Entwicklung eines Monte-Carlo-Verfahrens zum selbständigen Lernen von Gauß-Mischverteilungen

Zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

des Fachbereichs Mathematik/Informatik

der Universität Osnabrück

genehmigte

Dissertation

von

Martin Lauer

aus Karlsruhe

Tag der mündlichen Prüfung: 11. November 2004
Erstgutachter: Prof. Dr. Martin Riedmiller, Universität Osnabrück
Zweitgutachter: Prof. Dr. Klaus-Robert Müller, Universität Potsdam

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

1.1 Datenanalysetechniken

Das in den letzten Jahrzehnten zu beobachtende rapide Anwachsen der Datenmengen, die aus statistischen Erhebungen und aus Mess- und Überwachungsapparaturen in allen Bereichen von Wissenschaft und Technik gewonnen werden, erfordert in immer größerem Ausmaß den Einsatz automatischer, adaptiver, aber gleichzeitig auch zuverlässiger Methoden der Datenanalyse. Das Aufkommen und die zunehmende Bedeutung der Forschungsgebiete *Data Mining*, *Maschinelles Lernen* und nicht zuletzt der *mathematischen Statistik* verdeutlichen diesen Trend.

Allen Aufgabenstellungen gemein ist die Notwendigkeit, eine gegebene Datenmenge möglichst gut zu beschreiben, wobei sich die Art der Beschreibung problemspezifisch unterscheiden kann. Die Qualität einer Beschreibung misst sich an zwei wesentlichen Gesichtspunkten: zum einen soll die Modellierung die gegebenen *Trainingsdaten* möglichst gut beschreiben, zum anderen soll das Modell aber auch zu bisher unbekannten Daten aus der selben Datenquelle passen (*Generalisierung*). Sowohl eine mangelhafte Anpassung an die Trainingsdaten (*Unteranpassung*[1]) als auch eine schlechte Generalisierungsleistung (*Überanpassung*[2]) sollen vermieden werden, um eine möglichst gute Prognoseleistung des Modells zu ermöglichen. Weitere Gesichtspunkte sind die Größe der Datenbeschreibung, der Rechenaufwand zur Bestimmung und Auswertung des Modells sowie die Automatisierbarkeit der Analyseverfahren.

Eine zentrale Rolle im Bereich der Datenanalyse spielt die Schätzung von Wahrscheinlichkeitsverteilungen, da nahezu alle anderen Aufgabenstellung darauf zurückgeführt werden können. Diese Arbeit konzentriert sich auf den Spezialfall der Dichteschätzung, d.h. auf Wahrscheinlichkeitsverteilungen mit kontinuierlicher Grundgesamtheit, die bereits einen wichtigen Teil der Problemstellungen abdeckt.

Als sehr nützliche Modellklasse für die Dichteschätzung haben sich die *Gauß-Mischverteilungen* (GMM)[3] herauskristallisiert, da sie einerseits sehr flexibel sind und sich daher an viele verschiedene Verteilungen anpassen können, andererseits aber auch leicht in ihrer Größe variiert werden können, so dass sich sowohl einfache Verteilung mit kleinen GMMs als auch komplizierte Verteilungen mit großen GMMs beschreiben lassen. Gauß-Mischungen

[1] englisch: *underfitting*
[2] englisch: *overfitting*
[3] englisch: *Gaussian mixture model*

eignen sich daher sehr gut für den Einsatz auf Daten, über deren Verteilung keine Annahmen getroffen werden können.

Als Lernverfahren zur Bestimmung geeigneter Modellausprägungen hat sich für GMMs in erster Linie der sogenannte *Expectation-Maximization*-Algorithmus (EM) durchgesetzt, der von einer einfachen Grundidee ausgehend einen iterativen Lernprozess realisiert. Er basiert auf dem sehr weit verbreiteten *Maximum-Likelihood*-Schätzprinzip. Da der EM-Algorithmus stets von einer festen, vorgegebenen Modellgröße ausgeht, erfordert die Bestimmung einer geeigneten Modellkomplexität zusätzliche Ansätze zur Modellselektion.

1.2 Die Fallstricke des EM-Ansatzes

Der EM-Algorithmus für GMMs geht von der Annahme kontinuierlich verteilter Daten aus. Die GMM-Größe muss zu Struktur und Anzahl der Datenpunkte adäquat gewählt sein. Insbesondere führen sehr kleine Trainingsmengen zu Schwierigkeiten.

Die Beschaffenheit der Daten in vielen Messungen und Erhebungen steht im Gegensatz zu den Annahmen des EM-Ansatzes. Typische Phänomene sind:

- Datensätze sind häufig höherdimensional. Univariate Daten sind eher eine Ausnahme; oft haben Daten eine größere einstellige oder gar zweistellige Anzahl Attribute.

- Die Anzahl Trainingsdaten ist häufig sehr klein, vor allem wenn die Erhebung der Daten sehr aufwändig ist, z.B. in medizinischen Untersuchungen. Typische Trainingsmengen enthalten allenfalls einige Hundert Muster, selten mehr als Tausend Datenpunkte.

- Viele Attribute nehmen nur diskrete Werte an, selbst wenn sie aus an für sich kontinuierlich verteilten Grundgesamtheiten stammen. Beispielsweise ist das Alter eines Probanden in einer medizinischen Erhebung im Grunde ein kontinuierlich verteiltes Attribut, es wird jedoch häufig auf ganze Jahre gerundet, so dass nur ganzzahlige Werte angenommen werden. Andere Gründe für das Auftreten diskreter Werte ist eine beschränkte Messgenauigkeit oder die Einteilung von Werten in Klassen.

- Neben vollständig diskretisierten Attributen kann auch der Fall semikontinuierlicher Verteilungen auftreten. Dies bedeutet, die Verteilung enthält einen kontinuierlich verteilten Grundanteil und zusätzlich einen Anteil diskreter Werte, so dass bestimmte Werte bevorzugt werden. Dieser Fall ist insofern schwierig, als eine Modellausprägung, die einen bestimmten Wert stark bevorzugt, entweder gewünscht sein kann – dann nämlich, wenn dieser Wert zu dem diskreten Anteil gehört – oder aber auf Overfitting zurückzuführen ist und somit eine Degenerierung des Modells darstellt.

- Datensätze können sowohl kontinuierlich verteilte Attribute als auch Attribute mit diskreter Verteilung aufweisen, z.B. Körpergröße und Geschlecht eines Probanden in einer medizinischen Erhebung.

- Die Verteilung der Daten kann vielgestaltig sein; es können sowohl dünn besetzte Bereiche als auch dicht besetzte Bereiche auftreten, die Verteilungen können schief oder langschwänzig sein.

- Die Datensätze können untypische Datenpunkte und Ausreißer enthalten, z.B. aufgrund von Fehlmessungen.
- Die Daten können lückenhaft sein, d.h. es gibt Datenpunkte, für die die Werte einzelner Attribute fehlen.

Diesen komplexen Anforderungen stehen die Beschränkungen des EM-Algorithmus gegenüber. Im Wesentlichen lassen sie sich in vier Punkten fassen:

(i) Der EM-Algorithmus an sich ist nicht in der Lage, die passende Modellkomplexität zu bestimmen. Ein zu großes GMM führt jedoch zu Overfitting, während ein zu kleines GMM zu Underfitting führt.

(ii) Der EM-Algorithmus führt nur eine lokale Optimierung aus. Es ist bereits eine gute Initialisierung nötig, damit das Verfahren eine gute Lösung berechnen kann. Andernfalls dauert der Lernvorgang sehr lange und das resultierende GMM verteilt die einzelnen Komponenten ungeschickt im Datenraum.

(iii) Die Likelihood-Funktion ist unbeschränkt. Das EM-Verfahren als Maximum-Likelihood-Schätzer ist somit bereits prinzipiell schlecht fundiert. Es stellt sich heraus, dass die Modellausprägungen, die zu einer unbeschränkt wachsenden Likelihood führen, diejenigen sind, in denen sich das GMM an die gegebenen Daten überanpasst. Das bedeutet, die Daten-Likelihood ist nur in beschränktem Maße ein geeignetes Gütekriterium und die Maximierung derselben kann zu schlechten Ergebnissen führen. Insbesondere bei semikontinuierlichen oder diskreten Daten sowie bei der Existenz von Ausreißern oder bei Verwendung eines zu großen GMM tritt dieser Effekt auf.

(iv) Der EM-Algorithmus vermeidet nicht die Degenerierung einzelner Komponenten des GMM durch Auftreten von singulären Kovarianzmatrizen. Im Gegenteil, im Overfitting-Fall werden solche degenerierten Komponenten sogar bevorzugt erzeugt.

Verschiedene Varianten und Erweiterungen des EM-Verfahrens wurden entwickelt, um diesen Problemen Herr zu werden. Eine Auswahl der wichtigsten Ansätze:

- Wiederholte Ausführung des Lernverfahrens mit verschieden großen GMMs und die Verwendung analytischer Modellselektionskriterien wie AIC, BIC und Evidenz erlauben die Bestimmung einer geeigneten Modellgröße (i). Allerdings erfordern sie mehrere Trainingsläufe und vergrößern damit den Lernaufwand nicht unerheblich, vor allem wenn a priori keine Annahmen über eine geeignete Modellgröße gemacht werden können. Zudem kann der Overfittingeffekt zu einer fehlerhaften Modellauswahl führen.
- Wiederholte Ausführung des Lernverfahrens mit verschieden großen GMMs und die Verwendung von Kreuzvalidierung zur Bestimmung einer geeigneten Modellgröße (i). Dieser Ansatz wird zwar von Overfitting nicht fehlgeleitet, vergrößert aber den Lernaufwand um ein Vielfaches.
- Verschiedene heuristische Verfahren zur Vergrößerung oder Verkleinerung des GMMs können bei der Bestimmung einer geeigneten Modellgröße helfen, allerdings sind die Vorgehensweisen teilweise unklar oder hängen von Verfahrens-Parametern ab, deren Einstellung schwierig oder aufwändig ist.

- Vorverarbeitung des initialen GMM mit Clustering-Algorithmen, z.B. *k-means*-Clustering. Dadurch können gute Initialisierungen erzielt werden und damit dem Problem lokaler Optima und langsamen Lernfortschritts (ii) zumindest teilweise entgegnet werden.

- Der Split&Merge EM-Algorithmus [Ueda 98] kann helfen, eine ungeeignete Verteilung der Komponenten im Datenraum (ii) zu vermeiden, allerdings steigt die Rechenzeit massiv an.

- Stochastische Versionen des EM-Ansatzes (SEM, SAEM, MCEM) können ebenfalls lokale Optima umgehen und damit die Problematik (ii) abschwächen. Der positive Effekt bleibt allerdings in der Praxis eher gering.

- Der Übergang vom Maximum-Likelihood-Schätzer zum Maximum-a-posteriori-Schätzer kann helfen, Overfitting (iii) und Degenerierung (iv) zu vermeiden. Allerdings müssen dafür Prioriverteilungen bestimmt werden, deren Wahl die Ergebnisse massiv beeinflussen. Eine Bestimmung dieser Prioriverteilungen ist daher ein sehr großes Problem. Zudem kann eine unterschiedliche Ausprägung der Datenverteilung in verschiedenen Bereichen des Datenraums dazu führen, dass keine Prioriverteilung für alle Bereiche wirklich optimal ist.

- Die Verwendung von Komitees von GMMs kann die Ergebnisse deutlich verbessern, allerdings auf Kosten einer längeren Rechenzeit und eines insgesamt größeren Modells.

Aufgrund der beschriebenen Nachteile, vor allem im Hinblick auf die Rechenzeit, spielen nur die Vorverarbeitung der GMMs mit *k-means*-Clustering sowie die Modellselektion mit analytischen Kriterien oder Kreuzvalidierung eine größere Rolle in der praktischen Umsetzung. Vor allem die Probleme des Overfitting und der numerischen Instabilität, gerade bei Verwendung diskretisierter Datensätze, können damit allerdings nicht gelöst werden.

1.3 Markov-Chain-Monte-Carlo Techniken

Alternativ zum EM-Ansatz wurden, basierend auf dem Gibbs-Sampler und dem Metropolis-Hastings-Algorithmus, Monte-Carlo Verfahren für GMMs entwickelt. Diese samplen eine Folge von zufälligen Modellausprägungen gegeben eine Menge von Trainingsdaten. Die dabei verwendeten Samplingverteilungen entspringen einer Bayesschen Modellierung.

Die Monte-Carlo Verfahren unterscheiden sich in ihrem Verhalten grundlegend vom EM-Ansatz. Nicht die Optimierung eines Gütekriteriums durch sukzessive lokale Verbesserungen sondern die Erzeugung einer Zufallsfolge von Modellausprägungen kennzeichnen diese Algorithmen. Dabei sind in jeder Iteration prinzipiell auch größere Veränderungen der Modellausprägung möglich, so dass globale Konvergenz garantiert werden kann. Allerdings handelt es sich nicht um Konvergenz in eine bestimmte, beste Modellauprägung, sondern um die Konvergenz der gesampleten Modellausprägungen gegen die Posterioriverteilung. Zu jedem Zeitpunkt können also auch schlechte Modellausprägungen angenommen werden.

So vorteilhaft eine globale Konvergenz gegen eine bestimmte Verteilung auch ist, so schwierig ist es, aus der Folge gesampleter Modellausprägungen ein bestes Modell zu berechnen, da die gesampleten Ausprägungen fast durchweg suboptimal sind und auch die Auswahl einer besten Ausprägung unter allen gesampleten wiederum ein Selektionskriterium erfordert.

Auch eine einfache Mittelung über alle Modellausprägungen ist nicht zielführend, nicht zuletzt aufgrund des mitunter breit gestreuten Samplens über weite Bereiche des Parameterraums.

Die Wahl der Modellgröße ist ebenso eine offene Frage. Zwar existieren Markov-Chain-Monte-Carlo Verfahren, die auch über die GMM-Größe variieren, allerdings garantieren die theoretischen Aussagen lediglich, dass GMMs bestimmter Größe zu einem gewissen relativen Anteil in der Folge gesampleter Parameter auftreten und erlauben nicht die Ableitung einer besten Modellgröße.

Schließlich ist auch die experimentelle Evaluierung der Monte-Carlo Ansätze zum Lernen von GMMs bisher nur unzureichend erfolgt. Die empirischen Ergebnisse in [Diebolt 94], [Roeder 95], [Richardson 97] und [Stephens 97] beschränken sich weitgehend auf künstliche oder leicht visualisierbare uni- und bivariate Datensätze. Die Bewertung der Ergebnisse erfolgt überwiegend durch visuelle Inspektion und lässt somit keinen qualitativen Vergleich mit dem EM-Ansatz zu.

1.4 Zielsetzung der Arbeit

Das Ziel dieser Arbeit ist die Entwicklung eines randomisierten Lernverfahrens für GMMs. Es soll die zuverlässige Schätzung der Modellparameter sowie die Bestimmung der Modellgröße vereinen und somit die Gesamtaufgabe *Finde ein GMM zu gegebener Datenmenge* als Ganzes lösen. Dabei soll das Verfahren einen automatischen Lernprozess realisieren, der ohne menschliche Intervention auskommt, aber dennoch Lösungen mit guter Generalisierung auch auf semikontinuierlichen und diskretisierten Daten liefert.

Zur Konstruktion eines geeigneten Verfahrens ist zunächst die Arbeitsweise randomisierter Algorithmen für die Dichteschätzung zu untersuchen, wobei über die stochastische Analyse der stationären Eigenschaften hinaus auch die Dynamik des Ansatzes betrachtet werden wird. Durch Berücksichtigung dieser dynamischen Eigenschaften sollen Aussagen über die Generalisierungsleistung und die Abhängigkeiten zwischen Modellgröße und Lernverhalten abgeleitet werden.

Eine systematische und umfassende empirische Untersuchung der Algorithmen soll das Erreichen der Zielsetzung auch experimentell nachweisen. Dazu wird der neu entwickelte Ansatz mit bestehenden Verfahren anhand der Kriterien *Anpassungsgüte*, *Generalisierung* und *Robustheit* verglichen.

Die zentralen Untersuchungsschritte auf dem Weg zu einem Ansatz, der der formulierten Zielsetzung entspricht, lassen sich in vier Punkten zusammenfassen:

- Untersuchung des Verhaltens der Samplingverfahren im Hinblick auf die Generalisierungsleistung; Optimierung der Ansätze und Entwicklung von Empfehlungen für die praktische Anwendung der Algorithmen
- Erweiterung der Lernalgorithmen um eine Methode zur Bestimmung der GMM-Größe
- Entwicklung einer Methode zur Bestimmung und Berechnung der besten Modellausprägung
- Systematische und umfassende experimentelle Validierung der Ergebnisse und Vergleich mit dem EM-Algorithmus

Bei der Entwicklung des Lernverfahrens soll großer Wert auf den Aspekt der Robustheit und Zuverlässigkeit gelegt werden. Degenerierungserscheinungen, wie beim EM-Algorithmus, sollen weitgehend ausgeschlossen sein, und die Methodik soll auch bei Daten, die die Annahme der kontinuierlichen Verteilung nicht erfüllen, vernünftige Ergebnisse liefern. Erst diese Forderung erlaubt die autonome Verwendung des Ansatzes auch ohne ständige menschliche Kontrolle.

1.5 Aufbau und Vorgehensweise

Der Aufbau der Arbeit besteht aus drei Teilen: der Darstellung der theoretischen Grundlagen und der existierenden Verfahren, der Weiterentwicklung der Monte-Carlo Verfahren sowie der experimentellen Validierung und Bewertung.

Zunächst werden in Kapitel 2 die zugrunde liegende Lernaufgabe, die grundlegenden Lösungsansätze und die Begriffe *Generalisierung* und *Overfitting* im Kontext von Dichteschätzung dargestellt. Kapitel 3 beschreibt die Grundlagen der Markov-Chain-Monte-Carlo Verfahren und führt die wesentlichen Algorithmentypen dieses Gebietes ein. Anschließend werden in Kapitel 4 die beiden Gebiete *Dichteschätzung* und *Markov-Chain-Monte-Carlo* zusammengeführt, indem die prinzipielle Verwendung von Samplingverfahren für Mischverteilungen hergeleitet wird. In den ersten drei Kapiteln werden im Wesentlichen der Stand der Wissenschaft dargestellt und die notwendigen Grundlagen zusammengetragen.

In den Kapiteln 5 und 6 steht die Weiterentwicklung der Samplingverfahren für die praktische Anwendung im Mittelpunkt. Dazu wird in Kapitel 5 das Verhalten eines existierenden Ansatzes untersucht und im Hinblick auf Generalisierungsverhalten interpretiert. Es werden Erweiterungen zur Bestimmung der GMM-Größe und Berechnung eines besten GMMs erarbeitet. In Fortführung dieses Ansatzes werden in Kapitel 6 Möglichkeiten untersucht, um den Samplingansatz durch Kombination mit dem EM-Algorithmus sowie durch Komiteebildung zu optimieren. Dabei wird das Verfahren *Randomisiert EM (REM)* entwickelt, das der Zielsetzung dieser Arbeit in idealer Weise entspricht.

In Kapitel 7 werden die in Kapitel 5 und 6 entwickelten Verfahren empirisch untersucht. In verschiedenen Experimenten werden sowohl die Verfahren als Ganzes auf Benchmarkdaten verglichen, als auch Teilaspekte wie Laufzeit, Modellgröße und Komitee-Effekt analysiert. Die experimentellen Ergebnisse und die Beobachtungen, die zur Entwicklung der randomisierten Verfahren geführt haben, werden anschließend in Kapitel 8 zusammengeführt und bewertet.

Zur Verdeutlichung der Nützlichkeit randomisierter Ansätze zur GMM-Schätzung werden in Kapitel 9 eine Reihe von Erweiterungen und Anwendungen von GMMs dargestellt. Dieses Kapitel ist kurz gehalten, da eine ausführliche Darstellung der verschiedenen Möglichkeiten den Rahmen dieser Arbeit sprengen würde.

Schlussendlich werden in Kapitel 10 die Kernpunkte der Arbeit zusammengestellt und die Ergebnisse mit der Zielsetzung verglichen.

Kapitel 2

Dichteschätzung

2.1 Lernaufgabe

Die zentrale Aufgabenstellung im unüberwachten Lernen ist die Beschreibung einer gegebenen Datenmenge (Trainingsdaten) durch ein Modell. Je nach Struktur der Daten sowie konkreter Zielsetzung eignen sich hierfür Clusteranalyse [Kaufman 90], die Bestimmung von Bereichen hoher Dichte [Ben-David 97, Tax 99] oder Ausreißeranalyse [Barnett 78]. Die allgemeinste Form der Beschreibung ist die Bestimmung der Wahrscheinlichkeitsverteilung, aus der die gegebenen Trainingsdaten stammen. Einen Spezialfall der Bestimmung einer Wahrscheinlichkeitsverteilung stellt die Dichteschätzung dar.

Die Aufgabe der Dichteschätzung lässt sich folgendermaßen beschreiben:

> *Gegeben eine unabhängige Stichprobe $\{x_1, \ldots, x_n\}$ aus einer unbekannten Verteilung P mit Dichte im $\mathbb{R}^d$, bestimme eine Wahrscheinlichkeitsverteilung Q mit Dichte, so dass P und Q sich möglichst wenig unterscheiden.*

Die beiden Verteilungen P und Q können durch ihre Dichtefunktionen beschrieben werden, $p(\cdot)$ und $q(\cdot)$. Der Unterschied zwischen zwei Dichten kann mit Hilfe der Kullback-Leibler Divergenz $KL(p, q)$ gemessen werden (siehe z.B. [Vapnik 95, MacKay 03]):

$$KL(p, q) := \int p(t) \log \frac{p(t)}{q(t)} dt \tag{2.1}$$

Das Integral in (2.1) reicht über den gesamten Träger der beiden Verteilungen p und q. Die Kullback-Leibler Divergenz ist stets größer als Null für $p \neq q$ und Null falls $p = q$. Sie ist offensichtlich nicht symmetrisch. Ersetzt man p durch die empirische Verteilung der Stichprobe und ist q die gelernte Dichte, so ist die Minimierung der Kullback-Leibler Divergenz gleichbedeutend mit der Maximierung der Daten-Log-Likelihood $LL(x_1, \ldots, x_n|q)$:

$$LL(x_1, \ldots, x_n|q) := \sum_{i=1}^{n} \log q(x_i) \tag{2.2}$$

Die Reduzierung des Lernens auf die Suche nach einer Dichtefunktion q, die die Daten-Log-Likelihood $LL(x_1, \ldots, x_n|q)$ minimiert, wird der ursprünglichen Aufgabenstellung allerdings nicht gerecht, da die empirische Verteilung der Stichprobe nur eine Approximation der wahren Dichte p ist. Insbesondere ist die Daten-Log-Likelihood unbeschränkt; es lassen sich

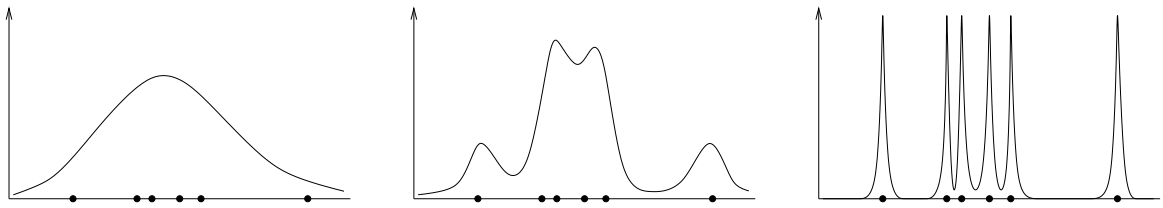

Abbildung 2.1: Folge von drei Dichten, die sich der Stichprobe (schwarze Punkte) immer stärker überanpassen. Die Dichte im rechten Diagramm beschreibt offensichtlich nicht mehr die erzeugende Verteilung, sondern nur noch die Stichprobe. Diese Dichte besitzt unter den dargestellten drei Verteilungen die größte Daten-Log-Likelihood auf den Trainingsdaten.

Folgen von Dichtefunktionen finden, deren Daten-Log-Likelihood unbeschränkt wächst und die sich immer mehr der empirischen Verteilung der Stichprobe annähern. Eine beispielhafte Folge derartiger Dichten zeigt Abbildung 2.1.

Es ist daher erforderlich, diesem *Overfitting*-Phänomen durch geeignete Maßnahmen entgegenzuwirken. Dies beinhaltet zum einen die Verkleinerung des Hypothesenraums und zum anderen die Einführung von Regularisierungstermen. Unter dem Hypothesenraum $\mathcal{H}$ versteht man die Menge der Dichten $q(\cdot)$, die man als Lösungen des Schätzproblems zulässt. Durch geeignete Wahl des Hypothesenraums können Dichten, die zum Overfitting neigen, ausgeschlossen werden.

