Clusteranzahlbestimmung und Clusterung unter Nebenbedingungen in der Musiksignalanalyse und in Energienetzen

Dissertation zur Erlangung des Grades eines Doktors der Naturwissenschaften der Technischen Universität Dortmund

Der Fakultät Statistik vorgelegt von

Sebastian Krey

16. Dezember 2022

Inhaltsverzeichnis

1 Einleitung

Clusterverfahren sind eine leistungsfähige Methode des unüberwachten maschinellen Lernens, um komplexe, mehrdimensionale Datensätze zu strukturieren. Im Laufe der Zeit sind verschiedene agglomerative und divisive Clusterverfahren eingeführt worden. Hierbei hat sich insbesondere das divisive Verfahren k-Means aufgrund der schnellen Berechenbarkeit mit dem Lloyd Algorithmus sehr breit in der Praxis durchgesetzt.

In Kapitel 8 wird auf Basis einer klassischen distanzbasierten Clusterung von Musikklangmerkmalen eine Klassifikation von Instrumentenklängen mit überwachten maschinellen Lernverfahren durchgeführt und gute Ergebnisse auch bei einer sehr hohen Anzahl von Klangfarben erreicht.

Bei verschiedenen Anwendungen müssen neben der Distanz zwischen den Datenpunkten aber zusätzliche Nebenbedingungen an eine sinnvolle Clusterung gestellt werden. Bei der in Kapitel 10 betrachteten Fragestellung zur automatisierten Unterteilung eines Energienetzes in Regionen, muss die durch einen Graph beschriebene Nachbarschaftsstruktur berücksichtigt werden, da nur über eine Stromleitung (Kante des Graphen) verbundene Knoten sinnvoll einem Cluster zugeordnet werden können. Für derartige Problemstellungen sind die Methoden des *Spectral Clustering* entwickelt worden und eignen sich hervorragend für diese Anwendung. Dies zeigen die Ergebnisse in Kapitel 10.4.

Bei Zeitreihendaten wie Tonaufnahmen, ist eine feste Nachbarschaftsstruktur oder zwingende Gruppenzugehörigkeiten oft nicht vorhanden, wohingegen es eine zeitliche Ordnung als Nebenbendingung für eine sinnvolle Clusterung gibt. Das für solche Fragestellungen entwickelte Verfahren *Order Constrained Solutions in k-Means Clustering* (Steinley und Hubert, 2008) integriert diese Nebenbedingung in das k-Means Optimierungsproblem. In Kapitel 9 wird dieses Verfahren vorgestellt und gezeigt, wie sich dadurch die Fehlklassifikationsrate bei der Erkennung von Instrumentenklangfarben im Vergleich zu Kapitel 8 weiter reduzieren lässt. Im weiteren Verlauf des Kapitels wird eine effiziente Methode zur Berechnung des komplexen Optimierungsproblems entwickelt.

Bei all diesen Verfahren muss am Ende festgelegt werden, in wie viele Cluster der Datensatz aufgeteilt werden soll. Bei k-Means und ähnlichen Verfahren muss dem Algorithmus die Anzahl der Cluster schon als Parameter k übergeben werden. Es gibt

zwar eine Vielzahl von Kennzahlen und Methoden anhand derer diese Festlegung objektiv möglich sein sollte, aber die Effektivität dieser Methoden ist stark von den Daten abhängig. In Kapitel 7 wird das etablierte Verfahren die Qualität der Clusterung anhand der Stabilität über Boostrapstichproben der ursprünglichen Daten zu beurteilen so weiterentwickelt, dass es auch für Clusterverfahren mit Nebenbedingungen einsetzbar ist. Zusätzlich wird die Betrachtung der Streuung, der mit Hilfe des mittleren *Rand-Index* berechneten Stabilität, als zweites Kriterium zur Festlegung der Clusteranzahl vorgeschlagen. Die Kombination dieser beiden Kriterien zum *Quartilsdispersionskoeffizienten* gibt dem Anwender wiederum eine einzelne Kennzahl zur Beurteilung der Clusterung an die Hand.

Zwischenstände der im Rahmen dieser Arbeit entwickelten Methoden und damit erzielten Ergebnisse sind in den Veröffentlichungen Krey und Ligges (2010), Ligges und Krey (2011), Krey, Ligges und Leisch (2014) und Krey, Brato u. a. (2015) publiziert. Die Resultate der Kapitel 6.2, 7.2, 7.4, 9.2, 9.6, 10.6 und 10.7 sind unveröffentlicht und werden in dieser Arbeit zum ersten Mal beschrieben.

Die ersten Kapitel dieser Arbeit legen Grundlagen zu Klang und Psychoakustik (Kapitel 2) sowie der Extraktion von Klangmerkmalen (Kapitel 3). Es handelt sich hierbei um die für eine bessere Verständlichkeit überarbeiteten Grundlagenkapitel aus Krey (2008). Der Visualisierung von Klangmerkmalen widmet sich Kapitel 4. Anschließend stellt Kapitel 5 die bis einschließlich Kapitel 9 verwendeten Klangdaten vor.

Das Kapitel 10 widmet sich einem anderen Teilgebiet der Clusterverfahren mit Nebenbedingungen. Hier wird eine durch einen Graph beschriebene Nachbarschaftsstruktur verwendet, um Höchstspannungsenergienetze unter Berücksichtigung ihrer Netzwerktopologie in Regionen zu clustern.

2 Grundlagen Klang und Psychoakustik

Dieses Kapitel ist das zur besseren Verständlichkeit überarbeitete Grundlagenkapitel aus Krey (2008).

Zur Motivation der in Kapitel 3 beschriebenen Klangmerkmale, werden im Folgenden einige Grundlagen zu Schall, Klängen und deren digitaler Aufzeichnung näher erläutert. Ferner werden Grundlagen zur Struktur von Klängen und Musikstücken vermittelt, um ein besseres Verständnis der Ziele, der in dieser Arbeit vorgestellten Methoden zu erreichen.

2.1 Klang

Die physikalische Grundlage aller Klänge und Töne, also die Grundbausteine von Musik, sind die als *Schall* bezeichneten, hörbaren Druckschwankungen in einem elastischen Medium. Sie breiten sich wellenartig (*Schallwellen*) aus. Zur Entstehung von Schallwellen ist neben einer Schallquelle ein elastisches Übertragungsmedium notwendig. In diesem breitet sich die Welle mit einer charakteristischen und konstanten Geschwindigkeit c (*Schallgeschwindigkeit*) aus. Die Schallgeschwindigkeit ist neben dem Ausbreitungmedium auch von den Umgebungsbedingungen (insb. der Temperatur) abhängig. Bei einer Temperatur von 20° C beträgt die Schallgeschwindigkeit in Luft $c = 343\,\mathrm{m/s}$ (Giancoli, 2010). Im Vakuum gibt es mangels Übertragungsmedium keinen Schall und daher auch keinerlei Töne, Klänge oder Geräusche.

Die Schallquelle überträgt ihre Schwingungen auf die Teilchen des Übertragungsmediums und sorgt hierdurch für die Fortpflanzung der Schwingung. Ist das Übertragungsmedium die Luft, so erzeugt die Schallquelle Luftdruckschwankungen, die sich in Form einer *Longitudinalwelle* im Raum ausbreiten. Diese Welle lässt sich allgemein im eindimensionalen Fall durch die Wellengleichung (Walker, 1996)

$$c^2 \frac{\partial^2 a(x,t)}{\partial x^2} + f(x,t) = \frac{\partial^2 a(x,t)}{\partial t^2}$$

beschreiben. Hierbei ist $a(x,t)$ die Auslenkung an Stelle x zum Zeitpunkt t, $f(x,t)$ die anregende Kraft zum Zeitpunkt t an der Stelle x und c die Ausbreitungsgeschwindigkeit der Welle.

Charakterisiert wird eine Welle durch die *Amplitude* A, welche die maximale Auslenkung beschreibt, die *Schwingungs- oder Periodendauer* T sowie die *Phase* j, die zur

Erfassung zeitlicher Verschiebung mehrerer Wellen untereinander oder einer Welle zu einem Referenzpunkt dient. Alternativ zur Periodendauer T wird auch die Frequenz $f = 1/T$ oder die Wellenlänge $\lambda = c/f$ angegeben. Die SI-Einheit der Frequenz ist das Hertz, $1\,\text{Hz} = 1\,1/\text{s}$. Die Angabe der Frequenz ist durch die Proportionalität zur Tonhöhe einfacher zu verstehen und daher am häufigsten anzutreffen. Die Wellenlänge ist dagegen für die Veranschaulichung des Zusammenhangs von Saitenlänge und Tonhöhe oder für akustische Berechnungen (Klangkörper, Grenzfrequenzen eines Raums, etc.) nützlich.

Die Amplitude der Welle ist ein Maß für die Stärke der Luftdruckschwankungen, dem Schalldruck P und ist damit Maß für die Lautstärke des wahrgenommenen Tons. Die Maßeinheit für den Druck ist das Pascal Pa, welches durch

$$1\,\text{Pa} = 1\,\text{N}/\text{m}^2 = 1\,\text{kg}/\text{ms}^2$$

definiert ist. Damit der Mensch einen Ton oder ein Geräusch wahrnehmen kann, muss die Luftdruckschwankung mindestens $P_{min} = 20\,\mu\text{Pa}$ betragen (Kremer, 2008).

Die einfachste Wellenform ist eine Sinusschwingung. Solch einfach aufgebauten Töne sind in der Praxis allerdings sehr selten.

Klänge als Überlagerung von Schwingungen

Zur Beschreibung von komplexen Schallereignissen, wie sie von Musikinstrumenten erzeugt werden, reichen einfache Schwingungen nicht mehr aus. Die angenehm klingenden Überlagerungen mehrerer Sinusschwingungen, wie sie von Instrumenten erzeugt werden, nennt man *Klänge*. Sie entstehen wenn sich Schwigungen überlagern, deren Frequenzen in ganzzahligen Verhältnissen zueinander stehen.

Die Schwingungen eines Klangs lassen sich somit schreiben als

$$k(t) = \sum_{k=0}^{K} A_k \cos\left(k\omega t + \varphi_k\right),$$

wobei $\omega = 2\pi/T$ und K die Anzahl der Oberschwingungen ist.

Dies ist typisch für die Schwingungen, welche beispielsweise die Saiten einer Gitarre, Violine oder eines Klaviers ausführen (Kremer, 2008). Die Schwingung mit der tiefsten Frequenz innerhalb eines Klangs bezeichnet man als *Grundschwingung*. Die Schwingungen mit ganzzahligen Vielfachen dieser Frequenz werden als *Oberschwingungen*

oder *Obertöne* bezeichnet. Die Obertonstruktur ist charakteristisch für ein Instrument oder eine Klangfarbe und die im nächsten Kapitel vorgestellten Klangmerkmale modellieren diese Struktur, um den Klang mittels Kennzahlen zu beschreiben.

Während Klänge aus Schwingungskomponenten bestehen, die in ganzzahligen Zahlenverhältnissen zueinander stehen, erfüllen *Geräusche* diese Bedingung nicht (Kremer, 2008). Geräusche werden daher oft als nicht wohlklingend empfunden.

2.2 Psychoakustik

Über die zuvor beschriebenen Merkmale Amplitude, Frequenz bzw. Wellenlänge kann man zwar Töne und Klänge basierend auf ihren physikalischen Eigenschaften charakterisieren, diese Merkmale entsprechen aber nicht der Wahrnehmung durch den menschlichen Hörapparat. Die Wahrnehmung von akustischen Ereignissen ist höchst individuell und wird neben anatomischen Unterschieden (z.B. Form der Ohrmuschel, Zustand der Gehörknöchelchen und Gehörschnecke) insb. aber vom Gehirn bestimmt. Die Psychoakustik widmet sich der Erforschung der Zusammenhänge zwischen den physikalischen Eigenschaften von Schallereignissen und der menschlichen Wahrnehmung.

Eine der wichtigsten Erkenntnisse für die Anwendung in der Musiksignalanalyse ist, dass das menschliche Gehör Unterschiede zwischen Tönen verschiedener Lautstärke und Tonhöhe nicht so empfindet, wie es die Zahlenverhältnisse der Frequenzen und Schalldrücke ausdrücken. Dementsprechend ist das Auflösungsvermögen über den gesamten hörbaren Lautstärke- und Tonhöhenbereich nicht gleichmäßig hoch. Diese Erkenntnis wird bei den im nächsten Kapitel beschriebenen Klangmerkmalen genutzt, um die komplexen Schwingungsstrukturen von Klängen in Kennzahlen zu fassen.

Zusätzlich zu Tonhöhe und Lautstärke haben auch die Umgebungsbedingungen (leise Bibliothek oder Baustelle) und zeitliche Effekte (zuvor oder gleichzeitig eintretende Schallereignisse) einen starken Einfluss auf die Wahrnehmung von Tönen. Dies macht die Entwicklung von Signalverarbeitungsalgorithmen für Hörgeräte zur Unterstützung von hörgeschädigten Menschen im Alltag hochkomplex (Friedrichs, 2016; Panda u. a., 2014).

2.2.1 Lautheit

Der für das menschliche Gehör wahrnehmbare Lautstärkebereich ist nicht tonhöhenunabhängig. Im Bereich zwischen 3 und 4 kHz ist den Hörschwelle des Menschen am niedrigsten und steigt sowohl in Richtung tieferer als auch höherer Frequenzen an (Kremer, 2008).

Schalldrücke und damit auch die Lautstärke von Tönen werden vom menschlichen Gehör über einen sehr großen Wertebereich wahrgenommen. Hierbei ist zu berücksichtigen, dass Unterschiede niedriger Schalldrücke, also von leisen Tönen, vom Gehör sehr viel feiner aufgelöst werden, als die von hohen Schalldrücken. Dies führt dazu, dass Schalldruckunterschiede abhängig von der Höhe des Schalldrucks unterschiedlich wahrgenommen werden. Ein Unterschied, der bei geringem Schalldruck zu einem gerade noch wahrnehmbaren Lautstärkeunterschied führt, ist bei hohem Schalldruck nicht mehr wahrnehmbar. Die gerade noch wahrnehmbare Schalldruckänderung nimmt, abhängig von der Höhe des Schalldrucks, nahezu exponentiell zu.

Um den großen Wertebereich des Schalldrucks besser beschreiben zu können, definiert man die logarithmische Größe des *Schalldruckpegels* S, welche in *Dezibel* dB gemessen wird. Die Bezugsgröße zur Definition des Schalldruckpegels ist der Mindestschalldruck hörbarer Schallereignisse $P_{min} = 20\,\mu\text{Pa}$, dem der Wert $0\,\text{dB}$ zugeordnet wird. Den Schalldruckpegel S erhält man aus dem Schalldruck P und dem Mindestschalldruck P_{min} durch (Kremer, 2008)

$$S(P) = 20\log_{10}\left(\frac{P}{P_{min}}\right). \tag{2.1}$$

Zwischen der Schmerzgrenze und der Hörbarkeitsschwelle liegt eine Schalldruckpegeldifferenz von bis zu $120\,\text{dB}$. Umgerechnet auf den Schalldruck liegt ein Faktor von 10^6 zwischen diesen beiden Werten. Die Wahrnehmbarkeitsschwelle für eine Änderung des Schalldruckpegels liegt bei $1\,\text{dB}$ (Kremer, 2008).

2.2.2 Tonheit

Ähnlich zur Wahrnehmung der Lautstärke verschiedener Schalldruckpegel verändert sich auch die Wahrnehmung von Tonhöhendifferenzen über den hörbaren Frequenzbereich. Das menschliche Gehör ist für die Wahrnehmung und Unterscheidung der menschlichen Stimme optimiert. Im Frequenzbereich der Grundtöne der menschlichen Stimme (ca. $120\,\text{Hz}$ bis $450\,\text{Hz}$) können sehr geringe Tonhöhendifferenzen von

weniger als 2 Hz wahrgenommen werden (Kremer, 2008). In diesem Bereich wird eine Frequenzverdopplung (Änderung der musikalischen Tonhöhe um eine Oktave) als Tonhöhenverdopplung wahrgenommen. Dieses hohe absolute Auflösungsvermögen ist auch bei tieferen Frequenzen gegeben.

Oberhalb von ca. 500 Hz verändert sich im menschlichen Gehör die Tonhöhenwahrnehmung (Panda u. a., 2014). Hierdurch nimmt das Frequenzauflösungsvermögen des Gehörs mit steigender Tonhöhe ab, bis nur noch Tonhöhendifferenzen, die einen festen Anteil der Frequenz überschreiten, wahrgenommen werden (Kremer, 2008). Eine Verdopplung der Frequenz wird nicht mehr als Tonhöhenverdopplung empfunden. Daher können auch Tonhöhenunterschiede ab einem gewissen Punkt durch eine logarithmische Skala erklärt werden. Um diesen Sachverhalt zu Modellieren sind basierend auf den Daten psychoakustischer Experimente (Steinberg, 1937; Stevens, Volkmann und Newman, 1937) verschiedene Skalen entwickelt worden. Die hier verwendete Skala ist die *Melskala*. Sie ist durch

$$Mel(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.2}$$

definiert (Kremer, 2008). Diese Kurve schneidet eine Gerade mit einer Steigung von eins bei 1000 Hz. Am oberen Ende des Grundtonbereichs (um 450 Hz) weicht der Verlauf nur minimal von einer um 107 Mel nach oben verschobenen Geraden mit Steigung eins ab. Im darunter liegenden Grundtonbereich, der vom menschlichen Gehör sehr fein aufgelöst werden kann, ist die Steigung steiler und im Bereich darüber sorgt die Logarithmusfunktion für einen deutlich flacheren Verlauf. Diese Kurve modelliert daher sehr gut das Tonhöhenempfinden des menschlichen Gehörs. Die Skala spiegelt das realistische Empfinden deutlich präziser wider als die Frequenz in Hertz. Zahlenverhältnisse verschiedener Melwerte haben über einen deutlich größeren Bereich der Frequenzskala eine vergleichbare Bedeutung. Abbildung 2.1 zeigt die Kurve zusammen mit den beschriebenen Vergleichspunkten.

2.3 Digitale Klangaufzeichnung

Um Klänge analysieren zu können, müssen sie in numerischer Form vorliegen. Dies erfordert eine digitale Aufzeichnung der Klänge. Hierzu wird das bei der Aufnahme durch ein Mikrofon erzeugte analoge Signal digitalisiert. Die Schallschwingungen des Klangs werden durch das Mikrofon in eine Wechselspannung umgewandelt. Diese wird von einem Analog-Digital-Wandler (AD-Wandler) in eine binäre Darstellung konvertiert. Hierzu wird in äquidistanten Abständen die Spannung gemessen und

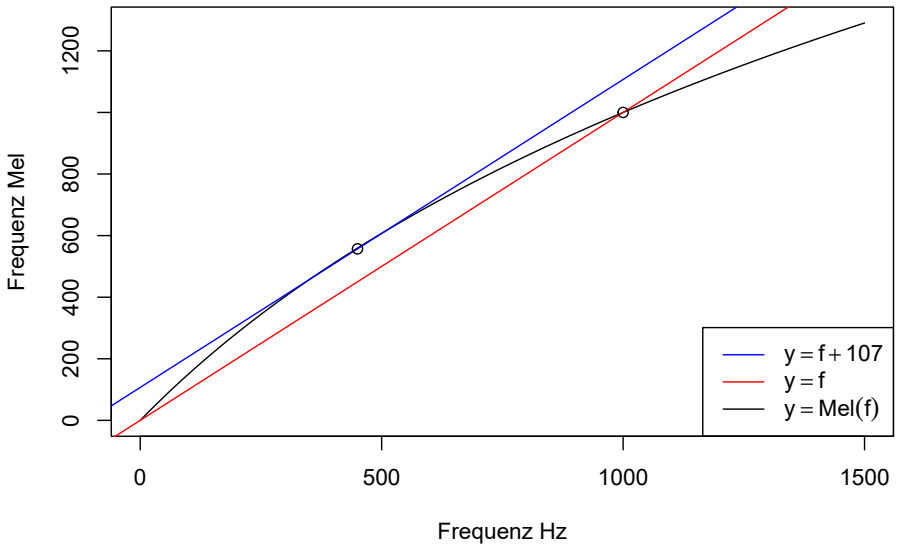

Abbildung 2.1: Veranschaulichung des Verlaufs der Melskala im Grundtonbereich

auf einen diskreten Wert abgebildet (*quantisiert*). Dieses Vorgehen wird als *Sampling* bezeichnet. Hierbei wird der stetige Verlauf der Schwingung durch eine Treppenfunktion approximiert. Die Qualität der Approximation wird durch die Häufigkeit der Abtastung und die Auflösung des Datenformats bestimmt. Die Häufigkeit der Abtastung des Signals wird *Samplerate* SR genannt und wird in Abtastungen pro Sekunde angegeben. Daraus ergibt sich als Einheit für die Samplerate das *Hertz* ($Hz = 1/s$). Die durch die AD-Wandlung erhaltenen diskreten Werte sind, abhängig vom gewählten Datenformat, vorzeichenbehaftete ganzzahlige Werte oder Fließkommazahlen. Die durch das Sampling erhaltenen Werte sind aus statistischer Sicht eine Zeitreihe mit diskreten, äquidistanten Zeitpunkten.

Wichtig bei der Durchführung des Samplings ist die Beachtung des Sampling Theorems (Shannon, 1998; Nyquist, 2002). Dies besagt, dass nur Frequenzen kleiner der Hälfte der Samplerate SR korrekt, das heißt ohne *Aliasing* erfasst werden. Die höchste eindeutig beschreibbare Frequenz ν, welche als *Nyquistfrequenz* bezeichnet wird, ist durch

$$\nu = \frac{SR}{2}$$

gegeben. Als *Aliasing* bezeichnet man das Phänomen, dass Frequenzen oberhalb der Nyquistfrequenz ν als Schwingungen von eigentlich nicht im Spektrum vorhandenen Frequenzen in dem gesampelten Signal erscheinen (Nyquist, 2002).

Bei dem bis zur Einführung der DVD als Tondatenträger im Jahr 1996 am häufigsten anzutreffenden Samplingformat wird das Signal 44 100 mal pro Sekunde abgetastet und auf einen ganzzahligen 16 bit Wert zwischen −32 768 und +32 767 (European Broadcasting Union, 2011) abgebildet. Man spricht hierbei häufig auch von *CD-Qualität*. Die Nyquistfrequenz liegt hier bei $\nu = 22\,050\,\mathrm{Hz}$. Durch die DVD hat sich 16 bit bei 48 000 Hz als weiteres Samplingformat sehr weit verbreitet und verdrängt die 44 100 Hz Samplingrate in immer mehr Anwendungsgebieten. Weitere wichtige Samplingformate sind 24 bit bei 96 000 Hz und 24 bit bei 192 000 Hz, die in der professionellen Tontechnik sowie bei High Definition Audio Formaten (z.B. für Audio-DVD, Blu-ray Disc) eingesetzt werden. Zwischen diesen Standards liegende Formate mit 18 oder 20 bit Auflösung und zusätzlichen Samplingraten von 88 200 Hz oder 176 400 Hz (Vielfache der Samplingrate der CD) sind definiert, werden aber nur für spezielle Anwendungen genutzt. Geringere Samplingraten als 44 100 Hz haben noch eine Anwendung bei der bandbreitensparenden Übertragung von Sprache, z.B. bei der Telefonie (8 000 Hz oder 16 000 Hz sowie seltener 32 000 Hz). Hier werden auch zum Teil Auflösungen unterhalb von 16 bit verwendet (z.B. 14, 13 oder 12 bit Auflösung, die zur Einsparung von Datenübertragungsbandbreite auf 8 bit komprimiert werden).

3 Merkmalsextraktion

Dieses Kapitel ist das zur besseren Verständlichkeit überarbeitete Kapitel zur Merkmalsextraktion aus Krey (2008).

Im vorigen Kapitel wurde die grundlegende Struktur von Klängen und ihre Datenaufzeichnung beschrieben. Die so erfassten Rohdaten müssen für eine sinnvolle Weiterverarbeitung in der Musiksignalanalyse zu Kennzahlen zusammengefasst werden, die den Klang beschreiben. Ferner müssen auf diesen Kennzahlen Distanzmaße definiert werden, die auch den vom Menschen empfundenen Klangunterschied widerspiegeln. Auf Basis dieser Distanzen können dann die verschiedenen Fragestellungen der Musiksignalanalyse zum Beispiel mit statistischen Lernverfahren bearbeitet werden.

Der Verarbeitungsschritt zur Bestimmung dieser Kennzahlen wird als *Merkmalsextraktion* (engl. *Feature Extraction*) bezeichnet. Im Laufe der Zeit wurden verschiedene Kennzahlen als Merkmale zur Beschreibung von Klängen entwickelt. Diese Arbeit fokussiert sich auf Merkmale, die auf einer *gefensterten diskreten Fourier-Transformation* (engl. Short Time Fourier Transformation) basieren und daher die Frequenzstruktur des Signals analysieren.

Die am häufigsten genutzten frequenzbasierten Klangmerkmale sind die *Mel Frequency Cepstral Coefficients (MFCC)* (vgl. Childers, Skinner und Kemerait, 1977; Davis und Mermelstein, 1980), siehe Abschnitt 3.3.1, und die *Perceptual Linear Prediction (PLP)* (Hermansky, 1990), siehe Abschnitt 3.3.2. Diese Merkmale wurden ursprünglich für die Spracherkennung entwickelt, haben sich aber auch bei der Analyse von Musiksignalen als sehr leistungsfähig erwiesen.

Die Merkmalsextraktion ist ein mehrstufiger Prozess, der in den folgenden Abschnitten beschrieben wird. Die ersten Verarbeitungsschritte Preemphasis Filterung, gefensterte Fourier-Transformation sowie die psychoakustisch motivierte Frequenztransformation sind bei MFCCs und PLPs identisch und werden gemeinsam beschrieben. Die weiteren Verarbeitungsschritte zur Berücksichtigung der psychoakustischen Eigenschaften des Hörens sowie die eigentlichen Kennzahlen zur Beschreibung des Klangs werden in den individuellen Abschnitten beschrieben.

3.1 Vorverarbeitung

Preemphasis Filterung

Zur Verstärkung der hohen Frequenzen und damit der Obertöne eines Klangs wird zu Beginn der Verarbeitungskette eine Preemphasis Filterung durchgeführt. Das Ziel hierbei ist, die für jeden Klang charakteristische Obertonstruktur zu verstärken und damit besser erkennbar zu machen. Dies ist notwendig, da die Energie der Obertöne meist geringer als die Energie der Schwingung mit der Grundfrequenz ist und (z.B. bei Saiteninstrumenten) mit steigendem Vielfachen der Grundfrequenz immer weiter abnimmt. Gleichzeitig beherbergt die Obertonstruktur einen großen Teil der Klanginformation und ist daher von großer Bedeutung. Das einfachste Werkzeug zur Realisation einer Preemphasis Filterung ist ein linearer Filter, der alle Frequenzen oberhalb einer bestimmten Grenzfrequenz anhebt.

Ein linearer Filter ist eine lineare Transformation der Zeitreihe x_t der Länge T mit einer Matrix L, in eine andere Zeitreihe y_t (Schlittgen und Streitberg, 2001):

$$y_t = Lx_t = \sum_{u=-q}^{s} a_u x_{t-u}, \qquad t = s+1, \ldots, T-q$$

Die Länge der Zeitreihe y_t beträgt $T-q-s$. Die Matrix L hat die Dimension $T-q-s \times T$ und die Gestalt

$$L = \begin{pmatrix} a_s & \ldots & a_0 & \ldots & a_{-q} & 0 & \ldots & \ldots & 0 \\ 0 & a_s & \ldots & a_0 & \ldots & a_{-q} & 0 & \ldots & 0 \\ \vdots & \ddots & \ddots & & \ddots & & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & a_s & \ldots & a_0 & \ldots & a_{-q} & 0 \\ 0 & \ldots & \ldots & 0 & a_s & \ldots & a_0 & \ldots & a_{-q} \end{pmatrix}.$$

Der zur Preemphasis Filterung verwendete lineare Filter wird als *One Zero Filter* bezeichnet und hat die Gestalt

$$y_t = x_t + ax_{t-1}, \qquad t = 2, \ldots, T,$$

wobei a geeignet zu wählen ist. Eine in der Praxis häufig empfohlene Wahl ist $a = -0.97$ (Slaney, 1998; Ellis, 2005).

Die frequenzabhängige Verstärkung des Filters erhält man durch Betrachtung seiner z-Transformation $Y(z)$, wobei hier die z-Tranformation die *Laplace-Transformation von*

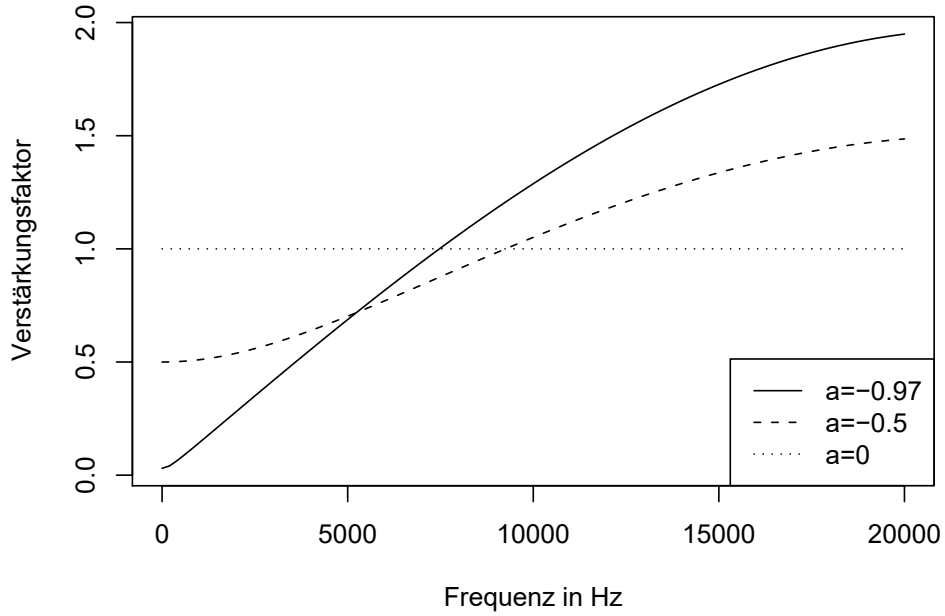

Abbildung 3.1: Frequenzabhängige Verstärkung des Preemphasisfilters

Abtastfunktionen meint, welche eine Laplace-Transformation für zeitdiskrete Funktionen ist. $Y(z)$ erhält man durch das Produkt der z-Transformation von X und der z-Transformation einer weiteren Funktion H (Smith, 2007):

$$Y(z) = H(z) \cdot X(z)$$

Die Funktion $H(z)$ hat die Form

$$H(z) = 1 + az^{-1}.$$

Durch Einsetzen von $z = \exp(2\imath\pi f/SR)$ erhält man die frequenzabhängige Formulierung von H:

$$H_f(f) = 1 + a\exp\left(-\imath\frac{2\pi f}{SR}\right) = 1 + a\cos\left(\frac{2\pi f}{SR}\right) - \imath a\sin\left(\frac{2\pi f}{SR}\right)$$

Durch Berechnung des Betrags von $H_f(f)$ erhält man die frequenzabhängige Verstärkung des Filters (Smith, 2007)

$$G_f(f) = \sqrt{1 + a^2 + 2a\cos\left(\frac{2\pi f}{SR}\right)}.$$

In Abbildung 3.1 für einige Werte von a und einer Samplerate von 44 100 Hz die Verstärkung gezeichnet.

3.2 Fourier-Transformation

Die in dieser Arbeit verwendeten Klangmerkmale sind frequenzbasiert. Es ist daher notwendig die durch den Preemphasis Filter transformierte Zeitreihe y_t in den Spektralraum zu transferieren. Dabei wird der Klang in seine einzelnen Komponenten (reine Schwingungen) zerlegt, die als Überlagerung den Klang bilden. Diese einzelnen Schwingungen können dann auf ihre Frequenz und Amplitude analysiert werden. Das theoretische Konzept, auf dem die Fourier-Transformation dabei aufbaut, ist die *Fourier-Analyse*.

