluster einzuteilen sind, anderen weit entfernt liegenden Clustern zugeordnet werden.

€

Die im Beispiel angegebenen Überlegungen lassen den Schluß zu, daß eine varianzbasierte Größe zur Bestimmung der Anzahl der gewichteten bzw. ungewichteten Zentroide geeignet erscheint. Da die Komponenten der Farbvektoren (RGB) aufgrund der Definition des RGB-Modells unabhängig sind, fallen eventuelle Kovarianzüberlegungen zur Bestimmung der Anzahl der gewichteten bzw. ungewichteten Zentroide nicht ins Gewicht. Die Kovarianz ist nahe bei Null (theoretisch ist sie bei Null), so daß es ausreicht, die komponentenweise Varianz zu betrachten. Ferner sind die R, G und B-Werte aufgrund des beschränkten Merkmalraums von der gleichen Größenordnung, so daß eine Standardisierung nicht notwendig ist. Mit diesen Annahmen und der Annahme, daß die Varianz ein ausreichendes Maß für die Definition ist, sei sie definiert durch

$$\kappa = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{3} \left(P_{ik} - \overline{P}_k \right)^2 \text{, mit } \overline{P}_k = \frac{1}{n} \sum_{j=1}^{n} P_{kj} \, . \tag{4.1}$$

In (4.1) ist die R-Komponente durch $\boldsymbol{P}_{i1}$ gegeben, die G-Komponente durch $\boldsymbol{P}_{i2}$ und die B-Komponente durch $\boldsymbol{P}_{i3}$.

Der Wertebereich von κ ist durch die Begrenzung des Merkmalraums beschränkt. Sind alle Farbwerte im Bild gleich Null, ergibt sich $\kappa = 0$. Sind hingegen nur die zwei extremen Farbwerte $(0,0,0)$ und $(255,255,255)$ vorhanden, ergibt sich für κ

$$\kappa = \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{3}\left(P_{i_k} - \left(\frac{255}{2}\right)\right)^2 = \frac{1}{2}\left(6\cdot\left(\frac{255}{2}\right)^2\right) = 48768{,}75$$

Also: $0 \leq \kappa \leq 48768{,}75$.

Für die Anzahl der gewichtet zu wählenden Zentriode sei folgendes betrachtet:
Die Anzahl ist abhängig von κ und der zuvor vom Anwender spezifierten Gesamtanzahl der Zentroide. Ferner sollte die Funktion die Eigenschaft besitzen, daß sie zu Beginn ihres Verlaufs unterhalb einer linearen Funktion und zum Schluß ihres Verlaufs oberhalb einer linearen Funktion liegt. Dieser Verlauf ist wichtig, damit bei kleiner Varianz (unterhalb einer linearen Verbindung) gewährleistet ist, daß ausschließlich ungewichtete Zentroide gewählt werden, deren Varianzen eine Mindestgröße nicht unterschreiten. Bei großer Varianz hingegen (Verlauf oberhalb einer linearen Verbindung) sollten viele gewichtete Zentroide gewählt werden. Eine Funktion, die diese Eigenschaft besitzt, definiert sich durch

$$f_W(\gamma, q) = \left\lfloor\left(\sin\left(\frac{\pi\kappa}{48768.75} - \frac{\pi}{2}\right) + 1\right)\frac{q}{2}\right\rfloor. \tag{4.2}$$

Durch die Vorgabe der Gesamtanzahl q der Zentroide ist der ungewichtet zu wählende Anteil durch Subtraktion geben.
Formal:

$$f_{UW}(\gamma, q) = q - f_W(\gamma, q). \tag{4.3}$$

4.1.2 Rechnerbedingte Variante der Distanzberechnung

Aufgrund der Konstruktion des Maximum-Linkage-Ansatzes ist zu Beginn des Algorithmus ein großer und zeitaufwendiger Rechenschritt erforderlich. In diesem werden zunächst alle Abstände zwischen den Farbwerten bestimmt und anschließend das Maximum dieser Abstände aufgesucht. Die berechneten Abstände werden in einer Distanzmatrix Δ zusammengefaßt und zwischengespeichert. Genau diese Eigenschaft des Zwischenspeicherns der gesamten Matrix erfordert einen hinrei-

chend großen Arbeitsspeicher. Da dieser nicht ausschließlich vom Analyseprogramm belegt werden kann, muß die Matrix vom Betriebssystem auf die Festplatte ausgelagert werden. Das schreiben der Information auf eine Festplatte ist um den Faktor 10^3 langsamer als ein Arbeitsspeicherzugriff. Da dabei für jede neu zu berechnende Distanz auf die Matrix zugegriffen werden muß, verlangsamt sich die Verarbeitung erheblich (mittlerer Lesezugriff einer Festplatte: 9 ms; mittlerer Arbeitsspeicherzugriff: 5 ns). Ferner bedarf es der sehr kompliziert zu programmierenden Datenstruktur der b-Bäume (Güting, 1992). Um die Verarbeitungsgeschwindigkeit hoch zu halten, wird die Struktur des Maximum-Linkage-Algorithmus ausgenutzt. Die Berechnung wird schrittweise dergestalt durchgeführt, daß immer nur das Maximum der Abstände sowie die aktuell berechnete Distanz im Arbeitsspeicher festgehalten wird. Darüberhinaus werden der Punkt oder die Punkte, die zum Maximum geführt haben, im Speicher vorgehalten. Ist die aktuell berechnete Distanz größer als das Maximum, wird die aktuelle Distanz zum Maximum.
Obgleich diese Methode relativ schnell ist, da nur wenige Informationen im Arbeitsspeicher vorgehalten werden müssen, birgt sie doch einen kleinen Nachteil in sich. Gibt es mehr als ein Maximum, weisen zwei Distanzen den maximalen Abstand auf, so wird zufällig einer dieser Werte gemäß einer diskreten Gleichverteilung ausgewählt. Aufgrund des oben beschriebenen Ablaufs kann es vorkommen, daß die berechnete Distanz und das Maximum identisch sind. Würde dieses mit nur zwei Punkten vorkommen, ist jeder Punkt mit Wahrscheinlichkeit ½ zu wählen. Es können aber im Laufe des Algorithmus mehrere Maxima hintereinander gefunden werden. Somit muß die Anzahl der gefundenen Maxima ebenfalls im Speicher vorgehalten werden. Sei diese Anzahl mit u_{max} bezeichnet, dann gilt, daß die Wahrscheinlichkeit, ein im Speicher befindlicher Wert als Maximum auszuwählen, gegeben ist durch $u_{\max}/(u_{\max}+1)$. Ein neuberechneter Wert wird mit der Wahrscheinlichkeit $1/(u_{\max}+1)$ gewählt. Daher wird jedes vorkommende Maximum mit gleicher Wahrscheinlichkeit gewählt.

4.1.3 Intra-Class-Varianz zur Bestimmung der Gesamtanzahl der Cluster

Bei einer Clusterung ist die Anzahl der zu bestimmenden Cluster ein essentieller Punkt. Wieviele Cluster müssen gebildet werden? Jeder Cluster, der nicht mehr berücksichtigt werden muß, erspart Rechenzeit. Im Regelfall bestimmt der Substanzwissenschaftler aufgrund seines Hintergrundwissens diese Anzahl. Wünschenswert ist aber eine programmbasierte Festsetzung der Anzahl aufgrund der Eingangsdaten. Wegen der Konstruktion des Maximum-Linkage-Ansatzes ist es möglich, eine gewisse „Automatik“ bei der Bestimmung der Clusteranzahl anzusetzen.

Die „Güte“ einer Clusterung ist u. a. von der Intra-Class-Varianz (SSW) abhängig (Särndal et al., 1992). Bei n Datenpunkten, die es zu clustern gilt und welche in q Cluster eingeteilt werden sollen, fällt die SSW monoton mit steigendem q. Analog wird die Gesamtvarianz bei nur einem Cluster durch die SSW beschrieben. Ist $q = n$ ist die SSW = 0. Das Ziel einer Clusterung ist es somit, die SSW möglichst zu minimieren. Gleichzeitig sollte die Clusteranzahl nicht „zu groß“ sein. Da beide Kriterien nicht gleichzeitig zu erfüllen sind, bedarf es eines Kompromisses. Genau hier setzt das Maximum-Linkage-Verfahren an. Bei einer Clusterung mit q zu bildenden Clustern entsprechen die ersten q - 1 Zentroide genau denjenigen Zentroiden einer Clusterung mit $q - 1$ zu bildenden Clustern. Somit ist es nach der Bestimmung eines neuen Zentroidwertes möglich, eine Klassifizierung der Werte durchzuführen und die SSW entsprechend zu berechnen. (Die ursprünglichen Zentroide werden beibehalten.) Dieses wird nun sukzessive durchgeführt, bis die vom Anwender spezifizierte Clusteranzahl erreicht ist. Mit Hilfe der SSW hat der Anwender eine notwendige Bedingung an der Hand, um zu entscheiden, welche Clusteranzahl geeignet ist. Das hinreichende Kriterium wird in Abschnitt 4.3.2 vorgestellt. Ebenfalls dort wird auch ein grafisches Verfahren vorgestellt, welches einen Anhaltspunkt für die Clusteranzahl liefert.

4.1.4 Erweitertes Maximum Linkage

Das theoretisch entwickelte erweiterte Maximum Linkage ist gemäß den theoretischen Vorgaben in eine rechnergestützte Umgebung zu implementieren. Aufgrund der Berücksichtigung der Häufigkeitsverteilung werden hierbei ausschließlich gewichtete Zentroide bestimmt. Ansonsten gelten die Überlegungen für die Implementierung des herkömmlichen Maximum-Linkage-Ansatzes aus Abschnitt 4.1. Sollten im Verlauf der Berechnungen, für einen neu zu bestimmenden Zentroid, mehrere Maxima auftreten, so wird ein Maximum mit der Wahrscheinlichkeit $1/(u_{max} + 1)$ ausgewählt (vgl. Abschnitt 4.1.2). Weitere Einschränkungen und Zugeständnisse der Theorie an die praktische Umsetzung sind bei diesem Ansatz nicht zu machen.

4.2 Neuronale Netze

Damit ein neuronales Netz mittels der Lerneingaben trainiert werden kann, muß zunächst die besondere Datensituation berücksichtigt werden. Es sollen als Lerneingabe u. a. diskrete Werte (die RGB-Werte) geschaltet werden, die zwischen 0 und 255 liegen. Während des Trainings können sehr kleine Adaptionsschritte durchgeführt werden. Diese Veränderungen sind in einem diskreten Raum nicht durchführbar. Daher sei angenommen, daß für die Lernphase stetige Werte zwischen 0 und 255 angenommen werden dürfen. Nachdem das Netz ausgelernt hat, werden die berechneten Werte durch Runden wieder in diskrete Werte zurückgeführt.

4.2.1 Winner-takes-all-Netz

Damit ein Winner-takes-all-Netz trainiert werden kann, müssen die Gewichte für den ersten Adaptionsschritt initialisiert werden. Empirische Untersuchungen sowie Kohonen (2001) zeigen, daß eine völlig zufällige Initialisierung der Gewichte nicht ratsam ist. Im ungünstigsten Fall werden Werte für die Gewichte vorbelegt, die in den Originaldaten (Bild) weit voneinander entfernt liegen. Als Konsequenz aus dieser Initialisierung, würden die Gewichte, aufgrund ihrer großen Distanz von der Lerneingabe, nicht mehr verändert. Die Gewichte verschieben sich folglich nicht mehr innerhalb des Parameterraums. Daher kann es vorkommen, daß zu einem Zentroid kein Wert des Originalbildes klassifiziert wird und der Cluster leer bleibt.
Um solche Fälle zu vermeiden, sollten die Gewichte schon gleich zu Beginn hinreichend gut initialisiert werden. Die Möglichkeit, die Gewichte stärker im Raum zu verschieben, bietet sich nicht an, da dadurch bereits gut verschobene Gewichte wieder so stark verändert werden können, daß verschiedene Gewichte den gleichen Wert annehmen. Das bedeutet, daß die Gewichte übereinander geschoben werden.
Für eine hinreichend gute Initialisierung werden die Gewichte aus einem ε-Schlauch um die real vorkommenden Daten ausgeschnitten. Der Anwender kann ε in Abhängigkeit der Datenlage spezifizieren. Dabei gilt: $\varepsilon \in [0\,,\infty)$. Wird $\varepsilon = 0$ gewählt, werden die Gewichte zufällig aus den Ausgangsdaten initialisiert. Bei $\varepsilon = \infty$ werden die Gewichte unabhängig von den Ausgangsdaten zufällig initialisiert. Als Initialisierungswerte kommen die Daten in Betracht, die der folgenden Bedingung genügen:

$$\text{Sei } \varepsilon \geq 0, \text{ fest: } \forall i \in \{1,\dots,q\}\ \exists j \in \{1,\dots,r\}: \ \left\| W_i(0) - P_j \right\| \leq \varepsilon \tag{4.4}$$

Dabei bezeichnet r die Anzahl unterschiedlicher Farbwerte und q die gesuchte Clusteranzahl. $W_i(0)$ repräsentiert das Gewicht des i-ten Neurons im Lernschritt 0. P_j ist der Farbwert des j-ten Pixels. Auch hier sei die euklidische Norm verwendet.
Die Breite des Schlauchs ist in der Programmumsetzuung Picana vom Anwender selbst zuwählen. In der Grundeinstellung ist er mit 20 vorbelegt, da sich dieser „Radius" in den empirischen Analysen als sehr geeignet herausgestellt hat.

4.2.2 Das Problem des Überlernens von neuronalen Netzen

Beim Training neuronaler Netze ist ein besonderes Augenmerk darauf zu richten, daß das Netz nicht „übertrainiert" wird. Übertrainiert bedeutet, daß zu viele Lerneingaben an das Netz geschaltet werden. Dadurch werden die Neuronen des Netzes überangepaßt. Die Folge ist: Keine hinreichend gute Repräsentation der Originaldaten des Merkmalsraums (des zu clusternden Bildes), sondern eine Darstellung der sich im Bild befindenden häufigsten Farbwerte. Damit das Überlernen verhindert wird, müssen die Lerneingaben an das Netz reduziert werden.
Um die Reduktion zu erreichen, wird zunächst eine Häufigkeitstabelle der Pixel des Originalbildes berechnet. Diese Häufigkeiten werden dergestalt transformiert, daß kleinere Häufigkeiten immer ausgewählt werden, während Farbwerte, die sehr oft im Bild vorkommen, zwar deutlich öfter als Lerneingabe geschaltet werden als seltene Farbwerte, jedoch weit weniger oft als ihre Häufigkeiten es suggerieren.
Bezeichne P_i den Pixel im Bild, mit $i=1,...,r$. Wobei r die Anzahl der unterschiedlichen Farbwerte des Originalbildes ist. Sei ferner h_i die Häufigkeit des Farbwertes P_i. Dann ist eine der obigen Beschreibung genügende Transformation gegeben durch

$$h_i^* = \left\lceil h_i^{\frac{\frac{\pi}{2} - \arctan(1/40(H(P_i)-80))}{4\pi}} + 0{,}75 \right\rceil, \quad i=1,...,r. \tag{4.5}$$

Die unterschiedlichen Werte werden gemäß der transformierten Häufigkeit aus (4.5) als Lerneingabe angelegt. Dieser Funktion liegt stets unterhalb der Identität. Ferner gilt:

$$\forall P_i, P_j : h_i \le h_j \Leftrightarrow h_i^* \le h_j^* \ . \tag{4.6}$$

Nachdem die Voraussetzungen und Vorüberlegungen des Lernens neuronaler Netze abgeschlossen sind, wird im folgenden auf die Spezifikation der Lernrate $\eta(s)$ und die Nachbarschaftsfunktion d_{ij*} eingegangen (vgl. Abschnitt 2.5).
An die Lernrate $\eta(s)$ wird die Bedingung gestellt, daß sie monoton fällt und aus dem Intervall $[0,1]$ stammt (Kohonen, 2001). Daher definieren wir:

$$\eta(s) = (0{,}005)^{1/s_{\max}} \ . \tag{4.7}$$

Dabei repräsentiert s_{max} die Gesamtanzahl an Lernschritten. Diese berechnet sich aus der Anzahl aller Pixel im Originalbild und der Gesamtanzahl an Lernzyklen zk, die durchgeführt werden sollen. Ein Lernzyklus zks entspricht hierbei allen potentiell zur Verfügung stehenden Lerneingaben. Mit anderen Worten, die Lernzyklenanzahl zk gibt an, wie häufig das Netz durch sämtliche zur Verfügung stehenden Lerneingaben nacheinander trainiert werden soll. s_{max} ist gegeben durch

$$s_{max} = \text{(Anzahl der Pixel) (Anzahl der Lernzyklen).}$$

Die Anzahl der Pixel berechnet sich aus der Pixelanzahl der Bildhöhe multipliziert mit der Anzahl der Pixel in der Bildbreite.
Die Zahl $0{,}005$ stellt sicher, daß im letzten Lernschritt noch ein „Lernfortschritt" von 5 Promill erreicht wird.
Zum Abschluß wird die verwendete Nachbarschaftsfunktion vorgestellt. Diese Funktion beruht bei Winner-takes-all-Netzen ausschließlich auf den Gewichten der Neuronen. Ferner ist diese Funktion vom Lernschritt s unabhängig, da dieser bereits in der Lernrate $\eta(s)$ enthalten ist und diese implizit auf die Nachbarschaftsfuktion Einfluß nimmt.
Um eine geeignete Nachbarschaftsfunktion zu bestimmen, bedarf es folgender Überlegung: Was muß diese Funktion leisten?
Wenn das Netz einen Lernschritt durchführt, ist sicherzustellen, daß die seltenen Elemente, die in unserem Zusammenhang sehr wichtig sind, im geclusterten Bild nicht verschwinden. Ferner sollte zwischen den Clustern eine möglichst große Trennung vorherrschen. Aufgrund der ersten Tatsache erscheint eine Standardwahl der Nachbarschaftsfunktion, wie sie etwa Kohonen (2001) oder Zell (2000) anregen, beispielsweise eine Gauss-Funktion (vgl. Abschnitt 2.5.1), nicht geeignet. Auch bei einer kleinen Varianz gehen noch zu viele seltene Elemente verloren. Vielmehr ist es sinnvoll in einer kleinen Umgebung um den Gewinner, die anderen Neuronen von diesem Gewinner geringfügig abzustoßen. Ein weiterer Effekt dieses Wegschiebens besteht darin, daß Neuronen nicht übereinander geschoben werden. Zudem ist davon auszugehen, daß dieser Bereich des Merkmalsraums hinreichend gut durch den Gewinner beschrieben ist. Daher erscheint dieses Fortbewegen angebracht. Darüber hinaus sollte mit zunehmendem Abstand zwischen dem Gewinner und einem anderen Neuron die Nachbarschaftsfunktion kleiner werden.
Die Nachbarschaftsfunktion, die diese Vorüberlegung einbezieht, ist wie folgt definiert:

$$d_{ij^*}(s) = \begin{cases} 0{,}5 & , \quad \left\| W_i(s) - W_{j^*}(s) \right\|_2 = 0 \\ -0{,}1 & , \quad \left\| W_i(s) - W_{j^*}(s) \right\|_2 < 15 \\ \dfrac{1}{\left(\left\| W_i(s) - W_{j^*}(s) \right\|_2 \right)^{3/2} + 1} & , \quad \text{sonst} \end{cases} . \tag{4.8}$$

Die Grenze von 15 für den „Abstoßungsbereich" ist durch Simulation und empirische Auswertung gewonnen worden. In der Praxis hat dieser Wert sich bewährt. Die Funktion in (4.8) hat die folgende Gestalt:

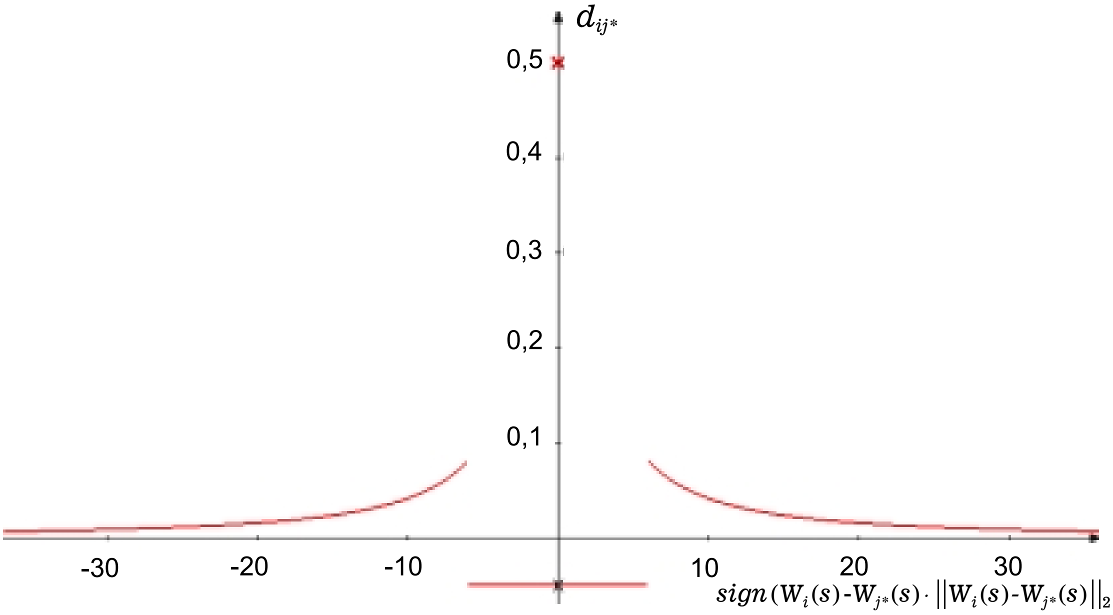


Abb. 18: Darstellung der Funktion aus (4.8).
Gezeigt ist der Gewinner sowie der Bereich, indem die Neuronen vom Gewinner abgestoßen werden.

4.2.3 Selbstorganisierende Karten

Für die programmtechnische Umsetzung der selbstorganisierenden Karten ist im Gegensatz zu den Winner-takes-all-Netzen eine andere Nachbarschaftsfunktion d_{ij^*} notwendig. Bei SOM's ist neben dem Abstand zwischen den Neuronengewichten auch der Abstand der Neuronen zueinander entscheidend. Häufig verwendete Beziehungsstrukturen zwischen den Neuronen sind gitterähnlich oder weisen eine hexagonale Struktur auf (Kohonen, 2001).

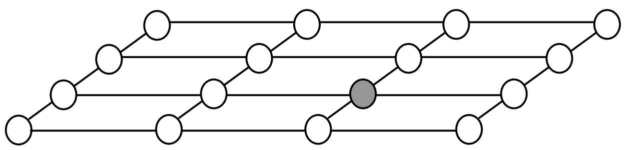


Abb. 19: Schematische Darstellung einer Gitterstruktur bei einer SOM. Das Gewinnerneuron ist grau markiert.

Der Abstand zwischen den Neuronen ist mittels der Wegverbindung zu beschreiben. Algorithmisch bedeutet diese Verbindung die Implementierung einer Kantenverfolgung, wie z. B. Tiefen- oder Breitendurchlauf (vgl. Güting, 1992). Bei einem Kantenverfolgungsalgorithmus werden die „besuchten" Kanten zwischen zwei Punkten (hier Neuronen) gezählt. Die kürzeste Verbindung weist entsprechend die wenigsten Kanten auf. Diese kürzeste Verbindung ist nicht eindeutig in dem Sinne, daß mehrere Wege die gleiche Kantenanzahl aufweisen können. Um diese Nichteindeutigkeit zu umgehen, werden die Neuronen, beginnend in der oberen linken (nord-westlichsten) Ecke mit den Koordinaten $(1{,}1)$ bezeichnet und alle weiteren Neuronen analog einer Matrix beschriftet. Ein Weg soll daher so definiert sein, daß ein Verbindungsschritt entlang der Kante nach rechts re_{som} (Osten) oder nach unten un_{som} (Süden) zulässig ist. Die Reihenfolge der Schritte, ob zuerst nach rechts oder nach unten, sei dabei irrelevant.
Die $(x{,}y)$-Koordinaten eines Neurons bzgl. seiner besuchten Kanten ist damit gegeben durch:

$$(x{,}y) = (\#un_{som} + 1,\ \#re_{som} + 1).$$

Unter Berücksichtigung dieser Vorüberlegung sowie der Einbeziehung des Abstandes zwischen den Gewichten und den Neuronen selbst sei die Nachbarschaftsfunktion der selbstorganisierenden Karten definiert als

$$d_{ij^*}^{\text{Standard-SOM}} = \exp\left[-\frac{\left[\left\|\begin{pmatrix} x(N_i) \\ y(N_i) \end{pmatrix} - \begin{pmatrix} x(N_{j^*}) \\ y(N_{j^*}) \end{pmatrix}\right\|_2\right]^2}{\left(\frac{ar(zk - zks)}{zk}\right)^2}\right], \tag{4.9}$$

mit ar als Adaptionsradius. Dieser gibt an, wie stark die Veränderung an den von Gewinner weiter entfernt liegenden Gewichten vorgenommen wird. N_i bezeichne das i-te Neuron, zks den aktuellen Lernzyklus und zk die Gesamtanzahl der Lernzyklen.
Zur Verdeutlichung für verschiedene Lernzyklen zks sei (4.9) in Abb. 20 dargestellt.

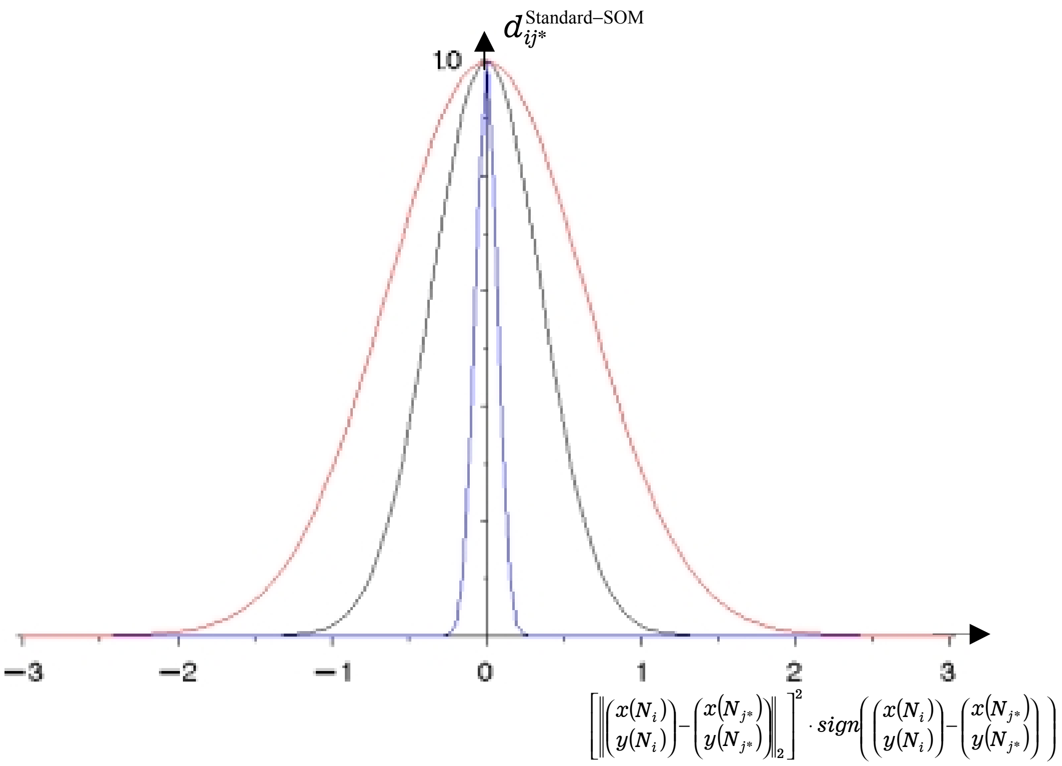


Abb. 20: Die Nachbarschaftfunktion aus (4.9) mit den Parametern Lernzyklenanzahl $zk = 1000$, Adaptionshöhe $ah = 0{,}999$ und den Werten für den aktuellen Lernzyklus $zks = 300$ (rot), $zks = 500$ (schwarz), $zks = 700$ (blau).

Für die weitere implementierte Variante der SOM's, den so bezeichneten Schnellen-SOM's lautet die Nachbarschaftsfunktion:

$$d_{ij^*}^{\text{Schnelle-SOM}} = 0{,}3 \cdot \left[s^9 \cdot \left\| \begin{pmatrix} x(N_i) \\ y(N_i) \end{pmatrix} - \begin{pmatrix} x(N_{j^*}) \\ y(N_{j^*}) \end{pmatrix} \right\|_2 + 1 \right]^{-1} . \tag{4.10}$$

Die Nachbarschaftsfunktion aus (4.10) weist eine ähnliche Struktur wie (4.8) auf. Dort steht der Abstand zwischen den Gewinner und dem zu ändernden Neuron im Nenner. Bei den Schnellen-SOM's ist der entscheidende Abstand nicht dieser, sondern der Abstand auf der U-Matrix. Das ist der Grund dafür, daß die Funktion aus (4.10) auch keinen Abstoßungsbereich benötigt.
Die Darstellung von (4.10) ist Abb. 21 zu entnehmen.

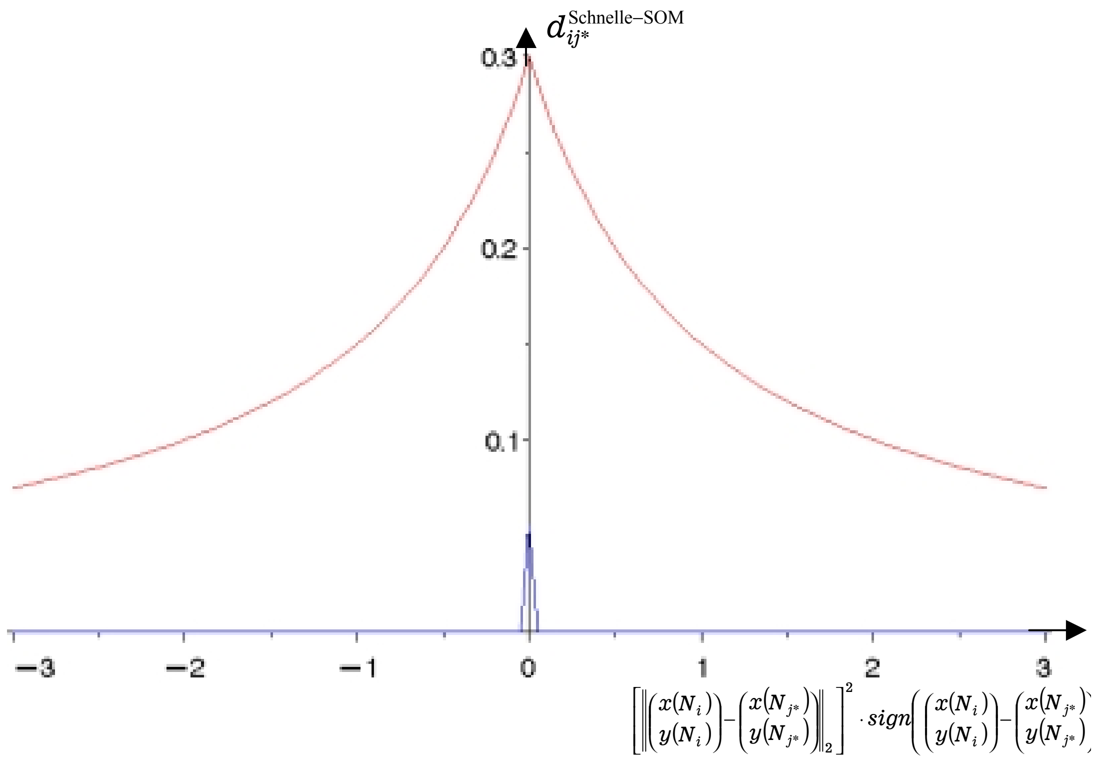


Abb. 21: Darstellung der Nachbarschaftsfunktion aus (4.10) mit den Parametern $s = 1$ bzw. $s = 20$

Anhand von Abb. 21 wird deutlich, wo die Nachbarschaftsfunktion aus (4.10) ihr Maximum hat. Dieses tritt auf, wenn der Abstand zwischen Neuron und Gewinner nahe bei Null liegt. Bei großen Abständen liefert Gleichung (4.10) kleine Werte zurück. Mit Zunahme der Lernschritte wird dieser Effekt verstärkt. Das bedeutet, daß im Laufe des Lernprozesses nur noch der Gewinner angelernt wird. Die übrigen Neuronen werden kaum noch adaptiert.
Die SOM's, so wie sie Kohonen (1982) vorgestellt, lernen langsam. Daher wird im Gegensatz zur allgemeinen Lernrate $\eta(s+1) = \alpha\,\eta(s)$, $\alpha \in (0,1)$, eine andere gewählt. Neben dem Iteratsionschritt s ist die Lernrate vom aktuellen Lernzyklus zks und der Gesamtanzahl der Lernzyklen zk ahängig. Damit wird erreicht, daß sich die Lernrate nicht mehr bei jeder neuen Lerneingabe verändert, sondern bei einem Wechsel des Lernzyklus. Neben diesen Größen wird die Adaptionshöhe ah mit aufgenommen. Dieser Parameter legt die Stärke der Anpassung für die Lerneingaben fest. Die Adpationshöhe wird vor Beginn der Lernphase festgelegt und kommt aus dem Intervall $(0,1]$. Die Lernrate für die Standard-SOM's lautet:

$$\eta(zk, zks, ah) = \frac{ah \cdot (zk - zks)}{zk}.$$

Für die Schnellen-SOM's wird die gleiche Lernrate verwendet wie für die WTANs.

4.3 Beurteilung von Clusterungen

Bei Clusterungen, wie sie die vorgestellten Verfahren Maximum Linkage sowie neuronale Netze berechnen, stellt sich die Frage der Clusterungsgüte. Unter welchen Bedingungen ist eine Clusterung als „gut“ einzustufen?

Ein Kriterium ist sicherlich die Zeit. Was nutzt dem Anwender ein Verfahren, welches bzgl. aller Vergleichskriterien hervorragende Clusterungen liefert, aber dafür mehrere Monate benötigt. Der Nutzen eines solchen Verfahrens ist beschränkt. An dieser Stelle soll nur kurz auf den Zeitgesichtspunkt eingegangen werden. Genauere Untersuchungen sind bei Zerbst (2001) nachzulesen.

Wegen der einfachen Struktur sind der erweiterte Maximum-Linkage-Algorithmus und der Maximum-Linkage-Algorithmus die schnellsten hier vorgestellten Verfahren. Das Winner-takes-all-Netz bildet den dritten Platz, gefolgt von den selbstorganisierenden Karten. Die Rechenzeitverlängerung bei den neuronalen Netzen ist durch das mehrfache Durchlaufen der Lernzyklen bestimmt. Die selbstorganisierenden Karten bilden aufgrund des Herausarbeitens der Nachbarschaftsstruktur das Schlußlicht. Die Gründe für die variierende Rechenzeit sind durch die unterschiedlichen Methoden zur Zentroidsbestimmung bedingt. Allen Verfahren gemeinsam ist das Dateneinlesen und die Erzeugung der Häufigkeitstabelle sowie die Klassifikation der Werte nach der Zentroidbestimmung. Der Hauptunterschied basiert auf der Distanzberechnung der Zentroide. Die Distanzberechnung fließt beim Maximum-Linkage-Verfahren zur Bestimmung der Maximin-Bedingung ein, während sie bei den neuronalen Netzen zur Identifikation des Gewinners dient.

Für ein Luftbild mit n Bildpunkten und r unterschiedlichen Bildpunkten, das in q Cluster aufgeteilt werden soll, ergeben sich im Falle von Maximum Linkage insgesamt

$$\frac{(r-1)r}{2}+2+\sum_{i=2}^{q-1}\left(i(r-i)+1\right) \tag{4.11}$$

Rechenoperationen. Als Rechenoperation zählen die Abstandsbestimmung zwischen zwei Punkten und das Aufnehmen eines Wertes zur Zentroidmenge.

In (4.11) stammen die ersten beiden Summanden aus der Bestimmung der ersten beiden Zentroide sowie deren Aufnahme in die Zentroidmenge (+ 2). Die folgende Summe repräsentiert die Operationsanzahl für die übrigen Zentroide.

Bei den neuronalen Netzen kommt neben den erwähnten Operationen noch die Gewichtsanpassung in der Lernphase hinzu. Für ein WTAN werden

$$6nq \tag{4.12}$$

Rechenoperationen benötigt.
In (4.12) werden q Operationen für die Distanzberechnung zwischen den Neuronengewichten benötigt. Für jeden der q Gewichtsadaptionsschritte werden 2 Operationen für die Bestimmung von d_{ij} und je eine Operation zur η Berechnung, zur Distanzbestimmung zwischen den Gewichten eines Neurons und dem Gewicht der Lerneingabe sowie zur Modifikation der Gewichte benötigt.
Bei den SOM's wird eine Operation zusätzlich benötigt, um d_{ij} zu bestimmen. Neben der Berechnung des Abstands zwischen den Gewichten der Neuronen wird auch der Abstand zwischen den Neuronen auf der „Kohonenkarte" (U-Matrix Darstellung) bestimmt. Insgesamt ergeben sich daher

$$7nq \tag{4.13}$$

Rechenoperationen.
Mit (4.11) – (4.13) und der Berücksichtigung der unterschiedlichen Lernzyklenanzahl bei den neuronalen Netzen ergibt sich für den Maximum- Linkage-Algorithmus eine deutlich geringere Laufzeit.
Bei einem Bild mit $n = 393216$ Bildpunkten, $r = 8730$ unterschiedlichen Bildpunkten und $q = 25$ gesuchten Cluster ist die Anzahl der Rechenoperationen der Tab. 12 zu entnehmen.

Methode	**Rechenoperationen**	**# Lernzyklen**
Maximum Linkage	$4{,}09\ 10^7$	/
WTAN	$5{,}90\ 10^9$	100
SOM	$1{,}03\ 10^{14}$	100

Tab. 12: Notwendige Rechenoperationen der Clusterungsmethoden für ein Bild mit $n = 393216$ Bildpunkten, $r = 8730$ unterschiedlichen Bildpunkten und $q = 25$ Cluster.
Die Lernzyklenanzahl ist auf 100 festgesetzt.

Der Zeitgesichtspunkt ist für den Anwender zwar wichtig, das entscheidende Kriterium ist aber die Qualität einer Clusterung. Die Beurteilung der Clusterungsqualität betrachten wir in den folgenden Abschnitten.

4.3.1 Das notwendige Kriterium: Intra-Class-Varianz

In Analogie zu (4.1) werden Kenngrößen eingeführt, um eine Clusterung zu beurteilen. Die Kenngrößen werden als Inter-Class-Varianz (κ_{SSB}) bzw. Intra-Class-Varianz (κ_{SSW}) bezeichnet. Mit diesen Kenngrößen kann die Qualität verschiedener Clusterungen verglichen werden. Beide Größen sind abhängig von der Gesamtanzahl n der zu verarbeitenden Bildpunkte sowie von der gewünschten Clusteranzahl q. Zum geeigneten Vergleich zweier Clusterungen müssen die Werte von n und q übereinstimmen, unabhängig von der Clusterungsmethode. Soll eine Clusterung mit Maximum Linkage mit einer Clusterung mit SOM's verglichen werden, so muß das gleiche Eingangsbild vorliegen.

Die Intra-Class-Varianz sei definiert gemäß

$$\kappa_{SSW} = \sum_{i=1}^{q} n_i \sum_{k=1}^{m} \left[\left(\frac{1}{n_i} \sum_{j=1}^{n_i} P_{ijk} \right) - \left(\frac{1}{n} \sum_{l=1}^{n} P_{lk} \right) \right]^2 , \tag{4.14}$$

wobei m der Dimension der zu verarbeitenden Daten P_j, $j = 1,\ldots,n$, entspricht. Im Falle von RGB-Daten, wie im vorliegenden Fall, ist $m = 3$. Die Anzahl der Werte P_j in dem i-ten Cluster ist mit n_i, $i = 1,\ldots,q$, bezeichnet.
Die Inter-Class-Varianz κ_{SSB} sei gegeben durch

$$\kappa_{SSB} = \frac{1}{n} \sum_{i=1}^{q} \sum_{j=1}^{n_i} \sum_{k=1}^{m} \left(P_{ijk} - \left(\frac{1}{n_i} \sum_{j=1}^{n_i} P_{ijk} \right) \right)^2 , \tag{4.15}$$

mit m, n_i und P_j wie in (4.14).
Die Gesamtvarianz κ_{SST} ergibt sich als Summe aus (4.14) und (4.15) gemäß

$$\kappa_{SST} = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{m} \left(P_{ik} - \frac{1}{n} \sum_{j=1}^{n} P_{jk} \right)^2 . \tag{4.16}$$

Die Gleichungen (4.14) – (4.16) weisen für die praktische, computergerechte Berechnung einen Nachteil auf. Da bei der Berechnung nicht alle n Datenwerte P_j, $j = 1,\ldots,n$, (hier: Bildpunkte) bekannt sind, sondern nur die r unterschiedlichen Datenwerte P_j, $j = 1,\ldots,r$, mit den Häufigkeiten h_i, bedarf es einer anderen Form dieser Gleichungen. Sei dazu r_i die Anzahl unterschiedlicher Datenwerte im Cluster

i und h_{ij} die Häufigkeit des j-ten Datenwertes im Cluster i, dann ist κ_{SSW} aus (4.14) gegeben durch

$$\kappa_{SSW} = \sum_{i=1}^{q} r_i \sum_{k=1}^{m} \left(\frac{\sum_{j=1}^{r_i} h_{ij} P_{ijk}}{\sum_{j=1}^{r_i} h_{ij}} - \frac{\sum_{i=1}^{q} \sum_{j=1}^{r_i} h_{ij} P_{ik}}{\sum_{i=1}^{q} \sum_{j=1}^{r_i} h_{ij}} \right)^2 . \tag{4.17}$$

Die Inter-Class-Varianz berechnet sich analog (vgl. (4.15)):

$$\kappa_{SSB} = \frac{\sum_{i=1}^{q} \sum_{j=1}^{r_i} \sum_{k=1}^{m} \left(P_{ijk} - \overline{P}_{i \bullet k} \right)^2}{\sum_{i=1}^{q} \sum_{j=1}^{r_i} h_{ij}} , \tag{4.18}$$

mit

$$\overline{P}_{i \bullet k} = \frac{1}{\sum_{j=1}^{n_i} h_{ij}} \sum_{j=1}^{r_i} h_{ij} P_{ijk} .$$

Zur Beurteilung von Clusterungen, die durch verschiedene Verfahren gewonnen wurden, dient κ_{SSW}. Die Clusterung, die die kleinste Intra-Class-Varianz aufweist, ist die Beste, da sie dem Clusterungskriterium nach Bock (1998) am nächsten kommt. Hier jedoch stellt diese Größe nur das notwendige Kriterium dar. Mittels κ_{SSW} wird sichergestellt, daß das gefundene Cluster homogen ist. Um ein ausreichendes Maß an Homogenität zu gewährleisten, sei folgende Bedingung eingeführt.

Sei $C = \{c_1,...,c_q\}$ eine Clusterung der Werte P_i, $i = 1,...,r$, mit $C_j = \left\{ P_{j1},...,P_{jr_i} \right\}$. Für die Clusterung soll gelten:

$$\kappa_{SSW}(C) \leq \frac{1}{3} \kappa_{SST}(C) . \tag{4.19}$$

Unter Zuhilfenahme von (4.19) werden Clusterungen gefunden, die ein Mindestmaß an Homogenität aufweisen. Clusterungen, die dieses Kriterium nicht erfüllen, werden von vornherein verworfen.