Die Verwendung von Regularisierungstermen schließt dagegen keine Dichten vollkommen als Lösung der Lernaufgabe aus. Statt dessen werden bestimmte Dichten bevorzugt und andere benachteiligt. Durch Addition eines geeigneten Regularisierungsterms zur Daten-Log-Likelihood ergibt sich die zu maximierende Zielfunktion dieses Ansatzes. Der Regularisierungsterm sollte so gewählt werden, dass zum Overfitting neigende Dichten bestraft werden und daher nur dann gelernt werden, wenn aufgrund der Trainingsdaten alle alternativen Lösungen ausgeschlossen werden können.

Die Schwierigkeit, im Bereich der Dichteschätzung die Balance zwischen optimaler Anpassung an die Trainingsdaten und Vermeidung von Overfitting zu wahren, wird in [Vapnik 95] so geschildert:

> *(i) Im Allgemeinen ist die Schätzung einer Dichte ein schlecht gestelltes Problem.*
> *(ii) Um dieses Problem gut zu lösen, muss man Regularisierungstechniken (d.h. keine selbsterklärenden Techniken) einsetzen.*

In der Stochastik haben sich zur Lösung des Dichteschätzproblems zwei Vorgehensweisen etabliert, die in den folgenden Abschnitten vorgestellt werden.

2.2 Parametrisierte Modelle

2.2.1 Lernen eines parametrisierten Verteilungsmodells

Eine sehr weit verbreitete Art der Dichteschätzung ist die Verwendung parametrisierter Dichten (siehe z.B. [Stahel 95]). Hierbei wird die Menge der möglichen Dichten auf einen

kleinen Hypothesenraum $\mathcal{H}$ eingeschränkt, der durch eine parametrisierte Funktion gegeben ist. Bezeichne nun $q(\cdot|\vartheta)$ eine durch ϑ parametrisierte Wahrscheinlichkeitsdichte, wobei ϑ aus einer Menge möglicher Parameter Θ stammt[1]. Dann ist der Hypothesenraum bestimmt durch $\mathcal{H} = \{q(\cdot|\vartheta)|\vartheta \in \Theta\}$. In der Regel ist der Hypothesenraum eine sehr kleine Teilmenge aller möglichen Dichtefunktionen.

Das Lernen einer Dichte aus gegebenen Trainingsdaten geschieht durch die Suche nach einem möglichst guten Parameter $\hat{\vartheta} \in \Theta$. Im Falle des Maximum-Likelihood-Schätzers (ML) wird $\hat{\vartheta}$ so gewählt, dass die Daten-Log-Likelihood (2.2) maximiert wird, d.h.:

$$\hat{\vartheta}_{ML} := \arg\max_{\vartheta \in \Theta} \sum_{i=1}^{n} \log q(x_i|\vartheta) \tag{2.3}$$

Die Dichte mit der höchsten Daten-Log-Likelihood aus dem Hypothesenraum $\mathcal{H}$ ist daher $q(\cdot|\hat{\vartheta}_{ML})$. Die Regularisierung erfolgt hierbei also ausschließlich durch geeignete Wahl des Hypothesenraums. Das bedeutet, dass die Maximum-Likelihood-Methode nur dann einen guten Schätzer liefert, wenn der Hypothesenraum geeignet gewählt ist. Gängige Hypothesenräume sind die Menge aller Normalverteilungen, die Menge aller Gammaverteilungen, oder ähnliche.

Beispiel (Univariate Normalverteilung) Die Menge der univariaten Normalverteilungen ist parametrisierbar durch zwei Parameter, den Erwartungswert und die Varianz, d.h. $\vartheta := (\mu, \sigma^2) \in \Theta := \mathbb{R} \times \mathbb{R}_{>0}$. Die Hypothesenmenge ist daher $\mathcal{H} = \{\varphi_{\mu,\sigma^2}(\cdot)|\mu \in \mathbb{R}, \sigma^2 \in \mathbb{R}_{>0}\}$ wobei $\varphi_{\mu,\sigma^2}(\cdot)$ die Dichte der Normalverteilung mit Erwartungswert μ und Varianz σ^2 bezeichne (siehe Anhang B).

Den Maximum-Likelihood-Schätzer für die gegebenen Trainingsdaten $\{x_1, \ldots, x_n\}$ erhält man direkt durch Ableiten der Daten-Log-Likelihood nach den beiden Parametern und Nullsetzen der Ableitungen zu: $\hat{\mu}_{ML} = \frac{1}{n}\sum_{i=1}^{n} x_i$ und $\hat{\sigma}^2_{ML} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu}_{ML})^2$.

Selbst in diesem Beispiel einer relativ kleinen Hypothesenmenge kann Overfitting auftreten. Für den pathologischen Fall einer ein-elementigen Stichprobe entartet der ML-Schätzer zu $\hat{\sigma}^2_{ML} \to 0$, da die Daten-Log-Likelihood für gegen Null strebende Varianzen unbeschränkt ist. Besteht die Stichprobe dagegen aus mindestens zwei nicht identischen Elementen, bleibt die Daten-Log-Likelihood nach oben beschränkt. Der geschilderte Fall extremen Overfittings kann daher nicht auftreten.

■

Neben dem beschriebenen Maximum-Likelihood-Schätzer werden auch verschiedene andere Schätzprinzipien verwendet, z.B. die Momentenmethode. Diese Ansätze sollen hier jedoch nicht weiter beschrieben werden.

2.2.2 Regularisierungsansätze für parametrisierte Methoden

Um Overfitting bei der Verwendung parametrisierter Methoden zu vermeiden, sind zwei Vorgehensweisen möglich. Zum einen kann der Hypothesenraum so verkleinert werden, dass die zum Overfitting neigenden Ausprägungen vollständig ausgeschlossen werden, und zum anderen kann durch Bevorzugung bestimmter Parameterwerte und Benachteiligung anderer die Ausprägung von Overfitting gehemmt werden.

[1] ϑ kann hier sowohl als skalare Größe als auch als vektorwertige Variable verstanden werden.

Eine Methodik, um bestimmte Parameterwerte und damit bestimmte Hypothesen zu bevorzugen bzw. zu benachteiligen, stellt die Maximum-a-posteriori-Schätzung (MAP) dar. Sie basiert auf der Bestimmung der Posterioriverteilung der Parameter gegeben die Trainingsdaten $\mathcal{P}(\vartheta|x_1 \ldots, x_n)$. Mit dem Satz von Bayes erhält man:

$$\mathcal{P}(\vartheta|x_1, \ldots, x_n) = \mathcal{P}(x_1, \ldots, x_n|\vartheta) \cdot \frac{\mathcal{P}(\vartheta)}{\mathcal{P}(x_1, \ldots, x_n)} \tag{2.4}$$

Geht man von einer fest vorgegebenen Trainingsmenge $\{x_1, \ldots, x_n\}$ aus, so nimmt die Größe $\mathcal{P}(x_1, \ldots, x_n)$ einen konstanten Wert an, unabhängig von ϑ.

Die MAP-Schätzung besteht darin, die logarithmierte Posterioriwahrscheinlichkeit zu maximieren, d.h. man bestimmt den wahrscheinlichsten aller möglichen Parameter im Bezug auf die gegebenen Trainingsdaten. Mit Gleichung (2.4) ergibt sich:

$$\hat{\vartheta}_{MAP} := \arg\max_{\vartheta \in \Theta}(\sum_{i=1}^{n} \log q(x_i|\vartheta) + \log \mathcal{P}(\vartheta)) \tag{2.5}$$

Die zu maximierende Funktion setzt sich aus zwei Teilen zusammen, einerseits der Daten-Log-Likelihood und andererseits der logarithmierten Prioriwahrscheinlichkeit $\mathcal{P}(\vartheta)$. Letztere modelliert eine Verteilung über dem Raum der Parameterwerte Θ und ermöglicht dadurch die Bevorzugung bestimmter Parameter (solche mit hoher Prioriwahrscheinlichkeit) und die Benachteiligung anderer (solche mit niedriger Prioriwahrscheinlichkeit). Der Maximum-Likelihood-Ansatz ergibt sich als Spezialfall für konstantes $\mathcal{P}(\vartheta)$, d.h. wenn keine Parameter bevorzugt oder benachteiligt werden.

Eine ausführlichere Betrachtung der Zusammenhänge zwischen Priori- und Posterioriverteilungen erfolgt in Abschnitt 4.2.2.

Beispiel (Univariate Normalverteilung, fortgesetzt) Im Beispiel der univariaten Normalverteilung aus Abschnitt 2.2.1 trat Overfitting vor allem dann auf, wenn die Varianzen zu klein wurden. Durch Einführung einer Prioriverteilung soll dies verhindert werden, indem kleinen Varianzen kleine Prioriverteilungen zugeordnet werden. Beim Erwartungswert sollen dagegen keine Werte bevorzugt werden, da der Lageparameter der Normalverteilung nicht kritisch im Hinblick auf Overfitting ist.

Eine geeignete Prioriverteilung für die Varianz stellt die Inverse Gammaverteilung (siehe Anhang B) dar: $\sigma^2 \sim \Gamma^{-1}(\beta, \alpha)$ mit geeigneten Werten für β und α. Damit ergibt sich als zu minimierende Funktion: $\sum_{i=1}^{n} \log \varphi_{\mu,\sigma^2}(x_i) - (\alpha + 1) \log \sigma^2 - \frac{\beta}{\sigma^2}$. Ableiten nach den beiden Parametern und Nullsetzen der Ableitungen ergibt den MAP-Schätzer: $\hat{\mu}_{MAP} = \frac{1}{n} \sum_{i=1}^{n} x_i$ und $\hat{\sigma}^2_{MAP} = \frac{\sum_{i=1}^{n}(x_i - \hat{\mu}_{MAP})^2 + 2\beta}{n+2+2\alpha}$.

Im Vergleich von Maximum-Likelihood- und Maximum-a-posteriori-Schätzer lassen sich die Unterschiede erkennen: für große Stichprobenumfänge unterscheiden sich ML- und MAP-Schätzer kaum, da die Größen 2β und $2+2\alpha$ klein bleiben gegenüber $\sum_{i=1}^{n}(x_i - \hat{\mu}_{MAP})^2$. Für kleine Stichprobenumfänge dagegen dominieren die Parameter α und β den MAP-Schätzer, d.h. in diesem Fall gehen die Daten nur zu einem geringeren Teil bei der Bestimmung der Varianz ein.

■

2.2.3 Bewertung parametrisierter Dichteschätzer

Parametrisierte Modelle sind eine in der Stochastik vielfach verwendete Methodik, die für viele Aufgabenstellungen vollkommen ausreichend ist. So lange ein Verteilungsmodell bekannt ist, reduziert sich die Lernaufgabe auf die Bestimmung einiger Parameter. Gängige Modelle besitzen ein, zwei oder maximal drei Parameter, die analytisch oder mit einfachen numerischen Algorithmen leicht zu bestimmen sind.

Vorteile der Methodik sind der geringe Rechenaufwand, der geringe Speicheraufwand, da lediglich einige wenige Parameter zu speichern sind, sowie die insgesamt relativ geringe Neigung zum Overfitting, sofern die Trainingsstichproben nicht zu klein sind.

Probleme bereiten diese Ansätze allerdings dann, wenn wenig oder gar nichts über die Quelle der Trainingsdaten bekannt ist, d.h. wenn keine Verteilungsannahme vorliegt. Viele Datensätze besitzen eine Verteilung, die nicht durch Standardmodelle beschrieben werden kann. Wird trotzdem eines der üblichen parametrisierten Modelle verwendet, kommt es zum *Underfitting*, d.h. zur suboptimalen Anpassung in Folge des zu kleinen oder unangemessenen Hypothesenraums.

Die Regularisierungsansätze mittels Maximum-a-posteriori-Schätzung besitzen ebenfalls einen großen Nachteil der ihren Einsatz erschwert: es muss vorweg eine Prioriverteilung festgelegt werden, die mitunter einen erheblichen Einfluss auf die Schätzung nimmt. Parameter der Prioriverteilung, z.B. die Werte β und α im Normalverteilungsbeispiel, sind i.d.R. nur experimentell bestimmbar, da häufig keine plausiblen Annahmen über die Verteilung der Modell-Parameter ϑ vorliegen. Ansätze zur automatischen Bestimmung der Prioriverteilungen aus dem Gebiet der empirischen/hierarchischen Bayesschen Methoden [Robert 94] erfordern erheblichen zusätzlichen Aufwand.

2.3 Verteilungsfreie Verfahren

Neben den Dichteschätzern basierend auf einem Verteilungsmodell hat sich eine weitere Klasse von Algorithmen herausgebildet, die als verteilungsfreie oder nichtparametrische Schätzer bekannt geworden sind. Im folgenden soll nur auf die Kernbasierten Verfahren eingegangen werden. Einen Überblick über die Thematik findet sich in [Duda 73, Devroye 87, Silverman 86].

Bei den Kernbasierten Verfahren, auch als Parzen-Fenster-Technik [Parzen 63] bekannt, wird versucht, durch eine Art Glättung der Stichprobe eine Dichte zu berechnen. Dazu wird eine Kernfunktion $K(\cdot)$ benötigt, die die Anforderungen an eine Wahrscheinlichkeitsdichte erfüllt, d.h. (i) $K(x) \geq 0$ für alle x und (ii) $\int K(x)dx = 1$. Typische Kernfunktionen sind radialsymmetrisch um den Ursprung, wobei die Dichte mit zunehmendem Abstand zum Ursprung abnimmt. Drei Beispiele für derartige Kernfunktionen zeigt Abbildung 2.2

Gegeben eine Kernfunktion K und einen Breitenparameter $h \in \mathbb{R}_{>0}$ ergibt sich die geschätzte Dichte zu gegebenen d-variaten Trainingsdaten $\{x_1, \ldots, x_n\}$ zu:

$$\hat{q}_{Parzen}(x) := \frac{1}{n \cdot h^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{2.6}$$

Der Parameter h legt den Grad der Glättung fest: je größer h desto stärker wird geglättet. Abbildung 2.3 zeigt das Ergebnis einer Berechnung für drei verschiedene Werte von h. Die Bestimmung eines geeigneten Wertes für h ist schwierig, zumal in verschiedenen Bereichen

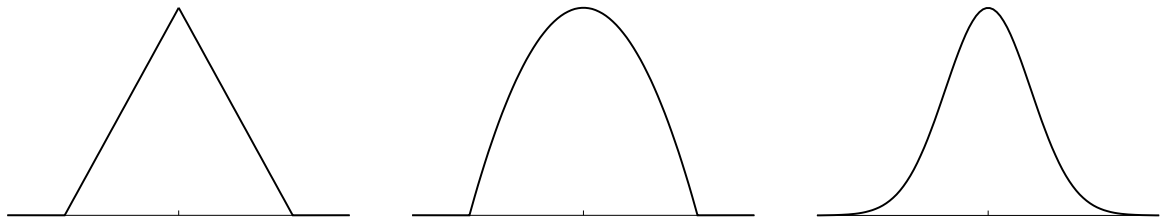

Abbildung 2.2: Drei typische Kernfunktionen, die für nichtparametrische Dichteschätzung genutzt werden: Dreieckskern (links), Epanechnikov-Kern (Mitte) und Gaußkern (rechts).

des Datenraums unterschiedliche Werte für h zu bevorzugen wären: große Glättung in den dünn besetzten Bereichen und geringere Glättung in den dicht besetzten Bereichen.

Ein weiterer Nachteil kernbasierter Dichteschätzer ist die Abhängigkeit des Rechenaufwandes zum Auswerten der Dichte von der Größe der Trainingsstichprobe: für jede Auswertung und jedes Element der Trainingsmenge muss die Kernfunktion einmal ausgewertet werden. Das bedeutet insbesondere, dass stets alle Trainingsdaten verfügbar sein müssen. Daher wachsen sowohl der Speicheraufwand als auch der Rechenaufwand zur Auswertung der Dichte linear mit der Größe der Trainingsmenge.

Neben der Interpretation Kernbasierter Schätzer als geglättete Stichproben lassen sie sich auch als einfache Formen von Mischverteilungen beschreiben. Gleichung (2.6) summiert die n verschiedenen Dichten $x \mapsto \frac{1}{h^d} K(\frac{x-x_i}{h})$ auf und gewichtet jeden der Summanden mit $\frac{1}{n}$. Die Gesamtverteilung ist also zusammengesetzt aus n verschiedenen Einzeldichten, jede mit dem selben Faktor gewichtet. Eine verallgemeinerte Form derartiger Mischverteilungen wird in Abschnitt 2.4 beschrieben.

Der Hypothesenraum für Kernschätzer mit Kernfunktion K zu gegebener Stichprobe ist folglich die Menge $\mathcal{H} = \left\{x \mapsto \frac{1}{n \cdot h^d} \sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) | h \in \mathbb{R}_{>0}\right\}$. Man beachte, dass der Hypothesenraum im Unterschied zu parametrisierten Modellen abhängig von der Trainingsstichprobe ist.

Eine Verallgemeinerung der Kernschätzer stellen die Support-Vector Methoden dar.

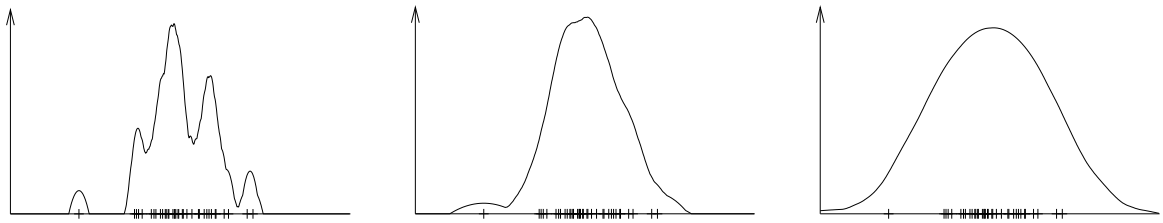

Abbildung 2.3: Kernschätzung für eine univariate Stichprobe (Markierungen auf der x-Achse) mit kleinem (links), mittlerem (Mitte) und großem (rechts) Wert des Glättungsparameters h. Für zu großes h sind keine Details mehr erkennbar, für zu kleines h neigt der Schätzer zum Overfitting. In diesen drei Beispielen wurde der Epanechnikov-Kern verwendet.

[Vapnik 00] stellt ein Verfahren vor, das eine Dichte der verallgemeinerten Form berechnet[2]:

$$x \mapsto \sum_{i=1}^{n} \left(w_i \cdot K \left(\frac{x - x_i}{h} \right) \right) \tag{2.7}$$

Dabei ist zu beachten, dass die einzelnen Trainingsmuster x_i nun individuell mit w_i gewichtet werden. Insbesondere sind viele der $w_i = 0$, d.h. die Summation reduziert sich auf eine kleinere Anzahl Trainingsdaten x_i mit $w_i > 0$. Diese Stützstellen werden auch *Support vectors* genannt. Der Lernalgorithmus bestimmt die Gewichte w_i aus den Daten. K und h werden als gegeben vorausgesetzt.

2.4 Mischverteilungen

2.4.1 Motivation

Die bisher vorgestellten Methoden zur Dichteschätzung sind beide auf bestimmte Einsatzbereiche beschränkt: parametrisierte Methoden eignen sich zwar sehr gut, wenn eine Verteilungsannahme bekannt ist, sie sind aber nicht in der Lage, unbekannte Daten aus komplexeren Verteilungen zu beschreiben. Nichtparametrische Verfahren dagegen treffen zwar keine Annahmen über die Verteilung der Daten, sind aber schwierig zu handhaben im Hinblick auf die Einstellung des Glättungsparameters und damit die Vermeidung von Overfitting.

Eine Möglichkeit, beide Vorgehensweisen miteinander zu verbinden und damit auch bei unbekannter Verteilungsannahme ein parametrisiertes Modell zu bestimmen, bieten die sogenannten *Mischverteilungen* [Titterington 85, McLachlan 88, McLachlan 00].

Mischverteilungsmodelle erweitern die Modellierung aus Gleichung (2.6) und (2.7), indem beliebige Dichten und Gewichte zugelassen werden:

Definition (Mischverteilung) Seien $k \in \mathbb{N}$, $p_1, \ldots, p_k$ Wahrscheinlichkeitsdichten über dem selben Raum, $w_1, \ldots, w_k \in [0, 1]$ mit $\sum_{j=1}^{n} w_j = 1$, dann ist $p : x \mapsto \sum_{j=1}^{k} (w_j \cdot p_j(x))$ die Dichte einer Mischverteilung.

Mischverteilungen sind also gewichtete Summen von einzelnen Wahrscheinlichkeitsverteilungen. Um zu beweisen, dass diese Definition tatsächlich eine Wahrscheinlichkeitsdichte definiert, ist zu zeigen:

(i) $p(x) \geq 0$ für alle x

(ii) $\int p(x)dx = 1$

Beide Forderungen sind erfüllt, wie sich durch nachrechnen leicht zeigen lässt. Dabei spielt die Art der Einzeldichten p_j keine Rolle. Die Einzeldichten werden im Folgenden als *Komponenten* der Mischverteilung bezeichnet.

In Erweiterung von Gleichung (2.7) können die Komponenten p_j auch parametrisiert sein. In dieser Arbeit soll nur der Fall behandelt werden, in dem jede Komponente eigene

[2] Die Originalformulierung in [Vapnik 00] geht sogar noch weiter und verwendet Kernfunktionen mit zwei Argumenten $K(x, x_i)$; die für die Praxis relevanten Kernfunktionen (insbesondere Gaußkerne) lassen sich allerdings auf solche mit einem Parameter $K(x - x_i)$ reduzieren.

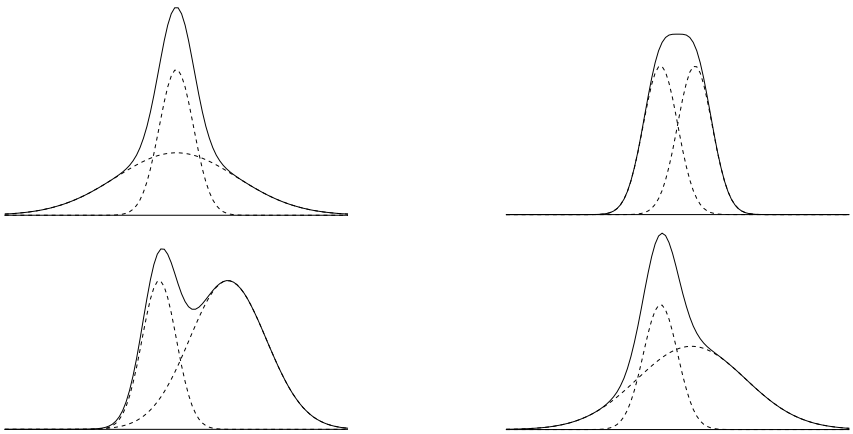

Abbildung 2.4: Beispiele für univariate Mischverteilungen mit zwei normalverteilten Komponenten. Dargestellt ist die Dichte der Mischverteilung (durchgezogene Linie) sowie die gewichteten Dichten der einzelnen Komponenten (gestrichelte Linien).

Parameter besitzt, unabhängig von den anderen Komponenten. Gemeinsame Parameter für alle Komponenten sollen an dieser Stelle also nicht behandelt werden. Bezeichnet ϑ_j die Parameter der Komponente p_j, so ergibt sich die Mischverteilung zu:

$$p|k, \vartheta_1, \ldots, \vartheta_k, w_1, \ldots, w_k : x \mapsto \sum_{j=1}^{k} (w_j \cdot p_j(x|\vartheta_j)) \tag{2.8}$$

Der gesamte Parametervektor der Mischverteilung umfasst $\vartheta = (w_1, \ldots, w_k, \vartheta_1, \ldots, \vartheta_k)$ Die Größe k bestimmt die Anzahl Komponenten und damit die Flexibilität der Mischverteilung. Sie stellt damit einen Parameter auf einer höheren Stufe dar.

Gegenüber den einfachen parametrisierten Modellen aus Abschnitt 2.2.1 weisen Mischverteilungmodelle eine wesentlich größere Anpassungsfähigkeit auf. Dadurch wird es möglich, auch komplexere Verteilungen gut zu approximieren. Andererseits steigt die Gefahr des Overfitting durch den größeren Hypothesenraum.