Die Fourier-Analyse einer Funktion $g(t)$ hat das Ziel die Funktion so zu zerlegen, dass sie durch eine Summe (Überlagerung) von Sinus- und Kosinusfunktionen approximiert wird (*Fourier-Reihe*):

$$g(t) = \sum_{k=-\infty}^{\infty} G_k \exp\left(\imath \frac{2\pi k}{P} t\right). \tag{3.1}$$

Dabei stellt jeder Summand obiger Reihe mit $G_k \neq 0$ eine Fourier-Frequenz $f_k = kf$, mit Grundfrequenz $f = 1/P$, der Funktion g dar (Schlittgen und Streitberg, 2001). Die Fourier-Koeffizienten G_k erhält man durch die Fourier-Transformation.

Die *Fourier-Transformation* $G = \{G_k\}$ an den Stellen $k \in \mathbb{N}_0$ einer stetigen und periodischen Funktion g mit Periodenlänge P ist definiert durch (Walker, 1996)

$$G_k = \frac{1}{P} \int_0^P g(t) \exp\left(-\imath \frac{2\pi k}{P} t\right) dt. \tag{3.2}$$

Den Zusammenhang zwischen der Exponentialdarstellung der Fourier-Reihe in Gleichung (3.1) und die erwähnte Überlagerung von Sinus- und Kosinusfunktionen erhält man unter Ausnutzung der *Eulerschen Formel* (Walker, 1996)

$$e^{\imath\varphi} = \cos\varphi + \imath \sin\varphi.$$

Durch das Sampling bei der digitalen Klangaufzeichnung werden die Schwingungen der Klänge nur an diskreten Zeitpunkten t, $t = 0, \ldots, T-1$ erfasst. Die Zeitreihe x_t ist daher zeitdiskret. Aus diesem Grund kann man das Integral in Gleichung (3.2) durch eine endliche Riemann Summe approximieren (siehe Walker, 1996). So erhält man anstatt einer stetigen Fourier-Transformation eine *diskrete Fourier-Transformation (DFT)* $X = \{X_k\}$, welche durch

$$X_k = \sum_{t=0}^{T-1} x_t \exp\left(-\imath \frac{2\pi k t}{T}\right)$$

definiert ist (Walker, 1996).

Da bei der Transformation einer endlichen Zeitreihe x_t, mit $t = 0, \ldots, T-1$ $X_{k+T} = X_k$ gilt, ist zur Rekonstruktion von x_t nur eine endliche Summe, anstatt der Reihe aus (3.1) notwendig. Damit lässt sich die ursprüngliche Zeitreihe durch

$$x_t = \frac{1}{T} \sum_{k=0}^{T-1} X_k \exp\left(\imath \frac{2\pi k t}{T}\right). \tag{3.3}$$

darstellen (Walker, 1996).

Zur Berechnung der diskreten Fourier-Transformation wird in der Praxis eine *schnelle Fourier-Transformation (FFT)* genutzt. Diese Algorithmen reduzieren den Rechenaufwand für die Berechnung der DFT dramatisch (Walker, 1996). Am bekanntesten ist der im folgenden Abschnitt beschriebene *Cooley-Tukey Algorithmus* (Cooley und Tukey, 1965).

3.2.1 Schnelle Fourier-Transformation

Der Cooley-Tukey Algorithmus reduziert den Rechenaufwand einer T Punkt DFT durch eine Halbierung der Länge der Sequenz in jedem von insgesamt $a = \log_2 T$ Schritten und anschließendem Zusammenfügen der Teilergebnisse. Durch die Anzahl der Schritte ergibt sich eine zwingende Voraussetzung an die Länge T der Sequenz. T muss eine Potenz von 2 betragen:

$$T = 2^a, \qquad a \in \mathbb{N}$$

Dies kann durch Auffüllen mit Nullen, *Zero Padding*, immer erreicht werden.

Rekursive Beschreibung des Algorithmus

Schritt 1: Für den ersten Schritt der FFT wird die T Punkt DFT $\{X_k\}$ folgendermaßen umgeschrieben:

$$\begin{aligned} X_k &= \sum_{t=0}^{T-1} x_t \exp\left(-\imath \frac{2\pi}{T} t k\right) \\ &= \sum_{t=0}^{\frac{1}{2}T-1} x_{2t} \exp\left(-\imath \frac{2\pi}{T} 2tk\right) + \sum_{t=0}^{\frac{1}{2}T-1} x_{2t+1} \exp\left(-\imath \frac{2\pi}{T} (2t+1)k\right) \\ &= \sum_{t=0}^{\frac{1}{2}T-1} x_{2t} \exp\left(-\imath \frac{2\pi}{T} 2tk\right) + \sum_{t=0}^{\frac{1}{2}T-1} x_{2t+1} \exp\left(-\imath \frac{2\pi}{T} 2tk\right) \exp\left(-\imath \frac{2\pi}{T} k\right) \end{aligned}$$

Definiert man nun $W := \exp\left(-\imath\frac{2\pi}{T}\right)$ und damit

$$X_k^0 := \sum_{t=0}^{\frac{1}{2}T-1} x_{2t}\left(W^2\right)^{tk}$$

sowie

$$X_k^1 := \sum_{t=0}^{\frac{1}{2}T-1} x_{2t+1}\left(W^2\right)^{tk}$$

so lässt sich die DFT $\{X_k\}$ schreiben als

$$X_k = X_k^0 + X_k^1 W^k. \tag{3.4}$$

Hierbei ist $\{X_k^0\}$ die DFT einer Sequenz, die sich aus den $T/2$ Punkten von $\{x_t\}$ zusammensetzt, die einen geradzahligen Index besitzen, und $\left\{X_k^1\right\}$ die DFT der $\{x_t\}$ mit ungeradzahligem Index. Da die Schrittlänge dieser Sequenzen $2/T$ anstatt $1/T$ bei $\{X_k\}$ beträgt, erfolgt die Berechnung dieser DFTs mit den Gewichten W^{2tk} anstatt W^{tk}. Der obere Index in der Bezeichnung dieser beiden DFTs ergibt sich aus der letzten Stelle der binären Darstellung gerader Zahlen (Null) und ungerader Zahlen (Eins).

Da $T = 2^a$ gilt, ist $T/2 \in \mathbb{N}$. Damit ergibt sich

$$W^{\frac{T}{2}} = \exp(-\pi\imath) = \cos(\pi) - \imath\sin(\pi) = \cos(\pi) = -1$$

Setzt man dies in (3.4) ein und nutzt die $T/2$ Periodizität der $\left\{X_k^0\right\}$ und $\left\{X_k^1\right\}$ (DFTs einer Sequenz der Länge $T/2$) aus, so ergibt sich:

$$\begin{aligned} X_{k+\frac{T}{2}} &= X^0_{k+\frac{T}{2}} + W^{k+\frac{T}{2}} H^1_{k+\frac{T}{2}} \\ &= X_k^0 + W^{\frac{T}{2}} W^k X_k^1 \\ &= X_k^0 - X_k^1 W^k \end{aligned}$$

Dies ermöglicht die effiziente Berechnung der X_k für $T/2 \leq k \leq T-1$ aus den X_k für $0 \leq k \leq T/2 - 1$.

Schritt 2: Im zweiten Schritt dieses FFT Algorithmus werden die beiden $T/2$ Punkt DFTs nach dem gleichen Schema in vier $T/4$ Punkt DFTs aufgeteilt. Es ergibt sich:

$$\begin{aligned} X_k^0 &= \sum_{t=0}^{\frac{1}{2}T-1} x_{2t}\left(W^2\right)^{tk} \\ &= \sum_{t=0}^{\frac{1}{4}T-1} x_{4t}\left(W^4\right)^{tk} + \sum_{t=0}^{\frac{1}{4}T-1} x_{4t+2}\left(W^4\right)^{tk} W^{2k} \end{aligned}$$

Durch Definition von

$$X_k^{00} := \sum_{t=0}^{\frac{1}{4}T-1} x_{4t} \left(W^4\right)^{tk}$$

und

$$X_k^{01} := \sum_{x=0}^{\frac{1}{4}T-1} x_{4t+2} \left(W^4\right)^{tk}$$

ergibt sich

$$X_k^0 = X_k^{00} + X_k^{01} W^{2k}.$$

Die X_k für $T/4 \leq k \leq T/2 - 1$ erhält man aus denen mit $0 \leq k \leq T/4 - 1$ durch

$$\begin{aligned} X_{k+\frac{T}{4}}^0 &= X_{k+\frac{T}{4}}^{00} + W^{2\left(k+\frac{T}{4}\right)} X_{k+\frac{T}{4}}^{01} \\ &= X_k^{00} + W^{\frac{T}{2}} W^{2k} X_k^{01} \\ &= X_k^{00} - X_k^{01} W^{2k}. \end{aligned}$$

Analog ergeben sich

$$\begin{aligned} X_k^1 &= X_k^{10} + X_k^{11} W^{2k} \\ X_{k+\frac{T}{4}}^1 &= X_k^{10} - X_k^{11} W^{2k}. \end{aligned}$$

Die oberen Indizes der $T/4$ Punkt DFTs ergeben sich aus der letzten und vorletzten Stelle der Binärdarstellung der Indizes der Sequenzpunkte von $\{x_t\}$, die der jeweiligen DFT zu Grunde liegen.

Durch rekursives Vorgehen erreicht man schließlich Schritt *a*.

Schritt *a*: In Schritt *a* werden T DFTs eines einzelnen Punktes berechnet. Diese ergeben sich auf triviale Weise durch

$$X_k = \sum_{t=0}^{1-1} x_t W^{tk} = x_0 W^0 = x_0, \qquad \forall k \in \mathbb{N}_0.$$

Die DFT eines einzelnen Punktes ist die Identität, es muss daher nichts mehr berechnet werden. Die Rekursion hat ihr Ende erreicht.

Eine weitere Beschleunigung der FFT kann durch hocheffiziente, an den jeweiligen Prozessor angepasste, Implementationen erreicht werden (Johnson und Frigo, 2008). Hierfür gibt es spezialisierte Softwarebibliotheken, wie zum Beispiel die *Fastest Fourier Transform in the West (FFTW)* (Frigo und Johnson, 2005).

3.2.2 Gefensterte Fourier-Transformation

Die Transformation in den Spektralraum durch eine Fourier-Transformation ermöglicht eine frequenzbasierte Analyse der Zeitreihe. Zeitliche Informationen gehen dabei aber vollständig verloren. Um die zeitliche Veränderung von Tönen und Klängen zur Strukturierung nutzen zu können, muss die Fourier-Transformation für einzelne Abschnitte getrennt berechnet werden. Man spricht hierbei von einer *gefensterten Fourier-Transformation* oder *Short Time Fourier Transformation (STFT)*. Hierbei wird durch Multiplikation der Zeitreihe mit einer Gewichtsfunktion (auch Fensterfunktion genannt) nur ein zeitlich begrenzter Teilabschnitt der Zeitreihe betrachtet. Diese Gewichtsfunktion ist nur innerhalb eines vorgegebenen Zeitabschnitts (Fenster) ungleich Null und löst so diesen Abschnitt aus der Zeitreihe heraus. Man erhält so einen *Frame* der STFT. Die Breite der Fensterfunktion muss abhängig von der gewünschten Anwendung gewählt werden. Sie sollte schmal genug sein, um die notwendige zeitliche Auflösung zu garantieren und innerhalb des Frames ein möglichst stationäres Signal zu erhalten. Gleichzeitig muss sie aber breit genug gewählt sein, um die notwendige Frequenzauflösung sicherzustellen.

Für die Wahl der Fensterfunktion gibt es eine Vielzahl von Möglichkeiten (Walker, 1996) mit unterschiedlichen Eigenschaften. Das Ideal ist eine Fensterfunktion, die das Signal der ursprünglichen Zeitreihe innerhalb des Zeitfensters so gering wie möglich verändert (z.B. Rechteckfenster), gleichzeitig aber möglichst glatte Übergänge an den Rändern aufweist um keine Artefakte im Spektrum durch den *Leck-Effekt* (Walker, 1996) zu erzeugen. Ein häufig genutzter Kompromiss zwischen diesen gewünschten Eigenschaften ist die *Hamming Fensterfunktion*, die als (Walker, 1996)

$$w_H(t) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi t}{T_W}\right), & 0 \leq t \leq T_W \\ 0 & \text{sonst} \end{cases}$$

definiert ist. Die Fensterweite ist damit T_W+1 Samples. Die Fensterweite in Sekunden erhält man dann durch

$$l = \frac{T_W + 1}{SR}.$$

Die Fensterweite stellt immer einen Kompromiss zwischen der Zeit- und Frequenzauflösung der gefensterten Fourier-Transformation dar. Ist die Schwingungsdauer einer Frequenz länger als die Fensterweite breit, können diese Frequenzen durch die gefensterte Fourier-Transformation nicht erkannt werden. Ist dagegen das Fenster zu lang, werden sehr schnelle Veränderungen des Klangs nicht abgebildet.

Durch schrittweises Verschieben des Fensters um die Schrittlänge T_S wird die gesamte Zeitreihe abgedeckt. Die Fourier-Koeffizienten des Frames s (Zeitpunkt $s \cdot {}^{T_S}/_{SR}$ in

der Zeitreihe/Tonaufnahme) der STFT lässt sich dann in folgender Form schreiben:

$$X(s,k) = \sum_{j=0}^{T} w(j)\, x_{s\cdot T_S+j} \exp\left(-2i\pi j \frac{k}{T}\right) = \sum_{j=0}^{T_W} w(j)\, x_{s\cdot T_S+j} \exp\left(-2i\pi j \frac{k}{T}\right), \tag{3.5}$$

da $w(j) = 0$ für $T_W + 1 \leq j \leq T$. Die DFT Länge T ist hierbei so gewählt, dass eine Berechnung mit Hilfe der FFT möglich ist. Durch das stückweise Verschieben des Zeitfensters über die gesamte Zeitreihe erhält man für jeden Frame einen Satz DFT-Koeffizienten, welcher die Frequenzen und Amplituden des Klangs innerhalb dieses Zeitabschnitts repräsentiert. Die Schrittweite der Verschiebung wird bei der Charakterisierung von Klängen oft so gewählt, dass sich die einzelnen Fenster überlappen. Möchte man dagegen die Tonaufnahme in zeitliche Abschnitte unterteilen, sind nicht überlappende Fenster zu bevorzugen. Für die Klangcharakterisierung beträgt die Fensterlänge 25 ms bei einer Schrittweite von 10 ms. Die Fensterlänge wird hier zeitlich festgelegt, sodass sie unabhängig von der Samplingrate SR ist. Bei einer Samplingrate von 44 100 Hz entsprechen 25 ms gerundet 1103 Samples. Für die schnelle Fourier Transformation werden diese Samples in ein auf die nächste Potenz von 2 erweitertes Fenster, also 2048, eingebettet. Das Auffüllen erfolgt mit Nullen (*Zero Padding*).

Für die zeitliche Strukturierung von längeren Tonaufnahmen ist die Fensterlänge deutlich verlängert worden. Es sind nicht überlappende Fenster von 3 s Länge (132300 Samples bei 44 100 Hz) gewählt worden. Da für die weiteren Schritte der Merkmalsextraktion die Phaseninformation der komplexwertigen DFT-Koeffizienten nicht von Relevanz ist, werden diese durch ihr Betragsquadrat ersetzt. Man erhält so das *Power Spektrum* jedes einzelnen Frames.

3.3 Klangmerkmale

Psychoakustische Frequenztransformation

Aus der STFT erhält man für jeden Zeitframe ein sehr detailliertes Frequenzspektrum. Dieses beinhaltet zum einen viele redundante Daten (eng zusammenliegende Frequenzen beinhalten wenig zusätzliche Informationen) und zum anderen sind die Frequenzbereiche, die als „eng zusammenliegend“ gelten, nicht über das gesamte Spektrum gleich breit. Um daher bei der schrittweisen Dimensionsreduktion möglichst wenig zur Unterscheidung von Klängen relevante Informationen des Spektrums zu verlieren, werden die DFT-Frequenzen auf die psychoakustisch motivierte Melskala (siehe Definition 2.2) transformiert. Auf der Melskala werden dann äquidistant 40

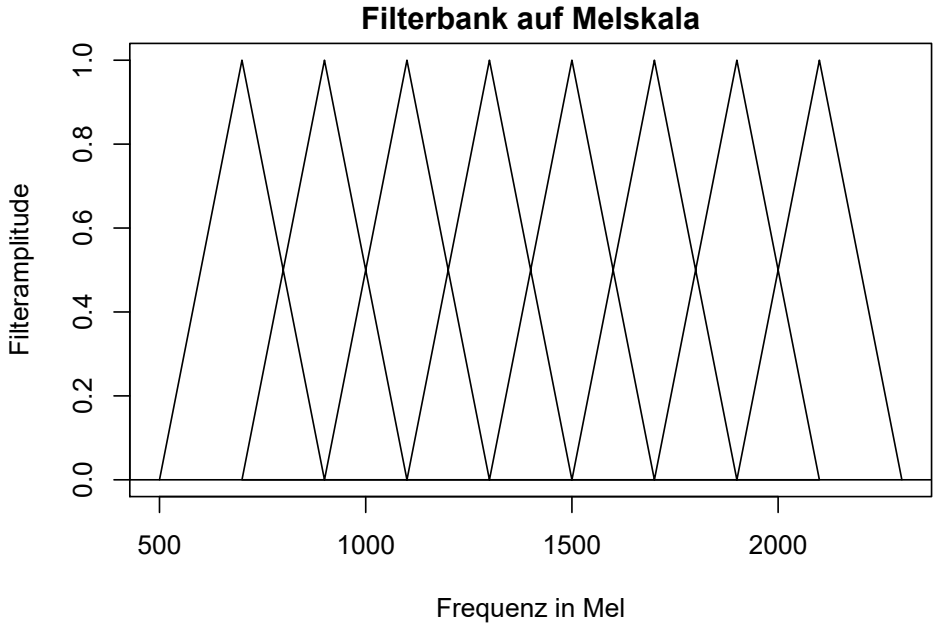

Abbildung 3.2: Eine Filterbank zur Frequenztransformation mit 8 Filtern auf der Melskala

dreieckförmige Bandpassfilter zur Zusammenfassung der Fourier-Frequenzen aufgespannt (Slaney, 1998; Childers, Skinner und Kemerait, 1977). Die Frequenzbänder der Bandpässe sind so überlappend gewählt, dass die Kombination aller Filter wieder zu einem linearen Frequenzgang führt. Das Ansprechverhalten der Filter ist für acht Filter über eine Bandbreite von 500 Mel bis 2300 Mel in den Abbildungen 3.2 und 3.3 skizziert.

Durch diese Zusammenfassung der Fourier-Frequenzen erhält man für jeden Frame j einen Vektor $(M_k(j))_{k=1}^{40}$ von 40 zusammengefassten Fourier-Koeffizienten. Die Zusammenfassung erfolgt durch Bildung einer mit der Frequenzkurve des Bandpassfilters gewichteten Summe der einzelnen Fourier-Frequenzen.

Lautstärkenkompression

Die Lautstärkenkompression der MFCCs ist direkt durch die Definition des Schalldruckpegels (siehe Gleichung 2.1) motiviert. Die nach der Frequenztransformation zusammengefassten Fourier-Koeffizienten $M_k(j)$ werden logarithmiert um eine dem Lautstärkeempfinden angepasste, linearere Skala zu erhalten. Man erhält damit

$$L_k(j) = \log M_k(j)$$

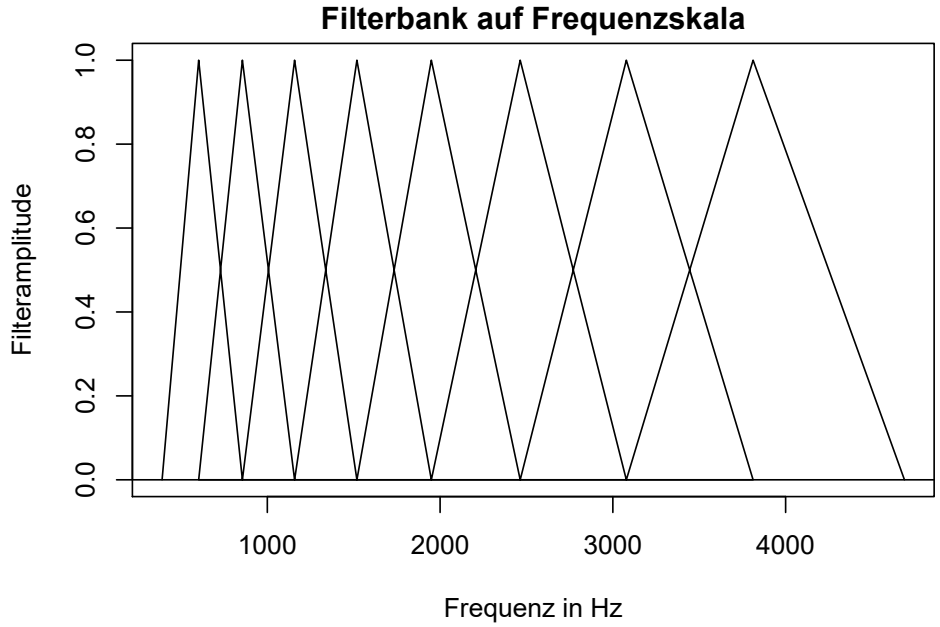

Abbildung 3.3: Eine Filterbank zur Frequenztransformation mit 8 Filtern auf der Frequenzskala

3.3.1 Mel Frequency Cepstral Coefficients (MFCC)

Um die Informationen des Klangs möglichst komprimiert darzustellen, sollen bei den MFCCs die einzelnen Koeffizienten untereinander möglichst unkorreliert sein. Für Frequenzen, die ein ganzzahliges Vielfaches einer festen Grundfrequenz sind, bilden die Kosinusfunktionen mit den jeweiligen Frequenzen ein orthonormales System von Eigenfunktionen (Walker, 1996) des Raums der quadratintegrierbaren 2π-periodischen Funktionen. Damit erfüllen die Komponenten einer Fourier-Kosinusreihe die Anforderung der Unkorreliertheit. Die diskrete Kosinusreihe einer reellwertigen Sequenz x_t der Länge N ist durch

$$x_t = \frac{1}{N} X_0^C + \frac{2}{N} \sum_{k=1}^{N-1} X_k^C \cos\left(\frac{\pi k}{N} t\right)$$

definiert (Walker, 1996). Die X_k^C erhält man durch eine *Diskrete Kosinustransformation (DCT)*, welche durch

$$X_k^C = \sum_{t=0}^{N-1} x_t \cos\left(\frac{\pi t k}{N}\right)$$

definiert ist (Walker, 1996). Die DCT ist ein nur für reellwertige x_t definierter Spezialfall der *DFT*.

Das transformierte Spektrum $L_k(j)$ des Klangs in jedem Zeitfenster j kann daher durch eine Summe von Kosinusschwingungen approximiert werden. Die dafür zu berechnende DCT beschreibt das Spektrum des Spektrums und befindet sich daher wieder in der Zeitdomäne. Aufgrund der Unkorreliertheit des Systems von Kosinusfunktionen und der mit k abnehmenden Größe der X_k^C, kann durch ausschließliche Berücksichtigung der X_k^C mit $k \leq p$ eine weitere Dimensionsreduktion erreicht werden. Gebräuchliche Werte für p liegen zwischen $p = 12$ und $p = 20$ (Ellis, 2005; Childers, Skinner und Kemerait, 1977). Hier wurde mit $p = 16$ gearbeitet.

Den Merkmalsvektor m_i eines Frames i erhält man daher aus den DCT-Koeffizienten $X_{i,k}^C$ durch $m_i = (X_{i,k}^C)_{i=1}^p$.

Alternativ zur DCT könnte auch eine Hauptkomponentenanalyse (PCA) durchgeführt werden. Als Merkmale wären dann die Eigenwerte der ersten Hauptkomponenten weiterzuverwenden. Logan (2000) hat gezeigt, dass die DCT eine gute Approximation der PCA im Falle der Musiksignalanalyse liefert. Effiziente Implementierungen zur Berechnung der DCT sind weit verbreitet, weshalb die DCT bei der Berechnung von MFCCs vorgezogen wird.

3.3.2 Perceptual Linear Prediction (PLP)

Die *Perceptual Linear Prediction (PLP)* (Hermansky, 1990) Koeffizienten sind eine weitere Möglichkeit Klänge zu charakterisieren. Ausgehend von den auf die Melskala transformierten Frequenzbändern der STFT werden weitere Berechnungsschritte durchgeführt um eine Gewichtung der verschiedenen Frequenzbänder gemäß ihrer Bedeutung für den Klang zu erreichen.

Zuerst erfolgt eine frequenzabhängige Laustärkenkorrektur durch Multiplikation der Amplituden der 40 Frequenzbänder mit Hermanskys Korrekturfaktor (Hermansky, 1990):

$$L(f) = \left(\frac{f^2}{f^2 + 160\,000}\right)^2 \cdot \frac{f^2 + 1\,440\,000}{f^2 + 9\,610\,000},$$

wobei f die Mittenfrequenz des Frequenzbands ist.

Anschließend erfolgt eine Lautstärkenkompression durch eine Kubikwurzel Transformation, um die Dominanz von hohen Amplituden zu reduzieren und so eine bessere Differenzierung im niedrigen Amplitudenbereich zu erreichen.

Anschließend werden die transformierten Frequenzen mit einer Inversen Fourier-Transformation zurück in die Zeitdomäne transformiert. An die so erhaltene Zeitreihe

wird ein autoregressives Modell für jeden Zeitframe angepasst:

$$y_t = \sum_{j=1}^{p} a_j y_{t-j} + e_t,$$

wobei die Fehlerterme e_t weißes Rauschen sind.

Die erhaltenen AR Koeffizienten $(a_1, \ldots, a_p)$ bilden dann den Merkmalsvektor jedes einzelnen Zeitframes. Der Parameter p (Lag des AR Modells) steuert die Komplexität. Ein typischer Wert im Bereich der Sprachsignalanalyse ist $p = 8$. Dieser wurde auch hier gewählt, da mit höheren Werten keine Verbesserung der Klassifikationsergebnisse erreicht werden konnte.

4 Visualisierung

Für die Betrachtung der Struktur einer Audioaufnahme ist eine Visualisierung der Features nützlich. Die Darstellung der paarweisen Distanzen der Feature Vektoren als Heatmap bietet sich hier als eine optisch sehr übersichtliche Möglichkeit an. Diese Darstellungsform wird als *Self Distance Matrix* bezeichnet (Paulus, Müller und Klapuri, 2010). Je dunkler die Farbe eine Punktes (m, n) dieser Heatmap ist, desto ähnlicher sind die entsprechenden Feature Vektoren x_m und x_n. Ähnliche Feature Vektoren bedeuten einen ähnlichen Klang. Dies ermöglicht Strukturen in einer Tonaufnahme zu erkennen. Bestimmte Abschnitte eines Musikstücks (z.B. Strophe, Refrain bei populärer Musik) zeichnen sich z.B. durch charakteristische Instrumentierung, melodische Elemente oder Dynamik aus. Dies führt zu einem typischen Klangbild. Dieses Klangbild bleibt auch bei Wiederholung dieser Abschnitte meist erhalten. Daher lassen sich diese ähnlich klingenden Phasen eines Musikstücks durch dunkle Blöcke in der Heatmap identifizieren. Auf der zeitlich sehr viel kürzeren Ebene einzelner Klänge funktioniert dies auch um z.B. stationäre Phasen (Haltephase/Sustain, evtl. auch mit vorhandenem Vibrato) von transienten Klangphasen (Anschlag/Attack, Ausklingen/Decay) zu unterscheiden.

Zur Quantifizierung der Ähnlichkeit kommen verschiedene Distanzmaße in Frage. Neben den klassischen p-Normen (Manhatten-, Euklidische Distanz, etc.) hat sich die Kosinus-Distanz als ein bewährtes Maß zur Bestimmung der (Un-)Ähnlichkeit zweier Merkmalsvektoren (Paulus, Müller und Klapuri, 2010) bewährt:

$$d_c(x_m, x_n) = \frac{1}{2}\left(1 - \frac{< x_m, x_n >}{||x_m||\,||x_n||}\right) \tag{4.1}$$

Der Quotient in Formel 4.1 (Skalarprodukt von x_m und x_n dividiert durch das Produkt der Längen der beiden Vektoren) ist das Kosinusähnlichkeitsmaß und misst den Kosinus des von den beiden Merkmalsvektoren eingeschlossenen Winkels. Dieser Wert liegt zwischen -1 für Merkmalsvektoren, die in entgegengesetzte Richtungen weisen und 1 für Vektoren, die in die gleiche Richtungen weisen. Für orthogonale (unkorrelierte) Merkmalsvektoren ist das Kosinusähnlichkeitsmaß gleich Null. Durch die Normierung ergibt sich damit für entgegengesetzte Merkmalsvektoren die Maximale Distanz von 1, für Vektoren in gleiche Richtung die minimale Distanz von 0 und für orthogonale Vektoren eine Distanz von $1/2$.

Die Kosinusdistanz wird auch aus folgendem Grund gerne eingesetzt: Für die Eukli-

dische Distanz $\|\cdot\|_2$ zweier n-dimensionaler Merkmalsvektoren A und B gilt:

$$\|A-B\|_2^2 = \sum_{i=1}^{n}(A_i-B_i)^2 = (A-B)'(A-B) = \|A\|_2^2+\|B\|_2^2-2A'B = \|A\|_2^2+\|B\|_2^2-2\cos(A,B)\,\|A\|_2\,\|B\|_2$$

Das heißt für auf Länge 1 normierte Vektoren A und B gilt:

$$\|A-B\|_2^2 = 1+1-2\frac{\cos(A,B)}{1\cdot 1} = 2\,(1-\cos(A,B))$$

Damit sind die euklidische Distanz und die Kosinusdistanz bis auf einen konstanten Skalierungsfaktor gleich.

Anschaulich bedeutet dies, dass die euklidischen Distanz neben der Lage zueinander (eingeschlossener Winkel) auch die Länge der Vektoren berücksichtigt. Die Kosinusdistanz ignoriert dagegen die Länge der Vektoren. Nur der eingeschlossene Winkel und damit die Lage der Vektoren zueinander bestimmt die Distanz. Dies ermöglicht bei Klangmerkmalen, die sich bei unterschiedlicher Lautstärke nur im Betrag und nicht im eingeschlossenen Winkel unterscheiden, eine sinnvolle Distanzberechnung ohne vorherige Lautstärkenormalisierung.

Ferner gilt, dass die Verteilung der Kosinusähnlichkeit von auf der Einheitssphäre gleichverteilten n-dimensionalen Zufallsvektoren gegen eine Normalverteilung mit Erwartungswert 0 und Varianz $1/n$ strebt (Spruill, 2007).

Die Einträge D_{mn} der Distanzmatrix D erhält man, indem man alle paarweisen Distanzen $d(x_m, x_n)$ zweier Merkmalsvektoren x_m und x_n berechnet und $D_{mn} = d(x_m, x_n)$ setzt. Für die Distanzmatrix der Kosinusdistanzen D^c gilt dementsprechend $D^c_{mn} = d_c(x_m, x_n)$.