Zur schlußendlichen Beurteilung der Güte einer Clusterung ist dieses Kriterium zwar notwendig, jedoch nicht hinreichend, da es eine bedeutende Schwäche aufweist. Diese Schwäche ist darin begründet, daß Clusterungen, die gemäß dieses Kriteriums gebildet wurden, Cluster mit Werten geringer Häufigkeit nicht erkennen. Für die Werte mit großer Häufigkeit werden eigene Cluster gebildet. Werte mit geringer Häufigkeit werden in die nächstgelegenen Cluster der Werte mit großer Häufigkeit einsortiert. Dadurch wird die Intra-Class-Varianz jedoch nur im geringen Umfang vergrößert. Aufgrund der geringen Häufigkeit haben diese Werte kaum Einfluß, obgleich sie einen größeren Abstand vom Clustermittelwert aufweisen. Durch diese Aufteilung der Cluster gehen seltene Elemente verloren, die in ein bestehendes Cluster eingegliedert werden, obwohl sie eigene Cluster repräsentieren sollten.

Darüber hinaus besteht noch das Problem, daß Werte mit großer Häufigkeit, die nur einen geringen Abstand aufweisen, trotzdem eigene Cluster bilden. Wegen der inhaltlichen Interpretation im Rahmen der Erosionsproblematik gehören diese Werte in ein gemeinsames Cluster. So entsteht eine Clusterung mit nur geringer Trennschärfe zwischen den Clustern. Ein hinreichendes Kriterium in dem vorliegenden Fall ist nachstehend beschrieben.

4.3.2 Das hinreichende Kriterium: Mittlere Minimaldistanz

Mit dem Kriterium mittlere Minimaldistanz sollen zwei Eigenschaften für eine „gute" Clusterung herausgearbeitet werden. Zum einen sollen Werte mit geringer Häufigkeit nicht in zu große und inhaltlich unpassende Cluster eingebettet werden, zum anderen sollen die einzelnen Cluster klar voneinander getrennt sein. Diese Forderungen werden von dem nachfolgend beschriebenen Kriterium des mittleren minimalen Abstands zwischen den Clusterzentroiden erfüllt. In diesem Kriterium gehen die Überlegung zur Maximum-Linkage-Bedingung (vgl. Gleichung (1.5)) ein. Dabei wird das Kriterium in zwei Schritten bestimmt. Zunächst wird zu jedem Zentroid sein nächster Nachbar bestimmt und die Distanz dazwischen berechnet; danach wird über die Distanzen das arithmetische Mittel gebildet. Die mittlere Minimaldistanz (MMD) wird berechnet durch

$$MMD = \frac{1}{q}\sum_{j=1}^{q} \min_{i \in \{1,\ldots,q\} \setminus j} \left\| z_i - z_j \right\|_2 , \qquad (4.20)$$

mit q als Clusteranzahl.

Je größer der Wert von MMD, desto höher ist die Trennschärfe zwischen den Clustern. Ferner steigt die Chance seltene Elemente eindeutig zu identifizieren. Bei Zerbst (2001) wird dieses Kriterium als Average Minimal Distance (AvgMinDist) bezeichnet. Im Unterschied zur Maximum-Linkage-Bedingung, die die Maximie-

rung des minimalen Abstandes zwischen den Zentroiden berücksichtigt, wird durch MMD der tatsächliche mittlere minimale Abstand der Clusterung bestimmt.
Um die „beste" Clusterung zu bestimmen, wird aus allen Clusterungen, die (4.19) erfüllen, die jenige Clusterung ausgewählt, die den größten Wert nach (4.20) aufweist. Die ausgewählte Clusterung besitzt dann sowohl homogene Cluster als auch eine hohe Trennschärfe zwischen den Clustern.
Da die Clusterungen, die mit diesem Kriterium bestimmt werden, „gut" sind, stellt sich die Frage, ob eine Beurteilung der Clusterung allein mit MMD nicht ausreicht?

Betrachten wir dazu die folgenden Abb. 22 und Abb. 23:

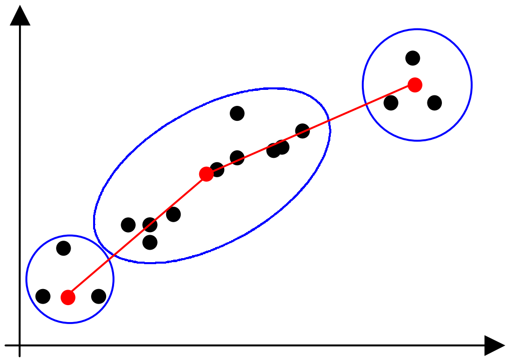


Abb. 22: Clusterung der schwarzen Punkte in drei Cluster. Die farbigen Punkte stellen die Zentroide dar. Die Länge der farbigen Strecken sind die Ausgangswerte der MMD. Diese Clusterung besitzt eine große MMD.

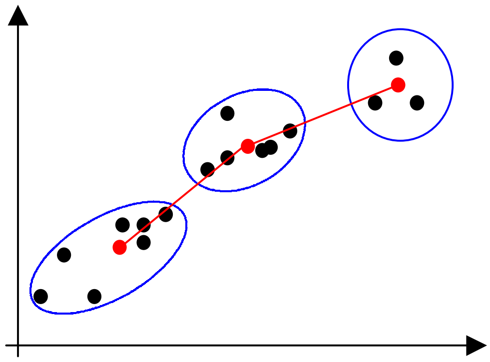


Abb. 23: Clusterung der schwarzen Punkte in drei Cluster. Die farbigen Punkte stellen die Zentroide dar. Die Länge der farbigen Strecken sind die Ausgangswerte der MMD. Diese Clusterung besitzt ein kleines κ_{SSW} .

Die mittlere Minimaldistanz der Clusterung aus Abb. 22 ist größer als die der Clusterung aus Abb. 23. Die Clusterung aus Abb. 23 ist jedoch vorzuziehen, da die Intra-Class-Varianz kleiner ist als bei der Clusterung aus Abb. 22. Das große Cluster aus Abb. 22 ist im Vergleich zu den anderen Clustern hetrogener, weil sich in diesem Cluster viele unterschiedliche Werte befinden. Eine inhaltliche Interpretation des Zentroids ist hierbei nicht möglich. Durch dieses Beispiel wird deutlich, daß eine kleine Intra-Class-Varianz Voraussetzung für eine „gute" Clusterung ist. Eine alleinige Beurteilung durch MMD ist nicht ausreichend.

4.3.3 Gewichtete mittlere Minimaldistanz

Die Entwicklung des Kriteriums der gewichteten mittleren Minimaldistanz (GMMD) fußt auf der Tatsache, daß das MMD-Kriterium einen Nachteil aufweist, da MMD auf dem arithmetischen Mittel basiert. Aus diesem Grund ist dieses Maß anfällig gegenüber Ausreißern. Daher führt eine unkritische Betrachtung dieser Kenngröße mitunter zur Mißinterpretation. Betrachten wir dazu zwei Beispiele.

Beispiel 1

Gegeben seien zwei Clusterungen A und B. Die geordneten Abstände $d_{(i)}$ zwischen den Zentroiden und ihren nächsten Nachbarn sind in Tab. 13 aufgeführt.

	$d_{(1)}$	$d_{(2)}$	$d_{(3)}$	$d_{(4)}$	$d_{(5)}$	$d_{(6)}$	$d_{(7)}$	$d_{(8)}$	$d_{(9)}$	$d_{(10)}$
Clusterung A	3,2	3,2	4,4	5,0	6,7	7,2	7,6	8,1	8,6	22,0
Clusterung B	4,7	4,7	5,1	7,0	7,0	8,0	8,0	9,0	9,5	9,5

Tab. 13: Distanzen für jeden Zentroid zu seinem nächsten Nachbarn für zwei Clusterungen A und B

Um einen besseren Eindruck über die Trennung zwischen den Clustern zu gewinnen, sei die nachstehende Abb. 24 betrachtet.

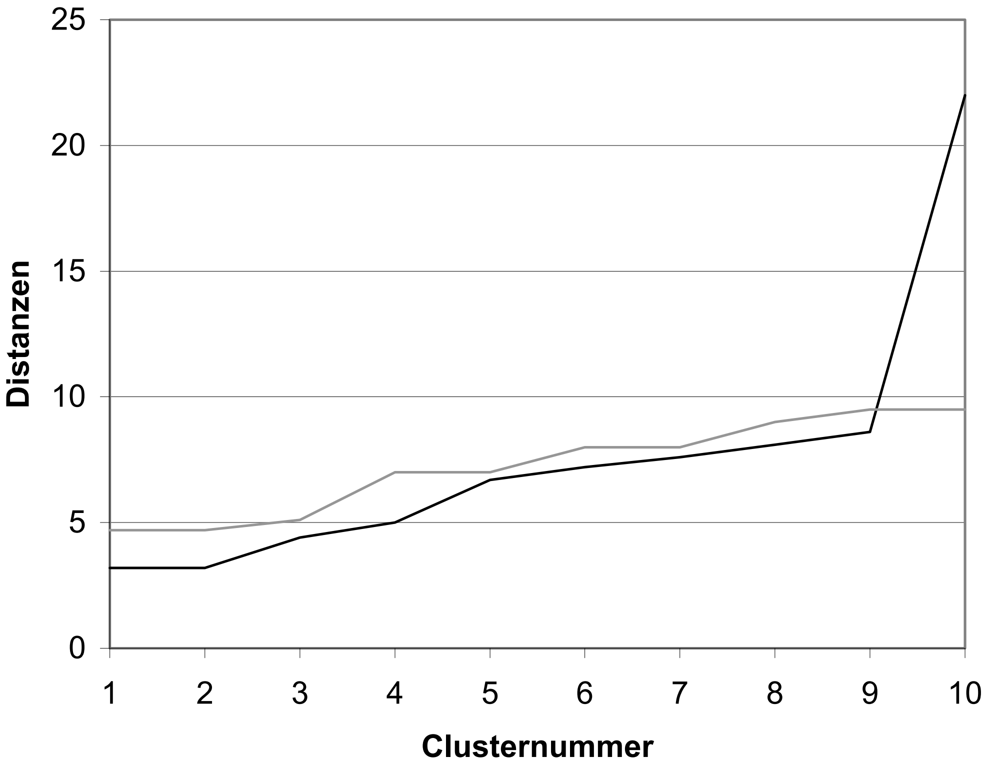


Abb. 24: Diagramm der minimalen Distanzen aus Tab. 13
Die schwarze Linie repräsentiert Clusterung A und die graue Linie Clusterung B.

Aus der Tab. 13 ist zu entnehmen, daß die Clusterung A aufgrund des größeren Werts von MMD (7,6) Clusterung B vorzuziehen ist ($MMD = 7{,}25$). Der größere Wert des MMD für Clusterung A basiert aber auf dem einen Ausreißer in der Distanz (22,0), der das arithmetische Mittel verzerrt. Aus der Abb. 24 ist zu sehen, daß die Entscheidung genau entgegengesetzt getroffen werden sollte, also ist Clusterung B vorzuziehen.

Die Entscheidung für eine Clusterung sollte nicht nur auf dem arithmetischen Mittel beruhen, sondern auch die Konzentration der Werte berücksichtigen. Betrachten wir dazu das folgende Beispiel.

Beispiel 2
Seien A und B zwei Clusterungen. Die geordneten Minimalabstände zwischen den Zentroiden und ihren nächsten Nachbarn sind in Tab. 14 aufgeführt.

	$d_{(1)}$	$d_{(2)}$	$d_{(3)}$	$d_{(4)}$	$d_{(5)}$	$d_{(6)}$	$d_{(7)}$	$d_{(8)}$	$d_{(9)}$	$d_{(10)}$
Clus. A	0,50	0,50	0,75	0,79	0,84	19,80	20,10	20,40	21,70	22,10
Clus. B	7,00	7,00	7,40	7,90	8,20	10,00	11,00	11,50	12,30	13,10

Tab. 14: Distanzen für jeden Zentroid zu seinem nächsten Nachbarn für zwei Clusterungen A und B

Die aufgeführten Distanzen können für jede Clusterung in zwei Gruppen eingeteilt werden. In Clusterung A sind die ersten fünf Werte sehr klein und die letzten fünf Werte deutlich größer. Clusterung B verhält sich analog, jedoch ist der Unterschied zwischen den Werten der einen Gruppe und den Werten der anderen Gruppe nicht so ausgeprägt.
Beurteilt man die Clusterung nach dem MMD-Kriterium, so fällt die Entscheidung auf Clusterung A ($MMD = 10{,}75$ gegenüber $MMD = 9{,}54$ bei Clusterung B). Das Kriterium kommt aufgrund der fünf dominierenden Werte der Gruppe mit den größeren Werten zu dieser „Entscheidung". Zur Verdeutlichung sind die Clusterungen in Abb. 25 dargestellt.

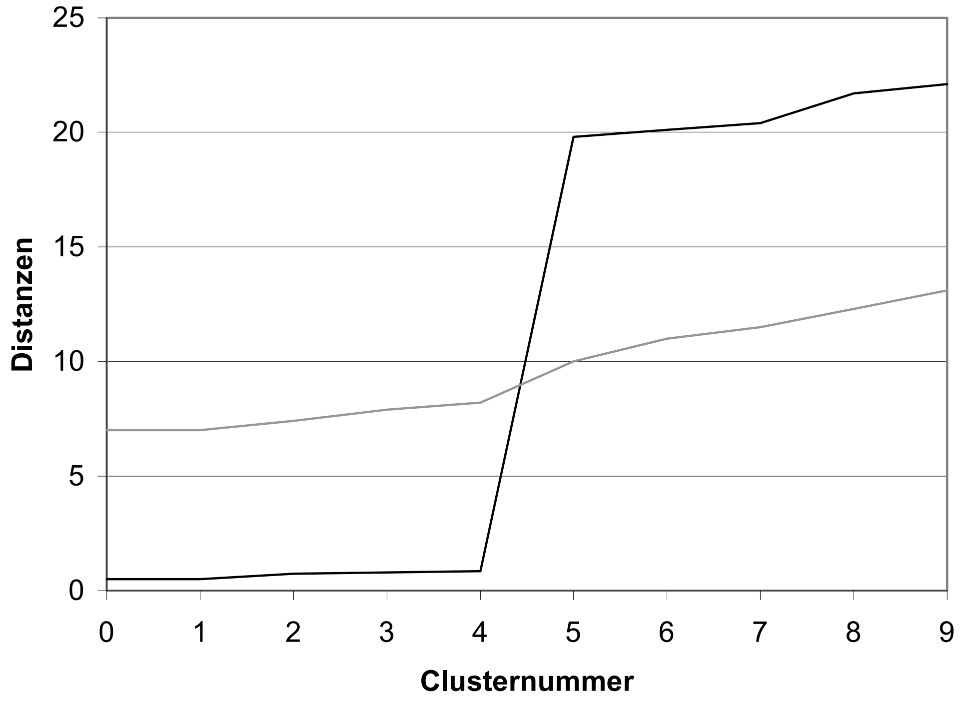


Abb. 25: Diagramm der minimalen Distanzen aus Tab. 14
Die schwarze Linie repräsentiert Clusterung A und die graue Linie Clusterung B.

Unter Einbeziehung der Überlegungen aus den obigen Beispielen soll das neue GMMD-Kriterium vorgestellt werden, das die Nachteile des MMD überwindet. Darüber hinaus findet es Clusterungen mit noch besserer Trennschärfe zwischen den Clustern.

Das Kriterium basiert auf zwei Ideen. Die Verwendung eines gewichteten, symmetrischen Mittels über die geordneten Distanzen d_i, $i=1,\ldots,q$, anstelle des gewöhnlichen arithmetischen Mittels. Hierbei sind die d_i gegeben durch

$$d_i = \min_{i \in \{1,\ldots,q\} \setminus j} \left\| z_i - z_j \right\|_2 , \tag{4.21}$$

wobei die z_i den Zentroiden entsprechen. Die Gewichtung berücksichtigt, daß die Werte, die am Rand der geordneten Distanzen liegen, weniger stark gewichtet werden als die Werte, die sich in der Mitte befinden. Gleichzeitig wird der Ausreißereffekt in jede Richtung abgeschwächt.

Die Gewichtung erfolgt gemäß

$$\omega(i) = \begin{cases} \dfrac{\dfrac{1-a}{\frac{q+1}{2}-1} \cdot i + a - \dfrac{1-a}{\frac{q+1}{2}-1}}{D_{Nenner}} & , \quad i \leq \dfrac{q}{2} \\[2ex] \dfrac{\dfrac{a-1}{q-\frac{q+1}{2}} \cdot i + 1 - \dfrac{q+1}{2} \dfrac{a-1}{q-\frac{q+1}{2}}}{D_{Nenner}} & , \quad i > \dfrac{q}{2} \end{cases} \tag{4.22}$$

mit

$$\begin{aligned} D_{Nenner} := & \sum_{j \leq q/2} \left(\frac{1-a}{\frac{q+1}{2}-1} \cdot j + a - \frac{1-a}{\frac{q+1}{2}-1} \right) \\ & + \sum_{j > q/2} \left(\frac{a-1}{q-\frac{q+1}{2}} \cdot j + 1 - \frac{q+1}{2} \frac{a-1}{q-\frac{q+1}{2}} \right). \end{aligned} \tag{4.23}$$

Der Parameter a gibt an, wie stark die jeweils äußeren Randpunkte berücksichtigt werden sollen. Es gilt: $a \in [0,1)$. Zur Veranschaulichung der Funktion aus (4.22) sei diese exemplarisch für $q = 10$ und $a = 0$ (schwarze Linie) bzw. $a = 0{,}5$ (graue Linie) dargestellt. Zur besseren Übersicht ist die an sich diskrete Funktion als durchgehender Graph gezeichnet.

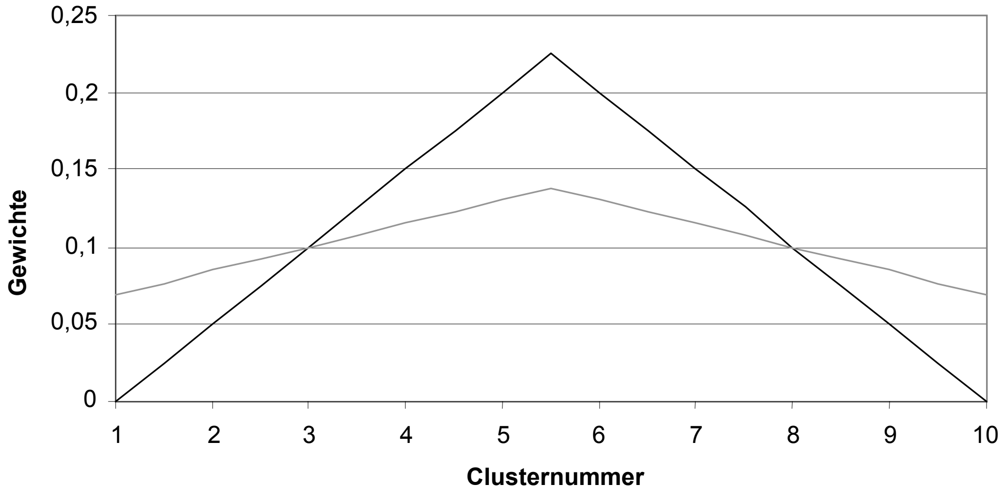


Abb. 26: Darstellung der Funktion aus (4.22) für $q = \mathbf{10}$ und $a = \mathbf{0}$ (schwarze Linie) und $a = \mathbf{0{,}5}$ (graue Linie)

Die gewichtete mittlere Minimaldistanz, basierend auf den Gewichten von (4.22), ist gegeben durch

$$GMMD_{d_i} = \sum_{i=1}^{q} \omega(i) \cdot d_{(i)} \,. \tag{4.24}$$

Je größer (4.24), desto besser die Clusterung. Des weiteren müssen wir nicht nur die wachsenden Distanzen berücksichtigen, sondern auch die Konzentration der Distanzen. Dieses führt zur Lorenzkurve, bzw. zum eng verwandten Gini-Koeffizienten. Ein kleiner Wert des Gini-Koeffizienten beschreibt eine geringe Konzentration. Für den hier betrachteten Kontext bedeutet das eine bessere Clusterung.

Um eine Clusterung zu beurteilen, sei das Verhältnis von $GMMD$ über die geordneten Distanzen zu dem Gini-Koeffizienten G betrachtet. Das GMMD-Kriterium ist daher gegeben durch

$$GMMDK_{d_i} = \frac{GMMD_{d_i}}{\left(G_{d_i}\right)^{1/2}} \,. \tag{4.25}$$

Wie bekannt, entspricht der Gini-Koeffizient dem doppelten der Konzentrationsfläche zwischen der Lorenzkurve und der Winkelhalbierenden. Im vorliegenden

Fall besitzt die Lorenzkurve $q+1$ Stützstellen. Um diese Stützstellen für die Abszisse (k_i) und die Ordinate (l_i) zu berechnen, betrachte

$$k_i = \frac{i}{q} \quad \text{und} \quad l_i = \frac{\sum_{j=1}^{i} d_{(j)}}{\sum_{j=1}^{q} d_j},$$

mit $i = 0,\dots,q$. Der Gini-Koeffizient ist gegeben durch

$$G = \sum_{i=1}^{q} (k_{i-1} + k_i) \frac{d_{(i)}}{\sum_{j=1}^{q} d_j} - 1.$$

Das $\boldsymbol{GMMDK}$ ist dann gegeben gemäß

$$GMMDK_{d_i} = \frac{\sum_{i=1}^{q} \omega(i) \cdot d_{(i)}}{\left(\sum_{i=1}^{q} \left(\frac{i-1}{q} + \frac{i}{q} \right) \frac{d_{(i)}}{\sum_{j=1}^{q} d_j} - 1 \right)^{1/2}}. \tag{4.26}$$

Im Gegensatz zum $\boldsymbol{MMD}$-Kriterium identifiziert $\boldsymbol{GMMDK}$ die „bessere“ Clusterung. Zum Vergleich seien die Beispiele 1 und 2 nochmals gegenübergestellt. Dabei wird zusätzlich auch der Wert von $\boldsymbol{GMMD}$ angegeben.

Beispiel	Clusterung	MMD	GMMD	Gini	GMMDK
1	A	7,60 *	6,51	0,154	16,60
	B	7,25	7,37 *	0,069 *	28,02 *
2	A	10,75 *	10,49 *	0,241	21,36
	B	9,54	9,33	0,065 *	36,66 *

Tab. 15: Parameter zur Beurteilung der Clusterungen aus Beispiel 1 und 2
Die grau hinterlegten Zeilen weisen auf die bessere Clusterung in Hinblick auf die optische Auswahl bezüglich der Diagramme der Minimaldistanzen. Die Werte, die durch * gekennzeichnet sind, markieren die bessere Clusterung entsprechend dem zugrundegelegten Kriterium.

In den grau hinterlegten Zeilen stehen die besten Clusterung des jeweiligen Beispiels. Die mit * gekennzeichneten Werte weisen auf die, bezüglich des verwendeten Kriteriums, beste Clusterung hin. Bei näherem Betrachten von Tab. 15 kommt man zu der Frage: Reicht es allein, den Gini-Koeffizienten zu betrachten, um die Clusterung zu beurteilen? Zur Beantwortung der Frage sei Beispiel 3 betrachtet.

Beispiel 3

Gegeben seien erneut zwei Clusterungen A und B. Die Minimaldistanzen zwischen den Zentroiden zu ihren nächsten Nachbarn seien gegeben gemäß Tab. 16.

	$d_{(1)}$	$d_{(2)}$	$d_{(3)}$	$d_{(4)}$	$d_{(5)}$	$d_{(6)}$	$d_{(7)}$	$d_{(8)}$	$d_{(9)}$	$d_{(10)}$
Clusterung A	5,8	5,8	5,9	6,1	7,2	7,6	8,1	8,4	8,8	9,2
Clusterung B	5,3	5,3	6,7	8,4	10,5	11,8	12,5	13,1	14,5	16,1

Tab. 16: Minimaldistanzen der Zentroide zu ihren nächsten Nachbarn

Clusterung A beginnt mit der größeren Minimaldistanz als Clusterung B. Würde man nur diese Tatsache betrachten, fiele die Entscheidung auf Clusterung A. Auch die nächsten Anhaltspunkte, wie das langsamere Anwachsen der Distanzen sowie die gleichmäßigere Konzentration der Werte, sprechen für Clusterung A. Der Gesamteindruck der Graphen läßt auf die Auswahl von Clusterung B schließen. Die nachfolgende Abb. 27 unterstreicht dieses.

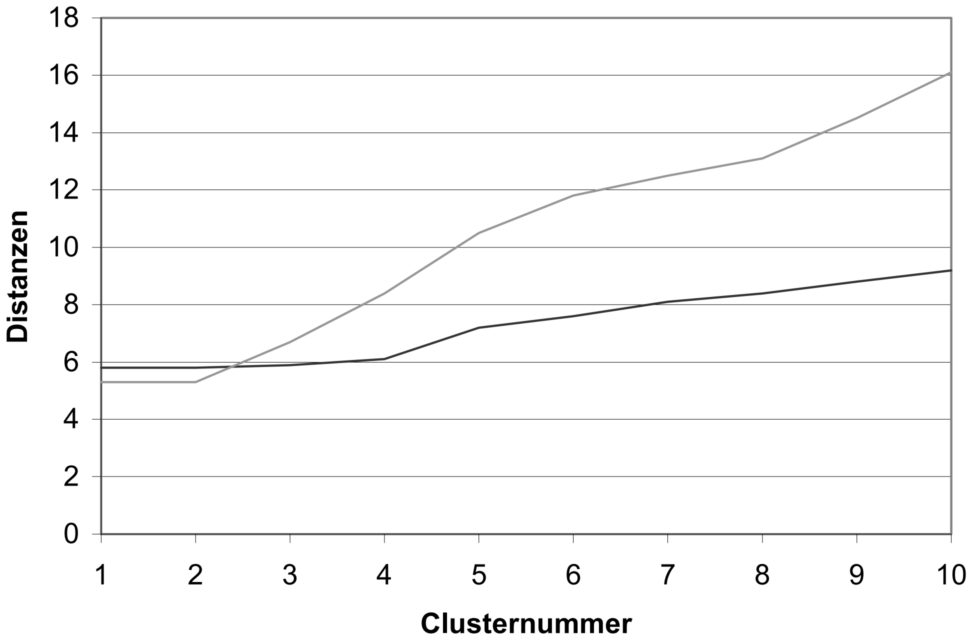


Abb. 27: Minimaldistanzen aus Tab. 16
Die schwarze Linie repräsentiert Clusterung A und die graue Linie Clusterung B.

Die Werte der Kriterien sind in Tab. 17 dargestellt:

Clusterung	MMD	GMMD	Gini	GMMDK
A	7,29	7,25	0,048 *	32,97
B	10,42 *	10,57 *	0,099	33,51 *

Tab. 17: Kriterien zur Auswahl der Clusterung

Mit Hilfe von Tab. 17 ist zu erkennen, daß der Gini-Koeffizient nicht stets die „beste" Clusterung identifiziert. Das $GMMDK$ hingegen findet die „beste" Clusterung.
Mit obigen Ausführungen ist leicht zu sehen, daß das $GMMDK$ dem MMD überlegen ist. Mit Hilfe des $GMMDK$ werden Clusterungen gefunden, die eine höhere Trennschärfe zwischen den Clustern aufweisen. Dabei ist es unerheblich, welche Clusterungsmethode benutzt wird.

Die grafischen Darstellungen der Minimaldistanzen in Bezug auf die Clusternummer ermöglichen es darüber hinaus die Anzahl der benötigten Cluster, bei einer Clusterung mit dem Maximum-Linkage-Algorithmus bzw. mit dem erweiterten Maximum-Linkage-Algorithmus, zu bestimmen.

Im Gegensatz zu den Abb. 24, Abb. 25 und Abb. 27 werden bei dieser Darstellung die Minimaldistanzen nicht aufsteigend, sondern absteigend sortiert. Eine typische Grafik hat folgende Gestalt:

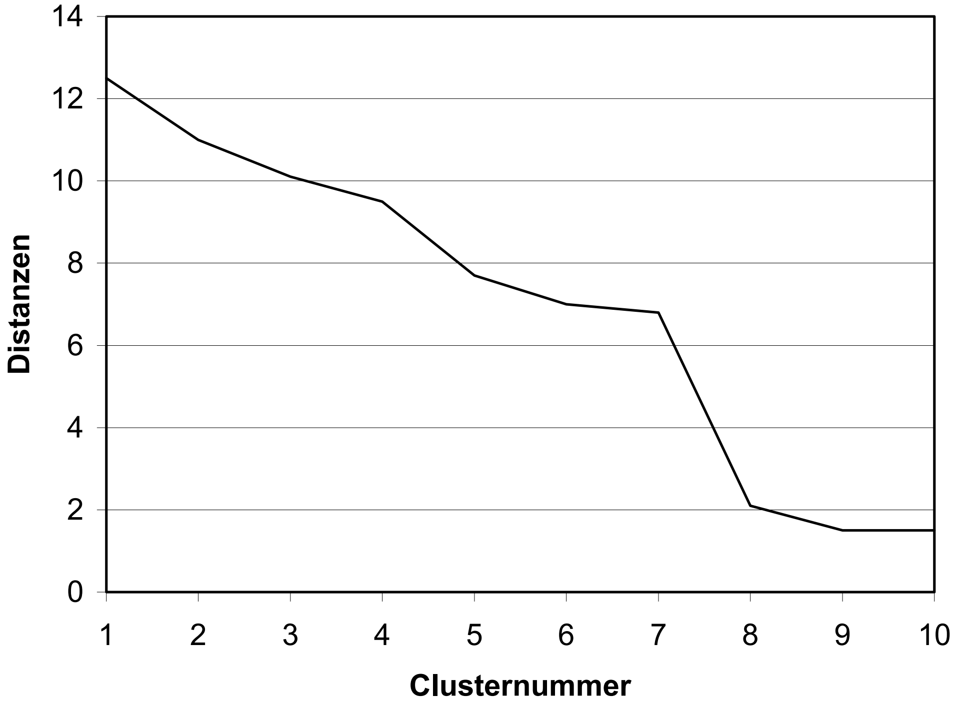


Abb. 28: Minimaldistanzen, absteigend sortiert

Verläuft die Kurve in Abb. 28 zu flach und zeigt sie einen extremen „Knick“ nach unten, wie hier zwischen dem siebten und achten Zentroid, so deutet es darauf hin, daß zu viele Zentroide bestimmt wurden. Einzig eine Clusterung mit weniger Clustern kann eine höhere Trennschärfe erreichen. Das ist darin begründet, daß die Trennschärfe maximal ist, sofern sie mit MLA / EMLA bestimmt wurde.
An dieser Stelle kann der Vorteil des MLA bzw. EMLA ausgenutzt werden, daß die Cluster sukzessive bestimmt werden. So stimmen bei einer Clusterung mit i zu bildenden Clustern die ersten i Zentroide mit den Zentroiden einer Clusterung mit $i + 1$ zu bildenden Clustern überein (vgl. Abschnitt 1.2.5). Das gewonnen $i + 1$. Cluster entspricht also einem zusätzlichen Cluster. Dieses wird bei der Bestimmung der Clusteranzahl berücksichtigt. Unterstellen wir eine Clusterung mit j Clustern. Ist die Trennschärfe der Clusterung mit j Clustern nicht hoch genug, beginnt folgender Ablauf:
Das letzte berechnete Cluster wird entfernt und im Diagramm werden die Minimaldistanzen gegen die Clusteranzahl aufgetragen (vgl. Abb. 28). Durch das Herausnehmen des letzten Clusters wird erreicht, daß der Zentroid mit der geringsten Distanz gelöscht wird. MLA / EMLA wählt die Zentroide bekanntlich so, daß in jedem Schritt die maximale Minimaldistanz berücksichtigt wird. Dieser Wert ist der letzte im Diagramm. Wird ein neues Diagramm berechnet (jetzt mit einer

Zentroidanzahl, die um eins vermindert ist), verläuft der Graph oberhalb des ersten (vgl. Abb. 29). Dabei nimmt der Wert des jetzt letzten Wertes (vormals der zweitletzte) zu. Dieser Sachverhalt ist in Abb. 29 dargestellt:

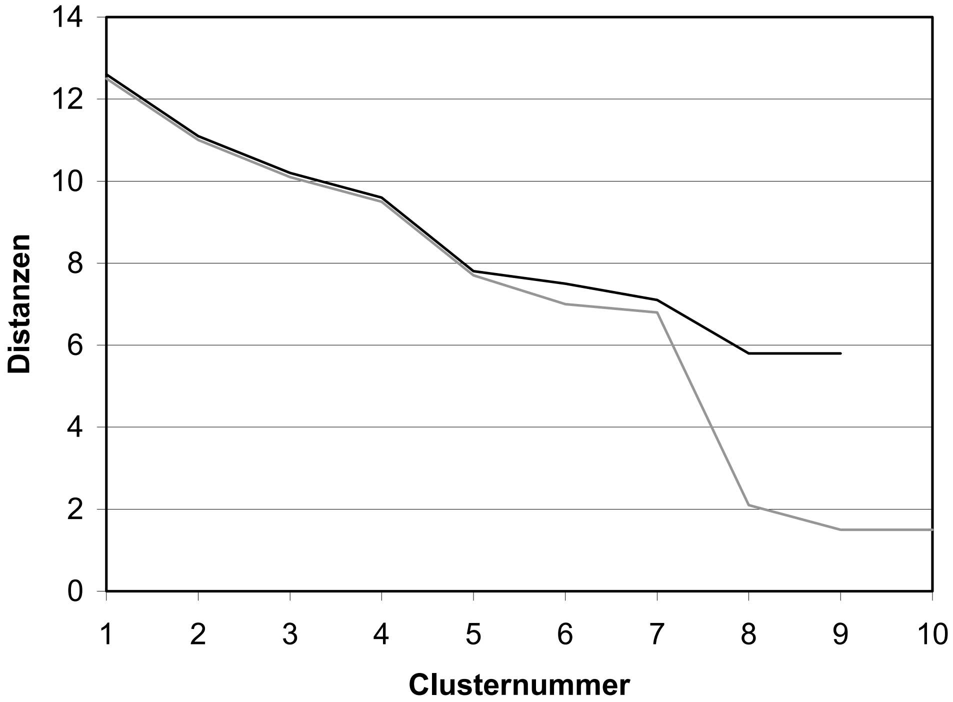


Abb. 29: Minimaldistanzen in absteigender Anordnung
Die graue Linie repräsentiert die Originalclusterung. Die schwarze Linie zeigt die Clusterung mit einem Zentroid weniger an.

Die Betrachtung der Minimaldistanzen ist sowohl auf den Zentroiden als auch auf den Klassenrepräsentanten möglich. Die Veränderung vom Zentroid zum Klassenrepräsentanten, die sich aufgrund der Klassifizierung der Werte zu den Zentroiden ergibt, ist im allgemeinen nicht sehr groß. Werden die Minimaldistanzen über die Zentroide gebildet, so ist beim Entfernen eines Zentroids keine zusätzliche Klassifizierung erforderlich. Der Rechenaufwand wird dadurch erheblich reduziert.
Das Auslassen des letzten Zentroids wird solange wiederholt, bis der kleinste Wert mit den Anforderungen des Anwenders übereinstimmt. Darüber hinaus muß die Veränderung der Intra-Class-Varianz betrachtet werden. Die Varianz sollte nicht „zu groß“ werden. Welche Werte zulässig sind, ist abhängig von der Anforderung des Anwenders und dem zugrundeliegenden Problem. Übersteigt die Intra-Class-Varianz einen vorgegeben Wert des Anwenders, so ist der zuletzt entfernte Zentroid wieder aufzunehmen. Das Wiederhineinnehmen eines Zentroids ist jedoch wesentlich aufwendiger als das Entfernen. Wird ein Zentroid wieder eingesetzt, müssen die Daten anschließend neu klassifiziert werden.

5 Anwendung der Methoden

Die in den vorangegangen Kapiteln vorgestellten Verfahren werden in diesem Kapitel anhand von Daten auf ihre Praxistauglichkeit getestet. Für die Anwendung der Clusterverfahren Maximum Linkage, erweitertes Maximum Linkage sowie die neuronalen Netze – Winner-takes-all und selbstorganisierende Karten – wird eine Luftaufnahme aus einem Untersuchungsgebiet im Niger (Bürkert und Lamers, 1999) exemplarisch zugrunde gelegt. Für die Verwendung des Zeitreihenansatzes, insbesondere des Zustandsraummodells und die Prognose zukünftiger Werte mittels Kalman Filter, werden die zur Luftaufnahme gehörigen Daten aus der Studie von Bürkert und Lamers (1999) verwendet.

5.1 Das Untersuchungsgebiet

Das an dieser Stelle analysierte Luftbild stammt aus einer Studie in Sadoré, Niger (13° 14‘N Breite, 2° 17‘E Länge). Das Dorf Sadoré liegt 35 km süd-östlich der Hauptstadt Nigers Niamey. Zwischen den Jahren 1992 und 1994 wurden dort Untersuchungen des Einflusses von Bodenbedeckung auf den Abtrag der Bodenoberfläche durchgeführt. Dazu wurde ein 171×25 m großes Gebiet in drei Bereiche eingeteilt. Jeder dieser drei Bereiche war seinerseits in neun Parzellen gegliedert. Auf diesen Parzellen wurden zufällig drei Arten von Bedeckung verteilt. Als Kontrolle wurde keine Bedeckung ausgebracht. Ferner wurden Hirsestengel einer gewissen Größe und Gewicht sowie Plastikröhrchen der gleichen Größe und Gewicht wie die Hirsestengel verteilt. Anschließend wurde über die Zeit gemessen, wie viel Boden jeweils in den Parzellen durch Wind und Wasser an- bzw. abgetragen wurde. Die Einzelheiten und Ergebnisse können der Studie von Bürkert und Lamers (1999) entnommen werden.

Zu bodenkundlichen Beobachtungen wurden Luftaufnahmen aus Fesseldrachen getätigt. Die Technik der Luftaufnahmen ist im Rahmen des SFB-Projekts „Adapted Farming in West Africa“ (SFB 308) entwickelt worden (Bürkert et al., 1996).

Die ebenfalls in Sadoré erhobenen Bodendaten dienen im weiteren Verlauf dazu, die zeitreihenanalytische Untersuchung (vgl. Abschnitt 5.3) zu tätigen. Aufgrund dieser Daten und insbesondere der in Erosionsstudien häufig anzutreffenden schlechten Datenlage, ist ein Modell entwickelt worden, welches es ermöglichen soll, Prognosen über die zukünftige Erosionsentwicklung dieses Gebietes zu machen.

Um einen Eindruck des Untersuchungsgebiets zu bekommen, sei Abb. 30 betrachtet.

Abb. 30: Luftaufnahme aus Sadoré, Niger. Die Aufnahme entstand im Rahmen einer Studie über Bodenveränderung und Hirsewachstum in der westlichen Sahelzone (Bürkert und Lamers, 1999).

Da diese Luftaufnahme mehr zeigt als den hier interessierenden Bereich, muß die Luftaufnahme zurechtgeschnitten werden. Gleichzeitig wird der Betrachtungswinkel so angepaßt, daß eine exakte Draufsicht entsteht (vgl. Abb. 31).

Abb. 31: Ausschnitt aus der Luftaufnahme aus Abb. 30. Diese Luftaufnahme hat **147420** Bildpunkte mit **10482** verschiedenen Farben.

Die Luftaufnahme in Abb. 30 besteht aus 744×495 (= 368280) Bildpunkten und besteht aus 27424 verschiedenen Farben. Im Vergleich dazu besteht die auf das Untersuchungsgebiet beschränkte Luftaufnahme in Abb. 31 aus 540×273 ($=147420$) Bildpunkten und 10482 verschiedenen Farben.
Das Gebiet ist in drei Gruppen zu je neun Parzellen eingeteilt. Um diese Struktur besser erkennen zu können, sei Abb. 32 betrachtet.

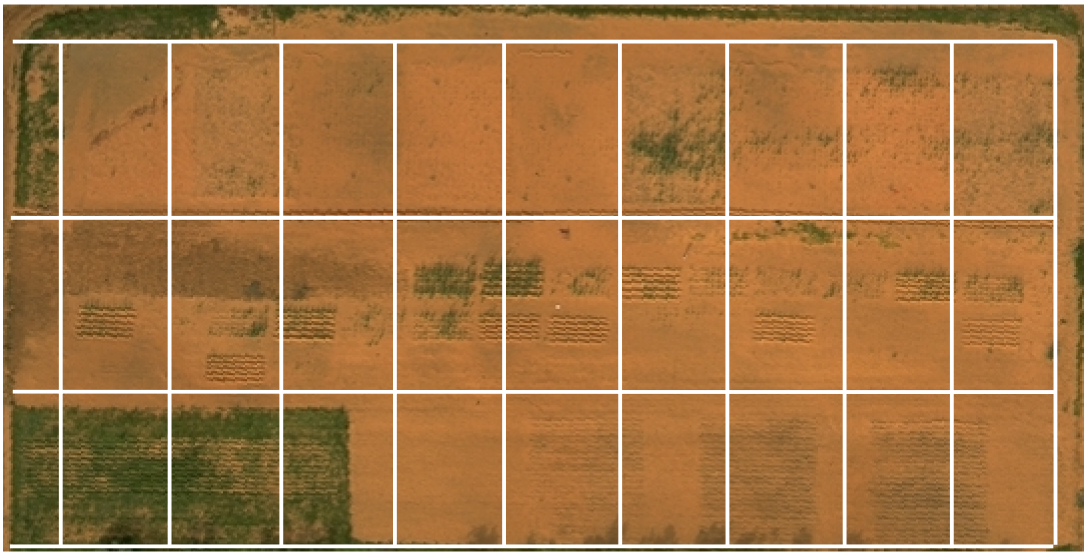


Abb. 32: Luftaufnahme aus Abb. 31 mit übergeblendetem Gitter zur Darstellung der Parzellenstruktur.

5.2 Vergleich der Clusterungsmethoden

Zur Clusterung des Luftbildes aus Abb. 31 werden die Methoden aus dem Kapiteln 1 und 2 angewendet. Das zur Algorithmusimplementierung entwickelte Programm Picana berechnet die Clusterung. Bei einer pixelbasierten Bildclusterung läßt sich das Ergebnis wieder als Bild darstellen. So erhält man schnell einen Überblick darüber, ob die entstandene Clusterung den Ansprüchen genügt. Die Ergebnisbilder einer Clusterung können dargestellt werden und ein direkter optischer Vergleich ist sofort möglich. Während bei den Maximum-Linkage-Verfahren die Anzahl der Lernzyklen keine Rolle spielt, müssen diese bei den neuronalen Netzen besonders berücksichtigt werden. Die Lernzyklenanzahl steht jedoch nicht von vornherein fest, sondern ist empirisch anhand des zu untersuchenden Bildes vorzunehmen. Im Anschluß an die Präsentation der geclusterten Bilder folgt eine Gegenüberstellung der Kenngrößen.

Abb. 33: Clusterung von Abb. 31 mit dem Maximum-Linkage-Verfahren bei 23 Clustern.

Abb. 34: Clusterung von Abb. 31 mit dem erweiteren Maximum-Linkage-Verfahren bei 23 Clustern und $\rho = 0{,}999$.