Einige Beispiele für Mischverteilungen bestehend aus zwei normalverteilten Komponenten zeigt Abbildung 2.4. Es wird deutlich, dass bereits mit zwei Komponenten eine wesentlich größere Vielfalt an Verteilungen modelliert werden kann als mit einer einzelnen Normalverteilung, unter anderem schiefe und langschwänzige Verteilungen.

Im Vergleich der Formeln (2.6) und (2.8) zeigen sich die Unterschiede zwischen Parzen-Fenster-Technik und Mischverteilungen:

- die Anzahl Komponenten ist unabhängig von der Größe der Trainingsmenge
- jede Komponente kann eine andere Dichte besitzen
- die Varianzen verschiedener Komponenten können unterschiedlich sein
- die Lage einer Komponente (ihr Erwartungswert) ist unabhängig von den Trainingsdaten

Modell	Vorteile	Nachteile
Parametrisierte Schätzer	• kompakte Darstellung • schnelles Lernen • schnelle Auswertung • wenig Overfitting	• Auswahl eines geeigneten Modells erforderlich • geringe Anpassungsfähigkeit der Modelle
Kernschätzer	• kein Lernen erforderlich • einfaches Prinzip • Anpassungsfähigkeit	• großer Speicherbedarf • langsame Auswertung • Glättungsparameter schwierig einzustellen • Overfitting möglich
Mischverteilungen	• kompakte Darstellung • schnelle Auswertung • Anpassungsfähigkeit	• aufwändigere Lernverfahren • Modellgröße muss bestimmt werden • Overfitting möglich

Tabelle 2.1: Zusammenstellung der Vor- und Nachteile der beschriebenen Modellklassen für die Dichteschätzung.

Durch die größere Flexibilität der einzelnen Komponenten haben Mischverteilungen i.d.R. deutlich weniger Komponenten als die Anzahl Trainingsdaten. Die Hypothesenmenge ist unabhängig von den Trainingsdaten. Ein Vergleich der Eigenschaften von Parzen-Fenster-Techniken, einfachen parametrisierten Modellen und Mischverteilungen ist in Tabelle 2.1 zu finden.

2.4.2 Gauß-Mischverteilungen

Eine häufig verwendete Form der Mischverteilungsmodelle ist die der Gaußmischverteilungen (GMM). Hierbei sind alle Komponenten der Mischverteilung normalverteilte Dichten, parametrisiert durch Mittelwert und Varianz. Im univariaten Fall ist die Dichte eines GMM gegeben durch:

$$p|k, \mu_1, \ldots, \mu_k, \sigma_1^2, \ldots, \sigma_k^2, w_1, \ldots, w_k : x \mapsto \sum_{j=1}^{k} (w_j \cdot \varphi_{\mu_j, \sigma_j^2}(x)) \tag{2.9}$$

Die Parameter μ_j und σ_j^2 parametrisieren die j-te Komponente des GMM, w_j sind wiederum die Gewichtungen der einzelnen Komponenten. Im multivariaten Fall sind die Komponenten entsprechend durch den Mittelwertvektor μ_j und die Kovarianzmatrix Σ_j parametrisiert:

$$p|k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k, w_1, \ldots, w_k : x \mapsto \sum_{j=1}^{k} (w_j \cdot \varphi_{\mu_j, \Sigma_j}(x)) \tag{2.10}$$

Lernverfahren für GMMs werden in Kapitel 4 und 5 diskutiert.

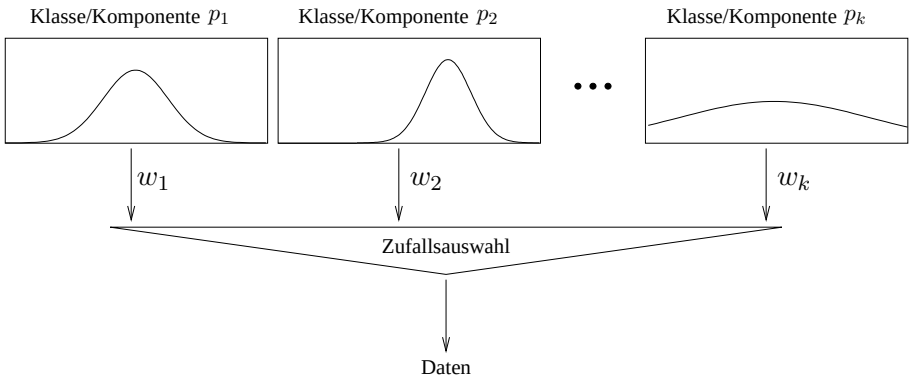

Abbildung 2.5: Generatives Modell einer Mischverteilung als Klassenmodell mit k einzelnen Klassen, aus denen die Muster mit bestimmten Wahrscheinlichkeiten w_j ausgewählt werden

2.4.3 Generatives Modell für Mischverteilungen

Die große Flexibilität von Mischverteilungen macht eine direkte Analyse der Verteilungen schwierig. Allerdings lassen sich Mischverteilungsmodelle aus einfacheren Verteilungen kombinieren.

Man nehme an, die Daten x_i entstammen verschiedenen Datenquellen (Klassen), beschrieben durch die jeweiligen erzeugenden Verteilungen $p_1, \ldots, p_k$. Die j-te Datenquelle trage jeweils mit einem Anteil von w_j zu den Daten bei und die Auswahl der Datenquelle erfolge zufällig. z_i sei die Nummer der Datenquelle, aus der das Muster x_i entstamme. Es gilt also: $\mathcal{P}(z_i = j) = w_j$ und $x_i|z_i \sim p_{z_i}$.

Das so beschriebene Modell ist das einer Klassenverteilung. Die Daten zerfallen in k Klassen, jede mit einer Auswahlwahrscheinlichkeit w_j und einer Verteilung p_j versehen. Die Daten dieses Modells sind Paare (x, z), wobei z die Nummer der Klasse angibt.

Die Verteilung der x ist damit die Randverteilung der Klassenverteilung.

$$\mathcal{P}(x) = \sum_{z=1}^{k} \mathcal{P}(x, z) = \sum_{z=1}^{k} \mathcal{P}(x|z)\mathcal{P}(z) = \sum_{z=1}^{k} \mathcal{P}(x|z)w_z \tag{2.11}$$

Da die Verteilung $\mathcal{P}(\cdot|z)$ durch ihre Dichte p_z gegeben ist, ergibt sich für die Randdichte der x:

$$p(x) = \sum_{z=1}^{k} p_z(x)w_z \tag{2.12}$$

Offensichtlich ist Gleichung (2.12) identisch mit der Dichte einer Mischverteilung. Jede Mischverteilung lässt sich somit auf eine Klassenverteilung zurückführen. Allerdings fehlen bei einer Mischverteilung die Variablen z, die die Klassenzugehörigkeit eines Musters x angeben. Abbildung 2.5 verdeutlicht den Aufbau einer Mischverteilung im Sinne einer Klassenverteilung.

Kapitel 3

MCMC-Algorithmen

3.1 Motivation

Markov-Chain Monte-Carlo Methoden (MCMC) sind eine Familie von Verfahren zur Erzeugung von Pseudozufallsgrößen aus einer gegebenen Verteilung. Übersichten über diese Algorithmen finden sich in [Gilks 96, Robert 99, Andrieu 03, MacKay 03]. MCMC-Verfahren werden teilweise auch als *Bayesssches Sampling* bezeichnet, da sie häufig, aber nicht ausschließlich, im Zusammenhang mit Bayesscher Modellierung verwendet werden.

MCMC-Verfahren sind vor allem dann geeignet, wenn die gegebene Verteilung zu kompliziert ist, als dass aus ihr durch direktes Auswerten der Verteilungs- oder Dichtefunktion eine Zufallsstichprobe erzeugt werden könnte, z.B. durch die Methode der inversen Verteilungsfunktion [Knuth 69] oder dem Accept/Reject-Sampling [Ripley 87, MacKay 03]. Im Gegensatz dazu bestimmen MCMC-Verfahren einen Übergangskern einer Markov-Kette, deren stationäre Verteilung die gegebene Zielverteilung ist. Durch Erzeugen einer Zufallsfolge aus dieser Markov-Kette erhält man eine abhängige Stichprobe der Zielverteilung.

3.2 Eigenschaften von Markov-Ketten

Markov-Ketten sind eine stochastische Struktur zur Beschreibung sequentieller, stochastisch abhängiger Zufallsfolgen. Es soll hier zunächst der einfachere Fall endlicher Zustandsräume betrachtet werden. Ferner beschränkt sich diese Abhandlung auf die Betrachtung homogener Markov-Ketten.

Definition (Markov-Kette) Eine Markov-Kette ist eine Folge von Zustandsvariablen $(X_t)_{t=0}^{\infty}$ über dem selben Zustandsraum $S = \{1, \ldots, n\}$ sowie eine Matrix von Übergangswahrscheinlichkeiten $P = (p_{ij})$ mit $p_{ij} \geq 0$ für alle $i, j \in \{1, \ldots, n\}$ und $\sum_{j=1}^{n} p_{ij} = 1$ für alle $i \in \{1, \ldots, n\}$, so dass für alle $t \in \mathbb{N}_0$ gilt:

$$\mathcal{P}(X_{t+1} = s_{t+1} | X_t = s_t, X_{t-1} = s_{t-1}, \ldots, X_0 = s_0) = \mathcal{P}(X_{t+1} = s_{t+1} | X_t = s_t) = p_{s_t s_{t+1}} \tag{3.1}$$

Die Realisierung einer Markov-Kette stellt eine Folge von Zuständen aus S dar. Sie setzt die Wahl eines initialen Zustandes s_0 voraus, der gemäß einer Anfangsverteilung auf dem Zustandsraum gezogen wird.

Nun sollen einige wichtige Eigenschaften von Markov-Ketten eingeführt werden. Ziel ist es, solche Markov-Ketten zu charakterisieren, die unabhängig von der Anfangsverteilung stets gegen eine bestimmte Grenzverteilung konvergieren. Diese Eigenschaft einer Markov-Kette wird als *Ergodizität* bezeichnet.

Definition (Ergodizität) Sei $P^\infty := \lim_{k\to\infty} P^k$ und $p_{ij}^{(\infty)}$ die Einträge der Matrix P^∞. Eine Markov-Kette heißt ergodisch, wenn

(i) P^∞ existiert

(ii) Alle Zeilen von P^∞ sind identisch

(iii) Alle Einträge von P^∞ sind positiv (also nicht Null)

(iv) Die Zeilen von P^∞ bilden eine Wahrscheinlichkeitsverteilung, d.h. $\sum_{j=1}^{n} p_{1j}^{(\infty)} = 1$

Die Eigenschaft der Ergodizität einer Markov-Kette lässt sich auf zwei andere Eigenschaften zurückführen, nämlich *Irreduzibilität* und *Aperiodizität*, die leichter fassbar sind. Irreduzibilität bedeutet, dass von jedem Zustand aus jeder andere Zustand in ein oder mehreren Schritten erreichbar ist, während Aperiodizität bedeutet, dass die Markov-Kette sich nicht in Zyklen fangen kann und dadurch das Auftreten bestimmter Zustände zu bestimmten Zeitpunkten ausgeschlossen ist.

Definition (Irreduzibilität) Ein Zustand s' heißt erreichbar von einem Zustand s, falls es Zustände $s_1, \ldots, s_k$ gibt mit $s_1 = s$, $s' = s_k$ und $p_{s_i s_{i+1}} > 0$ für alle $i \in \{1, \ldots, k-1\}$. Das heißt, es gibt einen Pfad von s zu s', der mit einer Wahrscheinlichkeit größer Null auftritt. Eine Markov-Kette heißt irreduzibel, falls jeder Zustand von jedem anderen Zustand aus erreichbar ist.

Definition (Aperiodizität) Die Periode d_s eines Zustandes s ist definiert als[1]:

$$\begin{aligned} d_s := ggT\{k \in \mathbb{N} |\ &\text{es gibt } s_1, \ldots, s_k \text{ mit } s = s_k \text{ und } p_{ss_1} > 0 \\ &\text{und } p_{s_i s_{i+1}} > 0 \text{ für alle } i \in \{1, \ldots, k-1\}\} \end{aligned} \tag{3.2}$$

Eine Markov-Kette heißt aperiodisch, wenn alle Zustände s die Periode 1 besitzen: $d_s = 1$

Definition (Stationarität) Eine Wahrscheinlichkeitsverteilung $\alpha = (\alpha_1, \ldots, \alpha_n)$ auf dem Zustandsraum einer Markov-Kette heißt stationär, wenn gilt:

$$\alpha = \alpha \cdot P \tag{3.3}$$

Mit diesen Definitionen lassen sich die Konvergenzeigenschaften von Markov-Ketten in folgendem Satz zusammenfassen:

[1] ggT: größter gemeinsamer Teiler

Satz (Konvergenz von Markov-Ketten)

(i) Eine Markov-Kette ist ergodisch genau dann wenn sie irreduzibel und aperiodisch ist

(ii) Eine ergodische Markov-Kette besitzt genau eine stationäre Verteilung

(iii) Die stationäre Verteilung einer ergodischen Markov-Kette ist gegeben durch die Matrix P^∞

(iv) Unabhängig von der Anfangsverteilung konvergiert jede ergodische Markov-Kette gegen ihre stationäre Verteilung

Der Beweis zu diesem Satz findet sich in [Feller 68, Schmidt 03]. Teil (i) des Satzes liefert uns eine einfache Charakterisierung ergodischer Markov-Ketten. Ergodische Markov-Ketten sind nach Teil (ii), (iii) und (iv) des Satzes in gewisser Weise als „gutmütig“ zu bezeichnen, da sie stets gegen eine eindeutige Verteilung konvergieren, unabhängig von ihrer Initialisierung. Die stationäre Verteilung selbst ist über die Fixpunktgleichung (3.3) gegeben.

Die Übertragung der Definition von Markov-Ketten auf kontinuierliche Zustandsräume erfolgt sinngemäß, ist aber sehr technisch und wird daher hier nicht im Detail dargestellt. Ausführliche Beschreibungen finden sich unter anderem in [Gilks 96, Robert 99].

Die Übergangsmatrix P wird im Fall kontinuierlicher Zustandsräume durch einen Übergangskern $T(s'|s)$ ersetzt, der die Übergangswahrscheinlichkeiten modelliert. Stationäre Verteilungen p sind nun die Eigenfunktionen p des Übergangskerns [Andrieu 03], die folgende Gleichung erfüllen:

$$p(s') = \int p(s)T(s'|s)ds \tag{3.4}$$

Die Konvergenzeigenschaften ergodischer Markov-Ketten bleiben auch im kontinuierlichen Fall erhalten (siehe [Robert 99]).

Eine sehr nützliche Eigenschaft der Übergangskerne von Markov-Ketten – egal ob im diskreten oder kontinuierlichen Zustandsraum – ist die Möglichkeit, verschiedene Übergangskerne durch Hintereinanderausführung miteinander zu kombinieren. Damit wird es möglich, komplizierte Übergangskerne als Verkettung mehrerer einfacher Übergänge zu konstruieren.

Satz (Verkettung von Übergangskernen) Seien T_1 und T_2 Übergangskerne mit stationärer Verteilung p. Dann ist $T(s'|s) := \int T_2(s'|s'') \cdot T_1(s''|s)ds''$ der Übergangskern einer Markov-Kette mit stationärer Verteilung p.

Der Beweis erfolgt durch Einsetzen:

$$\begin{aligned}
\int T(s'|s)p(s)ds &= \int\int T_2(s'|s'') \cdot T_1(s''|s)ds''p(s)ds \\
&= \int T_2(s'|s'') \left(\int T_1(s''|s)p(s)ds \right) ds'' \\
&= \int T_2(s'|s'')p(s'')ds'' \\
&= p(s')
\end{aligned} \tag{3.5}$$

Neben der Hintereinanderausführung von Übergangskernen ist auch das Bilden von Mischkernen möglich:

Satz (Mischen von Übergangskernen) Seien T_1 und T_2 Übergangskerne mit stationärer Verteilung p und $w_1, w_2 \geq 0$ mit $w_1 + w_2 = 1$. Dann ist $T(s'|s) := w_1 \cdot T_1(s'|s) + w_2 \cdot T_2(s'|s)$ der Übergangskern einer Markov-Kette mit stationärer Verteilung p.

Der Beweis erfolgt wiederum durch Einsetzen:

$$\begin{aligned} \int T(s'|s)p(s)ds &= \int (w_1 \cdot T_1(s'|s) + w_2 \cdot T_2(s'|s))p(s)ds \\ &= w_1 \int T_1(s'|s)p(s)ds + w_2 \int T_2(s'|s)p(s)ds \\ &= w_1 p(s') + w_2 p(s') \\ &= p(s') \end{aligned} \tag{3.6}$$

Ein weiteres Hilfsmittel, das die Konstruktion geeigneter Übergangskerne erleichtert, ist die Gleichgewichtsbedingung[2]. Sie liefert eine hinreichende Bedingung dafür, dass ein Übergangskern T eine stationäre Verteilung p besitzt:

Satz (Gleichgewichtsbedingung) Sei T ein Übergangskern und p eine Verteilung. Wenn

$$T(s'|s)p(s) = T(s|s')p(s') \tag{3.7}$$

für alle s, s' gilt, dann ist p stationäre Verteilung von T.

Der Beweis erfolgt durch Einsetzen:

$$\begin{aligned} \int T(s'|s)p(s)ds &= \int T(s|s')p(s')ds \\ &= \int T(s|s')\,ds\,p(s') \\ &= p(s') \end{aligned} \tag{3.8}$$

3.3 Metropolis-Hastings-Algorithmus

Einer der allgemeinsten MCMC-Algorithmen ist das Verfahren nach Metropolis und Hastings [Metropolis 53, Hastings 70]. Die meisten anderen Verfahren stellen eine Spezialisierung hiervon dar.

Die Idee des Metropolis-Hastings-Algorithmus' ist es, die Erzeugung eines Nachfolgepunktes für die Markov-Kette von der Zielverteilung unabhängig zu machen und in einem zweiten Schritt den Nachfolgezustand entweder zu akzeptieren oder zu verwerfen, ähnlich dem Accept/Reject-Sampling-Verfahren [Ripley 87].

Sei p die Dichte der Zielverteilung und $q(\cdot|s)$ die Dichte einer beliebigen bedingten Suchverteilung. In jeder Iteration des Metropolis-Hastings Verfahrens wird zunächst ein möglicher Nachfolgepunkt s' gemäß der Suchverteilung gezogen und dieser anschließend mit einer bestimmten Akzeptanzwahrscheinlichkeit entweder als Nachfolgezustand gewählt oder aber verworfen. In letzterem Fall bleibt der bisherige Zustand auch als Nachfolgezustand erhalten. Die Akzeptanzwahrscheinlichkeit ist so gewählt, dass die Gleichgewichtsbedingung (3.7) erfüllt wird. Abbildung 3.1 dokumentiert den Algorithmus in Pseudocode.

[2] englisch: *detailed balance condition*

```
1.   wähle initiales s^(0)
2.   wiederhole für t = 0, 1, 2, ...
3.      sample s' ~ q(·|s^(t))
4.      sample u ~ U(0,1)
5.      wenn u < p(s')q(s^(t)|s') / p(s^(t))q(s'|s^(t))
6.         setze s^(t+1) := s'
7.      sonst
8.         setze s^(t+1) := s^(t)
```

Abbildung 3.1: Metropolis-Hastings Algorithmus mit Zieldichte $p(\cdot)$ und Suchverteilung $q(\cdot|s^{(t)})$.

Der Übergangskern der auf diese Weise erzeugten Markov-Kette lautet:

$$\begin{aligned} T_{MH}(s^{(t+1)}|s^{(t)}) &= q(s^{(t+1)}|s^{(t)})A(s^{(t+1)}|s^{(t)}) \\ &\quad + \delta(s^{(t+1)} - s^{(t)}) \int q(s'|s^{(t)})(1 - A(s'|s^{(t)}))ds' \end{aligned} \tag{3.9}$$

mit

$$A(z|y) = \min\{1,\ \frac{p(z)q(y|z)}{p(y)q(z|y)}\}$$

p ist stationäre Dichte der Markov-Kette, denn es gilt:

$$\begin{aligned} \int T_{MH}(s^{(t+1)}|s^{(t)})p(s^{(t)})ds^{(t)} &= \int q(s^{(t+1)}|s^{(t)})A(s^{(t+1)}|s^{(t)})p(s^{(t)})ds^{(t)} \\ &\quad + \int \delta(s^{(t+1)} - s^{(t)}) \int q(s'|s^{(t)})(1 - A(s'|s^{(t)}))p(s^{(t)})ds'\ ds^{(t)} \\ &= \int \min\{p(s^{(t)})q(s^{(t+1)}|s^{(t)}),\ p(s^{(t+1)})q(s^{(t)}|s^{(t+1)})\}ds^{(t)} \\ &\quad + \int q(s'|s^{(t+1)})(1 - A(s'|s^{(t+1)}))p(s^{(t+1)})ds' \\ &= \int \min\{p(s^{(t)})q(s^{(t+1)}|s^{(t)}),\ p(s^{(t+1)})q(s^{(t)}|s^{(t+1)})\}ds^{(t)} \\ &\quad + \int p(s^{(t+1)})q(s'|s^{(t+1)})ds' \\ &\quad - \int p(s^{(t+1)})A(s'|s^{(t+1)})q(s'|s^{(t+1)})ds' \\ &= \int \min\{p(s^{(t)})q(s^{(t+1)}|s^{(t)}),\ p(s^{(t+1)})q(s^{(t)}|s^{(t+1)})\}ds^{(t)} \\ &\quad + p(s^{(t+1)}) \\ &\quad - \int \min\{p(s')q(s^{(t+1)}|s'),\ p(s^{(t+1)})q(s'|s^{(t+1)})\}ds' \\ &= p(s^{(t+1)}) \end{aligned} \tag{3.10}$$

Die Konvergenz des Algorithmus' hängt entscheidend von der Wahl der Suchverteilung q ab. q muss so gewählt werden, dass die resultierende Markov-Kette aperiodisch und irreduzibel ist. Bei entsprechender Wahl von q ist p die einzige stationäre Verteilung der Markov-Kette.

3.4 Gibbs-Sampler

Der Gibbs-Sampler [Geman 84] stellt eine spezialisierte Form des Metropolis-Hastings Algorithmus' dar. Er wird verwendet, wenn die Zustände mehrdimensionale Vektoren sind, d.h. $s = (s_1, \ldots, s_d)$. Die Idee ist, jede Komponente des Zustandes einzeln neu zu samplen, während die restlichen Komponenten unverändert bleiben.

Betrachte dazu für eine Komponente $j \in \{1, \ldots, d\}$ die folgende Suchverteilung:

$$q(s'|s) = \begin{cases} p(s'_j|s_{-j}) & \text{falls } s'_{-j} = s_{-j} \\ 0 & \text{sonst} \end{cases} \tag{3.11}$$

wobei $s_{-j} := (s_1, \ldots, s_{j-1}, s_{j+1}, \ldots, s_d)$ den Vektor der Komponenten ohne die j-te Komponente sowie $p(s'_j|s_{-j})$ die bedingte Verteilung der j-ten Komponente gegeben die restlichen Komponenten bezeichnet. Die Akzeptanzwahrscheinlichkeit des Metropolis-Hastings Verfahrens ist mit dieser Suchverteilung wegen $s^{(t)}_{-j} = s'_{-j}$:

$$\begin{aligned} A(s'|s^{(t)}) &= \min\{1, \frac{p(s')q(s^{(t)}|s')}{p(s^{(t)})q(s'|s^{(t)})}\} \\ &= \min\{1, \frac{p(s')p(s^{(t)}_j|s'_{-j})}{p(s^{(t)})p(s'_j|s^{(t)}_{-j})}\} \\ &= \min\{1, \frac{p(s'_j|s'_{-j})p(s'_{-j})p(s^{(t)}_j|s^{(t)}_{-j})}{p(s^{(t)}_j|s^{(t)}_{-j})p(s^{(t)}_{-j})p(s'_j|s'_{-j})}\} \\ &= \min\{1, \frac{p(s'_{-j})}{p(s^{(t)}_{-j})}\} \\ &= 1 \end{aligned} \tag{3.12}$$

Die Suchverteilung (3.11) kann offensichtlich nicht irreduzibel sein, da nur jeweils eine Komponente des Zustandsvektors verändert wird. Dieses Problem kann behoben werden, indem aus der Suchverteilung mehrfach hintereinander für unterschiedliche Komponenten j gesamplet wird. Mittels einmaligen Durchlaufens aller Komponenten $j = 1, \ldots, d$ werden alle Komponenten genau einmal neu gezogen. Der Übergangskern des Gibbs-Samplers ergibt sich durch Hintereinanderausführung der einzelnen Übergangsvorschriften. Der Übergang von $s^{(t)}$ zu $s^{(t+1)}$ erfordert also die Ausführung von d Einzelübergängen. Aufgrund des Satzes über die Verkettung von Übergangskernen ist die Zielverteilung $p(\cdot)$ auch für den durch Hintereinanderausführung der Einzelübergänge gebildeten Übergangskern stationär.