Die Abbildung 4.1 zeigt die Visualisierung des Depeche Mode Songs „Stripped“ vom Album „Black Celebration“ basierend auf MFCCs. Jedes Pixel (m, n) dieser Grafik zeigt die Ähnlichkeit der MFCC Featurevektoren x_m und x_n, wobei ein Featurevektor auf einem Zeitfenster (oder Zeitframe) der gefensterten Fouriertransformation berechnet worden ist. Es sind also alle Schritte der Vorverarbeitung (Preemphasis Filterung, Fourier Transformation, psychoakustische Transformationen) und die Merkmalsextraktion der MFCCs durchlaufen worden. Wie in Kapitel 3.2.2 beschrieben, ist für die Analyse kompletter Musikstücke eine Fensterlänge von 3 s gewählt worden. Je dunkler bzw. violetter das Pixel ist, desto ähnlicher ist der Klang der zugehörigen Zeitframes. Unterscheiden sich verschiedene Phasen des Musikstücks, wie im vorliegenden Fall, sehr deutlich, so zeichnen sich diese Phasen als dunkle, quadratische Blöcke entlang der Hauptdiagonalen ab. Findet man ausgehend von solch einem Block ähnlich dunkle Blöcke noch einmal abseits der Hauptdiagonalen (horizontal

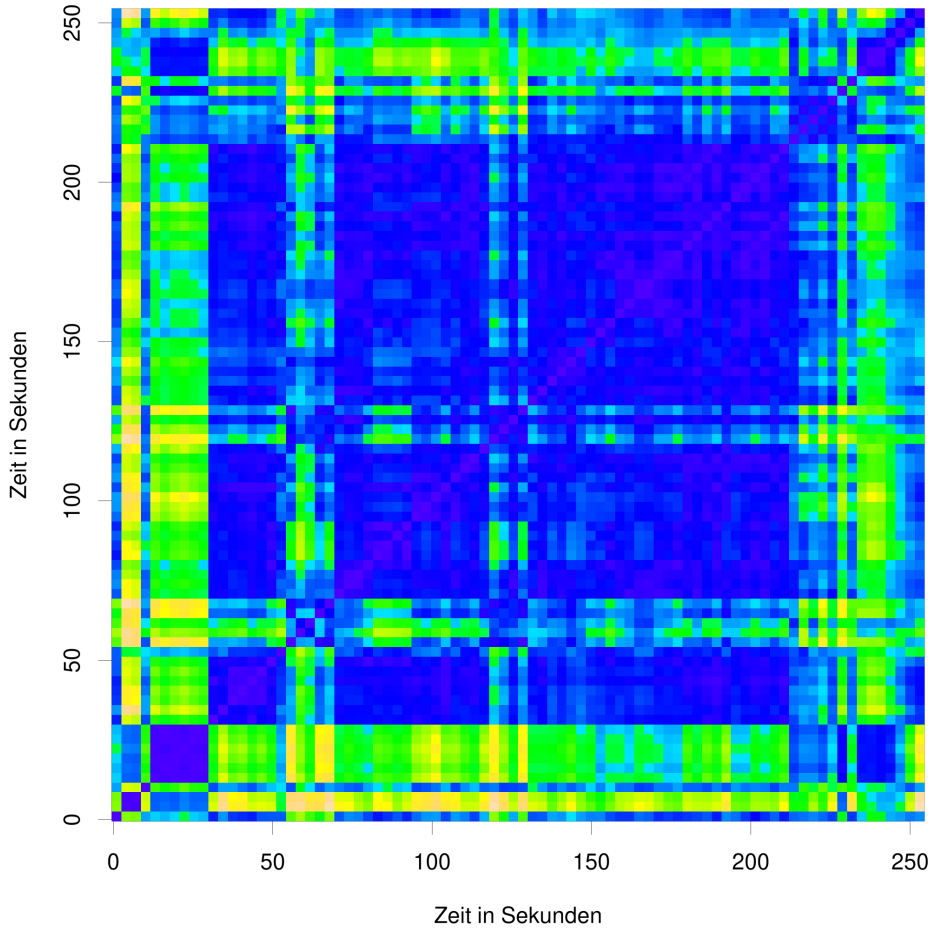

Abbildung 4.1: Self Distance Matrix der MFCCs des Depeche Mode Songs „Stripped“

oder vertikal), so wiederholt sich der Klang und damit diese Phase innerhalb des Musikstücks (z.B. der Refrain). Damit ist diese Darstellung sehr hilfreich bei der optischen Identifizierung von interessanten Bereichen innerhalb einer Tonaufnahme, für die sich eine tiefergehende Analyse lohnt.

5 Klangdaten

5.1 McGill Instrumentendatenbank

Die in den Kapiteln 7 bis 9 dieser Arbeit vorgestellten bzw. für diese Arbeit entwickelten Methoden werden am Beispiel der Tonaufnahmen der *McGill Instrumentendatenbank* (Opolko und Wapnick, 1987) präsentiert. Daher wird im folgenden Abschnitt diese Datenbank vorgestellt.

Die hier verwendete Fassung der McGill Datenbank besteht aus 1986 Tonaufnahmen von einzelnen Instrumentennoten, die $3-5$ Sekunden lang sind. Es liegen Aufnahmen von insgesamt 38 unterschiedlichen Instrumenten in unterschiedlichen Klangfarben vor, die durch unterschiedliche Spielweisen erzeugt wurden (Streichen, Zupfen, mit/ohne Vibrato, Einsatz von Verzerrung (bei der E-Gittare), etc.). Für jede dieser 60 Klangfarben liegen 6 bis 88 Aufnahmen über das gesamte Tonhöhenspektrum des Instruments vor.

Das Samplingformat der Aufnahmen ist 16 bit bei 44 100 Hz (CD-Qualität), wobei aber nur ein Kanal aufgezeichnet worden ist. Es handelt sich also um *mono* Aufnahmen. Die hohe Qualität der Tonaufnahmen hat diesen Datensatz zu einem Referenzdatensatz für die Erforschung und Entwicklung von Verfahren zur Musiksignalanalyse werden lassen.

Für die Klassifikation der Instrumentenklänge (überwachtes maschinelles Lernen) werden aus diesem Datensatz zwei unterschiedlich schwere Klassifikationsprobleme erzeugt. Das schwierigere Problem umfasst bis auf die Slapping und Popping Klänge des E-Basses alle Klangfarben der McGill Datenbank. Es handelt sich also um ein 59 Klassen Klassifikationsproblem. Durch zufälliges Raten beträgt daher die Erfolgswahrscheinlichkeit weniger als 1,7%.

Für das zweite Klassifikationsproblem werden die Klangfarben der McGill Datenbank zu 25 Instrumentenfamilien (Trompeten, Streicher gestrichen gespielt, Flöten, etc.) zusammengefasst. Dieses Klassifikationsproblem ist deutlich einfacher, aber die Erfolgswahrscheinlichkeit für zufälliges Raten ist mit 4% immer noch sehr gering.

Erste Versuche zur Zeitreihensegmentierung (siehe Kapitel 9.2) erfolgten mit Tonaufnahmen, die aus jeweils 10 aneinander gehängter McGill Tonaufnahmen bestehen,

welche zufällig mit Zurücklegen aus der Menge aller Tonaufnahmen gezogen wurden.

5.2 Musikstücke

Die Visualisierung von Klangmerkmalen sowie die Strukturierung längerer Musikstücke erfolgt beispielhaft an den Musikstücken „Queen - Bohemian Rhapsody“ sowie „Depeche Mode - Stripped“, da sie eine große Vielfalt an unterschiedlich arrangierten Teilstücken aufweisen, deren Unterteilung unterschiedliche Schwierigkeitsgrade aufweisen.

6 Clustering

Clusterverfahren sind Methoden des unüberwachten statistischen Lernens, die dazu dienen automatisiert Gruppen von ähnlichen Datenpunkten (*Cluster*) in einem Datensatz zu finden. Die Ähnlichkeit wird über ein Distanzmaß bestimmt und für jeden dieser Cluster wird ein Zentrum bestimmt. Je nach gewähltem Clusterverfahren ist dies ein Mittelwert aller Datenpunkte des Cluster (z.B. arithmetisches Mittel bei *k-Means*) oder ein ausgewählter repräsentativer Datenpunkt. Das so bestimmte Clusterzentrum dient dann als Repräsentant für alle Datenpunkte des Clusters. Man kann so einen großen Datensatz strukturieren und auf wenige Clusterzentren reduzieren, die dann die Basis für weitere Verarbeitungs- und Analyseschritte bilden.

Prinzipiell lassen sich Clusterverfahren in zwei Hauptgruppen unterteilen. Agglomerative Clusterverfahren gehen von den einzelnen Datenpunkten aus und fügen schrittweise ähnliche Datenpunkte bzw. Cluster zusammen. Bei divisiven Verfahren wird ausgehend vom kompletten Datensatz eine Aufteilung nach zuvor festgelegten Kriterien durchgeführt.

In der Musiksignalanalyse werden Tonaufnahmen in sehr kurze und sich überlappende Zeitfenster unterteilt. Für jedes dieser Zeitfenster werden dann Merkmalsvektoren extrahiert. Man erhält so einen sehr großen Datensatz von Merkmalsvektoren mit einer hohen Autokorrelation. Für die Unterscheidung oder die Strukturierung von Klängen ist die hohe Zahl von Merkmalsvektoren nicht hilfreich. Eine Zusammenfassung zu repräsentativen Clusterzentren verringert die Anzahl der Fehlklassifikationen bei der Klangerkennung, wie in Kapitel 8 dargestellt.

6.1 k-Means

Dieses Unterkapitel basiert auf den Ergebnissen aus Ligges und Krey (2011).

Das bekannteste Clusterverfahren ist das *k-Means Clustering*. Das Optimalitätskriterium ist hierbei die Minimierung der Qudratsumme innerhalb der Cluster, also die Minimierung der Summe der quadrierten Abstände der Datenpunkte der jeweiligen Cluster von den zugehörigen Clusterzentren $\mu^{(i)}$. Für eine vorher festgelegte Clusteranzahl $k \leq n$ und eine Partitionierung $C = \{C_1, C_2, \ldots, C_k\}$ des Datensatzes lässt sich

das Optimierungsproblem so formulieren:

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{Z \in C_i} \left\| Z - \mu^{(i)} \right\|^2 \tag{6.1}$$

Eine alternative Formulierung ist die Minimierung von

$$SSD\left(\bigcup_{l=1}^{k} C_l \right) = \sum_{l=1}^{k} \sum_{Z \in C_l} \sum_{j=1}^{p} \left(Z_j - m_j^{(l)} \right)^2, \tag{6.2}$$

wobei Z_j das Element j des Merkmalsvektors Z (aus Cluster C_k) und $m_j^{(k)}$ das Element j des Clusterzentrums $m^{(k)}$ des k-ten Clusters C_k ist.

Der bekannteste Algorithmus für eine schnelle approximative Lösung dieses *np*-harten Optimierungsproblems ist der Lloyd Algorithmus (Lloyd, 1957, 1982). Hierbei werden k zufällige Datenpunkte als initiale Clusterzentren ausgewählt, anschließend wird jeder Datenpunkt dem Clusterzentrum zugeordnet, das den geringsten euklidischen Abstand zu diesem aufweist. Anschließend werden die Clusterzentren als arithmetisches Mittel der zugeordneten Datenpunkte neu bestimmt und die Zuordnung der Datenpunkte zu den neuen Clusterzentren wiederholt. Die letzten beiden Schritte werden so lange wiederholt, bis die Clusterzuordnung stabil bleibt.

Dieses Verfahren konvergiert sehr schnell, findet aber meist nur ein lokales Optimum. Zusätzlich ist es sehr anfällig für eine ungeschickte Wahl der initialen Clusterzentren. Durch mehrere Läufe mit unterschiedlichen Datenpunkten als Startwerte für die Clusterzentren kann man die Chance für das Auffinden des globalen Optimums erhöhen, es gibt aber keine Garantien. Für die Lösung dieses Optimierungsproblems gibt es noch verschiedene andere Algorithmen wie z.B. Hartigan und Wong (1979) und MacQueen (1967), die aber alle ähnliche Schwächen aufweisen.

Trotz der genannten Schwächen funktioniert k-Means Clustering in vielen Bereichen sehr gut und der Lloyd Algorithmus konvergiert sehr schnell. Bei der Clusterung der Aufnahmen einzelner Instrumentklänge aus der McGill Datenbank (Opolko und Wapnick, 1987) werden insb. Saiteninstrumente sehr gut geclustert. Details zum Vorgehen befinden sich in Kapitel 8.1.2. In Abbildung 6.1 ist die Clusterung eines Klaviertons in 4 Cluster zu sehen. Die so erhaltenen Cluster sind als Anschlag- (Attack), Halte- (Sustain) und Ausklingphase (Decay) sowie Stille mit leichtem Rauschen am Anfang und Ende der Tonaufnahme zu interpretieren. Durch die sehr klare Clusterung können die Bezeichnungen für die Cluster der Klangphasen einfach in der Reihenfolge des Auftretens vergebeben werden. Die Anschlagphase ist eine Phase mit hoher Energie, die zusätzliche Obertöne durch den Anschlag mit dem Hammer

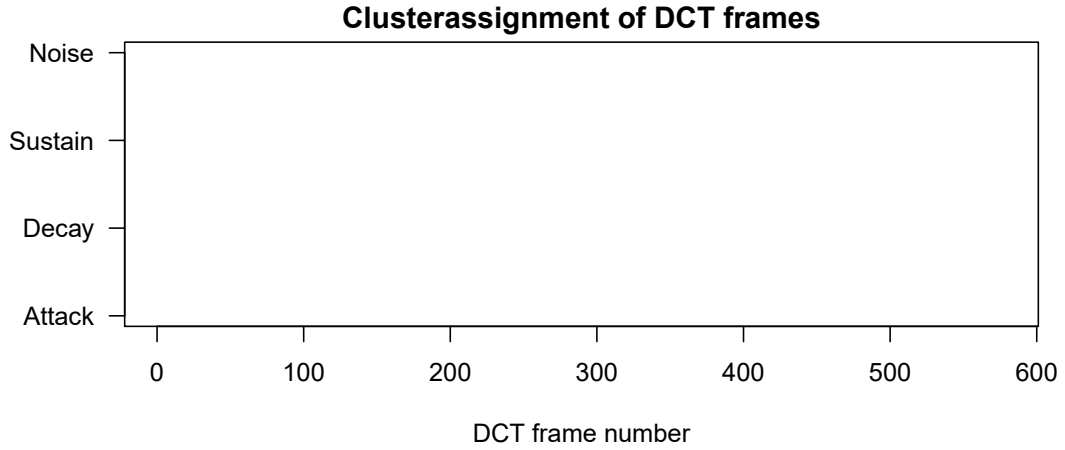

Abbildung 6.1: Clusterung eines Klaviertons

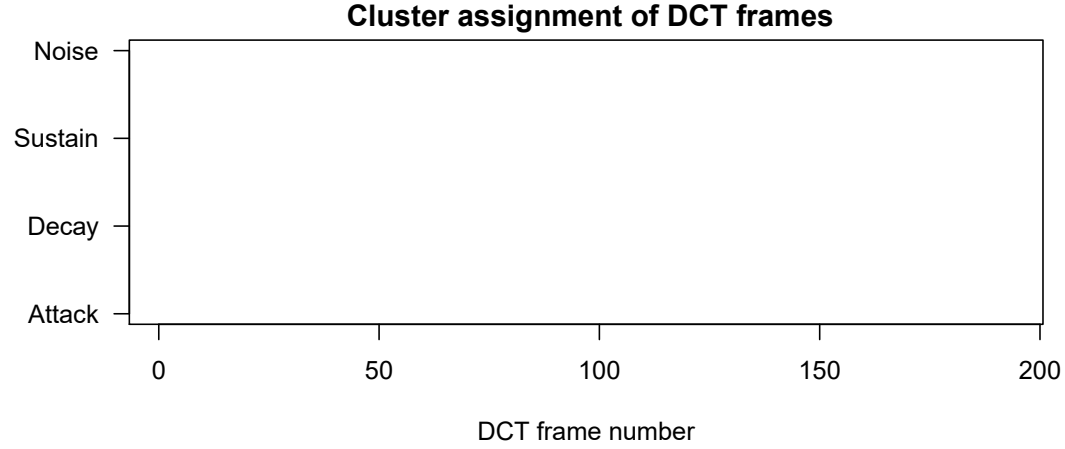

Abbildung 6.2: Clusterung eines gestrichen gespielten Bratschentons

aufweist. In der Haltephase sind diese Obertöne abgeklungen, bevor dann in der Ausklingphase die Energie abnimmt.

Für Streichinstrumente und einige Bläser werden ebenfalls hervorragende Clusterungen erzielt. Dies kann am Beispiel einer Bratsche (Abbildung 6.2) und eines Altsaxophons (Abbildung 6.3) nachvollzogen werden. Für den gestrichen gespieltem Bratschenton ist der mit *Attack* bezeichnete Cluster zweigeteilt, da sich der Klang beim Beschleunigen und Verlangsamen des Bogens ähnlich verhält und danach der Ton erst in die Ausschwingphase übergeht.

Bei einigen Blasinstrumenten sind die Ergebnisse aber meist sehr viel schlechter. Durch sehr kurze Anblas- und Ausklingphasen werden die Cluster bei einer zufälligen Initialisierung oft nicht erfasst. Bei den so entstehenden Clusterungen fehlen diese Phasen dann und die Haltephase wird künstlich in unterschiedliche sehr eng zusammenliegende Cluster aufgeteilt. Die Zuordnung zu den Clustern ist dann künstlich

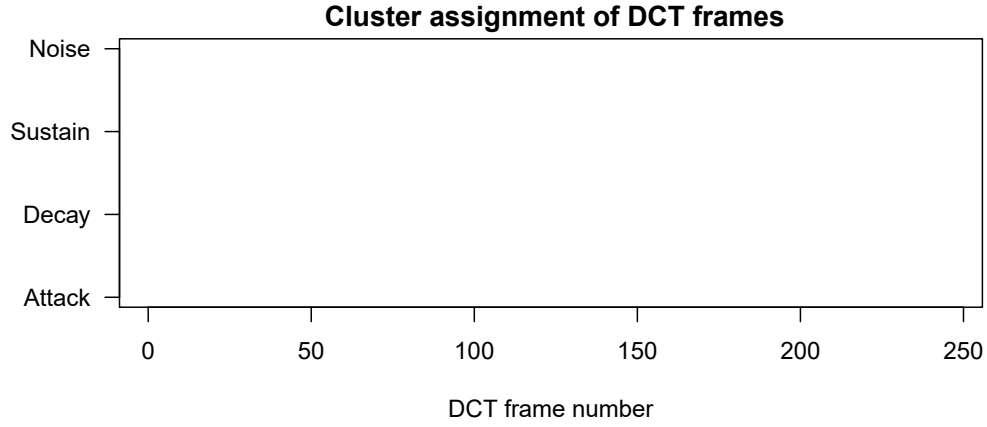

Abbildung 6.3: Clusterung eines Altsaxophontons

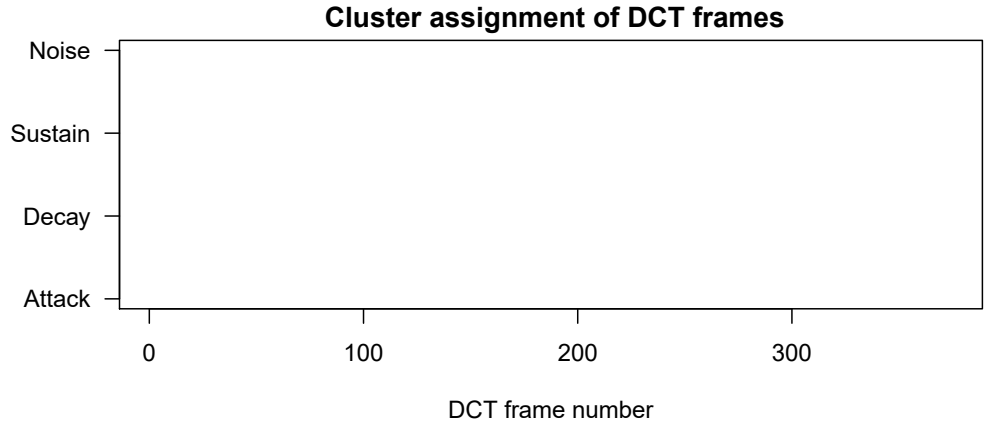

Abbildung 6.4: Clusterung eines Kontrafagotttons

und zufällig, was dazu führt, dass keine interpretierbare Clusterung mehr möglich ist. Dies kann man für eine Note eines Kontrafagotts in Abbildung 6.4 sehen. Zusätzlich sind diese Clusterungen oft nicht stabil und jede Ausführung des Algorithmus kann zu einem anderen Ergebnis führen. Abbildung 6.5 zeigt solch unterschiedliche Ergebnisse bei einem anderen Kontrafagottton.

6.2 Neural Gas

Ein alternativer Clusteralgorithmus ist *Neural Gas* (Martinetz, Berkovich und Schulten, 1993). Dieses Verfahren versucht ebenfalls die quadrierte euklidische Distanz der Datenpunkte zu ihren Clusterzentren zu minimieren, nutzt hierzu aber einen anderen Algorithmus, der in jedem Schritt alle Clusterzentren anpasst und diese Anpassung mit Fortschreiten des Verfahrens Schritt für Schritt verkleinert. Dies

sorgt für eine stabilere Konvergenz und reduziert die Abhängigkeit von der Initialisierung.

Ausgehend von einer Initialisierung des Verfahrens mit k unterschiedlichen Clusterzentren wird in jedem Schritt t des Algorithmus ein zufälliger Datenpunkt x ausgewählt und die Distanz zu den k Clusterzentren $\mu_1, \ldots, \mu_k$ bestimmt. Bezeichnet j_0 dann den Index des nächstgelegenen Clusterzentrums und j_{k-1} das am weitesten entfernte Clusterzentrum, kann man mit diesen j_m die neuen Clusterzentren für den Schritt $t+1$ des Algorithmus bestimmen:

$$\mu_{j_m}^{t+1} = \mu_{j_m}^{t} + \varepsilon(t)h(t,m)(x - \mu_{j_m}^{t}),$$

wobei $\varepsilon(t) \in [0,1]$ eine in t abnehmende Funktion ist. $h(t,m) \in [0,1]$ nimmt in t und m ab und $\forall t : h(t,0) = 1$. So erreicht man, dass die Änderung der zu dem Datenpunkt näher liegenden Clusterzentren stärker ist als die von weiter entfernt liegenden. Zusätzlich nimmt die Stärke der Änderung mit dem Voranschreiten des Algorithmus ab. Martinetz, Berkovich und Schulten zeigen, dass die Konvergenz des Verfahrens zwar langsamer als bei den klassischen k-Means Algorithmen ist, Neural-Gas aber regelmäßig das globale Optimum deutlich näher erreicht.

In Abbildung 6.6 ist die mit Neural Gas erhaltene stabilere Clusterung des selben Kontrafagotttons zu sehen, der mit k-Means nicht stabil zu Clustern ist (siehe Abbildung 6.5).

Die Cluster 2 und 3 sind im Vergleich zur k-Means Clusterung vertauscht. Die beiden im k-Means Clustering durch das durchgehende Alternieren zwischen diesen nicht interpretierbaren großen Cluster 1 und 4 wurden so getrennt, dass Cluster 4 der mittlere Bereich abgedeckt und Cluster 1 den Anfang und das Ende dieser Klangphase. Dies ist deutlich besser zu interpretieren. Es gibt aber weiter Bereiche der Überlappung zwischen den verschiedenen Clustern, die keine eindeutige Interpretation der entsprechenden Phasen ermöglichen.

Zusammenfassung Auch wenn die Ergebnisse der Clusterung nicht immer optimal sind, charakterisieren die Clusterzentren die Instrumentenklänge in den Tonaufnahme sehr gut. Sie ermöglichen so eine gute Erkennung der Instrumente mit Hilfe von Klassifikationsverfahren. Dies ist in Kapitel 8 beschrieben.

Im Kapitel 9 wird eine zusätzliche Restriktion in das Clusterverfahren eingeführt. Durch diese Nebenbedingung werden die Clusterergebnisse auch bei den schwierigeren Instrumentenklängen so stabilisiert, dass eine weitere Verbesserung der Klassifikationsleistung erreicht werden kann (siehe Kapitel 9.4.2).

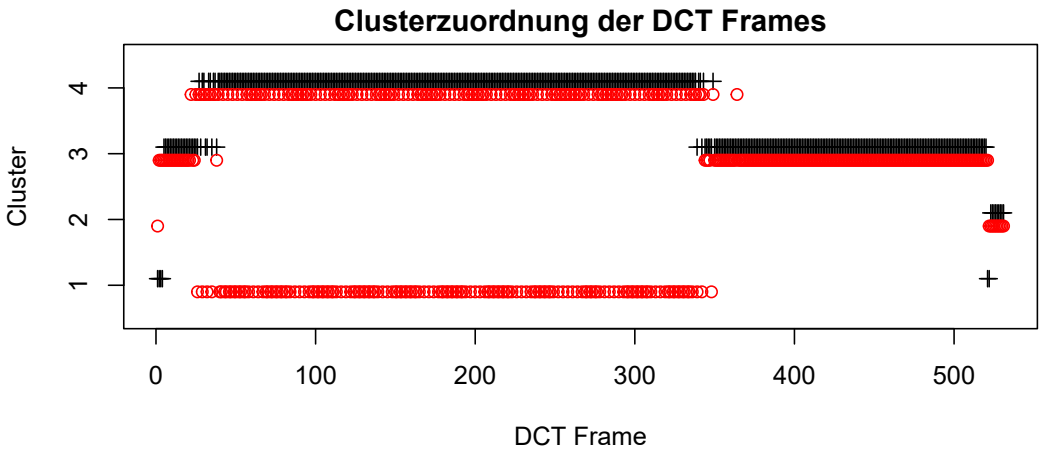

Abbildung 6.5: 2 unterschiedliche k-Means Clusterungen eines anderen Kontrafagotttons

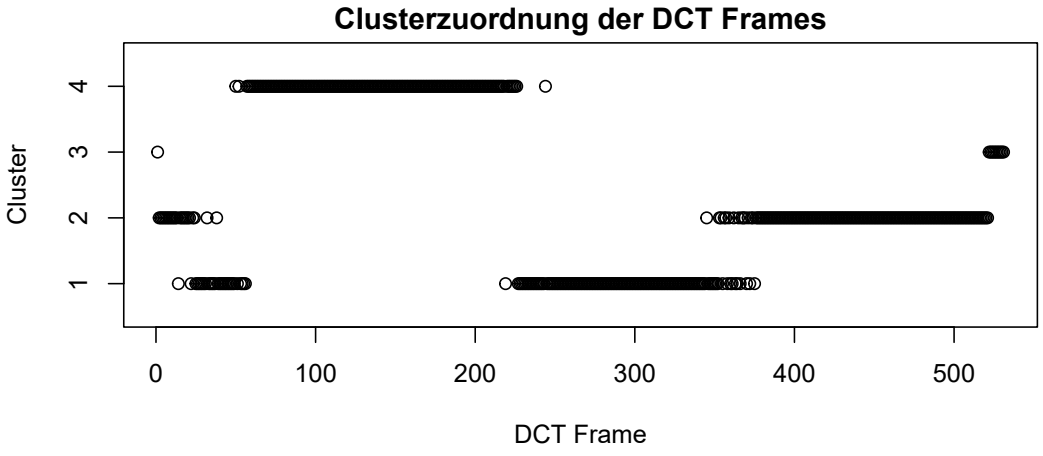

Abbildung 6.6: Stabilere Clusterung des Kontrafagotttons aus Abbildung 6.5 mit Neural Gas

7 Bestimmung der Clusteranzahl

7.1 Problemstellung

Eine grundlegende Frage bei der Anwendung von Clusterverfahren ist die Festlegung der Anzahl der zu bestimmenden Cluster. Die Clusteranzahl ist ein vor Anwendung des Verfahrens festzulegender Parameter (k-Means artige Verfahren) oder muss nach Abschluss des Verfahrens anhand einer sinnvollen Ähnlichkeit der Objekte innerhalb der Cluster und Unähnlichkeit der Cluster untereinander bestimmt werden. Die Anzahl der ermittelten Cluster hat einen großen Einfluss auf die Interpretation der Ergebnisse und sollte daher anhand objektiver Kriterien erfolgen. Die Frage nach objektiven Kriterien zur Bestimmung der Anzahl der Cluster hat schon viele Wissenschaftler beschäftigt und zu einer Vielzahl an vorgeschlagenen Verfahren und Indizes geführt. Die Arbeiten von Halkidi, Batistakis und Vazirgiannis (2001) und Weingessel, Dimitriadou und Dolnicar (1999) geben eine Übersicht über eine Vielzahl der seit den 1960er Jahren vorgestellten Methoden. Die Methoden lassen sich in zwei Hauptgruppen unterteilen. Zum einen sind es auf den Vergleich von Clusterergebnissen auf Bootstrap Stichproben der Daten oder Monte Carlo Tests basierende Methoden und zum anderen Methoden, die einen relativen Vergleich von statistischen Kennzahlen nutzen. Diese Kennzahlen werden auf Basis der ursprünglichen Daten und der aus der Anwendung des Clusterverfahrens resultierenden Partitionierung der Daten berechnet.

7.2 Schwächen relativer Clustervaliditätsindizes

Der Versuch mit Hilfe der auf relativen Kriterien basierenden Indizes eine sinnvolle Clusteranzahl für die McGill Aufnahmen von einzelnen Instrumenten zu bestimmen, war wenig erfolgreich. Die meisten Indizes nennen die Ränder des zu untersuchenden Bereichs an möglichen Clusteranzahlen als Optimum (siehe Abbildung 7.1). Ändert man das zu untersuchende Intervall, so ändert sich auch das Ergebnis.

Nur der Index von *Ratkowski und Lance* sowie der *Simple Structure Index* zeigen nicht dieses Verhalten. Bei komplexeren Tonaufnahmen mit mehr Clustern kann dieser Eindruck aber nicht bestätigt werden.

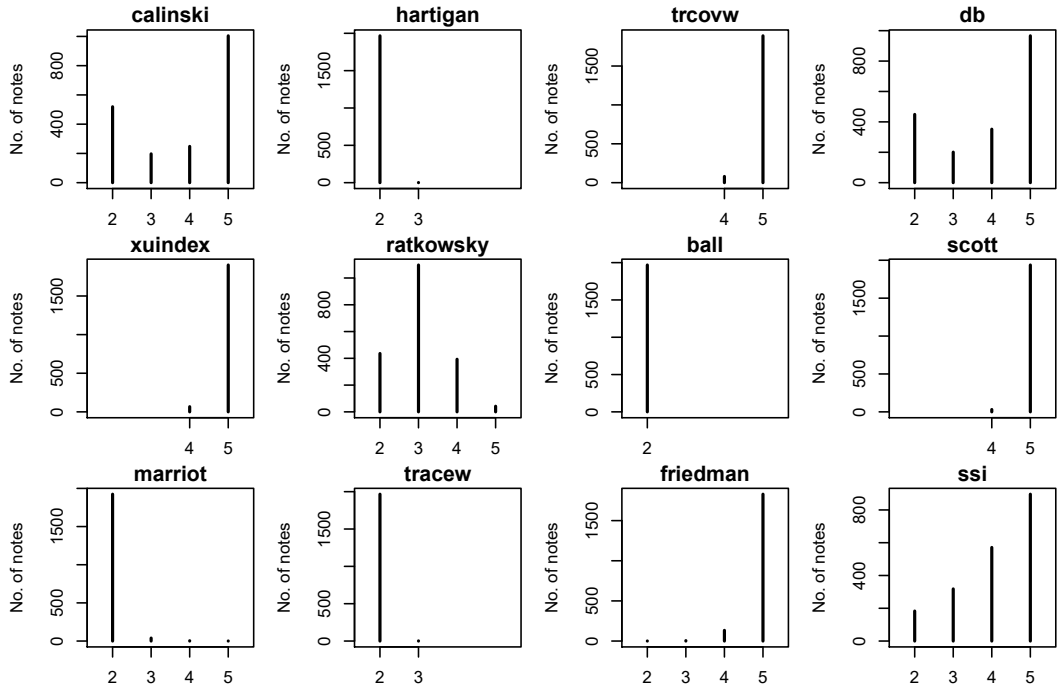

Abbildung 7.1: Übersicht der bestimmten Clusteranzahlen

7.3 Rand-Index zur Bestimmung der Clusteranzahl

Dieses Unterkapitel basiert auf den Beschreibungen zur Clusteranzahlbestimmung in Ligges und Krey (2011).