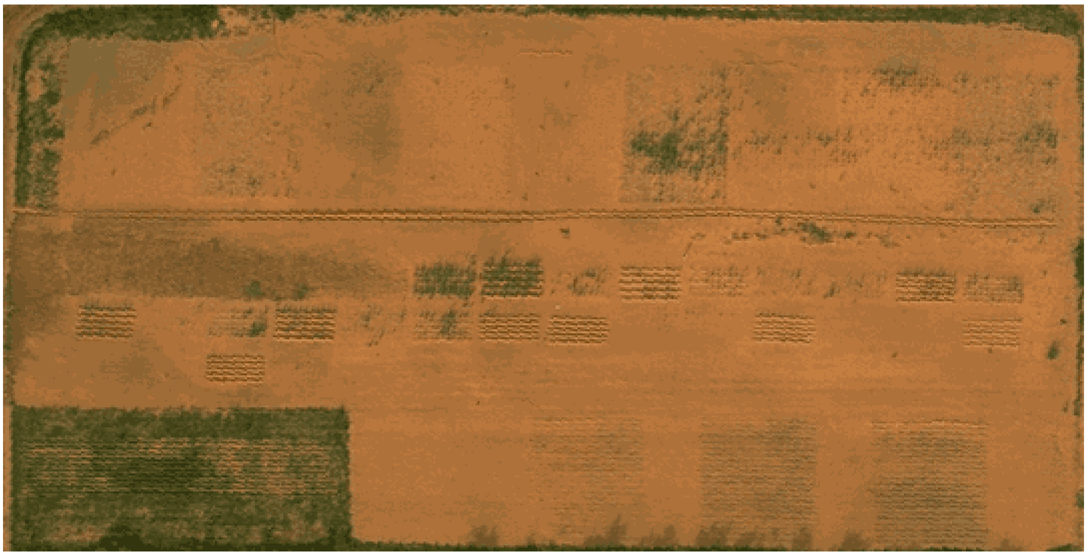


Abb. 35: Clusterung von Abb. 31 mit dem Winner-takes-all-Netz bei 23 Clustern und 80 Lernzyklen zum Trainieren des Netzes.

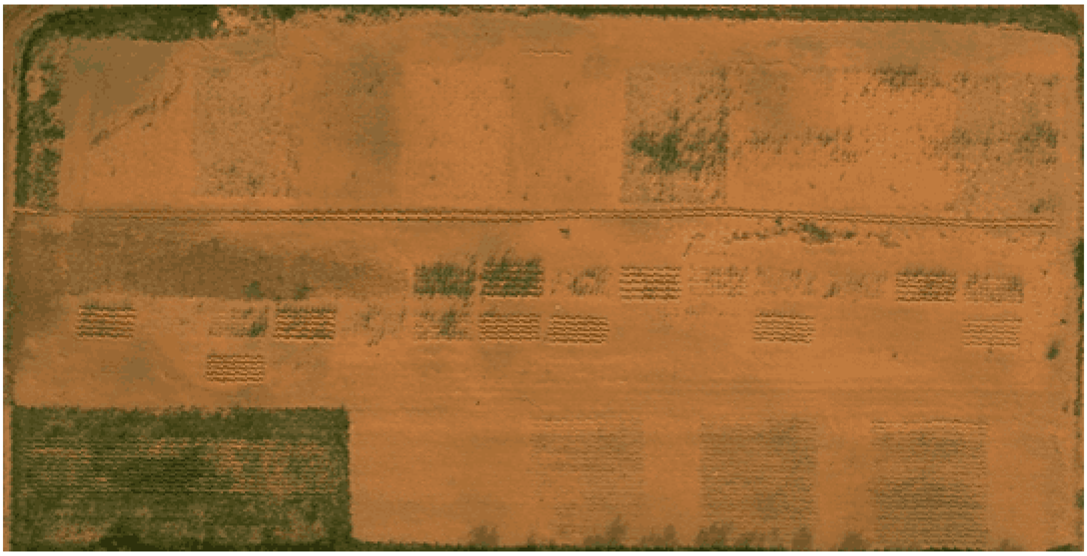


Abb. 36: Clusterung von Abb. 31 mit dem Standard-SOM mit einer quadratischen Nachbarschaftsstruktur, 23 Clustern und 2000 Lernzyklen zum Trainieren des Netzes.

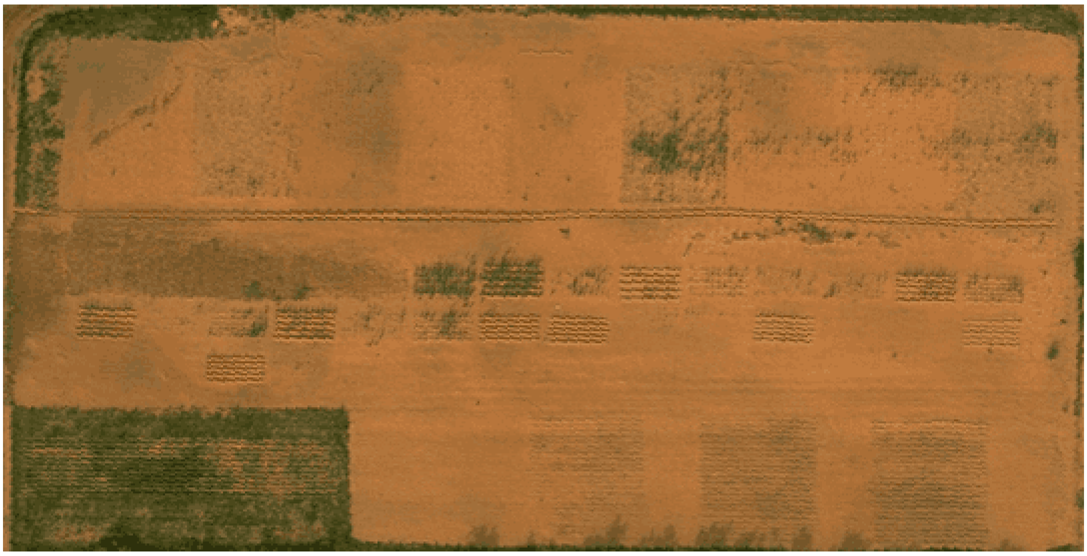


Abb. 37: Clusterung von Abb. 31 mit dem Standard-SOM mit einer hexagonalen Nachbarschaftsstruktur, **23** Clustern und **4000** Lernzyklen zum Trainieren des Netzes.

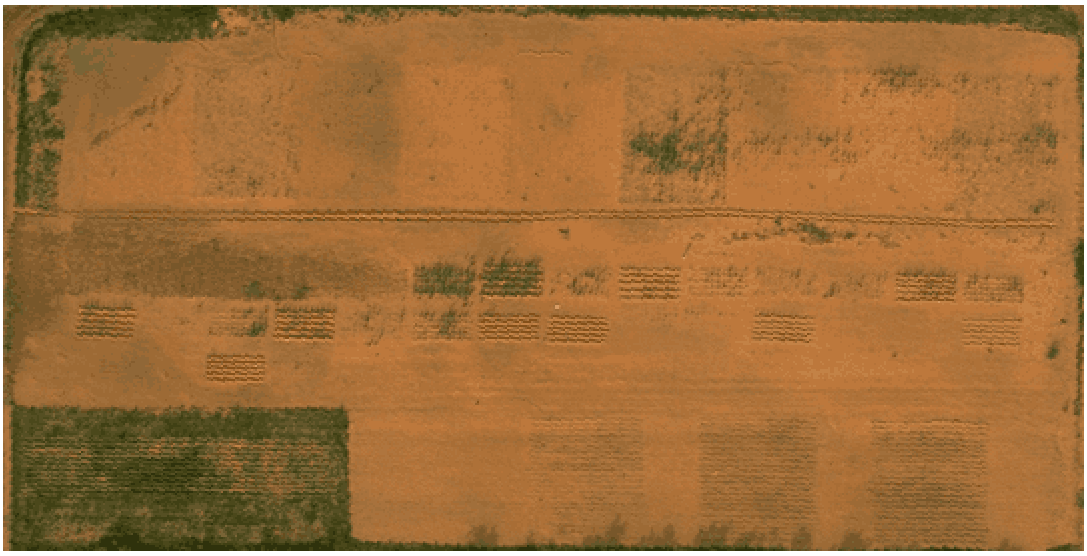


Abb. 38: Clusterung von Abb. 31 mit dem schnellen SOM mit einer quadratischen Nachbarschaftsstruktur, **23** Clustern und **40** Lernzyklen zum Trainieren des Netzes.

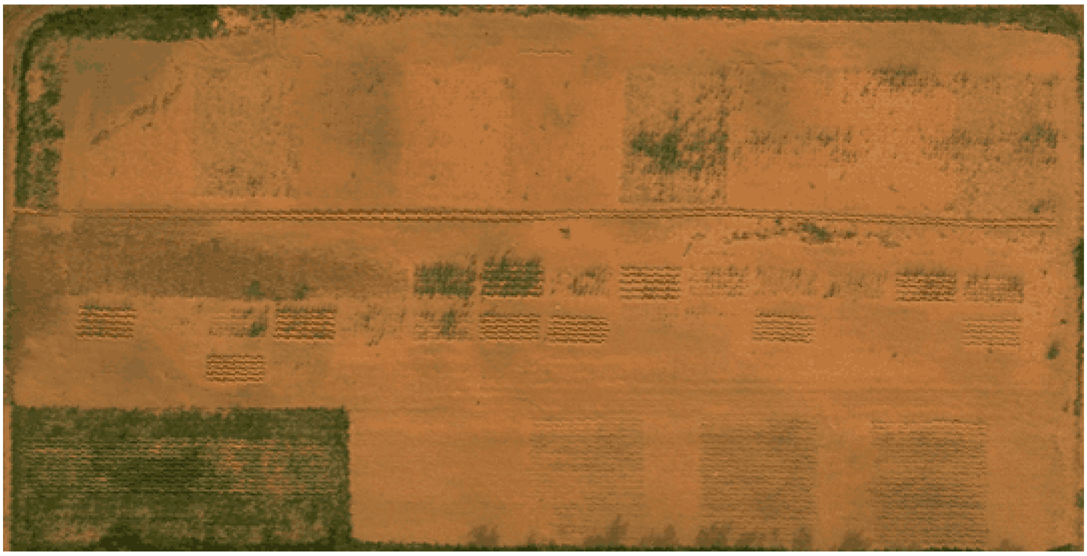


Abb. 39: Clusterung von Abb. 31 mit dem schnellen SOM mit einer hexagonalen Nachbarschaftsstruktur, 23 Clustern und 80 Lernzyklen zum Trainieren des Netzes.

Alle Ergebnisbilder aus den Abb. 33 bis Abb. 39 zeigen eine gute Repräsentation des Originalbildes aus Abb. 31. Alle Strukturen sind wiederzuerkennen. Es ist dabei zu bedenken, daß die geclusterten Bilder aus lediglich 23 Farben bestehen, im Vergleich zu den 10482 Farben aus dem Ursprungsbild. Beim Maximum-Linkage-Verfahren ist eine Tendenz zur Flächenbildung zu erkennen. Auch beim erweiterten Maximum-Linkage-Verfahren und ansatzweise beim WTAN sind Flächenbildungen zu beobachten. Durch die herausgebildeten Flächen ist eine Abgrenzung zwischen den Clustern gut ersichtlich. Es ist eben diese Forderung, die ursprünglich an ein Clusterungsverfahren gestellt wurde, damit es in der Erosionsüberwachung und -anaylse eingesetzt werden kann. Bei den Ergebnisbildern der neuronalen Netze ist auffällig, daß sie noch mehr dem Original entsprechen. Die Flächenbildung ist nicht mehr so ausgeprägt. Der Qualitätsunterschied der Ergebnisbilder, die mittels neuronaler Netze erzeugt wurden, ist optisch ebenfalls gering. Allerdings ist die Laufzeit zwischen den Methoden sehr unterschiedlich.

5.2.1 Laufzeit der Algorithmen

Obgleich das Ergebnisbild ähnlich gute Clusterungen für alle Methoden aufzeigt, verhält sich die Laufzeit sehr unterschiedlich. Für eine Clusterung mit Maximum Linkage werden 1,476 Minuten benötigt. Der erweiterte Maximum Linkage ist sogar noch schneller und benötigt nur 1,403 Minuten. Eine Clusterung mit einem

Winner-takes-all-Netz bei 80 Lernzyklen ist in 18,22 Minuten berechnet. Bei der Benutzung von Standard-SOM's werden mehrere Stunden benötigt. Bei einem Standard-SOM mit quadratischer Nachbarschaftsstruktur benötigt die Rechnung 260,394 Minuten (2000 Lernzyklen); mit einer hexagonalen Nachbarschaftsstruktur 600,656 Minuten (4000 Lernzyklen). Mit den speziell abgestimmten schnellen SOM's werden 5,774 Minuten (quadratisch, 40 Lernzyklen) bzw. 13,241 Minuten (hexagonal, 80 Lernzyklen) gebraucht. Bei der Betrachtung der Laufzeit fällt auf, daß das erweiterte Maximum Linkage nicht langsamer ist als Maximum Linkage. Desweiteren erkennt man, daß die Ausbildung der hexagonalen Struktur bei einem neuronalen Netz deutlich mehr Zeit und Lernzyklen benötigt. Das liegt u. a. daran, daß eine hexagonale Struktur schwerer auszubilden ist als eine einfache quadratische Struktur, weil Verbindungen zu mehr Nachbarn zu berücksichtigen sind. Selbst bei gleicher Zyklenanzahl benötigt eine Standard-SOM mit hexagonaler Struktur rund 22% mehr Rechenzeit als eine SOM mit quadratischer Struktur. Für eine Standard-SOM mit hexagonaler Struktur und 2000 Lernzyklen benötigte Picana 318,502 Minuten.

5.2.2 Vergleich der Methoden durch Kenngrößen

Zum Vergleich der Methoden werden die im Kapitel 4 vorgestellten und neu eingeführten Vergleichsgrößen bestimmt. Das Programm Picana berechnet die Intra-Class-Varianz und den mittleren Minimalabstand (*MMD*) zwischen den Zentroiden. Die Kenngrößen gewichtete mittlere Minimaldistanz (*GMMD*) bzw. der Gini-Koeffizient und den Quotient aus beiden wird mit einem SAS®-Programm berechnet (vgl. Anhang C). Für die vorgestellten Clusterungen (Abb. 33 - Abb. 39) sind die Kenngrößen Tab. 18 zu entnehmen.

Methode	**Lern-zyklen**	***SSW***	***MMD***	***GMMD***	***GMMDK***	**Gini-Koef.**
Maximum Linkage	/	85,44	448,99	381,42	868,69	0,19279
Erw. Maximum Linkage	/	72,36	495,92	439,66	1094,54	0,16135
WTAN	80	43,58	228,97	212,95	633,37	0,11304
S-SOM (quad.)	2000	33,76	189,74	178,20	404,88	0,19370

Methode	Lern-zyklen	*SSW*	*MMD*	*GMMD*	*GMMDK*	Gini-Koef.
S-SOM (hexg.)	4000	33,76	195,83	178,61	405,75	0,19378
Schnelle-SOM (quad.)	40	37,25	259,44	180,37	354,92	0,25827
Schnelle-SOM (hexg.)	80	39,08	170,80	151,145	323,98	0,21765

Tab. 18: Kenngrößen der Clusterungen des Luftbildes aus Abb. 31 mit den aufgeführten Methoden. Die vorgegebene Clusteranzahl ist 23. Das Ausgangsbild hat eine Gesamtvarianz von 1458,28. Die Zahlenwerte der Kenngrößen sind auf zwei Stellen nach dem Komma gerundet mit Ausnahme des Gini-Koeffizienten. Dieser dient als Divisor und soll möglichst genau sein.

Wird zur Beurteilung ausschließlich die Intra-Class-Varianz betrachtet, so wird die Clusterung ausgewählt, die den kleinsten Wert bzgl. dieser Größe aufweist. Aufgrund dessen ist die Clusterung mit einer Standard-SOM auszuwählen. Die Nachbarschaftsstruktur ist dabei nur von untergeordnetem Interesse. Nachfolgend sind die Schnellen-SOM's einzuordnen. Das WTAN weist einen noch etwas höheren Wert für SSW auf, so daß dieses Netz bzgl. dieses Kriterium weniger optimal ist. Das Maximum-Linkage-Verfahren sowie das erweitere Maximum-Linkage-Verfahren folgen mit größerem Abstand. Aufgrund der Clusterbildung der Linkage-Verfahren ist dieses Ergebnis zu erwarten. Die Gesamtvarianz des Ausgangsbildes beträgt 1458,28. In Bezug auf diesen Wert ist selbst eine Clusterung mit Maximum Linkage noch als „sehr gut" zu betrachten.
Zieht man die anderen Kriterien heran, so ist eine andere Reihenfolge zu erkennen. Aus den Ausführungen in Abschnitt 4.3.2 ist zu entnehmen, daß eine Clusterung mit einer großen mittleren Minimaldistanz zu wählen ist. Dieses Kriterium, als hinreichendes Kriterium bezeichnet, ist das letztlich zu verwendende Maß. Darüber hinaus ist auch die Konzentration der Werte zu berücksichtigen, also das *GMMDK*. Die sich ergebende Reihenfolge bzgl. dieser Kriterien ist in Tab. 19 abzulesen.

	Plazierung der Methode						
Kriterium	**ML**	**EML**	**WTAN**	**Schnelle-SOM (quad.)**	**Schnelle-SOM (hex.)**	**Stand.-SOM (quad.)**	**Stand.-SOM (hex.)**
MMD	2	1	4	3	7	6	5
$GMMD$	2	1	3	4	7	6	5
$GMMDK$	2	1	3	5	4	6	7
Gini-Koef.	3	2	1	7	6	4	5

Tab. 19: Rangfolge der Clusterungsverfahren bzgl. der verwendeten Beurteilungskriterien.

Es ist deutlich zu erkennen, daß die Linkage-Verfahren hier überlegen sind. Im vorderen Bereich ist ebenfalls das speziell angepaßte Winner-takes-all-Netz zu finden. Es folgen die schnellen-SOM's und erst dann die Standard-SOM's. Das bedeutet, daß die neuen Verfahren die Erwartungen erfüllen. Neben einer wesentlich verbesserten Laufzeit wird eine sehr gute Trennschärfe zwischen den Zentroiden erreicht. Aus Tab. 18 und Tab. 19 ist auch zu entnehmen, daß eine Auswahl nur bzgl. des Gini-Koeffizienten zu anderen Ergebnissen führt als eine Auswahl mit dem entwikkelten $GMMDK$.
Nachfolgend sind die Kriterienwerte für die verschiedenen Methoden bei einer unterschiedlichen Lernzyklenanzahl aufgelistet. Anhand dieser Ergebnisse werden die Ausführungen aus Abschnitt 4.3.3 untermauert. Eine Entscheidung nur aufgrund von SSW bzw. MMD kommt zu anderen Resultaten als eine Auswahl mit dem $GMMDK$.

Methode	**Zyklen**	**MMD**	**SSW**	**GMMD**	**GMMDK**	**G**
ML	-	448,99	85,44	381,42	868,69	0,19279
AML ρ ≤ 0,9	-	449,00	85,44	381,42	868,69	0,19279
AML ρ = 0,995	-	479,04	88,29	426,89	1018,36	0,17572
AML ρ = 0,999	-	495,92	72,36	439,66	1094,54	0,16135
AML ρ = 0,9995	-	476,07	61,28	410,91	1025,41	0,16058
WTAN	5	166,64	53,91	155,05	313,84	0,24407
	10	204,79	48,86	202,36	499,83	0,16391
	20	199,49	46,59	188,00	423,92	0,19667
	40	202,21	43,88	188,17	464,63	0,16401
	80	228,97	43,58	212,95	633,37	0,11304
	120	220,70	41,98	205,51	585,71	0,12311
	160	221,43	41,56	205,03	573,66	0,12774

Methode	Zyklen	MMD	SSW	GMMD	GMMDK	G
S-SOM-q	500	161,10	37,77	136,11	287,69	0,22382
	1000	172,84	35,45	154,95	345,75	0,20085
	2000	189,74	33,77	178,20	404,88	0,19370
	4000	189,77	34,27	171,60	380,40	0,20348
S-SOM-h	500	160,99	38,59	139,39	298,03	0,21874
	1000	179,48	35,50	161,96	351,62	0,21217
	2000	195,27	33,96	181,58	416,08	0,19044
	4000	195,84	33,76	178,61	405,75	0,19378
Q-SOM-q	5	513,18	46,98	183,56	297,96	0,37954
	10	388,94	45,33	158,22	262,16	0,36422
	20	306,31	43,45	160,78	276,77	0,33747
	40	259,44	37,25	180,37	354,92	0,25827
	80	167,30	40,03	138,94	276,97	0,25163
	120	158,97	40,79	137,23	279,89	0,24039
	160	171,64	38,23	151,07	318,87	0,22444
	2000	99,82	57,10	73,08	136,67	0,28593
Q-SOM-h	5	410,36	49,46	157,11	255,55	0,37798
	10	396,38	43,92	145,65	236,36	0,37971
	20	275,25	43,31	153,85	268,20	0,32906
	40	163,26	44,22	129,71	248,37	0,27272
	80	170,80	39,08	151,15	323,99	0,21765
	120	162,27	39,52	132,63	267,27	0,24626
	160	180,10	40,39	153,55	308,49	0,24777
	4000	267,86	72,07	55,07	84,87	0,42100

Tab. 20: Kenngrößen der Clusterungen des Bildes aus Abb. 31 mit verschiedenen Methoden. Die Clusteranzahl ist mit 23 vorgegeben. Das Ursprungsbild hat 10482 Farben und 147420 Bildpunkte. Die Zahleneinträge sind auf drei Nachkommastellen gerundet. Der Gini-Koeffizient auf fünf Stellen.

5.2.3 Visualisierung der Zentroide im Merkmalsraum

Neben dem Blick auf die entscheidenden Kenngrößen einer Clusterung ist es hilfreich, die Lage der berechneten Zentroide im Merkmalsraum zu betrachten. Anhand der nachstehenden Abbildungen kann die Lage der Zentroide nach der Clusterung in Augenschein genommen werden. Diese Darstellung ist Teil des berechneten Ergebnisses des Programms Picana. Aufgrund der besonderen Ausgangslage, daß ein Bild im RGB-Format vorliegt, läßt sich der Merkmalsraum in jeder Komponente auf den Bereich zwischen Null und 255 beschränken. Die resultierende Würfeldarstellung erlaubt auch einen dreidimensionalen Einblick auf die Datenlage.

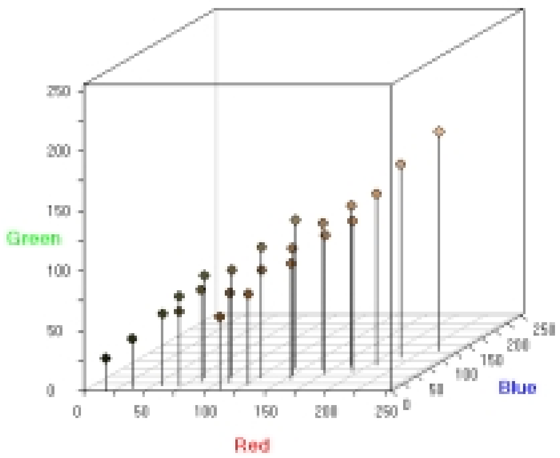


Abb. 40: Zentroide nach der Clusterung von Abb. 31 mit dem Maximum-Linkage-Verfahren. Clusteranzahl **23**.

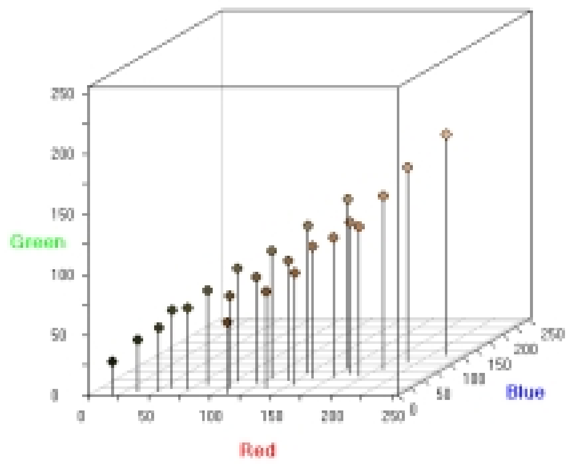


Abb. 41: Zentroide nach der Clusterung von Abb. 31 mit dem erweiterten Maximum-Linkage-Verfahren. Clusteranzahl **23** und $\rho = \mathbf{0{,}999}$.

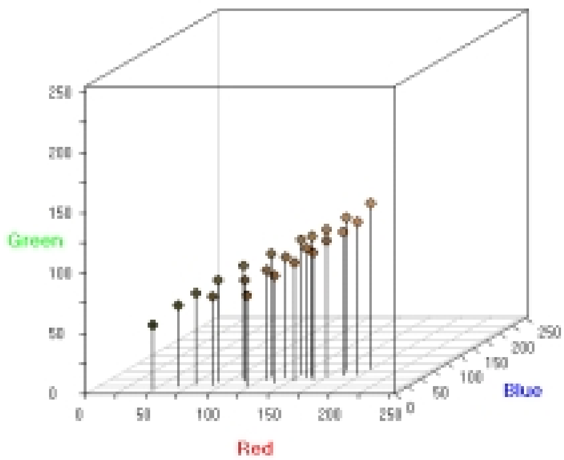


Abb. 42: Zentroide nach der Clusterung von Abb. 31 mit dem Winner-takes-all-Netz. Clusteranzahl **23** und **80** Lernzyklen..

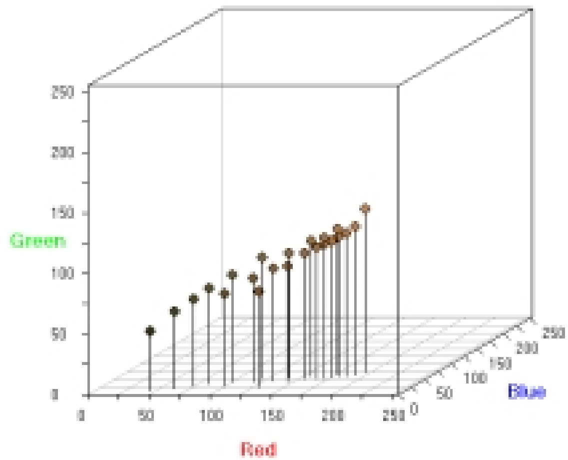


Abb. 43: Zentroide nach der Clusterung von Abb. **31** mit einer Standard-SOM bei quadratischer Nachbarschaftsstruktur. Clusteranzahl **23** und **2000** Lernzyklen.

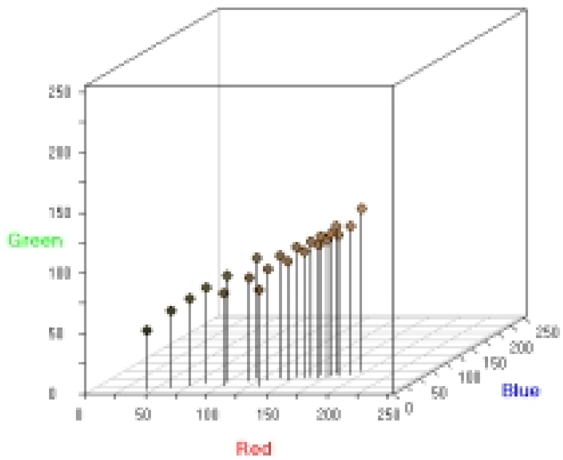


Abb. 44: Zentroide nach der Clusterung von Abb. **31** mit einer Standard-SOM bei hexagonaler Nachbarschaftsstruktur. Clusteranzahl **23** und **4000** Lernzyklen.

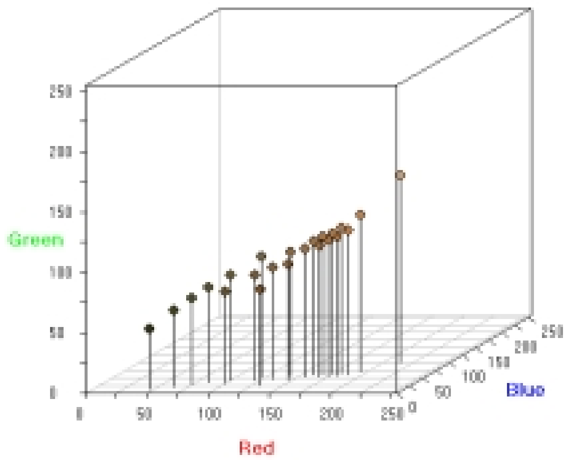


Abb. 45: Zentroide nach der Clusterung von Abb. **31** mit einer schnellen SOM bei quadratischer Nachbarschaftsstruktur. Clusteranzahl **23** und **40** Lernzyklen.

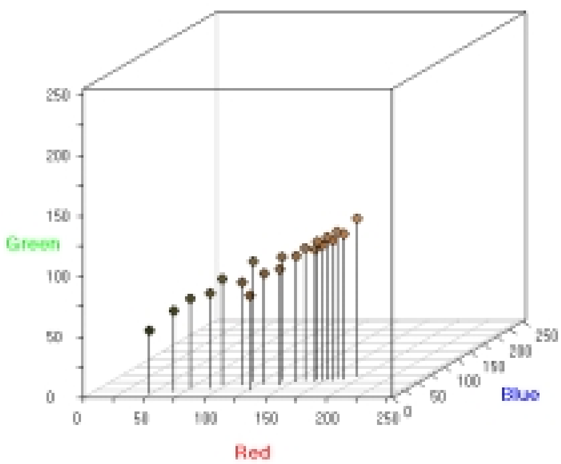


Abb. 46: Zentroide nach der Clusterung von Abb. **31** mit einer schnellen SOM bei hexagonaler Nachbarschaftsstruktur. Clusteranzahl **23** und **80** Lernzyklen.

Die Darstellungen der Zentroide im Merkmalsraum zeigen, daß alle Clusterungen einen Verlauf der Zentroide entlang einer Geraden aufweisen. Es hat sich eine annähernd lineare Struktur gebildet. Die Abb. 43 bis Abb. 46 zeigen große Ähnlichkeiten. Alle mit den SOM's berechneten Zentroiddarstellungen lassen erkennen, daß sich die gefundenen Cluster deutlicher in der Mitte des Merkmalsraumes konzentrieren. Die einzelnen Cluster sind dicht beieinander. Bei den schnellen SOM's mit quadratischer Nachbarschaftsstruktur gibt es ferner ein Wert, der sich deutlich von den anderen SOM-Darstellungen abhebt. Ein Cluster im oberen rechten Bereich ist abgesetzt von den übrigen. Die Trennung zwischen den Clustern ist nicht sehr hoch. Eine bessere Trennung zwischen den Clustern erreicht die Darstellung des

Winner-takes-all-Netzes (Abb. 42). Trotzdem ist die Trennung längst nicht so gut wie bei den Linkage-Verfahren. Bei diesen sind die Zentroide deutlich besser im Merkmalsraum verteilt, wodurch eine bessere Trennung zwischen den Clustern erreicht wird. Die Linkage-Verfahren finden außerdem vorhandene seltene Elemente im unteren bzw. oberen Bereich der Werteskala. Bei der Darstellung des erweiterten Maximum Linkage (Abb. 41) fällt zudem auf, daß dieser Algorithmus im mittleren Wertebereich geringfügig besser die Cluster trennt als der Maximum-Linkage-Algorithmus. Mit der Darstellung der Zentroide im Merkmalsraum kann optisch die Qualität der Clusterung beurteilt werden, da ein Eindruck davon entsteht, ob das Verfahren seltene Elemente findet und ob die Cluster hinreichend gut getrennt sind. Aufgrund dessen ist es ebenfalls möglich, die notwendige Clusteranzahl zu ermitteln. Warum haben die mit den verschiedenen Methoden durchgeführten Clusterungen eine Clusteranzahl von 23 Cluster? Diese Anzahl ermittelt sich aufgrund folgender Überlegungen. Analog zu Zerbst (2001), der Luftbilder ähnlichen Typs analysiert hat, werden Clusterungen mit 25 Cluster angesetzt. Die Visualisierung der Zentroide im Merkmalsraum für die 25 Cluster ist Abb. 47 zu entnehmen.

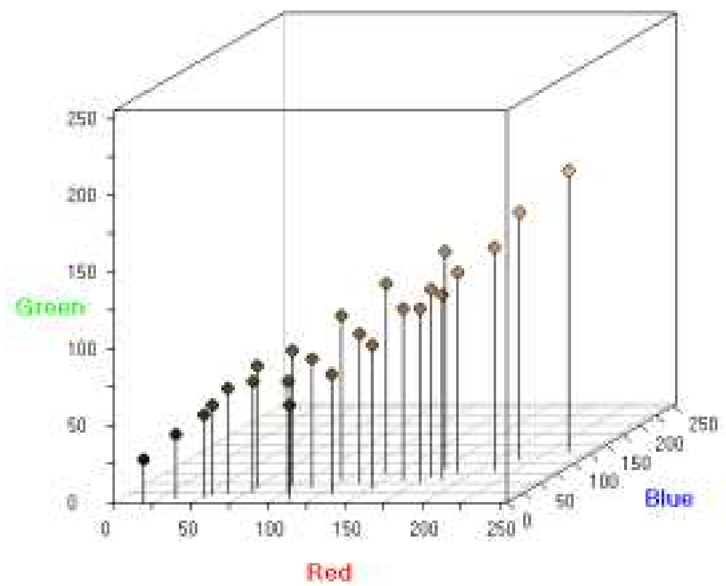


Abb. 47: Visualisierung der Zentroide im Merkmalsraum für eine Clusterung mit der Maximum-Linkage-Methode bei 25 Clustern

Es zeigt sich, daß einige Zentroide sehr dicht beieinander liegen, also die Trennung zwischen den Clustern nicht überall gut ist. Für diese Clusterung ergibt sich ein $GMMDK$ von 811,08. Um eine Verbesserung bzgl. der Trennschärfe zu erreichen, betrachten wir eine Clusterung mit 24 Clustern. Dafür ergibt eine $GMMDK$ von 817,60. Der Unterschied zwischen diesen Werten ist gering. Für die Trennschärfe bedeutet es keine hinreichende Verbesserung. Daher reduzieren wir die Clusteranzahl erneut. Für eine Clusterung mit 23 Clustern ergibt sich eine wesentliche Steigerung des $GMMDK$ auf 868,69. Die Verbesserung zeigt sich auch optisch durch einen Vergleich der Abb. 47 mit Abb. 40. Eine weitere Reduktion der Clusteranzahl auf 22 Cluster bewirkt nur eine geringe Steigerung des $GMMDK$ auf 875,09. Ferner vergrößert sich der Gini-Koeffizient beim Übergang von 23 zu 22 Clustern, was auf eine ungleichere Verteilung der Minimaldistanzen schließen läßt. Die besten Ergebnisse bei dieser Analyse ergeben sich daher bei der Verwendung von 23 Clustern.

Neben der Visualisierung der Zentroide im Merkmalsraum können zusätzlich bei den neuronalen Netzen die U-Matrix, also die Nachbarschaftsbeziehung, betrachtet werden (vgl. Abschnitt 2.5.1) Das Zentrum der als Flächen dargestellten quadratischen oder hexagonalen Struktur bildet je ein Neuron. Die Visualisierung der U-Matrix zeigt dann eine „Karte“ an. Diese „Karte“ verändert sich im Laufe der Lernzyklen und das Endergebnis kann dann dargestellt werden. Kohonen (2001) weist zudem daraufhin, daß viele Neuronen und damit eine große „Karte“ sich besser über den Merkmalsraum ausbreiten können als kleine. Das Ausbreiten der „Karte“ wird auch als Entwicklung der „Karte“ bezeichnet. Durch empirische Auswertungen unterschiedlicher Luftbilder bei verschiedenen Lernzyklen konnte dieser Sachverhalt ebenfalls bestätigt werden. Ferner entwickeln sich quadratische „Karten“, also „Karten“ mit einer quadratischen Anzahl von Neuronen besser, als wenn die Besetzung des Neuronengitters nicht vollständig ist (vgl. auch Zerbst, 2001). Die Lernzyklenanzahl muß zur vollständigen Entwicklung einer „Karte“ sehr hoch gewählt werden. Kohonen (2001) weist darauf hin, daß es mehrere 10000 Lernzyklen sein sollten, erwähnt aber auch, daß dadurch das Einsatzgebiet aufgrund der extrem langen Laufzeit begrenzt ist.

Eine entwickelte selbstorganisierende Karte ordnet die Neuronen so an, daß ähnliche (Farb-) Werte nebeneinander liegen und die (Farb-) Veränderung systematisch in jede Richtung der „Karte“ auftritt. Exemplarisch sei an dieser Stelle eine U-Matrix-Darstellung einer Standard-SOM mit quadratischer Nachbarschaftsstruktur, 23 Clustern und 10000 Lernzyklen dargestellt. Die Berechnung dieser „Karte“ benötigte 1482,48 Minuten (= 24,71 Std.). Es ist zu sehen, daß selbst bei der unteren Lernzyklenanzahl, die Kohonen vorschlägt, die Laufzeit extrem lang wird.

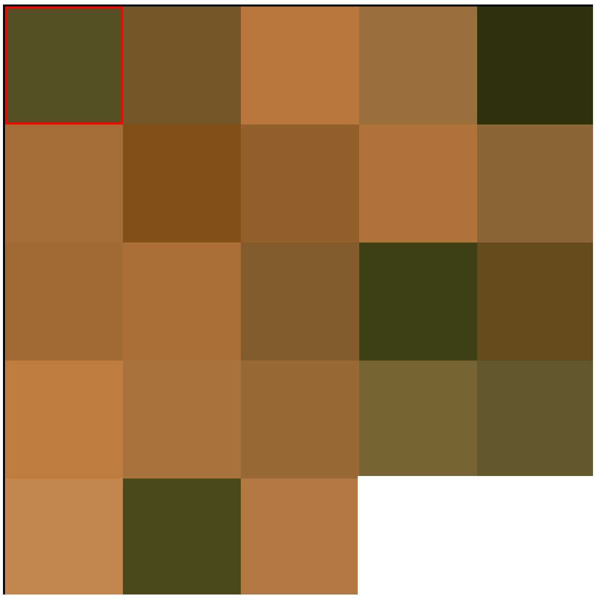


Abb. 48: U-Matrix-Darstellung der Clusterung von Abb. 31 mit einer Standard-SOM mit quadratischer Nachbarschaftsstruktur, 23 Clustern und 10000 Lernzyklen.

Ferner ist die U-Matrix Darstellung aus Abb. 48 noch nicht so entwickelt wie es zu erwarten stand. Ähnliche Farbwerte sind nicht benachbart. Eine als gut entwickelte Darstellung ist die in Abb. 49 gegeben U-Matrix. Dort ist zu erkennen, wie die Zunahme der Zentroidanzahl sich positiv auf die Entfaltung der U-Matrix-Darstellung auswirkt. Allerdings erscheinen die gewählten 100 Cluster für die Praxis zu viel.

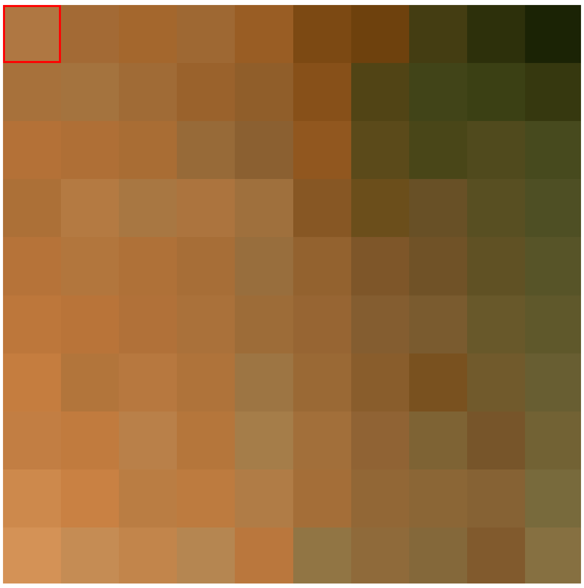


Abb. 49: Beispiel für eine gut entfaltete U-Matrix-Darstellung. Clusterung mit 100 Clustern und 1000 Lernzyklen mit einer Standard-SOM mit quadratischer Nachbarschaftsstruktur.

Anhand der Darstellung ist zu erkennen, daß sich farblich zusammengehörige Quadrate in einen Bereich zusammenziehen. Die dunklen Grüntöne befinden sich oben rechts, die braun Abstufungen folgen nach links unten hin. Die berechnete SSW beträgt $10{,}20$, was auf eine extrem gute Clusterung gemäß dieses Kriteriums schließen läßt. Der Wert von MMD ist entsprechend sehr niedrig, nämlich $64{,}424$.
Zum Vergleich dazu sei eine U-Matrix-Darstellung mit 100 Clustern und 1000 Lernzyklen für eine Standard-SOM mit hexagonaler Nachbarschaftsstruktur betrachtet. Die SSW beträgt $10{,}21$. Die mittlere Minimaldistanz MMD $64{,}28$. Zur Berechnung sind $368{,}52$ Minuten benötigt worden. Deutlich zu erkennen ist die erwähnte Tatsache, daß sich hexagonale Strukturen schlechter entwickeln als quadratische.

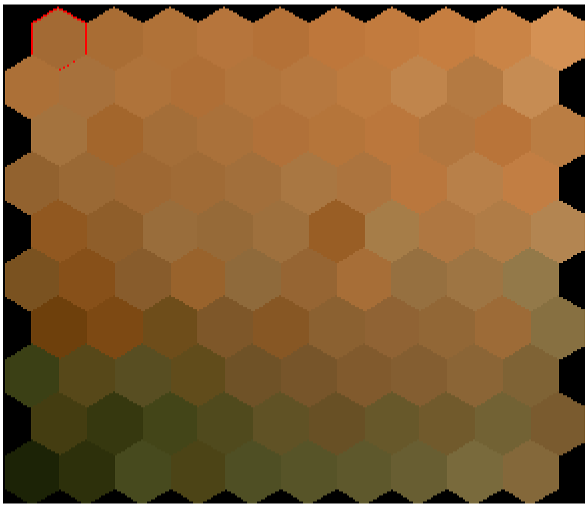
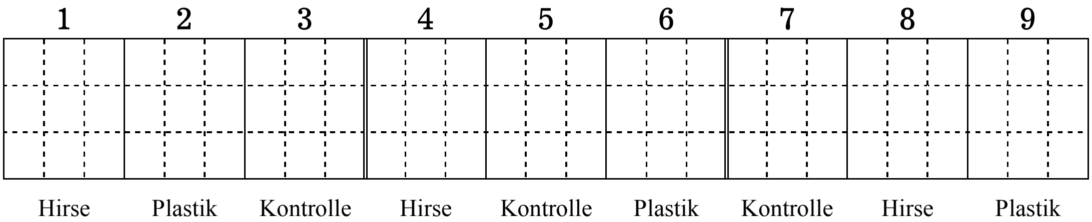


Abb. 50: Beispiel für eine gut entfaltete U-Matrix-Darstellung. Clusterung mit 100 Clustern und 1000 Lernzyklen mit einer Standard-SOM mit hexagoanler Nachbarschaftsstruktur.

5.3 Analyse der Bodendaten

Die in der Studie von Bürkert und Lamers (1999) gesammelten Bodendaten über die Bodenveränderung nach Wind- oder Wasserereignissen auf das in 5.1 beschriebene Untersuchungsgebiet weisen leider die gleichen Unzulänglichkeit auf wie viele Erosionsdatensätze. Da die Bodenänderung nur gemessen werden kann, wenn entweder ein Sturm- oder aber ein Regenfallereignis vorkamen, sind die gesammelten Daten naturgemäß unvollständig. Im vorliegenden Fall gab es zwar hinreichend viele Beobachtungsparzellen und Meßstellen auf diesen. Die Zeitpunkte, an denen tatsächlich die Bodenänderung gemessen werden konnte, sind jedoch gering.