Abbildung 3.2 gibt den Algorithmus in Pseudocode wieder.

Gemäß den Vorüberlegungen bildet die Folge der $\left(s^{(t)}\right)_{t=0}^{\infty}$ wiederum eine Markov-Kette mit stationärer Verteilung $p(\cdot)$. Aperiodizität und Irreduzibilität der Markov-Kette und

1. wähle initiales $(s_1^{(0)}, \ldots, s_d^{(0)})$
2. wiederhole für $t = 0, 1, 2, \ldots$
3. sample $s_1^{(t+1)} \sim p(s_1 | s_2^{(t)}, s_3^{(t)}, \ldots, s_d^{(t)})$
4. sample $s_2^{(t+1)} \sim p(s_2 | s_1^{(t+1)}, s_3^{(t)}, \ldots, s_d^{(t)})$
5. $\vdots$
6. sample $s_j^{(t+1)} \sim p(s_j | s_1^{(t+1)}, \ldots, s_{j-1}^{(t+1)}, s_{j+1}^{(t)}, \ldots, s_d^{(t)})$
7. $\vdots$
8. sample $s_d^{(t+1)} \sim p(s_j | s_1^{(t+1)}, s_2^{(t+1)}, \ldots, s_{d-1}^{(t)})$

Abbildung 3.2: Gibbs-Sampler mit Zieldichte $p(\cdot)$

damit die Eindeutigkeit der stationären Verteilung hängen wiederum von den Eigenschaften der Suchverteilungen, also in diesem Fall den bedingten Verteilungen $p(s_j|s_{-j})$, ab.

Man beachte, dass für das Gibbs-Sampling nur die bedingten Verteilungen $p(s_j|s_{-j})$ benötigt werden, nicht die eigentliche Zielverteilung $p(\cdot)$. Es ist also auch dann anwendbar, wenn die Zielverteilung selbst nicht explizit gegeben ist, sondern nur ihre bedingten Verteilungen bekannt sind.

Ein Spezialfall des Gibbs-Sampling stellt die Situation dar, in denen $d = 2$ ist, d.h. der Zustandsvektor wird in zwei Teilvektoren geteilt. Motiviert von Aufgabenstellungen mit unvollständigen Daten wird dieser Fall auch als *Data Augmentation*[Tanner 87] bezeichnet. Anders als im allgemeinen Fall bilden hierbei auch die Folgen der Einzelkomponenten $\left(s_1^{(t)}\right)_{t=1}^{\infty}$ und $\left(s_2^{(t)}\right)_{t=1}^{\infty}$ Markov-Ketten. Eine weiterführende Betrachtung von Data Augmentation im Zusammenhang mit Mischverteilungen erfolgt in Abschnitt 4.2.

3.5 Reversible jump MCMC-Algorithmen

In den vorangegangenen Abschnitten wurden MCMC-Algorithmen vorgestellt, die über einem einzelnen Zustandsraum S gearbeitet haben. Nun sollen Verfahren beschrieben werden, die zwischen verschiedenen Zustandsräumen unterschiedlicher Dimensionalität hin- und herspringen können. Dies ermöglicht beispielsweise die gemeinsame Betrachtung unterschiedlich großer Modelle oder Modelle unterschiedlicher Bauart. Die hier gewählte Darstellung orientiert sich an [Green 95]. Zunächst soll die Konstruktion eines Übergangskerns zum Springen zwischen zwei Zustandsräumen S_1 und S_2 beschrieben werden. Wir betrachten hier nur den Fall kontinuierlicher Zustandsräume.

Seien $S_1 = \mathbb{R}^{n_1}$ und $S_2 = \mathbb{R}^{n_2}$ Zustandsräume und p_1 bzw. p_2 Dichten auf den beiden Mengen, die als Zielverteilung des MCMC-Algorithmus' dienen sollen. Ferner seien π_1 bzw. π_2 die Zielwahrscheinlichkeiten, sich in den Zustandsräumen S_1 bzw. S_2 aufzuhalten.

Zunächst bilden wir einen gemeinsamen Zustandsraum über beide Teilräume wiefolgt:

$$S := (\{1\} \times S_1) \cup (\{2\} \times S_2) \tag{3.13}$$

Jeder Zustand aus S besteht also aus einem Paar (m, s), wobei $m \in \{1, 2\}$ einen der beiden Teil-Zustandsräume bezeichnet und $s \in S_m$ einen Zustand aus dem durch m bezeichneten Teil-Zustandsraum darstellt.

Zwei Übergangskerne T_1 auf S_1 und T_2 auf S_2 können zu einem neuen Übergangskern auf S kombiniert werden. Wenn $p_1(\cdot)$ die stationäre Verteilung von T_1 und $p_2(\cdot)$ die stationäre Verteilung von T_2 ist, dann besitzt der Übergangskern:

$$T((m',s')|(m,s)) = \begin{cases} T_1(s'|s) & \text{falls } m = m' = 1 \\ T_2(s'|s) & \text{falls } m = m' = 2 \\ 0 & \text{sonst} \end{cases} \tag{3.14}$$

die stationäre Verteilung: $\mathcal{P}(m,s) \propto \mathbb{I}_{[s \in S_m]} \cdot \pi_m \cdot p_m(s)$

Der Beweis erfolgt durch Einsetzen:

$$\begin{aligned} &\sum_{m=1}^{2} \int_{s \in S_m} T((m',s')|(m,s)) \cdot p_m(s|m) \cdot \pi_m \, ds \\ &= \int_{s \in S_1} T((m',s')|(1,s)) p_1(s) \, ds \, \pi_1 + \int_{s \in S_2} T((m',s')|(2,s)) p_2(s) \, ds \, \pi_2 \\ &= \mathbb{I}_{[m'=1 \wedge s' \in S_1]} p_1(s') \pi_1 + \mathbb{I}_{[m'=2 \wedge s' \in S_2]} p_2(s') \pi_2 \\ &= \mathbb{I}_{[s' \in S_{m'}]} p_{m'}(s') \pi_{m'} \end{aligned} \tag{3.15}$$

Man beachte, dass die Stationaritätsaussage für beliebige π_1 und π_2 gilt.

Um Übergangskerne konstruieren zu können, die von einem Teil des Zustandsraums in den anderen Teil wechseln können, müssen die beiden Teil-Zustandsräume zunächst in einen weiteren, gemeinsamen Raum eingebettet werden. Dieser ergänzt die Zustandsvektoren um zusätzliche Komponenten auf eine gemeinsame Länge:

$$\begin{aligned} &S' = (\{1\} \times S_1 \times U_1) \cup (\{2\} \times S_2 \times U_2) \\ &\text{mit } k_1, k_2 \in \mathbb{N}_0,\ U_1 = \mathbb{R}^{k_1},\ U_2 = \mathbb{R}^{k_2},\ n_1 + k_1 = n_2 + k_2 \end{aligned} \tag{3.16}$$

Das Ziel der Erweiterung ist es, eine bijektive Abbildung $\tau : S_1 \times U_1 \to S_2 \times U_2$ zu finden, die je zwei Elemente $(1, s_1, u_1)$ und $(2, s_2, u_2)$ mit $\tau(s_1, u_1) = (s_2, u_2)$ des erweiterten Zustandsraums S' miteinander in Beziehung setzt. Die Wahl von U_1, U_2 und τ erfolgt problemspezifisch. Die zusätzlichen Komponenten aus U_1 bzw. U_2 stellen dabei weitere Freiheitsgrade dar, die es erlauben, zu einem $s_1 \in S_1$ mehrere $s_2 \in S_2$ via τ in Beziehung zu setzen. Durch Vorgabe einer Wahrscheinlichkeitsverteilung für die zusätzlichen Komponenten, $u_1 \sim q_1(\cdot|s_1)$ bzw. $u_2 \sim q_2(\cdot|s_2)$ wird damit auch eine Wahrscheinlichkeitsverteilung $\mathcal{P}(s_2|s_1)$ induziert. Ein Übergangskern für den Wechsel des Zustandsraums arbeitet also nach dem Prinzip, das in Abbildung 3.3 dargestellt ist.

Um zu garantieren, dass die gewünschte Zielverteilung stationäre Verteilung der Markov-Kette wird, muss sowohl der Übergang von S_1 nach S_2 möglich sein, als auch der Übergang von S_2 nach S_1. Durch Abstimmung der Akzeptanzwahrscheinlichkeiten $A(s_2, u_2|s_1, u_1)$ bzw. $A(s_1, u_1|s_2, u_2)$ kann die Gleichgewichtsbedingung (3.7) erfüllt werden. Dazu betrachte die Wahrscheinlichkeit für einen Übergang von $(1, s_1, u_1)$ nach $(2, s_2, u_2)$ (mit $\tau(s_1, u_1) = (s_2, u_2)$) sowie zurück:

$$\mathcal{P}((1, s_1, u_1) \curvearrowright (2, s_2, u_2)) \propto \pi_1 \cdot p_1(s_1) \cdot q_1(u_1|s_1) \tag{3.17}$$

$$\mathcal{P}((2, s_2, u_2) \curvearrowright (1, s_1, u_1)) \propto \pi_2 \cdot p_2(s_2) \cdot q_2(u_2|s_2) \tag{3.18}$$

```
1.  Ausgangssituation: gegenwärtiger Zustand (1, s_1)
2.  sample u_1 ~ q_1(·|s_1)
3.  berechne (s_2, u_2) = τ(s_1, u_1)
4.  berechne eine Akzeptanzwahrscheinlichkeit A(s_2, u_2|s_1, u_1)
5.  mit Wahrscheinlichkeit A(s_2, u_2|s_1, u_1)
6.     setze (2, s_2) als neuen Zustand
7.  sonst
8.     setze (1, s_1) als neuen Zustand
9.  „vergesse“ u_1, u_2
```

Abbildung 3.3: Grundaufbau eines Übergangskern, der den Teil-Zustandsraum wechselt. Der Übergang von $(2, s_2)$ nach $(1, s_1)$ erfolgt analog unter Nutzung von τ^{-1}.

Die Gleichungen (3.17) und (3.18) sind zwar über dem selben Raum $\mathbb{R}^{n_1+k_1} = \mathbb{R}^{n_2+k_2}$ definiert, allerdings ist (3.17) eine Gleichung in s_1 und u_1 und (3.18) eine Gleichung in s_2 und u_2. Daher sind auch die Proportionalitätskonstanten der beiden Gleichungen nicht identisch. Um einheitliche Proportionalitätskonstanten zu gewährleisten, muss gemäß der Substitutionsregel für Integrale (siehe z.B. [Heuser 86]) die Gleichung (3.18) mit der Determinante der Jacobimatrix der Substitution τ multipliziert werden. Gleichung (3.18) wird daher ersetzt durch:

$$\begin{gathered}\mathcal{P}((2, s_2, u_2) \curvearrowright (1, s_1, u_1)) \propto \pi_2 \cdot p_2(s_2) \cdot q_2(u_2|s_2) \cdot |\mathcal{J}_\tau(s_1, u_1)| \\ \text{mit} \\ \mathcal{J}_\tau(s_1, u_1) = \frac{\partial \tau(s_1, u_1)}{\partial (s_1, u_1)}\end{gathered} \tag{3.19}$$

Um die Gleichgewichtsbedingung (3.7) zu erfüllen, ergibt sich als Akzeptanzwahrscheinlichkeit für einen Übergang von S_1 nach S_2 bzw. zurück analog zum Metropolis-Hastings-Algorithmus (3.9):

$$A(s_2, u_2|s_1, u_1) = \min\left\{1, \frac{\pi_2 \cdot p_2(s_2) \cdot q_2(u_2|s_2) \cdot |\mathcal{J}_\tau(s_1, u_1)|}{\pi_1 \cdot p_1(s_1) \cdot q_1(u_1|s_1)}\right\} \tag{3.20}$$

$$A(s_1, u_1|s_2, u_2) = \min\left\{1, \frac{\pi_1 \cdot p_1(s_1) \cdot q_1(u_1|s_1)}{\pi_2 \cdot p_2(s_2) \cdot q_2(u_2|s_2) \cdot |\mathcal{J}_\tau(s_1, u_1)|}\right\} \tag{3.21}$$

Der in Abbildung 3.3 angegebene Übergangskern mit der Akzeptanzwahrscheinlichkeit (3.20), ergänzt um den analogen Kern zum Übergang von S_2 nach S_1 mit Akzeptanzwahrscheinlichkeit (3.21) hat damit die vorgegebene Zielverteilung $\mathcal{P}(m, s) \propto \mathbb{I}_{[s \in S_m]} \cdot \pi_m \cdot p_m(s)$. Durch Mischen dieser Zustandswechsel-Kerne mit Übergangskernen der Bauart (3.14) lassen sich für den Fall eines zweigeteilten Zustandsraums bereits ergodische Markov-Ketten konstruieren.

Aufbauend auf der Modellierung für einen in zwei Teile zerfallenden Zustandsraum ergibt sich die Modellierung für Zustandsräume, die in $M \geq 2$ Teile zerfallen. Für jeden Teil-Zustandsraum m sei eine Ziel-Wahrscheinlichkeit π_m vorgegeben, die angibt, wie häufig sich der MCMC-Algorithmus in diesem Teilraum aufhalten soll, sowie eine Dichte $p_m(\cdot)$ als gewünschte Aufenthaltsverteilung innerhalb des m-ten Teilraums.

Der gemeinsame Zustandsraum ergibt sich analog (3.13) zu:

$$S := \bigcup_{m=1}^{M} (\{m\} \times S_m) \tag{3.22}$$

Übergangskerne innerhalb der einzelnen Teil-Zustandsräume lassen sich auch für allgemeines M analog zu (3.14) kombinieren.

Für den Übergang vom Teil-Zustandsraum m in den Teilraum m' kann wiederum eine Konstruktion ähnlich der in Abbildung 3.3 verwendet werden, wobei der Übergang in der Gegenrichtung von m' nach m wiederum mitmodelliert werden sollte. Im Fall $M \geq 3$ lassen sich allerdings die Übergangskerne nicht mehr so einfach gemäß dem Satz aus Abschnitt 3.2 mischen, da unklar ist, was bei Wahl des Übergangskerns für den Wechsel von Teilraum 1 in Teilraum 2 passiert, wenn der aktuelle Zustand aus keinem der beiden Teilräume stammt.

Für diesen Fall sollen die Übergangsmöglichkeiten vom aktuellen Zustand abhängen. Ist der aktuelle Zustand beispielsweise aus dem ersten Teilraum, sollen nur solche Übergänge gewählt werden können, die innerhalb des ersten Teilraums bleiben oder die aus dem ersten Teilraum in einen anderen Teilraum wechseln. Dazu werden zunächst eine Variante der Gleichgewichtsbedingung (3.7) eingeführt und anschließend die Akzeptanzwahrscheinlichkeiten (3.20) und (3.21) angepasst.

Die Gleichgewichtsbedingung (3.7) geht davon aus, dass T der einzig mögliche Übergangskern ist, der in den Zuständen s bzw. s' gewählt werden kann. Dies ist eine zu weitgehende Annahme für den hier betrachteten Fall. Wir erweitern die Modellierung, indem wir in jedem Teil-Zustandsraum m verschiedene Übergangskerne zulassen. Jedem Teil-Zustandraum m seien M Übergangskerne zugeordnet $T_{1|m}, T_{2|m}, \ldots, T_{M|m}$. Der Übergangskern $T_{\mu'|\mu}$ modelliere die Übergänge aus Teil-Zustandsraum μ in den Teilraum μ', d.h. es gelte: $T_{\mu'|\mu}((m', s')|(m, s)) = 0$ falls $m' \neq \mu'$ oder $m \neq \mu$ oder $s' \notin S_{m'}$ oder $s \notin S_m$.

Die Übergangskerne werden mit bestimmten Wahrscheinlichkeiten ausgewählt. Bei einem aktuellen Zustand aus dem Teilraum m sei $w_{m'|m} \geq 0$ die Auswahlwahrscheinlichkeit für den Übergangskern $T_{m'|m}$. Um sicherzustellen, dass nie ein anderer als die gegebenen Übergangskerne $T_{m'|m}$ gewählt wird, gelte $\sum_{m'=1}^{M} w_{m'|m} = 1$.

Unter diesen Voraussetzungen ist ein direkter Übergang von einem Zustand (m, s) zu (m', s') in einem Schritt nur unter Anwendung des Übergangskernes $T_{m'|m}$ möglich, der Rücksprung nur unter Anwendung des Übergangskernes $T_{m|m'}$. Die jeweiligen Kerne werden mit den Wahrscheinlichkeiten $w_{m'|m}$ bzw. $w_{m|m'}$ gewählt. Daraus leitet sich die erweiterte Gleichgewichtsbedingung ab:

$$\pi_m p_m(s) w_{m'|m} T_{m'|m}((m', s')|(m, s)) = \pi_{m'} p_{m'}(s') w_{m|m'} T_{m|m'}((m, s)|(m', s')) \tag{3.23}$$

Gilt diese erweiterte Gleichgewichtsbedingung paarweise für alle Teilräume m und m' und alle Zustände $s \in S_m$, $s' \in S_{m'}$, so ist $\mathcal{P}(m, s) \propto \mathbb{I}_{[s \in S_m]} \pi_m p_m(s)$ stationäre Verteilung

der Markov-Kette. Der Beweis erfolgt durch Einsetzen:

$$\begin{aligned}
&\sum_{m=1}^{M} \int_{s\in S_m} \left(\sum_{\mu=1}^{M} w_{\mu|m} T_{\mu|m}((m',s')|(m,s)) \right) p_m(s)\pi_m \, ds \\
&= \mathbb{I}_{[s'\in S_{m'}]} \sum_{m=1}^{M} \int_{s\in S_m} w_{m'|m} T_{m'|m}((m',s')|(m,s)) p_m(s)\pi_m \, ds \\
&= \mathbb{I}_{[s'\in S_{m'}]} \sum_{m=1}^{M} \int_{s\in S_m} w_{m|m'} T_{m|m'}((m,s)|(m',s')) p_{m'}(s')\pi_{m'} \, ds \\
&= \mathbb{I}_{[s'\in S_{m'}]} \sum_{m=1}^{M} w_{m|m'} p_{m'}(s')\pi_{m'} = \mathbb{I}_{[s'\in S_{m'}]} p_{m'}(s')\pi_{m'}
\end{aligned} \tag{3.24}$$

Damit die Übergangskerne $T_{m'|m}$ und $T_{m|m'}$ die erweiterte Gleichgewichtsbedingung erfüllen, müssen die Auswahlwahrscheinlichkeiten berücksichtigt werden. Dies erfolgt durch Abänderung der Akzeptanzwahrscheinlichkeiten (3.20) und (3.21). Berücksichtigung von $w_{m'|m}$ und $w_{m|m'}$ ergibt:

$$A(s_2,u_2|s_1,u_1) = \min\left\{1, \frac{\pi_2 \cdot p_2(s_2) \cdot w_{1|2} \cdot q_2(u_2|s_2) \cdot |\mathcal{J}_\tau(s_1,u_1)|}{\pi_1 \cdot p_1(s_1) \cdot w_{2|1} \cdot q_1(u_1|s_1)}\right\} \tag{3.25}$$

$$A(s_1,u_1|s_2,u_2) = \min\left\{1, \frac{\pi_1 \cdot p_1(s_1) \cdot w_{2|1} \cdot q_1(u_1|s_1)}{\pi_2 \cdot p_2(s_2) \cdot w_{1|2} \cdot q_2(u_2|s_2) \cdot |\mathcal{J}_\tau(s_1,u_1)|}\right\} \tag{3.26}$$

Die Gleichungen (3.20) und (3.21) ergeben sich als Spezialfälle für $w_{1|2} = w_{2|1}$. Analog den Übergängen zwischen den Teil-Zustandsräumen 1 und 2 lassen sich auch Übergänge zwischen anderen Teilräumen konstruieren.

Beispiel (Reversible-jump Konstruktion) Ein einfaches Beispiel soll die Konstruktion von Reversible-jump MCMC-Verfahren verdeutlichen. Man betrachte zwei Zustandsräume $S_1 = \mathbb{R}$ und $S_2 = \mathbb{R}^2$ mit den Priori-Wahrscheinlichkeiten π_1 und π_2. Die Zielverteilung in S_1 sei eine Gleichverteilung im Intervall $[0,2)$, d.h. $p_1(x) = \begin{cases} \frac{1}{2} & \text{falls } 0 \le x < 2 \\ 0 & \text{sonst} \end{cases}$, die Zielverteilung in S_2 eine Gleichverteilung über dem Einheitsquadrat, d.h. $p_2(x_1,x_2) = \begin{cases} 1 & \text{falls } 0 \le x_1, x_2 < 1 \\ 0 & \text{sonst} \end{cases}$.

Wir konstruieren einen Übergangskern $T_{2|1}$ für den Übergang von S_1 nach S_2 sowie einen inversen Übergang $T_{1|2}$, die mit Wahrscheinlichkeit $w_{2|1}$ bzw. $w_{1|2}$ gewählt werden. Für den Übergang von S_1 nach S_2 wird zunächst eine Hilfsgröße $u \sim U(0,1)$ gleichverteilt gesamplet und anschließend die Abbildung $\tau : (x,u) \mapsto (\frac{x}{2}, u)$ angewendet. Die Jacobimatrix lautet demnach $\mathcal{J}_\tau = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{pmatrix}$ und ihre Determinante folglich $|\mathcal{J}_\tau| = \frac{1}{2}$.

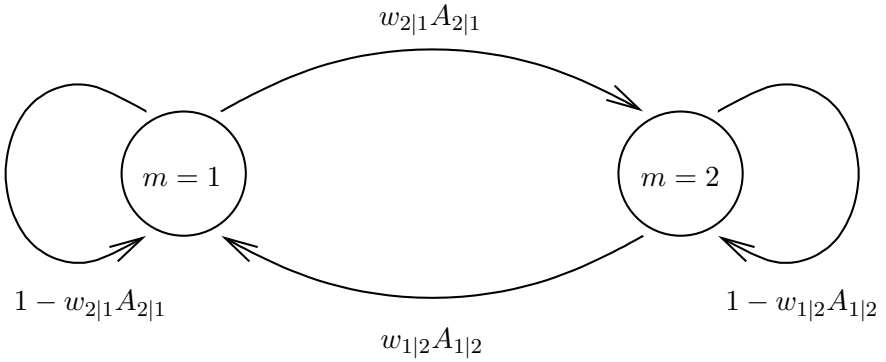

Abbildung 3.4: Beispiel zur Konstruktion von Reversible-jump MCMC-Kernen: Übergangsgraph zwischen den Teilräumen S_1 und S_2 mit Angabe der Übergangswahrscheinlichkeiten.