Als gut geeignet für die Bestimmung der Clusteranzahl hat sich der Vergleich von Clusterergebnissen auf Bootstrapstichproben der Daten mit Hilfe des *adjustierten Rand-Index* erwiesen. Dies ist in Dolnicar und Leisch (2010) beschrieben und eine Weiterentwicklung des Vergleichs mittels externer Kriterien wie es in Halkidi, Batistakis und Vazirgiannis (2001) beschrieben ist. Es werden b (hier $b = 20$) Paare von Bootstrap Stichproben der Daten gezogen und das Clusterverfahren auf jede der Stichproben angewendet. Die Ähnlichkeit der Ergebnisses eines Stichprobenpaars wird mit dem adjustierten Rand-Index bewertet. Die Clusterzahl mit dem höchsten mittleren adjustierten Rand-Index wird gewählt.

Der Rand-Index misst die Übereinstimmung zweier Partitionierungen der Daten (Rand, 1971). Der adjustierte Rand-Index ist dafür korrigiert, wie wahrscheinlich die beobachtete als Kontingenztafel notierte aus einem Clusterverfahren erhaltene Partitionierung der Daten für eine Hypergeometrische Verteilung mit den gegebenen Zeilen- und Spaltensummen ist (Hubert und Arabie, 1985). Dies ermöglicht ei-

ne einfache Interpretation des Index. Ein Wert nahe Null ist ein starker Indikator für ein durch Zufall entstandenes Ergebnis. Die obere Schranke des adjustierten Rand-Index ist 1 und wird bei perfekter Übereinstimmung angenommen. Für zwei Clusterungen $A = \{A^1, A^2, \ldots, A^{K_1}\}$ und $B = \{B^1, B^2, \ldots, B^{K_2}\}$ bezeichne n_{ij} die Anzahl der Objekte, die A^i und B^j gemeinsam haben. Dann berechnet sich der adjustierte Rand-Index durch

$$R = \frac{\sum_{i=1}^{K_1}\sum_{j=1}^{K_2}\binom{n_{ij}}{2} - \frac{1}{\binom{n}{2}}\left(\sum_{i=1}^{K_1}\binom{n_{i\cdot}}{2}\sum_{j=1}^{K_2}\binom{n_{\cdot j}}{2}\right)}{\frac{1}{2}\left(\sum_{i=1}^{K_1}\binom{n_{i\cdot}}{2} + \sum_{j=1}^{K_2}\binom{n_{\cdot j}}{2}\right) - \frac{1}{\binom{n}{2}}\left(\sum_{i=1}^{K_1}\binom{n_{i\cdot}}{2}\sum_{j=1}^{K_2}\binom{n_{\cdot j}}{2}\right)},$$

wobei $n = \sum_{i=1}^{K_1}\sum_{j=1}^{K_2} n_{ij}$, $n_{i\cdot} = \sum_{j=1}^{K_2} n_{ij}$ und $n_{\cdot j} = \sum_{i=1}^{K_1} n_{ij}$. Da hier Clusterungen von Bootstrap Stichproben der Daten mit derselben Anzahl von Clustern verglichen werden, gilt $K_1 = K_2$.

Die Idee hinter dieser Methode ist, dass bei Wahl der richtigen Anzahl von Clustern das Ergebnis des Clusterverfahrens auf unterschiedlichen Bootstrapstichproben der Daten sehr ähnlich sein sollte. Diese Ähnlichkeit wird mit dem adjustierten Rand-Index gemessen. Wählt man eine zu hohe Clusterzahl kommt es zu künstlichen Trennungen von Clustern, die durch die Variation der Bootstrapstichproben immer etwas anders aussehen und so durch ihre geringe Stabilität den Rand-Index reduzieren. Auf ähnliche Weise wird eine zu niedrige Anzahl von Clustern zu mit den Bootstrapstichproben sich ändernden Zusammenlegungen von Clustern führen, die ebenfalls den Rand-Index reduzieren. Daher ist die Clusterzahl, die über alle Bootstrapstichproben den höchsten mittleren adjustierten Rand-Index aufweist, ein guter Indikator für die richtige Wahl der Anzahl von Clustern.

7.3.1 Varianz der Rand-Indizes als zusätzliches Entscheidungskriterium

Leider führt dieses Verfahren nicht immer zu sinnvollen Ergebnissen. Bei der Anwendung des in Kapitel 9.3 beschriebenen Order Constrained k-Means Clusterings zur Strukturierung von längeren Musikstücken ist zu beobachten, dass es oft zwei oder mehrere Clusterzahlen mit sehr ähnlichen mittleren adjustieren Rand-Index gibt, darunter auch Clusterzahlen, die unrealistisch niedrig sind. Die oft zu niedrig geschätzte Clusteranzahl ist auch eine von Steinley und Brusco (2011) geäußerte Kritik an dieser Vorgehensweise. Steinley und Brusco (2011) sprechen sich daher gegen die alleinige Verwendung des mittleren adjustierten Rand-Index für die Wahl der Clusteranzahl aus. Es ist daher empfehlenswert ein weiteres Kriterium zur Bestimmung der Clusteranzahl heranzuziehen. Eine geringe Streuung des adjustierten Rand-Index ist ein weiterer Indikator für ein über die unterschiedlichen Boot-

strapstichproben stabiles Ergebnis eines Clusterverfahrens. Für die gemeinsame Betrachtung dieser beiden Optimalitätskriterien minimale Streuung und maximaler mittlerer Wert bietet sich als klassische statistische Kennzahl die Betrachtung des Verhältnisses von Standardabweichung und arithmetischem Mittel, der Variationskoeffizient, an.

7.3.2 Robuster Variationskoeffizient

Bei Tonaufnahmen mit sehr kurzen Klangabschnitten, d.h. mit einer Dauer von nur wenigen Zeitfenstern, kann es zu Bootstrapstichproben kommen, die bestimmte Cluster gar nicht enthalten. Dies führt zu stark abweichenden Clusterergebnissen mit daraus resultierenden sehr extremen Rand-Indizes. Die Folge ist eine Verzerrung von arithmetischem Mittel und Standardabweichung. Daher ist in solchen Fällen eine robuste Variante des Variationskoeffizients sinnvoller. Der *Quartilsdispersionskoeffizient* als Quotient aus Interquartilsabstand und Median ist daher eine zu bevorzugende robuste Alternative des Variationskoeffizients.

Man erhält also die optimale Clusteranzahl dadurch, dass man für jeweils Paare von Bootstrap-Stichproben für alle $K = 2, \ldots, K_m$ die Clusterungen bestimmt und mittels des Rand-Index die Übereinstimmung bestimmt. Für jedes $K = k$ wird dann die p-Quantile q_p^k der Rand-Indizes bestimmt. Wobei K_m die maximal zu untersuchende Clusteranzahl ist und geeignet gewählt werden muss. Die optimale Clusteranzahl ist dann

$$K_o = \arg\min_{K=2,\ldots,K_m} \frac{q_{0.75}^K - q_{0.25}^K}{q_{0.5}^K}.$$

7.4 Stichprobenverfahren für Clusterverfahren mit Nebenbedingungen

Während man bei klassischen Clusterverfahren die Bootstrap-Stichproben für die Clusterzahlbestimmung durch eine einfache Zufallsstichprobe (Ziehen mit Zurücklegen) erhält, ist dies bei den in dieser Arbeit betrachteten Clusterverfahren mit Nebenbedingungen nicht so einfach möglich. Mit einer einfachen Zufallsstichprobe wird man in den meisten Fällen Bootstrapsamples erhalten, die die geforderten Nebenbedingungen verletzten. Man muss also die Stichprobenziehung anpassen.

7.4.1 Stichproben für Clusterverfahren mit Ordnungsbedingung

Die Anpassung des Stichprobenverfahrens für eine Ordnungsrestriktion ist sehr einfach möglich. Die Ziehung der Bootstrap-Stichprobe erfolgt wie im Fall ohne Nebenbedingungen. Anschließend sortiert man die Elemente gemäß der Ordnungsbedingung. Sei π die Permutation, die die Merkmalsvektoren Z gemäß der Ordnungsbedingung umsortiert und Z^{B_i} die Merkmalvektoren der Bootstrap-Stichprobe B_i. Mit $\pi(Z^{B_i})$ erhält man eine gemäß Ordnungsbedingung sortierte Bootstrapstichprobe der Merkmalsvektoren, welche zeilenweise in eine Matrix geschrieben werden. Man erhält so eine Stichprobe in der alle Datenpunkte, die gemäß der Ordnungsbedingung innerhalb eines Clusters sein können, in aufeinander folgenden Zeilen der Datenmatrix stehen. Das eigentliche Clusterverfahren arbeitet dann mit der natürlichen Ordnung der Zeilen der Bootstrap-Stichprobe. Enthält die Bootstrap-Stichprobe durch das Ziehen mit Zurücklegen einen Datenpunkt mehrfach, so stehen diese Punkte untereinander in der Datenmatrix und sind ab der ersten Iteration im gleichen Cluster. Lässt man sehr kleine Cluster mit nur zwei oder drei Datenpunkten zu, kann dies zu Clustern mit einer Varianz von Null führen. Falls weitergehende Datenverarbeitungsschritte eine Varianz ungleich Null voraussetzen, muss die minimale Clustergröße entsprechend angehoben werden.

7.4.2 Stichproben für Spectral Clustering (Graphbedingung)

Beim Spectral Clustering, also dem Clustern auf einer durch einen Graphen beschriebenen Nachbarschaftsstruktur, kann eine einfache Zufallsstichprobe der Datenpunkte nicht für die Erzeugung von neuen Datensätzen für das Bootstrap-Verfahren verwendet werden. Fehlt ein bestimmter Datenpunkt (Knoten) in einer Bootstrap-Stichprobe, so wird die Nachbarschaftsstruktur des Graphen an dessen Stelle stark verändert und ist nicht mehr repräsentativ für den ursprünglichen Graphen. Anstatt auf Basis der Datenpunkte, also den Knoten des Graphen, zu arbeiten, bietet sich die Nachbarschaftsstruktur, also die Kanten des Graphen, an, um Variation in den Datensatz für die Untersuchung der Stabilität des Clusterverfahrens zu bringen. Durch Weglassen oder Hinzufügen einzelner oder mehrerer Kanten an zufälligen Orten im Graphen bringt man kleine Veränderungen in die Nachbarschaftsstruktur ein. So erhält man Variationen des Datensatzes, deren unterschiedlichen Clusterergebnisse dann wieder mit dem Rand-Index verglichen werden können. Die genaue Wahl des Verfahrens muss sich an der konkreten Anwendung orientieren um geeignete Ergebnisse zu ermöglichen. Nimmt man in einem nur dünn vermaschten Graphen

mehrere Kanten weg, können einzelne Knoten komplett vom Graphen abgeschnitten werden. Das Hinzufügen neuer Kanten kann mitunter unrealistisch sein, da es eine Nachbarschaft zwischen Knoten des Graphen herstellt, die in der Realität evtl. nicht realisierbar ist. Bei der in dieser Arbeit betrachteten Anwendung im Bereich der Höchstspannungsverteilnetze (siehe Kapitel 10) wurde das Entfernen einzelner Kanten verwendet. Aus Gründen der Redundanz sind alle Knoten eines Energienetzes über min. zwei Leitungen (Kanten) mit ihren Nachbarknoten verbunden. Das Entfernen einer Kante (Trennung einer Leitung) isoliert also keinen Knoten vom Rest des Netzes. Gleichzeitig ist die Trennung einer einzelnen Leitung ein realistisches Szenario für eine durch einen Störfall oder Wartung entstehende Änderung der Nachbarschaftsstruktur des Graphen.

Bezeichnet man den vollständigen Graphen mit G, die Knoten (vertices) mit V bzw. den i-ten Knoten mit V_i sowie die Kanten (edges) mit E bzw. E_i für die i-te Kante, so kann mit G^{-E_i} den Graphen bezeichnen, bei dem die i-te Kante entfernt wurde. Sei N_E die Anzahl der Kanten des vollständigen Graphen, so kann man N_E „leave-one-out“ Graphen G^{-E_i}, $i = 1, \ldots, N_E$ erstellen. Auf jeden dieser Teilgraphen G^{-E_i} wendet man das gewählte Clusterverfahren an. So erhält man dann die Stichprobe für die Stabilitätsuntersuchung mit Hilfe des Rand-Index.

Für sehr große Graphen erhält meine eine hohe Anzahl von Teilgraphen G^{-E_i}. Falls der Rechenaufwand zu groß ist, um eine Clusterung aller Teilgraphen durchzuführen, so gibt es mehrere Möglichkeiten das vorgestellte Verfahren an diese Bedingungen anzupassen. Zwei Beispiele:

1. Wenn es für die Anwendung plausibel ist, kann mehr als eine Kante entfernt werden, so lange man sicherstellt, dass der Graph dadurch nicht in disjunkte Teile zerfällt.

2. Eine einfachere Möglichkeit ist ein Sampling (Bootstrap-Stichprobe) aus der Menge der Teilgraphen G^{-E_i}. Das heißt man zieht eine festgelegt Anzahl von Graphen aus G^{-E_i}, der Menge der Graphen mit einer entfernten Kante, Der Rechenaufwand für die Stabilitätsuntersuchung kann so über die Größe der Stichprobe kontrolliert werden. Dieses Vorgehen ist eine direkte Adaption der ursprünglichen Methodik der Stabilitätsuntersuchung der Clusterungen von Bootstrap Stichproben der ursprünglichen Daten mit Hilfe des adjustierten Rand-Index auf das Clustern eines Graphen.

8 Klassifikation von Instrumentenklängen auf Basis geclusterter Merkmalsvektoren

Dieses Kapitel basiert auf den Ergebnissen aus Ligges und Krey (2011).

Bei der Analyse von Musikstücken ist eine oft interessante Frage, welche Instrumente beim Einspielen verwendet wurden. Die Klassifikation der Klänge in einer Tonaufnahme mit Hilfe von überwachten maschinellen Lernverfahren ist eine Möglichkeit dies rechnergestützt durchzuführen.

8.1 Clustern von Instrumentenklängen

8.1.1 Clustern als Zwischenschritt zur Klassifikation

Es gibt verschiedene Möglichkeiten and die rechnergestützte Klassifikation von Klängen heranzugehen.

Der einfachste Ansatz ist die Auswahl eines einzelnen (möglichst repräsentativen) Abschnitts aus dem Tonsignal um auf diesem Abschnitt Klangmerkmale, wie z.B. MFCCs zu berechnen. Diese Herangehensweise hält den Datensatz einfach und relativ klein, da es für jeden Klang/Tonaufnahme nur einen einzelnen Merkmalsvektor gibt, der diesen repräsentiert. Wenn sich Klänge aber über die Zeit verändern wird die Auswahl eine geeigneten Zeitfensters in der Aufnahme sehr schwer. Der intuitive Ansatz einen Zeitpunkt zu wählen, wo sich der Klang stabilisiert hat, macht es unmöglich die klanglichen Unterschiede z.B. zwischen einer gezupften, geschlagenen oder gestrichenen Seite zu erkennen, die hauptsächlich in der Obertonstruktur während der Einschwingphase erkennbar sind.

Eine andere Herangehensweise ist die Verwendung aller mit Hilfe eines gleitenden Fensters über die Tonaufnahme extrahierten Merkmalsvektoren als Datenpunkte für die Klassifikation. Dies stellt das andere Extrem zum zuvor genannten Vorgehen dar und enthält alle notwendigen Daten, um auch Unterschiede in transienten Teilen eines Klangs erkennen zu können. Der Ansatz hat aber den Nachteil den Datensatz sehr stark zu vergrößern, da je nach Länge der Aufnahme und Breite des Zeifensters hunderte oder gar tausende Merkmalsvektoren aus einem Klang generiert werden. Merkmalsvektoren, die aus Zeitfenstern extrahiert wurden, die zeitlich

sehr eng zueinander liegen, weisen eine sehr hohe Autokorrelation auf und enthalten daher redundante Informationen. Gleichzeitig kann man die einzelnen Merkmalsvektoren nicht mehr als unabhängige Datenpunkte betrachten und muss daher bei Modelloptimierung und -validierung stratifizierte Stichprobenverfahren anwenden, die aufwändiger sind.

Bei der Zusammenfassung aller Merkmalsvektoren eines Klangs zu einem Mittelwert können die transienten Anteile zwar einen Einfluss auf den Mittelwert ausüben, dieser wird aber durch die Kürze der entsprechenden Phasen gering sein, weshalb nur eine minimale Verbesserung im Vergleich zur Auswahl eines einzelnen Repräsentanten vorliegt.

Durch Clusterverfahren ist es dagegen möglich eine kleine Anzahl repräsentativer Clusterzentren für verschiedene Phasen eines Klangs zu identifzieren und zu einem hochdimensionalen Merkmalsvektor zu aggregieren.

8.1.2 Vorgehen

Nach der in Kapitel 3 beschrieben Vorverarbeitung der Tonaufnahmen und anschließenden Merkmalsextraktion liegen für jedes Zeitfenster $i = 1, \ldots, N$ der gefensterten Fouriertransformation ein p-dimensionaler Merkmalsvektor Z_i vor. Diese Vektoren werden zeilenweise in die Nxp Matrix $\mathbf{Z}$ geschrieben. Es liegen für jede Tonaufnahme eines Klangs also $p \cdot N$ Merkmale vor, wobei N von der Länge der Aufnahme abhängt und daher für jeden Klang/jede Tonaufnahme unterschiedlich sein kann.

Auf die Datenmatrix $\mathbf{Z}$ wird dann k-Means Clustering mit 25 zufälligen Starts angewendet, um $k = 4$ Clusterzentren zu bestimmen. Die Wahl auf $k = 4$ ist mit der Überlegung getroffen worden, dass für jeden Klang die Einschwing- (Attack), Halte- (Sustain) und Ausschwingphase (Decay) erfasst werden soll, sowie ein weiterer Cluster für die am Ende jeder Tonaufnahme vorhandende Stille, die nur Rauschen enhält. Die Klangmerkmale für diesen Cluster werden verworfen, da sie keine sinnvolle Information enthalten. Die 3 Klangphasen entsprechen der üblichen Modellierung von Klängen, wie man sie z.B. auch in Synthesizern vornimmt. Die Anzahl der Cluster konnte in diesem Fall durch Überlegungen aus der Anwendung getroffen werden. Für Situationen in denen dies nicht möglich ist bzw. es mehrere Möglichkeiten gibt, werden in Kapitel 7 Methoden vorgestellt und verglichen, die objektive Entscheidungskriterien für eine sinnvolle Clusterzahl bieten.

Die durch die Clusterung erhaltenen und nach Entfernen des Rauschcluster verbliebenen $k - 1 = 3$ Clusterzentren können dann als $(k-1)xp$ Matrix $\mathbf{Z}^c$ geschrieben wer-

Tabelle 8.1: Vergleich Anzahl Datenpunkte und Merkmale

Verarbeitungsschritt	Datensatzgröße	Features
Rohdaten	1980 Aufnahmen	Mittlere Länge 180000 Samples
Merkmalsextraktion	800000 Fenster	p dimensionale Klangmerkmale
Clusterung	1980 Aufnahmen	4 Cluster mit p dim. Clusterzentren
Clusterung nach Rauschentfernung	1980 Aufnahmen	3 Cluster mit p dim. Clusterzentren

den. Damit liegt eine standardisierte, von der Aufnahmenlänge unabhängige Form der Klangmerkmale für jeden Klang der McGill Instrumentendatenbank vor, die trotzdem eine Unterscheidung von Klangfarben aufgrund von sich zeitlich verändernden Klangmerkmalen ermöglicht. Gleichzeitig wurde die Datenmenge stark reduziert. Ein Vergleich für die verschiedenen Formen der Klangmerkmale befindet sich in Tabelle 8.1. Die Reihenfolge der Clusterzentren in der Matrix erfolgt in der Reihenfolge des ersten Auftretens des zugehörigen Clusters in der Tonaufnahme.

8.2 Klassifikation

Für die Klassifikation der McGill Instrumentenklänge sind verschiedene maschinelle Lernverfahren verwendet worden. Der Fokus liegt auf den *Support Vektor Maschinen (SVM)* mit verschieden Kernelfunktionen im Vergleich zu *k-nächsten Nachbarn (kNN)*, zum *Random Forest (RandFor)* und zur *Linearen Diskriminanzanalyse (LDA)*. Eine detaillierte Beschreibung der Verfahren findet sich in Hastie, Tibshirani und Friedman, 2001.

8.2.1 Support Vektor Maschinen (SVM)

Die SVM ist ein binäres Klassifikationsverfahren, das seine optimale Klassifikationsregel durch Lösen des folgenden Optimierungsproblems findet (Hastie, Tibshirani und Friedman, 2001; Hsu, Chang und Lin, 2008):

$$\min_{w,b,\xi}\left(\frac{1}{2}w'w + C\sum_{i=1}^{n}\xi_i\right) \quad \text{mit } \xi_i \geq 0,\ y_i(w'\varphi(x_i) + b) \geq 1 - \xi_i$$

Hierbei ist $\varphi(x_i)'\varphi(x_j) \equiv K(x_i, x_j)$ eine Kernelfunktion und C der Regularisierungsparameter der SVM. Das Resultat ist eine lineare Entscheidungsgrenze.

Durch Verwendung von Kernelfunktionen können auch nichtlineare Klassifikationsregeln erstellt werden. Hier wurden folgende 4 Standardkernelfunktionen auf ihre Eignung für das vorliegende Klassifikationsproblem untersucht:

Linear	$K(x_i, x_j) = x_i' x_j$
Polynomial	$K(x_i, x_j) = (\sigma x_i' x_j + r)^d$
Gauß Radialbasisfunktion	$K(x_i, x_j) = \exp(-\sigma \|x_i - x_j\|^2)$
ANOVA Radialbasisfunktion	$K(x_i, x_j) = (\sum_{k=1}^{p} \exp(-\sigma(x_{ik} - x_{jk})^2))^d$

wobei für die Parameter gilt: $r \in \mathbb{R}$, $\sigma > 0$ und $d \in \mathbb{N}$.

Die SVM ist ein binäres Klassifikationsverfahren und muss daher für Aufgabenstellungen mit mehr als $c = 2$ Klassen modifiziert werden. Es gibt verschiedene Möglichkeiten die Entscheidungsregeln der SVM an solche Probleme anzupassen. Die beiden häufigsten Ansätze sind die Kombinationen mehrerer SVM Entscheidungsregeln.

Der erste Ansatz trainiert c SVMs, wobei für jede der c Klassen eine SVM Entscheidungsregel für diese Klasse gegen alle anderen erstellt wird.

Bei der zweiten Möglichkeit werden immer paarweise Vergleiche gebildet. Es werden also für alle $c(c-1)/2$ Kombinationen eine SVM Entscheidungsregel erstellt und die finale Klassifikation eines Objekts ist das Mehrheitsvotum dieser $c(c-1)/2$ Klassifikatoren. Dieses Verfahren ist sehr viel rechenaufwändiger als das zuvor genannte, hat sich aber als leistungsfähiger erwiesen.

8.2.2 Validierung der Ergebnisse

Zur Parameteroptimierung der Klassifikatoren und Bestimmung der Fehlklassifikationsrate der verschiedenen Verfahren ist eine geschachtelte Kreuzvalidierung verwendet worden.

In der inneren 5-fachen Kreuzvalidierung sind die optimalen Parameter für die Verfahren bestimmt worden. Die Kandidaten für den Regularisierungsparameter C der SVM, Grad d des polynomiellen Kernels, die Anzahl der Bäume U und die Anzahl der Kandidatvariablen V des Random Forest waren:

$$
\begin{aligned}
C: &\quad 0.25, 0.5, 0.6, \ldots, 1.4, 1.5, 2 \\
d: &\quad 2, 3, 4, 5 \\
U: &\quad 500, 1000, 1500 \\
V: &\quad 3, 4, 5, 6, 7, 8
\end{aligned}
$$

Für die Bestimmung des Parameter σ der Radialbasisfunktion wurde das Standardverfahren des `kernlab` Pakets verwendet. Hierzu werden zwei 50% Stichproben aus Z^c gezogen und der Median der quadrierten euklidischen Distanz bestimmt. Der Kehrwert dieses Werts ist der gesuchte Parameter σ.

Die Fehlklassifikationsrate der Verfahren ist dann in der äußeren 5-fachen Kreuzvalidierung erfolgt. Durch dieses Vorgehen erreicht man eine realistische Schätzung für die echte Fehlklassifikationsrate eines überwachten maschinellen Lernverfahrens und kann sich vor einer Überanpassung an die Daten schützen.

8.3 Software

Für die Klassifikation ist die SVM Implementierung des `R` Pakets `kernlab` (Karatzoglou u. a., 2004), der Random Forests aus `RandomForest` (Liaw und Wiener, 2002), die Lineare Diskriminanzanalyse aus `MASS` (Venables und Ripley, 2002) sowie k-nächtsen Nachbarn aus `FNN` (Beygelzimer u. a., 2013) verwendet worden. Die Parameteroptimierung sowie Klassifikatorvalidierung ist mit `mlr` (Bischl u. a., 2016), einer Machine Learning Toolbox für `R`, durchgeführt worden.

8.4 Ergebnisse

Die Anwendung der beschriebenen Vorgehensweise auf die McGill Instrumentendatenbank liefert sehr zufriedenstellende Ergebnisse. Die Fehlklassifikationsraten in den nachfolgenden Tabellen basieren auf der Validierung in der äußeren 5-fach Kreuzvalidierung. Als optimaler Parameter ist für jeden Parameter der Median aus diesen 5 Läufen angegeben.

Zuerst werden die Ergebnisse des einfacheren Klassifikationsproblems, der Zuordnung der Klänge zu einer von 25 Instrumentenfamilien präsentiert. Diese Ergebnisse sind für die PLP-Klangmerkmale in Tabelle 8.2 und für die MFCC-Klangmerkmale in

Tabelle 8.3 zu finden. Die Leistung der Lineare Diskriminanzanalyse, des k-nächsten Nachbarn Klassifikators sowie der Random Forest dienen als Referenz. Die ersten beiden Verfahren sind mathematisch und algorithmisch sehr einfach und daher die am weitesten bekannten Klassifikationsverfahren. Der Random Forest ist ein Klassifiaktionsverfahren, das bei vielen Fragestellung schon mit den Standardparametern sehr gute Leistung liefert, aber aufgrund des Designs als Ensembleklassifikator mit einer großen Anzahl an Entscheidungsbäumen einen hohen Speicherbedarf aufweist und daher nicht überall einsetzbar ist. Der polynomielle Kernel hat sich für beide Klangmerkmale (MFCCs und PLPs) als am leistungsfähigsten erwiesen. Mit einem Polynomgrad von $d = 3$ erreichen die PLPs eine Fehlklassifikationsrate von 26%. Diese Leistung ist vergleichbar mit den für dieses Klassifikationsproblem in Roever (2003) unter Verwendeung einer regularisierten Diskriminanzanalyse erreichten 26% Fehlklassifikationsrate. Mit dem k-nächste Nachbarn Klassifikator sind mit 32% Fehlklassifikationsrate im Vergleich zu 31% ebenfalls vergleichbare Ergebnisse erreicht worden. Der häufig als Standardkernel für die SVM empfohlene Gauß-Radialbasisfunktion-Kernel (Hsu, Chang und Lin, 2008) hat hier mit 47% Fehlklassifiaktionsrate eine deutlich höhere Fehlerrate als der polynomielle Kernel und ist vergleichbar mit der linearen Diskriminanzanalyse. Das Klassifikationsverfahren mit der geringsten Fehlklassifikationsrate ist in diesem Fall der Random Forest mit 22%, wobei sich aber schon ein Konfidenzintervall von einer Standardabweichung mit dem der SVM mit polynomiellen Kernel überlappen würde.

Durch Verwendung der MFCC Klangmerkmale kann die Leistung aller Klassifikatoren bei dieser Problemstellung deutlich verbessert werden (siehe Tabelle 8.3). Die LDA verbessert sich auf 25% und k-nächste Nachbarn auf 17%. Die SVM mit den verschiedenen Kernelfunktionen und der Random Forest sind mit einer Fehlklassifikationsrate von 10 – 11% auf einem vergleichbaren sehr hohen Niveau. In diesem Fall

Tabelle 8.2: Klassifikationsleistung der PLPs beim 25 Instrumentenfamilien Problem

Klassifikator	Parameter	Fehler in %	Std.Abw. in %
SVM-Poly	$C = 1.4,\ d = 3$	26	3.7
SVM-RBF	$C = 1.4,\ \sigma = 0.133$	47	3.3
SVM-ARBF	$C = 1.4,\ \sigma = 0.142$	37	3.7
SVM-Lin	$C = 1.5$	38	2.9
RandFor	$U = 500,\ V = 7$	22	2.5
LDA		46	1.8
kNN	$k = 4$	32	0.9

Tabelle 8.3: Klassifikationsleistung der MFCCs beim 25 Instrumentenfamilien Problem

Klassifikator	Parameter	Fehler in %	Std.Abw. in %
SVM-Poly	$C = 0.6,\ d = 2$	10	2.7
SVM-RBF	$C = 1.5,\ \sigma = 0.011$	11	2.7
SVM-ARBF	$C = 0.8,\ \sigma = 0.011$	10	2.2
SVM-Lin	$C = 1$	11	2.6
RandFor	$U = 1500,\ V = 7$	10	3.2
LDA		25	1.5
kNN	$k = 1$	17	2.0
RandFor ungecl.	$U = 500,\ V = 5,\ sampsize = 80000$	18	1.7
kNN ungecl.	$k = 1$	27	2.3

hat sich ein Polynomgrad von $d = 2$ für die SVM mit polynomiellen Kernel als optimal herausgestellt.

Zum Vergleich sind die der Random Forest (auf einem Subsampling der Daten, um den Arbeitsspeicherbedarf zu kontrollieren) sowie die k-nächsten Nachbarn auch noch einmal auf den ungeclusterten MFCCs angewendet worden. Die Fehlklassifikationsrate war in diesem Fall 8% bzw. 10% höher als bei der Verwendung der Clusterzentren und damit deutlich schlechter (bei gleichzeitig sehr viel höherem Rechenzeit- und Arbeitsspeicherbedarf).

Die Klassifikationsleistung bei dem sehr viel komplexeren Problem alle 59 Klangfarben der Instrumente zu unterscheiden sind in Tabelle 8.4 für die PLPs und Tabelle 8.6 für die MFCCs zusammengefasst.

Das Leistungsverhältnisse zwischen den Klassifikatoren sind bei Verwendung der PLP Klangmerkmale sehr ähnlich zum 25 Klassen Problem, wobei natürlich die Fehlerrate deutlich höher ist. Der Random Forest weist wieder mit jetzt 34% Fehlklasifikationsrate die höchste Leistung auf, gefolgt von der SVM mit polynomiellen Kernel mit Grad $d = 3$ und 43% Fehlerrate sowie dem k-nächste Nachbarn Klassifikator, der 48% Fehlklassifiaktionsrate erreicht hat. Die LDA (60% Fehlerrate) und die SVM mit Gauß-RBF Kernel (71% Fehlerrate) bilden wieder das Schlusslicht.

Die Klassifikationsleistung unter Verwendung der PLPs kann durch Standardisierung der Autoregressionskoeffizienten a_i jedes Frames vor der Clusterung gesteigert werden. Hierzu werden die a_i jedes Frames mit ihrem arithm. Mittel zentriert und durch ihre Standardabweichung standardisiert. Diese Ergebnisse sind in Tabelle 8.5

zu finden. Insbesondere die SVM (mit allen Kernelfunktionen) sowie der k-nächste Nachbarn Klassifikator profitieren sehr deutlich von dieser Modifikation und können ihr Ergebnis um 10% (kNN und SVM mit polynomiellen Kernel) bzw. 27% (SVM mit Gauß-RBF Kernel) steigern. Die LDA (5% Verbesserung) und der Random Forest (2%) profitieren deutlich weniger von dieser Veränderung bei der Merkmalsextraktion bzw. Clusterung.

Durch Veränderung des *Lag* p im autoregressiven Modell der PLPs ist keine Verbesserung der Klassifikationsleistung in dem hier betrachteten Anwendungsfall zu beobachten.