Tab. 21: Schematische Darstellung der Parzelleneinteilung des Untersuchungsgebiets.

In einem Zeitraum von zwei Jahren konnten aufgrund der vorherrschenden Klimabedingung nur elf Beobachtungen an den Meßstellen gesammelt werden. Für eine Zeitreihenanalyse sind das zu wenige Beobachtungen. Um trotz dieses Mißstandes die später folgenden Analysemethoden aufzeigen zu können, sind stochastische Prozesse für jede Meßstelle innerhalb einer Gruppe simuliert worden. Dadurch steht eine ausreichende Datenmenge zur Verfügung. Die Verläufe der Prozesse entsprechen den Beobachtungen von Bürkert und Lamers (1999). Die auftretenden Gruppen sind eine Kontroll-Gruppe (keine Bodenbedeckung), eine Hirse-Gruppe (Bodenbedeckung mit Hirsestengel) sowie eine Plastik-Gruppe (Bodenbedeckung mit Plastikröhrchen, vgl. Abschnitt 5.1). In der Studie ist beobachtet worden, wie die Bodenveränderung sich über die Zeit entwickelt. Die Entwicklung wurde in jeder Gruppe durchgeführt. Das graphische Resultat ist in Abb. 51 dargestellt.

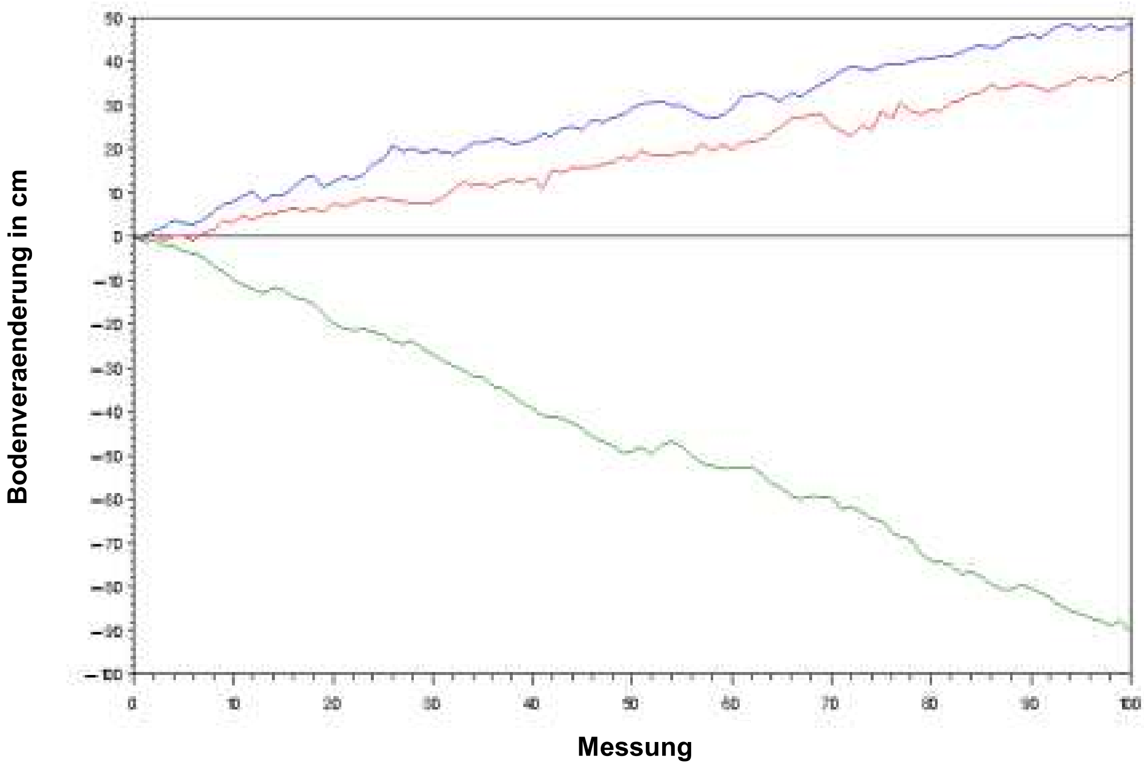


Abb. 51: Simulierte stochastische Prozesse des Bodenveränderung in den Gruppen Kontrolle (untere Kurve = grün), Hirse (obere Kurve = blau) und Plastik (mittlere Kurve = rot). das Ergebnis entspricht den Beobachtungen der Studie von Bürkert und Lamers (1999).

Die Prozeßgleichungen der simulierten Prozesse lauten für die Hirse-Gruppe:

$$y_t = 0{,}45\, y_{t-1} + 0{,}17\, y_{t-2} + 0{,}3\, \varepsilon_{t-1} + 0{,}3\, \varepsilon_{t-2} + \varepsilon_t \quad , \tag{5.1}$$

für die Plastik-Gruppe:

$$y_t = 0{,}38\, y_{t-1} + 0{,}15\, y_{t-2} + 0{,}2\, \varepsilon_{t-1} + 0{,}34 \varepsilon_{t-2} + \varepsilon_t \quad , \tag{5.2}$$

und für die Kontroll-Gruppe:

$$y_t = 0{,}8\, y_{t-1} + \varepsilon_t \quad . \tag{5.3}$$

Man beachte, daß die Kontroll-Gruppe einem AR(1)-Prozeß folgt. Im vorliegenden Fall bedeutet das, daß der Bodenverlust zum Zeitpunkt t von seinem vorherigen Zustand und von einem Zufallsschock abhängig ist. In diesem Zufallsschock sind die Wetterveränderungen enthalten. Für die Hirse-Gruppe soll die Analyse an dieser Stelle erläutert werden. Die Analyse der anderen beiden Gruppen verläuft dann analog.

Diese simulierten Daten sollen so betrachtet werden, als ob diese die real erhobenen Daten sind. Durch dieses Vorgehen soll die komplette Analyse betrachtet werden, die im Falle von Erosionsdaten durchzuführen sind. Die nachstehende Analyse behandelt daher die simulierten Zeitreihen so, als ob sie die Meßdaten der erwähnten Studie sind, um aufzuzeigen, wie die Analyse im allgemeinen ablaufen sollte. Insofern erscheint es dann auch legitim, von keinem Vorwissen über die Beschaffenheit der Zeitreihen respektive stochastischen Prozesse auszugehen und die Analyse zu starten. Ausgehend von einem ersten optischen Eindruck der Zeitreihe beginnt die Analyse. Da sich die Zeitreihen mit einem Trend behaftet präsentieren, ist zunächst eine Trendbereinigung notwendig. Dazu betrachtet man die Residuen einer Zeitreihe als trendbereinigt. Mit diesen trendbereiningten Zeitreihen werden die weiteren Analysen durchgeführt.

Für einen ersten optischen Eindruck sei die trendbereinigte Reihe für eine Meßstelle in der Hirse-Gruppe betrachtet. Dazu sind die berechneten Autokorrelationsfunktion (ACF), partielle Autokorrelationsfunktion (PACF) sowie das geschätzte Spektrum dargestellt. Jede Zeitreihe (Meßstelle) in jeder Gruppe ist so untersucht worden. Anhand dieser Darstellungen läßt sich überprüfen, ob die Zeitreihen die gewünschte Eigenschaft der Stationarität aufweist. Für alle Zeitreihen konnte eine hinreichend gute Stationarität für die praktische Zeitreihenanalyse nachgewiesen werden. Einen Überblick darüber geben die für die erste Meßstelle der Hirse-Gruppe dargestellte trendbereinigte Zeitreihe nebst ACF, PACF und Periodogramm.

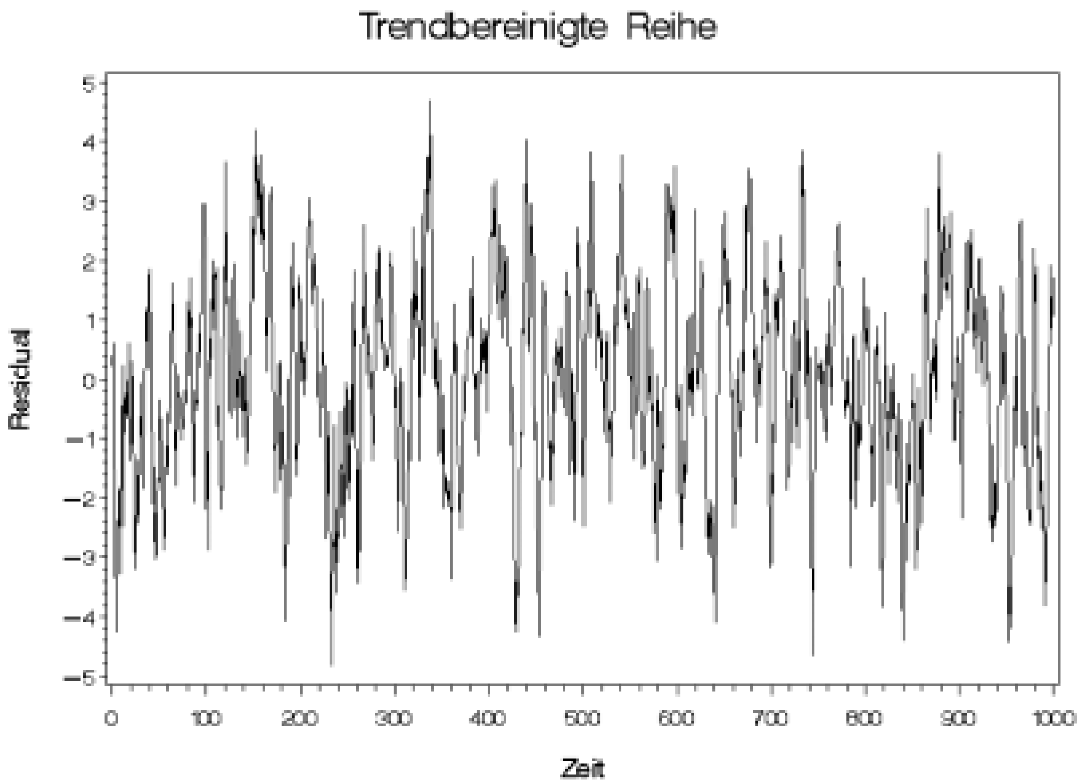


Abb. 52: Beispiel für die trendbereiningte Zeitreihe eines Beobachtungspunktes der Hirse-Gruppe.

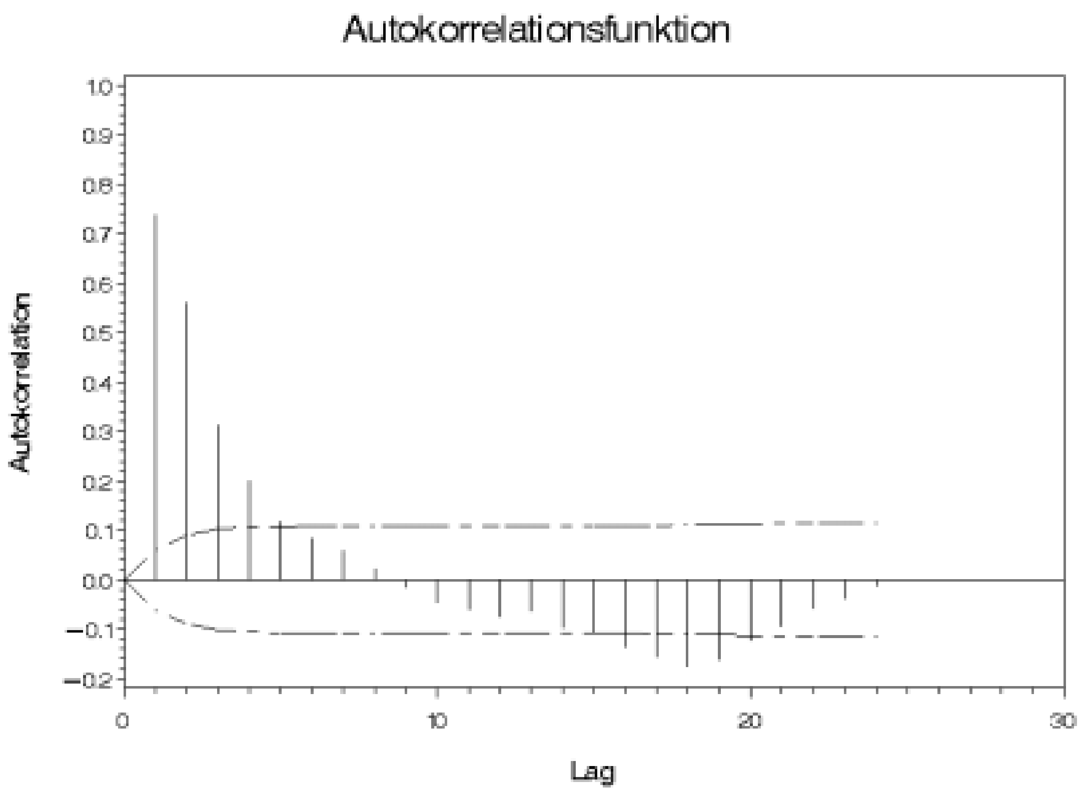


Abb. 53: Autokorrelationsfunktion der Zeitreihe aus Abb. 52.

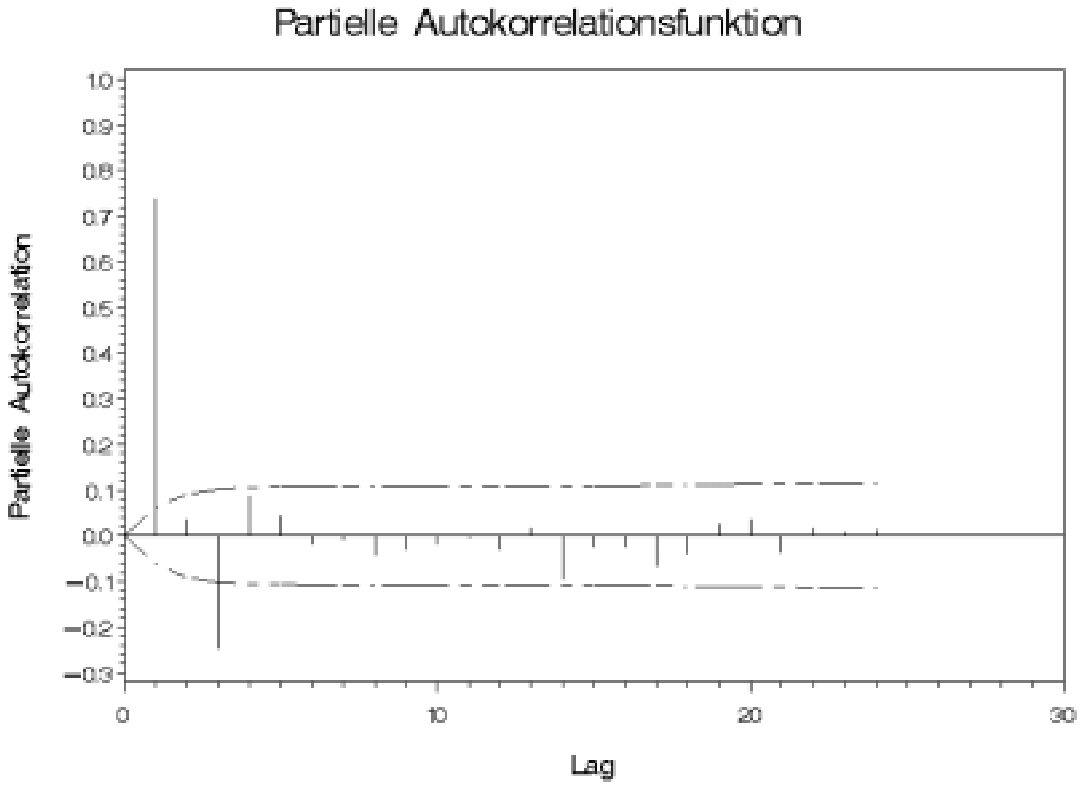


Abb. 54: Partielle Autokorrelationsfunktion der Zeitreihe aus Abb. 52.

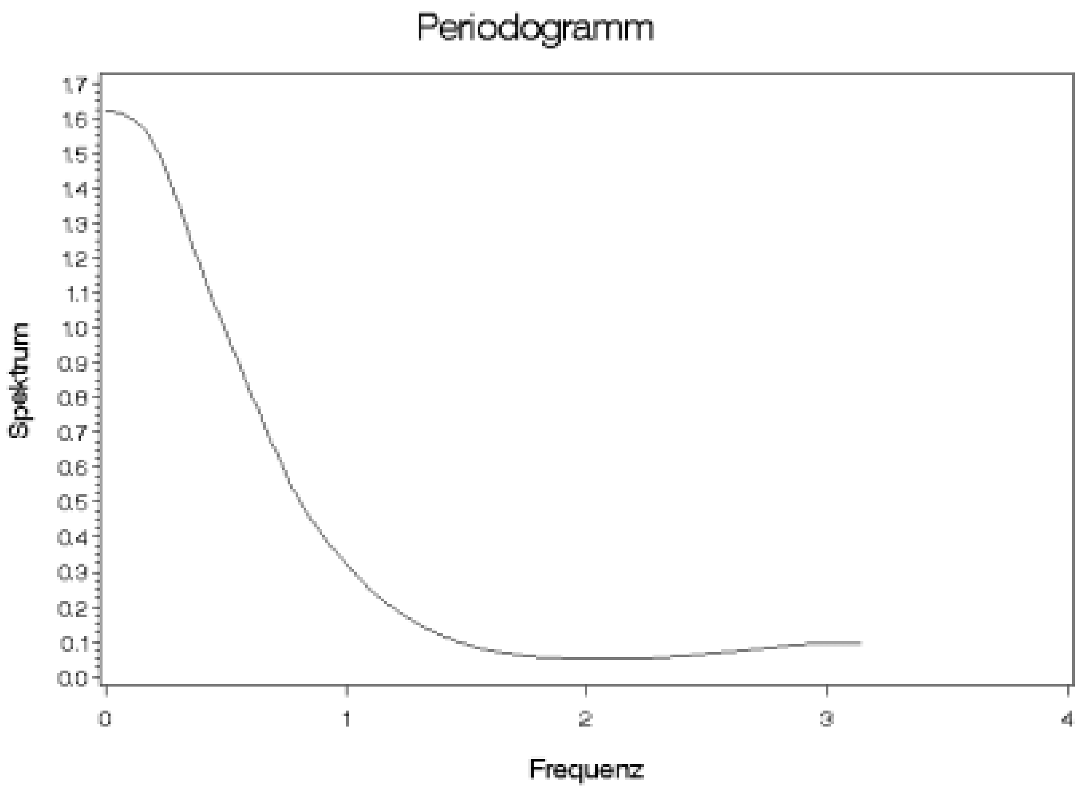


Abb. 55: Geschätztes Spektrum der Zeitreihe aus Abb. 52.

Blickt man auf die Darstellung der trendbereinigten Zeitreihe (Abb. 52), so ist keine Regelmäßigkeit im Verlauf zu sehen. Auch ein Strukturbruch oder langes Gedächtnis ist nicht zu erkennen. Die ACF (Abb. 53) bzw. PACF (Abb. 54) mit ihrem Konfidenzbereichen zeigen keine besondere Auffälligkeiten und deuten auf einen ARMA-Prozeß hin. Die PACF weist zudem auf einen starken AR-Anteil hin, der um die Ordnung 2 oder 3 schwankt. Gleichung (5.1) zeigt, daß die Koeffizienten des AR-Anteils einen großen Einfluß auf die Zeitreihe haben, insbesondere der Koeffizient des AR(1)-Anteils dominiert. Eine genauere Analyse der Prozeßordnung liefert das gewünschte Ergebnis. Das Analyseprogramm befindet sich im Anhang D. Das Periodogramm zeigt ebenfalls keine Auffälligkeiten.
Beispielhaft sei an dieser Stelle ein Resultat der ARIMA Prozedur des SAS-Systems vorgestellt. Anhand der nachfolgenden Tabellen (Tab. 22, Tab. 23) ist die Ordnung des analysierten Prozesses zu erkennen. Dabei ist in den Tabellen ein möglichst großes Rechteck zu suchen, in dem alle Elemente nicht signifikant sind (5%-iges Testniveau). Tritt mehr als eine Möglichkeit auf, ein Rechteck dergestalt zu bilden, gilt das Gesetz der Sparsamkeit. In diesem Fall ist das Modell zu wählen, welches die kleinste Ordnung aufweist. Die Methode ist von Tsay und Tiao (1985) entwickelt und von Box et al. (1994) verbessert worden. Zur Verdeutlichung ist das entsprechende Rechteck deutlich umrandet hervorgehoben. Ist das Rechteck bestimmt, so wird die obere linke Ecke gewählt, um die Prozeßordnung zu bestimmen. In diesem Fall erkennen wir ein ARMA(2,2)-Modell.

Squared Canonical Correlation Estimates						
Lags	**MA 0**	**MA 1**	**MA 2**	**MA 3**	**MA 4**	**MA 5**
AR 0	0.6208	0.4022	0.1791	0.0991	0.0627	0.0444
AR 1	0.0013	0.0527	0.0070	0.0018	0.0008	0.0003
AR 2	0.0522	0.0271	0.0047	<.0001	<.0001	<.0001
AR 3	0.0112	0.0173	0.0041	<.0001	<.0001	<.0001
AR 4	0.0102	0.0073	0.0017	<.0001	<.0001	<.0001
AR 5	0.0009	<.0001	0.0008	0.0003	<.0001	<.0001

Tab. 22: Beispiel für einen „Output“ aus dem SAS-System, erzeugt von der Prozedur ARIMA für die simulierte Zeitreihe des ersten Meßpunktes der Hirse-Gruppe.

SCAN Chi-Square[1] Probability Values						
Lags	**MA 0**	**MA 1**	**MA 2**	**MA 3**	**MA 4**	**MA 5**
AR 0	<.0001	<.0001	<.0001	<.0001	<.0001	0.0006
AR 1	0.2599	<.0001	0.0165	0.2032	0.3957	0.6089
AR 2	<.0001	<.0001	0.0718	0.8666	0.9990	0.8527
AR 3	0.0008	0.0001	0.1107	0.9875	0.8300	0.9475
AR 4	0.0014	0.0199	0.2306	0.8454	0.9353	0.9551
AR 5	0.3328	0.8835	0.4038	0.6317	0.9613	0.9357

Tab. 23: Beispiel für einen „Output“ aus dem SAS-System, erzeugt von der Prozedur ARIMA für eine simulierte Zeitreihe des ersten Meßpunktes der Hirse-Gruppe.

5.3.1 Zustandsraummodell

Zur Bestimmung des Zustandsraummodells ist für die 36 Meßstellen der drei Gruppen (Hirse, Plastik und Kontrolle) der Trend aus den Zeitreihen entfernt und Stationarität sichergestellt worden (vgl. 5.3). Zur korrekten Bestimmung des Zustandsraummodells ist die Stationarität notwendig. Exemplarisch sei die Anpassung eines solchen Modells für die Hirse-Gruppe erläutert. Die trendbereinigten Zeitreihen der verschiedenen Meßstellen einer Gruppe werden dabei zusammengefaßt. Die Berechnung des Zustandsraummodells erfolgt unter Zuhilfenahme des in Anhang D befindlichen SAS®-Programms.
Die Initialisierung des Zustandsraummodells (Gleichung (3.1) bis (3.4)) ist gegeben durch:

Der Zustandsvektor α_t beinhaltet eine Variable pro Meßstelle zum Zeitpunkt t. Aus diesem Grund wird auf eine genauere Angabe verzichtet. Die Übergangsmatrix ist gegeben durch:

$$T_t^{Init} = \begin{bmatrix} T_t^{11} & T_t^{12} & T_t^{13} & T_t^{14} & T_t^{15} & T_t^{16} \\ T_t^{21} & T_t^{22} & T_t^{23} & T_t^{24} & T_t^{25} & T_t^{26} \end{bmatrix} .$$

Die Matrizen T_t^{11} bis T_t^{26} des Initialisierungsschritts sind in Anhang E nachzulesen. Die geschätzte Kovarianzmatrix des Fehlerterms der Initialisierungsphase hat die Gestalt:

$$Q_t^{Init} = \begin{bmatrix} Q_t^{11} & Q_t^{12} & Q_t^{13} & Q_t^{14} & Q_t^{15} & Q_t^{16} \\ Q_t^{21} & Q_t^{22} & Q_t^{23} & Q_t^{24} & Q_t^{25} & Q_t^{26} \end{bmatrix} .$$

Die Matrizen Q_t^{11} bis Q_t^{26} des Initialisierungsschritts sind in Anhang E nachzulesen.

Die Matrix $\boldsymbol{R}_t$ aus (vgl. Gleichung (3.3)) entspricht einer Einheitsmatrix der Dimension 36×36.
Für das angepaßte Modell ergibt sich, unter Einbeziehung der Initialisierung, folgende Übergangsmatrix:

$$\boldsymbol{T}_t = \begin{bmatrix} \boldsymbol{T}_t^{11} & \boldsymbol{T}_t^{12} & \boldsymbol{T}_t^{13} & \boldsymbol{T}_t^{14} & \boldsymbol{T}_t^{15} & \boldsymbol{T}_t^{16} \\ \boldsymbol{T}_t^{21} & \boldsymbol{T}_t^{22} & \boldsymbol{T}_t^{23} & \boldsymbol{T}_t^{24} & \boldsymbol{T}_t^{25} & \boldsymbol{T}_t^{26} \end{bmatrix} .$$

Für die geschätze Kovarianzmatrix ergibt sich:

$$\boldsymbol{Q}_t = \begin{bmatrix} \boldsymbol{Q}_t^{11} & \boldsymbol{Q}_t^{12} & \boldsymbol{Q}_t^{13} & \boldsymbol{Q}_t^{14} & \boldsymbol{Q}_t^{15} & \boldsymbol{Q}_t^{16} \\ \boldsymbol{Q}_t^{21} & \boldsymbol{Q}_t^{22} & \boldsymbol{Q}_t^{23} & \boldsymbol{Q}_t^{24} & \boldsymbol{Q}_t^{25} & \boldsymbol{Q}_t^{26} \end{bmatrix} .$$

Die Matrizen $\boldsymbol{T}_t^{11}$ bis $\boldsymbol{T}_t^{26}$ und $\boldsymbol{Q}_t^{11}$ bis $\boldsymbol{Q}_t^{26}$ des angepaßten Modells sind ebenfalls in Anhang E nachzulesen.
Auch in diesem Fall ist die Matrix $\boldsymbol{R}_t$ eine Einheitsmatrix der Dimension 36×36. Der Vektor $\boldsymbol{c}_t$ ist konstant Null. Die Maximum-Likelihood-Schätzung der Parameter des Modells vom Übergang des Modells aus der Initialisierungsphase zum angepaßten Modell ist konvergent.
Mittels des Kalman-Filters sind für jede Meßstelle fünf Werte prognostiziert. Mit diesen Wert bekommt man einen Hinweis auf das zukünftige Verhalten der Bodenveränderung an den Meßstellen der Hirse-Gruppe. Die Vorhersagewerte sowie ihre Standardfehler sind in Tab. 24 aufgelistet.
Um die Vorhersagewerte für die ursprüglichen Zeitreihen (trendbehaftet) zu erhalten, muß der Trend auf die Vorhersagewerte addiert werden. Zur Veranschaulichung der Vorhersagewert ist der Verlauf der letzten 100 Beobachtungen der Zeitreihe plus den fünf Vorhersagewerten aufgetragen. Einen genaueren Einblick ermöglicht Abb. 57 die die letzten 20 Werte der Zeitreihe des ersten Meßpunktes angibt und die Vorhersagewerte ebenfalls mit aufgeführt.
Anhand der Abb. 56 und Abb. 57 ist die gute Anpassung durch die durch das Modell geschätzte Zeitreihe ersichtlich. Der Verlauf der Kurven deutet auf einen positiven Effekt durch die Hirse hin. Werden die Parzellen mit Hirsestengel bedeckt, so findet über die Zeit hinweg eine Kumulation des Bodens auf dieser Parzelle statt. Dieses Phänomen ist nicht nur an einer Meßstelle der Hirse-Gruppe zu erkennen, sondern an allen Meßstellen. Keine Bodenbedeckung, wie bei der Kontroll-Gruppe, führt dagegen zu einem raschen Anstieg des Bodenverlusts. Der Verlust ist in den Abb. 58 und Abb. 59 deutlich zu erkennen.
Bei einer Darstellung aller Beobachtungen sind aufgrund der beschränkten Größe der Grafik nur zwei übereinanderliegende Kurvenverläufe zu erkennen. Aus diesem Grund wird auf eine Darstellung an dieser Stelle verzichtet.

	1		2		3		4		5	
Meß-stelle	Vor-her-sage-wert	Std.-fehler	Vor-her-sage-wert	Std.-fehler	Vor-her-sage-wert	Std.-fehler	Vor-her-sage-wert	Std.-fehler	Vor-her-sage-wert	Std.-fehler
1	-1,109	0,996	-0,916	1,249	-0,766	1,378	-0,643	1,453	-0,540	1,499
2	-2,204	1,004	-1,543	1,241	-1,073	1,357	-0,740	1,422	-0,505	1,462
3	0,088	1,025	-0,046	1,262	-0,073	1,376	-0,053	1,440	-0,019	1,478
4	1,149	1,003	0,715	1,243	0,419	1,365	0,225	1,435	0,103	1,478
5	0,052	1,021	-0,095	1,277	-0,175	1,403	-0,209	1,473	-0,212	1,513
6	0,918	0,984	0,889	1,233	0,828	1,363	0,749	1,441	0,664	1,491
7	0,171	1,013	0,096	1,276	0,055	1,413	0,033	1,494	0,020	1,545
8	0,239	1,021	0,011	1,301	-0,146	1,454	-0,246	1,547	-0,302	1,608
9	-0,849	1,006	-0,816	1,284	-0,741	1,433	-0,647	1,520	-0,546	1,574
10	0,221	1,019	-0,125	1,286	-0,276	1,426	-0,315	1,509	-0,297	1,560
11	-0,577	1,019	-0,494	1,293	-0,415	1,440	-0,344	1,528	-0,282	1,583
12	-1,524	1,057	-1,059	1,321	-0,746	1,453	-0,530	1,527	-0,378	1,570
13	0,111	0,995	0,094	1,254	0,091	1,391	0,094	1,474	0,099	1,528
14	-0,757	1,047	-0,611	1,312	-0,486	1,449	-0,380	1,528	-0,293	1,578
15	1,506	0,955	1,234	1,203	0,984	1,333	0,771	1,410	0,599	1,458
16	-0,268	0,986	-0,249	1,250	-0,217	1,390	-0,179	1,473	-0,140	1,525
17	-0,759	0,969	-0,573	1,217	-0,445	1,345	-0,357	1,418	-0,295	1,462
18	3,523	1,095	2,686	1,388	2,049	1,545	1,566	1,637	1,202	1,695
19	-2,436	1,035	-2,207	1,313	-1,959	1,462	-1,708	1,553	-1,467	1,611
20	-1,610	0,984	-1,271	1,239	-1,022	1,373	-0,829	1,454	-0,674	1,505
21	-1,472	0,982	-1,045	1,236	-0,754	1,368	-0,551	1,445	-0,406	1,493
22	-1,437	1,034	-1,048	1,302	-0,758	1,442	-0,542	1,524	-0,382	1,575
23	0,630	1,005	0,553	1,276	0,457	1,422	0,362	1,510	0,277	1,566
24	0,156	1,041	0,226	1,320	0,263	1,470	0,275	1,561	0,268	1,618
25	1,990	1,011	1,689	1,253	1,400	1,373	1,136	1,442	0,905	1,485
26	-0,007	1,028	0,107	1,287	0,175	1,416	0,210	1,487	0,223	1,529
27	-0,066	1,023	-0,052	1,271	-0,021	1,394	0,011	1,465	0,035	1,508
28	0,621	1,021	0,443	1,304	0,302	1,459	0,193	1,554	0,112	1,615
29	-1,720	1,007	-1,160	1,266	-0,745	1,403	-0,447	1,485	-0,237	1,536
30	-1,420	1,016	-0,872	1,281	-0,545	1,419	-0,353	1,500	-0,243	1,549
31	-1,079	0,994	-0,586	1,253	-0,261	1,383	-0,052	1,455	0,075	1,497
32	0,181	0,994	0,056	1,261	-0,026	1,407	-0,079	1,497	-0,113	1,556
33	1,560	0,998	1,141	1,245	0,851	1,374	0,639	1,452	0,479	1,502
34	0,853	1,045	0,429	1,307	0,156	1,442	-0,018	1,522	-0,126	1,572
35	0,456	1,007	0,237	1,267	0,127	1,405	0,074	1,486	0,051	1,537
36	-1,482	1,024	-1,255	1,279	-1,058	1,413	-0,889	1,494	-0,744	1,547

Tab. 24: Fünf Prognosewerte und deren Standardfehler für die 36 Meßstellen

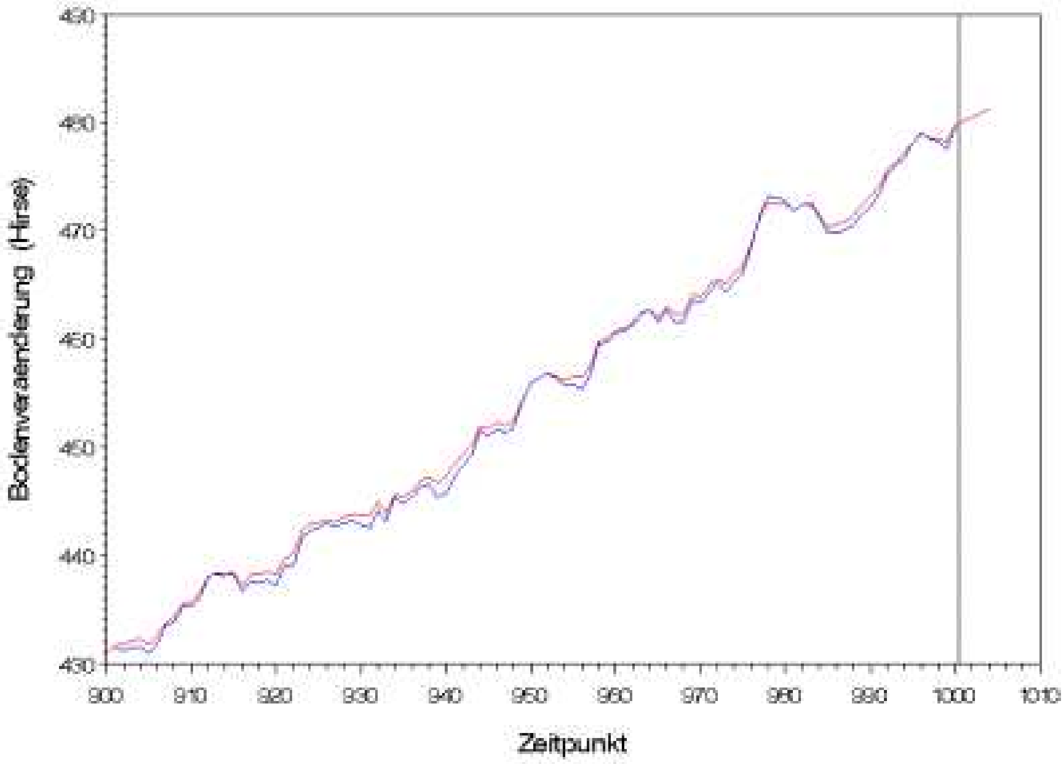


Abb. 56: Zeitreihe (blaue Kurve) und Schätzung durch das Zustandsraummodell inklusive Vorhersagewerte (rote Kurve) für die letzten **100** Beobachtungen der Hirse-Gruppe. Die senkrechte Linie markiert den Übergang zwischen der letzten Beobachtung und dem ersten Prognosewert.

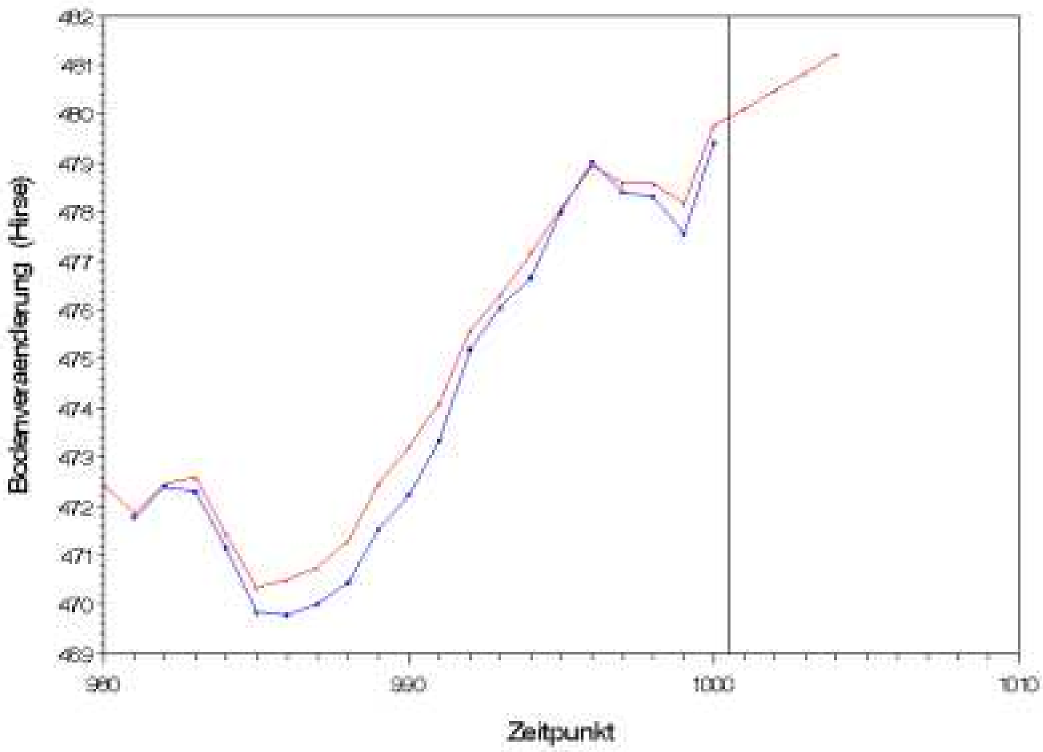


Abb. 57: Zeitreihe (blaue Kurve) und Schätzung durch das Zustandsraummodell inklusive Vorhersagewerte (rote Kurve) für die letzten **20** Beobachtungen der Hirse-Gruppe. Die senkrechte Linie markiert den Übergang zwischen der letzten Beobachtung und dem ersten Prognosewert.

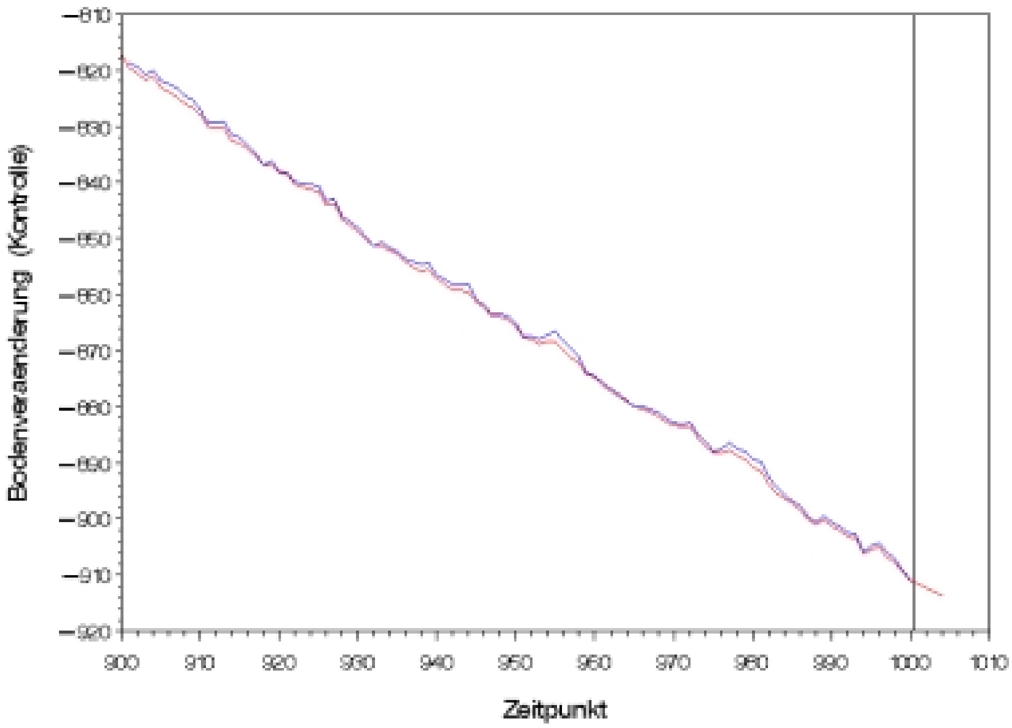


Abb. 58: Zeitreihe (blaue Kurve) und Schätzung durch das Zustandsraummodell inklusive Vorhersagewerte (rote Kurve) für die letzten **100** Beobachtungen der Kontroll-Gruppe. Die senkrechte Linie markiert den Übergang zwischen der letzten Beobachtung und dem ersten Prognosewert.

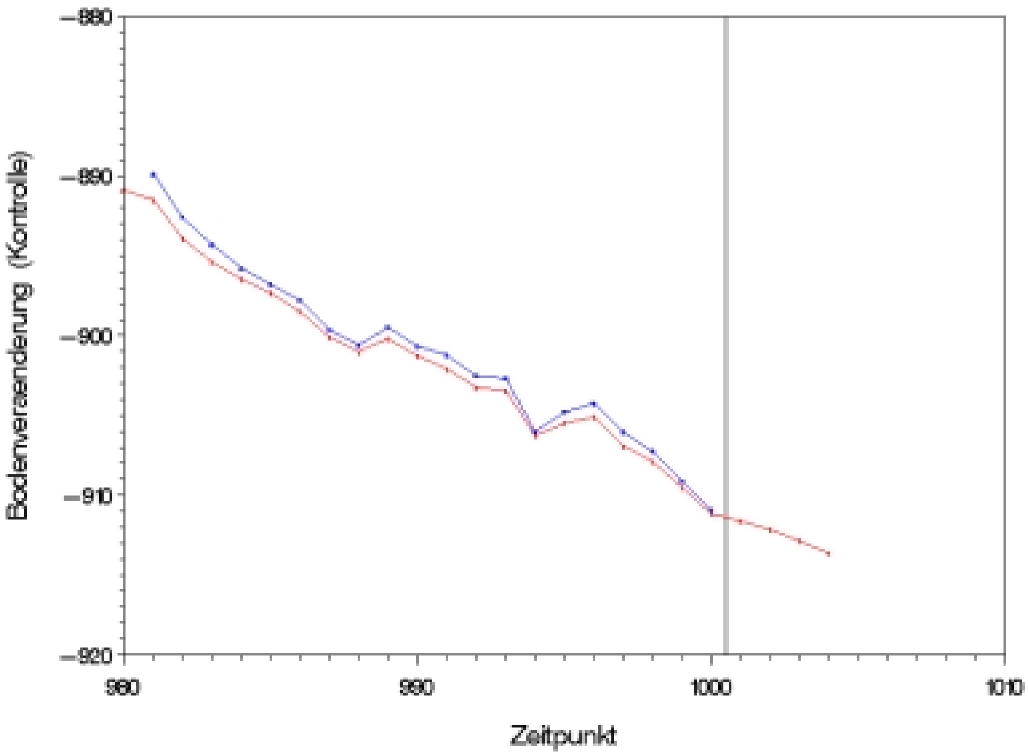


Abb. 59: Zeitreihe (blaue Kurve) und Schätzung durch das Zustandsraummodell inklusive Vorhersagewerte (rote Kurve) für die letzten **20** Beobachtungen der Kontroll-Gruppe. Die senkrechte Linie markiert den Übergang zwischen der letzten Beobachtung und dem ersten Prognosewert.