Damit ergeben sich gemäß (3.25) und (3.26) die Akzeptanzwahrscheinlichkeiten zu:

$$A_{2|1} = \min\left\{1, \frac{\pi_2 \cdot 1 \cdot \frac{1}{2} \cdot w_{1|2}}{\pi_1 \cdot \frac{1}{2} \cdot 1 \cdot w_{2|1}}\right\} = \min\left\{1, \frac{\pi_2}{\pi_1} \cdot \frac{w_{1|2}}{w_{2|1}}\right\} \tag{3.27}$$

$$A_{2|1} = \min\left\{1, \frac{\pi_1 \cdot \frac{1}{2} \cdot 1 \cdot w_{2|1}}{\pi_2 \cdot 1 \cdot \frac{1}{2} \cdot w_{1|2}}\right\} = \min\left\{1, \frac{\pi_1}{\pi_2} \cdot \frac{w_{2|1}}{w_{1|2}}\right\} \tag{3.28}$$

Ergänzt um geeignete Übergangskerne $T_{1|1}$ und $T_{2|2}$ erhält man dann in Bezug auf die Übergänge zwischen S_1 und S_2 eine Markov-Kette, die in Abbildung 3.4 dargestellt ist. Die Übergangsmatrix für Übergänge zwischen S_1 und S_2 lautet also:

$$P = \begin{pmatrix} 1 - \min\{w_{2|1}, \frac{\pi_2}{\pi_1} w_{1|2}\} & \min\{w_{2|1}, \frac{\pi_2}{\pi_1} w_{1|2}\} \\ \min\{w_{1|2}, \frac{\pi_1}{\pi_2} w_{2|1}\} & 1 - \min\{w_{1|2}, \frac{\pi_1}{\pi_2} w_{2|1}\} \end{pmatrix} \tag{3.29}$$

Als stationäre Verteilung ergibt sich:

$$\mathcal{P}(m = 1) = \frac{w_{1|2}A_{1|2}}{w_{1|2}A_{1|2} + w_{2|1}A_{2|1}}, \quad \mathcal{P}(m = 2) = \frac{w_{2|1}A_{2|1}}{w_{1|2}A_{1|2} + w_{2|1}A_{2|1}} \tag{3.30}$$

Sowohl für den Fall $\frac{\pi_1}{\pi_2}\frac{w_{2|1}}{w_{1|2}} \geq 1$ als auch $\frac{\pi_1}{\pi_2}\frac{w_{2|1}}{w_{1|2}} < 1$ vereinfacht sich dieser Ausdruck zu:

$$\mathcal{P}(m = 1) = \pi_1, \quad \mathcal{P}(m = 2) = \pi_2 \tag{3.31}$$

was den vorgegebenen Prioriwahrscheinlichkeiten entspricht.
■

Kapitel 4

Lernverfahren für Mischverteilungen

4.1 Der EM-Algorithmus

4.1.1 Allgemeine Formulierung

Der *Expectation-Maximization*-Algorithmus (EM) ist das am meisten verwendete Verfahren zur Parameterschätzung von Mischverteilungen. Es basiert auf einer recht allgemeinen Formulierung zur Lösung von Problemen mit unvollständigen Trainingsdaten. Das Trainieren von Mischverteilungen kann unter Zuhilfenahme des Marginalisierungstricks (siehe Abschnitt 2.4.3) als derartige Aufgabenstellung interpretiert werden. Daher soll zunächst die allgemeine Form des EM-Algorithmus' vorgestellt werden, die dann im nachfolgenden Abschnitt auf die Verwendung im Zusammenhang mit Gauß-Mischverteilungen hin spezialisiert wird.

Der EM-Algorithmus wurde in seiner allgemeinen Form in [Dempster 77] eingeführt. Weitere Darstellungen finden sich unter anderem in [Schafer 97].

Ausgangsbasis ist das Problem des Lernens aus lückenhaften vektorwertigen Daten. Ziel ist es, die Parameter ϑ einer parametrisierten Verteilungsfamilie zu schätzen. Dazu nehmen wir an, dass jedes Element der Trainingsstichprobe in einen bekannten Anteil x_i und einen fehlenden Anteil z_i zerfällt. x_i und z_i sind Teilvektoren des vollständigen Stichprobenelements. Die Aufteilung in vorhandene und fehlende Anteile kann für jedes Trainingsmuster unterschiedlich sein. Ferner erfülle die Trainingsstichprobe die *Missing at random*-Bedingung, d.h. das Fehlen eines Teils eines Trainingsmusters ist stochastisch unabhängig von dem Wert des fehlenden Teils. Die Daten sind also nicht zensiert. Eine genaue Definition der Missing-at-random Bedingung findet sich in [Schafer 97].

Der EM-Algorithmus fußt auf der Idee, die fehlenden Teile der Daten künstlich auszufüllen und die Parameter aus den vervollständigten Daten zu bestimmen. Um die fehlenden Anteile zu ergänzen, greift der Algorithmus auf eine vorherige Parameterschätzung zurück. Damit entsteht ein iterativer Prozess aus zwei Schritten, dem Ausfüllen der fehlenden Anteile der Trainingsdaten sowie der Parameterbestimmung des Verteilungsmodells. Zur Vereinfachung der Schreibweise wird im folgenden die Menge der bekannten Anteile der Trainingsdaten mit $\mathcal{X}$ bezeichnet und die Menge der fehlenden Anteile mit $\mathcal{Z}$.

Der EM-Algorithmus ergänzt den Maximum-Likelihood Ansatz um einen Term für die fehlenden Anteile der Trainingsdaten. Ausgehend von einer initialen Parameterschätzung ϑ'

```
1.  wähle initiales ϑ^(0)
2.  wiederhole für t = 0, 1, 2, ...
3.     EXPECTATION-STEP (E-STEP): berechne Q(ϑ|ϑ^(t))
4.     MAXIMIZATION-STEP (M-STEP): setze ϑ^(t+1) := arg max_ϑ Q(ϑ|ϑ^(t))
```

Abbildung 4.1: EM-Algorithmus in allgemeiner Form

wird die Größe $Q(\vartheta|\vartheta')$ definiert:

$$\begin{aligned} Q(\vartheta|\vartheta') &:= E_{\mathcal{Z}}\left[log\mathcal{P}(\mathcal{X},\mathcal{Z}|\vartheta)|\mathcal{X},\vartheta'\right] \\ &= \int \log \mathcal{P}(\mathcal{X},\mathcal{Z}|\vartheta)\mathcal{P}(\mathcal{Z}|\mathcal{X},\vartheta')d\mathcal{Z} \\ &= \log \mathcal{P}(\mathcal{X}|\vartheta) + \int \log \mathcal{P}(\mathcal{Z}|\mathcal{X},\vartheta)\mathcal{P}(\mathcal{Z}|\mathcal{X},\vartheta')d\mathcal{Z} \end{aligned} \tag{4.1}$$

Statt der Daten-Log-Likelihood der vollständigen Daten wird also die erwartete Daten-Log-Likelihood mit Bezug auf die fehlenden Anteile berechnet. Die Maximierung von $Q(\vartheta|\vartheta')$ liefert einen neuen Parameter, der in die nächste Iteration des Verfahrens eingeht.

[Dempster 77] haben gezeigt, dass die Daten-Likelihood $\mathcal{P}(\mathcal{X}|\vartheta)$ in jeder Iteration steigt oder gleich bleibt. Unter bestimmten Bedingungen kann sogar gezeigt werden, dass das EM-Verfahren gegen das Maximum der Daten-Log-Likelihood konvergiert. Eine Zusammenfassung des Algorithmus' zeigt Abbildung 4.1.

Neben dem Maximum-Likelihood Ansatz ist es auch möglich, den EM-Algorithmus für eine Maximum-a-posteriori Schätzung einzusetzen. In diesem Fall wird die Funktion Q folgendermaßen definiert:

$$\begin{aligned} Q(\vartheta|\vartheta') &:= E_{\mathcal{Z}}\left[log\mathcal{P}(\vartheta|\mathcal{X},\mathcal{Z})|\mathcal{X},\vartheta'\right] \\ &= E_{\mathcal{Z}}\left[log\mathcal{P}(\mathcal{X},\mathcal{Z}|\vartheta)|\mathcal{X},\vartheta'\right] + log\mathcal{P}(\vartheta) - const \end{aligned} \tag{4.2}$$

4.1.2 Der EM-Algorithmus für GMMs

Um den EM-Algorithmus auf Gauß-Mischverteilungen anwenden zu können, interpretiere man die Mischverteilung wie in Abschnitt 2.4.3 als eine Klassenverteilung, bei der Klassenzugehörigkeiten unbekannt sind [Redner 84, Ghahramani 94]. Hierbei wird davon ausgegangen, dass die Größe k des GMM fest und vorgegeben ist.

Die bisherige Schätzung sei repräsentiert durch den Parametervektor $\vartheta' = (w'_1, \ldots, w'_k, \mu'_1, \ldots, \mu'_k, \Sigma'_1, \ldots, \Sigma'_k)$, die neu zu bestimmende Schätzung durch den Vektor $\vartheta = (w_1, \ldots, w_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k)$. Der Term $\varphi_{\mu,\Sigma}(\cdot)$ bezeichne die Dichte der multivariaten Normalverteilung mit Mittelwertvektor μ und Kovarianzmatrix Σ.

Durch Umformen und Einsetzen der Dichte der Normalverteilung in Formel (4.1) erhält

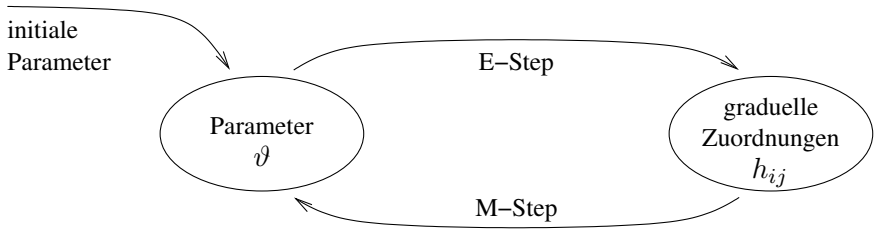

Abbildung 4.2: Schematische Darstellung der Arbeitsweise des EM-Algorithmus' für Mischverteilungen

man bei gegebener Trainingsstichprobe $\{x_1, \ldots, x_n\}$:

$$Q(\vartheta|\vartheta') := \sum_{i=1}^{n} \left\{ \sum_{j=1}^{k} h_{ij} (\log w_j + \log \varphi_{\mu_j, \Sigma_j}(x_i)) \right\} \tag{4.3}$$

mit

$$h_{ij} := \mathcal{P}(Z = j | X = x_i, \vartheta') = \frac{w'_j \varphi_{\mu'_j, \Sigma'_j}(x_i)}{\sum_{l=1}^{k} w'_l \varphi_{\mu'_l, \Sigma'_l}(x_i)} \tag{4.4}$$

Die h_{ij} modellieren einen Grad der Zugehörigkeit eines Trainingsmusters zu einer Komponente. Durch Ableiten der Gleichung (4.3) nach den Parametern ergeben sich die neuen Schätzer:

$$w_j = \frac{\sum_{i=1}^{n} h_{ij}}{n} \tag{4.5}$$

$$\mu_j = \frac{\sum_{i=1}^{n} h_{ij} x_i}{\sum_{i=1}^{n} h_{ij}} \tag{4.6}$$

$$\Sigma_j = \frac{\sum_{i=1}^{n} h_{ij} (x_i - \mu_j)(x_i - \mu_j)^T}{\sum_{i=1}^{n} h_{ij}} \tag{4.7}$$

Der E-step kann in dieser Anwendung des EM-Algorithmus' als das Berechnen der Zugehörigkeitsgrade h_{ij} angesehen werden, der M-step als Berechnung der Größen (4.5), (4.6) und (4.7). Eine schematische Abbildung des Algorithmus' ist in Abbildung 4.2 dargestellt.

4.1.3 Maximum-a-posteriori Schätzer für GMMs

Wie in Abschnitt 4.1.1 geschildert, kann der EM-Algorithmus nicht nur als Maximum-Likelihood Ansatz verwendet werden, sondern auch als Maximum-a-posteriori Ansatz. Dazu werden geeignete Prioriverteilungen ergänzt, wie in Gleichung (4.2) angegeben.

Für die Anwendung im Kontext von GMM-Parameterschätzung wurde eine mögliche Vorgehensweise in [Ormoneit 95] vorgestellt (vergleiche auch [Schafer 97]). Als Prioriver-

teilungen wurden Dirichlet-, Inverse Wishart- und Normalverteilung wiefolgt verwendet:

$$(w_1, \ldots, w_k) \sim D(\gamma, \ldots, \gamma) \tag{4.8}$$

$$\Sigma_j \sim W^{-1}(q, \Lambda) \tag{4.9}$$

$$\mu_j|\Sigma_j \sim N(m, \frac{1}{\eta}\Sigma_j) \tag{4.10}$$

Die Wahl der Prioriverteilungen und Parameter der Prioriverteilungen wird in Abschnitt 4.2.2 diskutiert. Damit ergeben sich die folgenden Berechnungsvorschriften für den M-step[1]:

$$w_j = \frac{\sum_{i=1}^{n} h_{ij} + \gamma - 1}{n + k(\gamma - 1)} \tag{4.11}$$

$$\mu_j = \frac{\sum_{i=1}^{n} h_{ij} x_i + \eta m}{\sum_{i=1}^{n} h_{ij} + \eta} \tag{4.12}$$

$$\Sigma_j = \frac{\sum_{i=1}^{n} h_{ij}(x_i - \mu_j)(x_i - \mu_j)^T + \eta(\mu_j - m)(\mu_j - m)^T + \Lambda^{-1}}{\sum_{i=1}^{n} h_{ij} + q - d} \tag{4.13}$$

Im Vergleich der Formeln (4.11),(4.12) und (4.13) mit den Berechnungsvorschriften des Maximum-Likelihood-Schätzers (4.5),(4.6) und (4.7) lässt sich der Einfluss der Prioriverteilungen erkennen: ist $\sum_{i=1}^{n} h_{ij}$ klein, werden die Terme von den Parametern der Prioriverteilungen dominiert, während andernfalls die Prioriparameter eine geringere Rolle spielen.

4.2 Data Augmentation

4.2.1 Allgemeine Formulierung von Data Augmentation

Unabhängig vom Gibbs-Sampler wurde Data Augmentation als Ansatz entwickelt, um unvollständigen Daten zu verarbeiten [Tanner 87]. Ausgangspunkt ist, wie beim EM-Algorithmus, eine unvollständige Trainingsstichprobe, die die Missing-at-random Bedingung erfüllt.

Im Sinne einer Bayesschen Analyse wird die Verteilung $\mathcal{P}(\vartheta, \mathcal{Z}|\mathcal{X})$ betrachtet, bzw. deren Randverteilung $\mathcal{P}(\vartheta|\mathcal{X})$. Durch Faktorisierung und Marginalisierung erhält man:

$$\begin{aligned}
\mathcal{P}(\vartheta, \mathcal{Z}|\mathcal{X}) &= \mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X}) \cdot \mathcal{P}(\vartheta|\mathcal{X}) \\
&= \mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X}) \cdot \int \mathcal{P}(\vartheta, \mathcal{Z}|\mathcal{X}) d\mathcal{Z} \\
&= \mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X}) \cdot \int \mathcal{P}(\vartheta|\mathcal{Z}, \mathcal{X}) \cdot \mathcal{P}(\mathcal{Z}|\mathcal{X}) d\mathcal{Z} \\
&= \mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X}) \cdot \int \mathcal{P}(\vartheta|\mathcal{Z}, \mathcal{X}) \cdot \int \mathcal{P}(\vartheta, \mathcal{Z}|\mathcal{X}) d\vartheta d\mathcal{Z} \\
&= \int (\mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X}) \mathcal{P}(\vartheta|\mathcal{Z}, \mathcal{X})) \cdot \mathcal{P}(\vartheta, \mathcal{Z}|\mathcal{X}) d(\vartheta, \mathcal{Z})
\end{aligned} \tag{4.14}$$

Gleichung (4.14) definiert den Übergangskern einer Markov-Kette mit stationärer Verteilung $\mathcal{P}(\vartheta, \mathcal{Z}|\mathcal{X})$. Der Übergangskern ist faktorisierbar in die beiden Teile $\mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X})$ und

[1] d bezeichnet wiederum die Dimensionalität der Daten (Größe der Datenvektoren)

1. wähle initiales $\vartheta^{(0)}$
2. wiederhole für $t = 0, 1, 2, \ldots$
3. IMPUTATION-STEP (I-STEP): sample $\mathcal{Z}^{(t+1)}$ aus $\mathcal{P}(\mathcal{Z}|\vartheta^{(t)}, \mathcal{X})$
4. POSTERIOR-STEP (P-STEP): sample $\vartheta^{(t+1)}$ aus $\mathcal{P}(\vartheta|\mathcal{Z}^{(t+1)}, \mathcal{X})$

Abbildung 4.3: Data Augmentation Algorithmus in allgemeiner Form

$\mathcal{P}(\vartheta|\mathcal{Z}, \mathcal{X})$, entsprechend einer Hintereinanderausführung zweier Einzelübergänge, einmal mit Wahrscheinlichkeit $\mathcal{P}(\mathcal{Z}|\vartheta, \mathcal{X})$ und danach mit Wahrscheinlichkeit $\mathcal{P}(\vartheta|\mathcal{Z}, \mathcal{X})$.

In algorithmische Form gebracht erhält man nach diesem Prinzip einen Algorithmus mit zwei Teilschritten, dem *Imputation-step* (I-step), in dem die fehlenden Einträge künstlich gesamplet werden, sowie dem *Posterior-step* (P-step), in dem Parameter aus der Posterioriverteilung gezogen werden (siehe Abbildung 4.3).

Im Gegensatz zum EM-Algorithmus berechnet Data Augmentation nicht einen besten Parameter, sondern erzeugt eine Zufallsfolge möglicher Parameter. Die Güte der erzeugten Parameter hängt von der Form der Posterioriverteilung ab.

4.2.2 Bayessche Modellierung von GMMs

Die Samplingverfahren erfordern eine Bayessche Modellierung für GMMs, da aus der Posterioriverteilung zufällige Parameter gezogen werden müssen. Nach dem Satz von Bayes ergibt sich die Posterioriverteilung wiefolgt:

$$\mathcal{P}(\vartheta|\mathcal{X}, \mathcal{Z}) = \frac{\mathcal{P}(\mathcal{X}, \mathcal{Z}|\vartheta)\mathcal{P}(\vartheta)}{\mathcal{P}(\mathcal{X}, \mathcal{Z})} \tag{4.15}$$

wobei wir davon ausgehen, dass die vollständigen Trainingsdaten $(\mathcal{X}, \mathcal{Z})$ gegeben und fest sind. Wir betrachten also eigentlich das Klassenmodell aus Abschnitt 2.4.3 mit bekannten Klassenzugehörigkeiten. Da über $(\mathcal{X}, \mathcal{Z})$ nicht variiert werden soll, ist der Nenner in (4.15) konstant und daher vereinfacht sich der Zusammenhang zu:

$$\mathcal{P}(\vartheta|\mathcal{X}, \mathcal{Z}) \propto \mathcal{P}(\mathcal{X}, \mathcal{Z}|\vartheta)\mathcal{P}(\vartheta) \tag{4.16}$$

Der erste Ausdruck auf der rechten Seite stellt die Daten-Likelihood dar, die direkt aus den Daten berechnet werden kann, während der zweite Term die Prioriverteilung der Parameter beschreibt. Im weiteren soll diskutiert werden, wie sich die Wahl einer Prioriverteilung auf die Gestalt der Posterioriverteilung auswirkt.

Zur Analyse der Priori-/Posterioriverteilungen betrachte man das in Abschnitt 2.4.3 beschriebene Klassenmodell, wobei davon ausgegangen wird, dass die Klassenzugehörigkeiten bekannt sind. Dann lassen sich die Daten nach Klassenzugehörigkeit aufteilen und das Problem zerfällt in die Analyse einer Multinomialverteilung (Klassenauswahl) und von einfachen Normalverteilungen (Komponenten der Mischverteilung).

Arten von Prioriverteilungen Unter den möglichen Prioriverteilungen gibt es bestimmte Formen, die besondere Eigenschaften aufweisen. Ein einfaches Rechnen erlauben die sogenannten *konjugierte* Verteilungen. Diese zeichnen dich dadurch aus, dass Priori- und

Posterioriverteilung aus der selben Verteilungsfamilie stammen. Dies ermöglicht eine vereinheitlichte Analyse und die Verwendung der selben Werkzeuge, z.B. die selben Zufallszahlengeneratoren.

Eine zweite Charakterisierung von Prioriverteilungen ist der Grad, mit dem sie die Parameterschätzung beeinflussen. Solche Prioriverteilungen, die keinen Einfluss ausüben, sich in gewissem Sinne also „neutral“ verhalten, werden als *nichtinformative* Prioriverteilungen [Robert 94] bezeichnet. Der Begriff der Nichtinformativität ist allerdings nicht klar definiert, es sind verschiedene Motivationen möglich:

1. eine Gleichverteilung über den gesamten Parameterraum bevorzugt keinen Parameter; allerdings kann eine einfache Parametertransformation aus der Gleichverteilung eine Prioriverteilung erzeugen, die nicht gleichverteilt ist

2. um invariant gegen Parametertransformation zu werden, wurde in [Jeffreys 61] (siehe auch [Robert 94]) eine informationstheoretische Methodik zur Bestimmung nichtinformativer Prioriverteilungen entwickelt. Diese Form ist auch als *Jeffreys' Prior* bekannt.

3. durch Vergleich des Maximum Likelihood und Maximum-a-posteriori Schätzers lassen sich solche Prioriverteilungen bestimmen, bei denen beide Schätzer zusammenfallen.

Obwohl der Begriff der nichtinformativen Prioriverteilung nicht eindeutig ist, ist die Beurteilung von Prioriverteilungen nach ihrem Einfluss auf die Posterioriverteilung ein zentraler Aspekt des Bayesschen Lernens. In Abschnitt 5.2 wird dieser Gedanke für die Anwendung auf GMMs weiter veranschaulicht.

Manche Funktionen, die z.B. aufgrund von Jeffreys' Invarianzprinzip als Prioriverteilungen verwendet werden sollten, widersprechen dem Prinzip der totalen Wahrscheinlichkeit, d.h. das Integral über diese Funktionen ist nicht endlich[2] Solche Funktionen werden als *unechte Verteilungen* bezeichnet. Sie können wie Dichten verwendet werden, so lange kein qualitativer Vergleich mit anderen Dichten oder ein Samplen aus einer solchen Dichte notwendig wird. Mithin eignen sie sich als Prioriverteilungen, nicht aber als Posterioriverteilungen. Ein Beispiel einer solchen unechten Verteilung ist die Gleichverteilung über die komplette Menge der reellen Zahlen.

Bayessche Modellierung der Multinomialverteilung Die Mischgewichte w_j eines GMM bzw. die Klassenwahrscheinlichkeiten eines Klassenmodells sind interpretierbar als Parameter einer Multinomialverteilung $\mathcal{M}(\nu, w_1, \ldots, w_k)$. Die hierzu konjugierte Prioriverteilung ist eine Dirichletverteilung $D(\gamma_1, \ldots, \gamma_k)$ (siehe Anhang B). Die Parameter γ_j der Dirichletverteilung bestimmen den Einflussgrad, den die Prioriverteilung hat: je größer γ_j, desto stärker konzentriert sich die Prioriverteilung um einen bestimmten Wert herum.

Der Begriff der nichtinformativen Prioriverteilung ist im Zusammenhang mit der Multinomialverteilung nicht eindeutig. Je nach Argumentation erhält man als nichtinformative Prioriverteilung:

1. $D(1, \ldots, 1)$ als Gleichverteilung über dem Parameterraum

2. $D(\frac{1}{2}, \ldots, \frac{1}{2})$ als Jeffreys Prior

[2] insbesondere ist das Integral ungleich 1, auch nach einer Skalierung der Funktion.

3. $D(0,\ldots,0)$ bei Vergleich von Maximum-Likelihood und Maximum-a-posteriori Schätzer. Diese Verteilung ist unecht.

In [Schafer 97] wird daher argumentiert, dass jede Prioriverteilung $D(c,\ldots,c)$ mit $0 \leq c \leq 1$ als, in gewissem Sinne, nichtinformativ anzusehen ist. Bei den Experimenten, die im Rahmen dieser Arbeit durchgeführt wurden, konnten nur geringe Unterschiede bei der Verwendung unterschiedlicher nichtinformativer Prioriverteilungen festgestellt werden.

Der Zusammenhang zwischen Priori- und Posterioriverteilung für gegebene Klassenhäufigkeiten $(u_1,\ldots,u_k)$ ergibt sich zu:

$$\begin{aligned}(w_1,\ldots,w_k) &\sim D(\gamma_1,\ldots,\gamma_k)\\ (u_1,\ldots,u_k) &\sim \mathcal{M}(\nu,w_1,\ldots,w_k)\\ (w_1,\ldots,w_k)|(u_1,\ldots,u_k) &\sim D(\gamma_1+u_1,\ldots,\gamma_k+u_k)\end{aligned} \tag{4.17}$$

wegen:

$$\begin{aligned}\mathcal{P}(w_1,\ldots,w_k|u_1,\ldots,u_k) &\propto \mathcal{P}(u_1,\ldots,u_k|w_1,\ldots,w_k)\cdot\mathcal{P}(w_1,\ldots,w_k)\\ &\propto \frac{\nu!}{u_1!\cdot\cdots\cdot u_k!}\prod_{j=1}^{k} w_j^{u_j}\cdot\frac{\Gamma(\gamma_1+\cdots+\gamma_k)}{\Gamma(\gamma_1)\cdot\cdots\cdot\Gamma(\gamma_k)}\prod_{j=1}^{k} w_j^{\gamma_j-1}\\ &\propto \prod_{j=1}^{k} w_j^{u_j+\gamma_j-1}\end{aligned} \tag{4.18}$$

Als Spezialfall der Dirichletverteilung für einen zweidimensionalen Parametervektor ergibt sich die Betaverteilung $B(\gamma_1,\gamma_2)$. Abbildung 4.4 veranschaulicht diese Verteilung für verschiedene Werte von α und β.