Tabelle 8.4: Klassifikationsleistung der PLPs beim 59 Klangfarben Problem

Klassifikator	Parameter	Fehler in %	Std.Abw. in %
SVM-Poly	$C = 1.6,\ d = 3$	43	2.6
SVM-RBF	$C = 1.5,\ \sigma = 0.122$	71	5.0
SVM-ARBF	$C = 1.4,\ \sigma = 0.117$	55	3.0
SVM-Lin	$C = 1.5$	51	2.8
RandFor	$U = 1500,\ V = 6$	34	3.9
LDA		60	2.5
kNN	$k = 6$	48	2.4

Bei den MFCCs kommt es beim 59 Klassen Problem zu einer deutlicheren Verschiebung der Leistungsverhältnisse zwischen den verschiedenen Klassifikationsverfahren. Während beim 25 Klassen Fall die SVM mit den verschiedenen Kernelfunktionen und der Random Forest nahezu gleichauf liegen, übernimmt die SVM mit polynomiellen und ANOVA-RBF Kernel mit 19% Fehlklassifiaktionsrate knapp gefolgt von der SVM mit linearem Kernel (also ohne Kernelfunktion) und 20% Fehlerrate die Führung. Der Random Forest liegt mit einer Fehlerrate von 28% deutlich dahinter, aber weiter vor der LDA und dem k-nächste Nachbarn Klassifikator. Diese beiden

Tabelle 8.5: Klassifikationsleistung der std. PLPs beim 59 Klangfarben Problem

Klassifikator	Parameter	Fehler in %	Std.Abw. in %
SVM-Poly	$C = 1.4,\ d = 3$	33	3.1
SVM-RBF	$C = 1.4,\ \sigma = 0.023$	44	3.2
RandFor	$U = 1500,\ V = 3$	32	3.2
LDA		55	1.8
kNN	$k = 4$	38	2.9

Klassifikationsverfahren haben im Vergleich zum 25 Klassenproblem die Reihenfolge getauscht. Bei 25 liegt kNN mit 17% Fehlerrate deutlich vor der LDA mit 25%. Jetzt liegt die LDA mit 34% Fehlerrate knapp vor kNN mit 37%. Der Unterschied beträgt aber weniger als eine Standardabweichung, ist daher nicht statistisch signifikant.

An der deutlich schlechteren Fehlklassifikationsrate des ungeclusterten Random Forest (34%) und des ungeclusterten k-nächste Nachbarn Klassifikators (50%) kann man deutlich den Leistungsgewinn durch Clusterung der Klangmerkmale erkennen.

Tabelle 8.6: Klassifikationsleistung der MFCCs beim 59 Klangfarben Problem

Klassifikator	Parameter	Fehler in %	Std.Abw. in %
SVM-Poly	$C = 1,\ d = 2$	19	3.0
SVM-RBF	$C = 1.5,\ \sigma = 0.011$	36	4.3
SVM-ARBF	$C = 0.6,\ \sigma = 0.11$	19	3.4
SVM-Lin	$C = 0.8$	20	3.2
RandFor	$U = 500,\ V = 7$	28	2.6
LDA		34	3.1
kNN	$k = 4$	37	1.8
RandFor ungecl.	$U = 500,\ V = 5,\ sampsize = 80000$	34	1.9
kNN ungecl.	$k = 1$	50	0.8

Die hier präsentierten Klassifikationsleistungen sind im Vergleich zu den in Klapuri und Davy (2006) zusammengefassten Publikationen sehr gut. In den dort beschriebenen Problemstellungen sind Fehlklassifikationsraten erreicht worden, die zwischen 5% für ein 11 Klassenproblem und 70% für ein 29 Klassenproblem liegen. Die Ergebnisse sind aber nicht direkt mit der hier vorliegenden Problemstellung vergleichbar, da sich sowohl die Klassenanzahl als auch die Typen der Klassen deutlich unterscheiden (z.B. Instrumentengruppen gegenüber einzelnen Instrumenten).

Durch Kombination der MFCCs und PLPs konnte keine weitere Steigerung der Klassifikationsleistung erreicht werden.

Für eine weitere Steigerung müssen die Defizite der klassischen distanzbasierten Clusterverfahren insb. bei einigen Blasinstrumenten behoben werden. Hierzu sind in erster Linie die Ergebnisse des Clusterverfahrens auch bei schwierigen Klängen zu stabilisieren. Ein vielversprechender Ansatz ist hierbei die Einführung von Nebenbedingungen in Form einer Ordnungsrestriktion, die im nächsten Kapitel eingeführt wird.

9 Zeitliche Ordnung beim Clustern

Die Kapitel 9.1 und 9.3 bis 9.5 basieren auf den in Krey, Ligges und Leisch (2014) präsentierten Ergebnissen.

9.1 Vorteile zeitlicher Ordnung beim Clustern

Wie in Kapitel 6 beschrieben, liefern klassische Clusterverfahren bei der Anwendung auf Musiksignaldaten zwar oft (siehe z.B. Abbildung 6.1, 6.2, 6.3), aber nicht immer gute Ergebnisse (siehe Abbildung 6.5 und 6.4).

Eine zufällige Initialisierung erfasst die unter Umständen sehr kleinen Cluster von sehr kurzen Klangphasen (Anblasen eines Blasinstruments) nicht. Für die klassischen distanzbasierten Clusterverfahren ist eine Unterteilung von sehr großen Clustern, die z.B. durch ein Vibrato Klangveränderungen aufweisen, beim Optimieren des mittleren quadratischen Fehlers oft lohnenswerter als das Abtrennen sehr kurzer Klangphasen, die sich deutlich von den anderen unterscheiden. Dies tritt insb. bei Blasinstrumenten auf und führt zu Clusterergebnissen, die keine praktisch sinnvolle Interpretierbarkeit haben und oft sehr instabil sind, d.h. durch geringe Änderungen an den Klangmerkmalen entstehen ganz andere Clusterungen. Bei längeren Tonaufnahmen mit mehreren Klängen oder gar ganzen Musikstücken tritt dieses Verhalten verstärkt auf und schränkt die Verwendbarkeit der Clusterergebnisse stark ein.

Gleichzeitig ignorieren die klassischen Clusterverfahren die zeitliche Struktur der Klangmerkmale, die mit einem gleitenden Fenster aus einem gesampleten Tonsignal (also einer Zeitreihe) extrahiert wurden. Eine Formulierung dieser zeitlichen Abhängigkeit als eine Gruppenabhängigkeit (Leisch und Grün, 2006) ist nicht mit ausreichender Allgemeingültigkeit möglich, die eine direkte Verwendung mit unterschiedlichen Arten von Tonaufnahmen ermöglicht. Man kann diese zeitliche Ordnung aber als eine Ordnungsrestriktion betrachten. Dies ermöglicht eine sinnvolle Interpretation der so erhaltenen Cluster als Klangphasen bei einzelnen Tönen (z.B. Anblas-/Einschwing-, Halte-, Ausklingphase) oder Teile eines größeren Musikstücks (z.B. Strophe, Refrain, Solo bei U-Musik/populärer Musik). Man kann diese Strukturelemente dann für eine weitergehende Analyse heranziehen.

9.2 Generische Zeitreihensegmentierung

Der erste evaluierte Ansatz, die zeitliche Ordnung einer Musikaufnahme bei einer unüberwachten Strukturierung zu integrieren, basierte auf dem generischen Segmentierungsansatz von Graves und Pedrycz (2009) für multivariate Zeitreihen.

Die Methode basiert darauf, über erste Differenzen $x_{t+1,i} - x_{t,i}$ die Steigung jeder Variable i der Zeitreihe zu bestimmen und dann die Standardabweichung dieser Steigungen zu berechnen. Zielfunktion ist die Summe aus dem Produkt der mittleren Abweichungen und der Länge des Segments. Es ist also wieder ein Kriterium, das die mittlere quadratische Abweichung in den Segmenten/Clustern minimiert. Die Autoren des zuvor genannten Artikels schlagen den genetischen Algorithmus *Differential Evolution* zur Optimerung dieser Zielfunktion vor, da eine explizite Lösung sehr rechenaufwändig ist. Durch die generische Formulierung des Optimierungsproblems, kann jedes numerisches Optimierungsverfahren eingesetzt werden, das keine Ableitungen der Zielfunktion benötigt. Der Einsatz der *Covariance Matrix Adaption Evolution Strategy* hat bei deutlich geringerem Rechenzeitbedarf vergleichbare Resultate zu *Differential Evolution* geliefert. Die Abbildungen 9.1 und 9.2 zeigen das Resultat für eine der in Abschnitt 5.1 beschriebenen kombinierten Tonaufnahmen aus 10 McGill Klängen.

Der generische Segmentierungsansatz trifft einige der Übergänge sehr präzise, verfehlt andere aber komplett bzw. trennt Segmente, die von k-Means als Cluster identifiziert wurden, sehr kleinteilig. Die nicht gefundenden Trennlinien werden auch nicht durch eine weitere Erhöhung der Segmentanzahl gefunden. Die Festlegung der Segementzahl ist ein weiteres Problem dieses Ansatzes. Schon bei dieser relativ einfachen zusammegesetzten Tonaufnahme, ist kein „Knick“ in der Verlustfunktion mehr zu identifizieren, ab dem durch eine Erhöhung der Segmentanzahl keine nennenswerte Reduktion der Zielfunktion mehr erreicht werden kann. Daher ist auf diesem Weg keine zuverlässige Identifikation einer sinnvollen Segmentanzahl möglich. Zusätzlich ist die Reproduzierbarkeit des Ergebnisses in der sehr „hügeligen“ Landschaft der Zielfunktion sehr schwierig. Daher ist dieser Ansatz in der vorliegenden Arbeit nicht weiter verfolgt worden.

9.3 Order Constrained Solutions in k-Means Clustering (OCKC)

Steinley und Hubert (2008) haben daher vorgeschlagen das k-Means Optimierungsproblem um eine Ordnungsrestriktion zu ergänzen, um das Risiko einer Konvergenz

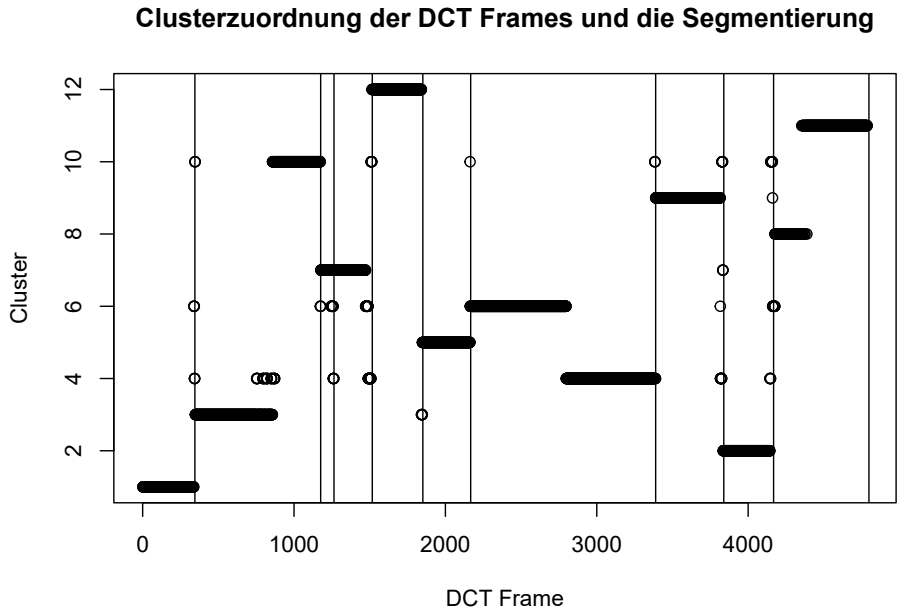

Abbildung 9.1: Die wahren Trennlinien der aus 10 McGill Klängen kombinierten Tonaufnahme und eine k-Means Clusterung

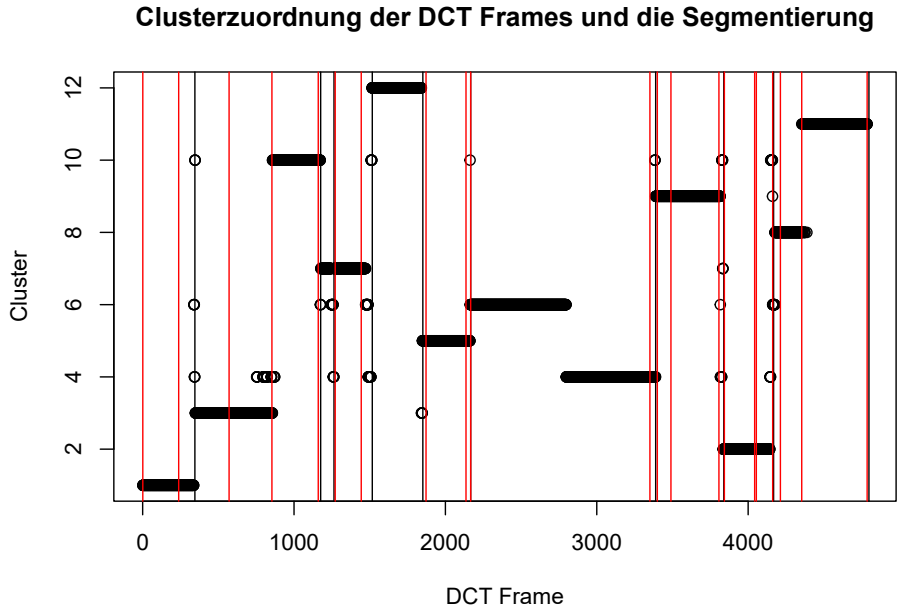

Abbildung 9.2: Segmentierung der aus 10 McGill Klängen kombinierten Tonaufnahme mittels generischer Zeitreihensegmentierung

in einem lokalen Minimum zu reduzieren. Sie nennen ihre Methode *Order Constrained Solutions in k-Means Clustering (OCKC)*.

Die Ordnungsrestriktion lässt sich als zusätzliche Bedingung an die C_i in Formel 6.1 bzw. 6.2 formulieren. Sei π die Permutation, die die natürliche Reihenfolge der Merkmalsvektoren gemäß der Ordnungsrestriktion umsortiert. Damit sind $\pi_i := \pi(Z)_i$, wobei $Z = (Z^1, Z^2, \ldots, Z^N)$, die gemäß der Ordnungsrestriktion permutierten Merkmalsvektoren. Dann muss für die Partitionierung $C = \{C_1, C_2, \ldots, C_k\}$ des Datensatzes gelten

$$\begin{aligned} C_1 &= \{\pi_1, \pi_2, \ldots \pi_{n_1}\} \\ C_2 &= \{\pi_{n_1+1}, \pi_{n_1+2}, \ldots, \pi_{n_1+n_2}\} \\ &\vdots \\ C_K &= \{\pi_{\sum_{j=1}^{K-1} n_j+1}, \pi_{\sum_{j=1}^{K-1} n_j+2}, \ldots, \pi_N\}, \end{aligned}$$

wobei n_i die Anzahl der Objekte in C_i ist und $N = \sum_{j=1}^{k} n_j$ ist die Anzahl aller Merkmalsvektoren. Allgemein formuliert:

$$C_k = \{\pi(Z)_{\sum_{j=1}^{k-1} n_j+1}, \pi(Z)_{\sum_{j=1}^{k-1} n_j+2}, \ldots, \pi(Z)_{\sum_{j=1}^{k} n_j}\}$$

Das k-Means Optimierungsproblem (siehe Formel 6.2) lässt sich folgendermaßen umschreiben:

$$SSD\left(\bigcup_{l=1}^{K} C_k\right) = \sum_{k=1}^{K} \sum_{Z \in C_k} \sum_{j=1}^{p} \left(Z_j - m_j^{(k)}\right)^2 = \sum_{k=1}^{K} \frac{1}{2N_k} \sum_{Z, Z' \in C_k} \sum_{j=1}^{p} (Z_j - Z'_j)^2.$$

In dieser Formulierung können alle für die Berechnung der optimalen Lösung des Order Constrained k-Means Problem notwendigen, kumulierten Distanzen (Summe der quadrierten Distanzen, wenn der Cluster in π_i startet und in π_j endet) vorausberechnet werden (Steinley und Hubert, 2008). Nach der Berechnung der kumulierten Distanzen muss dann die Kombination von Clusterlängen gefunden werden, sodass die Summe der kumulierten Distanzen über alle Cluster minimiert wird.

Eine Implementierung dieser Methode in der Programmiersprache `R` (R Core Team, 2014) ist von Hoffmeister (2009) geschrieben worden und funktioniert im Framework des sehr flexiblen Softwarepakets `flexclust` (Leisch, 2006), einem vielseitigen Werkzeug für K-Zentroid Clusterverfahren.

Wie durch Ersetzen des klassischen k-Means Clusterings mit Order Constrained k-Means Clustering die Klassifikationsleistung in der Problemstellung aus dem Kapitel 8 verbessert werden kann, ist in Kapitel 9.4.2 beschrieben.

Die durch die Einführung der Ordnungsrestriktion deutlich erhöhte Stabilität der Clusterungen ermöglicht es deutlich komplexere Tonaufnahmen wie komplette Musikstücke automatisch in unterschiedlich klingende Phasen zu unterteilen, die eine weitergehende musikalische Analyse ermöglichen. Diese Anwendungsmöglichkeit wird in Kapitel 9.5.1 präsentiert.

9.4 Verbesserung der Clusterung von Klangmerkmalen durch Order Constrained Clustering

In Kapitel 8 ist beschrieben worden, wie man durch Clusterung von verschiedenen Klangmerkmalen (insb. MFCCs und PLPs) und Verwendung der so erhaltenen Clusterzentren anstatt der rohen Klangmerkmale oder eines einzelnen Repräsentaten die Leistung verschiedener Klassifikationsverfahren bei der Erkennung von Instrumenten so stark verbessert, dass sogar eine Unterscheidung von Klangfarben dieser Instrumente, die durch unterschiedliche Spielweisen erzeugt werden können, mit hoher Genauigkeit möglich ist.

Bei der Analyse der mit dem klassischen distanzbasierten k-Means Verfahren erhaltenen Clusterungen ist aufgefallen, dass die Ergebnisse oft nicht interpretierbar und instabil sind (siehe Abschnitt 9.1). In Kapitel 9.3 ist daher das von Steinley und Hubert (2008) präsentierte *Order Constrained k-Means Clustering* eingeführt worden. Die Methode ergänzt das k-Means Clusterverfahren mit einer Nebenbedingung, um z.B. eine zeitliche Ordnung innerhalb der Cluster zu erzwingen.

Die von Hoffmeister (2009) geschriebene R Implementierung des Verfahrens wird im folgenden Abschnitt verwendet, um die Clusterungen der aus den Aufnahmen der McGill Instrumentendatenbank extrahierten Klangmerkmale zu stabilisieren und so die Klassifikationsleistung bei der Erkennung der Klangfarben zu verbessern (Kapitel 8).

9.4.1 Order Constrained Clustering von Instrumentenklängen

Wendet man OCKC auf die in Abschnitt 6.1 und 9.1 als „schwierig“ zu clusternde Klänge an, so erhält man durch die Einführung der der Nebenbedingung der zeitlichen Ordnung stabile Clusterergebnisse, die man gut interpretieren kann. Für das Kontrafagott ist dies in Abbildung 9.3 zu sehen.

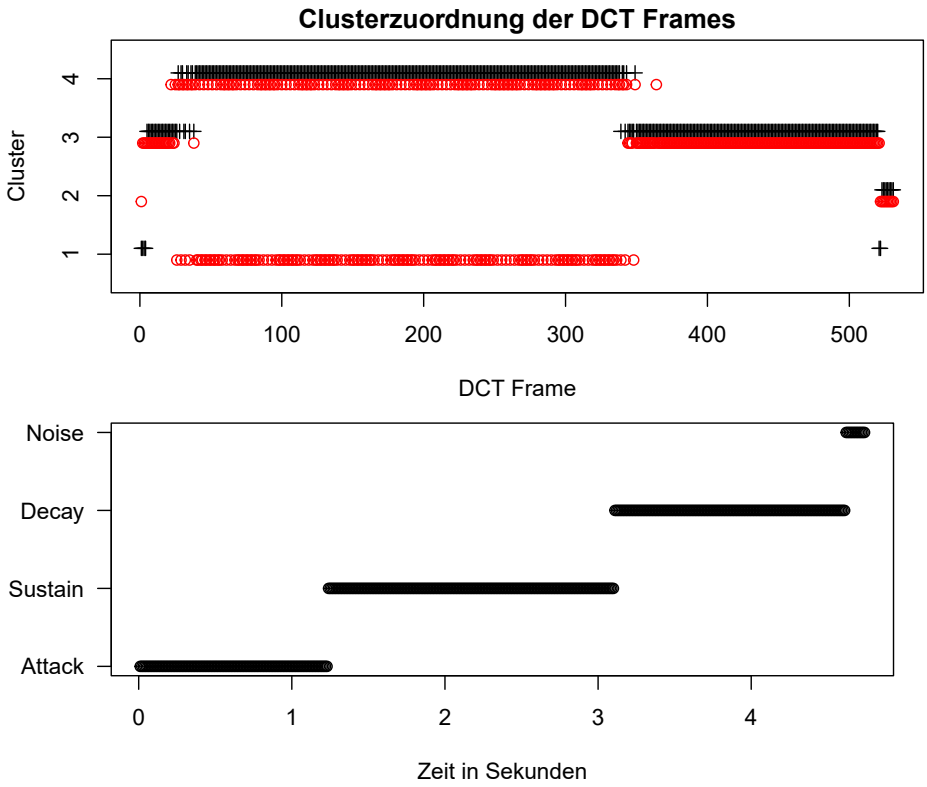

Abbildung 9.3: Vergleich der Clusterung eines Kontrafagotttons zwischen k-Means (oben) und Order Constrained k-means Clustering (unten)

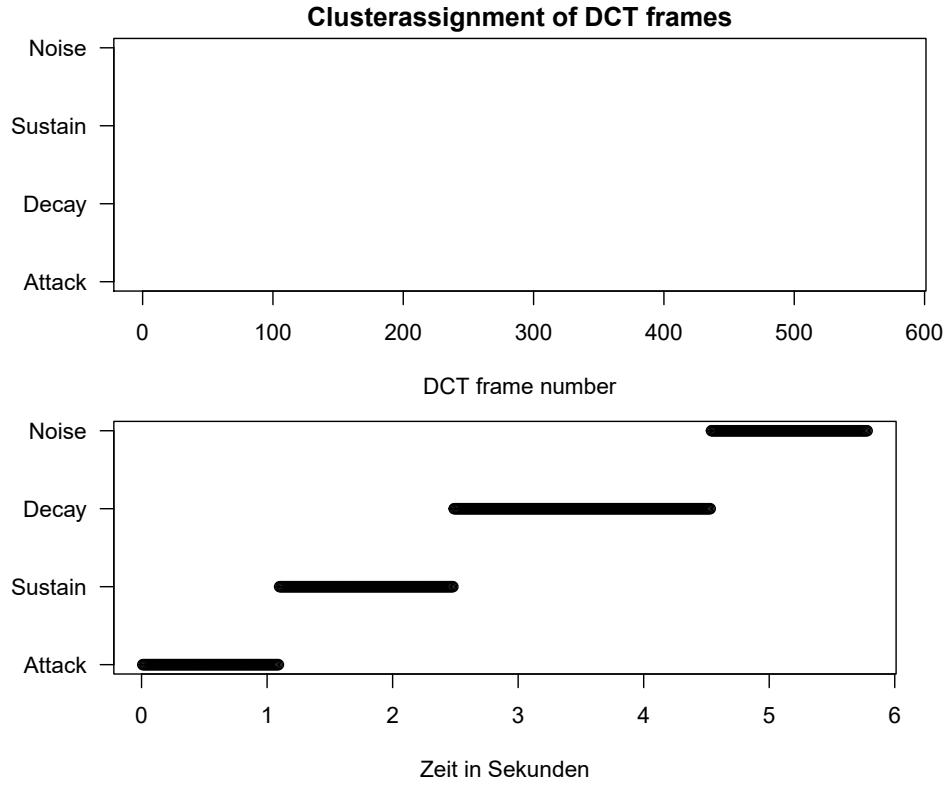

Abbildung 9.4: Vergleich der Clusterung eines Klaviertons zwischen k-Means (oben) und Order Constrained k-means Clustering (unten)

Gleichzeitig hat sich das Clusterergebnis für die „einfacher“ zu clusternden Klänge durch die Einführung der Nebendingung nicht verschlechtert. Die Clusterung des Klaviertons aus Abschnitt 6.1 mit OCKC ist in Abbildung 9.4 zu sehen. Es sind zwei Unterschiede zu erkennen: Aufgrund der Ordnungsrestriktion konnten die wenigen Millisekunden Rauschen am Anfang der Aufnahme nicht vom Anschlagscluster abgetrennt und dem Rauschcluster zugeordnet werden. Die andere Änderung ist eine leichte Verschiebung des Übergangs zum Cluster für die Ausklingphase.

9.4.2 Verbesserung der Instrumentenklassifikation durch OCKC

Bei der Klassifikation der Instrumentenklänge wird sich jetzt auf das schwierigere 59 Klassenproblem zur Erkennung der unterschiedlichen Klangfarben konzentriert, da hier noch ein deutliches Verbesserungspotential im Vergleich zu Kapitel 8 besteht.

Bei den betrachteten Klassifikationsverfahren und Klangmerkmalen erfolgt eine Fokussierung auf die im Kapitel 8 erfolgreichsten Methoden. Es werden daher die MFCC

Klangmerkmale und als Klassifikationsverfahren die SVM mit polynomiellem- sowie Gauß-RBF-Kernel und der Random Forest verwendet.

Die Methodik beim Vergleich der Klassifikationsverfahren ist identisch wie in Kapitel 8. Es werden also wieder die Implementationen in den `R` (R Core Team, 2014) Paketen `kernlab` (Karatzoglou u. a., 2004) und `RandomForest` (Liaw und Wiener, 2002) verwendet. Die Verfahren sind wieder mit `mlr` (Bischl u. a., 2016) in einer geschachtelten Kreuzvalidierung mit jeweils 5 Iterationen optimiert und validiert worden.

Tabelle 9.1: Klassifikationsleistung der MFCCs beim 59 Klangfarben Problem mit vorherigem Order Constrained Clustering im Vergleich zu klassischem distanzbasierten Clustering

	Fehlklassifikationsrate in Prozent	
Klassifikator	k-Means	OCKC
SVM-Poly	19	19
SVM-RBF	36	17
RandFor	28	19

Durch dieses Vorgehen ist eine deutliche Verbesserung der Klassifikationsleistung erreicht worden. Dies ist in Tabelle 9.1 dargestellt. Die Fehlklassifikationsrate des Random Forest verbessert sich von 28% auf 19%. Die SVM mit Gauß-RBF Kernel steigert sich von 36% auf 17%. Bei Verwendung des polynomiellen Kernels ist bei der SVM keine Veränderung festzustellen. Sie verbleibt bei den weiter sehr guten 19% Fehlklassifikationsrate.

Eine detaillierte Übersicht der Klassifikationsergebnisse für jedes Instrument/jede Klangfarbe für die SVM mit Gauß-RBF Kernel ist in Abbildung 9.5 zu sehen. Es handelt sich hierbei um eine als Heatmap aufbereitete Wahrheits-/Konfusionsmatrix. Hierzu wurden die relativen Häufigkeiten auf eine Grauskala übertragen. Ein schwarzer Punkt in einer Spalte bedeutet, dass 100%, also alle verfügbaren Aufnahmen dieser Klangfarbe (Zeile der Matrix), dieser Klasse zugeordnet worden sind. Weiß entspricht einem Wert von 0%, also keine Aufnahme wurde dieser Klasse zugeordnet.

Anhand der sehr dunklen Diagonale in der Grafik ist die sehr gute Leistung der Klassifikation zu erkennen. Die meisten Klangfarben sind der richtigen Klasse zugeordnet worden. Große Teile der Fehlklassifikationen sind innerhalb der gleichen Instrumentenfamilie (verschiedene Streichinstrumente) oder zwischen verschiedenen Klangfarben eines Instruments (z.B. beim Klavier und den Flöten) passiert. Dies kann man anhand der grauen Quadrate um die Diagonale erkennen. Die deutlich entfernt von

Anzahl der Cluster k	CPU Zeit in Sek.
2	0.9
3	1.0
4	1.1
5	6.2
6	558.4

Tabelle 9.2: Berechnungszeiten für Order Constrained k-Means Clustering mit der Implementation von Hoffmeister (2009) für unterschiedliche Clusteranzahlen bei einem Datensatz mit $N = 168$

der Diagonale liegenden grauen Punkte sind sehr hell, repräsentieren also sehr geringe relative Häufigkeiten. Ein großer Teil ist aber auch durch Verwandtschaften zwischen den Instrumenten zu erklären, wie bei den Holzblasinstrumenten Klarinette, Oboe und Saxophon oder der E-Gitarre und dem Klavier (beides Saiteninstrumente mit Stahlsaiten). Diese Fehlklassifikationen sind daher aufgrund ihrer Schwierigkeit bei der Unterscheidung der entsprechenden Klangfarben sehr gut zu erklären.

9.5 Recursive Order Constrained Clustering

Aufgrund der erfolgreichen Verwendung von OCKC in der Vorverarbeitung für die Klassifikation von Instrumentenklängen, ist das nächste Ziel die Clusterung von ganzen Musikstücken, um automatisiert zusammenhängende Teile in der Komposition zu erkennen. Um dies zu ermöglichen, muss die Anzahl der Cluster k deutlich erhöht werden. Hierbei ist aufgefallen, dass die `R` Implementierung von OCKC, geschrieben von Hoffmeister (2009), ein sehr ungünstiges Laufzeitverhalten hat. Wie man Tabelle 9.2 entnehmen kann, steigt die Laufzeit ab $k = 4$ sehr stark an, sodass $k \geq 6$ praktisch nicht einsetzbar ist.

Bei den Tests mit unterschiedlichen Clusterzahlen bei verschiedenen Tonaufnahmen ist aufgefallen, dass die Clustergrenzen sich bei steigendem k meist nur um wenige Zeitfenster verschieben. Dies ermöglicht ein rekursives Vorgehen bei der Clusterung, wobei ausgehend von einer initialen Clusterung mit kleinem k die erhaltenen Cluster erneut geclustert werden und solange rekursiv vorgegangen wird, bis die gewünschte Anzahl an Cluster gefunden ist. Der zusätzliche Quantisierungsfehler durch dieses Vorgehen ist, aufgrund der Beobachtung der minimalen Clustergrenzenverschiebung bei Erhöhung von k gering.