6 Zusammenfassung und Ausblick

Die in der Arbeit von Zerbst (2001) erwähnten Erweiterungen und Ziele sind alle erarbeitet worden. Insbesondere die Weiterentwicklung des Maximum-Linkage-Ansatzes hin zur Einbeziehung der Verteilung der Farbwerte im Ausgangsbild einer Clusterung sind im neuen erweiterten Maximum-Linkage-Verfahren umgesetzt. Der Algorithmus liefert die erwarteten Ergebnisse. Auch die Einführung des neuen Beurteilungskriteriums für eine Clusterung erleichtert die korrekte Auswahl der „richtigen“ Clusterung. Die Auswahl einer Clusterung aufgrund der mittleren Minimaldistanz führte noch zu einer unzutreffenden Clusteranzahl. Mit dem neu entwickelten Kriterium der gewichteten mittleren Minimaldistanz wird das vermieden. Die Einbeziehung der Gewichtung und der Konzentration der Werte hat zu dem gewünschten Ergebnis geführt. Eine Auswahl auf der Basis des $\boldsymbol{GMMDK}$ liefert die „beste“ Clusterung im vorliegenden Problemkreis der Erosionserkennung und allgemein für Clusterungen mit hoher Trennschärfe.

Die ebenfalls durchgeführte Analyse der Bodendaten liefert nicht immer das gewünschte Ergebnis. Die Datenanzahl war jedoch für eine fundierte Analyse nicht ausreichend. Dennoch ist es mit den auf der Analyse aufbauenden Simulationen gelungen, einen Einblick in die Datenlage zu bekommen. Ein Erosionsprozeß ist sicherlich von vielen Faktoren abhänging. Aber die Beeinflussung des zukünftigen Bodenverlusts ist neben dem Einfluß von Störgrößen eben auch von der vergangenen, respektive gegenwärtigen Bodensituation abhängig. Insofern erscheint eine Modellierung mit einem ARMA-Prozeß kleiner Ordnung (Gesetz der Sparsamkeit) sinnvoll. Ein ARMA(2,2)-Modell ist hier sicherlich ein guter Startpunkt für zukünftige Analysen. Zum einen erscheint dem Autor der AR(2)-Anteil hinreichend, die Einflüsse des zukünftigen Bodenverlusts zu beschreiben. Der MA(2)-Anteil hingegen beschreibt die zufälligen Störungen wie Wetterveränderung und mögliche Veränderung der Umgebungssituation ausreichend gut. In weiterführenden Arbeiten sollten die genaueren Gegebenheiten des Untersuchungsgebietes mit einfließen. Die Zusammensetzung des Bodens, der Bewuchs mit Pflanzen in Hinblick auf die Tiefe etwaiger Wurzeln und deren Rückhalteeffekt oder aber Ablegen von Steinen am Feldrand können berücksichtigt werden. Mit diesen zusätzlichen Informationen ist es dann denkbar, ein strukturelles Zeitreihenmodell aufzubauen und die Erkenntnisse, die durch die Clusterungen des Luftbildes erreicht werden, zusammen mit den Meßdaten und Zusatzdaten über die Beschaffenheit des Gebietes zu modellieren. Bezugnehemend auf den strukturellen Zeitreihenansatz in Kapitel 3 ließe sich ein Modell mit Trend- und Saisonkomponente gemäß (3.42) beschreiben durch

$$y_t = \mu_t + \gamma_t + \varepsilon_t\,, \quad t = 1,\ldots,T^* \tag{6.1}$$

mit μ_t wie in (3.36) und (3.37), γ_t wie in (3.41), ε_t ist weißes Rauschen. In der allgemeinen Trendkomponente μ_t sowie in β_t sind die Informationen aus den Beobachtungen im Feld, also Bodentyp, Bodenbedeckung und Bewuchs enthalten. Die saisonale Veränderung ergibt sich aus den Veränderungen der Cluster die durch die Clusterung der Luftaufnahmen mit ins Modell einbezogen werden. Diese Veränderung kann über γ_t modelliert werden.

Übertragen in die Zustandsform läßt sich das Modell für eine Saison auf Quartalsbasis wie folgt angeben:

$$y_t = (1 \quad 0 \quad 1 \quad 0 \quad 0)\alpha_t + \varepsilon_t, \tag{6.2}$$

mit

$$\alpha_t = \begin{bmatrix} \mu_t \\ \\ \beta_t \\ \cdots \\ \gamma_t \\ \gamma_{t-1} \\ \gamma_{t-2} \end{bmatrix} = \begin{bmatrix} 1 & & 1 & \vdots & & & \\ & & & & & 0 & \\ 0 & & 1 & \vdots & & & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & \vdots & -1 & -1 & -1 \\ & 0 & & & 1 & 0 & 0 \\ & & & \vdots & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mu_{t-1} \\ \\ \beta_{t-1} \\ \cdots \\ \gamma_{t-1} \\ \gamma_{t-2} \\ \gamma_{t-3} \end{bmatrix} + \begin{bmatrix} \eta_t \\ \\ \zeta_t \\ \cdots \\ \omega_t \\ 0 \\ 0 \end{bmatrix}. \tag{6.3}$$

Doch wie sollen die Erkenntnisse durch eine Clusterung genutzt werden? Wie sollen die beiden Datentypen „Bodendaten" und „Bilddaten" zusammengeführt werden?

Ein möglicher Ansatz ließe sich wie folgt gestalten. Zum einen sollte ein Untersuchungsgebiet gewählt werden, welches hinreichend viel erodierte Fläche liefern kann. Desweiteren muß eine Studie von vornherein so geplant sein, daß sie langfristig – 10 Jahre und mehr – Daten sammeln kann. Aufgrund von seltenen Regen- und Windereignissen im westlichen Afrika erscheint die Zeitspanne als untere Grenze. Auch muß die Zuverlässigkeit der Meßpunkte gewährleistet sein. Ein gelegentlicher Ausfall einer Meßstation ist zu verantworten. Der Ausfall ganzer Meßreihen an einem oder mehreren Meßstellen ist nicht vertretbar. Hat man diese Schwierigkeiten überwunden, so gilt das nächste Augenmerk auf die Erstellung der Bilder zu legen. Mir war es leider nicht vergönnt, trotz intensiver internationaler Kontakte, ausreichend Bildmaterial von dem Untersuchungsgebiet zu erhalten. Das Ausweichen auf Satellitenbilder ist jedoch keine echte Alternative. Aufnahmen von den gängigen Satellitentypen wie Landsat-7 oder Resurs-01 sind zum einen äußerst kostenträchtig und zum anderen mit dem Makel behaftet, daß sie aufgrund ihrer Umlaufbahn und Position nie ganz exakt die gleichen Bereiche abtasten. Zudem können Wolkenfelder die Aufnahme empfindlich stören. Somit ist die Gewinnung der gewünschten Bilder durch Flugdrachen oder Fesselballons in einer geringen Höhe (etwa 200-300m) kostengünstiger und weniger anfällig. Insbesondere da die Technik im Rahmen des SFB 308 „Adapted Farming in West Afrika“ (Bürkert et al., 1996) schon entwickelt und erprobt wurde. Mit dieser Technik sollten Aufnahmen des Gebietes in einem bestimmten zeitlichen Abstand getätigt werden. Denkbar sind etwa Aufnahmen nach jedem Regenereignis oder einem Sturm, durch den ebenfalls eine Veränderung der Erdoberfläche einher geht. Anschließend werden aus den Bildern Veränderungsraten bestimmt, die als Information mit ins (strukturelle Zeitreihen-) Modell (6.1) aufgenommen werden. Die Änderungsraten dieser Aufnahmen werden dabei folgendermaßen bestimmt. Zunächst ist sicherzustellen, daß stets der gleiche Winkel und die gleiche Höhe für die Luftaufnahme gilt. Jedes Bild muß einzelnen für sich vorverarbeitet werden. Diese Bearbeitung geschieht dergestalt, daß jedes Bild geclustert wird. Die Clusterung sollte dabei mit einem Verfahren stattfinden, das möglichst gut zwischen den Clustern trennen kann, also große Abstände zwischen den Clustern bildet. Zur Clusterung empfehlen sich somit die vorgestellten Linkage-Verfahren. Neben der guten Clustertrennung spricht zudem die extrem kurze Laufzeit für diese Methoden. Durch die gute Trennung wird eine gewisse „Glättung“ erreicht, die wesentlich dazu beiträgt, die gewünschte Information aus dem Bild zu extrahieren. Ist die Clusterung durchgeführt, wird ein Raster über die Bilder gelegt. Denkbar ist zum Beispiel ein Raster, was den Parzellen des Untersuchungsgebietes entspricht. In jeder dieser Parzellen wird nun beim Vergleich der zusammengehörigen Parzellen eine Änderungsrate bestimmt. Dazu wird eine Kenngröße der Parzelle bestimmt. Eine Möglichkeit ist ein gewichtetes Zentroidmittel, also etwa ein Flächenanteil des Clusters an der Parzelle. Die Differenzen dieser so berechneten Kenngrößen ergeben einen Vektor der Änderungsraten, die ins Modell eingestellt werden. Ist das Modell mit diesen zusätzlichen Informationen aufgestellt und wurden zudem die Daten wie beschrieben bestimmt, kann auch das Ziel erreicht werden, was der Autor ursprünglich als Endziel ausgegeben hat: Anhand eines Modells für ein Untersuchungsgebiet in der Lage zu sein, die Erosion prognostizieren zu können. Der Ablauf dieser Prozedur läßt sich anhand der nachstehenden Abb. 60 verdeutlichen.

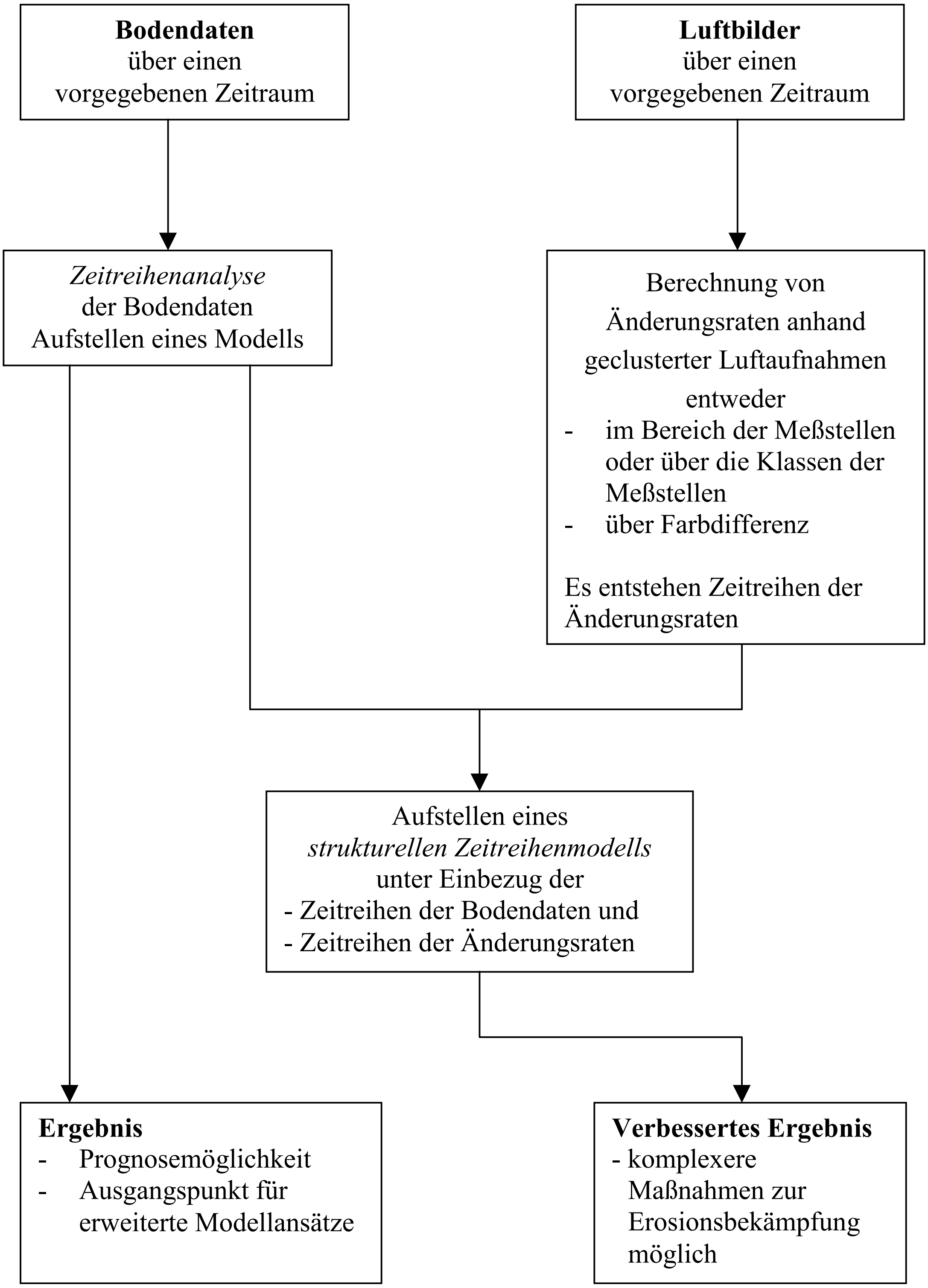


Abb. 60: Schematische Darstellung des Ablaufs von Datenbereitstellung bis zum Ergebnis, einem strukturellen Zeitreihenmodell zur Verbesserung der Prognose über Bodenveränderungen in Erosionsgebieten.

Ist das verbesserte Ergebnis verfügbar, können die Erkenntnisse benutzt werden, um neue Daten zu gewinnen. Durch die Ergebnisse ist es dann möglich die Erhebung der Bodendaten zu beeinflussen und somit die Qualität zu verbessern. So setzt ein Prozeß des fortwährenden Lernens ein, welcher über Jahre hinweg zu einer substantiellen Verbesserung hinsichtlich der Erosionsausbreitung führen kann. In einer weiteren Abstraktionsstufe sollte das Modell dann soweit wie möglich verallgemeinert werden, um für möglichst viele Gebiete, bei Vorhandensein der entsprechenden Daten, Prognosen über die zukünftige Entwicklung der Erosion zu ermöglichen.

Anhang A: Herleitung der Backpropagationregel

Lineare Netze können nur lineare Funktionen berechnen. Für die mächtigeren mehrstufigen Netze sind Trainigsverfahren notwendig, die nichtlineare Netze anlernen wie die sogenannte Backpropagationregel. Nach Rumelhart (1986) ist diese Regel eine Verallgemeinerung der Delta-Regel für Netze mit mehreren Neuronenebenen.

Die Netzeingabe sei gegeben durch:

$$net_{pj} = \sum_i o_{pi} w_{ij} \,, \qquad (0.1)$$

wobei w_{ij} das Gewicht der Verbindung zwischen Neuron i und j , o_i Ausgabe des Neurons i und p ein gegebenes Eingabemuster ist.

Analog zur Delta-Regel gilt:

$$\Delta w_{ij} = \sum_p -\eta \frac{\partial E_p}{\partial w_{ij}} \,, \qquad (0.2)$$

mit $E_p = \frac{1}{2}\sum_j (t_{pj} - o_{pj})^2$ als Fehlerfunktion für ein Muster p und t_{pj} die Lerneingabe, o_{pj} die Ausgabe von Neuron i bei einem Muster p und η die Lernrate. Der Faktor ½ ist aus technischen Gründen gewählt worden. Dieser Fehler ist bei Gradientenabstiegsverfahren wie dem Backpropagationsverfahren zu minimieren. Dabei ist es unerheblich, ob ein Fehler oder ein halber Fehler minimiert wird.
Mit der Kettenregel folgt:

$$\frac{\partial E_p}{\partial w_{ij}} = \frac{\partial E_p}{\partial net_{pj}} \frac{\partial net_{pj}}{\partial w_{ij}} \;. \qquad (0.3)$$

Aus Gleichung (0.1) ergibt sich für den zweiten Faktor aus (0.3):

$$\frac{\partial net_{pj}}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum_i o_{pi} w_{ij} = o_{pi} \,. \qquad (0.4)$$

Setzt man nun

$$\delta_{pj} = -\frac{\partial E_p}{\partial net_{pj}}, \tag{0.5}$$

als das sogenannte Fehlersignal, siehe Rumelhart (1986), so ergibt sich nach dem Einsetzen der Gleichungen (0.4) und (0.5) in Gleichung (0.3) für die Gleichung (0.2):

$$\Delta w_{ij} = \eta \sum_p o_{pi} \delta_{pj} \, . \tag{0.6}$$

Dies entspricht der Standard-Delta-Regel, nur daß die Fehlersignale definiert werden als:

$$\delta_{pj} = -\frac{\partial E_p}{\partial net_{pj}} = -\frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial net_{pj}}. \tag{0.7}$$

Abschließend betrachte man den ersten Faktor aus (0.7). Hierfür gibt es zwei Fälle. Der erste Fall tritt ein, wenn j Index aus Ausgabeneuronen ist, der zweite Fall wenn j Index eines Neurons aus einer verdeckten Schicht ist.
Im ersten Fall ist

$$-\frac{\partial E_p}{\partial o_{pj}} = \left(t_{pj} - o_{pj}\right). \tag{0.8}$$

Im zweiten Fall kann die partielle Ableitung aus (0.8) nur indirekt mit Hilfe der Kettenregel und der Gleichungen (0.1) und (0.5) berechnet werden. Es ergibt sich:

$$-\frac{\partial E_p}{\partial p_{pj}} = -\sum_k \frac{\partial E_p}{\partial net_{pk}} \frac{\partial net_{pk}}{\partial o_{pj}} = \sum_k \left(\delta_{pk} \frac{\partial}{\partial o_{pj}} \sum_i o_{pi} w_{ik} \right) = \sum_k \delta_{pk} w_{jk} \tag{0.9}$$

Der Gesamtfehler eines Neurons j mit Muster p berechnet sich aus dem gewichteten Fehler δ_{pk} aller Nachfolgerneuronen k und der Gewichte von j zu diesen k.
Zusammengefaßt erhält man die Backpropagationregel:

$$\Delta_p w_{ij} = \eta o_{pi} \delta_{pj}. \tag{0.10}$$

Anhang B: Grundkonzepte der Farbdarstellung

Farbmodelle

Um Farbbilder in einem Rechner zu verarbeiten und darzustellen, bedarf es der technischen Umsetzung des Begriffes Farbe. Physikalisch betrachtet wird Farbe durch Licht erzeugt. Licht ist eine elektromagnetische Welle und kann durch eine Frequenz oder die Wellenlänge beschrieben werden. Für den Menschen ist nur ein kleiner Ausschnitt aus dem elektromagnetischen Spektrum sichtbar. Dieser Ausschnitt liegt zwischen 400 nm und 700 nm. Trifft Licht mit einer bestimmten Wellenlänge auf das Auge, so wird eine Farbempfindung hervorgerufen, die durch die Farben des Regenbogenspektrums, von Violett (400 nm) bis Rot (700 nm), beschrieben werden kann (Haberäcker, 1995). Das Auge spricht vor allem auf die Farben Rot, Grün und Blau an. Wobei die intensivste Reaktion mit Grün hervorgerufen wird (Haberäcker, 1995). Aufgrund dieser Tatsache geht man davon aus, daß sich Farben aus einer Mischung der Farbsignale dieser drei Farben (Rot, Grün, Blau) zusammensetzen lassen. Man spricht von einem „Drei- Farbenmodell". Einziger Nachteil: Es gibt keine drei Grundfarben, also Ausgangsfarben für eine Farbmischung aus denen die anderen Farben subtraktiv ermittelt werden können, aus denen sich alle Farben zusammensetzen ließen. Um eine gewisse Normierung für die Farbwerte zu schaffen, hat die Commision Internationale de L'Eclairage (CIE) 1931 drei künstliche Grundfarben eingeführt. Diese Primärfarben werden mit X,Y und Z bezeichnet und sind durch eine Energieverteilungskurve charakterisiert. Neben der Einführung der Primärfarben entwickelte die CIE eine Farbtabelle dergestalt, daß jeder Punkt des in Abb. 61 dargestellten Diagramms eine Farbe repräsentiert. Ferner sind alle Farben auf der Strecke zwischen zwei Farbpunkten durch Mischen der Farben der Endpunkte herzustellen und alle Farben innerhalb des Dreiecks können durch Mischen der Farben der Eckpunkte erhalten werden.

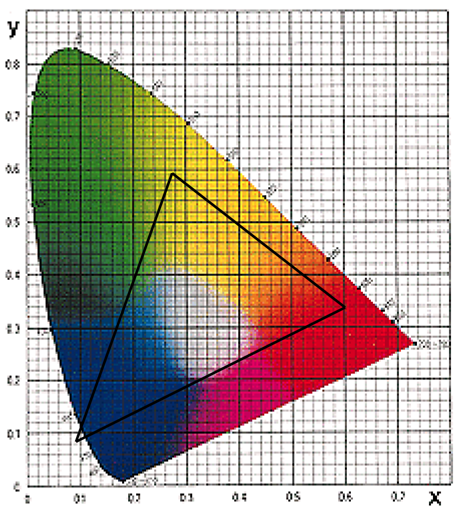


Abb. 61: Normfarbtafel der Commission International de L'Eclairage (CIE), 1931.

In der Mitte des Dreiecks erscheinen die Farben für das menschliche Auge weiß. Eine beliebige Farbe kann als Koordinatenpunkt im Dreieck als Anteile der drei Primärfarben X,Y und Z beschrieben werden. Die Position im CIE-Dreieck läßt sich darstellen als (vgl. Haberäcker, 1995):

$$x = \frac{X}{X+Y+Z} \;,\; y = \frac{Y}{X+Y+Z} \;,\; z = \frac{Z}{X+Y+Z} \;.$$

Dabei ist wegen $z = 1 - x - y$ auf die Angabe von z zu verzichten, so daß eine zweidimensionales Diagramm ausreicht. Wird eine Farbe durch die (x,y)-Koordinaten beschrieben, so muß diese noch um die Helligkeit ergänzt werden. Dies ist aber gerade der Y-Wert. Der X-Wert und der Z-Wert berechnen sich dann gemäß

$$X = x\frac{Y}{y} \quad , \qquad Z = z\frac{Y}{y} \,.$$

Die CIE hat für Rot, Grün und Blau einen Standard eingeführt, der Tab. 25 zu entnehmen ist.

CIE-Farbe	**Wellenlänge nm**	**x**	**y**	**z**
Rot	700,0	0,73467	0,26533	0,0
Grün	546,1	0,27367	0,71741	0,00892
Blau	435,8	0,16658	0,00886	0,82456

Tab. 25: CIE-Festlegung der Standardwerte für Rot, Grün und Blau in x,y und z-Anteil. (vgl. Haberäcker, 1995; Seite 51).

Auf Farbmonitore werden die Farben durch rot, grün und blau leuchtende Phosphore erzeugt, denen Punkte im CIE-Dreieck entsprechen. Für moderne Farbmonitore werden folgende Werte für die Grundfarben angegeben:

Grundfarbe	**x**	**y**	**z**
Rot	0,628	0,346	0,026
Grün	0,268	0,588	0,144
Blau	0,150	0,070	0,780

Tab. 26: Monitorfarbeinstellung für Rot, Grün und Blau.

Aufbauend auf dieser Grundüberlegung und –definition sind verschiedene Farbmodelle entwickelt worden. Das bekannteste und das am häufigsten verwendete Verfahren, was auch die eingangs des Kapitels beschrieben Probleme löst, ist das RGB-Modell. Dieses wird im folgenden Abschnitt beschrieben.

Das RGB-Modell

Aufgrund der Überlegung aus obigen Abschnitt ist eine Farbe somit als Linearkombination der drei Primärfarben zu bestimmen. Daher wird der RGB-Farbraum als dreidimensionales, kartesisches Koordinatensystem dargestellt. Die Farbwerte werden so normiert, daß die Rot-, Grün- und Blau-Komponenten zwischen 0 und 1 liegen.

Das so beschriebene Modell muß jedoch aus der Mischung aller Farben die Farbe Weiß erzeugen. Dieses läßt sich aufgrund der CIE-Definition nicht erreichen. Daher wird ein Punkt im CIE-Dreieck als Weiß definiert. In der Regel wird dazu das Weiß eines auf 6500 Kelvin erhitzten schwarzen Körpers verwendet. Dieser hat die x,y und z – Koordinaten gemäß: $x = 0{,}313$, $y = 0{,}329$ und $z = 0{,}358$.

Den Monitorgrundfarben entsprechen die CIE-Normfarbanteile (X_i, Y_i, Z_i) mit $i = r,g,b$. Diese, zu gleichen Anteilen gemischt, sollen Weiß ergeben. Es ist somit das Gleichungssystem

$$\begin{aligned} X_r + X_g + X_b &= X_w \\ Y_r + Y_g + Y_b &= Y_w \\ Z_r + Z_g + Z_b &= Z_w \end{aligned}$$

zu lösen. Als Lösung für die Monitorwerte ergibt sich (vgl. Haberäcker, 1995) $r = 0{,}761$, $g = 1{,}114$ und $b = 1{,}164$.

Die Umrechnung zwischen den X, Y und Z-Werten und den R, G und B-Werten ergibt sich aus der Matrizenmultiplikation

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.478 & 0.299 & 0.175 \\ 0.263 & 0.655 & 0.081 \\ 0.020 & 0.160 & 0.908 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

bzw.

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 2.741 & -1.147 & -0.426 \\ -1.118 & 2.028 & 0.034 \\ 0.137 & -0.332 & 1.105 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}.$$

Diese Umrechnung gilt jedoch nur, wenn die Farbskala der Monitorfarben gemäß Tab. 26 vorausgesetzt wird und wenn der angegebene Weißpunkt verwendet wird.

Für ein rechnerbasiertes RGB-Modell muß das kontinuierliche Modell in ein diskretes überführt werden. Der Parameterraum liegt nun für jede Komponente zwischen $\{0,1,\ldots,255\}$. Daraus ergibt sich, daß ein rechnerbasiertes RGB-Modell in einer 24-Bit-Darstellung gespeichert ist, welches eine maximale Farbanzahl von $16{,}7$ Millionen Farben impliziert. Das RGB-Farbsystem ist ein sogenanntes additives Farbmodell, weil es die Farbempfindung durch die Überlagerung des von den einzelnen Phosphorpunkten ausgestrahlten Lichtes bestimmt.
Während sich das RGB-Modell gut für Darstellungen am Monitor eignet, lassen sich Farbbilder mittels dieses Farbmodells nicht ausdrucken. Bei Druckern wird das subtraktive Modell CMY benutzt. In ihm sind die Grundfarben nicht Rot, Grün und Blau, sondern Zyan (Cyan), Magenta (Magenta) und Gelb (Yellow). Der Farbeindruck entsteht dadurch, daß die Farbe mit unterschiedlicher Intensität auf das Papier gebracht wird. Diese Farben absorbieren das weiße Licht. Die Umrechnung zwischen den Farbmodellen geschieht mittels

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} C \\ M \\ Y \end{pmatrix}.$$

Wie das RGB-Modell beschreibt das CMY-Modell einen dreidimensionalen kartesischen Raum. Während beim RGB-Modell im Koordinatenursprung die Farbe Schwarz liegt, ist beim CMY-Modell dort die Farbe Weiß.

Anhang C: SAS®-Programm zur Berechnung des GMMD

```
/ ****************************************************
  *                                                  *
  * SAS-PROGRAMM:    Berechnung des GMMDC            *
  *                                                  *
  * Autor:           Lars Tschiersch                 *
  * Erstellt:        19.04.2002                      *
  * Zuletzt gändert: 17.05.2002                      *
  *                                                  *
  * Bemerkung:       STV-Datei aus PICANA wird       *
  *                  eingeladen                      *
  ****************************************************/


/* Einlesen des STV-Datei aus PICANA. Es interessiert
   nur die Variable min_dis
*/


data eingang (keep=min_dis a k);
 infile 'e:\niger1.stv' firstobs=2;
 input r g b min_dis;
  k=_n_;
  a=0;
run;


*Berechnen des Maximums;
proc means data=eingang max noprint;
 var k;
 output out = maxobs max=n;
run;



data maxobs2 (keep=n);
 set maxobs;
  do k=1 to _freq_;
   output;
  end;
run;


proc sort data=eingang;
 by min_dis;
run;


/* Berechnen der Gewichte */
data eingang2;
 merge maxobs2 eingang;
  gew_1=round(((1-a)/(((n+1)/2)-1))*_n_+a-((1-a)/(((n+1)/2)-
1)),0.01);
  gew_2=round(((a-1)/(n-((n+1)/2)))*_n_+1-((n+1)/2)*((a-1)/(n-
((n+1)/2))),0.01);
  b=_n_;
run;
```

```
* Summe über Gewicht1 (aufsteigender Arm);
proc means data=eingang2 sum noprint;
 var gew_1;
 output out=sum_g_1 sum=sum_gew1;
 where b <=(n/2);
run;


* Summe über Gewicht2 (abfallender Arm);
proc means data=eingang2 sum noprint;
 var gew_2;
 output out=sum_g_2 sum=sum_gew2;
 where b >(n/2);
run;


data summe (keep=sum_gew1 sum_gew2);
 merge sum_g_1 sum_g_2 maxobs;
 do i=1 to n;
  output;
 end;
run;


* Bestimmen: Maximum von Min_dis;
proc means data=eingang2 max noprint;
 var min_dis;
 output out=maximum max=max_dis;
run;


data maximum2;
 set maximum;
 do i=1 to _freq_;
  output;
 end;
run;


*Berechnen des GMMD;
data rechnung_1;
 merge eingang2 maximum2 summe;
  norm_teil_int = (lag(min_dis) + ((min_dis -
lag(min_dis))/2))/max_dis;
  teil_int = (lag(min_dis) + ((min_dis - lag(min_dis))/2));
  if _n_ = 1 then teil_int = 0;

*Berechnung des gewichtetetn Mittels der Minimaldistanzen;
 if _n_ <= n/2 then
   t_wamd = (gew_1/(sum_gew1 + sum_gew2)) * min_dis;
 if _n_ > n/2 then
   t_wamd = (gew_2/(sum_gew1 + sum_gew2)) * min_dis;

 lag_min_dis = lag(min_dis);
 if _n_ = 1 then lag_min_dis = 0;
run;

proc means data=rechnung_1 sum noprint;
 var teil_int norm_teil_int min_dis t_wamd;
 output out=rechnung_2 sum=int norm_int sum_dis wamd;
run;
```

```
/* Berechnung der x-Koordinaten der Lorenzkurbe über die
   Kozentration der Minimaldistanzen. */

data rechnung_3;
 set rechnung_2;
 do i=1 to _freq_;
  lor_x = i*1/_freq_;
  sum_k = lor_x + lag(lor_x);
  if sum_k = . then sum_k = lor_x;
  output;
 end;
run;

data rechnung_4;
 merge rechnung_1 rechnung_3;
 drop _type_ _freq_;
run;

/* Berechnung der y-Koordinaten der Lorenzkurbe über die
   Kozentration der Minimaldistanzen. */

data rechnung_5;
 set rechnung_4;
  lor_y_h = min_dis / sum_dis;
  teil_g = (sum_k/2) * (lor_y_h);
   *Weil lor_y_h = lor_y - lag(lor_Y);
run;

proc means data=rechnung_5 sum noprint;
 var teil_g;
 output out=Gini_vor sum=g1 teil_g;
run;

*Berechnung Gini-Koeffizient;
data gini (drop=g1);
 set gini_vor;
  g = g1-0.5;
run;

*Berechnung des GMMDC;
data gmmdc;
 merge gini rechnung_2;
  gmmdc_1_2 = gmmd / (g**(1/2));
  gmmdc_1_3 = gmmd / (g**(1/3));
run;

proc print data=gmmdc;
 var mmd gmmdc_1_2 g;
run;
quit;


proc gplot data=eingang2;
 symbol1 i=join c=blue;
 plot min_dis*i=1;
run;
quit;
```

Anhang D: SAS®-Programm zur Zeitreihenanalyse

```
/ ****************************************************
  *                                                  *
  * SAS-PROGRAMM:    Einlesen der Excel-Datei        *
  *                  'Bürkert_import.xls'            *
  *                  Aufteilen in drei einzelne Daten-   *
  *                  sätze je einen  für HIRSE, PLASTIK, *
  *                  KONTROLLE                       *
  *                                                  *
  * Autor:           Lars Tschiersch                 *
  * Erstellt am:     19.03.2002                      *
  *                                                  *
  * Beschreibung:                                    *
  * -------------                                    *
  * Einlesen mit proc import, da proc Access nur bis     *
  * Excel 5 einlesen kann. Anschließden werden alle      *
  * Beobachtungen größer 33 gelöscht.                *
  * Aufteilung in die Gruppen (s.o.). Dazu Erzeugung der*
  * Variablen X1-X36 welche die Bodenabtragung des       *
  * Meßzeitpunktes zum ersten Zeitpunkt mißt. Die im     *
  * Excel-Datensatz enthaltenen Variablen H1-H36 und     *
  * M1-M36 werden gelöscht.                          *
  *                                                  *
  * Zuletzt geändert am:  21.03.2002                 *
  * Beinhaltet MACRO                                 *
  *                                           (1)    *
  ****************************************************/

PROC IMPORT OUT= WORK.DATEN
            DATAFILE= "E:\buerkert_import.xls"
            DBMS=EXCEL2000 REPLACE;
     GETNAMES=YES;
RUN;

data dat2;
set daten;
if _n_ > 33 then delete;
run;

%Macro teilen(gruppe, dat);
  data &dat (drop=gruppe m: h:);
   set dat2;
   where gruppe=&gruppe;
   %do i=1 %to 36;
      x&i=h&i-m&i;
   %end;
  run;
%Mend;

%teilen('Kontroll', kontrolle);
%teilen('Residues', hirse);
%teilen('Plastik', plastik);
```

```
/ ***************************************************
  *                                                 *
  * SAS-PROGRAMM:         Erzeugung der Datenspalte TAGE *
  *                       für alle drei Versuchsfelder   *
  *                       HIRSE, PLASTIK und Kontrolle   *
  *                                                 *
  * Autor:                Lars Tschiersch           *
  * Erstellt am:          20.03.2002                *
  *                                                 *
  * Zuletzt geändert am:    20.03.2002              *
  *                                          (2)    *
  ***************************************************/


data hirse1;
 set hirse;
 if _N_ = 1 then  do;
   tage=0;
   output;
 end;

 if _N_ = 2 then do;
  tage=7;
   output;
 end;
 if _N_ = 3 then do;
  tage=41;output;
 end;
 if _N_ = 4 then do;
  tage=94;output;
 end;
 if _N_ = 5 then do;
   tage=242;output;
 end;

 if _N_ = 6 then do;
  tage=314;output;
 end;
 if _N_ = 7 then do;
    tage=362;output;
 end;
 if _N_ = 8 then do;
   tage=382;output;
 end;
 if _N_ = 9 then do;
   tage=413;output;
 end;
 if _N_ = 10 then do;
   tage=658;output;
 end;
 if _N_ = 11 then do;
   tage=688;output;
 end;
run;

data plastik1;
 set plastik;
 if _N_ = 1 then  do;
   tage=0;
   output;
 end;

 if _N_ = 2 then do;
```

```
    tage=7;output;
  end;
  if _N_ = 3 then do;
   tage=41;output;
  end;
  if _N_ = 4 then do;
    tage=94;output;
  end;
  if _N_ = 5 then do;
    tage=242;output;
  end;

  if _N_ = 6 then do;
   tage=314;output;
  end;
  if _N_ = 7 then do;
    tage=362;output;
  end;
  if _N_ = 8 then do;
    tage=382;output;
  end;
  if _N_ = 9 then do;
    tage=413;output;
  end;
  if _N_ = 10 then do;
   tage=658;output;
  end;
  if _N_ = 11 then do;
   tage=688;output;
  end;
run;

data kontrolle1;
 set kontrolle;

  if _N_ = 1 then  do;
    tage=0;
    output;
  end;

  if _N_ = 2 then do;
   tage=7;output;
  end;
  if _N_ = 3 then do;
   tage=41;output;
  end;
  if _N_ = 4 then do;
   tage=94;output;
  end;
  if _N_ = 5 then do;
    tage=242;output;
  end;

  if _N_ = 6 then do;
   tage=314;output;
  end;
  if _N_ = 7 then do;
     tage=362;output;
  end;
  if _N_ = 8 then do;
    tage=382;output;
  end;
```

```
 if _N_ = 9 then do;
   tage=413;output;
 end;
 if _N_ = 10 then do;
  tage=658;output;
 end;
 if _N_ = 11 then do;
   tage=688;output;
 end;
run;

proc print data=hirse1;title 'Hirse';run;

proc print data=kontrolle1;title 'Kontrolle';run;

proc print data=plastik1;title 'Plastik';run;

quit;
```

```
/ *****************************************************
  *                                                   *
  * SAS-PROGRAMM:      Regression über alle Gruppen von *
  *                    Versuchsfeldern (Hirse, Plastik, *
  *                    Kontrolle                      *
  * Autor:             Lars Tschiersch                *
  * Erstellt am:       21.03.2002                     *
  *                                                   *
  * Beschreibung:                                     *
  * -------------                                     *
  *  Durchlauf über alle 36 Meßpunkte und Berechnung der *
  * Parameterschätzer.                                *
  *                                                   *
  * Zuletzt geändert am:  21.03.2002                  *
  *                                        (3)        *
  *****************************************************/


%Macro rg(datensatz, tagevar);
 %do i=1 %to 36;
  proc reg data=&datensatz;
   model x&i=&tagevar;
  run;
 %end;
%Mend;

title 'Datensatz: HIRSE';
%rg(hirse1, tage);

title 'Datensatz: PLASTIK';
%rg(plastik1, tage);

title 'Datensatz: Kontrolle';
%rg(kontrolle1, tage);

quit;
```

```
/ ****************************************************
  *                                                  *
  * SAS-Programm:         Erzeugung von Stoch. Prozessen *
  *                       für die einzelnen Parzellen.   *
  *                                                  *
  * Erstellt:             25.03.02                   *
  * Autor:                Lars Tschiersch            *
  *                                                  *
  * Zuletzt geändert:     10.04.02                   *
  * Bemerkung:            Beinhaltet Makro           *
  *                                                  *
  *                                          (4)     *
  ****************************************************/

%Macro sim;
%do i=1 %to 36;
  * Erzeugung HIRSE-Prozess;

  data hirse_p&i (drop=i y y_alt1 y_alt2 y_0 e_alt1 e_alt2 e);
   y_alt1=0; y_alt2=0; y_0=0;
   e_alt1=0; e_alt2=0;

   do i=1 to 3000;
     e=normal(0);
     y=(2*i/11)+(0.45*y_alt1+0.17*y_alt2+0.3*e_alt1+0.3*e_alt2+e);
     y_alt2=y_alt1;e_alt2=e_alt1;
     y_alt1=y;e_alt1=e;
     if i=2000 then y_0=y;
     yh&i=y-y_0;
     ii=i-2000;
     if i>=2000 then output;
   end;
  run;

  * Erzeugung PLASTIK-Prozess;

  data plastik_p&i (drop=i y y_alt1 y_alt2 y_0 e_alt1 e_alt2 e);
   y_alt1=0; y_alt2=0; y_0=0;
   e_alt1=0; e_alt2=0;

   do i=1 to 3000;
    e=normal(0);
    y=(2*i/11)+(0.38*y_alt1+0.15*y_alt2+0.2*e_alt1+0.4*e_alt2+e);
    y_alt2=y_alt1;e_alt2=e_alt1;
    y_alt1=y;e_alt1=e;
    if i=1000 then y_0=y;
    yp&i=y-y_0;
    ii=i-1000;
    if i>=1000 then output;
   end;
  run;

  * Erzeugung KONTROLL-Prozess;

  data kontrolle_p&i (drop=i y y_alt1 y_0 e);
   y_alt1=0; y_0=0;

   do i = 1 to 3000;
    e=normal(0);
    y=-(2*i/11)+(0.8*y_alt1+e);
    y_alt1=y;
    if i=1000 then y_0=y;
    yk&i=y-y_0;
```

```
    ii=i-1000;
    if i>=1000 then output;
   end;
  run;

%end;
%Mend;

%sim;

%Macro zus_h;
 data hirse2;
 %do i=1 %to 36;
  merge hirse_p&i;
  by ii;
 %end;
%Mend;

%Macro zus_p;
 data plastik2;
 %do i=1 %to 36;
  merge plastik_p&i;
  by ii;
 %end;
%Mend;

%Macro zus_k;
 data kontrolle2;
 %do i=1 %to 36;
  merge kontrolle_p&i;
  by ii;
 %end;
%Mend;


%zus_h;
%zus_p;
%zus_k;
```

```
/ ******************************************************
  *                                                    *
  * SAS-Programm:       Untersuchung der Zeitreihen auf  *
  *                     Stationarität                  *
  *                                                    *
  * Erstellt:           25.03.02                       *
  * Autor:              Lars Tschiersch                *
  *                                                    *
  * Zuletzt geändert:   11.04.02                       *
  * Bemerkung:          Beinhaltet Makro               *
  *                                          (5)       *
  ******************************************************/


options linesize=80 pagesize=61 nodate pageno=1;

%Macro ausw(dat);
 %do i=1 %to 36;
  proc reg data=&dat noprint outest=e_hr&i;
   model yh&i=ii;
   output out=hr&i r=res;
  run;

 data est_hr&i;
 set e_hr&i;
  beta&i=ii;
  interc&i=intercept;
  keep interc&i beta&i;
 run;

 proc datasets library=work memtype=data;
  delete e_hr&i;
 run;

 proc gplot data=hr&i gout=b;
  title "Trendbereinigte Reihe von yh"&i" ";
  symbol1 i=join c=black;
  axis1 label=(f=swiss h=1.2 angle=90) value=(f=swiss h=0.8);
  axis2 label=(f=swiss h=1.2 'Zeit') value=(f=swiss h=0.8);
  plot res*ii=1 /vaxis=axis1 haxis=axis2;
 run;

 proc arima data=hr&i;
  title "ARIMA-Auswertung von yh"&i" ";
  identify var=res(0) scan stationarity=(pp=6) outcov=est_yh&i;
 run;

 data n_est_yh&i;
  set est_yh&i;
  stderr_o=probit(0.975)*stderr;
  stderr_u=-stderr_o;
 run;

 proc spectra data=hr&i s whitetest out=b;
  title "Output Spectra fuer yh"&i" ";
  var res;
  weights bart 100 0;
 run;

 proc gplot data=b gout=b;
  title "Periodogramm von yh"&i" ";
  axis1  label=(f=swiss h=1.2 angle=90 'Spektrum')
         value=(f=swiss h=0.8);
```

```
 axis2   label=(f=swiss h=1.2 'Frequenz') value=(f=swiss h=0.8);
 plot s_01*freq /vaxis=axis1 haxis=axis2;
run;
quit;

proc gplot data=n_est_yh&i gout=b;
 title "ACF von yh"&i" ";
 symbol1 i=needle l=1 c=black;
 symbol2 i=join l=2 c=black;
 axis1   label=(f=swiss h=1.2 angle=90 'Autokorrelation')
         value=(f=swiss h=0.8);
 axis2   label=(f=swiss h=1.2 'Lag') value=(f=swiss h=0.8)
         offset=(0,0);
 plot corr*lag=1 stderr_o*lag=2 stderr_u*lag=2
          / overlay vaxis=axis1 axis=axis2;
run;
quit;

proc gplot data=n_est_yh&i gout=b;
 symbol1 i=needle l=1 c=black;
 symbol2 i=join l=2 c=black;
 axis1   label=(f=swiss h=1.2 angle=90 'Partielle Autokorrelation')
         value=(f=swiss h=0.8);
axis2    label=(f=swiss h=1.2 'Lag') value=(f=swiss h=0.8)
         offset=(0,0);
title "PACF von yh"&i" ";
plot partcorr*lag=1 stderr_o*lag=2 stderr_u*lag=2
         /overlay vaxis=axis1 haxis=axis2;
run;
quit;

proc greplay tc=temp1 template=tt nofs;
tdef tt des=' '

 1/llx=1  lly=51
   ulx=1  uly=99
   urx=49 ury=99
   lrx=49 lry=51

 2/llx=1 lly=1
   ulx=1 uly=49
   urx=49 ury=49
   lrx=49 lry=1

 3/llx=51 lly=51
   ulx=51 uly=99
   urx=99 ury=99
   lrx=99 lry=51

 4/llx=51 lly=1
   ulx=51 uly=49
   lrx=99 lry=1
   urx=99 ury=49
 ;
 igout=b;
 gout=b;
 treplay 1:1 2:2 3:3 4:4;
 run;
 quit;

 %end;
%Mend;
%ausw(hirse2);
```

```
/ ***************************************************
  *                                                  *
  * SAS-PROGRAMM:         Zusammenfügen des Datensatzes  *
  *                       und anpassen eines Zustands-   *
  *                       raummodells mittels Statespace.*
  *                       Modell und Schätzer werden in  *
  *                       Datensätzen gespeichert.       *
  *                       Anschließend grafische Ausgabe *
  *                       der letzten 20 Beobachtungen   *
  *                       plus des Vorhersagewertes      *
  *                                                  *
  * Autor:                Lars Tschiersch            *
  * Erstellt:             11.04.2002                 *
  * Zuletzt geändert:     11.04.2002                 *
  *                                                  *
  * Bemerkung:            Beinhaltet Makro und       *
  *                       ODS-Ausgaben               *
  *                                                  *
  *                                          (6)     *
  ***************************************************/


%macro h_alle;
 data h_alle (drop=res);
 %do i=1 %to 36;
  set hr&i;
  res&i=res;
  drop yh&i;
 %end;
 run;
%mend;

%h_alle;

/* outmodel=  Paramterschätzer und standarderror
   outar=     Yule-Walker-Schätzungen
   out=       Vorhersagewerte werden hier abgelegt
   print      Ausgabe der Vorhersagewerte im Outputfenster
   lead=      Anzahl der vorhergesagten Werte
   dimmax=    Obergrenze der Anzahl der Elemente im Zustandsvektor
*/


ods rtf file='ZR-Modell.rtf';
proc statespace  data=h_alle dimmax=40 printout=short print
                 lead=5 out=h_alle_out outmodel=h_alle_outmodell;
 var res1-res36;
 id ii;
run;
ods rtf close;
quit;

data h_alle_out1;
 set h_alle_out;
 ii2=ii-1;
run;

/* Rote=  Vorhersagewerte der trendbereinigten reihe
   Blau=  Trendbereinigte Reihe
*/
```

```
goptions keymap=winansi devmap=winansi;

proc gplot data=h_alle_out1;
 symbol1 i=join v=circle c=red l=1;
 symbol2 i=join v=dot c=blue l=1;
 title f=swiss h=2 'Vorhersage';
 axis1 label=(f=swiss h=1.2 angle=90 'Vorhersagewerte für Res1')
       value=(f=swiss h=0.8) offset=(0,0);
 axis2 label=(f=swiss h=1.2 'Lag') value=(f=swiss h=0.8)
       offset=(0,0);
 plot for1*ii2=1 res1*ii=2
       /overlay vaxis=axis1 haxis=axis2 href=1000.5;
 where ii>980;
run;
quit;

%macro betas;
 data beta;
 %do i=1 %to 36;
  merge est_hr&i;
 %end;
 run;
%mend;

%betas;

data betas;
 set beta;
 do ii=1 to 1006;
  output;
 end;
run;

data vorhersage;
 merge betas h_alle_out1;
run;

%macro zum_schluss;
 data finale;
 set vorhersage;
 %do i=1 %to 36;
  yh_final&i=interc&i+beta&i*(ii+1)+for&i;
  yh_res_final&i=interc&i+beta&i*(ii+1)+res&i;
 %end;
 run;
%mend;

%zum_schluss;

data finale1;
 set finale;
 u=yh_final1-probit(0.975)*(std1/sqrt(100));
 o=yh_final1+probit(0.975)*(std1/sqrt(100));
run;



/* Rote Kurve=   Nichttrendbereinigte Reihe (plus Vorhersagewerten)
   Blaue Kurve=  Trendbereinigte Reihe plus Vorhersage
*/
```

```
goptions keymap=winansi devmap=winansi;

proc gplot data=finale;
 symbol1 i=join v=none c=red l=1;
 symbol2 i=join v=none c=blue l=1;
 symbol3 i=join v=none c=black l=2;
 axis1 label=(f=swiss h=1.5 angle=90 'Bodenveraenderung (Hirse)')
       value=(f=swiss h=1.2) offset=(0,0);
 axis2 label=(f=swiss h=1.5 'Zeitpunkt') value=(f=swiss h=1.2)
       offset=(0,0);
 plot yh_final1*ii2=1 yh_res_final1*ii=2
       /overlay href=1000.5 vaxis=axis1 haxis=axis2;
 where ii>900;
run;
quit;

proc gplot data=finale;
 symbol1 i=join v=cirlce h=0.3 c=red l=1;
 symbol2 i=join v=circle h=0.3 c=blue l=1;
 axis1 label=(f=swiss h=1.5 angle=90 'Bodenveraenderung (Hirse)')
       value=(f=swiss h=1.2) offset=(0,0);
 axis2 label=(f=swiss h=1.5 'Zeitpunkt') value=(f=swiss h=1.2)
       offset=(0,0);
 plot yh_final1*ii2=1 yh_res_final1*ii=2
       /overlay href=1000.5 vaxis=axis1 haxis=axis2;
 where ii>980;
run;
quit;

proc print data=finale;
 where ii<10 or ii>990;
run;
quit;
```