Bayessche Modellierung der Normalverteilung Die Bayessche Modellierung einer Normalverteilung erfordert eine Prioriverteilung für die Parameter μ (Erwartungswert) und Σ (Kovarianzmatrix). Die Prioriverteilung lässt sich faktorisieren als: $\mathcal{P}(\mu,\Sigma) = \mathcal{P}(\mu|\Sigma)\mathcal{P}(\Sigma)$.

Konjugierte Prioriverteilungen für das Normalverteilungsmodell sind die inverse Wishartverteilung sowie die Normalverteilung:

$$\Sigma \sim W^{-1}(q,\Lambda) \tag{4.19}$$

$$\mu|\Sigma \sim N(m,\frac{1}{\eta}\Sigma) \tag{4.20}$$

Der Parameter q bestimmt den Einfluss der Prioriverteilung: je größer q ist, desto stärker wird die Posterioriverteilung von der Prioriverteilung beeinflusst. Λ stellt einen mehrdimensionalen Skalierungparameter dar, der die Größe der Varianzen und Kovarianzen beeinflusst. Mit m wird die erwartete Lage der Normalverteilung angegeben, während η den Grad der Koppelung zwischen Erwartungswert und Kovarianzmatrix beschreibt.

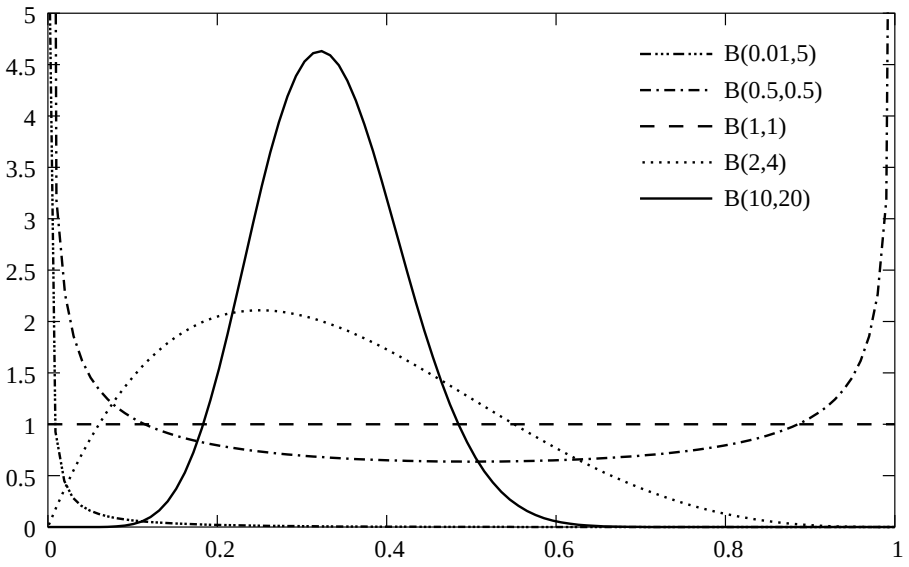

Abbildung 4.4: Dichte der Betaverteilung für unterschiedliche Parameterwerte. Je größer die Parameter sind, desto stärker konzentriert sich die Verteilung um einen bestimmten Punkt herum.

Mit diesen Prioriverteilungen erhält man als Posterioriverteilungen:

$$\begin{aligned} \Sigma &\sim W^{-1}(q, \Lambda) \\ \mu|\Sigma &\sim N(m, \frac{1}{\eta}\Sigma) \\ x_1, \ldots, x_n &\sim N(\mu, \Sigma) \\ \Sigma|x_1, \ldots, x_n &\sim W^{-1}(\hat{q}, \hat{\Lambda}) \\ \mu|\Sigma, x_1, \ldots, x_n &\sim N(\hat{m}, \frac{1}{\hat{\eta}}\Sigma) \end{aligned} \tag{4.21}$$

mit:

$$\hat{\eta} = \eta + n \tag{4.22}$$

$$\hat{q} = q + n \tag{4.23}$$

$$\hat{m} = \frac{n}{n+\eta}\bar{x} + \frac{\eta}{n+\eta}m \tag{4.24}$$

$$\hat{\Lambda} = \left(\Lambda^{-1} + n\hat{S} + \frac{n\eta}{n+\eta}(\bar{x} - m)(\bar{x} - m)^T\right)^{-1} \tag{4.25}$$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.26}$$

$$\hat{S} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T \tag{4.27}$$

Die Posterioriverteilungen (4.21) lassen sich über den Ansatz:

$$\begin{aligned}\mathcal{P}(\Sigma|x_1,\ldots,x_n) &= \int \mathcal{P}(\mu,\Sigma|x_1,\ldots,x_n)d\mu \\ &\propto \int \mathcal{P}(x_1,\ldots,x_n|\mu,\Sigma)\mathcal{P}(\mu,\Sigma)d\mu \qquad (4.28)\\ &= \int \mathcal{P}(x_1,\ldots,x_n|\mu,\Sigma)\mathcal{P}(\mu|\Sigma)\mathcal{P}(\Sigma)d\mu \\ \mathcal{P}(\mu|\Sigma,x_1,\ldots,x_n) &\propto \mathcal{P}(x_1,\ldots,x_n|\mu,\Sigma)\mathcal{P}(\mu|\Sigma) \qquad (4.29)\end{aligned}$$

und Einsetzen der jeweilige Dichtefunktionen berechnen. Die Berechnung ist direkt, aber etwas aufwändig, und soll daher an dieser Stelle ausgelassen werden. Herleitungen finden sich beispielsweise in [Robert 94] und [Schafer 97].

Im Fall univariater Daten vereinfachen sich die Verteilungen etwas. Unter Ausnutzung der Identitäten: $W^{-1}(q,\Lambda) = \Gamma^{-1}(\beta,\alpha)$ mit $q = 2\alpha$ und $\Lambda = (\frac{1}{2\beta})$ ergeben sich die Priori- und Posterioriverteilungen:

$$\begin{aligned}\sigma^2 &\sim \Gamma^{-1}(\beta,\alpha)\\ \mu|\sigma^2 &\sim N(m,\frac{1}{\eta}\sigma^2)\\ x_1,\ldots,x_n &\sim N(\mu,\sigma^2)\\ \sigma^2|x_1,\ldots,x_n &\sim \Gamma^{-1}(\hat{\beta},\hat{\alpha})\\ \mu|\sigma^2,x_1,\ldots,x_n &\sim N(\hat{m},\frac{1}{\hat{\eta}}\sigma^2)\end{aligned} \qquad (4.30)$$

mit:

$$\hat{\eta} = \eta + n \qquad (4.31)$$

$$\hat{\alpha} = \alpha + \frac{n}{2} \qquad (4.32)$$

$$\hat{m} = \frac{n}{n+\eta}\bar{x} + \frac{\eta}{n+\eta}m \qquad (4.33)$$

$$\hat{\beta} = \beta + \frac{1}{2}\left(n\hat{s}^2 + \frac{n\eta}{n+\eta}(\bar{x}-m)^2\right) \qquad (4.34)$$

$$\hat{s}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2 \qquad (4.35)$$

$$(4.36)$$

Integriert man die Posterioriverteilung des Erwartungswertes über die Varianz, erhält man eine t-Verteilung (siehe Anhang B), d.h. es gilt:

$$\begin{aligned}&\mu|x_1,\ldots,x_n \sim t(2\alpha+n,\hat{m},\rho^2)\\ &\text{mit}\\ &\rho^2 = \frac{1}{2\alpha+n}\left(\frac{n\eta(\bar{x}-m)^2}{(n+\eta)^2} + \frac{n\hat{s}^2+2\beta}{n+\eta}\right)\end{aligned} \qquad (4.37)$$

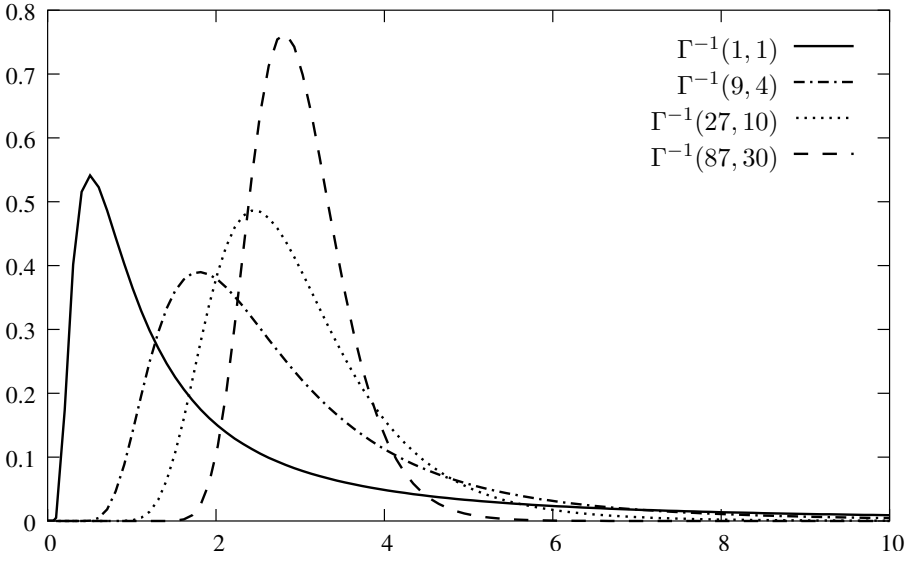

Abbildung 4.5: Dichte der inversen Gammaverteilung für unterschiedliche Parameterwerte. Je größer der zweite Parameter ist, desto stärker konzentriert sich die Verteilung um einen bestimmten Punkt herum.

Abbildung 4.5 zeigt die Dichte der inversen Gammaverteilung für verschiedene Parametereinstellungen, Abbildung 4.6 die der t-Verteilung. Es fällt auf, dass beide Verteilungen für kleine Werte der jeweiligen Formparameter sehr langschwänzige Verteilungen aufweisen. Teilweise existieren für derartige Parametereinstellungen die ersten und zweiten Momente (Erwartungswert und Varianz) nicht.

Als nichtinformative Prioriverteilung für das Normalverteilungsmodell werden in der Literatur [Schafer 97] die unechten Verteilungen mit $\eta \to 0$, $q \to -1$ und $\Lambda^{-1} \to 0$ angegeben, die sowohl durch Jeffreys' Invarianzprinzip als auch durch den Vergleich von Maximum-Likelihood- und Maximum-a-posteriori-Schätzer begründet sind. In diesem Fall ergeben sich als Posterioriverteilungen:

$$\begin{aligned} \Sigma|x_1,\ldots,x_n &\sim W^{-1}(n-1,(n\hat{S})^{-1}) \\ \mu|\Sigma,x_1,\ldots,x_n &\sim N(\bar{x},\frac{1}{n}\Sigma) \end{aligned} \tag{4.38}$$

Gesamtmodellierung und Alternativen Die Bayessche Modellierung der GMMs setzt sich zusammen aus:

1. Modellierung der Mischgewichte mit Dirichlet-Prioriverteilung
2. Modellierung der normalverteilten Komponenten mit invers-Wishartverteilten und normalverteilten Prioriverteilungen

Abbildung 4.7 gibt die Modellierung in Form eines Abhängigkeitsgraphen wieder.

Neben den beschriebenen Prioriverteilungen wurden in der Literatur auch alternative Verteilungen vorgeschlagen, z.B. in [Richardson 97] sowie [Stephens 97]. Diese entkoppeln

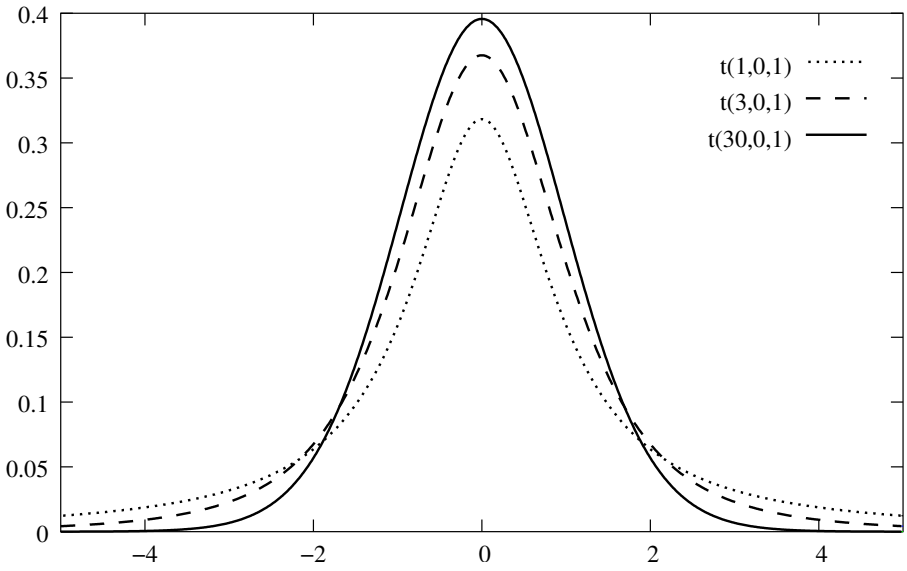

Abbildung 4.6: Dichte der t-Verteilung für unterschiedliche Parameterwerte. Je größer der Freiheitsgrad ist, desto stärker konzentriert sich die Verteilung um einen bestimmten Punkt herum.

die Prioriverteilungen für Erwartungswert und Varianz/Kovarianz einer Normalverteilung scheinbar. Dadurch lässt sich allerdings die gemeinsame Posterioriverteilung von Erwartungswert und Varianz/Kovarianz nicht mehr in geschlossener Form herleiten, wie es für das Data Augmentation Verfahren notwendig ist. Eine derartige Modellierung ist daher zwar für den Gibbs-Sampler bei inkrementeller Berechnung von Erwartungswert und Varianz/ Kovarianz geeignet, nicht aber für eine gleichzeitige Berechnung der beiden Größen.

4.2.3 Data Augmentation für GMMs

Die Bayessche Modellierung aus dem vorangegangenen Abschnitt sowie das algorithmische Grundgerüst aus Abbildung 4.3 bilden die Grundlage für das Verfahren zum Lernen von GMMs [Diebolt 94].

Imputation-step Im Imputation-step werden die fehlenden Daten gesamplet, im Fall der GMMs also die Zuordnungen der Daten zu den Komponenten. Dazu betrachte man die Verteilung:

$$\begin{aligned}\mathcal{P}(Z=j|X=x_i,\vartheta) &= \frac{\mathcal{P}(X=x_i|Z=j,\vartheta)\mathcal{P}(Z=j|\vartheta)}{\mathcal{P}(X=x_i|\vartheta)} \\ &= \frac{\mathcal{P}(X=x_i|Z=j,\vartheta)\mathcal{P}(Z=j|\vartheta)}{\sum_{l=1}^{k}(\mathcal{P}(X=x_i|Z=l,\vartheta)\mathcal{P}(Z=l|\vartheta))}\end{aligned} \tag{4.39}$$

Durch Einsetzen der entsprechenden Dichten ergibt sich:

$$\mathcal{P}(Z=j|X=x_i,\vartheta) = \frac{\varphi_{\mu_j,\Sigma_j}(x)\cdot w_j}{\sum_{l=1}^{k}(\varphi_{\mu_l,\Sigma_l}(x)\cdot w_l)} \tag{4.40}$$

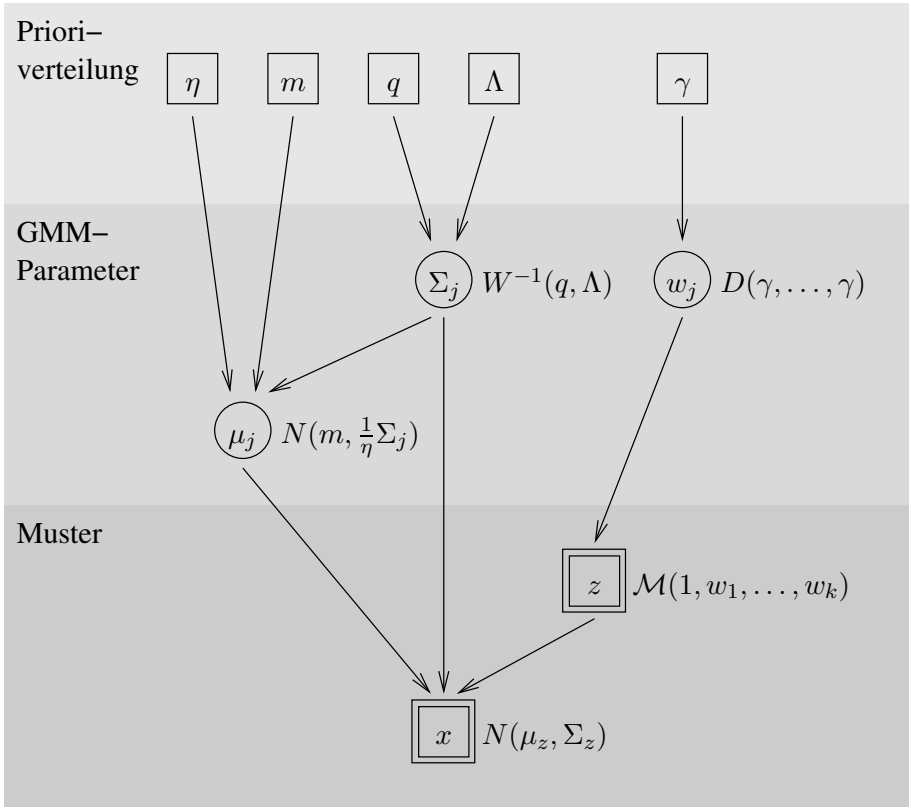

Abbildung 4.7: Graph, der die gegenseitigen Abhängigkeiten in der Bayesschen Modellierung wiedergibt. Die Modellparameter in der mittleren Schicht hängen von den Parametern der Prioriverteilungen ab, die Daten (untere Schicht) hängen von den Parametern des GMM ab (mittlere Schicht). Zwischen den Zuordnungen z und den reellwertigen Ausgabedaten x sowie den Kovarianzmatrizen Σ_j und den Erwartungswerten μ_j bestehen Abhängigkeiten innerhalb einer Schicht. Dennoch ist der Graph azyklisch.

Die Wahrscheinlichkeit aus Gleichung (4.40) ist identisch mit der Größe h_{ij} aus dem Expectation-step des EM-Algorithmus (4.4). Im Imputation-step wird zufällig aus der Verteilung $\mathcal{P}(\mathcal{Z}|\mathcal{X},\vartheta)$ gesamplet, die nach (4.41) durch eine Multinomialverteilung gegeben ist:

$$Z|X = x_i, \vartheta \sim \mathcal{M}(1, h_{i1}, \ldots, h_{ik}) \tag{4.41}$$

Posterior-step Im Imputation-step werden die Daten x_i jeweils einer Komponenten z_i zugeordnet. Mit dieser festen Aufteilung der Trainingsdaten werden im Posterior-step die Parameter der Mischverteilung gesamplet. Dabei greifen wir auf die Bayessche Modellierung aus Abschnitt 4.2.2 zurück.

Bezeichne n_j die Anzahl Trainingsdaten, die der j-ten Komponente zugeordnet wurden, d.h. $n_j = \#\{i \in \{1,\ldots,n\}|z_i = j\}$. Dann ergibt sich die Posterioriwahrscheinlichkeit der Mischgewichte w_j gemäß (4.17) zu einer Dirichletverteilung $D(\gamma_1 + n_1, \ldots, \gamma_k + n_k)$. Aus dieser Verteilung wird eine Zufallsstichprobe gezogen. Verfahren zum Samplen aus einer Dirichletverteilung sind in Anhang C beschrieben.

```
1.   wähle initiale (w_1, ..., w_k, μ_1, ..., μ_k, Σ_1, ..., Σ_k)
2.   wiederhole für t = 0, 1, 2, ...
        IMPUTATION-STEP (I-STEP):
3.      wiederhole für jedes Trainingsmuster x_i
4.         wiederhole für jede Komponente j
5.            berechne h_ij gemäß (4.40)
6.         sample Zuordnung z_i gemäß (4.41)
        POSTERIOR-STEP (P-STEP):
7.      sample neue Mischgewichte (w_1, ..., w_k) gemäß (4.17)
8.      wiederhole für jede Komponente j
9.         sample neue Kovarianzmatrix Σ_j gemäß (4.21)
10.        sample neuen Erwartungswert μ_j gemäß (4.21)
```

Abbildung 4.8: Data Augmentation Algorithmus für GMMs

Die Berechnung neuer Erwartungswerte μ_j und Kovarianzmatrizen Σ_j für die einzelnen Komponenten erfolgt für jede Komponente individuell. Zur Berechnung der Parameter der j-ten Komponente werden nur die Trainingsdaten verwendet, die im vorangegangenen Imputation-step dieser Komponente zugeordnet wurden, d.h. die Trainingsdaten x_i mit $z_i = j$.

Betrachte nun eine einzelne Komponente j. O.B.d.A. seinen genau die Trainingsdaten $\{x_i | i = 1, \ldots, n_j\}$ dieser Komponenten zugeordnet. Die Posterioriwahrscheinlichkeiten für Erwartungswert und Kovarianzmatrix dieser Komponenten erhält man gemäß (4.21), wobei in den Formeln n durch n_j zu ersetzen ist. Nun wird zunächst die neue Kovarianzmatrix der j-ten Komponente gesamplet und anschließend der neue Erwartungswert. Verfahren zum Samplen aus Inversen Wishartverteilungen und Normalverteilungen können Anhang C entnommen werden.

Abbildung 4.8 fasst den Data Augmentation Algorithmus für GMMs in algorithmischer Schreibweise zusammen, Abbildung 4.9 verdeutlicht den Ablauf grafisch. [Diebolt 94] kann entnommen werden, dass der Data Augmentation Algorithmus für GMMs eine ergodische Markov-Kette mit der stationären Verteilung $\mathcal{P}(\vartheta | x_1, \ldots, x_k)$ erzeugt, wenn die gewählten Prioriverteilungen nicht unecht sind.

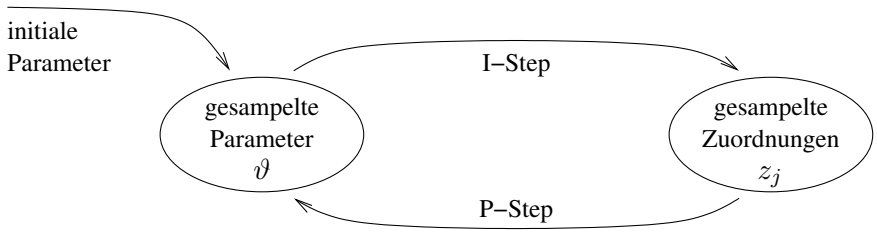

Abbildung 4.9: Schematische Darstellung der Arbeitsweise des Data Augmentation Algorithmus' für Mischverteilungen

4.3 Bestimmung der Größe einer Mischverteilung

4.3.1 Motivation

Bisher wurde bei allen Lernverfahren eine feste Größe der Mischverteilung vorausgesetzt, die durch den Lernalgorithmus nicht verändert werden kann, d.h. k wurde als konstant betrachtet. Nun soll geschildert werden, in welcher Weise die Wahl von k den Lernvorgang beeinflusst und welche Ansätze zur Bestimmung der Komponentenanzahl existieren. Dabei konzentriert sich diese Abhandlung auf eine Auswahl gängiger Verfahren. Weitere Ansätze können z.B. [McLachlan 00] entnommen werden.

Die Anzahl Komponenten k bestimmt die Fähigkeit der Mischverteilung, sich an verschiedenen Verteilungen anzupassen, erheblich. Je größer k, desto mehr Freiheitsgrade besitzt das Verteilungsmodell und desto größer ist die Hypothesenmenge $\mathcal{H}$. Dies ermöglicht einerseits die Anpassung an vielfältige Datensätze, andererseits vergrößert sich die Gefahr des Overfitting (siehe Abschnitt 2.1). Eine geeignete Wahl von k ist daher ein wesentlicher Beitrag zum Erreichen einer optimalen Schätzung.

Der Größenparameter k unterscheidet sich wesentlich von den restlichen Parametern der Mischverteilungen, da er a) nur diskrete Werte annehmen kann und b) die Größe des Parametervektors beeinflusst. In gewisser Hinsicht lässt sich k daher als ein Parameter auf einer zweiten Stufe bezeichnen.