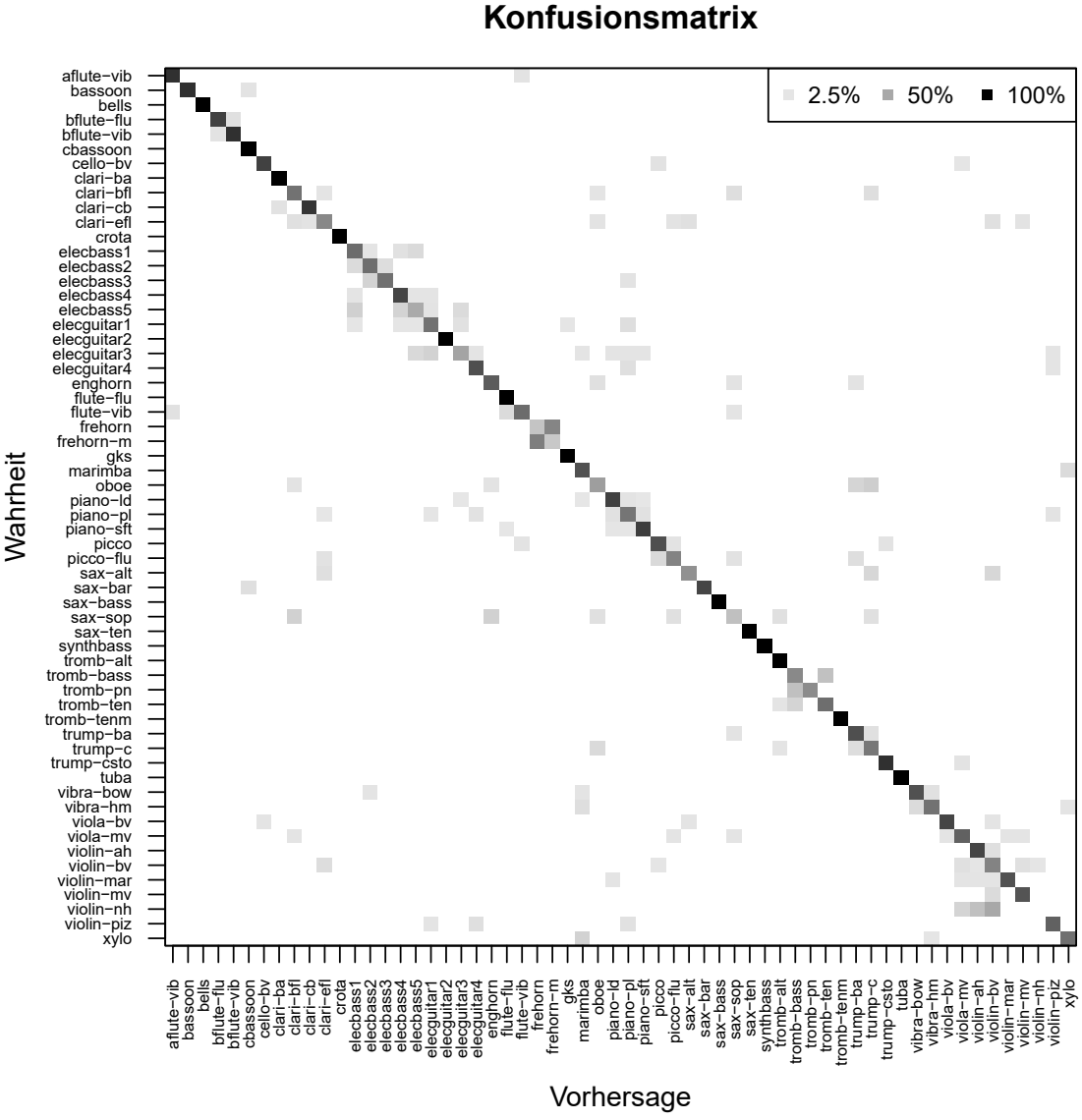

Abbildung 9.5: Heatmap der Konfusionsmatrix zur Darstellung der Klassifikationsergebnisse der SVM mit Gauß-RBF Kernel

Implementierung und formale Beschreibung

Konkret wird mit einer initialen Clusterung mit $k = 2$ gestartet. Danach wird der Cluster m identifiziert, der bei einer Aufspaltung in 2 Cluster die größte Reduktion der Summe der quadrierten Distanzen zur Folge hat. Dieser Cluster m wird dann in 2 Cluster aufgeteilt um die Clusterung für $k = 3$ zu erhalten. Dies wird fortgesetzt bis die gewünschte Clusterzahl k erreicht ist.

Formal bedeutet dies, dass man ausgehend von $k = j$ mit der Clusterung $C^1, \ldots, C^j$ die Clusterung mit $k = j + 1$ dadurch erhält, dass man den Cluster m in zwei Teile clustert, der die größte Reduktion der Summe der quadrierten Distanzen zur Folge hat. Man wählt also:

$$m = \arg\max_{i=1,\ldots,j} SSD(C^i) - (SSD(C_1^i) + SSD(C_2^i))$$

Hierbei sind C_1^i und C_2^i die Cluster, die man erhält, wenn man Cluster C^i mit OCKC in $k = 2$ Cluster aufteilt. $SSD(C^i)$ bezeichnet die Summe der quadrierten euklidischen Distanzen des Clusters C^i. Die neue Clusterung mit $k = j + 1$ ist dann $C^1, \ldots, C^{m-1}$, C_1^m, C_2^m, $C^{m+1}, \ldots, C^j$.

Da bei steigendem k in jedem Teilschritt des Algorithmus immer kleinere Teile des Datensatzes in $k = 2$ Teilcluster aufgeteilt werden, kann so eine starke Reduktion der Ausführungszeit erreicht werden. In Tabelle 9.3 können für den gleichen Testdatensatz wie zuvor die Berechnungszeiten durch rekursive Anwendung von OCKC bis $k = 10$ abgelesen werden.

9.5.1 Musical Structure Analysis

Für die Musikgenreklassifikation ermöglicht die Clusterung der Klangmerkmale eine sehr viel differenziertere Zuordnung eines Musikstücks zu (möglicherweise unterschiedlichen) Genres. Kompositionen von Bands wie „Queen“ weisen oft starke stylistische Unterschiede zwischen den verschiedenen Teilen eines Musikstücks auf. Der Song „Bohemian Rhapsody“ ist hierfür ein Beispiel mit einer sehr großen musikalischen Spannweite. Für solche Musikstücke ist die etablierte Selektion eines kurzen Abschnitts in der Mitte des Musikstücks und Berechnung der Klangmerkmale auf diesem Teilstück unzureichend, da sie nur den Stil dieses Teilstücks erfasst. Wenn durch eine Clusterung alle relevanten Teilstücke eines Musikstücks identifiziert werden, können Klangmerkmale von jedem dieser Teilstücke für die Klassifikation verwendet werden und so eine präzisere Vorhersage ermöglichen.

	CPU Zeit in Sek. bei $N = 168$	
Anzahl der Cluster k	Original	Rekursive Anwendung
3	1.0	2.1
4	1.1	2.9
5	6.2	3.1
6	558.4	3.2
7		3.3
8		3.4
9		3.5
10		3.6

Tabelle 9.3: Berechnungszeiten für Order Constrained k-Means Clustering bei einer rekursiven Anwendung der Implementation von Hoffmeister (2009) im Vergleich zur direkten Verwendung für unterschiedliche Clusteranzahlen mit einem Datensatz mit $N = 168$

Ferner kann die durch die Clusterung erhaltene Strukturierung eine hilfreiche Arbeitserleichterung für den Musikwissenschaftler sein. Die durch die Clusterung identifizierten Teilstücke können dann tiefergehend analyisiert werden.

Cluster Ergebnisse

Die Anwendung von Rekursivem Order Constrained k-Means zur Identifzierung der Struktur eines Musikstücks soll anhand von zwei Werken der Populärmusik erfolgen. Das erste Stück ist der Depeche Mode Song „Stripped“. Es handelt sich um elektronische Musik aus den frühen 1980er Jahren, die eine sehr klare Struktur mit kontrastierenden Klangelementen aufweist. Queens „Bohemian Rhapsody“ ist das zweite, deutlich längere Werk mit einer sehr abwechslungsreichen Komposition, die von einer Pop Ballade über opernartigen Elementen, instrumentaler Klaviermusik, einem Gitarrensolo bis hin zu Hardrockelementen sich sehr vieler Genres bedient.

Da für die Strukturierung eines Musikstücks keine Zeitauflösung im hundertstel Sekunden Bereich notwendig ist, ist die Merkmalsextraktion hier mit 3 Sekunden langen, nicht überlappenden Zeitfenstern bei der gefensterten Fouriertransformation erfolgt.

In Abbildung 9.6 ist die Struktur von Depeche Modes „Stripped“ zu sehen. Hierzu wurde die Kosinusdistanz zwischen den aus der Aufnahme extrahierten MFCCs wie

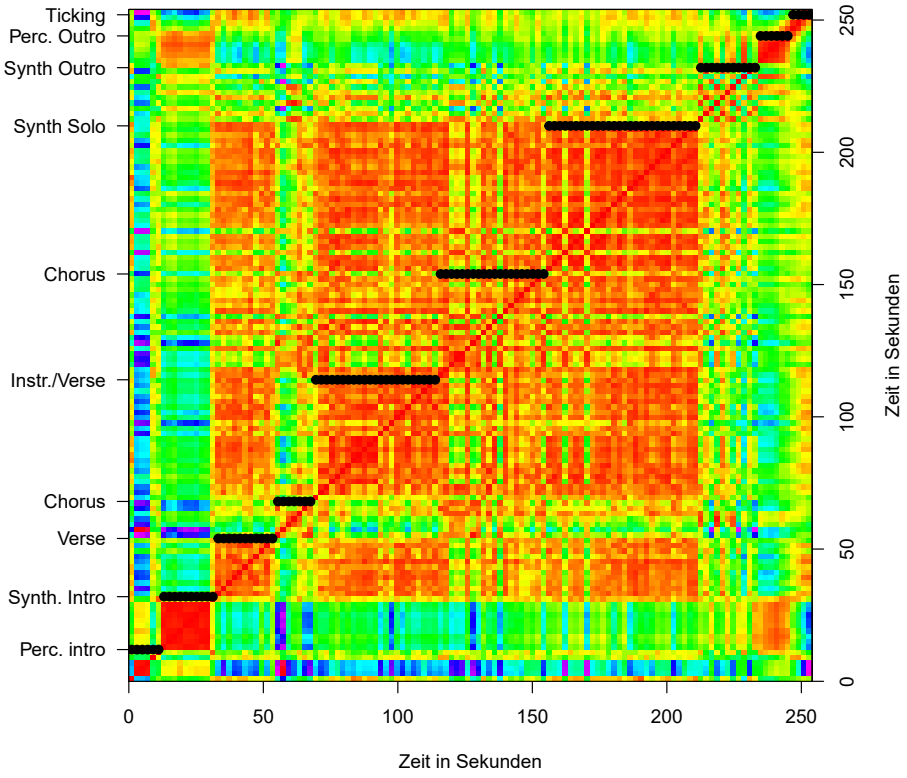

Abbildung 9.6: Musikalische Struktur von Depeche Modes „Stripped“

in Kapitel 4 beschrieben als Heatmap geplottet. Die dunklen Quadrate von Passagen mit sehr ähnlichem Klang sind in der grafischen Darstellung gut zu erkennen und die Cluster bilden diese Struktur sehr gut ab. Wendet man die in Kapitel 7 beschriebene Methodik zur Clusterzahlbestimmung an, so erhält man die hier gezeigte Clusterzahl von $k = 10$. Die Annotation der Cluster ist, aufgrund der übersichtlichen Struktur der Komposition, sehr einfach möglich.

Die Darstellung von Queens „Bohemian Rhapsody“ ist etwas komplizierter, wie man in Abbildung 9.7 sehen kann. In der Darstellung sind die Quadrate weniger dunkel, aber trotzdem gut zu erkennen. Die erhaltenen Cluster passen auch hier gut zu den grafisch erkennbaren Segmenten. Hört man sich das Stück an, so stellt man fest, dass die Clustergrenzen mit Änderungen der vorwiegend eingesetzten Instrumente oder Stimmen korrespondieren. Die anhand des minimalen Verhältnisses aus Interquartilsabstand und Median bestimmte Clusterzahl ist $k = 14$.

Die erhaltene Clusterung hält auch einem Vergleich mit der von Musikwissenschaftlern manuell erstellten Strukturierung des Stücks stand (*Wikipedia: Bohemian Rhapsody* 2021; *Queen Songs, Bohemian Rhapsody* 2015). Die von Experten erstellte Strukturierung besteht aus 6 Segmenten:

Intro 0:00 - 0:49

Ballade 0:49 - 2:37

Gitarren Solo 2:37 - 3:03

Opern-Parodie 3:03 - 4:08

Hard Rock 4:08 - 4:55

Outro 4:55 - 5:55

Der erste Cluster ist das *Intro*, die folgenden fünf Cluster sind die *Ballade* und der siebte Cluster ist das *Gitarrensolo*. Die *Opern-Parodie* ist der achte und der *Hard-Rock* Teil der neunte Cluster. Das *Outro* ist dann wieder in fünf kleine Cluster aufgeteilt.

9.6 Optimierung der OCKC Berechnung

Das im vorigen Kapitel vorgestellte rekursive Vorgehen hat zwar eine Möglichkeit geschaffen die bestehende OCKC Implementierung auch für die Analyse längerer Musikstücke zu verwenden, diese liefert zwar gute aber nicht mathematisch optimale

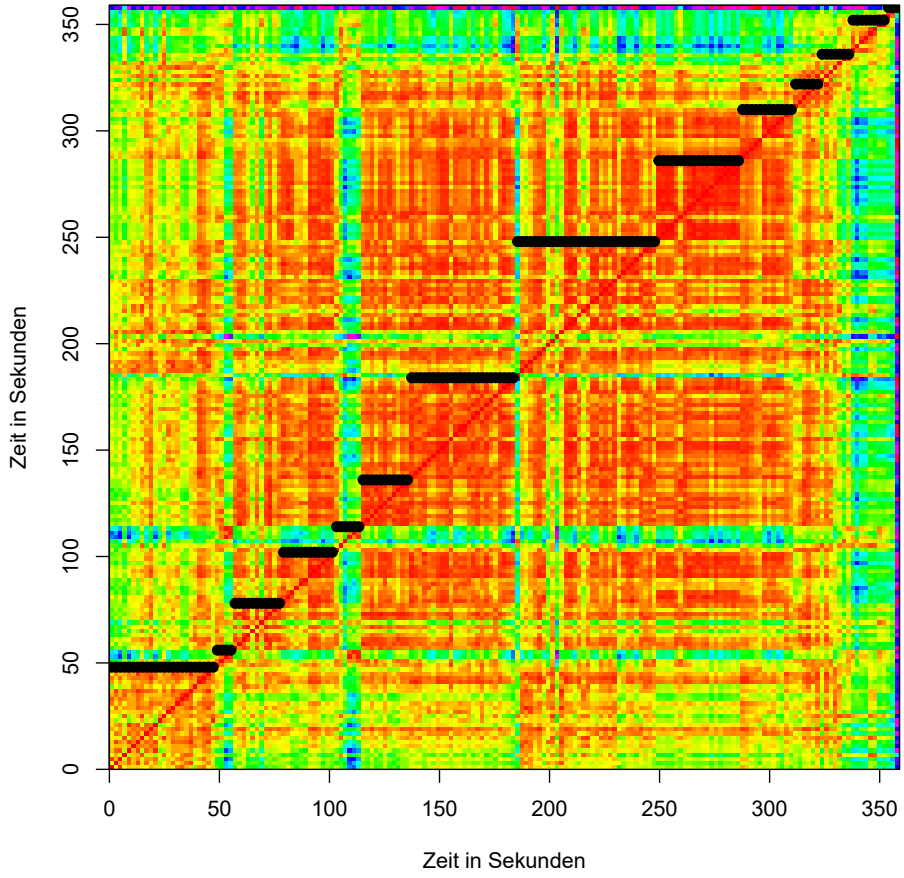

Abbildung 9.7: Musikalische Struktur von Queens „Bohemian Rhapsody“

Ergebnisse. Im folgenden Kapitel werden die Schwächen der Implementierung analysiert und eine Umformulierung des Algorithmus präsentiert, die sowohl für eine große Anzahl an Zeitframes (großes n) als auch für eine hohe Anzahl an Clustern (großes k) eine optimale Berechnung der Clusterung ermöglicht.

9.6.1 Probleme der existierenden Implementierung

Beim Profiling der initialen R Implementierung von Hoffmeister (2009) für OCKC besteht in beiden Hauptberechnungschritten *Berechnung der kumulierten Distanzen* und *Berechnung der optimalen Clusterung* starkes Optimierungspotenzial.

Clusterung Wie in Tabelle 9.2 dargestellt, steigt der Rechenzeitbedarf ab $k = 5$ rasant an und schon $k = 7$ ist nicht mehr praktikabel. Die Identifikation einer optimalen Clusterung ist ein kombinatorisches Optimierungsproblem, dessen Komplexität und damit der Rechenzeitbedarf extrem schnell in k wächst. Die initiale OCKC Implementierung für R löst das Optimierungsproblem mit einem naiven Ansatz, der systematisch alle Kombinationen durchprobiert, was zu dem beschriebenen Laufzeitverhalten führt. Auch die von Hoffmeister durchgeführte Auslagerung dieser Berechnungsfunktion in kompilierten C-Code sorgt nur für eine minimale Verbesserung der Laufzeit, da es das ursächliche Problem nicht beseitigt.

Kumulierte Distanzen Wählt man ein festes k und erhöht n, sieht man, dass für großes n die Berechnung der Matrix der kumulierten Distanzen ebenfalls zu einem Flaschenhals wird.

Es wird eine obere Dreiecksmatrix zur Speicherung der kumulierten Distanzen verwendet. Die Berechnung der Einträge erfolgt durch zwei geschachtelte Schleifen, wobei die innere Schleife abhängig von der Zeile der Matrix gewählt wird um nur die notwendigen Einträge zu berechnen. Hierzu müssen dann entsprechende Teilmatrizen aus der Distanzmatrix D extrahiert werden. Dieser Ansatz hat zwar den Vorteil, dass nur die wirklich benötigten Berechnungen durchgeführt werden, es wird also die Anzahl der Fließkommaoperationen (FLOPS) minimiert, aber es sind sehr ungünstige Speicherzugriffe notwendig. Die Einträge einer Matrix werden in R spaltenweise als Vektor mit Attributen für die Zeilen- und Spaltenanzahl abgespeichert. Das Extrahieren einer Teilmatrix mit j Spalten macht j Sprünge innerhalb dieses Vektors notwendig, um dann die notwendige Anzahl an Einträgen auszulesen. Solche Suchoperationen im Speicher sind langsam.

Zusätzlich kann die Berechnung der kumulierten Distanzen durch den Einsatz geschachtelter Schleifen, die von Fallunterscheidungen durchbrochen werden, nicht mit vektorisierten und hocheffizienten Numerikbibliotheken (insb. die *Basic Linear Algebra Subprograms (BLAS)*) beschleunigt werden. Die Befehlssatzerweiterungen für *Single Instruction Multiple Data (SIMD)* können daher nicht eingesetzt werden, wodurch der Großteil der Fließkommaleistung moderner Prozessoren brachliegt.

9.6.2 Berechnung der kumulierten Distanzen

Wie schon zuvor beschrieben, lässt sich das k-Means Optimierungsproblem folgendermaßen umschreiben

$$SSD\left(\bigcup_{l=1}^{K} C_k\right) = \sum_{k=1}^{K} \sum_{Z \in C_k} \sum_{j=1}^{p} \left(Z_j - m_j^{(k)}\right)^2 = \sum_{k=1}^{K} \frac{1}{2N_k} \sum_{Z,Z' \in C_k} \sum_{j=1}^{p} (Z_j - Z'_j)^2,$$

um alle für die Berechnung der optimalen Lösung des Order Constrained k-Means Problems notwendigen kumulierten Distanzen vorauszuberechnen (Steinley und Hubert, 2008).

Als Datenstruktur ist eine Liste `kd` von Vektoren verwendet worden, wobei das erste Listenelement alle kumulierten Distanzen enthält, wenn man von Zeitframe 1 startet. Es ist also ein Vektor der Länge N, dessen erster Eintrag immer Null ist, da bei Start und Ende im gleichen Zeitframe die Distanz zwischen Start- und Endframe Null ist. Das letzte Listenelement N ist ein Vektor der Länge 1 mit dem Wert Null. Im Vergleich zur zeilenweise aufgebauten Matrix der Hoffmeister Implementierung wird so die Hälfte des Speicherplatzes eingespart. Da Matrizen in `R` spaltenweise gespeichert werden, sind beim zeilenweise Schreiben einer Matrix unvorteilhafte Sprünge im Speicher notwendig. Daher ist die Verwendung von Vektoren aufgrund des zusammenhängenden Speicherzugriffs zusätzlich auch deutlich schneller.

Formulierung mit Matrixoperationen

Ausgehend von der Distanzmatrix D benötigt man für die effiziente Berechnung noch mehrere Hilfsobjekte. Dies ist zum einen eine Matrix I mit Einsen im oberen Dreieck und auf der Diagonalen. Zum anderen ist ihre transponierte Matrix I' notwendig (untere Dreiecksmatrix). Zur korrekten Normierung wird zusätzlich noch ein Vektor s der Spaltensummen von I benötigt, der in diesem Fall einfach ein Vektor von Eins

bis zur Anzahl der Zeilen von I ist, also im ersten Schritt N. Berechnet man nun das Matrixprodukt

$$R = I'DI,$$

so erhält man eine Matrix R auf deren Diagonale bis auf einen spaltenabhängigen Skalierungsfaktor die kumulierten Distanzen für einen Start im ersten Zeitframe und in Spalte j ein Ende in Zeitframe j enthält. Die korrekte Normierung erhält man dadurch, dass man jedes Element der Diagonale durch die entsprechende Spaltensumme von I, also den zugehörigen Eintrag von s, dividiert.

Die Berechnung des Vektors der kumulierten Distanzen mit Start in Zeitframe i erfolgt analog. Anstatt der vollständigen Matrizen D und I verwendet man hierfür aber die Teilmatrizen $D[i:n, i:n]$ und $I[i:n, i:n]$, die aus den Zeilen i bis n und den Spalten i bis n von D bzw. I bestehen.

Nachfolgend steht der für diese Berechnung notwendige `R` Quelltext:

```
I <- upper.tri(D, diag=TRUE)
triang.j <- function(j, I, D) {
  result <- diag((t(I) %*% D %*% I))/ seq(1, n-j+1)
  return(result)
}
kd <- mclapply(1:N, function(i)
 triang.j(i, I=I[i:n,i:n, drop=FALSE], D=D[i:n, i:n, drop=FALSE))
```

Optimierungsergebnis

Durch die Formulierung aller notwendigen Operationen zur Berechnungen der kumulierten Distanzen als Matrix-/Vektoroperationen kann die Laufzeit der Funktion drastisch reduziert werden. Verwendet man eine leistungsfähige BLAS Bibliothek, die für den eingesetzten Prozessor hochoptimierte Berechnungskernel bereitstellt (z.B. OpenBLAS oder die herstellerspezifischen Mathematikbibliotheken), so ist eine 10-fache Beschleunigung der Berechnung pro CPU Kern gegenüber der initialen Implementierung möglich. Ist die BLAS Bibliothek multithreadingfähig, so kann durch Parallelisierung über mehrere CPU Kerne eine weitere Beschleunigung erreicht werden.

Da die Berechnung der einzelnen Vektoren der kumulierten Distanzen vollständig unabhängig voneinander ist, kann zusätzlich zur Parallelisierung in der BLAS Bibliothek auch auf dieser Ebene parallelisiert werden. Die Grenzen der Parallelisierung

sind theoretisch erst bei der gleichzeitigen Berechnung aller Zeitframes erreicht, in die die Tonaufnahme bei der gefensterten Fouriertransformation zerlegt worden ist. Somit können auch Prozessoren mit einer sehr hohen Anzahl an CPU Kernen effektiv genutzt werden.

9.6.3 Clusterung durch dynamische Programmierung

Die Berechnung der optimalen Clusterung ist ein diskretes, kombinatorisches Optimierungsproblem. Solche Probleme lassen sich durch *dynamische Programmierung* effizient lösen. Schon im Originalartikel zu OCKC wird von den Autoren ein rekursiver Algorithmus skizziert, der auf dem Prinzip der dynamischen Programmierung basiert.

Rekursive Algorithmen lassen sich zwar sehr elegant formulieren und programmieren, weisen aber oft eine sehr viel geringere Berechnungsgeschwindigkeit auf als eine äquivalente iterative Formulierung. Daher ist der hier vorgestellte Algorithmus iterativ formuliert und implementiert.

Das Prinzip der dynamischen Programmierung basiert darauf, dass man das Optimierungsproblem in viele gleichartige Teilprobleme zerlegen kann und die Lösung für das ursprüngliche Problem sich aus optimalen Lösungen der Teilprobleme zusammensetzen lässt. Die berechneten Lösungen für die Teilprobleme werden systematisch gespeichert und können so effizient für die Zusammensetzung der Lösung des komplexeren Problems wiederverwendet werden. Das Konzept ist von Bellman entwickelt worden und in Bellman (1957, 1972) beschrieben.

Für die Bestimmung der optimalen Clusterung mit dynamischer Programmierung sind zwei Teilschritte notwendig. Im ersten Schritt werden die resultierenden Summen der quadrierten Distanzen für alle möglichen Start- und Endpunkte eine Clusters berechnet. Danach muss aus diesen Kandidaten für die gewünschte Clusteranzahl k die optimale Lösung extrahiert werden.

Summe der Distanzen für optimale Clusterungen

Der erste Cluster startet immer mit Zeitframe 1 und als Endpunkte kommen alle Zeitframes in Frage, die weiter als die Mindestclustergröße h vom Startframe entfernt sind, also Zeiframe h bis N. Man erhält also den Vektor der kumulierten Distanzen `cv` aus den Einträgen `h` bis `N` des ersten Listenelements (da Startframe 1) der Liste der kumulierten Distanzen `kd`. Für den ersten Cluster sind die Gesamtlänge aller Cluster

l1 und die Länge des aktuellen Clusters p identisch. Die möglichen Längen sind die ganzzahlige Sequenz von der Mindestclustergröße h bis zur Gesamtanzahl der Zeitframes N. Diese Werte werden alle als erstes Listenelement von curRes gespeichert:

```
curRes <- list()
l1 <- h:n
curRes[[1]] <- list(cv=kd[[1]][l1], l=l1, p=l1)
```

Für jeden weiteren Cluster ck bis zur gewünschten Gesamtanzahl der Cluster wird auf dem vorherigen Ergebnis aufgebaut:

Zuerst wird die Anzahl der Lösungen im vorherigen Iterationsschritt bestimmt:

```
nlold <- length(curRes[[ck-1]]$l)
```

Anschließend wird die Anzahl der möglichen Lösungen für den aktuellen Cluster bestimmt. Dies ist die Gesamtanzahl der Zeitframes n minus die Anzahl der erstellen Cluster ck mal der Clustermindestgröße h plus Eins.

```
nlnew <- n-ck*h+1
```

Danach müssen für jeden möglichen Endpunkt des vorherigen Clusters alle möglichen Endpunkte des aktuellen Clusters und die daraus resultierenden Summen der Distanzen berechnet werden. Hierzu wird über alle Endpunkte des vorherigen Clusters iteriert, mit Ausnahme der letzten h Endpunkte, da mit diesen Endpunkten des vorherigen Clusters kein weiterer Cluster mit Mindestlänge h mehr erstellt werden kann. Der neue Cluster startet dann einen Zeitframe später. Die resultierende Summe der Distanzen ist der vorherige Wert plus die entsprechenden Werte aus dem nstart-ten Listenelement aus Abschnitt 9.6.2. Die so erhaltenen Werte werden spaltenweise in einer mit unendlich initialisierten Matrix gespeichert. Der Matrixeintrag a[i,j] enthält dann die summierten Distanzen für die Sequenz aller Datenpunkte bis zum Zeitframe ck * h + i - 1, wobei der Cluster ck eine Länge von j+h-1 Zeitframes hat.

```
a <- matrix(Inf, nlnew, nlold-h)
for(i in 1:(nlold-h)){
  nstart <- curRes[[ck-1]]$l[i] + 1
  a[i:nlnew,i] <- curRes[[ck-1]]$cv[i] + kd[[nstart]][h:(n-nstart+1)]
}
```

Im nächsten Schritt wird für jeden möglichen Endpunkt des Clusters ck, also für jede Zeile von a die Spalte identifziert, die die geringste Distanzsumme aufweist. Über den Spaltenindex kann dann die Länge des Clusters ck bestimmt werden. Diese Werte werden wieder im Vektor p abgelegt.

```
wmina <- max.col(-a, ties.method="first")
p <- h:(n-(ck-1)*h) - wmina +1
l <- (ck*h):n
```

Zum Schluss werden noch die Werte der kumulierten Distanzen der besten Lösungen aus der Matrix a extrahiert und alle Resultate als neues Listenelement gespeichert.

```
w <- sapply(1:nlnew, function(x) a[x,wmina[x]])
curRes[[ck]] <- list(cv=w, l=l, p=p)
```

Im nächsten Iterationsschritt startet man dann wieder bei den neuen Endpunkten der Cluster und berechnet optimale Lösungskandidaten für diesen Cluster bis man die gewünschte Clusteranzahl erreicht hat.

Dadurch, dass in jedem Iterationsschritt die Berechnungen immer nur davon abhängen, dass im Schritt davor ein bestimmter Endpunkt erreicht wurde, der genaue Weg zu diesem Endpunkt aber ignoriert werden kann, ist die Berechnung sehr schnell und effizient durchführbar. Zusätzlich muss der Algorithmus explizit nur für die größte gewünschte Clusteranzahl ausgeführt werden. Alle kleineren Clusteranzahlen werden implizit mitberechnet und müssen im nächsten Schritt nur ausgelesen werden.

Extraktion der optimalen Lösungen

Für jedes Element k[s] des Vektors der gewünschten Clusteranzahlen k wird auf folgendem Weg die Clusterstart- und -endpunkte bestimmt. Der Vektor y speichert die Endpunkte der Cluster. Es wird mit dem k[s]-ten Cluster gestartet. Der Endpunkt diese Clusters ist das letzte Listenelement des Längenvektors l. Danach bestimmt man den Endpunkt des k[s]-1-ten Clusters, indem man vom Endpunkt des Clusters k[s] dessen Länge abzieht. So geht man iterativ vor, bis man den Endpunkt des ersten Clusters bestimmt hat. Zum Schluss erstellt man noch einen Vektor der Länge N, der für jeden Zeitframe den zugehörigen Clusterindex enthält.

```
y <- numeric(k[s])
ind <- length(curRes[[k[s]]]$l)
y[k[s]] <- curRes[[k[s]]]$l[ind]
for(j in (k[s]-1):1){
  y[j] <- curRes[[j+1]]$l[ind] - curRes[[j+1]]$p[ind]
  ind <- which(curRes[[j]]$l == y[j])
}
cluster <- rep(1:k[s], diff(c(0,y[1:k[s]])))
```

So erhält man auf effiziente Weise für jede gewünschte Clusteranzahl die optimale Clusterung. Alle weiteren Kennzahlen zur Erstellung eines `flexclust` `kccasimple` Objekts sind ebenfalls in der Liste `curRes` enthalten und können mit Hilfe der `cluster` Vektoren der Clusterindizes für jeden Zeitframe extrahiert werden.

9.6.4 Geschwindigkeitsvergleich der Implementierungen

Wendet man die optimierte Implementierung auf den gleichen Testdatensatz wie in Kapitel 9.5 an, so zeigt sich eine weitere sehr starke Reduktion der notwendigen CPU Zeit für die Berechnung. Dies ist in Tabelle 9.4 dargestellt. Im Vergleich zur rekursiven Anwendung der Originalimplementierung erhält man hier nicht nur in der vorliegenden Anwendung, sondern auch in allen theoretischen Fällen die optimale Lösung, da es sich hier um eine effiziente explizite Lösung des vollständigen Optimierungsproblems handelt.

Die so erhaltenen Clusterungen der Beispielsong „Depeche Mode - Stripped“ und „Queen - Bohemian Rhapsody“ sind in den Abbildungen 9.8 und 9.9 dargestellt.

Durch die Optimierung hat sich die Berechnungsgeschwindigkeit der Order Constrained k-Means Clusterung sehr stark erhöht und ermöglicht die Anwendung des Verfahrens jetzt auch bei komplexen Datensätzen mit einer hohen Anzahl von Datenpunkten N und insb. einer hohen Clusteranzahl k. Während die ursprüngliche Implementierung ein exponentielles Laufzeitverhalten in k aufweist (dominiert durch das kombinatorische Optimierungsproblem), ist das Laufzeitverhalten der optimierten Implementierung kubisch in N (das heißt $O(N^3)$). Das Verhalten wird durch die Berechnung der kumulierten Distanzen, welche auf N Matrixmultiplikationen basiert, bestimmt. Die diskrete Optimierung mit dem gewählten Algorithmus spielt mit $O(k \cdot N^2)$ beim asymptotischen Laufzeitverhalten nur noch eine untergeordnete Rolle, da k sehr viel kleiner als N ist.