Anhang E: Matrizen des Zustandsraummodells

<u>Matrizen der Initialisierungsphase:</u>

$$T_t^{11} = \begin{bmatrix} 0{,}7473 & 0{,}0218 & -0{,}0040 & 0{,}0057 & -0{,}0064 & 0{,}0471 \\ -0{,}0034 & 0{,}7149 & -0{,}0050 & -0{,}0558 & 0{,}0413 & 0{,}0052 \\ -0{,}0191 & -0{,}0087 & 0{,}7114 & 0{,}0327 & 0{,}0076 & -0{,}0132 \\ -0{,}0425 & -0{,}0158 & 0{,}0207 & 0{,}7177 & 0{,}0647 & -0{,}0468 \\ 0{,}0086 & 0{,}0151 & 0{,}0010 & 0{,}0004 & 0{,}7460 & -0{,}0064 \\ -0{,}0041 & -0{,}0259 & 0{,}0371 & 0{,}0084 & -0{,}0009 & 0{,}7366 \\ 0{,}0386 & 0{,}0110 & 0{,}0239 & 0{,}0529 & -0{,}0432 & 0{,}0078 \\ 0{,}0204 & -0{,}0065 & -0{,}0017 & 0{,}0076 & 0{,}0171 & -0{,}0001 \\ -0{,}0451 & 0{,}0226 & 0{,}0394 & -0{,}0353 & 0{,}0009 & -0{,}0417 \\ -0{,}0291 & -0{,}0085 & 0{,}0156 & -0{,}0469 & -0{,}0398 & -0{,}0157 \\ 0{,}0253 & -0{,}0183 & 0{,}0066 & -0{,}0112 & -0{,}0086 & 0{,}0039 \\ -0{,}0229 & 0{,}0017 & 0{,}0181 & -0{,}0489 & 0{,}0249 & 0{,}0023 \\ -0{,}0053 & -0{,}0152 & 0{,}0115 & -0{,}0146 & -0{,}0229 & -0{,}0059 \\ -0{,}0161 & -0{,}0256 & 0{,}0407 & -0{,}0258 & -0{,}0518 & 0{,}0365 \\ 0{,}0145 & 0{,}0056 & 0{,}0017 & 0{,}0086 & 0{,}0065 & 0{,}0081 \\ -0{,}0249 & 0{,}0218 & -0{,}0063 & 0{,}0085 & 0{,}0245 & 0{,}0043 \\ 0{,}0194 & -0{,}0046 & 0{,}0146 & 0{,}0046 & 0{,}0045 & 0{,}0080 \\ -0{,}0145 & 0{,}0194 & 0{,}0462 & -0{,}0328 & 0{,}0033 & -0{,}0319 \end{bmatrix}$$

$$T_t^{21} = \begin{bmatrix} 0{,}0128 & 0{,}0450 & 0{,}0456 & 0{,}0223 & 0{,}0093 & -0{,}0080 \\ -0{,}0164 & 0{,}0188 & 0{,}0146 & 0{,}0081 & 0{,}0425 & -0{,}0132 \\ -0{,}0613 & -0{,}0093 & 0{,}0095 & -0{,}0078 & 0{,}0089 & -0{,}0060 \\ -0{,}0125 & 0{,}0457 & 0{,}0211 & -0{,}0126 & -0{,}0202 & 0{,}0068 \\ 0{,}0017 & -0{,}0094 & -0{,}0080 & 0{,}0401 & 0{,}0535 & -0{,}0188 \\ -0{,}0202 & 0{,}0074 & 0{,}0058 & 0{,}0591 & 0{,}0122 & 0{,}0120 \\ -0{,}0266 & -0{,}0323 & -0{,}0364 & -0{,}0274 & 0{,}0269 & -0{,}0625 \\ -0{,}0212 & 0{,}0149 & 0{,}0249 & 0{,}0041 & -0{,}0071 & -0{,}0001 \\ 0{,}0321 & 0{,}0094 & -0{,}0018 & -0{,}0323 & -0{,}0021 & 0{,}0015 \\ 0{,}0331 & -0{,}0054 & 0{,}0080 & -0{,}0176 & 0{,}0139 & -0{,}0132 \\ -0{,}0029 & -0{,}0391 & -0{,}0181 & 0{,}0206 & 0{,}0056 & 0{,}0330 \\ 0{,}0252 & -0{,}0151 & 0{,}0111 & 0{,}0023 & 0{,}0172 & 0{,}0027 \\ 0{,}0062 & -0{,}0077 & 0{,}0029 & 0{,}0061 & 0{,}0153 & 0{,}0260 \\ -0{,}0226 & -0{,}0061 & 0{,}0042 & -0{,}0377 & -0{,}0294 & -0{,}0083 \\ 0{,}0424 & -0{,}0283 & -0{,}0608 & 0{,}0562 & 0{,}0195 & -0{,}0066 \\ -0{,}0132 & -0{,}0323 & -0{,}0138 & 0{,}0074 & -0{,}0038 & -0{,}0096 \\ 0{,}0299 & -0{,}0134 & 0{,}0048 & 0{,}0436 & -0{,}0695 & 0{,}0235 \\ 0{,}0089 & -0{,}0185 & -0{,}0158 & 0{,}0401 & 0{,}0005 & -0{,}0227 \end{bmatrix}$$

$$T_t^{12} = \begin{bmatrix} 0{,}0070 & -0{,}0013 & 0{,}0398 & 0{,}0215 & -0{,}0108 & 0{,}0030 \\ -0{,}0035 & 0{,}0031 & -0{,}0049 & 0{,}0195 & -0{,}0095 & -0{,}0111 \\ -0{,}0162 & -0{,}0012 & -0{,}0339 & -0{,}0204 & -0{,}0285 & 0{,}0317 \\ 0{,}0012 & -0{,}0148 & 0{,}0160 & 0{,}0313 & 0{,}0044 & -0{,}0015 \\ -0{,}0172 & -0{,}0277 & -0{,}0105 & 0{,}0304 & 0{,}0081 & -0{,}0180 \\ -0{,}0225 & -0{,}0120 & 0{,}0417 & -0{,}0225 & -0{,}0076 & 0{,}0029 \\ 0{,}7569 & -0{,}0203 & -0{,}0033 & -0{,}0552 & -0{,}0148 & -0{,}0210 \\ 0{,}0034 & 0{,}7816 & -0{,}0067 & 0{,}0320 & -0{,}0068 & 0{,}0263 \\ 0{,}0042 & -0{,}0100 & 0{,}7826 & -0{,}0203 & 0{,}0070 & -0{,}0078 \\ -0{,}0179 & -0{,}0164 & -0{,}0007 & 0{,}7542 & 0{,}0073 & 0{,}0308 \\ -0{,}0071 & -0{,}0410 & 0{,}0015 & -0{,}0247 & 0{,}7715 & -0{,}0035 \\ 0{,}0231 & -0{,}0265 & 0{,}0113 & 0{,}0049 & 0{,}0010 & 0{,}7372 \\ -0{,}0302 & 0{,}0081 & -0{,}0058 & 0{,}0237 & -0{,}0462 & 0{,}0277 \\ -0{,}0074 & 0{,}0169 & -0{,}0397 & -0{,}0242 & 0{,}0082 & 0{,}0233 \\ -0{,}0017 & -0{,}0239 & -0{,}0160 & 0{,}0240 & 0{,}0059 & 0{,}0348 \\ 0{,}0308 & 0{,}0175 & -0{,}0040 & 0{,}0084 & -0{,}0090 & -0{,}0015 \\ 0{,}0026 & 0{,}0020 & -0{,}0111 & -0{,}0095 & -0{,}0180 & 0{,}0225 \\ -0{,}0253 & -0{,}0351 & -0{,}0074 & 0{,}0135 & 0{,}0024 & 0{,}0004 \end{bmatrix}$$

$$T_t^{22} = \begin{bmatrix} 0{,}0090 & 0{,}0112 & 0{,}0270 & -0{,}0017 & -0{,}0292 & 0{,}0405 \\ -0{,}0075 & 0{,}0061 & 0{,}0012 & 0{,}0250 & -0{,}0149 & -0{,}0523 \\ 0{,}0123 & 0{,}0102 & 0{,}0090 & -0{,}0145 & -0{,}0031 & 0{,}0076 \\ 0{,}0123 & 0{,}0059 & 0{,}0292 & -0{,}0255 & -0{,}0497 & -0{,}0060 \\ -0{,}0060 & -0{,}0032 & 0{,}0511 & 0{,}0202 & 0{,}0015 & 0{,}0055 \\ 0{,}0150 & 0{,}0184 & -0{,}0264 & -0{,}0236 & -0{,}0042 & 0{,}0099 \\ -0{,}0147 & 0{,}0310 & 0{,}0103 & -0{,}0298 & -0{,}0263 & 0{,}0136 \\ -0{,}0058 & 0{,}0096 & -0{,}0108 & -0{,}0235 & -0{,}0111 & 0{,}0176 \\ -0{,}0067 & 0{,}0167 & -0{,}0043 & -0{,}0264 & -0{,}0168 & 0{,}0103 \\ 0{,}0007 & -0{,}0199 & -0{,}0103 & -0{,}0184 & -0{,}0343 & 0{,}0176 \\ 0{,}0060 & -0{,}0092 & 0{,}0043 & -0{,}0095 & -0{,}0400 & 0{,}0530 \\ 0{,}0441 & 0{,}0063 & 0{,}0181 & 0{,}0043 & -0{,}0128 & -0{,}0243 \\ -0{,}0021 & -0{,}0045 & -0{,}0060 & -0{,}0187 & -0{,}0237 & 0{,}0013 \\ -0{,}0171 & 0{,}0360 & -0{,}0338 & -0{,}0108 & -0{,}0250 & 0{,}0191 \\ -0{,}0364 & 0{,}0174 & -0{,}0686 & 0{,}0133 & -0{,}0058 & 0{,}0195 \\ -0{,}0117 & 0{,}0274 & -0{,}0053 & 0{,}0117 & 0{,}0102 & 0{,}0610 \\ -0{,}0079 & -0{,}0088 & -0{,}0191 & -0{,}0144 & -0{,}0007 & -0{,}0116 \\ 0{,}0118 & -0{,}0017 & -0{,}0034 & -0{,}0157 & -0{,}0543 & 0{,}0404 \end{bmatrix}$$

$$T_t^{13} = \begin{bmatrix} 0,0029 & 0,0235 & -0,0120 & 0,0201 & 0,0042 & -0,0183 \\ -0,0330 & 0,0472 & -0,0034 & 0,0054 & -0,0021 & -0,0493 \\ -0,0194 & 0,0274 & 0,0084 & 0,0223 & -0,0060 & -0,0108 \\ 0,0039 & 0,0362 & 0,0021 & 0,0125 & 0,0409 & -0,0334 \\ 0,0132 & 0,0222 & -0,0219 & -0,0090 & -0,0157 & -0,0119 \\ -0,0054 & 0,0123 & 0,0155 & 0,0199 & 0,0123 & -0,0289 \\ 0,0196 & -0,0054 & 0,0242 & -0,0196 & 0,0307 & -0,0056 \\ -0,0122 & -0,0226 & -0,0050 & 0,0062 & 0,0136 & -0,0109 \\ 0,0197 & 0,0087 & -0,0312 & -0,0246 & 0,0017 & -0,0294 \\ 0,0282 & -0,0170 & 0,0235 & 0,0013 & 0,0029 & -0,0379 \\ -0,0007 & -0,0059 & -0,0072 & -0,0008 & 0,0149 & 0,0161 \\ -0,0027 & 0,0094 & -0,0112 & -0,0134 & -0,0267 & 0,0246 \\ 0,7520 & -0,0070 & 0,0488 & 0,0361 & 0,0177 & -0,0459 \\ -0,0401 & 0,7502 & -0,0173 & -0,0023 & 0,0014 & -0,0138 \\ -0,0316 & 0,0080 & 0,7492 & 0,0130 & 0,0098 & 0,0123 \\ -0,0040 & 0,0428 & -0,0015 & 0,7666 & -0,0256 & -0,0184 \\ -0,0618 & -0,0097 & -0,0124 & -0,0120 & 0,7438 & -0,0060 \\ -0,0035 & -0,0155 & 0,0131 & 0,0123 & 0,0134 & 0,7677 \end{bmatrix}$$

$$T_t^{23} = \begin{bmatrix} 0,0249 & 0,0047 & -0,0341 & -0,0073 & 0,0151 & -0,0145 \\ 0,0116 & 0,0019 & -0,0210 & 0,0249 & 0,0137 & -0,0081 \\ 0,0180 & 0,0004 & 0,0001 & 0,0215 & -0,0049 & -0,0078 \\ 0,0019 & 0,0217 & -0,0032 & 0,0414 & -0,0051 & 0,0190 \\ -0,0115 & -0,0109 & 0,0055 & 0,0058 & 0,0458 & 0,0028 \\ 0,0270 & -0,0062 & -0,0332 & -0,0376 & 0,0266 & 0,0024 \\ 0,0386 & -0,0097 & 0,0235 & 0,0319 & 0,0070 & 0,0187 \\ 0,0291 & -0,0281 & 0,0371 & -0,0043 & 0,0407 & 0,0113 \\ 0,0498 & -0,0046 & 0,0329 & 0,0045 & -0,0060 & 0,0083 \\ -0,0373 & 0,0193 & -0,0195 & 0,0250 & -0,0473 & 0,0192 \\ 0,0023 & 0,0161 & 0,0242 & -0,0439 & -0,0326 & 0,0090 \\ -0,0211 & 0,0196 & -0,0328 & 0,0148 & -0,0528 & 0,0052 \\ -0,0024 & -0,0031 & -0,0059 & 0,0045 & 0,0060 & 0,0066 \\ -0,0405 & 0,0181 & 0,0205 & -0,0173 & -0,0539 & -0,0288 \\ 0,0101 & 0,0245 & 0,0196 & 0,0247 & 0,0298 & -0,0179 \\ 0,0074 & 0,0020 & 0,0310 & -0,0098 & 0,0142 & -0,0423 \\ 0,0118 & -0,0150 & -0,0214 & -0,0315 & -0,0196 & -0,0154 \\ 0,0211 & -0,0336 & -0,0354 & -0,0125 & -0,0033 & 0,0087 \end{bmatrix}$$

$$T_t^{14} = \begin{bmatrix} 0{,}0269 & -0{,}0055 & 0{,}0132 & -0{,}0287 & 0{,}0211 & -0{,}0457 \\ -0{,}0156 & -0{,}0251 & -0{,}0217 & -0{,}0009 & 0{,}0059 & 0{,}0132 \\ -0{,}0173 & 0{,}0169 & -0{,}0014 & -0{,}0207 & 0{,}0009 & 0{,}0053 \\ -0{,}0302 & -0{,}0678 & -0{,}0051 & -0{,}0122 & 0{,}0083 & 0{,}0321 \\ 0{,}0180 & 0{,}0071 & 0{,}0073 & -0{,}0074 & -0{,}0191 & 0{,}0063 \\ -0{,}0680 & -0{,}0287 & -0{,}0366 & 0{,}0351 & -0{,}0321 & 0{,}0050 \\ 0{,}0108 & 0{,}0099 & -0{,}0033 & -0{,}0036 & -0{,}0102 & -0{,}0097 \\ 0{,}0298 & -0{,}0298 & -0{,}0448 & 0{,}0225 & -0{,}0327 & 0{,}0336 \\ 0{,}0007 & -0{,}0127 & 0{,}0067 & -0{,}0040 & -0{,}0005 & 0{,}0332 \\ -0{,}0109 & 0{,}0278 & 0{,}0234 & 0{,}0252 & 0{,}0127 & -0{,}0147 \\ 0{,}0168 & 0{,}0551 & 0{,}0239 & -0{,}0058 & -0{,}0057 & 0{,}0232 \\ 0{,}0109 & 0{,}0179 & -0{,}0265 & -0{,}0214 & 0{,}0013 & 0{,}0216 \\ -0{,}0493 & -0{,}0171 & -0{,}0082 & -0{,}0064 & -0{,}0027 & 0{,}0234 \\ -0{,}0149 & 0{,}0388 & 0{,}0054 & -0{,}0073 & -0{,}0023 & -0{,}0084 \\ -0{,}0075 & -0{,}0400 & -0{,}0060 & 0{,}0132 & 0{,}0067 & 0{,}0096 \\ -0{,}0318 & 0{,}0128 & -0{,}0042 & -0{,}0040 & -0{,}0110 & 0{,}0265 \\ 0{,}0275 & -0{,}0244 & 0{,}0205 & 0{,}0367 & 0{,}0104 & -0{,}0351 \\ -0{,}0293 & -0{,}0011 & -0{,}0292 & 0{,}0142 & -0{,}0054 & 0{,}0108 \end{bmatrix}$$

$$T_t^{24} = \begin{bmatrix} 0{,}7650 & -0{,}0226 & -0{,}0305 & 0{,}0316 & 0{,}0045 & 0{,}0219 \\ -0{,}0035 & 0{,}7556 & -0{,}0012 & -0{,}0004 & -0{,}0068 & 0{,}0135 \\ 0{,}0102 & -0{,}0011 & 0{,}7544 & 0{,}0122 & -0{,}0129 & 0{,}0109 \\ -0{,}0157 & -0{,}0026 & 0{,}0229 & 0{,}7538 & 0{,}0274 & -0{,}0073 \\ -0{,}0267 & -0{,}0132 & -0{,}0343 & 0{,}0011 & 0{,}7732 & -0{,}0043 \\ -0{,}0236 & 0{,}0079 & -0{,}0039 & -0{,}0229 & -0{,}0138 & 0{,}7653 \\ -0{,}0055 & -0{,}0215 & -0{,}0010 & -0{,}0487 & -0{,}0164 & 0{,}0007 \\ -0{,}0042 & 0{,}0148 & 0{,}0150 & 0{,}0082 & -0{,}0029 & 0{,}0043 \\ -0{,}0001 & -0{,}0120 & 0{,}0007 & -0{,}0292 & -0{,}0203 & -0{,}0188 \\ 0{,}0199 & 0{,}0145 & 0{,}0089 & -0{,}0173 & 0{,}0379 & 0{,}0063 \\ -0{,}0009 & 0{,}0297 & 0{,}0182 & -0{,}0079 & -0{,}0047 & 0{,}0182 \\ 0{,}0107 & -0{,}0252 & -0{,}0012 & -0{,}0144 & 0{,}0073 & -0{,}0251 \\ 0{,}0048 & -0{,}0279 & 0{,}0052 & -0{,}0433 & 0{,}0167 & 0{,}0061 \\ -0{,}0013 & 0{,}0625 & -0{,}0067 & -0{,}0178 & 0{,}0183 & 0{,}0666 \\ -0{,}0081 & 0{,}0165 & 0{,}0151 & 0{,}0012 & -0{,}0185 & -0{,}0258 \\ -0{,}0082 & 0{,}0465 & -0{,}0117 & -0{,}0050 & -0{,}0037 & -0{,}0236 \\ 0{,}0229 & 0{,}0110 & -0{,}0045 & 0{,}0025 & -0{,}0082 & -0{,}0085 \\ 0{,}0014 & -0{,}0146 & 0{,}0285 & 0{,}0349 & -0{,}0231 & -0{,}0164 \end{bmatrix}$$

$$T_t^{15} = \begin{bmatrix} -0{,}0199 & 0{,}0403 & 0{,}0057 & -0{,}0036 & -0{,}0087 & -0{,}0056 \\ 0{,}0275 & -0{,}0179 & -0{,}0163 & -0{,}0391 & -0{,}0514 & 0{,}0093 \\ 0{,}0126 & 0{,}0140 & 0{,}0412 & -0{,}0338 & 0{,}0268 & 0{,}0154 \\ -0{,}0111 & 0{,}0086 & 0{,}0073 & -0{,}0115 & 0{,}0056 & -0{,}0023 \\ 0{,}0277 & 0{,}0143 & 0{,}0277 & -0{,}0031 & 0{,}0069 & -0{,}0129 \\ 0{,}0238 & 0{,}0048 & -0{,}0754 & 0{,}0580 & -0{,}0320 & -0{,}0325 \\ -0{,}0077 & -0{,}0006 & -0{,}0093 & -0{,}0022 & 0{,}0008 & -0{,}0098 \\ 0{,}0058 & 0{,}0055 & 0{,}0169 & -0{,}0084 & 0{,}0122 & -0{,}0100 \\ 0{,}0184 & 0{,}0169 & -0{,}0043 & -0{,}0133 & -0{,}0074 & 0{,}0087 \\ 0{,}0312 & 0{,}0009 & -0{,}0235 & -0{,}0095 & 0{,}0247 & -0{,}0080 \\ -0{,}0049 & 0{,}0015 & -0{,}0077 & -0{,}0408 & -0{,}0251 & -0{,}0004 \\ -0{,}0022 & -0{,}0078 & 0{,}0228 & -0{,}0259 & -0{,}0170 & 0{,}0125 \\ -0{,}0498 & 0{,}0041 & 0{,}0253 & 0{,}0333 & -0{,}0029 & -0{,}0178 \\ -0{,}0022 & -0{,}0441 & 0{,}0161 & 0{,}0157 & 0{,}0083 & -0{,}0366 \\ -0{,}0483 & -0{,}0363 & 0{,}0158 & 0{,}0046 & 0{,}0060 & -0{,}0309 \\ -0{,}0116 & 0{,}0227 & 0{,}0152 & -0{,}0110 & -0{,}0008 & 0{,}0147 \\ 0{,}0443 & -0{,}0257 & 0{,}0311 & 0{,}0108 & -0{,}0005 & -0{,}0180 \\ 0{,}0141 & -0{,}0204 & -0{,}0392 & 0{,}0174 & 0{,}0116 & 0{,}0032 \end{bmatrix}$$

$$T_t^{25} = \begin{bmatrix} -0{,}0078 & 0{,}0077 & -0{,}0016 & -0{,}0015 & 0{,}0261 & -0{,}0102 \\ 0{,}0024 & -0{,}0308 & 0{,}0574 & -0{,}0345 & 0{,}0016 & 0{,}0010 \\ -0{,}0124 & 0{,}0311 & -0{,}0126 & 0{,}0137 & -0{,}0001 & -0{,}0445 \\ -0{,}0168 & -0{,}0293 & 0{,}0336 & -0{,}0066 & -0{,}0325 & 0{,}0131 \\ -0{,}0037 & -0{,}0025 & -0{,}0060 & -0{,}0037 & 0{,}0248 & 0{,}0046 \\ -0{,}0053 & 0{,}0143 & 0{,}0039 & 0{,}0530 & 0{,}0290 & 0{,}0348 \\ 0{,}7153 & -0{,}0003 & -0{,}0192 & 0{,}0592 & -0{,}0024 & -0{,}0092 \\ -0{,}0128 & 0{,}7508 & 0{,}0044 & -0{,}0236 & -0{,}0150 & -0{,}0148 \\ 0{,}0333 & -0{,}0182 & 0{,}7228 & 0{,}0217 & -0{,}0100 & 0{,}0047 \\ -0{,}0234 & 0{,}0114 & 0{,}0355 & 0{,}7862 & 0{,}0034 & -0{,}0397 \\ 0{,}0096 & 0{,}0302 & 0{,}0449 & -0{,}0459 & 0{,}7556 & 0{,}0339 \\ 0{,}0091 & -0{,}0012 & -0{,}0246 & -0{,}0032 & -0{,}0311 & 0{,}7589 \\ 0{,}0164 & 0{,}0066 & 0{,}0050 & 0{,}0049 & 0{,}0246 & -0{,}0133 \\ -0{,}0220 & -0{,}0174 & -0{,}0208 & -0{,}0156 & -0{,}0300 & -0{,}0268 \\ 0{,}0485 & -0{,}0588 & -0{,}0044 & 0{,}0219 & 0{,}0074 & 0{,}0288 \\ -0{,}0113 & -0{,}0243 & 0{,}0273 & 0{,}0058 & 0{,}0133 & 0{,}0162 \\ 0{,}0542 & 0{,}0045 & -0{,}0240 & -0{,}0116 & 0{,}0502 & 0{,}0659 \\ -0{,}0266 & -0{,}0455 & 0{,}0451 & -0{,}0258 & -0{,}0013 & 0{,}0226 \end{bmatrix}$$

$$T_t^{16} = \begin{bmatrix} 0{,}0044 & 0{,}0042 & 0{,}0517 & -0{,}0059 & -0{,}0007 & -0{,}0163 \\ 0{,}0115 & -0{,}0066 & 0{,}0053 & 0{,}0054 & 0{,}0196 & -0{,}0279 \\ -0{,}0049 & -0{,}0210 & -0{,}0326 & -0{,}0348 & -0{,}0642 & -0{,}0089 \\ 0{,}0208 & -0{,}0237 & -0{,}0405 & 0{,}0140 & -0{,}0215 & 0{,}0418 \\ 0{,}0246 & -0{,}0019 & -0{,}0197 & 0{,}0167 & -0{,}0101 & -0{,}0126 \\ -0{,}0163 & -0{,}0021 & -0{,}0337 & -0{,}0177 & -0{,}0318 & 0{,}0216 \\ 0{,}0415 & 0{,}0360 & 0{,}0002 & 0{,}0200 & 0{,}0142 & 0{,}0050 \\ 0{,}0125 & -0{,}0234 & -0{,}0326 & 0{,}0308 & -0{,}0301 & 0{,}0323 \\ 0{,}0244 & 0{,}0080 & 0{,}0367 & 0{,}0148 & 0{,}0051 & 0{,}0161 \\ 0{,}0501 & 0{,}0286 & -0{,}0392 & 0{,}0062 & 0{,}0165 & -0{,}0149 \\ -0{,}0048 & 0{,}0217 & 0{,}0044 & -0{,}0047 & 0{,}0253 & -0{,}0308 \\ -0{,}0243 & -0{,}0635 & 0{,}0102 & 0{,}0188 & 0{,}0280 & 0{,}0330 \\ -0{,}0006 & 0{,}0223 & -0{,}0395 & 0{,}0066 & -0{,}0057 & -0{,}0278 \\ 0{,}0021 & -0{,}0033 & -0{,}0312 & 0{,}0247 & 0{,}0108 & 0{,}0166 \\ 0{,}0031 & 0{,}0282 & 0{,}0264 & 0{,}0278 & 0{,}0391 & -0{,}0236 \\ -0{,}0444 & -0{,}0210 & -0{,}0306 & -0{,}0198 & 0{,}0110 & -0{,}0134 \\ -0{,}0153 & 0{,}0102 & 0{,}0022 & 0{,}0153 & -0{,}0009 & 0{,}0024 \\ -0{,}0093 & 0{,}0155 & -0{,}0010 & 0{,}0102 & 0{,}0035 & 0{,}0505 \end{bmatrix}$$

$$T_t^{26} = \begin{bmatrix} -0{,}0047 & 0{,}0217 & -0{,}0304 & 0{,}0078 & -0{,}0224 & 0{,}0201 \\ 0{,}0409 & -0{,}0298 & 0{,}0215 & 0{,}0269 & -0{,}0071 & 0{,}0243 \\ -0{,}0002 & -0{,}0292 & 0{,}0183 & 0{,}0317 & -0{,}0081 & 0{,}0092 \\ 0{,}0013 & -0{,}0079 & 0{,}0381 & 0{,}0075 & 0{,}0201 & 0{,}0103 \\ -0{,}0165 & -0{,}0160 & -0{,}0126 & -0{,}0049 & 0{,}0061 & -0{,}0052 \\ -0{,}0270 & 0{,}0052 & 0{,}0235 & 0{,}0162 & -0{,}0389 & 0{,}0055 \\ 0{,}0015 & -0{,}0100 & -0{,}0066 & -0{,}0028 & 0{,}0497 & -0{,}0011 \\ -0{,}0271 & -0{,}0268 & 0{,}0298 & -0{,}0056 & -0{,}0079 & -0{,}0130 \\ 0{,}0422 & 0{,}0119 & -0{,}0341 & -0{,}0152 & 0{,}0373 & 0{,}0276 \\ 0{,}0032 & 0{,}0299 & -0{,}0251 & 0{,}0207 & -0{,}0048 & -0{,}0167 \\ -0{,}0537 & -0{,}0287 & 0{,}0071 & -0{,}0170 & 0{,}0051 & -0{,}0271 \\ -0{,}0189 & 0{,}0221 & -0{,}0013 & 0{,}0182 & 0{,}0090 & -0{,}0099 \\ 0{,}7617 & 0{,}0166 & 0{,}0182 & 0{,}0216 & 0{,}0032 & -0{,}0064 \\ -0{,}0434 & 0{,}7651 & -0{,}0299 & -0{,}0119 & 0{,}0233 & 0{,}0273 \\ 0{,}0146 & 0{,}0074 & 0{,}7175 & -0{,}0426 & 0{,}0224 & -0{,}0204 \\ -0{,}0218 & 0{,}0285 & -0{,}0296 & 0{,}7469 & -0{,}0164 & -0{,}0234 \\ 0{,}0125 & 0{,}0126 & 0{,}0022 & 0{,}0249 & 0{,}7457 & 0{,}0023 \\ -0{,}0037 & 0{,}0329 & -0{,}0221 & -0{,}0069 & 0{,}0334 & 0{,}7325 \end{bmatrix}$$

$$Q_t^{11} = \begin{bmatrix} 0{,}9914 & 0{,}0078 & 0{,}0239 & -0{,}0617 & -0{,}0020 & -0{,}0461 \\ 0{,}0078 & 1{,}0079 & 0{,}0172 & 0{,}0301 & 0{,}0136 & -0{,}0217 \\ 0{,}0239 & 0{,}0172 & 1{,}0510 & 0{,}0011 & 0{,}0428 & -0{,}0433 \\ -0{,}0617 & 0{,}0301 & 0{,}0011 & 1{,}0052 & 0{,}0339 & 0{,}0516 \\ -0{,}0020 & 0{,}0136 & 0{,}0428 & 0{,}0339 & 1{,}0420 & 0{,}0428 \\ -0{,}0461 & -0{,}0217 & -0{,}0433 & 0{,}0516 & 0{,}0428 & 0{,}9690 \\ -0{,}0342 & -0{,}0026 & -0{,}0130 & 0{,}0264 & 0{,}0132 & 0{,}0278 \\ -0{,}0241 & 0{,}0312 & 0{,}0366 & -0{,}0237 & 0{,}0016 & 0{,}0138 \\ 0{,}0349 & -0{,}0073 & -0{,}0206 & -0{,}0350 & -0{,}0614 & 0{,}0290 \\ 0{,}0385 & -0{,}0358 & 0{,}0585 & 0{,}0118 & -0{,}0066 & -0{,}0053 \\ 0{,}0213 & 0{,}0462 & 0{,}0261 & -0{,}0229 & 0{,}0014 & 0{,}0002 \\ -0{,}0140 & -0{,}0407 & 0{,}0215 & -0{,}0115 & -0{,}0070 & 0{,}0217 \\ 0{,}0177 & 0{,}0574 & 0{,}0981 & -0{,}0070 & 0{,}0077 & -0{,}0520 \\ 0{,}0256 & 0{,}0598 & -0{,}0541 & 0{,}0361 & -0{,}0055 & 0{,}0044 \\ 0{,}0753 & -0{,}0173 & 0{,}0397 & -0{,}0046 & 0{,}0323 & 0{,}0212 \\ -0{,}0045 & -0{,}0491 & -0{,}0523 & 0{,}0203 & 0{,}0083 & 0{,}0339 \\ 0{,}0236 & -0{,}0126 & 0{,}0475 & -0{,}0050 & 0{,}0386 & 0{,}0107 \\ -0{,}0236 & -0{,}0041 & 0{,}0882 & -0{,}0610 & 0{,}0156 & 0{,}0775 \end{bmatrix}$$

$$Q_t^{21} = \begin{bmatrix} -0{,}0119 & 0{,}0502 & 0{,}0278 & -0{,}0158 & 0{,}0365 & 0{,}0633 \\ -0{,}0302 & -0{,}0048 & -0{,}0236 & 0{,}0489 & 0{,}0086 & 0{,}0412 \\ -0{,}0499 & -0{,}0012 & 0{,}0358 & 0{,}0351 & -0{,}0041 & -0{,}0050 \\ 0{,}0391 & 0{,}0672 & 0{,}0317 & 0{,}0336 & 0{,}0338 & 0{,}0529 \\ -0{,}0445 & -0{,}0419 & -0{,}0359 & -0{,}0199 & 0{,}0249 & -0{,}0209 \\ 0{,}0045 & 0{,}0159 & -0{,}0508 & -0{,}0280 & -0{,}0707 & -0{,}0126 \\ 0{,}0041 & -0{,}0029 & 0{,}0091 & 0{,}0154 & -0{,}0014 & 0{,}0221 \\ 0{,}0070 & 0{,}0414 & 0{,}0287 & 0{,}0097 & -0{,}0312 & -0{,}0571 \\ -0{,}0208 & -0{,}0322 & -0{,}0180 & -0{,}0694 & -0{,}0519 & 0{,}0069 \\ 0{,}0458 & -0{,}0003 & 0{,}0316 & 0{,}0281 & -0{,}0191 & -0{,}0147 \\ 0{,}0360 & 0{,}0361 & 0{,}0951 & -0{,}0551 & -0{,}0739 & -0{,}0746 \\ 0{,}0106 & 0{,}0265 & -0{,}0414 & 0{,}0679 & 0{,}0450 & 0{,}0276 \\ 0{,}0534 & 0{,}0326 & 0{,}0091 & -0{,}0894 & 0{,}1103 & -0{,}0548 \\ -0{,}0055 & -0{,}0193 & 0{,}0005 & -0{,}0068 & 0{,}0499 & 0{,}0181 \\ 0{,}0502 & -0{,}0772 & -0{,}0034 & -0{,}0033 & 0{,}0018 & -0{,}0184 \\ -0{,}0461 & 0{,}0227 & -0{,}0573 & 0{,}0468 & 0{,}0588 & 0{,}0756 \\ -0{,}0245 & 0{,}0202 & 0{,}0321 & 0{,}0255 & -0{,}0053 & -0{,}0118 \\ -0{,}0062 & 0{,}0083 & -0{,}0703 & -0{,}0139 & 0{,}0066 & -0{,}0106 \end{bmatrix}$$

$$Q_t^{12} = \begin{bmatrix} -0{,}0342 & -0{,}0241 & 0{,}0349 & 0{,}0385 & 0{,}0213 & -0{,}0140 \\ -0{,}0026 & 0{,}0312 & -0{,}0073 & -0{,}0358 & 0{,}0462 & -0{,}0407 \\ -0{,}0130 & 0{,}0366 & -0{,}0206 & 0{,}0585 & 0{,}0261 & 0{,}0215 \\ 0{,}0264 & -0{,}0237 & -0{,}0350 & 0{,}0118 & -0{,}0229 & -0{,}0115 \\ 0{,}0132 & 0{,}0016 & -0{,}0614 & -0{,}0066 & 0{,}0014 & -0{,}0070 \\ 0{,}0278 & 0{,}0138 & 0{,}0290 & -0{,}0053 & 0{,}0002 & 0{,}0217 \\ 1{,}0255 & 0{,}0276 & 0{,}0662 & -0{,}0238 & -0{,}0163 & 0{,}0408 \\ 0{,}0276 & 1{,}0416 & -0{,}0224 & -0{,}0301 & -0{,}0393 & -0{,}0406 \\ 0{,}0662 & -0{,}0224 & 1{,}0111 & -0{,}0348 & 0{,}0116 & -0{,}0393 \\ -0{,}0238 & -0{,}0301 & -0{,}0348 & 1{,}0374 & -0{,}0084 & 0{,}1096 \\ -0{,}0163 & -0{,}0393 & 0{,}0116 & -0{,}0084 & 1{,}0385 & -0{,}0563 \\ 0{,}0408 & -0{,}0406 & -0{,}0393 & 0{,}1096 & -0{,}0563 & 1{,}1175 \\ -0{,}0075 & -0{,}0342 & -0{,}0300 & 0{,}0153 & 0{,}0097 & 0{,}0721 \\ -0{,}0201 & -0{,}0311 & -0{,}0487 & -0{,}0007 & -0{,}0427 & -0{,}0142 \\ 0{,}0433 & 0{,}0341 & 0{,}0171 & 0{,}0415 & -0{,}0552 & -0{,}0276 \\ -0{,}0033 & 0{,}0399 & -0{,}0312 & 0{,}0434 & 0{,}0108 & 0{,}0133 \\ -0{,}0173 & 0{,}0325 & -0{,}0052 & 0{,}0375 & 0{,}0150 & 0{,}0355 \\ 0{,}0382 & 0{,}0003 & 0{,}0123 & 0{,}0574 & -0{,}0419 & -0{,}0036 \end{bmatrix}$$