Unter den Verfahren zur Größenbestimmung für GMMs lassen sich drei Arten unterscheiden:

1. Empirische Vorgehensweise (Kreuzvalidierung)
2. Analytische Verfahren (z.B. AIC, BIC, Evidenz)
3. Randomisierte Verfahren (MCMC-Algorithmen)

Die Verfahren der ersten beiden Kategorien bauen auf dem EM-Algorithmus auf, indem Mischverteilungen verschiedener Größe trainiert werden und anschließend durch ein empirisches oder analytisches Kriterium eine der untersuchten GMM-Größen ausgewählt werden. Diese Verfahren sind i.d.R. generisch, d.h. sie können auch für andere Anwendungen im unüberwachten oder überwachten Lernen verwendet werden.

Die Verfahren der dritten Kategorie greifen hingegen direkt in den Lernalgorithmus – meistens ein Gibbs-Sampler ähnlich dem Data Augmentation Ansatz – ein, indem zusätzliche Zwischenschritte zum Erzeugen oder Löschen einzelner Komponenten integriert werden. Die Bestimmung der GMM-Größe lässt sich bei diesen Verfahren daher nicht mehr vom Lernen der GMM-Parameter loslösen. Die Verfahren sind damit spezifisch auf die Lernaufgabe abgestimmt.

4.3.2 Empirische Verfahren

Empirische Verfahren zur Größenbestimmung basieren auf dem Prinzip der Validierung/ Kreuzvalidierung. Hierzu wird die Gesamtmenge der zur Verfügung stehenden Daten in eine *Trainingsmenge* und eine *Validierungsmenge* unterteilt. Die Modelle werden mit den Trainingsdaten trainiert, während die Güte des gelernten Modells auf den Validierungsdaten gemessen wird. Wichtig ist, dass Trainings- und Validierungsdaten disjunkte Mengen sind,

da sonst Overfitting nicht erkannt werden könnte. Als Trainingsverfahren kann z.B. der EM-Algorithmus verwendet werden.

Eine Variante dieses Prinzips stellt die sogenannte ν-fache Kreuzvalidierung dar. Hierbei wird die gesamte Datenmenge in ν gleichgroße, disjunkte Teilmengen U_l aufgeteilt. Anschließend werden ν verschiedene Modelle mit unterschiedlichen Trainingsmengen, aber gleicher Größe, nach folgendem Schema trainiert:

Modell-Nr.	Trainingsmenge	Validierungsmenge
1.	$U_2 \cup U_3 \cup U_4 \cup \cdots \cup U_\nu$	U_1
2.	$U_1 \cup U_3 \cup U_4 \cup \cdots \cup U_\nu$	U_2
$\vdots$	$\vdots$	$\vdots$
ν.	$U_1 \cup U_2 \cup U_3 \cup \cdots \cup U_{\nu-1}$	U_ν

(4.42)

Die Güte der ν Modelle auf den jeweiligen Validierungsdaten wird anschließend addiert. Als Gütemaß wird i.d.R. die Daten-Log-Likelihood verwendet. Ein Spezialfall der Kreuzvalidierung ist der *leave-one-out error*, bei dem die gesamte Datenmenge in einelementige Teilmengen U_l aufgeteilt wird.

Um die adäquate Größe einer Mischverteilung zu bestimmen, werden nach dem Prinzip der Validierung/Kreuzvalidierung Mischverteilungen verschiedener Größen trainiert und die Güte auf den Validierungsmengen verglichen. Die optimale Größe ist diejenige mit der größten Güte auf der Validierungsmenge.

4.3.3 Analytische Verfahren

Analog den im vorangegangenen Abschnitt beschriebenen empirischen Verfahren erfordern auch die analytischen Verfahren das Training mehrerer verschieden großer Modelle. Die Bestimmung der optimalen Modellgröße erfolgt erst in einem nachgelagerten Auswahlschritt.

Die analytischen Verfahren kommen ohne Validierungsmenge aus, d.h. die gesamten Daten stehen zum Training zur Verfügung. Statt dessen wird die Güte des gelernten Modells auf der Trainingsmenge gemessen und um einen Strafterm ergänzt, der die Modellkomplexität berücksichtigt: je größer das verwendete Modell ist, desto größer wird der Strafterm, so dass sehr große Modelle nur dann gewählt werden, wenn die Güte der Anpassung sehr groß ist. In diesem Abschnitt soll eine Auswahl von drei Verfahren vorgestellt werden: AIC, BIC und Evidenzansatz.

AIC Der AIC-Ansatz[3] geht auf [Akaike 73] zurück. Akaike konnte zeigen, dass der Maximum-Likelihood Schätzer nicht erwartungstreu ist und dass der systematische Fehler unter bestimmten Randbedingungen durch die Anzahl der Modellparameter abgeschätzt werden kann (siehe auch [Burnham 98]). Durch Subtraktion des systematischen Fehlers von der Log-Likelihood und Multiplikation[4] mit -2 erhält man das AIC:

$$\mathrm{AIC} := -2 \sum_{i=1}^{n} \log q(x_i|\vartheta) + 2\kappa \tag{4.43}$$

[3] ursprünglich Abkürzung für „an information criterion“, heute oft mit „Akaike's information criterion“ bezeichnet

[4] aus historischen Gründen

wobei κ die Anzahl skalarer Parameter des durch ϑ gegebenen Modells ist. Durch Minimierung des AIC (4.43) über alle betrachteten Modellgrößen und alle möglichen Parameter erhält man den AIC-Schätzer.

Die Parameterzahl κ ist die Anzahl der *skalaren* Parameter des Modells. Ein d-dimensionaler Erwartungswert als Parameter einer Normalverteilung trägt daher zur Parameterzahl mit dem Wert d bei; die $d \times d$-große symmetrische Kovarianzmatrix einer Normalverteilung enthält $\frac{d(d+1)}{2}$ skalare Parameter, der d-dimensionale Parametervektor einer Multinomialverteilung nur $d-1$ freie, skalare Parameter. In der Anwendung auf d-variate GMMs mit k Komponenten und allgemeinen Kovarianzmatrizen [Bozdogan 84] ergibt sich die Parameterzahl somit zu:

$$\kappa_{GMM(k,d)} = (d-1) + k(d + \frac{d(d+1)}{2}) = (\frac{3}{2}k + 1)d + \frac{k}{2}d^2 - 1 \tag{4.44}$$

BIC Ein weiteres Kriterium berücksichtigt neben der Parameterzahl auch die Größe der Trainingsmenge. Dieses „Bayesian information criterion“ (BIC) geht auf [Schwarz 78] zurück. Es basiert auf einem Modellvergleich auf der Grundlage von Bayesfaktoren, wobei die Prioriwahrscheinlichkeiten geeignet abgeschätzt werden (siehe auch [Ghosh 01]). Dadurch erhält man das BIC zu:

$$\text{BIC} := -2 \sum_{i=1}^{n} \log q(x_i|\vartheta) + \kappa \log n \tag{4.45}$$

wobei κ wiederum die Anzahl skalarer Parameter des Modells und n die Größe der Trainingsmenge bezeichnet. Zu beachten ist, dass n im Fall multivariater Daten die Anzahl Datenvektoren angibt, nicht die Anzahl skalarer Einträge.

Durch Minimierung von (4.45) über alle Modellgrößen und alle möglichen Parameter erhält man das optimale Modell. Für kleine Trainingsmengen verhalten sich AIC und BIC sehr ähnlich, bei großen Trainingsstichproben dagegen bestraft das BIC große Modelle stärker als das AIC und bevorzugt daher kleinere Modelle. Die Möglichkeit, das BIC im Zusammenhang mit GMMs einzusetzen, wurde u. a. in [Fraley 98] untersucht.

Evidenzansatz Einem sehr allgemeinen Prinzip folgend, aber in der Anwendung spezialisiert auf GMMs, bietet der Evidenzansatz eine dritte Möglichkeit der Modellselektion. Dabei wird die Modellgröße nach der größtmöglichen Evidenz bestimmt. Erfolgreiche Anwendungen dieser Vorgehensweise finden sich u.a. im Bereich neuronaler Netze [Ragg 00] sowie GMM-Bestimmung [Roberts 98].

Als die Evidenz einer Modellklasse bezeichnet man die Wahrscheinlichkeit der Trainingsdaten gegeben eine Modellklasse, unabhängig von den Parametern der Modellklasse. Für GMMs mit k Komponenten ist die Evidenz gegeben durch:

$$\mathcal{P}(x_1, \ldots, x_n|k) = \int_{\vartheta_k \in \Theta_k} \mathcal{P}(x_1, \ldots, x_n|k, \vartheta_k)\mathcal{P}(\vartheta_k|k)d\vartheta_k \tag{4.46}$$

wobei $\vartheta_k \in \Theta_k$ die möglichen Parameter eines GMM mit k Parametern durchläuft.

Nach Wahl einer nichtinformativen Prioriverteilung für $\mathcal{P}(\vartheta_k|k)$ sowie Approximation von (4.46) durch eine Taylor-Entwicklung zweiten Grades um den Maximum-Likelihood-Schätzer ϑ_{ML} lässt sich die Evidenz des Modells nährungsweise berechnen. Eine genaue

Darstellung der Vorgehensweise findet sich in [Roberts 98]. Der resultierende Ausdruck ist von der Form:

$$\log \mathcal{P}(x_1, \ldots, x_n|k) \approx LL(x_1, \ldots, x_n|\hat{\vartheta}_{ML}) - c(\hat{\vartheta}_{ML}, k, n) \tag{4.47}$$

wobei c ein komplizierter Strafterm ist. Die Modellauswahl erfolgt durch Maximierung der Evidenz (4.47) über der Modellgröße k. Die genaue Form des Evidenzansatzes kann [Roberts 98] entnommen werden.

4.3.4 Randomisierte Verfahren

Die randomisierten Verfahren zur Größenbestimmung bauen auf den MCMC-Ansätzen zur Parameterbestimmung (Gibbs-Sampling, Data Augmentation) auf, indem sie die Übergangskerne dieser Verfahren um einen Anteil erweitern, der die Größe der Mischverteilung verändert. Hierbei müssen also nicht – wie bei den empirischen und analytischen Ansätzen – mehrere Mischverteilungen trainiert werden, um anschließend anhand eines geeigneten Kriterium die beste Größe auszuwählen, sondern die Größenbestimmung erfolgt im Zusammenhang mit der Parameterbestimmung.

Ein Ansatz besteht darin, einen reversible-jump MCMC-Algorithmus zu entwickeln (siehe Abschnitt 3.5), dessen Zustandsraum die Parametersätze unterschiedlich großer Mischverteilungen umfasst und der Übergänge zwischen diesen unterschiedlichen Parametergrößen erlaubt.

In [Richardson 97] wird ein derartiger Ansatz entwickelt, dessen Übergangskern folgende Übergänge zulässt:

1. Neubestimmung der Mischgewichte $(w_1, \ldots, w_k)$
2. Neubestimmung der Erwartungswerte und Varianzen/Kovarianzen μ_j, Σ_j
3. Neubestimmung der Daten-Zuordnungen z_i
4. Aufspaltung einer Komponenten
5. Verschmelzen zweier Komponenten
6. Entstehung einer leeren Komponente
7. Entfernung einer leeren Komponente

Daneben existiert in dieser Modellierung noch ein Übergang zur Veränderung eines Hyperparameters, der hier nicht weiter betrachtet werden soll. Durch Gewichtung der verschiedenen Übergangstypen können bestimmte Übergänge bevorzugt werden und damit die Häufigkeit eines Wechsels der Größe des Modells verändert werden.

Die Übergangskerne des ersten, zweiten und dritten Schrittes entsprechen der jeweiligen Modellierung von Data Augmentation bzw. Gibbs-Sampling ohne Variation über die Größe der GMM. Der vierte, fünfte, sechste und siebente Übergang verändern dagegen die Größe der Mischverteilung, springen also zwischen verschiedenen Teil-Zustandsräumen.

Eine Variante der beschriebenen Vorgehensweise stellt die Modellierung von Stephens dar [Stephens 97, Stephens 00]. Anstatt die Übergänge zur Veränderung der Größe des GMM in die Markov-Kette zu integrieren wird dort ein zeitkontinuierlicher Markovprozess der Markov-Kette überlagert. Dieser Markovprozess bestimmt die Zeitpunkte des Entstehens und des Entfernens einzelner Komponenten aus der Mischverteilung.

Kapitel 5

Weiterentwicklung von Data Augmentation

5.1 Motivation und Grundidee

5.1.1 Beschränkungen der bestehenden Verfahren

Obgleich der EM-Algorithmus als Standardverfahren zum Lernen von GMMs bezeichnet werden kann, besitzt er einige systematische Schwächen, die zu fehlerhaften und degenerierten Ergebnissen führen können. In Abschnitt 1.2 wurden diese Schwächen bereits skizziert, im folgenden sollen sie nochmals anhand eines Beispiels verdeutlicht werden.

Man betrachte hierzu eine univariate Verteilung, die durch ein GMM mit zwei Komponenten beschrieben werden kann und deren Dichte in Abbildung 5.1 (links) dargestellt ist. Aus dieser Verteilung sei eine unabhängige Trainingsstichprobe gegeben, aus der ein GMM gelernt werden soll.

Ergebnisse bei bekannter GMM-Größe Wird der EM-Algorithmus auf diesen Trainingsdaten mit einem GMM der korrekten Größe, d.h. mit zwei Komponenten, trainiert, so erhält man oft ein befriedigendes Ergebnis, d.h. das trainierte GMM ist der in Abbildung 5.1 (links) dargestellten Dichte ähnlich. Allerdings können auch andere Fälle auftreten:

- *Overfitting und Degenerierung:* eine der Komponenten passt sich an die Gesamtheit der Daten an, während sich die andere Komponente auf einen Punkt der Trainingsdaten spezialisiert und zu einer sehr hohen, sehr schmalen Spitze verengt (siehe Abbildung 5.1 Mitte) und im Extremfall degeneriert, d.h. ihre Varianz wird Null. Die Lage der degenerierten Komponente hängt offensichtlich nicht von der ursprünglichen Dichte ab, aus der die Daten stammen, sondern von der zufälligen Lage eines Punktes der Trainingsmenge, denn würde eine zweite zufällige Stichprobe aus der Ursprungsverteilung gezogen, so läge fast sicher an der betreffenden Stelle kein Datenpunkt. Ein derartiges GMM stellt daher ein Artefakt des Lernvorgangs dar.

  Die Entstehung derart degenerierter GMMs erklärt sich aus dem Maximum-Liklihood-Prinzip, da im Fall von GMMs mit mindestens zwei Komponenten die Likelihood-Funktion unbeschränkt ist. Dabei markieren die überangepassten Situationen mit degenerierten Komponenten gerade die Ausprägungen mit unbeschränkter Likelihood,

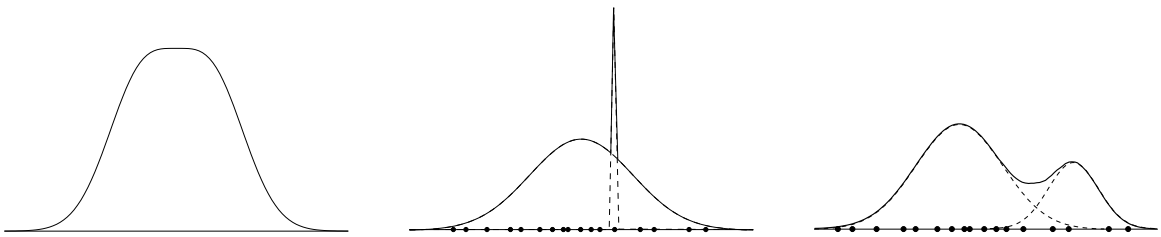

Abbildung 5.1: Beispiel für die Probleme des EM-Algorithmus. Links: die Ausgangsdichte, aus der die Trainingsdaten stammen lässt sich durch ein GMM mit zwei Komponenten exakt beschreiben. Mitte: Overfitting und Degenerierung einer Komponente. Rechts: Underfitting durch Konvergenz in ein lokales Optimum. Die Punkte stellen die Trainingsdaten dar, die durchgezogene Linie die Dichte des GMM, die gebrochenen Linien die Dichte der einzelnen Komponenten.

da durch Verkleinerung der Varianz einer Komponente die Likelihood am Mittelwert der Komponente ins Unendliche getrieben werden kann. Da der EM-Algorithmus die Likelihood auf den Trainingsdaten iterativ maximiert, sind derartige Overfitting-Situationen unausweichlich.

- *Underfitting*: In manchen Fällen tritt auch eine Unteranpassung auf, d.h. die Modellkomplexität erlaubt ein besseres Ergebnis als vom EM-Algorithmus tatsächlich berechnet wird. Dieser Fall ist beispielhaft in Abbildung 5.1 (rechts) dargestellt.

  Eine derartige Unteranpassung kann durch vorzeitiges Abbrechen des Verfahrens (sog. *early stopping*) oder durch Konvergenz in ein lokales Maximum der Likelihoodfunktion hervorgerufen werden. Hierbei ist zu beachten, dass einerseits die Konvergenzgeschwindigkeit des EM-Ansatzes gering ist und andererseits *early stopping* häufig verwendet wird, um Overfitting zu vermeiden.

Bestimmung der GMM-Größe Ist die GMM-Größe nicht bekannt und muss ebenfalls bestimmt werden, treten mit dem EM-Algorithmus weitere Probleme auf:

- *Kreuzvalidierung:* Die Verwendung von Kreuzvalidierung erfordert ein vielfaches Wiederholen des Lernvorgangs für verschiedene GMM-Größen sowie verschiedene Aufteilungen der Daten in Trainings- und Validierungmenge. Dieses Verfahren ist daher sehr zeitaufwändig. Ferner kann es von Underfitting in die Irre geführt werden.

- *Analytische Modellselektionskriterien:* Die Verwendung analytischer Selektionskriterien erfordert ebenfalls die Wiederholung des Lernvorgangs für verschiedene GMM-Größen. Aufgrund der nicht erforderlichen Aufteilung der Daten in Trainings- und Validierungsmenge ist dieser Ansatz jedoch nicht ganz so zeitaufwändig wie die Kreuzvalidierung. Dafür kann er nicht nur durch Underfitting, sondern auch durch Overfitting in die Irre geleitet werden, da ein überangepasstes GMM aufgrund der großen Likelihood auf der Trainingsmenge für das Selektionskriterium attraktiver erscheint als es tatsächlich ist. Dieses Problem ist kritisch, da Overfitting umso häufiger auftritt, je größer das GMM ist.

MCMC-Ansätze MCMC-Ansätze wie Data Augmentation und der Gibbs-Sampler erlauben das Samplen von Parametern aus den Posterioriverteilungen. Allerdings ist unklar, wie aus der Sequenz gesampleter Parameter ein bester Schätzer berechnet werden kann.

Bisherige Ansätze in [Diebolt 94] und [Stephens 97] führen eine einfache Mittelung über alle gesampleten Parameter durch, wobei teilweise eine Vorverarbeitung stattfindet, um sog. *Label-switching* zu vermeiden. Ob eine derartige Verarbeitung der gesampleten Parameter sinnvoll ist, wurde bisher weder analytisch noch experimentell untersucht. Zweifel ergeben sich schon aus der Tatsache heraus, dass die Posterioriverteilung mulimodal und eine einfache Mittelung somit nicht adäquat ist.

Als weiterer Mangel bisheriger MCMC-Ansätze ist die Frage nach einer geeigneten Prioriverteilung zu stellen, da bisherige Ansätze eine willkürliche Wahl treffen. Dieser Punkt ist insofern kritisch, als die Prioriverteilungen die Ergebnisse des MCMC-Samplers massiv beeinflussen. Ein ausführlicher Vergleich verschiedener Prioriverteilungen ist daher unabdingbar; er ist bisher aber noch nicht erfolgt.

5.1.2 Grundidee für ein neuartiges Lernverfahren

Die Motivation des neuartigen Lernverfahrens erfolgt in fünf Schritten (i)-(v):

(i) Betrachte Bereiche des Parameterraums statt einzelner Stellen
Die Beschreibung der Schwächen des EM-Ansatzes hat gezeigt, worin die Probleme des Maximum-Likelihood-Prinzips für GMMs bestehen. Zum einen existieren suboptimale Maxima, die zu Underfitting führen können, zum anderen existieren einzelne Stellen mit unbeschränkter Likelihood, die Overfitting und Degenerierung hervorrufen. Abbildung 5.2 (links) veranschaulicht diesen Sachverhalt: die Stelle M_3 ist die eigentlich gesuchte beste Anpassung, während die Stelle M_2 eine aufgrund von Underfitting suboptimale Stelle und M_1 eine Overfittingsituation darstellen.

Im Sinne des Maximum-Likelihood-Prinzips (ML) stellen Underfittingsituationen kein prinzipielles Problem dar: sie können durch globale Suchverfahren umgangen werden. Anders dagegen verhält sich die Lage bei Overfittingsituationen: diese scheinen gemäß dem ML-Prinzip sehr gute Anpassungen zu sein, während sie tatsächlich unbrauchbar sind. Das bedeutet, dass ML als Optimierungsziel ungeeignet ist, da es nicht die tatsächliche Modellierungsgüte misst.

Betrachtet man die Overfittingsituationen genauer, so stellt man fest, dass diese zwar ein Maximum der Likelihoodfunktion besitzen[1], diese Maximumstelle aber sehr schmal ist, d.h. kleine Veränderungen der Parameter führen zu einem drastischen Einbruch der Likelihood. Im Fall von GMMs beispielsweise führt eine kleine Lageveränderung einer überangepassten Komponente dazu, dass die Komponente keine Datenpunkte mehr beschreibt. Im Gegensatz dazu führen in den nicht überangepassten Bereichen des Parameterraums kleine Veränderungen der Parameter auch nur zu kleinen Veränderungen der Likelihood.

Um Überanpassung zu vermeiden sollte also nicht nur die Likelihood an einer bestimmten Stelle, sondern auch in der Umgebung dieser Stelle betrachtet werden: gute Modellparameter zeichnen sich dadurch aus, dass die Likelihood auch in der Umgebung der Parameter groß ist, während überangepasste Modelle nur an einer einzigen Stelle eine sehr große Likelihood

[1] genau genommen: eine unbeschränkte Likelihood

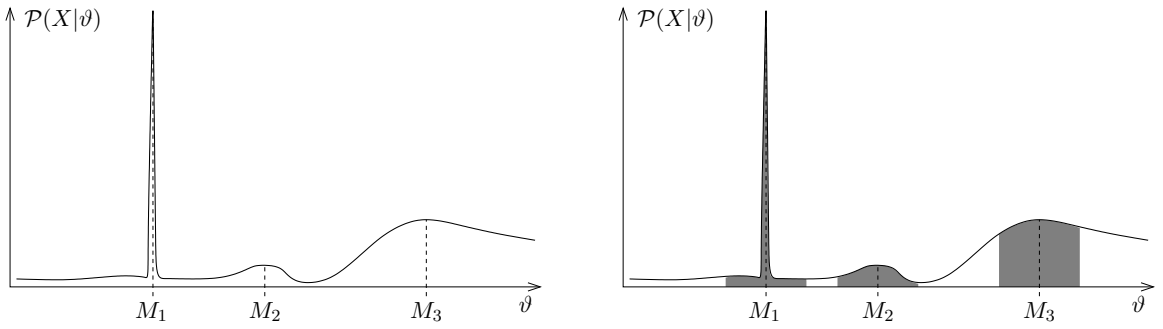

Abbildung 5.2: Links: Typischer Verlauf der Likelihoodfunktion. Stelle M_1 stellt eine Overfittingsituation dar, Stelle M_2 eine Underfittingsituation und Stelle M_3 den gesuchten optimalen Parametersatz. Rechts: Die selbe Likelihoodfunktion mit lokalen Integralen um die Maxima der Likelihoodfunktion.

besitzen, während die Likelihood in der Umgebung klein ist. Suboptimale Anpassungen, d.h. Underfittingsituationen, besitzen in der gesamten Umgebung eine kleine Likelihood.

An Stelle des Prinzips: *suche eine Stelle, an der die Likelihood groß ist* tritt also das Prinzip: *suche einen kleinen Bereich, in dem die Likelihood durchweg groß ist.*

Abbildung 5.2 (rechts) veranschaulicht dieses Prinzip, indem statt der Likelihood an einem Punkt das Integral der Likelihoodfunktion über einen kleinen Bereich gebildet wird. Offensichtlich ist dieses Integral für die Stelle M_3 größer als für M_1 und M_2.

(ii) Betrachte Posterioriverteilung statt Likelihood

In Abschnitt 4.2 wurde die Posterioriverteilung sowie verschiedene Arten von Prioriverteilungen eingeführt. Es konnte gezeigt werden, dass für eine bestimmte Art von Prioriverteilungen – den sog. nichtinformativen Prioriverteilungen – die Dichte der Posterioriverteilung sehr ähnlich der Likelihoodfunktion ist. In bestimmten Fällen ist die Dichte der Posteriorverteilung sogar proportional zur Likelihoodfunktion.