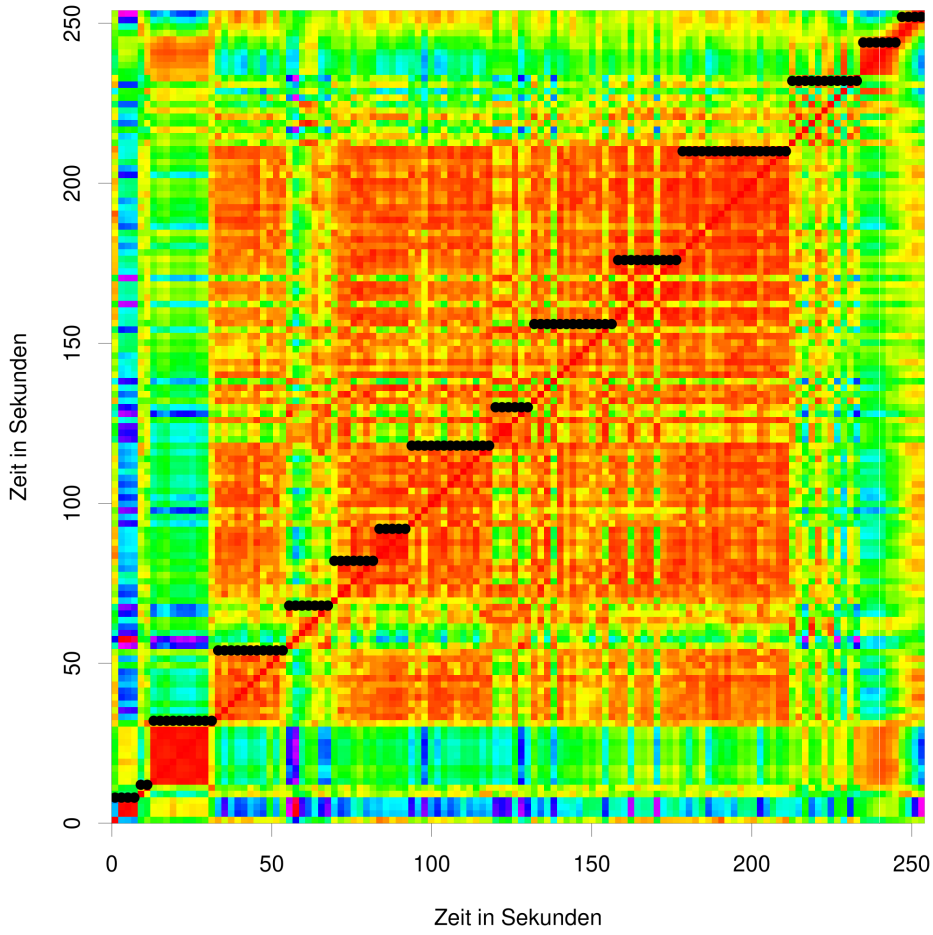

Abbildung 9.8: Musikalische Struktur von Depeche Modes „Stripped“ mit der optimierten OCKC Implementierung

Anzahl der Cluster k	CPU Zeit in Sek. bei $N = 168$		
	Original	Rekursive Anwendung	Optimierte Implementierung
2	0.9	1.2	0.55
3	1.0	2.1	0.6
4	1.1	2.9	0.62
5	6.2	3.1	0.62
6	558.4	3.2	0.63

Tabelle 9.4: Berechnungszeiten der optimierten Order Constrained k-Means Clustering Implementation für unterschiedliche Clusteranzahlen mit einem Datensatz mit $N = 168$ im Vergleich zur Originalimplementierung und der in Kapitel 9.5 vorgestellten rekursiven Anwendung des Originals

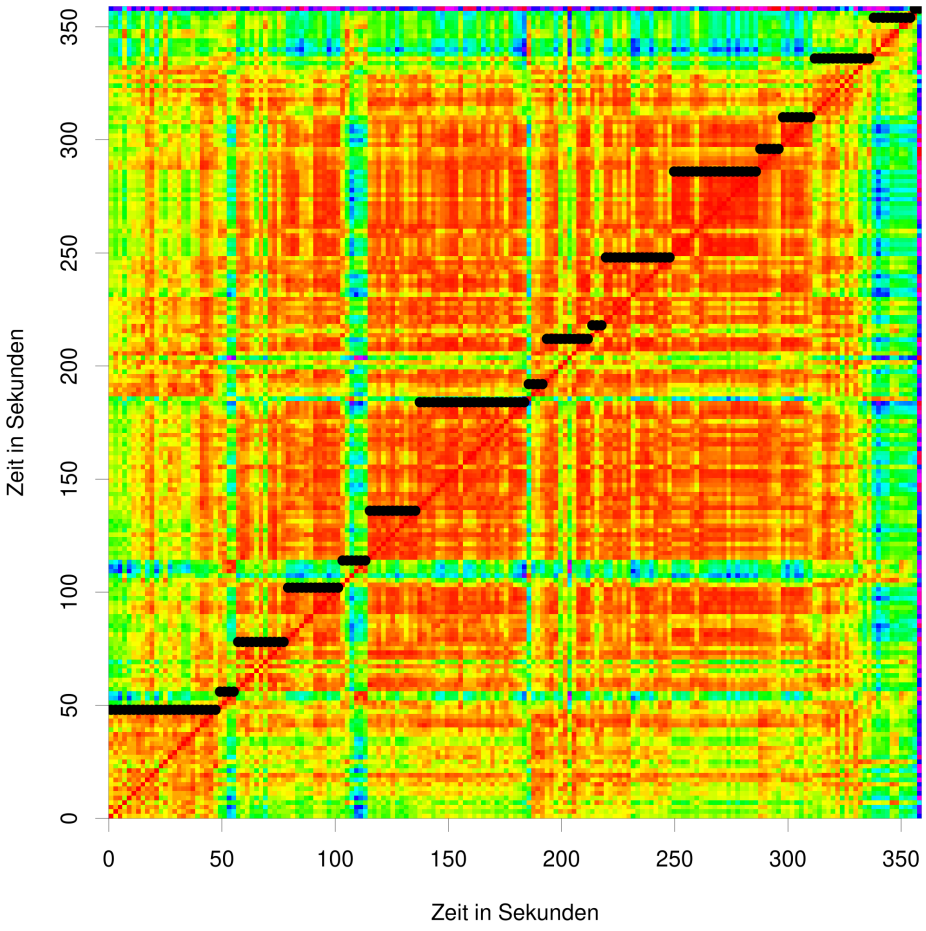

Abbildung 9.9: Musikalische Struktur von Queens „Bohemian Rhapsody“ mit der optimierten OCKC Implementierung

Die hier vorgestellte effiziente Implementierung von OCKC ist als Paket `ockc` für die Programmiersprache `R` im Paketrepository `CRAN` veröffentlicht (Krey, Leisch und Hoffmeister, 2016).

10 Clusterung von Energienetzen basierend auf der Netzwerktopologie

Die Kapitel 10.1 bis 10.5 basieren auf den in Krey, Brato u. a. (2015) präsentierten Ergebnissen zum Spectral Clustering von Energienetzen.

10.1 Problemstellung

Die unterbrechungsfreie Versorgung mit Elektrizität ist für unsere Wirtschaft und Gesellschaft lebensnotwendig. Das europäische Elektrizitätsnetz ist das weltweit größte frequenzsynchronisierte Elektrizitätsnetz. Die Ausdehnung reicht von Dänemark bis Nordafrika und von Portugal bis in die Türkei. Der Betrieb eines solchen Netzes ist extrem komplex und stellt ständig neue Herausforderungen an die Betreiber. In den letzten Jahren hat die zunehmende Dezentralisierung der Energieerzeugung durch Abschaltung oder Verdrängung von Großkraftwerken (Atomkraft oder Kohle) durch kleinere, über den gesamten Kontinent verteilte, Erzeuger erneuerbarer Energien wie Windkraft und Photovoltaik eine starke Überarbeitung des gesamten Netzmanagements notwendig gemacht. Durch den Ersatz von an wenigen Standorten aufgestellten Großgeneratoren durch eine Vielzahl von kleinen Generatoren, ist eine Top-Down Netzsteuerung ausgehend von den große Kraftwerken auf den höchsten Spannungsebenen nicht mehr ausreichend. Angefangen von der Leistungsflussregelung über die Stabilisierung von Spannung und Frequenz bis hin zu den Notfallplänen für Störungen sind viel detailiertere Modelle notwendig, um alle relevanten Einflüsse berücksichtigen zu können.

Ein wichtiger Aspekt der Notfallpläne für den Netzbetrieb sind mögliche Segmentierungen des gesamten Netzes in Inseln, die für sich alleine funktionsfähig sind, um im Falle einer Großstörung die Versorgung möglichst großer Netzbereiche aufrecht zu erhalten. Durch die wetterabhängige Dynamik der Energieerzeugung von Windenergie und Photovoltaik sind statische Notfallpläne nicht mehr zeitgemäß und müssen ständig an sich veränderte Last- und Einspeisesituationen angepasst werden. Gleichzeitig muss die Netztopologie und regulatorische Anforderungen, insbesondere im Bezug auf die Netzredundanz berücksichtigt werden. Hierzu eignen sich Clusterverfahren, die die Nachbarschaftsstruktur in einem Graphen berücksichtigen können, die also auf dem Prinzip des *Spectral Clustering* basieren.

Durch Simulation verschiedener Ausfallszenarien und Berechnung der sich dadurch ergebenden möglichen Netzsegmentierungen ist es ferner möglich für den Betrieb notwendige Steuer- und Rechenzentren optimal zu platzieren. Dies ist im Notfall wichtig, um die Kontrolle über alle Netzsegmente zu behalten.

10.2 Grundlagen Energienetze

Durch die Frequenzsynchronität des kontinentaleuropäischen Elektrizitätsnetzes schwingt die Wechselspannung an allen Orten in diesem Netz synchron mit $50\,Hz$. Die Erhaltung der Synchronität bei exakt dieser Frequenz ist eine hochkomplexe Aufgabe, aber gleichzeitig für einen störungsfreien Betrieb essentiell. Im normalen Betrieb darf die Frequenz nicht stärker als $0,2\,Hz$ vom Sollwert abweichen, kurzfristig ist bei Störungen eine maximale Abweichung von $0,8\,Hz$ erlaubt. Ist zu viel elektrische Energie im Netz steigt die Frequenz, bei einem Mangel sinkt sie ab.

Für die Einhaltung der Frequenzvorgabe von $50\,Hz$ sind die Transportnetzbetreiber verantwortlich. Sie betreiben die höchsten Spannungsebenen von $220\,kV$ und $380\,kV$ des hierarchisch organisierten Netzbetriebs und sind für den Langstreckentransport der Energie durch das Netz verantwortlich. Großkraftwerke sind auf kürzest möglichem Weg an das Transportnetz angeschlossen und haben so einen direkten Einfluss auf die Netzfrequenz. Diese Architektur hat traditionell für eine gute Kontrolle über das Netz gesorgt.

Auf den darunter liegenden Spannungsebenen (Hoch- und Mittelspannung mit $60\,kV$ und $110\,kV$ bzw. $3 - 30\,kV$) kümmern sich die Verteilnetzbetreiber um die regionale Verteilung der Elektrizität. Kleinere Kraftwerke werden meist auf diesen Spannungsebenen angeschlossen. Die Verteilnetzbetreiber übergeben dann, je nach Größe des Abnehmers, auf unterschiedlichen Spannungsebenen die Elektrizität an die kommunalen Netzbetreiber. Diese stellen den Endverbrauchern dann die gewohnten Niederspannungsanschlüsse mit $230\,V$ und $400\,V$ zur Verfügung.

Der Anschluss der dezentral erzeugten eneuerbaren Energien, wie Photovoltaik auf den Hausdächern der Endverbraucher und einzelne Windkraftanlagen, erfolgt auf den untersten Spannungsebenen. Kleinere Windparks und kleinere von Biogasanlagen betriebene Kraftwerke werden auf Mittelspannungsebene an das Elektrizitätsnetz angeschlossen. Diese Anlagen sind damit weit entfernt von der Kontrolle der Transportnetzbetreiber, haben aber mit dem zunehmenden Ausbau einen immer größeren Einfluss auf die Netzfrequenz. Die engmaschigen niedrigen Spannungsebenen können

nicht klassisch zentral gesteuert werden. Hier ist es notwendig zunehmend automatisierte Verfahren zu verwenden, indem auf Basis von Messwerten des aktuellen Netzzustands algorithmengestützt Schutz- und Kontrollsysteme passend parametrisiert werden. Nur so können diese Werte auch bei Änderungen des Betriebszustands schnell aktualisiert werden, um die Kontrolle über das Gesamtnetz sicherzustellen.

10.3 Datenerzeugung

Da es nur wenige frei verfügbare Daten von großräumigen Energienetzen gibt, besteht die Datenbasis der vorliegenden Arbeit aus Ergebnissen der mit der kommerziellen Energienetzsimulationssoftware *DIgSilent PowerFactory* durchgeführten Simulationen. DIgSilent PowerFactory ermöglicht neben statischen Lastflussberechnungen auch eine Simulation des dynamischen Verhaltens eines Wechselspannungenergienetzes mit Zeitauflösungen bis in den Millisekundenbereich. Die hierbei genutzten Modelle sind das *New England Test System* (NETS) (Fink und Trygar, 1979; Pai, 1994; Rogers, 2000) und dessen Erweiterung *New York Power System* (NYPS) (Rogers, 2000). Diese Modelle sind den Hoch- und Höchstspannungsnetzen der Region nachempfunden, basierend auf dem zum Zeitpunkt der Veröffentlichungen aktuellen Stand des Netzausbaus und sind eine weitverbreitete Grundlage für die Entwicklung und das Testen von Regelungs- und Steuerungskonzepten für Hoch- und Höchstspannungstransportnetze. Die Verwendung einer Simulationssoftware hat den Vorteil auch Daten von Betriebszuständen generieren zu können, die im realen Betrieb nie auftreten sollten. Insbesondere extreme Last- und Fehlersituationen, die die Aufrechterhaltung des Betriebs gefährden, können so untersucht werden.

Das *New England Test System* (auch *New England IEEE 39-Bus System*) ist ein etabliertes Referenzsystem für die Entwicklung und Evaluierung von Algorithmen und Konzepten zur Steuerung von Elektrizitätsnetzen. Das NETS ist eine Modell des Hoch- und Höchstspannungnetzes des US Bundesstaats New England in den 1970er Jahren. Es besteht aus 39 Sammelschienen, die 10 Generatoren und 18 Lasten an das Netz anschließen. Untereinander sind die Sammelschienen mit 37 Hochspannungsleitungen vernetzt. Eine schematische Darstellung des Netzes ist in Abbildung 10.1 zu sehen. Wir betrachten hierbei die Sammelschienen als Knoten eines Graphen und die Hochspannungsleitungen als Kanten. Für jeden Knoten und jede Kante liegen eine Vielzahl von energietechnischen Parametern und Messwerten vor. In der vorliegenden Arbeit sind vorrangig die Leistungen von Generatoren und Lasten, der Phasenwinkel an jeder Sammelschiene sowie die Admittanz der Leitungen, welche zur Konstruktion eines Distanzmaßes verwendet wird, von Interesse.

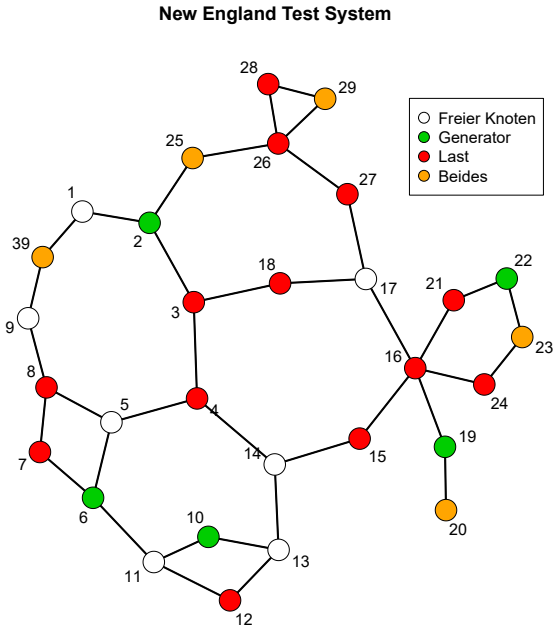

Abbildung 10.1: Das New England Test System (NETS)

Das zweite hier genutzte Test System ist das *New York Power System* (auch *New England 68-Bus Test System*). Es ist eine Erweiterung des NETS Systems. Es ergänzt das NETS um 6 zusätzliche Generatoren und 17 Lasten. Es ergibt sich somit ein Netz mit insgesamt 53 Knoten, die über 68 Leitungen verbunden sind, welches in Abbildung 10.2 schematisch dargestellt ist.

10.4 Clusterung des Netzwerkgraphen

Ein so weiträumiges Elektrizitätsnetz wie das kontinentaleuropäische Netz der ENTSO-E (European Network of Transmission System Operators for Electricity) kann nicht zentral von einer Stelle aus gesteuert werden. Schon auf den in Relation zu den Verteil- und Niederspannungsnetzen sehr weitmaschigen höchsten Spannungsebenen der Transportnetze weist das Netz über 10000 Knoten auf. Ein Zusammenführen von Messdaten aller Knoten an einem Ort würde extrem viel Zeit, Datentransferkapazität und die Auswertung eine sehr hohe Rechenleistung erfordern. Dies steht in extremen Widerspruch zu den benötigten Reaktionszeiten für einen sicheren Betrieb. Hierfür müssen Notfallsysteme zum Teil innerhalb von Bruchteilen von Sekunden in den Betrieb eingreifen, um teure Schäden zu vermeiden. Es ist daher notwendig die Steuerung dezentral zu organisieren. Dabei ist zu beachten, dass bei einer Notsituation, die einen Zerfall des Netzes in einzelne Inseln zur Folge hat, die Kontrolle

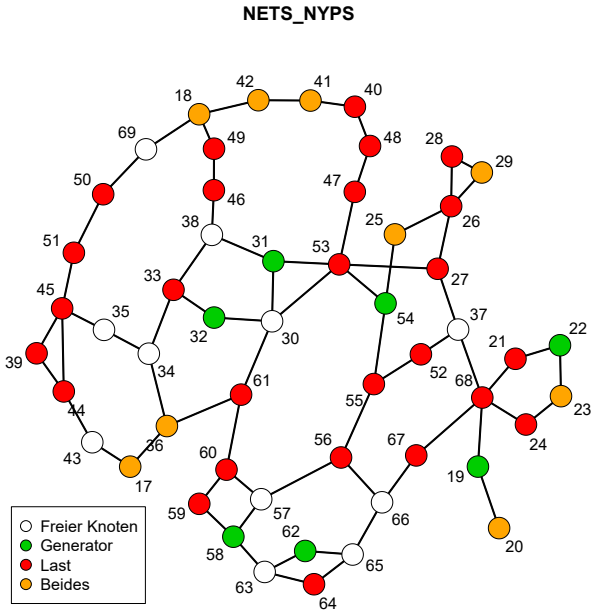

Abbildung 10.2: Das New York Power System (NYPS)

über möglichst große Bereiche des Netzes (im Idealfall des gesamten Netzes) erhalten bleibt. Während solche Planungen auf den höchsten Spannungsebenen prinzipiell manuell machbar sind, müssen sie auf den sehr viel enger vermaschten niedrigeren Spannungsebenen automatisiert werden. Insbesondere die Dynamik des Netzes, die je nach Einspeise- und Lastsituation zu unterschiedlichen Möglichkeiten der Inselnetzbildung führen können, macht eine algorithmengestützte Planung notwendig. Die hier vorgestellte Methode zur Clusterung des Netzwerkgraphens ist eine Möglichkeit um relevante Segmentierungen des Netzes zu finden und so die Entscheidungsfindung zu unterstützen.

Werden bei der Suche nach Strukturen bzw. ähnlichen Knoten nur elektrotechnische Kriterien zugrunde gelegt, können zwar Knoten identifiziert werden, die vergleichbare Lastprofile oder ein ähnliches dynamisches Verhalten aufweisen, diese Information alleine ist aber nicht ausreichend um das Netz in sinnvolle Segemente zu partitionieren (Krey, Brato u. a., 2015). Es ist notwendig die Nachbarschaftsstruktur in den Prozess einzubeziehen. Hierzu betrachtet man das Elektrizitätsnetz als ungerichteten Graphen G. Die Sammelschienen bilden die Knoten V des Graphens und die Stromleitungen die Kanten E. Diese Struktur kann man in einer Nachbarschaftsmatrix abbilden. Der Eintrag in der i-ten Zeile und j-ten Spalte beschreibt dann, ob die Knoten i und j eine direkte Verbindung über eine Stromleitung haben (Wert 1) oder nicht (Wert 0). Diese einfachste Form der Nachbarschaftsmatrix berücksichtigt noch

nicht die Distanz zwischen den Knoten, also die Länge der verbindenden Stromleitung.

Die elektrische Distanz zweier Knoten in einem Elektrizitätsnetz ist zusätzlich zur Länge der Leitung auch von deren elektrischen Eigenschaften abhängig. Bei einem Gleichspannungsnetz ist dies insbesondere der elektrische Widerstand. Die entsprechende Größe bei Wechselspannung ist die komplexwertige *Impedanz* z. Im Folgenden wird immer der Betrag dieser Größe betrachtet. Die jeweiligen Werte steigen mit der Länge der Leitung und der Abnahme des Leiterquerschnitts und damit der Leitungskapazität. Die Impedanz ist eine Größe, die die wichtigste elektrische Eigenschaft und die Länge der Leitung vereint und ist damit ein geeignetes Distanzmaß.

Zur Konstruktion der Nachbarschaftsmatrix für die algorithmische Analyse eines Graphen mit Methoden des *Spectral Clustering* benötigt man ein Ähnlichkeitsmaß. Analog zur Konstruktion eines Distanzmaßes aus einem Ähnlichkeitsmaß wie in Kapitel 4, könnte man auch den umgekehrten Weg gehen. Da für den Betrag der Impedanz $|z|$ gilt $|z| \geq 0$, kann auch durch einfache Kehrwertbildung eine Umwandlung vom Distanzmaß $|z|$ zu einem Ähnlichkeitsmaß erfolgen:

$$a = \frac{1}{|z|}$$

Diese Größe heißt *Admittanz*. Bildet man die Einträge einer Matrix A auf folgende Art

$$A_{ij} = \begin{cases} -a_{ij} & i \neq j \\ \sum_{k=1}^{n} a_{ik} & i = j \end{cases},$$

wobei a_{ij} die Admittanz aller Leitungen zwischen zwei Knoten i und j des Graphens und n die Anzahl der Knoten des Graphens ist, so erhält man die Knotenadmittanzmatrix A. Die Knotenadmittanzmatrix ist für Lastflussberechnungen in einem Elektrizitätsnetz notwendig und daher eine wichtige Größe für die Elektrotechnik. Die Daten zur Aufstellung der Knotenadmittanzmatrix liegen somit immer vor. Zusätzlich ist es eine im Normalbetrieb statische Größe, die nur durch Baumaßnahmen (Inbetriebnahme neuer Leitungen, Hinzufügen neuer Generatoren oder Lasten) oder aktive Schalthandlungen (Trennung von Knoten durch Abschaltung von Leitungen) verändert wird. Gibt es keine direkte Verbindung zwischen zwei Knoten k und l, so ist der entsprechende Matrixeintrag $a_{kl} = 0$. Die so erhaltene Nachbarschaftsmatrix ist singulär und damit nicht invertierbar. Durch vergrößern der Diagonalelemente um den Faktor $(1 + \varepsilon)$, mit einem geeigneten $\varepsilon > 0$ (hier wurde $\varepsilon = 0.1$ verwendet) erhält

man die strikt diagonal dominante und damit invertierbare Matrix A' mit

$$A'_{ij} = \begin{cases} A_{ij} & i \neq j \\ (1+\varepsilon) \cdot A_{ij} & i = j \end{cases}.$$

Eine Moore Penrose Inverse einer Nachbarschaftsmatrix kann als Kernel Matrix interpretiert werden (Saerens u. a., 2004). Dementsprechend gilt das auch für eine echte Inverse der Matrix A'.

Man bekommt daher die für das Spectral Clustering notwendige Kernel Matrix durch Invertieren von A':

$$S = A'^{-1}. \tag{10.1}$$

10.5 Spectral Clustering

Spectral Clustering ist ein Sammelbegriff für verschiedene Clusterverfahren, die auf der Eigenwert Struktur einer (Kernel-) Matrix basieren. Mit Spectral Clustering kann man einen ungerichteten Graphen unter Berücksichtigung seiner Nachbarschaftsstruktur clustern. Hier wurde der Algorithmus von Ng, Jordan und Weiss (Ng, Jordan und Weiss, 2002) verwendet.

Algorithmus

Der erste Schritt des Spectral Clustering Algorithmus ist die Normalisierung der in Formel 10.1 erhaltenen Kernel Matrix S mit einer Diagonalmatrix D, deren Einträge die Zeilensummen von S sind, also $D_{ii} = \sum_{j=1}^{n} S_{ij}$:

$$L = D^{-\frac{1}{2}} S D^{-\frac{1}{2}}.$$

Von der so erhaltenen Matrix L werden die Eigenwerte und -vektoren bestimmt. Bezeichne mit $L^E_{(1)}$ den größten Eigenwert und mit $L^V_{(1)}$ den zugehörigen Eigenvektor. Analog bezeichnet $L^E_{(i)}$ den Eigenwert an der i-ten Stelle der Ordnungsstatistik mit seinem zugehörigen Eigenvektor $L^V_{(i)}$.

Sei K nun die Anzahl der zu bildenden Cluster. Dann wählt man die zu den K größten Eigenwerten gehörenden Eigenvektoren und konstruiert die Spalten der Matrix E durch

$$E = \begin{pmatrix} L^V_{(1)} & L^V_{(2)} & \cdots & L^V_{(K)} \end{pmatrix}.$$

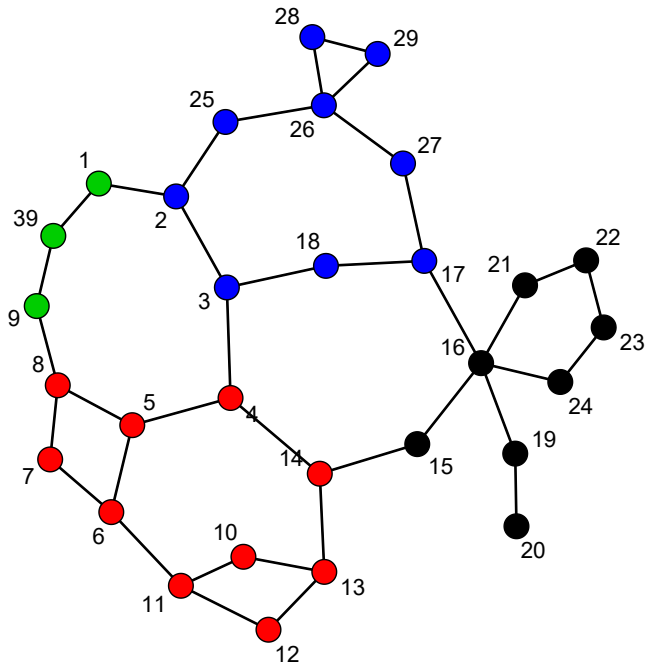

Abbildung 10.3: Spectral Clustering des New England Test System

Die Anzahl der Zeilen der Matrix E ist immer noch gleich n, der Anzahl der Knoten im Netzwerkgraphen. Die Anzahl der Spalten hat sich aber auf K reduziert.

Anschließend werden die Zeilen von E normalisiert:

$$E^s_{ij} = \frac{E_{ij}}{\left(\sum_{k=1}^{K} E^2_{ik}\right)^{\frac{1}{2}}}.$$

Der letzte Schritt des Verfahrens ist das Clustern der Zeilen von E^s mit dem k-Means Algorithmus.

Die mit diesem Verfahren erhaltene Clusterung des NETS Systems in $K = 4$ cluster ist in Abbildung 10.3 zu sehen. Diese Clusteranzahl wurden auf Basis der in Kaptitel 7 im Abschnitt 7.4.2 beschriebenen Methodik bestimmt.

In Abbildung 10.4 ist die Clusterung der NETS Erweiterung NYPS in $K = 5$ Cluster gezeigt.

Eine Schwäche iterativer Cluster Algorithmen wie k-Means ist das Risiko nicht das globale Minimum des Optimierungsproblems zu finden. Durch wiederholtes Durchführen des Verfahrens mit unterschiedlichen initialen Clusterzentren als Startwerten, kann dieses Risiko reduziert werden. Hier sind 20 Läufe verwendet worden. Die

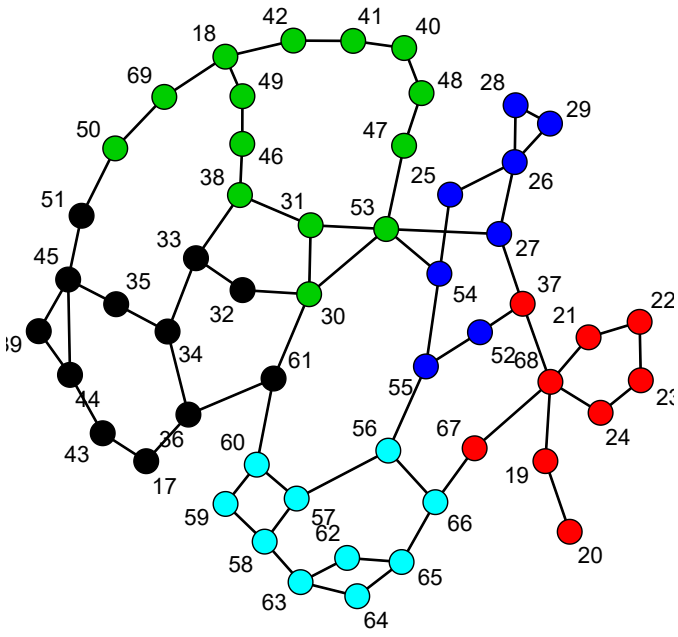

Abbildung 10.4: Spectral Clustering des New York Power System

Resultate aller Läufe sind gespeichert worden, um wie im nächsten Abschnitt beschrieben noch ein anderes Kriterium für die Auswahl der besten Clusterung verwenden zu können.

10.6 Clusterung für Schutzsysteme

Wie in der Beschreibung des Spectral Clustering Algorithmus (siehe Abschnitt 10.5) beschrieben, können die durch mehrere Starts des k-Means Verfahrens erhaltenen evtl. unterschiedlichen Clusterungen zusätzlich zum primären kleinste Quadrate Kriterium auch noch nach einem sekundären Kriterium angeordnet werden. Ein solches Kriterium kann die Energiebilanz innnerhalb eines Clusters sein. Falls es in einem Störungsfall zu einer Aufteilung des Elektrizitätsnetzes in einzelne Inseln kommt, kann nur in den Inseln die Stromversorgung aufrecht erhalten werden, in denen für die Beziehung zwischen der Erzeugungsleistung der Kraftwerke P_G und dem Leistungsbedarf der Verbraucher P_L gilt

$$P_G \geq P_L.$$

Es ist daher erstrebenswert, dass die Differenz zwischen Erzeugungsleistung und Verbrauch möglichst gering ist. Dies verringert das Risiko eines großflächigen Blackouts wie im November 2006 (ENTSOE, 2007). Bei diesem Ereignis wurde das kontinentaleuropäische Energienetz in mehrere Inseln aufgeteilt. Durch starke Windkrafteinspeisung im Nordosten des Netzes und einer Leistungsreduktion konventioneller Kraftwerke zur Kompensation insb. im Süden und Westen kam es zu einem starken Leistungsfluss quer durch das Netz, der durch einen Störfall zur Notabschaltung mehrerer Höchstpannungsleitungen im Transportnetz führte. Bei der dann erfolgten Bildung von 3 Inselnetzen hatten die beiden Inseln im Südosten und Westen Europas deutlich zu wenig Erzeugungsleistung, weshalb es zu einem großflächigen Stromausfall kam.

Besteht die Möglichkeit Clusterungen für eine evtl. notwendige Inselnetzbildung automatisiert zu erstellen, so kann man diese regelmäßig abhängig von der aktuellen Last- und Einspeisesituation basierend auf den von den Netzbetreibern durchgeführten Lastflussberechnungen (mindestens alle 15 Minuten) aktualisieren. Bei einer automatisierten Berechnung ist es zusätzlich möglich die Inselnetzbildung mit einer höheren regionalen Auflösung durchzuführen und so evtl. notwendige Lastabwürfe zur Stabilisierung des Netzes, die zu Stromausfällen bei den Verbrauchern führen, regional zu begrenzen. Da für die Berechnungen nur die Ergebnisse einer statischen Lastflussberechnung notwendig sind, können die Clusterungen für verschiedene „typische“ Szenarien vorausberechnet und für einen späteren schnellen Zugriff gespeichert werden.