$$Q_t^{22} = \begin{bmatrix} -0{,}0029 & 0{,}0234 & 0{,}0304 & 0{,}0050 & 0{,}0338 & 0{,}0273 \\ -0{,}0153 & 0{,}0309 & 0{,}0197 & 0{,}0079 & -0{,}0095 & 0{,}0646 \\ -0{,}0270 & -0{,}0490 & -0{,}0012 & -0{,}0161 & -0{,}0293 & 0{,}0271 \\ 0{,}0089 & -0{,}0377 & 0{,}0347 & 0{,}0438 & 0{,}0632 & 0{,}0098 \\ 0{,}0745 & -0{,}0036 & 0{,}0346 & -0{,}0241 & 0{,}0168 & -0{,}0147 \\ -0{,}0372 & 0{,}0609 & -0{,}0049 & -0{,}0183 & 0{,}0603 & 0{,}0261 \\ 0{,}0069 & -0{,}0025 & 0{,}0155 & 0{,}0497 & 0{,}0218 & 0{,}0110 \\ -0{,}0354 & 0{,}0055 & -0{,}0181 & 0{,}0106 & -0{,}0044 & 0{,}0019 \\ 0{,}0186 & -0{,}0287 & 0{,}0718 & -0{,}0245 & 0{,}0568 & 0{,}0105 \\ 0{,}0413 & 0{,}0127 & -0{,}0310 & 0{,}0409 & 0{,}0198 & 0{,}0719 \\ 0{,}0251 & 0{,}0326 & 0{,}0231 & -0{,}0333 & 0{,}0573 & 0{,}0104 \\ -0{,}0230 & -0{,}0098 & 0{,}0312 & 0{,}0292 & 0{,}0369 & 0{,}0253 \\ 0{,}0409 & -0{,}0082 & 0{,}0127 & 0{,}0081 & -0{,}0663 & -0{,}0367 \\ -0{,}0222 & 0{,}0016 & -0{,}0126 & 0{,}0060 & -0{,}0258 & 0{,}0066 \\ 0{,}0546 & 0{,}0461 & 0{,}0502 & 0{,}0122 & 0{,}0318 & 0{,}0014 \\ -0{,}0213 & -0{,}0066 & 0{,}0140 & -0{,}0160 & -0{,}0219 & -0{,}0578 \\ -0{,}0505 & 0{,}0233 & -0{,}0319 & 0{,}0225 & 0{,}0121 & 0{,}0534 \\ -0{,}0018 & -0{,}0332 & 0{,}0710 & 0{,}0556 & -0{,}0547 & -0{,}0238 \end{bmatrix}$$

$$Q_t^{13} = \begin{bmatrix} 0{,}0177 & 0{,}0256 & 0{,}0753 & -0{,}0045 & 0{,}0236 & -0{,}0236 \\ 0{,}0574 & 0{,}0598 & -0{,}0173 & -0{,}0491 & -0{,}0126 & -0{,}0041 \\ 0{,}0981 & -0{,}0541 & 0{,}0397 & -0{,}0523 & 0{,}0475 & 0{,}0882 \\ -0{,}0070 & 0{,}0361 & -0{,}0046 & 0{,}0203 & -0{,}0050 & -0{,}0610 \\ 0{,}0077 & -0{,}0055 & 0{,}0323 & 0{,}0083 & 0{,}0386 & 0{,}0156 \\ -0{,}0520 & 0{,}0044 & 0{,}0212 & 0{,}0339 & 0{,}0107 & 0{,}0775 \\ -0{,}0075 & -0{,}0201 & 0{,}0433 & -0{,}0033 & -0{,}0173 & 0{,}0382 \\ -0{,}0342 & -0{,}0311 & 0{,}0341 & 0{,}0399 & 0{,}0325 & 0{,}0003 \\ -0{,}0300 & -0{,}0487 & 0{,}0171 & -0{,}0312 & -0{,}0052 & 0{,}0123 \\ 0{,}0153 & -0{,}0007 & 0{,}0415 & 0{,}0434 & 0{,}0375 & 0{,}0574 \\ 0{,}0097 & -0{,}0427 & -0{,}0552 & 0{,}0108 & 0{,}0150 & -0{,}0419 \\ 0{,}0721 & -0{,}0142 & -0{,}0276 & 0{,}0133 & 0{,}0355 & -0{,}0036 \\ 0{,}9898 & 0{,}0199 & -0{,}0382 & 0{,}0182 & -0{,}0169 & -0{,}0041 \\ 0{,}0199 & 1{,}0964 & 0{,}0461 & -0{,}0049 & -0{,}0117 & -0{,}0085 \\ -0{,}0382 & 0{,}0461 & 0{,}9115 & 0{,}0570 & -0{,}0292 & -0{,}0256 \\ 0{,}0182 & -0{,}0049 & 0{,}0570 & 0{,}9716 & 0{,}0773 & -0{,}0604 \\ -0{,}0169 & -0{,}0117 & -0{,}0292 & 0{,}0773 & 0{,}9395 & 0{,}0275 \\ -0{,}0041 & -0{,}0085 & -0{,}0256 & -0{,}0604 & 0{,}0275 & 1{,}1981 \end{bmatrix}$$

$$Q_t^{23} = \begin{bmatrix} -0{,}0037 & -0{,}0295 & -0{,}0493 & -0{,}0806 & -0{,}0407 & 0{,}0273 \\ 0{,}0093 & 0{,}0263 & -0{,}0386 & 0{,}0213 & 0{,}0355 & 0{,}0163 \\ -0{,}0301 & 0{,}0462 & 0{,}0234 & 0{,}0156 & 0{,}0262 & -0{,}0360 \\ 0{,}0652 & 0{,}0769 & 0{,}0018 & 0{,}0092 & 0{,}0376 & 0{,}0401 \\ 0{,}0510 & 0{,}0613 & 0{,}0176 & 0{,}0050 & -0{,}0024 & 0{,}0052 \\ 0{,}0079 & -0{,}0057 & -0{,}0158 & -0{,}0076 & -0{,}0187 & -0{,}0086 \\ 0{,}0514 & 0{,}0106 & 0{,}0213 & 0{,}0166 & 0{,}0594 & 0{,}0698 \\ -0{,}0302 & 0{,}0172 & -0{,}0212 & 0{,}0252 & -0{,}0307 & -0{,}0129 \\ 0{,}0013 & -0{,}0627 & -0{,}0508 & 0{,}0248 & -0{,}0201 & 0{,}0682 \\ 0{,}0689 & 0{,}0163 & -0{,}0267 & 0{,}0068 & 0{,}0033 & 0{,}0406 \\ 0{,}0792 & 0{,}0357 & -0{,}0213 & 0{,}0247 & 0{,}0459 & 0{,}0451 \\ -0{,}0335 & 0{,}0494 & 0{,}0108 & 0{,}0298 & -0{,}0657 & -0{,}0159 \\ 0{,}0394 & -0{,}0099 & -0{,}0073 & -0{,}0234 & 0{,}0260 & 0{,}0036 \\ -0{,}0623 & 0{,}0390 & 0{,}0422 & 0{,}0157 & -0{,}0051 & 0{,}0086 \\ 0{,}0131 & -0{,}0083 & 0{,}0251 & -0{,}0098 & -0{,}0057 & -0{,}0219 \\ -0{,}0132 & 0{,}0163 & 0{,}0044 & 0{,}0031 & -0{,}0370 & -0{,}0270 \\ -0{,}0124 & 0{,}0549 & 0{,}0198 & 0{,}0447 & 0{,}0190 & -0{,}0038 \\ -0{,}0055 & -0{,}0751 & -0{,}0214 & 0{,}0236 & 0{,}0669 & 0{,}0359 \end{bmatrix}$$

$$
Q_t^{14} = \begin{bmatrix}
-0{,}0119 & -0{,}0302 & -0{,}0499 & 0{,}0391 & -0{,}0445 & 0{,}0045 \\
0{,}0502 & -0{,}0048 & -0{,}0012 & 0{,}0672 & -0{,}0419 & 0{,}0159 \\
0{,}0278 & -0{,}0236 & 0{,}0358 & 0{,}0317 & -0{,}0359 & -0{,}0508 \\
-0{,}0158 & 0{,}0489 & 0{,}0351 & 0{,}0336 & -0{,}0199 & -0{,}0280 \\
0{,}0365 & 0{,}0086 & -0{,}0041 & 0{,}0338 & 0{,}0249 & -0{,}0707 \\
0{,}0633 & 0{,}0412 & -0{,}0050 & 0{,}0529 & -0{,}0209 & -0{,}0126 \\
-0{,}0029 & -0{,}0153 & -0{,}0270 & 0{,}0089 & 0{,}0745 & -0{,}0372 \\
0{,}0234 & 0{,}0309 & -0{,}0490 & -0{,}0377 & -0{,}0036 & 0{,}0609 \\
0{,}0304 & 0{,}0197 & -0{,}0012 & 0{,}0347 & 0{,}0346 & -0{,}0049 \\
0{,}0050 & 0{,}0079 & -0{,}0161 & 0{,}0438 & -0{,}0241 & -0{,}0183 \\
0{,}0338 & -0{,}0095 & -0{,}0293 & 0{,}0632 & 0{,}0168 & 0{,}0603 \\
0{,}0273 & 0{,}0646 & 0{,}0271 & 0{,}0098 & -0{,}0147 & 0{,}0261 \\
-0{,}0037 & 0{,}0093 & -0{,}0301 & 0{,}0652 & 0{,}0510 & 0{,}0079 \\
-0{,}0295 & 0{,}0263 & 0{,}0462 & 0{,}0769 & 0{,}0613 & -0{,}0057 \\
-0{,}0493 & -0{,}0386 & 0{,}0234 & 0{,}0018 & 0{,}0176 & -0{,}0158 \\
-0{,}0806 & 0{,}0213 & 0{,}0156 & 0{,}0092 & 0{,}0050 & -0{,}0076 \\
-0{,}0407 & 0{,}0355 & 0{,}0262 & 0{,}0376 & -0{,}0024 & -0{,}0187 \\
0{,}0273 & 0{,}0163 & -0{,}0360 & 0{,}0401 & 0{,}0052 & -0{,}0086
\end{bmatrix}
$$

$$
Q_t^{24} = \begin{bmatrix}
1{,}0707 & -0{,}0023 & 0{,}0344 & 0{,}0026 & -0{,}0173 & 0{,}0369 \\
-0{,}0023 & 0{,}9689 & -0{,}0109 & 0{,}0573 & 0{,}0081 & -0{,}0299 \\
0{,}0344 & -0{,}0109 & 0{,}9636 & -0{,}0401 & 0{,}0018 & 0{,}0120 \\
0{,}0026 & 0{,}0573 & -0{,}0401 & 1{,}0699 & 0{,}1205 & 0{,}0170 \\
-0{,}0173 & 0{,}0081 & 0{,}0018 & 0{,}1205 & 1{,}0100 & -0{,}0056 \\
0{,}0369 & -0{,}0299 & 0{,}0120 & 0{,}0170 & -0{,}0056 & 1{,}0838 \\
0{,}0010 & 0{,}0881 & 0{,}0042 & 0{,}0134 & -0{,}0089 & 0{,}0067 \\
0{,}0629 & 0{,}0288 & -0{,}0209 & 0{,}0257 & 0{,}0342 & -0{,}0021 \\
-0{,}0153 & 0{,}0424 & 0{,}0242 & -0{,}0330 & 0{,}0011 & -0{,}0253 \\
0{,}0361 & 0{,}0597 & 0{,}0199 & 0{,}0992 & 0{,}0119 & 0{,}0339 \\
0{,}0018 & -0{,}0115 & 0{,}0240 & 0{,}0458 & 0{,}0094 & 0{,}0297 \\
0{,}0471 & 0{,}0510 & -0{,}0038 & 0{,}0093 & 0{,}0219 & 0{,}0626 \\
0{,}0469 & -0{,}0135 & -0{,}0239 & 0{,}0104 & 0{,}0329 & 0{,}0271 \\
0{,}0098 & 0{,}0505 & 0{,}0377 & 0{,}0137 & 0{,}0413 & -0{,}0730 \\
-0{,}0313 & -0{,}0106 & 0{,}0011 & 0{,}0332 & -0{,}0130 & 0{,}0354 \\
0{,}0064 & -0{,}0005 & 0{,}0466 & -0{,}0345 & 0{,}0024 & 0{,}0282 \\
-0{,}0542 & 0{,}0193 & -0{,}0385 & 0{,}0327 & -0{,}0460 & -0{,}0026 \\
-0{,}0389 & 0{,}0130 & -0{,}0338 & -0{,}0436 & -0{,}0478 & 0{,}0086
\end{bmatrix}
$$

$$
Q_t^{15} = \begin{bmatrix}
0,0041 & 0,0070 & -0,0208 & 0,0458 & 0,0360 & 0,0106 \\
-0,0029 & 0,0414 & -0,0322 & -0,0003 & 0,0361 & 0,0265 \\
0,0091 & 0,0287 & -0,0180 & 0,0316 & 0,0951 & -0,0414 \\
0,0154 & 0,0097 & -0,0694 & 0,0281 & -0,0551 & 0,0679 \\
-0,0014 & -0,0312 & -0,0519 & -0,0191 & -0,0739 & 0,0450 \\
0,0221 & -0,0571 & 0,0069 & -0,0147 & -0,0746 & 0,0276 \\
0,0069 & -0,0354 & 0,0186 & 0,0413 & 0,0251 & -0,0230 \\
-0,0025 & 0,0055 & -0,0287 & 0,0127 & 0,0326 & -0,0098 \\
0,0155 & -0,0181 & 0,0718 & -0,0310 & 0,0231 & 0,0312 \\
0,0497 & 0,0106 & -0,0245 & 0,0409 & -0,0333 & 0,0292 \\
0,0218 & -0,0044 & 0,0568 & 0,0198 & 0,0573 & 0,0369 \\
0,0110 & 0,0019 & 0,0105 & 0,0719 & 0,0104 & 0,0253 \\
0,0514 & -0,0302 & 0,0013 & 0,0689 & 0,0792 & -0,0335 \\
0,0106 & 0,0172 & -0,0627 & 0,0163 & 0,0357 & 0,0494 \\
0,0213 & -0,0212 & -0,0508 & -0,0267 & -0,0213 & 0,0108 \\
0,0166 & 0,0252 & 0,0248 & 0,0068 & 0,0247 & 0,0298 \\
0,0594 & -0,0307 & -0,0201 & 0,0033 & 0,0459 & -0,0657 \\
0,0698 & -0,0129 & 0,0682 & 0,0406 & 0,0451 & -0,0159
\end{bmatrix}
$$

$$
Q_t^{25} = \begin{bmatrix}
0,0010 & 0,0629 & -0,0153 & 0,0361 & 0,0018 & 0,0471 \\
0,0881 & 0,0288 & 0,0424 & 0,0597 & -0,0115 & 0,0510 \\
0,0042 & -0,0209 & 0,0242 & 0,0199 & 0,0240 & -0,0038 \\
0,0134 & 0,0257 & -0,0330 & 0,0992 & 0,0458 & 0,0093 \\
-0,0089 & 0,0342 & 0,0011 & 0,0119 & 0,0094 & 0,0219 \\
0,0067 & -0,0021 & -0,0253 & 0,0339 & 0,0297 & 0,0626 \\
1,0212 & 0,0163 & 0,0313 & 0,0270 & 0,0453 & -0,0748 \\
0,0163 & 1,0562 & -0,0454 & 0,0533 & -0,0494 & -0,0254 \\
0,0313 & -0,0454 & 1,0462 & 0,0079 & 0,0014 & -0,0304 \\
0,0270 & 0,0533 & 0,0079 & 1,0431 & 0,0144 & 0,0866 \\
0,0453 & -0,0494 & 0,0014 & 0,0144 & 1,0134 & 0,0083 \\
-0,0748 & -0,0254 & -0,0304 & 0,0866 & 0,0083 & 1,0315 \\
-0,0380 & 0,0132 & -0,0383 & 0,0266 & 0,0089 & 0,0249 \\
0,0169 & 0,0204 & 0,0889 & 0,0177 & -0,0655 & 0,0276 \\
0,0355 & -0,0224 & -0,0311 & 0,0212 & 0,0120 & 0,0216 \\
0,0650 & -0,0239 & -0,0249 & -0,0105 & -0,0178 & 0,0012 \\
0,0289 & 0,0645 & 0,0054 & 0,0860 & 0,0290 & 0,0505 \\
0,0776 & -0,0327 & 0,0007 & 0,0170 & 0,0129 & 0,0284
\end{bmatrix}
$$

$$Q_t^{16} = \begin{bmatrix} 0,0534 & -0,0055 & 0,0502 & -0,0461 & -0,0245 & -0,0062 \\ 0,0326 & -0,0193 & -0,0772 & 0,0227 & 0,0202 & 0,0083 \\ 0,0091 & 0,0005 & -0,0034 & -0,0573 & 0,0321 & -0,0703 \\ -0,0894 & -0,0068 & -0,0033 & 0,0468 & 0,0255 & -0,0139 \\ 0,1103 & 0,0499 & 0,0018 & 0,0588 & -0,0053 & 0,0066 \\ -0,0548 & 0,0181 & -0,0184 & 0,0756 & -0,0118 & -0,0106 \\ 0,0409 & -0,0222 & 0,0546 & -0,0213 & -0,0505 & -0,0018 \\ -0,0082 & 0,0016 & 0,0461 & -0,0066 & 0,0233 & -0,0332 \\ 0,0127 & -0,0126 & 0,0502 & 0,0140 & -0,0319 & 0,0710 \\ 0,0081 & 0,0060 & 0,0122 & -0,0160 & 0,0225 & 0,0556 \\ -0,0663 & -0,0258 & 0,0318 & -0,0219 & 0,0121 & -0,0547 \\ -0,0367 & 0,0066 & 0,0014 & -0,0578 & 0,0534 & -0,0238 \\ 0,0394 & -0,0623 & 0,0131 & -0,0132 & -0,0124 & -0,0055 \\ -0,0099 & 0,0390 & -0,0083 & 0,0163 & 0,0549 & -0,0751 \\ -0,0073 & 0,0422 & 0,0251 & 0,0044 & 0,0198 & -0,0214 \\ -0,0234 & 0,0157 & -0,0098 & 0,0031 & 0,0447 & 0,0236 \\ 0,0260 & -0,0051 & -0,0057 & -0,0370 & 0,0190 & 0,0669 \\ 0,0036 & 0,0086 & -0,0219 & -0,0270 & -0,0038 & 0,0359 \end{bmatrix}$$

$$Q_t^{26} = \begin{bmatrix} 0,0469 & 0,0098 & -0,0313 & 0,0064 & -0,0542 & -0,0389 \\ -0,0135 & 0,0505 & -0,0106 & -0,0005 & 0,0193 & 0,0130 \\ -0,0239 & 0,0377 & 0,0011 & 0,0466 & -0,0385 & -0,0338 \\ 0,0104 & 0,0137 & 0,0332 & -0,0345 & 0,0327 & -0,0436 \\ 0,0329 & 0,0413 & -0,0130 & 0,0024 & -0,0460 & -0,0478 \\ 0,0271 & -0,0730 & 0,0354 & 0,0282 & -0,0026 & 0,0086 \\ -0,0380 & 0,0169 & 0,0355 & 0,0650 & 0,0289 & 0,0776 \\ 0,0132 & 0,0204 & -0,0224 & -0,0239 & 0,0645 & -0,0327 \\ -0,0383 & 0,0889 & -0,0311 & -0,0249 & 0,0054 & 0,0007 \\ 0,0266 & 0,0177 & 0,0212 & -0,0105 & 0,0860 & 0,0170 \\ 0,0089 & -0,0655 & 0,0120 & -0,0178 & 0,0290 & 0,0129 \\ 0,0249 & 0,0276 & 0,0216 & 0,0012 & 0,0505 & 0,0284 \\ 0,9889 & 0,0058 & 0,0186 & -0,0363 & 0,0179 & 0,0854 \\ 0,0058 & 0,9875 & 0,0640 & -0,0112 & -0,0298 & -0,0766 \\ 0,0186 & 0,0640 & 0,9967 & 0,0225 & 0,0284 & 0,0180 \\ -0,0363 & -0,0112 & 0,0225 & 1,0914 & 0,0604 & 0,0405 \\ 0,0179 & -0,0298 & 0,0284 & 0,0604 & 1,0139 & 0,0625 \\ 0,0854 & -0,0766 & 0,0180 & 0,0405 & 0,0625 & 1,0493 \end{bmatrix}$$

<u>Matrizen des angepaßten Modells:</u>

$$T_t^{11} = \begin{bmatrix} 0{,}7473 & 0{,}0218 & -0{,}0040 & 0{,}0057 & -0{,}0064 & 0{,}0471 \\ -0{,}0034 & 0{,}7149 & -0{,}0050 & -0{,}0558 & 0{,}0413 & 0{,}0052 \\ -0{,}0191 & -0{,}0087 & 0{,}7114 & 0{,}0327 & 0{,}0076 & -0{,}0132 \\ -0{,}0425 & -0{,}0158 & 0{,}0207 & 0{,}7177 & 0{,}0647 & -0{,}0468 \\ 0{,}0086 & 0{,}0151 & 0{,}0010 & 0{,}0004 & 0{,}7460 & -0{,}0064 \\ -0{,}0041 & -0{,}0259 & 0{,}0371 & 0{,}0084 & -0{,}0009 & 0{,}7366 \\ 0{,}0386 & 0{,}0110 & 0{,}0239 & 0{,}0529 & -0{,}0432 & 0{,}0078 \\ 0{,}0204 & -0{,}0065 & -0{,}0017 & 0{,}0076 & 0{,}0171 & -0{,}0001 \\ -0{,}0451 & 0{,}0226 & 0{,}0394 & -0{,}0353 & 0{,}0009 & -0{,}0417 \\ -0{,}0291 & -0{,}0085 & 0{,}0156 & -0{,}0469 & -0{,}0398 & -0{,}0157 \\ 0{,}0253 & -0{,}0183 & 0{,}0066 & -0{,}0112 & -0{,}0086 & 0{,}0039 \\ -0{,}0229 & 0{,}0017 & 0{,}0181 & -0{,}0489 & 0{,}0249 & 0{,}0023 \\ -0{,}0053 & -0{,}0152 & 0{,}0115 & -0{,}0146 & -0{,}0229 & -0{,}0059 \\ -0{,}0161 & -0{,}0256 & 0{,}0407 & -0{,}0258 & -0{,}0518 & 0{,}0365 \\ 0{,}0145 & 0{,}0056 & 0{,}0017 & 0{,}0086 & 0{,}0065 & 0{,}0081 \\ -0{,}0249 & 0{,}0218 & -0{,}0063 & 0{,}0085 & 0{,}0245 & 0{,}0043 \\ 0{,}0194 & -0{,}0046 & 0{,}0146 & 0{,}0046 & 0{,}0045 & 0{,}0080 \\ -0{,}0145 & 0{,}0194 & 0{,}0462 & -0{,}0328 & 0{,}0033 & -0{,}0319 \end{bmatrix}$$

$$T_t^{21} = \begin{bmatrix} 0{,}0128 & 0{,}0450 & 0{,}0456 & 0{,}0223 & 0{,}0093 & -0{,}0080 \\ -0{,}0164 & 0{,}0188 & 0{,}0146 & 0{,}0081 & 0{,}0425 & -0{,}0132 \\ -0{,}0613 & -0{,}0093 & 0{,}0095 & -0{,}0078 & 0{,}0089 & -0{,}0060 \\ -0{,}0125 & 0{,}0457 & 0{,}0211 & -0{,}0126 & -0{,}0202 & 0{,}0068 \\ 0{,}0017 & -0{,}0094 & -0{,}0080 & 0{,}0401 & 0{,}0535 & -0{,}0188 \\ -0{,}0202 & 0{,}0074 & 0{,}0058 & 0{,}0591 & 0{,}0122 & 0{,}0120 \\ -0{,}0266 & -0{,}0323 & -0{,}0364 & -0{,}0274 & 0{,}0269 & -0{,}0625 \\ -0{,}0212 & 0{,}0149 & 0{,}0249 & 0{,}0041 & -0{,}0071 & -0{,}0001 \\ 0{,}0321 & 0{,}0094 & -0{,}0018 & -0{,}0323 & -0{,}0021 & 0{,}0015 \\ 0{,}0331 & -0{,}0054 & 0{,}0080 & -0{,}0176 & 0{,}0139 & -0{,}0132 \\ -0{,}0029 & -0{,}0391 & -0{,}0181 & 0{,}0206 & 0{,}0056 & 0{,}0330 \\ 0{,}0252 & -0{,}0151 & 0{,}0111 & 0{,}0023 & 0{,}0172 & 0{,}0027 \\ 0{,}0062 & -0{,}0077 & 0{,}0029 & 0{,}0061 & 0{,}0153 & 0{,}0260 \\ -0{,}0226 & -0{,}0061 & 0{,}0042 & -0{,}0377 & -0{,}0294 & -0{,}0083 \\ 0{,}0424 & -0{,}0283 & -0{,}0608 & 0{,}0562 & 0{,}0195 & -0{,}0066 \\ -0{,}0132 & -0{,}0323 & -0{,}0138 & 0{,}0074 & -0{,}0038 & -0{,}0096 \\ 0{,}0299 & -0{,}0134 & 0{,}0048 & 0{,}0436 & -0{,}0695 & 0{,}0235 \\ 0{,}0089 & -0{,}0185 & -0{,}0158 & 0{,}0401 & 0{,}0005 & -0{,}0227 \end{bmatrix}$$

$$T_t^{12} = \begin{bmatrix} 0{,}0070 & -0{,}0013 & 0{,}0398 & 0{,}0215 & -0{,}0108 & 0{,}0030 \\ -0{,}0035 & 0{,}0031 & -0{,}0049 & 0{,}0195 & -0{,}0095 & -0{,}0111 \\ -0{,}0162 & -0{,}0012 & -0{,}0339 & -0{,}0204 & -0{,}0285 & 0{,}0317 \\ 0{,}0012 & -0{,}0148 & 0{,}0160 & 0{,}0313 & 0{,}0044 & -0{,}0015 \\ -0{,}0172 & -0{,}0277 & -0{,}0105 & 0{,}0304 & 0{,}0081 & -0{,}0180 \\ -0{,}0225 & -0{,}0120 & 0{,}0417 & -0{,}0225 & -0{,}0076 & 0{,}0029 \\ 0{,}7569 & -0{,}0203 & -0{,}0033 & -0{,}0552 & -0{,}0148 & -0{,}0210 \\ 0{,}0034 & 0{,}7816 & -0{,}0067 & 0{,}0320 & -0{,}0068 & 0{,}0263 \\ 0{,}0042 & -0{,}0100 & 0{,}7826 & -0{,}0203 & 0{,}0070 & -0{,}0078 \\ -0{,}0179 & -0{,}0164 & -0{,}0007 & 0{,}7542 & 0{,}0073 & 0{,}0308 \\ -0{,}0071 & -0{,}0410 & 0{,}0015 & -0{,}0247 & 0{,}7715 & -0{,}0035 \\ 0{,}0231 & -0{,}0265 & 0{,}0113 & 0{,}0049 & 0{,}0010 & 0{,}7372 \\ -0{,}0302 & 0{,}0081 & -0{,}0058 & 0{,}0237 & -0{,}0462 & 0{,}0277 \\ -0{,}0074 & 0{,}0169 & -0{,}0397 & -0{,}0242 & 0{,}0082 & 0{,}0233 \\ -0{,}0017 & -0{,}0239 & -0{,}0160 & 0{,}0240 & 0{,}0059 & 0{,}0348 \\ 0{,}0308 & 0{,}0175 & -0{,}0040 & 0{,}0084 & -0{,}0090 & -0{,}0015 \\ 0{,}0026 & 0{,}0020 & -0{,}0111 & -0{,}0095 & -0{,}0180 & 0{,}0225 \\ -0{,}0253 & -0{,}0351 & -0{,}0074 & 0{,}0135 & 0{,}0024 & 0{,}0004 \end{bmatrix}$$

$$T_t^{22} = \begin{bmatrix} 0{,}0090 & 0{,}0112 & 0{,}0270 & -0{,}0017 & -0{,}0292 & 0{,}0405 \\ -0{,}0075 & 0{,}0061 & 0{,}0012 & 0{,}0250 & -0{,}0149 & -0{,}0523 \\ 0{,}0123 & 0{,}0102 & 0{,}0090 & -0{,}0145 & -0{,}0031 & 0{,}0076 \\ 0{,}0123 & 0{,}0059 & 0{,}0292 & -0{,}0255 & -0{,}0497 & -0{,}0060 \\ -0{,}0060 & -0{,}0032 & 0{,}0511 & 0{,}0202 & 0{,}0015 & 0{,}0055 \\ 0{,}0150 & 0{,}0184 & -0{,}0264 & -0{,}0236 & -0{,}0042 & 0{,}0099 \\ -0{,}0147 & 0{,}0310 & 0{,}0103 & -0{,}0298 & -0{,}0263 & 0{,}0136 \\ -0{,}0058 & 0{,}0096 & -0{,}0108 & -0{,}0235 & -0{,}0111 & 0{,}0176 \\ -0{,}0067 & 0{,}0167 & -0{,}0043 & -0{,}0264 & -0{,}0168 & 0{,}0103 \\ 0{,}0007 & -0{,}0199 & -0{,}0103 & -0{,}0184 & -0{,}0343 & 0{,}0176 \\ 0{,}0060 & -0{,}0092 & 0{,}0043 & -0{,}0095 & -0{,}0400 & 0{,}0530 \\ 0{,}0441 & 0{,}0063 & 0{,}0181 & 0{,}0043 & -0{,}0128 & -0{,}0243 \\ -0{,}0021 & -0{,}0045 & -0{,}0060 & -0{,}0187 & -0{,}0237 & 0{,}0013 \\ -0{,}0171 & 0{,}0360 & -0{,}0338 & -0{,}0108 & -0{,}0250 & 0{,}0191 \\ -0{,}0364 & 0{,}0174 & -0{,}0686 & 0{,}0133 & -0{,}0058 & 0{,}0195 \\ -0{,}0117 & 0{,}0274 & -0{,}0053 & 0{,}0117 & 0{,}0102 & 0{,}0610 \\ -0{,}0079 & -0{,}0088 & -0{,}0191 & -0{,}0144 & -0{,}0007 & -0{,}0116 \\ 0{,}0118 & -0{,}0017 & -0{,}0034 & -0{,}0157 & -0{,}0543 & 0{,}0404 \end{bmatrix}$$

$$T_t^{13} = \begin{bmatrix} 0,0029 & 0,0235 & -0,0120 & 0,0201 & 0,0042 & -0,0183 \\ -0,0330 & 0,0472 & -0,0034 & 0,0054 & -0,0021 & -0,0493 \\ -0,0194 & 0,0274 & 0,0084 & 0,0223 & -0,0060 & -0,0108 \\ 0,0039 & 0,0362 & 0,0021 & 0,0125 & 0,0409 & -0,0334 \\ 0,0132 & 0,0222 & -0,0219 & -0,0090 & -0,0157 & -0,0119 \\ -0,0054 & 0,0123 & 0,0155 & 0,0199 & 0,0123 & -0,0289 \\ 0,0196 & -0,0054 & 0,0242 & -0,0196 & 0,0307 & -0,0056 \\ -0,0122 & -0,0226 & -0,0050 & 0,0062 & 0,0136 & -0,0109 \\ 0,0197 & 0,0087 & -0,0312 & -0,0246 & 0,0017 & -0,0294 \\ 0,0282 & -0,0170 & 0,0235 & 0,0013 & 0,0029 & -0,0379 \\ -0,0007 & -0,0059 & -0,0072 & -0,0008 & 0,0149 & 0,0161 \\ -0,0027 & 0,0094 & -0,0112 & -0,0134 & -0,0267 & 0,0246 \\ 0,7520 & -0,0070 & 0,0488 & 0,0361 & 0,0177 & -0,0459 \\ -0,0401 & 0,7502 & -0,0173 & -0,0023 & 0,0014 & -0,0138 \\ -0,0316 & 0,0080 & 0,7492 & 0,0130 & 0,0098 & 0,0123 \\ -0,0040 & 0,0428 & -0,0015 & 0,7666 & -0,0256 & -0,0184 \\ -0,0618 & -0,0097 & -0,0124 & -0,0120 & 0,7438 & -0,0060 \\ -0,0035 & -0,0155 & 0,0131 & 0,0123 & 0,0134 & 0,7677 \end{bmatrix}$$

$$T_t^{23} = \begin{bmatrix} 0,0269 & -0,0055 & 0,0132 & -0,0287 & 0,0211 & -0,0457 \\ -0,0156 & -0,0251 & -0,0217 & -0,0009 & 0,0059 & 0,0132 \\ -0,0173 & 0,0169 & -0,0014 & -0,0207 & 0,0009 & 0,0053 \\ -0,0302 & -0,0678 & -0,0051 & -0,0122 & 0,0083 & 0,0321 \\ 0,0180 & 0,0071 & 0,0073 & -0,0074 & -0,0191 & 0,0063 \\ -0,0680 & -0,0287 & -0,0366 & 0,0351 & -0,0321 & 0,0050 \\ 0,0108 & 0,0099 & -0,0033 & -0,0036 & -0,0102 & -0,0097 \\ 0,0298 & -0,0298 & -0,0448 & 0,0225 & -0,0327 & 0,0336 \\ 0,0007 & -0,0127 & 0,0067 & -0,0040 & -0,0005 & 0,0332 \\ -0,0109 & 0,0278 & 0,0234 & 0,0252 & 0,0127 & -0,0147 \\ 0,0168 & 0,0551 & 0,0239 & -0,0058 & -0,0057 & 0,0232 \\ 0,0109 & 0,0179 & -0,0265 & -0,0214 & 0,0013 & 0,0216 \\ -0,0493 & -0,0171 & -0,0082 & -0,0064 & -0,0027 & 0,0234 \\ -0,0149 & 0,0388 & 0,0054 & -0,0073 & -0,0023 & -0,0084 \\ -0,0075 & -0,0400 & -0,0060 & 0,0132 & 0,0067 & 0,0096 \\ -0,0318 & 0,0128 & -0,0042 & -0,0040 & -0,0110 & 0,0265 \\ 0,0275 & -0,0244 & 0,0205 & 0,0367 & 0,0104 & -0,0351 \\ -0,0293 & -0,0011 & -0,0292 & 0,0142 & -0,0054 & 0,0108 \end{bmatrix}$$

$$T_t^{14} = \begin{bmatrix} 0,0249 & 0,0047 & -0,0341 & -0,0073 & 0,0151 & -0,0145 \\ 0,0116 & 0,0019 & -0,0210 & 0,0249 & 0,0137 & -0,0081 \\ 0,0180 & 0,0004 & 0,0001 & 0,0215 & -0,0049 & -0,0078 \\ 0,0019 & 0,0217 & -0,0032 & 0,0414 & -0,0051 & 0,0190 \\ -0,0115 & -0,0109 & 0,0055 & 0,0058 & 0,0458 & 0,0028 \\ 0,0270 & -0,0062 & -0,0332 & -0,0376 & 0,0266 & 0,0024 \\ 0,0386 & -0,0097 & 0,0235 & 0,0319 & 0,0070 & 0,0187 \\ 0,0291 & -0,0281 & 0,0371 & -0,0043 & 0,0407 & 0,0113 \\ 0,0498 & -0,0046 & 0,0329 & 0,0045 & -0,0060 & 0,0083 \\ -0,0373 & 0,0193 & -0,0195 & 0,0250 & -0,0473 & 0,0192 \\ 0,0023 & 0,0161 & 0,0242 & -0,0439 & -0,0326 & 0,0090 \\ -0,0211 & 0,0196 & -0,0328 & 0,0148 & -0,0528 & 0,0052 \\ -0,0024 & -0,0031 & -0,0059 & 0,0045 & 0,0060 & 0,0066 \\ -0,0405 & 0,0181 & 0,0205 & -0,0173 & -0,0539 & -0,0288 \\ 0,0101 & 0,0245 & 0,0196 & 0,0247 & 0,0298 & -0,0179 \\ 0,0074 & 0,0020 & 0,0310 & -0,0098 & 0,0142 & -0,0423 \\ 0,0118 & -0,0150 & -0,0214 & -0,0315 & -0,0196 & -0,0154 \\ 0,0211 & -0,0336 & -0,0354 & -0,0125 & -0,0033 & 0,0087 \end{bmatrix}$$

$$T_t^{24} = \begin{bmatrix} 0,7650 & -0,0226 & -0,0305 & 0,0316 & 0,0045 & 0,0219 \\ -0,0035 & 0,7556 & -0,0012 & -0,0004 & -0,0068 & 0,0135 \\ 0,0102 & -0,0011 & 0,7544 & 0,0122 & -0,0129 & 0,0109 \\ -0,0157 & -0,0026 & 0,0229 & 0,7538 & 0,0274 & -0,0073 \\ -0,0267 & -0,0132 & -0,0343 & 0,0011 & 0,7732 & -0,0043 \\ -0,0236 & 0,0079 & -0,0039 & -0,0229 & -0,0138 & 0,7653 \\ -0,0055 & -0,0215 & -0,0010 & -0,0487 & -0,0164 & 0,0007 \\ -0,0042 & 0,0148 & 0,0150 & 0,0082 & -0,0029 & 0,0043 \\ -0,0001 & -0,0120 & 0,0007 & -0,0292 & -0,0203 & -0,0188 \\ 0,0199 & 0,0145 & 0,0089 & -0,0173 & 0,0379 & 0,0063 \\ -0,0009 & 0,0297 & 0,0182 & -0,0079 & -0,0047 & 0,0182 \\ 0,0107 & -0,0252 & -0,0012 & -0,0144 & 0,0073 & -0,0251 \\ 0,0048 & -0,0279 & 0,0052 & -0,0433 & 0,0167 & 0,0061 \\ -0,0013 & 0,0625 & -0,0067 & -0,0178 & 0,0183 & 0,0666 \\ -0,0081 & 0,0165 & 0,0151 & 0,0012 & -0,0185 & -0,0258 \\ -0,0082 & 0,0465 & -0,0117 & -0,0050 & -0,0037 & -0,0236 \\ 0,0229 & 0,0110 & -0,0045 & 0,0025 & -0,0082 & -0,0085 \\ 0,0014 & -0,0146 & 0,0285 & 0,0349 & -0,0231 & -0,0164 \end{bmatrix}$$

$$T_t^{15} = \begin{bmatrix} -0{,}0199 & 0{,}0403 & 0{,}0057 & -0{,}0036 & -0{,}0087 & -0{,}0056 \\ 0{,}0275 & -0{,}0179 & -0{,}0163 & -0{,}0391 & -0{,}0514 & 0{,}0093 \\ 0{,}0126 & 0{,}0140 & 0{,}0412 & -0{,}0338 & 0{,}0268 & 0{,}0154 \\ -0{,}0111 & 0{,}0086 & 0{,}0073 & -0{,}0115 & 0{,}0056 & -0{,}0023 \\ 0{,}0277 & 0{,}0143 & 0{,}0277 & -0{,}0031 & 0{,}0069 & -0{,}0129 \\ 0{,}0238 & 0{,}0048 & -0{,}0754 & 0{,}0580 & -0{,}0320 & -0{,}0325 \\ -0{,}0077 & -0{,}0006 & -0{,}0093 & -0{,}0022 & 0{,}0008 & -0{,}0098 \\ 0{,}0058 & 0{,}0055 & 0{,}0169 & -0{,}0084 & 0{,}0122 & -0{,}0100 \\ 0{,}0184 & 0{,}0169 & -0{,}0043 & -0{,}0133 & -0{,}0074 & 0{,}0087 \\ 0{,}0312 & 0{,}0009 & -0{,}0235 & -0{,}0095 & 0{,}0247 & -0{,}0080 \\ -0{,}0049 & 0{,}0015 & -0{,}0077 & -0{,}0408 & -0{,}0251 & -0{,}0004 \\ -0{,}0022 & -0{,}0078 & 0{,}0228 & -0{,}0259 & -0{,}0170 & 0{,}0125 \\ -0{,}0498 & 0{,}0041 & 0{,}0253 & 0{,}0333 & -0{,}0029 & -0{,}0178 \\ -0{,}0022 & -0{,}0441 & 0{,}0161 & 0{,}0157 & 0{,}0083 & -0{,}0366 \\ -0{,}0483 & -0{,}0363 & 0{,}0158 & 0{,}0046 & 0{,}0060 & -0{,}0309 \\ -0{,}0116 & 0{,}0227 & 0{,}0152 & -0{,}0110 & -0{,}0008 & 0{,}0147 \\ 0{,}0443 & -0{,}0257 & 0{,}0311 & 0{,}0108 & -0{,}0005 & -0{,}0180 \\ 0{,}0141 & -0{,}0204 & -0{,}0392 & 0{,}0174 & 0{,}0116 & 0{,}0032 \end{bmatrix}$$

$$T_t^{25} = \begin{bmatrix} -0{,}0078 & 0{,}0077 & -0{,}0016 & -0{,}0015 & 0{,}0261 & -0{,}0102 \\ 0{,}0024 & -0{,}0308 & 0{,}0574 & -0{,}0345 & 0{,}0016 & 0{,}0010 \\ -0{,}0124 & 0{,}0311 & -0{,}0126 & 0{,}0137 & -0{,}0001 & -0{,}0445 \\ -0{,}0168 & -0{,}0293 & 0{,}0336 & -0{,}0066 & -0{,}0325 & 0{,}0131 \\ -0{,}0037 & -0{,}0025 & -0{,}0060 & -0{,}0037 & 0{,}0248 & 0{,}0046 \\ -0{,}0053 & 0{,}0143 & 0{,}0039 & 0{,}0530 & 0{,}0290 & 0{,}0348 \\ 0{,}7153 & -0{,}0003 & -0{,}0192 & 0{,}0592 & -0{,}0024 & -0{,}0092 \\ -0{,}0128 & 0{,}7508 & 0{,}0044 & -0{,}0236 & -0{,}0150 & -0{,}0148 \\ 0{,}0333 & -0{,}0182 & 0{,}7228 & 0{,}0217 & -0{,}0100 & 0{,}0047 \\ -0{,}0234 & 0{,}0114 & 0{,}0355 & 0{,}7862 & 0{,}0034 & -0{,}0397 \\ 0{,}0096 & 0{,}0302 & 0{,}0449 & -0{,}0459 & 0{,}7556 & 0{,}0339 \\ 0{,}0091 & -0{,}0012 & -0{,}0246 & -0{,}0032 & -0{,}0311 & 0{,}7589 \\ 0{,}0164 & 0{,}0066 & 0{,}0050 & 0{,}0049 & 0{,}0246 & -0{,}0133 \\ -0{,}0220 & -0{,}0174 & -0{,}0208 & -0{,}0156 & -0{,}0300 & -0{,}0268 \\ 0{,}0485 & -0{,}0588 & -0{,}0044 & 0{,}0219 & 0{,}0074 & 0{,}0288 \\ -0{,}0113 & -0{,}0243 & 0{,}0273 & 0{,}0058 & 0{,}0133 & 0{,}0162 \\ 0{,}0542 & 0{,}0045 & -0{,}0240 & -0{,}0116 & 0{,}0502 & 0{,}0659 \\ -0{,}0266 & -0{,}0455 & 0{,}0451 & -0{,}0258 & -0{,}0013 & 0{,}0226 \end{bmatrix}$$