Der hier vorgestellte Lösungsansatz basiert auf den durch unterschiedliche Starts erhaltenen Clusterungen des Netzes. Jede dieser Lösungen stellt ein lokales Minimum des Optimierungsproblems dar und wird anschließend auf die Leistungsbilanz innerhalb der Cluster untersucht. Hierzu wird die Summe der quadrierten Differenzen zwischen Erzeugungsleistung und Last innerhalb der Cluster gebildet. Die Clusterung mit der geringsten Summe von quadrierten Differenzen der Leistungen ist der optimale Kandidat bzgl. dieses Kriteriums

In Abbildung 10.5 und Abbildung 10.6 sind die Clusterungen mit der am besten ausgeglichenen Bilanz zwischen Einspeiseleistung und Last zu sehen. Die Clusterungen berücksichtigen immer noch die Nachbarschaftsstruktur, sind aber im Vergleich zu den ursprünglichen Ergebnissen des Spectral Clusterings etwas verschoben.

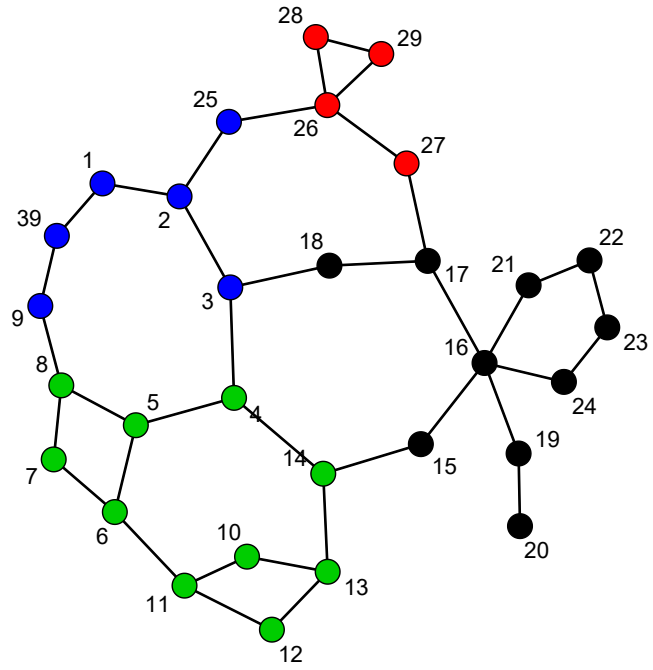

Abbildung 10.5: Das New England Test System mit leistungsbalancierten Clustern

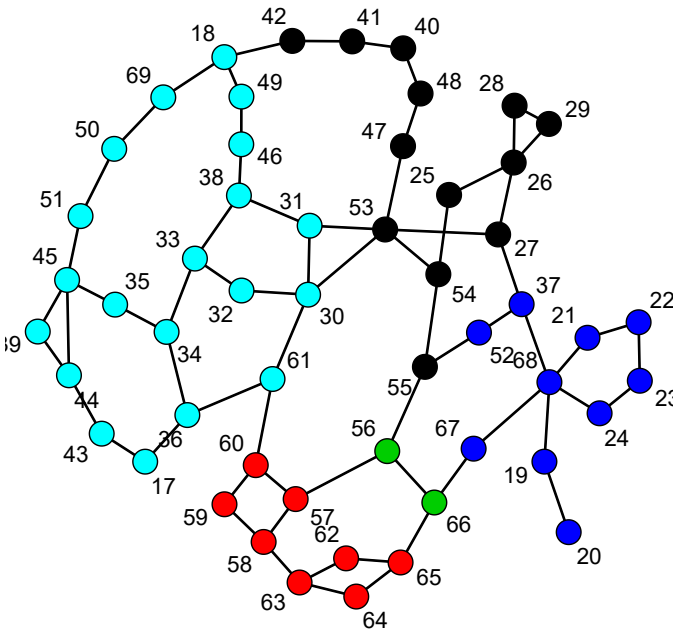

Abbildung 10.6: Das New York Power System mit leistungsbalancierten Clustern

10.7 Vergleich der Clusterungen

Um die durch das Clusterverfahren automatisiert erhaltenen Partitionierungen des Netzwerkgraphen qualitativ einordnen zu können, sollen hier zwei wichtige Vergleiche gemacht werden.

In Abbildung 10.7 ist die manuelle Clusterung des NETS in Regionen zu sehen, die von Energietechnikexperten durchgeführt wurde und in der Literatur zu finden ist (*Power Systems Engineering Research Center: Transfer Capability Calculator* 2001). Die automatisiert durchgeführte Aufteilung des Netzes in Regionen mit Spectral Clustering und algorithmischer Clusterzahlbestimmung stimmt in großen Teilen mit der Einteilung durch den Experten überein. Die Knoten 28 und 29 können von Spectral Clustering nicht dem Cluster unten rechts zugeordnet werden, da eine der grundlegenden Anforderungen war, dass Knoten eines Cluster zwingend eine direkte Leitungstrasse als Verbindung haben müssen. Daher musste Spectral Clustering diese beiden Knoten dem oberen Cluster zuordnen. Interessanterweise hat das Spectral Clustering für die Knoten 1, 9 und 39 einen eigenen Cluster vorgesehen. Diese 3 Knoten sind Ersatzschaltungen für die gesamte Erweiterung des NETS zum NYPS. Es ist daher sinnvoll, dass diese drei Knoten einem eigenen Cluster zugeordnet wurden, da sie sich deutlich von den anderen Knoten, die einfache Sammelschienen zur Zusammenführung einzelner Lasten und Generatoren sind, unterscheiden. Misst man die Übereinstimmung dieser Ergebnisse mit dem adjustierten Rand-Index, so erhält man einen Wert von $0,67$.

Ein zweiter wichtiger Vergleich ist, wie sich die Clusterung unter typischen Veränderungen in den Betriebsbedingungen verhält. Eine wichtige Anforderung für die praktische Einsetzbarkeit eines solch automatisierten Verfahrens ist, dass normale Änderungen im Betrieb nur zu kleinen Veränderungen in der Clusterung führen dürfen. Die Abschaltung einer Leitungstrasse (z.B. für Wartungsarbeiten) ist solch eine typische Netzänderung, die aufgrund der geforderten Redundanz des Netzes ($n-1$ Stabilität (Wadhwa, 2005)) unkritisch ist und zu keinen großen Änderungen in den Betriebsbedingungen führt.

In Abbildung 10.8 ist die Clusterung zu sehen, wenn die Stromtrasse zwischen Knoten 1 und 2 abgeschaltet wird. Die einzige Änderung im Vergleich zur Clusterung des vollen Graphens (siehe Abbildung 10.3) ist die geänderte Zuordnung von Knoten 17. Damit ergibt sich ein sehr hoher adjustierter Rand-Index von $0,9$. Dies belegt zusätzlich die Stabilität der Ergebnisse. Die Wahl von 4 Clustern als optimale Clusteranzahl für dieses Netz wird damit bestätigt.

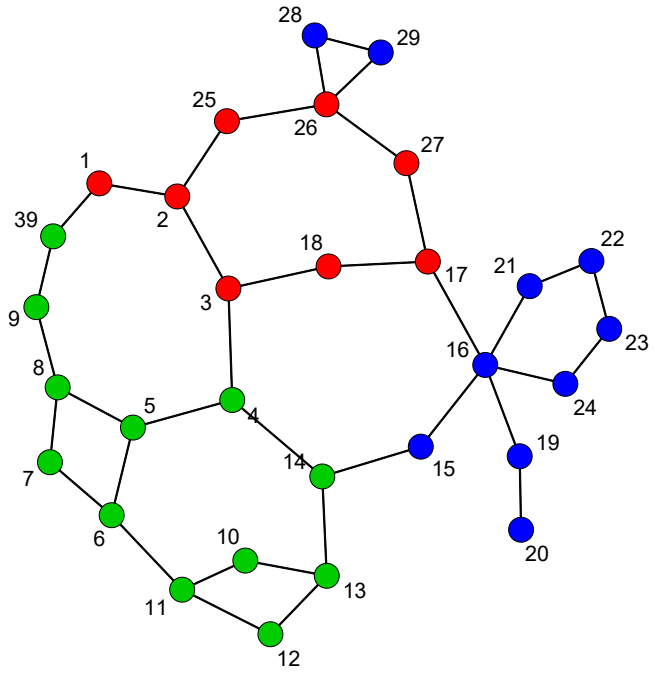

Abbildung 10.7: Manuelle Clusterung des New England Test System durch einen Energietechniker

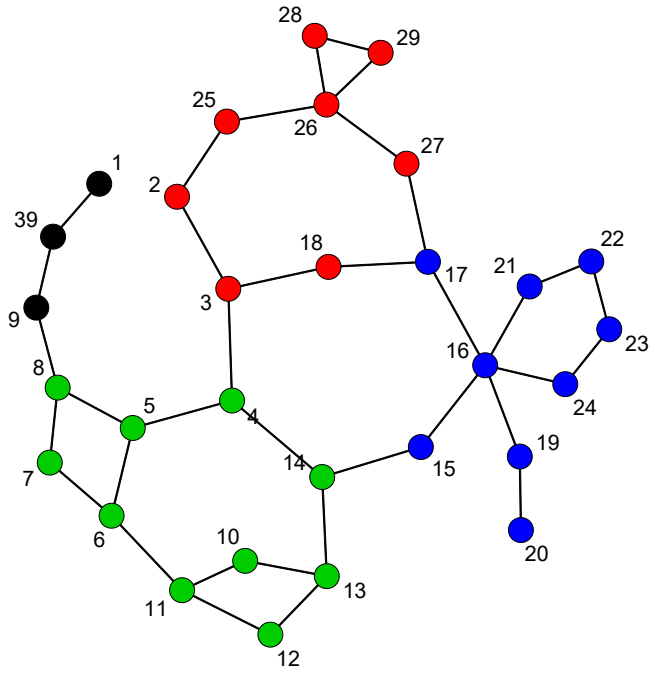

Abbildung 10.8: Spectral Clustering des New England Test Systems mit abgeschalteter Stromleitung zwischen Knoten 1 und 2

Wie in Abschnitt 10.6 beschrieben, kann man die auf Basis der Netzwerktopologie erhaltenen Clusterungen auch nach sekundären Kriterien bewerten. Während die Clusterung streng nach Netzwerktopolgie, durch die offline Berechenbarkeit, hilfreich für die optimale verteilte Platzierung von Steuerungs- und Kontrollsystemen ist, kann die Bewertung der Energiebilanz innerhalb der Cluster ein nützliches Werkzeug zur Parametrisierung von Schutzsystemen sein. Diese können dann in ihre Entscheidungen, wie ein Netz im Störungsfall in Inseln aufgeteilt wird, Informationen zu den Leistungsflüssen einbeziehen. So kann sichergestellt werden, dass möglichst viele dieser Inseln autark den Betrieb des Netzes ohne Lastabwürfe, die zu Stromausfällen auf der Verbraucherseite führen, aufrecht erhalten können.

11 Zusammenfassung

Clusterverfahren sind ein wichtiges Werkzeug des unüberwachten maschinellen Lernens. Sie ermöglichen eine automatisierte Strukturierung von großen Datenmengen und können so ein wichtiges Werkzeug zur weiteren Datenverarbeitung bzw. -analyse sein oder das Fundament für Entscheidungen bilden.

Die in dieser Arbeit betrachteten Anwendungsbeispiele aus der Musiksignalanalyse sowie der Elektrotechnik zeigen, dass reine distanzbasierte Clusterverfahren nicht immer ausreichend sind und Nebenbedingungen in die zugrundeliegenden Optimierungsprobleme eingefügt werden müssen, um sinnvolle Clusterungen zu erhalten, die für den Anwender hilfreich sind.

Für die Berücksichtigung der durch einen Graphen abgebildeten Nachbarschaftsstruktur eines Energienetzes gibt es mit den Methoden des Spectral Clustering eine Reihe von Verfahren, deren Berechnungsschritte aus klassischen Komponenten der Numerik bestehen. Für diese Komponenten stehen hochoptimierte Bibliotheken bereit, die für eine effiziente Implementierung des Spectral Clusterings genutzt werden können. Wendet man Spectral Clustering auf die in dieser Arbeit betrachteten Referenznetze an, so erhält man sinnvolle Unterteilungen der Netze in Regionen, die eine wichtige Grundlage für die Planung von Schutzsystemen zur Sicherstellung eines störungsfreien Betriebs bilden können. Insbesondere die in Kapitel 10.6 eingeführte zusätzliche Bedingung der möglichst ausgeglichenen Energiebilanz zwischen Erzeugung und Verbrauch ist hilfreich um großflächige Stromausfälle bei einer Betriebsstörung im Höchstspannungsnetz zu vermeiden.

Bei der für die Anwendung in der Musiksignalanalyse nützlichen Nebenbedingung einer zeitlichen Ordnung ist die Methodenauswahl weniger einfach und naheliegend. Die zeitliche Ordnung lässt sich weder als Nachbarschaftsstruktur noch als Gruppenrestriktion formulieren. Man ist eher im Bereich einer multivariaten Zeitreihensegmentierung. Der betrachtete generische Ansatz ist zwar flexibel, hatte aber Schwierigkeiten die relevanten Veränderungen im Klang zu identifizieren, zusätzlich ist keine klare Definition eines Kriteriums zum Beenden der Segmentierung möglich. Die von Steinley und Hubert (2008) eingeführten *Order Constrained Solutions in k-Means Clustering (OCKC)* haben sich als sehr leistungsfähiges Werkzeug erwiesen. Mit dieser Methode konnte die Clusterung der Klangmerkmale bei der Erkennung von Instrumentenklangfarben so verbessert werden, dass die Fehlklassifikationsrate der eingesetzen überwachten maschinellen Lernverfahren in Kapitel 9.4.2 im Vergleich

zu den guten Ergebnissen aus Kapitel 8.4 noch einmal verbessert werden konnten. Insbesondere die SVM mit RBF Kernfunktion konnte mit 17% im Vergleich zu 36% Fehklassifikationsrate ihre Leistung deutlich verbessern. Auch der Random Forest konnte mit 19% gegenüber zuvor 28% eine deutliche Verbesserung erreichen, kommt aber nicht ganz an die SVM mit RBF Kern heran.

Leider ist das Optimierungsproblem von OCKC sehr aufwändig und es hat keine effiziente Implementierung vorgelegen, die eine Verwendung mit $k > 6$ ermöglicht hätte. In Kapitel 9.5 und insbesondere in Kapitel 9.6 ist dann ein Algorithmus entwickelt und in `R` implementiert worden, der sowohl die Berechnung der kumulierten Distanzen als auch die eigentliche Clusterung effizient durchführt. Für die Berechnung der kumulierten Distanzen ist das Problem als eine Reihe von Vektor-/Matrixoperationen formuliert worden, die sich mit hochoptimierten *Basic Linear Algebra Subprograms (BLAS)*) Bibliotheken effizient berechnen lassen. Gleichzeitig ist die Formulierung so gewählt, dass eine Parallelisierung der Berechnung möglich ist. Für die Clusterung ist auf das Konzept der *dynamischen Programmierung* (Bellman, 1957, 1972) zurückgegriffen worden, um dieses diskrete, kombinatorische Optimierungsproblem durch einen iterativen Algorithmus effizient zu lösen. Dies verbessert das Laufzeitverhalten für das Lösen des Optimierungsproblems von exponentieller zu quadratischer Komplexität. Die Implementierung des Verfahrens ist als Paket `ockc` auf `CRAN` veröffentlicht (Krey, Leisch und Hoffmeister, 2016).

Eine gemeinsame Herausforderung aller Clusterverfahren ist die Festlegung der Anzahl der Cluster. In der Literatur findet sich eine Vielzahl von Kriterien zur Wahl dieses Parameters (Halkidi, Batistakis und Vazirgiannis, 2001), wobei sich die Mehrzahl dieser Kriterien nur in engen Grenzen, für die sie entwickelt wurden, einsetzen lassen. Die Betrachtung der mit dem *Rand-Index* gemessenen Clusterstabilität auf Bootstrap-Stichproben der Daten (Dolnicar und Leisch, 2010) ist dabei der am besten universell einsetzbare Ansatz. Der an diesem Ansatz geäußerten Kritik, dass oft zu niedrige Clusteranzahlen empfohlen werden (Steinley und Brusco, 2011), wird hier mit der zusätzlichen Betrachtung der Streuung des Rand-Index begegnet. Durch Einbeziehung der Streuung und Betrachtung des *Quartilsdispersionskoeffizienten* als robuste Alternative zum Variationskoeffizienten konnte in den betrachteten Anwendungen zur *Musical Structure Analysis* (Kapitel 9.5.1) und zur Strukturierung von Energienetzen (Kapitel 10.7) automatisiert eine Clusteranzahl bestimmt werden, die eine gute Interpretation der Ergebnisse ermöglicht.

Da es sich bei den hier betrachteten Clusterverfahren um Methoden mit Nebenbedingungen handelt, kann eine klassische Bootstrap-Stichprobe nicht für die Beurteilung

der Clusterstabilität verwendet werden, da diese unter Umständen die Nebenbedindung verletzt. Es sind daher in den Kapiteln 7.4.1 und 7.4.2 Alternativen präsentiert worden, die sowohl unter Ordnungsrestriktion, als auch bei einer Nachbarschaftsbedingung die Einhaltung der Nebenbedingung in den generierten Datensätzen sicherstellen. Mit diesen Methoden ist auch bei den Clusterverfahren mit Nebenbedingungen eine Beurteilung der Clusterstabilität mit dem Rand-Index möglich.

Es sind in dieser Arbeit für zwei Anwendungsfälle mit unterschiedlichen Nebenbedingungen geeignete Clusterverfahren vorgestellt worden. Für die Clusterung unter Ordnungsrestriktion ist zusätzlich eine effiziente Implementierung entwickelt und so bereitgestellt worden, dass sie dem Anwender zur Verfügung steht. Um eine vollständige Automatisierung der Clusterung zu ermöglichen ist ferner eine Methode zur Clusteranzahlbestimmung so weiterentwickelt worden, dass sie zum einen durch Hinzufügen der Betrachtung der Streuung des Qualitätsmaßes praktisch relevantere Ergebnisse liefert und zum anderen auch bei Clusterverfahren unter Nebenbedingungen einsetzbar ist.

Referenzen

Eigene Arbeiten

Krey, Sebastian (2008). *SVM basierte Klangklassifikation*. TU Dortmund. Diplomarbeit, Fakultät Statistik.

Krey, Sebastian, Sebastian Brato, Uwe Ligges, Jürgen Götze und Claus Weihs (2015). „Clustering of electrical transmission systems based on network topology and stability“. In: *Journal of Statistical Computation and Simulation* 85.1, S. 47–61. DOI: 10.1080/00949655.2014.924517.

Krey, Sebastian, Friedrich Leisch und Sebastian Hoffmeister (2016). *Order Constrained Solutions in k-Means Clustering for R*. URL: https://CRAN.R-project.org/package=ockc.

Krey, Sebastian und Uwe Ligges (2010). „SVM Based Instrument and Timbre Classification“. In: *Classification as a Tool for Research*. Hrsg. von Hermann Locarek-Junge und Claus Weihs. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 759–766. ISBN: 978-3-642-10745-0.

Krey, Sebastian, Uwe Ligges und Friedrich Leisch (2014). „Music and timbre segmentation by recursive constrained K-means clustering“. In: *Computational Statistics* 29.1, S. 37–50. DOI: 10.1007/s00180-012-0358-5.

Ligges, Uwe und Sebastian Krey (2011). „Feature Clustering for Instrument Classification“. In: *Computational Statistics* 26.2, S. 279–291. DOI: 10.1007/s00180-011-0234-8.

Literaturverzeichnis

Bellman, Richard (1957, 1972). *Dynamic Programming*. Princeton, New Jersey,USA: Princeton University Press. ISBN: 0-691-07951-X.

Beygelzimer, Alina, Sham Kakadet, John Langford, Sunil Arya, David Mount und Shengqiao Li (2013). *FNN: Fast Nearest Neighbor Search Algorithms and Applications*. URL: https://CRAN.R-project.org/package=FNN.

Bischl, Bernd, Michel Lang, Jakob Richter, Jakob Bossek, Leonard Judt, Tobias Kuehn, Erich Studerus, Lars Kotthoff und Julia Schiffner (2016). *mlr: Machine Learning in R*. URL: https://CRAN.R-project.org/package=mlr.

Childers, Donald G., David P. Skinner und Robert C. Kemerait (Okt. 1977). „The Cepstrum: A Guide to Processing“. In: *Proceedings of the IEEE* 65.10.

Cooley, James und John Tukey (1965). „An Algorithm for the Machine Calculation of Complex Fourier Series“. In: *Mathematics of Computation* 19.90, S. 297–301.

Davis, Steven B. und Paul Mermelstein (Aug. 1980). „Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences“. In: *IEEE Transactions on Acoustics, Speech and Signal Processing* ASSP-28.4.

Dolnicar, Sara und Friedrich Leisch (2010). „Evaluation of structure and reproducibility of cluster solutions using the bootstrap“. In: *Marketing Letters* 21, S. 83–101.

Ellis, Daniel P. W. (2005). *PLP and RASTA (and MFCC, and inversion) in Matlab*. online web resource. URL: http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/.

ENTSOE, Union for the co-ordination of transmission of electricity (2007). *Final Report System Disturbance on 4 November 2006*. URL: https://www.entsoe.eu/fileadmin/user_upload/_library/publications/ce/otherreports/Final-Report-20070130.pdf.

European Broadcasting Union (2011). *Specification of the Broadcast Wave Format (BWF)*. URL: https://tech.ebu.ch/docs/tech/tech3285.pdf.

Fink, Lester A. und Tobias A. Trygar, Hrsg. (1979). *Systems Engineering for Power: Emergency Operating State Control*. Davos, Switzerland.

Friedrichs, Klaus (2016). „Musikklassifikation mittels auditorischer Modelle zur Optimierung von Hörgeräten“. Diss. TU-Dortmund, Fakultät Statistik. URL: http://dx.doi.org/10.17877/DE290R-17176.

Frigo, Matteo und Steven G. Johnson (2005). „The Design and Implementation of FFTW3“. In: *Proceedings of the IEEE* 93.2. Special issue on “Program Generation, Optimization, and Platform Adaptation”, S. 216–231.

Giancoli, Douglas C. (2010). *Physik*. München: Pearson Deutschland GmbH, S. 561.

Graves, Daniel und W. Pedrycz (2009). „Multivariate Segmentation of Time Series with Differential Evolution“. In: *Proceedings of the Joint 2009 International Fuzzy Systems Association World Congress and 2009 European Society of Fuzzy Logic and Technology Conference*. Hrsg. von J.P. Carvalho, D. Dubois, U. Kaymak und J.M.C. Sousa, S. 1108–1113.

Halkidi, Maria, Yannis Batistakis und Michalis Vazirgiannis (2001). „On Clustering Validation Techniques“. In: *Journal of Intelligent Information Systems* 17.2-3, S. 107–145.

Hartigan, John Anthony und M. A. Wong (1979). „A K-means clustering algorithm“. In: *Applied Statistics* 28, S. 100–108.

Hastie, Trevor, Robert Tibshirani und Jerome Friedman (2001). *The Elements of Statistical Learning*. New York: Springer.

Hermansky, Hynek (Apr. 1990). „Perceptual linear predictive (PLP) analysis of speech“. In: *Journal of Acoustical Society of America* 87.4, S. 1738–1752.

Hoffmeister, Sebastian (2009). *Partitionierende Clusterverfahren unter Ordnungs-Nebenbedingungen*. Ludwig-Maximilians-Universität München. Diplomarbeit, Institut für Statistik.

Hsu, Chih-Wei, Chih-Chung Chang und Chih-Jen Lin (Mai 2008). *A Practical Guide to Support Vector Classification*. National Taiwan University. Taipei. URL: https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

Hubert, Lawrence und Phipps Arabie (1985). „Comparing partitions“. In: *Journal of Classification* 2, S. 193–218.

Johnson, Steven G. und Matteo Frigo (Sep. 2008). „Implementing FFTs in Practice“. In: *Fast Fourier Transforms*. Hrsg. von C. Sidney Burrus. Rice University, Houston TX: Connexions. Kap. 11. URL: http://cnx.org/content/m16336/.

Karatzoglou, Alexandros, Alex Smola, Kurt Hornik und Achim Zeileis (2004). „kernlab – An S4 Package for Kernel Methods in R“. In: *Journal of Statistical Software* 11.9, S. 1–20. URL: http://www.jstatsoft.org/v11/i09/.

Klapuri, Anssi und Manuel Davy (2006). *Signal Processing Methods for Music Transcription*. New York: Springer.

Kremer, Martina (2008). *ars auditus Akustik-Gehör-Psychoakustik*. Fachbereich E Elektrotechnik, Informationstechnik, Medientechnik: Bergische Universität Wuppertal. URL: http://web.archive.org/web/20080624225002/http://www.dasp.uni-wuppertal.de/ars_auditus.html.

Leisch, Friedrich (2006). „A Toolbox for K-Centroids Cluster Analysis“. In: *Computational Statistics and Data Analysis* 51.2, S. 526–544.

Leisch, Friedrich und Bettina Grün (2006). „Extending Standard Cluster Algorithms to Allow for Group Constraints“. In: *Compstat 2006—Proceedings in Computational Statistics*. Hrsg. von Alfredo Rizzi und Maurizio Vichi. Physica Verlag, Heidelberg, Germany, S. 885–892. ISBN: 3-7908-1708-2.

Liaw, Andy und Matthew Wiener (2002). „Classification and Regression by randomForest“. In: *R News* 2.3, S. 18–22. URL: http://CRAN.R-project.org/doc/Rnews/.

Lloyd, Stuart P. (1957, 1982). „Least squares quantization in PCM“. In: *IEEE Transactions on Information Theory* 28, S. 128–137.

Logan, Beth (2000). *Mel Frequency Cepstral Coefficients for Music Modeling*. Cambridge: Cambridge Research Laboratory Compaq Computer Corporation. URL: http://ismir2000.ismir.net/papers/logan_paper.pdf.

MacQueen, J.B. (1967). „Some methods for classification and analysis of multivariate observations“. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. Hrsg. von Lucien M. Le Cam und Jerzy Neyman. Berkeley, CA: University of California Press, S. 281–297.

Martinetz, Thomas M., Stanislav G. Berkovich und Klaus J. Schulten (1993). „"Neural-gas" Network for Vector Quantization and its Application to Time-Series Prediction". In: *IEEE-Transactions on Neural Networks* 4.4, S. 558–569.

Ng, Andrew Y., Michael I. Jordan und Yair Weiss (2002). „On Spectral Clustering: Analysis and an algorithm". In: *Advances in Neural Information Processing Systems 14*. Hrsg. von T. G. Dietterich, S. Becker und Z. Ghahramani. MIT Press, S. 849–856. URL: http://papers.nips.cc/paper/2092-on-spectral-clustering-analysis-and-an-algorithm.pdf.

Nyquist, H. (Feb. 2002). „Certain Topics in Telegraph Transmission Theory". In: *Proceedings of the IEEE* 90.2.

Opolko, Frank J. und Joel Wapnick (1987). *McGill University master samples (CDs)*.

Pai, Anantha (1994). *Energy Function Analysis for Power System Stability*. McGraw Hill.

Panda, Manasa R., Wendy Lecluyse, Christine M. Tan, Tim Jürgens und Ray Meddis (2014). „Hearing dummies: Individualized computer models of hearing impairment". In: *International Journal of Audiology* 53.10, S. 699–709. DOI: 10.3109/14992027.2014.917206.

Paulus, Jouni, Meinard Müller und Anssi Klapuri (2010). „Audio-based Music Structure Analysis". In: *Proceedings of the 11th International Society for Music Information Retrieval Conference*. Hrsg. von John Stephen Downie und Remco C. Veltkamp, S. 625–636.

Power Systems Engineering Research Center: Transfer Capability Calculator (2001). online web resource, abgerufen 16.02.2021. URL: https://www.pserc.cornell.edu/tcc/.

Queen Songs, Bohemian Rhapsody (2015). online web resource, abgerufen 16.02.2021. URL: http://web.archive.org/web/20210123014026/https://www.queensongs.info/song-analysis/songwriting-analyses/no-synth-era/a-night-at-the-opera/bohemian-rhapsody.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. Englisch. Vienna, Austria. URL: http://www.r-project.org.

Rand, William M. (1971). „Objective criteria for the evaluation of clustering methods". In: *Journal of the American Statistical Association* 66.336, S. 846–850.

Roever, Christian (2003). *Musikinstrumentenerkennung mit Hilfe der Hough-Transformation*. Universität Dortmund. Diplomarbeit, Fakultät Statistik. URL: http://wwwuser.gwdg.de/~croever/pub/RoeverDiplom.pdf.

Rogers, Graham (2000). *Power System Oscillations*. Kluwer Academic Publishers.

Saerens, Marco, François Fouss, Luh Yen und Pierre Dupont (2004). „The Principal Components Analysis of a Graph, and its Relationships to Spectral Clustering".

In: *Proceedings of the 15th European Conference on Machine Learning (ECML 2004). Lecture Notes in Artificial Intelligence*. Springer-Verlag, S. 371–383.

Schlittgen, Rainer und Bernd H.J. Streitberg (2001). *Zeitreihenanalyse*. 9. Auflage. Lehr- und Handbücher der Statistik. München: Oldenbourg.

Shannon, Claude E. (Feb. 1998). „Communication in the Presence of Noise“. In: *Proceedings of th IEEE* 86.2.

Slaney, Malcolm (1998). *Auditory Toolbox: A MATLAB Toolbox for Auditory Modeling Work Version 2*. Techn. Ber. 1998-010. Interval Research Corporation. URL: http://rvl4.ecn.purdue.edu/~malcolm/interval/1998-010/.

Smith, Julius O. III (Okt. 2007). *Introduction to Digital Filters: with Audio Applications*. W3K Publishing. URL: http://www.dsprelated.com/dspbooks/filters/.

Spruill, Marcus (2007). „Asymptotic Distribution of Coordinates on High Dimensional Spheres“. In: *Electron. Commun. Probab.* 12, S. 234–247. DOI: 10.1214/ECP.v12-1294.

Steinberg, John C. (1937). „Positions of Stimulation in the Cochlea by Pure Tones“. In: *The Journal of the Acoustical Society of America* 8.3, S. 176–180. DOI: 10.1121/1.1915891.

Steinley, Douglas und Michael J. Brusco (2011). „Choosing the Number of Clusters in K-Means Clustering“. In: *Psychological Methods* 16.3, S. 285–297.

Steinley, Douglas und Lawrence Hubert (2008). „Order-constrained solutions in K-means clustering: Even better than being globally optimal“. In: *Psychometrika* 73.5, S. 647–664.

Stevens, S.S., J. Volkmann und E. B. Newman (1937). „A Scale for the Measurement of the Psychological Magnitude Pitch“. In: *The Journal of the Acoustical Society of America* 8.3, S. 185–190. DOI: 10.1121/1.1915893.

Venables, William N. und Brian D. Ripley (2002). *Modern Applied Statistics with S*. Fourth Edition. ISBN 0-387-95457-0. New York: Springer. URL: http://www.stats.ox.ac.uk/pub/MASS4.

Wadhwa, C.L. (2005). *Electrical Power Systems*. New Delhi: New Age International.

Walker, James S. (1996). *Fast Fourier Transforms*. Second. Studies in Advanced Mathematics. Boca Raton, Florida: CRC Press.

Weingessel, Andreas, Evgenia Dimitriadou und Sara Dolnicar (1999). „An Examination Of Indexes For Determining The Number Of Clusters In Binary Data Sets“. In: *Working Paper Series* 29. URL: https://epub.wu.ac.at/1542/.

Wikipedia: Bohemian Rhapsody (2021). online web resource, abgerufen 16.02.2021. URL: https://de.wikipedia.org/wiki/Bohemian_Rhapsody.

Abbildungsverzeichnis

Tabellenverzeichnis