$$T_t^{16} = \begin{bmatrix} 0{,}0044 & 0{,}0042 & 0{,}0517 & -0{,}0059 & -0{,}0007 & -0{,}0163 \\ 0{,}0115 & -0{,}0066 & 0{,}0053 & 0{,}0054 & 0{,}0196 & -0{,}0279 \\ -0{,}0049 & -0{,}0210 & -0{,}0326 & -0{,}0348 & -0{,}0642 & -0{,}0089 \\ 0{,}0208 & -0{,}0237 & -0{,}0405 & 0{,}0140 & -0{,}0215 & 0{,}0418 \\ 0{,}0246 & -0{,}0019 & -0{,}0197 & 0{,}0167 & -0{,}0101 & -0{,}0126 \\ -0{,}0163 & -0{,}0021 & -0{,}0337 & -0{,}0177 & -0{,}0318 & 0{,}0216 \\ 0{,}0415 & 0{,}0360 & 0{,}0002 & 0{,}0200 & 0{,}0142 & 0{,}0050 \\ 0{,}0125 & -0{,}0234 & -0{,}0326 & 0{,}0308 & -0{,}0301 & 0{,}0323 \\ 0{,}0244 & 0{,}0080 & 0{,}0367 & 0{,}0148 & 0{,}0051 & 0{,}0161 \\ 0{,}0501 & 0{,}0286 & -0{,}0392 & 0{,}0062 & 0{,}0165 & -0{,}0149 \\ -0{,}0048 & 0{,}0217 & 0{,}0044 & -0{,}0047 & 0{,}0253 & -0{,}0308 \\ -0{,}0243 & -0{,}0635 & 0{,}0102 & 0{,}0188 & 0{,}0280 & 0{,}0330 \\ -0{,}0006 & 0{,}0223 & -0{,}0395 & 0{,}0066 & -0{,}0057 & -0{,}0278 \\ 0{,}0021 & -0{,}0033 & -0{,}0312 & 0{,}0247 & 0{,}0108 & 0{,}0166 \\ 0{,}0031 & 0{,}0282 & 0{,}0264 & 0{,}0278 & 0{,}0391 & -0{,}0236 \\ -0{,}0444 & -0{,}0210 & -0{,}0306 & -0{,}0198 & 0{,}0110 & -0{,}0134 \\ -0{,}0153 & 0{,}0102 & 0{,}0022 & 0{,}0153 & -0{,}0009 & 0{,}0024 \\ -0{,}0093 & 0{,}0155 & -0{,}0010 & 0{,}0102 & 0{,}0035 & 0{,}0505 \end{bmatrix}$$

$$T_t^{26} = \begin{bmatrix} -0{,}0047 & 0{,}0217 & -0{,}0304 & 0{,}0078 & -0{,}0224 & 0{,}0201 \\ 0{,}0409 & -0{,}0298 & 0{,}0215 & 0{,}0269 & -0{,}0071 & 0{,}0243 \\ -0{,}0002 & -0{,}0292 & 0{,}0183 & 0{,}0317 & -0{,}0081 & 0{,}0092 \\ 0{,}0013 & -0{,}0079 & 0{,}0381 & 0{,}0075 & 0{,}0201 & 0{,}0103 \\ -0{,}0165 & -0{,}0160 & -0{,}0126 & -0{,}0049 & 0{,}0061 & -0{,}0052 \\ -0{,}0270 & 0{,}0052 & 0{,}0235 & 0{,}0162 & -0{,}0389 & 0{,}0055 \\ 0{,}0015 & -0{,}0100 & -0{,}0066 & -0{,}0028 & 0{,}0497 & -0{,}0011 \\ -0{,}0271 & -0{,}0268 & 0{,}0298 & -0{,}0056 & -0{,}0079 & -0{,}0130 \\ 0{,}0422 & 0{,}0119 & -0{,}0341 & -0{,}0152 & 0{,}0373 & 0{,}0276 \\ 0{,}0032 & 0{,}0299 & -0{,}0251 & 0{,}0207 & -0{,}0048 & -0{,}0167 \\ -0{,}0537 & -0{,}0287 & 0{,}0071 & -0{,}0170 & 0{,}0051 & -0{,}0271 \\ -0{,}0189 & 0{,}0221 & -0{,}0013 & 0{,}0182 & 0{,}0090 & -0{,}0099 \\ 0{,}7617 & 0{,}0166 & 0{,}0182 & 0{,}0216 & 0{,}0032 & -0{,}0064 \\ -0{,}0434 & 0{,}7651 & -0{,}0299 & -0{,}0119 & 0{,}0233 & 0{,}0273 \\ 0{,}0146 & 0{,}0074 & 0{,}7175 & -0{,}0426 & 0{,}0224 & -0{,}0204 \\ -0{,}0218 & 0{,}0285 & -0{,}0296 & 0{,}7469 & -0{,}0164 & -0{,}0234 \\ 0{,}0125 & 0{,}0126 & 0{,}0022 & 0{,}0249 & 0{,}7457 & 0{,}0023 \\ -0{,}0037 & 0{,}0329 & -0{,}0221 & -0{,}0069 & 0{,}0334 & 0{,}7325 \end{bmatrix}$$

$$Q_t^{11} = \begin{bmatrix} 0,9914 & 0,0078 & 0,0239 & -0,0617 & -0,0020 & -0,0461 \\ 0,0078 & 1,0079 & 0,0172 & 0,0301 & 0,0136 & -0,0217 \\ 0,0239 & 0,0172 & 1,0510 & 0,0011 & 0,0428 & -0,0433 \\ -0,0617 & 0,0301 & 0,0011 & 1,0052 & 0,0339 & 0,0516 \\ -0,0020 & 0,0136 & 0,0428 & 0,0339 & 1,0420 & 0,0428 \\ -0,0461 & -0,0217 & -0,0433 & 0,0516 & 0,0428 & 0,9690 \\ -0,0342 & -0,0026 & -0,0130 & 0,0264 & 0,0132 & 0,0278 \\ -0,0241 & 0,0312 & 0,0366 & -0,0237 & 0,0016 & 0,0138 \\ 0,0349 & -0,0073 & -0,0206 & -0,0350 & -0,0614 & 0,0290 \\ 0,0385 & -0,0358 & 0,0585 & 0,0118 & -0,0066 & -0,0053 \\ 0,0213 & 0,0462 & 0,0261 & -0,0229 & 0,0014 & 0,0002 \\ -0,0140 & -0,0407 & 0,0215 & -0,0115 & -0,0070 & 0,0217 \\ 0,0177 & 0,0574 & 0,0981 & -0,0070 & 0,0077 & -0,0520 \\ 0,0256 & 0,0598 & -0,0541 & 0,0361 & -0,0055 & 0,0044 \\ 0,0753 & -0,0173 & 0,0397 & -0,0046 & 0,0323 & 0,0212 \\ -0,0045 & -0,0491 & -0,0523 & 0,0203 & 0,0083 & 0,0339 \\ 0,0236 & -0,0126 & 0,0475 & -0,0050 & 0,0386 & 0,0107 \\ -0,0236 & -0,0041 & 0,0882 & -0,0610 & 0,0156 & 0,0775 \end{bmatrix}$$

$$Q_t^{21} = \begin{bmatrix} -0,0119 & 0,0502 & 0,0278 & -0,0158 & 0,0365 & 0,0633 \\ -0,0302 & -0,0048 & -0,0236 & 0,0489 & 0,0086 & 0,0412 \\ -0,0499 & -0,0012 & 0,0358 & 0,0351 & -0,0041 & -0,0050 \\ 0,0391 & 0,0672 & 0,0317 & 0,0336 & 0,0338 & 0,0529 \\ -0,0445 & -0,0419 & -0,0359 & -0,0199 & 0,0249 & -0,0209 \\ 0,0045 & 0,0159 & -0,0508 & -0,0280 & -0,0707 & -0,0126 \\ 0,0041 & -0,0029 & 0,0091 & 0,0154 & -0,0014 & 0,0221 \\ 0,0070 & 0,0414 & 0,0287 & 0,0097 & -0,0312 & -0,0571 \\ -0,0208 & -0,0322 & -0,0180 & -0,0694 & -0,0519 & 0,0069 \\ 0,0458 & -0,0003 & 0,0316 & 0,0281 & -0,0191 & -0,0147 \\ 0,0360 & 0,0361 & 0,0951 & -0,0551 & -0,0739 & -0,0746 \\ 0,0106 & 0,0265 & -0,0414 & 0,0679 & 0,0450 & 0,0276 \\ 0,0534 & 0,0326 & 0,0091 & -0,0894 & 0,1103 & -0,0548 \\ -0,0055 & -0,0193 & 0,0005 & -0,0068 & 0,0499 & 0,0181 \\ 0,0502 & -0,0772 & -0,0034 & -0,0033 & 0,0018 & -0,0184 \\ -0,0461 & 0,0227 & -0,0573 & 0,0468 & 0,0588 & 0,0756 \\ -0,0245 & 0,0202 & 0,0321 & 0,0255 & -0,0053 & -0,0118 \\ -0,0062 & 0,0083 & -0,0703 & -0,0139 & 0,0066 & -0,0106 \end{bmatrix}$$

$$Q_t^{12} = \begin{bmatrix} -0,0342 & -0,0241 & 0,0349 & 0,0385 & 0,0213 & -0,0140 \\ -0,0026 & 0,0312 & -0,0073 & -0,0358 & 0,0462 & -0,0407 \\ -0,0130 & 0,0366 & -0,0206 & 0,0585 & 0,0261 & 0,0215 \\ 0,0264 & -0,0237 & -0,0350 & 0,0118 & -0,0229 & -0,0115 \\ 0,0132 & 0,0016 & -0,0614 & -0,0066 & 0,0014 & -0,0070 \\ 0,0278 & 0,0138 & 0,0290 & -0,0053 & 0,0002 & 0,0217 \\ 1,0255 & 0,0276 & 0,0662 & -0,0238 & -0,0163 & 0,0408 \\ 0,0276 & 1,0416 & -0,0224 & -0,0301 & -0,0393 & -0,0406 \\ 0,0662 & -0,0224 & 1,0111 & -0,0348 & 0,0116 & -0,0393 \\ -0,0238 & -0,0301 & -0,0348 & 1,0374 & -0,0084 & 0,1096 \\ -0,0163 & -0,0393 & 0,0116 & -0,0084 & 1,0385 & -0,0563 \\ 0,0408 & -0,0406 & -0,0393 & 0,1096 & -0,0563 & 1,1175 \\ -0,0075 & -0,0342 & -0,0300 & 0,0153 & 0,0097 & 0,0721 \\ -0,0201 & -0,0311 & -0,0487 & -0,0007 & -0,0427 & -0,0142 \\ 0,0433 & 0,0341 & 0,0171 & 0,0415 & -0,0552 & -0,0276 \\ -0,0033 & 0,0399 & -0,0312 & 0,0434 & 0,0108 & 0,0133 \\ -0,0173 & 0,0325 & -0,0052 & 0,0375 & 0,0150 & 0,0355 \\ 0,0382 & 0,0003 & 0,0123 & 0,0574 & -0,0419 & -0,0036 \end{bmatrix}$$

$$Q_t^{22} = \begin{bmatrix} -0,0342 & -0,0241 & 0,0349 & 0,0385 & 0,0213 & -0,0140 \\ -0,0026 & 0,0312 & -0,0073 & -0,0358 & 0,0462 & -0,0407 \\ -0,0130 & 0,0366 & -0,0206 & 0,0585 & 0,0261 & 0,0215 \\ 0,0264 & -0,0237 & -0,0350 & 0,0118 & -0,0229 & -0,0115 \\ 0,0132 & 0,0016 & -0,0614 & -0,0066 & 0,0014 & -0,0070 \\ 0,0278 & 0,0138 & 0,0290 & -0,0053 & 0,0002 & 0,0217 \\ 1,0255 & 0,0276 & 0,0662 & -0,0238 & -0,0163 & 0,0408 \\ 0,0276 & 1,0416 & -0,0224 & -0,0301 & -0,0393 & -0,0406 \\ 0,0662 & -0,0224 & 1,0111 & -0,0348 & 0,0116 & -0,0393 \\ -0,0238 & -0,0301 & -0,0348 & 1,0374 & -0,0084 & 0,1096 \\ -0,0163 & -0,0393 & 0,0116 & -0,0084 & 1,0385 & -0,0563 \\ 0,0408 & -0,0406 & -0,0393 & 0,1096 & -0,0563 & 1,1175 \\ -0,0075 & -0,0342 & -0,0300 & 0,0153 & 0,0097 & 0,0721 \\ -0,0201 & -0,0311 & -0,0487 & -0,0007 & -0,0427 & -0,0142 \\ 0,0433 & 0,0341 & 0,0171 & 0,0415 & -0,0552 & -0,0276 \\ -0,0033 & 0,0399 & -0,0312 & 0,0434 & 0,0108 & 0,0133 \\ -0,0173 & 0,0325 & -0,0052 & 0,0375 & 0,0150 & 0,0355 \\ 0,0382 & 0,0003 & 0,0123 & 0,0574 & -0,0419 & -0,0036 \end{bmatrix}$$

$$Q_t^{13} = \begin{bmatrix} 0{,}0177 & 0{,}0256 & 0{,}0753 & -0{,}0045 & 0{,}0236 & -0{,}0236 \\ 0{,}0574 & 0{,}0598 & -0{,}0173 & -0{,}0491 & -0{,}0126 & -0{,}0041 \\ 0{,}0981 & -0{,}0541 & 0{,}0397 & -0{,}0523 & 0{,}0475 & 0{,}0882 \\ -0{,}0070 & 0{,}0361 & -0{,}0046 & 0{,}0203 & -0{,}0050 & -0{,}0610 \\ 0{,}0077 & -0{,}0055 & 0{,}0323 & 0{,}0083 & 0{,}0386 & 0{,}0156 \\ -0{,}0520 & 0{,}0044 & 0{,}0212 & 0{,}0339 & 0{,}0107 & 0{,}0775 \\ -0{,}0075 & -0{,}0201 & 0{,}0433 & -0{,}0033 & -0{,}0173 & 0{,}0382 \\ -0{,}0342 & -0{,}0311 & 0{,}0341 & 0{,}0399 & 0{,}0325 & 0{,}0003 \\ -0{,}0300 & -0{,}0487 & 0{,}0171 & -0{,}0312 & -0{,}0052 & 0{,}0123 \\ 0{,}0153 & -0{,}0007 & 0{,}0415 & 0{,}0434 & 0{,}0375 & 0{,}0574 \\ 0{,}0097 & -0{,}0427 & -0{,}0552 & 0{,}0108 & 0{,}0150 & -0{,}0419 \\ 0{,}0721 & -0{,}0142 & -0{,}0276 & 0{,}0133 & 0{,}0355 & -0{,}0036 \\ 0{,}9898 & 0{,}0199 & -0{,}0382 & 0{,}0182 & -0{,}0169 & -0{,}0041 \\ 0{,}0199 & 1{,}0964 & 0{,}0461 & -0{,}0049 & -0{,}0117 & -0{,}0085 \\ -0{,}0382 & 0{,}0461 & 0{,}9115 & 0{,}0570 & -0{,}0292 & -0{,}0256 \\ 0{,}0182 & -0{,}0049 & 0{,}0570 & 0{,}9716 & 0{,}0773 & -0{,}0604 \\ -0{,}0169 & -0{,}0117 & -0{,}0292 & 0{,}0773 & 0{,}9395 & 0{,}0275 \\ -0{,}0041 & -0{,}0085 & -0{,}0256 & -0{,}0604 & 0{,}0275 & 1{,}1981 \end{bmatrix}$$

$$Q_t^{23} = \begin{bmatrix} -0{,}0037 & -0{,}0295 & -0{,}0493 & -0{,}0806 & -0{,}0407 & 0{,}0273 \\ 0{,}0093 & 0{,}0263 & -0{,}0386 & 0{,}0213 & 0{,}0355 & 0{,}0163 \\ -0{,}0301 & 0{,}0462 & 0{,}0234 & 0{,}0156 & 0{,}0262 & -0{,}0360 \\ 0{,}0652 & 0{,}0769 & 0{,}0018 & 0{,}0092 & 0{,}0376 & 0{,}0401 \\ 0{,}0510 & 0{,}0613 & 0{,}0176 & 0{,}0050 & -0{,}0024 & 0{,}0052 \\ 0{,}0079 & -0{,}0057 & -0{,}0158 & -0{,}0076 & -0{,}0187 & -0{,}0086 \\ 0{,}0514 & 0{,}0106 & 0{,}0213 & 0{,}0166 & 0{,}0594 & 0{,}0698 \\ -0{,}0302 & 0{,}0172 & -0{,}0212 & 0{,}0252 & -0{,}0307 & -0{,}0129 \\ 0{,}0013 & -0{,}0627 & -0{,}0508 & 0{,}0248 & -0{,}0201 & 0{,}0682 \\ 0{,}0689 & 0{,}0163 & -0{,}0267 & 0{,}0068 & 0{,}0033 & 0{,}0406 \\ 0{,}0792 & 0{,}0357 & -0{,}0213 & 0{,}0247 & 0{,}0459 & 0{,}0451 \\ -0{,}0335 & 0{,}0494 & 0{,}0108 & 0{,}0298 & -0{,}0657 & -0{,}0159 \\ 0{,}0394 & -0{,}0099 & -0{,}0073 & -0{,}0234 & 0{,}0260 & 0{,}0036 \\ -0{,}0623 & 0{,}0390 & 0{,}0422 & 0{,}0157 & -0{,}0051 & 0{,}0086 \\ 0{,}0131 & -0{,}0083 & 0{,}0251 & -0{,}0098 & -0{,}0057 & -0{,}0219 \\ -0{,}0132 & 0{,}0163 & 0{,}0044 & 0{,}0031 & -0{,}0370 & -0{,}0270 \\ -0{,}0124 & 0{,}0549 & 0{,}0198 & 0{,}0447 & 0{,}0190 & -0{,}0038 \\ -0{,}0055 & -0{,}0751 & -0{,}0214 & 0{,}0236 & 0{,}0669 & 0{,}0359 \end{bmatrix}$$

$$Q_t^{14} = \begin{bmatrix} -0{,}0119 & -0{,}0302 & -0{,}0499 & 0{,}0391 & -0{,}0445 & 0{,}0045 \\ 0{,}0502 & -0{,}0048 & -0{,}0012 & 0{,}0672 & -0{,}0419 & 0{,}0159 \\ 0{,}0278 & -0{,}0236 & 0{,}0358 & 0{,}0317 & -0{,}0359 & -0{,}0508 \\ -0{,}0158 & 0{,}0489 & 0{,}0351 & 0{,}0336 & -0{,}0199 & -0{,}0280 \\ 0{,}0365 & 0{,}0086 & -0{,}0041 & 0{,}0338 & 0{,}0249 & -0{,}0707 \\ 0{,}0633 & 0{,}0412 & -0{,}0050 & 0{,}0529 & -0{,}0209 & -0{,}0126 \\ -0{,}0029 & -0{,}0153 & -0{,}0270 & 0{,}0089 & 0{,}0745 & -0{,}0372 \\ 0{,}0234 & 0{,}0309 & -0{,}0490 & -0{,}0377 & -0{,}0036 & 0{,}0609 \\ 0{,}0304 & 0{,}0197 & -0{,}0012 & 0{,}0347 & 0{,}0346 & -0{,}0049 \\ 0{,}0050 & 0{,}0079 & -0{,}0161 & 0{,}0438 & -0{,}0241 & -0{,}0183 \\ 0{,}0338 & -0{,}0095 & -0{,}0293 & 0{,}0632 & 0{,}0168 & 0{,}0603 \\ 0{,}0273 & 0{,}0646 & 0{,}0271 & 0{,}0098 & -0{,}0147 & 0{,}0261 \\ -0{,}0037 & 0{,}0093 & -0{,}0301 & 0{,}0652 & 0{,}0510 & 0{,}0079 \\ -0{,}0295 & 0{,}0263 & 0{,}0462 & 0{,}0769 & 0{,}0613 & -0{,}0057 \\ -0{,}0493 & -0{,}0386 & 0{,}0234 & 0{,}0018 & 0{,}0176 & -0{,}0158 \\ -0{,}0806 & 0{,}0213 & 0{,}0156 & 0{,}0092 & 0{,}0050 & -0{,}0076 \\ -0{,}0407 & 0{,}0355 & 0{,}0262 & 0{,}0376 & -0{,}0024 & -0{,}0187 \\ 0{,}0273 & 0{,}0163 & -0{,}0360 & 0{,}0401 & 0{,}0052 & -0{,}0086 \end{bmatrix}$$

$$Q_t^{24} = \begin{bmatrix} 1{,}0707 & -0{,}0023 & 0{,}0344 & 0{,}0026 & -0{,}0173 & 0{,}0369 \\ -0{,}0023 & 0{,}9689 & -0{,}0109 & 0{,}0573 & 0{,}0081 & -0{,}0299 \\ 0{,}0344 & -0{,}0109 & 0{,}9636 & -0{,}0401 & 0{,}0018 & 0{,}0120 \\ 0{,}0026 & 0{,}0573 & -0{,}0401 & 1{,}0699 & 0{,}1205 & 0{,}0170 \\ -0{,}0173 & 0{,}0081 & 0{,}0018 & 0{,}1205 & 1{,}0100 & -0{,}0056 \\ 0{,}0369 & -0{,}0299 & 0{,}0120 & 0{,}0170 & -0{,}0056 & 1{,}0838 \\ 0{,}0010 & 0{,}0881 & 0{,}0042 & 0{,}0134 & -0{,}0089 & 0{,}0067 \\ 0{,}0629 & 0{,}0288 & -0{,}0209 & 0{,}0257 & 0{,}0342 & -0{,}0021 \\ -0{,}0153 & 0{,}0424 & 0{,}0242 & -0{,}0330 & 0{,}0011 & -0{,}0253 \\ 0{,}0361 & 0{,}0597 & 0{,}0199 & 0{,}0992 & 0{,}0119 & 0{,}0339 \\ 0{,}0018 & -0{,}0115 & 0{,}0240 & 0{,}0458 & 0{,}0094 & 0{,}0297 \\ 0{,}0471 & 0{,}0510 & -0{,}0038 & 0{,}0093 & 0{,}0219 & 0{,}0626 \\ 0{,}0469 & -0{,}0135 & -0{,}0239 & 0{,}0104 & 0{,}0329 & 0{,}0271 \\ 0{,}0098 & 0{,}0505 & 0{,}0377 & 0{,}0137 & 0{,}0413 & -0{,}0730 \\ -0{,}0313 & -0{,}0106 & 0{,}0011 & 0{,}0332 & -0{,}0130 & 0{,}0354 \\ 0{,}0064 & -0{,}0005 & 0{,}0466 & -0{,}0345 & 0{,}0024 & 0{,}0282 \\ -0{,}0542 & 0{,}0193 & -0{,}0385 & 0{,}0327 & -0{,}0460 & -0{,}0026 \\ -0{,}0389 & 0{,}0130 & -0{,}0338 & -0{,}0436 & -0{,}0478 & 0{,}0086 \end{bmatrix}$$

$$Q_t^{15} = \begin{bmatrix} 0{,}0041 & 0{,}0070 & -0{,}0208 & 0{,}0458 & 0{,}0360 & 0{,}0106 \\ -0{,}0029 & 0{,}0414 & -0{,}0322 & -0{,}0003 & 0{,}0361 & 0{,}0265 \\ 0{,}0091 & 0{,}0287 & -0{,}0180 & 0{,}0316 & 0{,}0951 & -0{,}0414 \\ 0{,}0154 & 0{,}0097 & -0{,}0694 & 0{,}0281 & -0{,}0551 & 0{,}0679 \\ -0{,}0014 & -0{,}0312 & -0{,}0519 & -0{,}0191 & -0{,}0739 & 0{,}0450 \\ 0{,}0221 & -0{,}0571 & 0{,}0069 & -0{,}0147 & -0{,}0746 & 0{,}0276 \\ 0{,}0069 & -0{,}0354 & 0{,}0186 & 0{,}0413 & 0{,}0251 & -0{,}0230 \\ -0{,}0025 & 0{,}0055 & -0{,}0287 & 0{,}0127 & 0{,}0326 & -0{,}0098 \\ 0{,}0155 & -0{,}0181 & 0{,}0718 & -0{,}0310 & 0{,}0231 & 0{,}0312 \\ 0{,}0497 & 0{,}0106 & -0{,}0245 & 0{,}0409 & -0{,}0333 & 0{,}0292 \\ 0{,}0218 & -0{,}0044 & 0{,}0568 & 0{,}0198 & 0{,}0573 & 0{,}0369 \\ 0{,}0110 & 0{,}0019 & 0{,}0105 & 0{,}0719 & 0{,}0104 & 0{,}0253 \\ 0{,}0514 & -0{,}0302 & 0{,}0013 & 0{,}0689 & 0{,}0792 & -0{,}0335 \\ 0{,}0106 & 0{,}0172 & -0{,}0627 & 0{,}0163 & 0{,}0357 & 0{,}0494 \\ 0{,}0213 & -0{,}0212 & -0{,}0508 & -0{,}0267 & -0{,}0213 & 0{,}0108 \\ 0{,}0166 & 0{,}0252 & 0{,}0248 & 0{,}0068 & 0{,}0247 & 0{,}0298 \\ 0{,}0594 & -0{,}0307 & -0{,}0201 & 0{,}0033 & 0{,}0459 & -0{,}0657 \\ 0{,}0698 & -0{,}0129 & 0{,}0682 & 0{,}0406 & 0{,}0451 & -0{,}0159 \end{bmatrix}$$

$$Q_t^{25} = \begin{bmatrix} 0{,}0010 & 0{,}0629 & -0{,}0153 & 0{,}0361 & 0{,}0018 & 0{,}0471 \\ 0{,}0881 & 0{,}0288 & 0{,}0424 & 0{,}0597 & -0{,}0115 & 0{,}0510 \\ 0{,}0042 & -0{,}0209 & 0{,}0242 & 0{,}0199 & 0{,}0240 & -0{,}0038 \\ 0{,}0134 & 0{,}0257 & -0{,}0330 & 0{,}0992 & 0{,}0458 & 0{,}0093 \\ -0{,}0089 & 0{,}0342 & 0{,}0011 & 0{,}0119 & 0{,}0094 & 0{,}0219 \\ 0{,}0067 & -0{,}0021 & -0{,}0253 & 0{,}0339 & 0{,}0297 & 0{,}0626 \\ 1{,}0212 & 0{,}0163 & 0{,}0313 & 0{,}0270 & 0{,}0453 & -0{,}0748 \\ 0{,}0163 & 1{,}0562 & -0{,}0454 & 0{,}0533 & -0{,}0494 & -0{,}0254 \\ 0{,}0313 & -0{,}0454 & 1{,}0462 & 0{,}0079 & 0{,}0014 & -0{,}0304 \\ 0{,}0270 & 0{,}0533 & 0{,}0079 & 1{,}0431 & 0{,}0144 & 0{,}0866 \\ 0{,}0453 & -0{,}0494 & 0{,}0014 & 0{,}0144 & 1{,}0134 & 0{,}0083 \\ -0{,}0748 & -0{,}0254 & -0{,}0304 & 0{,}0866 & 0{,}0083 & 1{,}0315 \\ -0{,}0380 & 0{,}0132 & -0{,}0383 & 0{,}0266 & 0{,}0089 & 0{,}0249 \\ 0{,}0169 & 0{,}0204 & 0{,}0889 & 0{,}0177 & -0{,}0655 & 0{,}0276 \\ 0{,}0355 & -0{,}0224 & -0{,}0311 & 0{,}0212 & 0{,}0120 & 0{,}0216 \\ 0{,}0650 & -0{,}0239 & -0{,}0249 & -0{,}0105 & -0{,}0178 & 0{,}0012 \\ 0{,}0289 & 0{,}0645 & 0{,}0054 & 0{,}0860 & 0{,}0290 & 0{,}0505 \\ 0{,}0776 & -0{,}0327 & 0{,}0007 & 0{,}0170 & 0{,}0129 & 0{,}0284 \end{bmatrix}$$

$$Q_t^{16} = \begin{bmatrix} 0{,}0534 & -0{,}0055 & 0{,}0502 & -0{,}0461 & -0{,}0245 & -0{,}0062 \\ 0{,}0326 & -0{,}0193 & -0{,}0772 & 0{,}0227 & 0{,}0202 & 0{,}0083 \\ 0{,}0091 & 0{,}0005 & -0{,}0034 & -0{,}0573 & 0{,}0321 & -0{,}0703 \\ -0{,}0894 & -0{,}0068 & -0{,}0033 & 0{,}0468 & 0{,}0255 & -0{,}0139 \\ 0{,}1103 & 0{,}0499 & 0{,}0018 & 0{,}0588 & -0{,}0053 & 0{,}0066 \\ -0{,}0548 & 0{,}0181 & -0{,}0184 & 0{,}0756 & -0{,}0118 & -0{,}0106 \\ 0{,}0409 & -0{,}0222 & 0{,}0546 & -0{,}0213 & -0{,}0505 & -0{,}0018 \\ -0{,}0082 & 0{,}0016 & 0{,}0461 & -0{,}0066 & 0{,}0233 & -0{,}0332 \\ 0{,}0127 & -0{,}0126 & 0{,}0502 & 0{,}0140 & -0{,}0319 & 0{,}0710 \\ 0{,}0081 & 0{,}0060 & 0{,}0122 & -0{,}0160 & 0{,}0225 & 0{,}0556 \\ -0{,}0663 & -0{,}0258 & 0{,}0318 & -0{,}0219 & 0{,}0121 & -0{,}0547 \\ -0{,}0367 & 0{,}0066 & 0{,}0014 & -0{,}0578 & 0{,}0534 & -0{,}0238 \\ 0{,}0394 & -0{,}0623 & 0{,}0131 & -0{,}0132 & -0{,}0124 & -0{,}0055 \\ -0{,}0099 & 0{,}0390 & -0{,}0083 & 0{,}0163 & 0{,}0549 & -0{,}0751 \\ -0{,}0073 & 0{,}0422 & 0{,}0251 & 0{,}0044 & 0{,}0198 & -0{,}0214 \\ -0{,}0234 & 0{,}0157 & -0{,}0098 & 0{,}0031 & 0{,}0447 & 0{,}0236 \\ 0{,}0260 & -0{,}0051 & -0{,}0057 & -0{,}0370 & 0{,}0190 & 0{,}0669 \\ 0{,}0036 & 0{,}0086 & -0{,}0219 & -0{,}0270 & -0{,}0038 & 0{,}0359 \end{bmatrix}$$

$$Q_t^{26} = \begin{bmatrix} 0{,}0469 & 0{,}0098 & -0{,}0313 & 0{,}0064 & -0{,}0542 & -0{,}0389 \\ -0{,}0135 & 0{,}0505 & -0{,}0106 & -0{,}0005 & 0{,}0193 & 0{,}0130 \\ -0{,}0239 & 0{,}0377 & 0{,}0011 & 0{,}0466 & -0{,}0385 & -0{,}0338 \\ 0{,}0104 & 0{,}0137 & 0{,}0332 & -0{,}0345 & 0{,}0327 & -0{,}0436 \\ 0{,}0329 & 0{,}0413 & -0{,}0130 & 0{,}0024 & -0{,}0460 & -0{,}0478 \\ 0{,}0271 & -0{,}0730 & 0{,}0354 & 0{,}0282 & -0{,}0026 & 0{,}0086 \\ -0{,}0380 & 0{,}0169 & 0{,}0355 & 0{,}0650 & 0{,}0289 & 0{,}0776 \\ 0{,}0132 & 0{,}0204 & -0{,}0224 & -0{,}0239 & 0{,}0645 & -0{,}0327 \\ -0{,}0383 & 0{,}0889 & -0{,}0311 & -0{,}0249 & 0{,}0054 & 0{,}0007 \\ 0{,}0266 & 0{,}0177 & 0{,}0212 & -0{,}0105 & 0{,}0860 & 0{,}0170 \\ 0{,}0089 & -0{,}0655 & 0{,}0120 & -0{,}0178 & 0{,}0290 & 0{,}0129 \\ 0{,}0249 & 0{,}0276 & 0{,}0216 & 0{,}0012 & 0{,}0505 & 0{,}0284 \\ 0{,}9889 & 0{,}0058 & 0{,}0186 & -0{,}0363 & 0{,}0179 & 0{,}0854 \\ 0{,}0058 & 0{,}9875 & 0{,}0640 & -0{,}0112 & -0{,}0298 & -0{,}0766 \\ 0{,}0186 & 0{,}0640 & 0{,}9967 & 0{,}0225 & 0{,}0284 & 0{,}0180 \\ -0{,}0363 & -0{,}0112 & 0{,}0225 & 1{,}0914 & 0{,}0604 & 0{,}0405 \\ 0{,}0179 & -0{,}0298 & 0{,}0284 & 0{,}0604 & 1{,}0139 & 0{,}0625 \\ 0{,}0854 & -0{,}0766 & 0{,}0180 & 0{,}0405 & 0{,}0625 & 1{,}0493 \end{bmatrix}$$

Literaturverzeichnis

Anderberg, M.R. (1973): *Cluster Analysis for Applications.* Academic Press, New York, New York.

Anderson, B. D. O.; Moore, J.B. (1979): *Optimal Filtering.* Prentice-Hall, Englewood Cliffs.

Beugler-Bell, H. (1996): *Öko-pedeologische Untersuchungen im Etoscha Nationalpark und angrenzenden Landschaften in Nordnambia*. Dissertation, Universität Regensburg.

Bishop, C.M., Svensén, M. und Williams, C.K.I. (1996): GTM: a principle alternative to the self-organizing map. *Proceedings of ICANN'96, International Conference on Artifical Neural Network*, *Lecture Notes in Computer Science, Vol. 1112*, Seiten 165-170, Springer, Berlin.

Bock, H.-H. (1974): *Automatische Klassifikation*. Vandenhoeck & Ruprecht, Göttingen.

Bock, H.-H. (1998): *Clustering and neural networks.* Aus: Rizzi, A.; Vichi, M.; Bock, H.-H. (Hrsg.): Advances in data science and classification. Springer, Heidelberg, Seiten 265-278.

Box, G.E.P.; Jenkins, G.M. (1976): *Time series analysis.* Holden-Day, San Fransisco.

Box, G.E.P.; Jenkins, G.M. und Reinsel, G.C. (1994): *Time Series Analysis: Forcasting and Control"*, 3. Auflage, Prentice Hall, Engelwood Cliffs, N.J., USA.

Braun, H. (1997): *Neuroanle Netze*. Springer, Berlin.

Breburda, J. (1983): *Die Winderosion.* Aus: Bodenerosion und Bodenhaltung, DLB-Verlag, Frankfurt am Main.

Bürkert, A., Lamers, J.P.A. (1999): Soil erosion and deposition effects on surface characteristics and pearl millet growth in the West Afican Sahel. *Plant and Soil,* Vol. 215, Seiten 239 - 253

Bürkert, A.; Mahler, F. und Marschner, H. (1996): Soil productivity management and plant growth in the Sahel: Potential of an aerial monitoring technique. *Plant and Soil,* Vol. 180, Seiten 29-38

Davenport, A.G. (1960): Rationale for determining design wind velocities. *Journal of Atm. Soc. Civ., Vol. 86*, Seite 39-68.

Dempster, A.P., Laird, N.M. und Rubin, D.B. (1977): Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, Series B, Vol. 39, Seiten 1-38.

Düwel, O. (1995): *Die Bedeutung der Bodenrauhigkeit für die Bodenerosion durch Wind. - Ein Beitrag zur Quantifizierung der Bodenverluste-* . Dissertation, Universität Göttingen.

Fritzke, B. (1994): Growing cell structures – a self-organizing network for unsupervised and supervised learning. *Neural Network*, Vol. 7, Seite 1441-1460.

Gomer, D. (1994): *Oberflächenabfluß und Bodenerosion in Kleineinzugsgebieten mit Mergelböden unter einem semiariden mediterranen Klima.* Dissertation, Universität Karlsruhe.

Güting, R.H. (1992): *Datenstrukturen und Algorithmen*. Teubner Verlag, Stuttgart.

Haberäcker, P. (1995): *Praxis der digitalen Bildverarbeitung und Mustererkennung.* Hanser, München.

Hartigan, J.A. (1975): *Clustering algorithms.* Wiley, New York.

Harvey, A.C. (1990): Forecasting, structural time series models and the Kalman filter. Cambridge University Press.

Harvey, A.C. (1995): Zeitreihenmodelle. 2. Auflage, Oldenbourg, München.

Harvey, A.C.; Todd, P.H.J. (1983): Forecasting economic time series with structural and Box-Jenkins Models. A Case Study. Journal of Business & Economic Statistics, 4, Seiten 299-315. Mit Kommentaren von Ansley, C.F.; Findley, D.F. und Newbold, P.

Hebb, D.O. (1949): The Organization of behavior. Aus: *Anderson, J.A., Rosenfeld, E. (1988): Neurocomputing: Foundations of Research*. Kapitel 4, Seiten 45-46, MIT-Press, Massachuchetts.

Hoffman, K. (1975): *Analysis in Euclidean Space.* Prentice-Hall, Englewood-Hills, New Jersey.

Holtorf, K. (1996): *Das neue Handbuch der Grafikformate.* 3. neubearbeitete und erweiterte Auflage, Franzis-Verlag, Feldkirchen.

ISO 42871/1 (1984): *Surface roughness.*

Johnson, R.A., Wichern. D.W. (1992): *Applied multivariate statistical analysis.* 3rd Edition, Prentice Hall, Englewood-Cliffs, California.

Kalman, R. E. (1960): A new approach to linear filtering and prediction problems. *Journal of basic engineering*, Seite 35 – 45.

Kelly, P.; White, J. (1993): Preprocessing remotely-sensed data for efficient analysis and classification. Application of artificial intelligence. *Proceedings, SPIE 1993*, Knowledge Based Systems in Aerospace and Industry, Seiten 24-30.

Kohonen, T. (1982): Self-organizing formation of topologically correct feature maps. *Biological Cybernetics*, Vol. 43, Seiten 59-69.

Kohonen, T. (1990): The self-organizing map. *Proceedings of the IEEE*, Vol. 78, Seiten 1464-1480.

**Kohonen, T.; Kangas, J.; Laaksonen, J. (1992*): SOM-PAK. The self-organizimg map program package.* Report A30, Helsinki Univeritsy of Technology, Faculty of Infromation Technology, Laboratory of Computer and information Science, Espoo, Finnland.

Kohonen, T.;Hynninen, J. ;Kangas, J.; Laaksonen, J.; Torkkola, K. (1996): *LVQ-PAK. The learning vector quantization program package.* Report A31, Helsinki Univeritsy of Technology, Faculty of Infromation Technology, Laboratory of Computer and information Science, Espoo, Finnland.

Kohonen, T. (2001): *Self-organizing maps.* 3. Ausgabe, Springer, Heidelberg.

Koikkalainen, P. (1994): Progress with the tree-structured self-organizing map. *Proceedings of ECAI'94*, 11th European Conference on Artifical Intelligence, Seiten 211-215, Wiley, London.

Kolonko, P. (2002): Sand vor Augen: Die Monoglei ergreift Vorkehrungen gegen Wüstenbildung und Überweidung. Frankfurter Allgemeine Zeitung,, Nr. 111 (15.05.), Seite 9

Kraaijveld, M.A., Mao, J. und Jain, A.K. (1995): A nonlinear projection method based on Kohonen's topology preserving maps. *IEEE Transactions on Neural Networks*, Vol. 6, Seiten 548-559.

Kuntze, H. (1992): Bodenschutz in der Landschaft. *Recht der Landwirtschaft*, 44. Jahrgang, Seite 57-59.

MacQueen, J.B. (1967): Some methods for classification and analysis of multivariate observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, Seiten 281-297, Berkely, USA.

Matecki, U. (1999): *Automatische Merkmalsauswahl für Neuronale Netze mit Anwendung in der pixelbezogenen Klassifiaktion von Bildern.* Dissertation Fachbereich Informatik, Universität Osnabrück, Shaker-Verlag, Aachen.

McCulloch, W.S., Pitts, W. (1943): A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, Seiten 115-133.

Meyers (1999): *Meyers grosses Taschenlexikon in 25 Bänden.* B.I.-Taschenbuchverlag, Berlin.

Minsky, M., Pappert, S. (1969): Perceptrons. Aus: Anderson, J.A., Rosenfeld, E. (1988): *Neurocomputing: Foundations of research*. Kapitel 13, Seiten 161-170, MIT-Press, Massachuchetts.

Myers, W., Patil, G.P. und Taillie, C. (1997a*): PHASE formulation of synoptic multivariate landsape data. Technical report Nr. 97-1102*, Center for statistical ecology and environmental statistics, Department of Statistics, PennState University, Pennsylvania.

Myers, W., Patil, G.P. und Taillie, C. (1997b): Adapting quantitative multivariate geographic information system data for purposes of sample design: The PHASE approach. *Technical report Nr. 97-1201*, Center for statistical ecology and environmental statistics, Department of Statistics, PennState University, Pennsylvania.

Orey, S. (1971): *Limit theorems for markov chain transition probabilites.* van Nostrand Verlag, London.

Pye, K. (1987): *Aeolian dust and dust deposits.* Academic Press, London.

Rojas, R. (1996): *Theorie der neuronalen Netze. Eine systematische Einführung.* 4.korrigierter Nachdruck, Springer, Heidelberg.

Rosenblatt, F.D: (1958): The perceptron: a probabilistic model for information storage and organization in the brain. Aus: Anderson, J.A., Rosenfeld, E. (1988): *Neurocomputing: Foundations of research*. Kapitel 8, Seiten 92-116, MIT-Press, Massachuchetts.

Rumelhart, D.E. (1986): *Parallel distributed processing.* Vol. 1, MIT-Press, Cambridge, MA., USA.

Schlittgen, R.; Streitberg, B. H.-J. (1997): *Zeitreihenanalyse.* 7. Auflage, Oldenbourg, München.

Schmidt, J.W. (1983): *Relational database systems.* Springer, Berlin.

Särndal, C.-E., Swensson, B, und Wretman, J. (1992): *Model Assisted survey sampling.* Springer, New York.

Stier, W. (2001): *Methoden der Zeitreihenanalyse*. Springer, Berlin.

Tou, J.T.; Gonzalez, R.C. (1974): *Pattern recognition principles.* Addison-Wesley, Reading, Mass.

Tsay, R.S.; Tiao, G.C. (1985): Use of Canonical Analysis in Time Series Model identification. *Biometrica,* 72, Seite 299 – 315.

Tschiersch, L. (1998): *Auswertung von zeitabhängigen Daten von Erosions-Dauerbeobachtungsflächen im Wassereinzugsgebiet von Rabat.* Diplomarbeit, Universität Dortmund.

Ultsch, A.G.H.; Siemon, H.P. (1989): *Exploratory data anlysis using Kohonen networks on tranputers.* Technical Report 329, Fachbereich Informatik, Universität Dortmund, Dortmund.

UNCCD 241/27 (1994): *Elaboration of an international convention to combat desertification in countries experiencing seroius drought and/or desertification, particulary in Africa.*

UNFPA (1999): http://www.unfpa.org/popin/wdtrends/pop/1998/3.htm

UNFPA (2001): http://www.unfpa.org/modules/6billion/facts.htm

Widow, B., Hoff, M.E. (1960): Adaptive switching circuits. Aus: Anderson, J.A., Rosenfeld, E. (1988): *Neurocomputing: Foundations of research*. Kapitel 10, Seiten 126-134, MIT-Press, Massachuchetts.

Zell, A. (2000): *Simulation neuronaler Netze.* 3.Auflage, Oldenbourg, München.

Zerbst, M. (1999): *Untersuchung von Einflußgrößen der Erosion in Waldgebieten Marokkos.* Diplomarbeit, Universität Dortmund.

Zerbst, M.; Tschiersch, L.; Guimaraes, G.; Talbi, M.; Urfer, U. (2000): On clustering of aerial photographs and high resolution satellite images. *Technical Report 28/2000*, University Dortmund, Germany.

Zerbst, M. (2001): *Die pixelbasierte Clusterung von Luftaufnahmen im Rahmen von Erosionsuntersuchungen*. Dissertation, Universität Dortmund, Dortmund.