Es ist daher möglich, statt der Likelihoodfunktion die Posterioriverteilung mit nichtinformativen Prioriverteilungen zu analysieren. Den Bereichen mit durchweg großer Likelihood entsprechen dabei die Bereiche mit großer Wahrscheinlichkeitsmasse bezüglich der Posterioriverteilung.

(iii) Betrachte Zufallsstichprobe statt der Verteilung selbst

Um kleine Bereiche des Parameterraums mit großer Posteriori-Wahrscheinlichkeit zu finden, muss die Posterioriverteilung mathematisch repräsentiert werden. Eine analytische Darstellung ist jedoch für GMMs nicht möglich. Auch eine numerische Repräsentation scheitert in der Praxis an der Größe des Parametervektors: schon für kleine GMMs mit wenigen Komponenten beträgt die Anzahl skalarer Modellparameter mehrere Dutzend.

Obwohl die Posterioriverteilung nicht als solche dargestellt werden kann, erlaubt die MCMC-Technik, eine Zufallsstichprobe dieser Verteilung zu erzeugen. Die Konvergenzaussagen aus Kapitel 3 garantieren die korrekte Verteilung der gesampleten Parameter. Insbesondere werden aus den Bereichen hoher Wahrscheinlichkeit sehr viele Parameter gezogen, während aus den Bereichen niedriger Posterioriwahrscheinlichkeit wenige Parameter gezogen

werden. Die Zufallsstichprobe erlaubt also eine numerische Repräsentation der Posterioriverteilung.

(iv) Betrachte Dynamik der Markov-Kette
Die Zufallsstischprobe des MCMC-Samplers ist stochastisch abhängig, wobei die Art der Abhängigkeit durch die darunterliegende Markov-Kette gegeben ist. Dabei kann beobachtet werden: in Bereichen hoher Posterioriwahrscheinlichkeit ist die Schrittweite der Markov-Kette gering, d.h. die Parameter in aufeinanderfolgenden Iteration sind sehr ähnlich. Die Dynamik der Markov-Kette ist somit gering und der Sampler hält sich lange Zeit in diesen Bereichen hoher Wahrscheinlichkeit auf, bevor er sie wieder verlässt. In Bereichen geringer Wahrscheinlichkeit ist die Dynamik der Markov-Kette dagegen groß, d.h. neue Parameter unterscheiden sich sehr stark von bisherigen.

Diese stochastische Abhängigkeit der Markov-Kette kann ausgenutzt werden, um Bereiche hoher Posterioriwahrscheinlichkeit zu identifizieren: ist die Dynamik der Markov-Kette gering, so befindet man sich offensichtlich in einem solchen Bereich, ansonsten befindet man sich in einem Bereich niedriger Wahrscheinlichkeit. Um Bereiche hoher Wahrscheinlichkeit zu finden, muss man also nicht alle gesampleten Parameter gleichzeitig betrachten, sondern man kann sich auf eine Analyse der Dynamik von Teilsequenzen der Markov-Kette konzentrieren.

(v) Optimiere Parameter durch Mittelung
Der MCMC-Sampler erzeugt zufällig Parameter aus der Posterioriverteilung und garantiert damit in keiner Weise, besonders gute Parameter zu finden. Allerdings hält sich die Markov-Kette längere Zeit in den Bereichen hoher Posterioriwahrscheinlichkeit auf.

Durch Analyse der Übergangswahrscheinlichkeiten kann gezeigt werden, dass die Bereiche hoher Posterioriwahrscheinlichkeit lokal um das lokale Maximum durch eine Normalverteilung approximiert werden können. Befindet sich die Markov-Kette in einem Bereich hoher Wahrscheinlichkeit, so kann demzufolge durch eine Mittelwertbildung über die gesampleten Parameter eine bessere Modellausprägung berechnet werden als die einzelnen gesampleten Parametersätze darstellen.

Skizze des neuen Lernverfahrens
Ausgehend von den Überlegungen (i)-(v) wird im Folgenden ein neues Lernverfahren entwickelt, das aus folgenden Teilen besteht:

- mit Hilfe eines MCMC-Samplers wird eine Zufallssequenz von Modellparametern erzeugt. Dabei kommen nichtinformative Prioriverteilungen zum Einsatz.
- durch Mittelung der Parameter über ein gleitendes Fenster werden die gesampleten Parameter optimiert. Die somit gefundenen Kandidaten sind nicht an die Trainingsdaten überangepasst.
- die Auswahl des besten Parameters erfolgt nach dem Maximum-Likelihood-Prinzip.
- die Modellkomplexität wird eingestellt durch Elimination überzähliger Komponenten. Welche Komponenten überzählig sind, ergibt sich aus dem MCMC-Sampler.

5.1.3 Weitere Vorgehensweise

In den folgenden Abschnitten sollen der beschriebene Lösungsansatz im Detail entwickelt und untersucht werden. Dazu erfolgt zunächst eine Analyse der Samplingalgorithmen. Als Ausgangspunkt dient das bereits eingeführte Verfahren *Data Augmentation*.

Aufbauend auf der Analyse werden Schlussfolgerungen im Hinblick auf die Konstruktion eines Lernverfahrens für GMMs gezogen und das Data Augmentation Verfahren entsprechend erweitert, so dass es zur Schätzung von GMM-Parametern einschließlich der GMM-Größe eingesetzt werden kann.

Durch weitere Optimierung des Verfahrens und Kombination mit dem EM-Ansatz wird in Kapitel 6 das REM-Verfahren entwickelt, das eine weitere Verbesserung der Ergebnisse ermöglicht. Eine ausführliche experimentelle Untersuchung in Kapitel 7 schließt die Untersuchung ab.

5.2 Das Verhalten von Data Augmentation

5.2.1 Verhalten bei Verwendung echter Prioriverteilungen

Zunächst soll der Fall untersucht werden, in dem die gewählten Prioriverteilungen echte Verteilungen im Sinne der Stochastik sind. Das bedeutet zumindest im Bezug auf die Prioriverteilung der Erwartungswerte und Varianzen/Kovarianzen auch, dass die gewählten Prioriverteilungen nicht nicht-informativ sind.

Fallbeispiel Die Abbildungen 5.3 und 5.4 zeigen ein Fallbeispiel, in dem ein GMM mit vier Komponenten auf 200 univariaten Trainingsdaten mit dem Data Augmentation Verfahren trainiert wurde. Die univariaten Daten sind gleichverteilt aus dem Intervall $[0, 1]$ gezogen. Die Prioriverteilungen wurden gemäß der Modellierung aus Abschnitt 4.2.2 mit den Parametern: $\gamma = 3$, $q = 3$, $\eta = 3$, $m = \bar{x} \approx 0.49$ und $\Lambda^{-1} = (\hat{s}^2) \approx (0.078)$ gewählt.

Die Prioriverteilung für die Mischgewichte konzentriert die Wahrscheinlichkeitsmasse auf das Intervall $[0, 0.6]$ mit einem Maximum der Dichte zwischen den Werten 0.1 und 0.3. Mischgewichte größer als 0.6 haben eine verschwindend kleine Prioriwahrscheinlichkeit. Die Prioriverteilung der Varianzen ist so gewählt, dass der Erwartungswert der Prioriverteilung genau der empirischen Varianz der Trainingsstichprobe entspricht, so dass ein großer Teil der Wahrscheinlichkeitsmasse der Prioriverteilung im tatsächlich interessanten Teil der Parameterwerte liegt. Durch die Wahl von $\eta = 3$ sind die Prioriverteilungen von Erwartungswert und Varianz gekoppelt, d.h. je größer der Erwartungswert, desto stärker werden große Varianzen bevorzugt.

Verhalten von Data Augmentation In den beiden Abbildungen 5.3 und 5.4 kann man gut die große Dynamik[2] der erzeugten Markov-Kette erkennen: Mischgewichte und Erwartungswerte springen sehr stark von einem Wert zum anderen und die Anordnung der Komponenten verändert sich fortwährend. Dabei ist allerdings zu beachten, dass dennoch der Bereich der sinnvollen Werte nicht verlassen wird. Beispielsweise bleiben die Erwartungswerte stets im Intervall $[0, 1]$, in dem auch die Trainingsdaten liegen.

[2] englisch: *good mixing property*

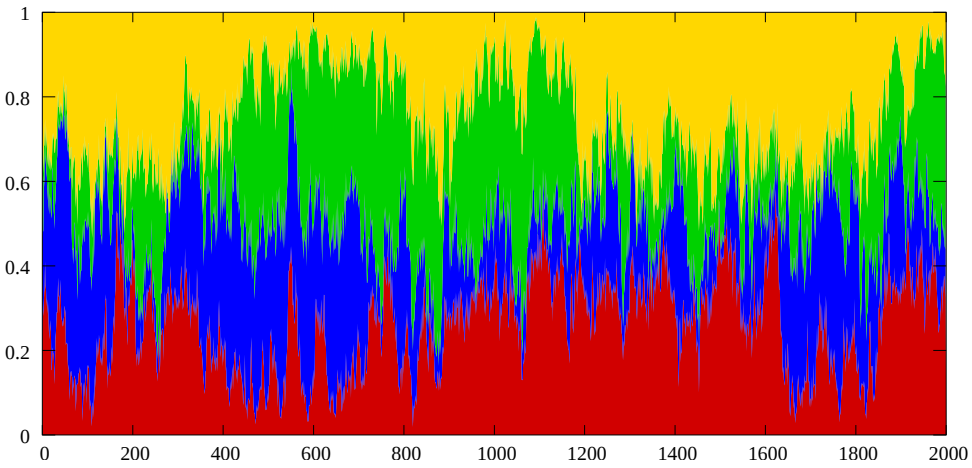

Abbildung 5.3: Entwicklung der Mischgewichte eines GMM mit 4 Komponenten für univariate, uniform verteilte Daten bei 2000 Iterationen Data Augmentation. Die Anteile der Komponenten sind übereinander aufgetragen.

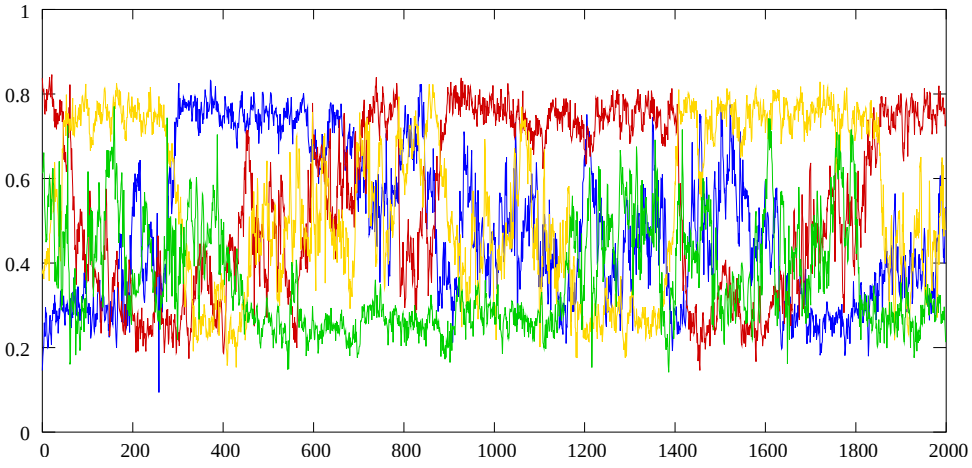

Abbildung 5.4: Entwicklung der Erwartungswerte eines GMM mit 4 Komponenten. Die Daten sind im Intervall $[0, 1]$ gleichverteilt. Jede Linie zeigt die Entwicklung des Erwartungswertes einer Komponenten während 2000 Iterationen Data Augmentation.

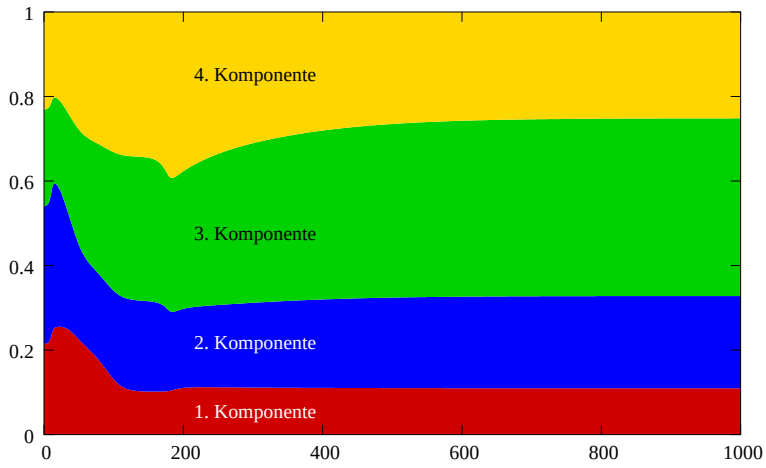

Abbildung 5.5: Entwicklung der Mischgewichte eines vier-komponentigen GMM im Zuge von 1000 Iterationen des EM-Algorithmus'.

Es fällt auf, dass trotz des randomisierten Verhaltens bestimmte Werte bevorzugt werden, z.B. in Abbildung 5.4 die Werte um 0.25 und 0.8. Nimmt der Erwartungswert einer Komponente einen solchen Wert an, bleibt dieser über mehrere hundert Iterationen ungefähr gleich. Beim Vergleich der Abbildungen 5.3 und 5.4 erkennt man, dass einer Komponente, die sich an einer solchen bevorzugten Stelle aufhält, auch ein Mischgewicht zugeordnet ist, das aus einem für diesen Bereich typischen Intervall stammt. Ferner lässt sich beobachten, dass Werte in den bevorzugten Bereichen fast durchweg von je genau einer Komponente angenommen werden, d.h. es gibt keine Iteration, in der diese Bereiche ohne Komponente blieben, es halten sich aber auch fast nie mehrere Komponenten gleichzeitig in diesen Bereichen auf.

Neben den bevorzugten Stellen gibt es auch Bereiche des Parameterraums, in denen zwar Werte angenommen werden, aber keine Stabilisierung stattfindet. In Abbildung 5.4 ist dies beispielsweise das Intervall von 0.3 bis 0.7. Die Erwartungswerte der Komponenten, die in diesem Intervall liegen, schwanken sehr stark, bis dieser Bereich irgendwann ganz verlassen wird.

Vergleich mit dem EM-Algorithmus Zum Vergleich stellt Abbildung 5.5 die Entwicklung der Mischgewichte, Abbildung 5.6 die Entwicklung der Erwartungswerte eines vierkomponentigen GMM im Verlauf des EM-Algorithmus' dar. Deutlich ist zu erkennen, dass sich die Werte nach einer Einschwingphase schnell stabilisieren. Durch eine bessere Initialisierung ließe sich die Einschwingphase sogar weiter verkürzen.

Damit wird deutlich, worin die Unterschiede zwischen einem klassischen Lernverfahren wie dem EM-Algorithmus und einem MCMC-Ansatz bestehen: der klassische Lernansatz versucht, durch lokale Optimierung der Anpassungsgüte eine zügige Konzentration auf die interessanten Bereiche des Parameterraums zu erzielen, allerdings unter Inkaufnahme einer unzureichenden Exploration des Suchraums. Beim MCMC-Ansatz steht dagegen die Exploration im Vordergrund; eine Konzentration auf bestimmte Bereiche des Parameterraums erfolgt nur in dem Rahmen, in dem die Posterioriverteilung die Wahrscheinlichkeits-

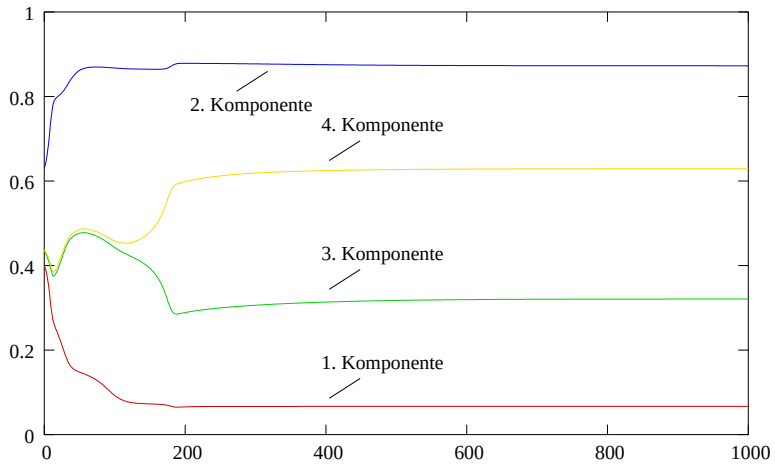

Abbildung 5.6: Entwicklung der Erwartungswerte eines vier-komponentigen GMM auf 200 gleichverteilten Trainingsdaten im Zuge von 1000 Iterationen des EM-Algorithmus'. Nach einer Einschwingphase stabilisieren sich die Werte.

masse auf bestimmte Bereiche konzentriert. Ein Verlernen einer bereits gefundenen Lösung ist dabei nicht ausgeschlossen.

Die Abbildung 5.7 zeigt die Entwicklung der Daten-Log-Likelihood auf Trainings- und Testmenge im Verlauf von 2000 Iterationen Data Augmentation. Beide Kurven verlaufen parallel und schwanken um den Wert von -0.09 bzw. -0.12. Ein Trend oder eine Konvergenz sind nicht zu erkennen.

Im Gegensatz dazu zeigt Abbildung 5.8 den Verlauf der Daten-Log-Likelihood im Zuge des EM-Algorithmus: deutlich ist die monotone Zunahme der Likelihood auf den Trainingsdaten bis hin zur Sättigung zu erkennen. Die Likelihood auf den Testdaten wächst dagegen nicht so schnell an, zeigt jedoch auch einen steigenden Trend. Die kleine Delle zwischen den Iterationen Nr. 200 und 400 deutet bereits auf ein leichtes Overfitting hin, da hier zwar die Likelihood auf den Trainingsdaten steigt, nicht aber die auf den Testdaten, d.h. es wird eine Besonderheit der Trainingsdaten gelernt. Die resultierende Likelihood ist beim EM-Algorithmus deutlich größer als bei Data Augmentation.

Schlussfolgerungen Der Vergleich der beiden Verfahren Data Augmentation und EM-Algorithmus zeigt deutlich die prinzipiell unterschiedlichen Arbeitsweisen: während der EM-Algorithmus versucht, einen möglichst guten Parameterwert durch schrittweise lokale Optimierung zu berechnen, sucht Data Augmentation zufällig im Parameterraum. Eine Optimierung findet nicht statt, so dass die gefundenen Parametersätze zwar gut, aber nicht optimal sind.

5.2.2 Verhalten bei Verwendung nichtinformativer Prioriverteilungen

Wie sehr die Prioriverteilungen die Arbeitsweise von Data Augmentation beeinflussen, lässt sich erkennen, wenn man die Ergebnisse des vorangegangenen Abschnitts mit dem Verhalten bei der Verwendung nichtinformativer Prioriverteilungen vergleicht. Dazu wurde der Algo-

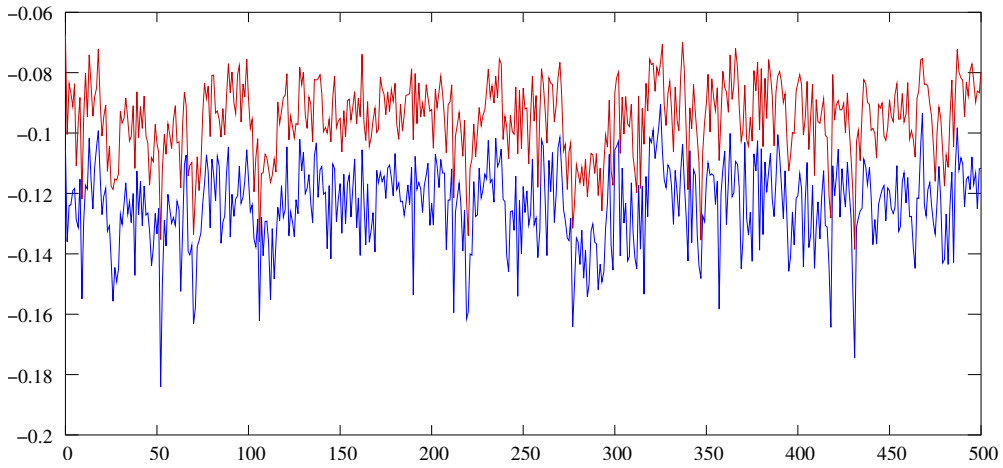

Abbildung 5.7: Entwicklung der mittleren Loglikelihood während eines Laufs von Data Augmentation. Die obere Linie misst die Loglikelihood auf den Trainingsdaten, die untere misst die Loglikelihood auf den Testdaten.

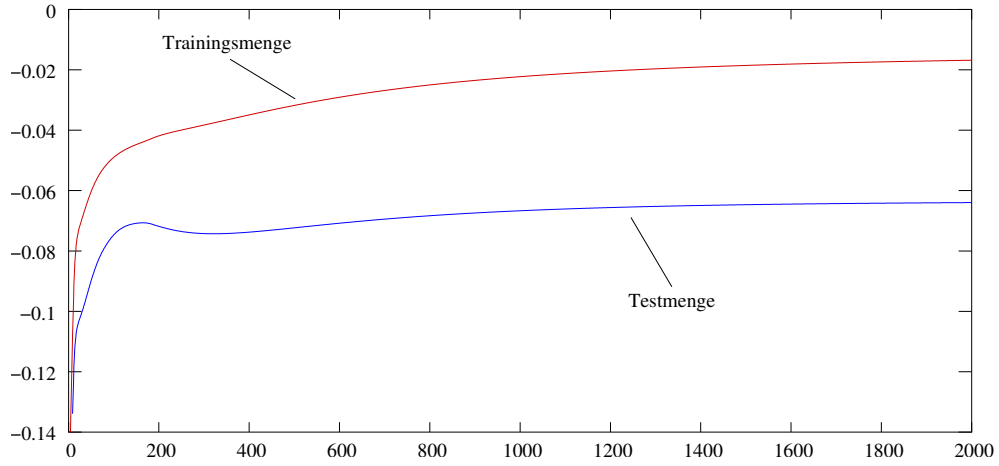

Abbildung 5.8: Entwicklung der mittleren Loglikelihood während eines Laufs des EM-Algorithmus'. Die obere Linie misst die Loglikelihood auf den Trainingsdaten, die untere misst die Loglikelihood auf den Testdaten.

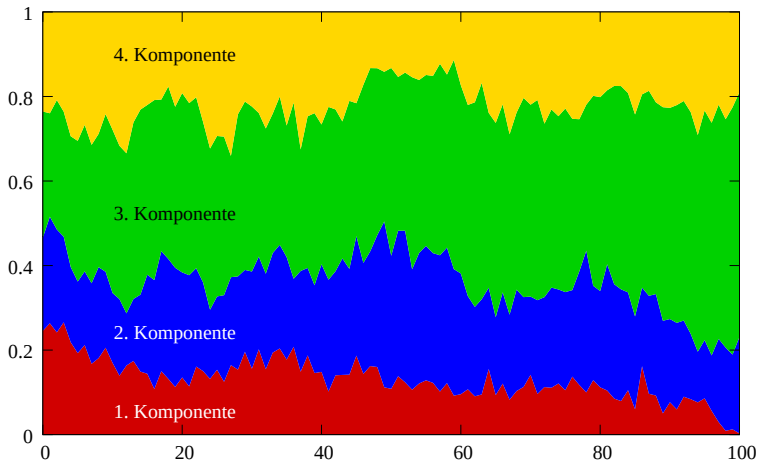

Abbildung 5.9: Entwicklung der Mischgewichte eines GMM mit 4 Komponenten für univariate, uniform verteilte Daten bei den ersten 100 Iterationen Data Augmentation mit nichtinformativen Prioriverteilungen. Die Anteile der Komponenten sind übereinander aufgetragen.

rithmus auf den selben Daten, aber mit den in Abschnitt 4.2.2 beschriebenen nichtinformativen Prioriverteilungen mit $\gamma = 0$, $q = -1$, $\eta = 0$, $m = \bar{x}$ und $\Lambda^{-1} = 0$ angewendet.

Verhalten von Data Augmentation Die Entwicklung der Mischgewichte, Erwartungswerte und Standardabweichungen während der ersten 100 Iterationen kann den Abbildungen 5.9, 5.10 und 5.11 entnommen werden. Im Vergleich zu der Entwicklung bei echten Prioriverteilungen können folgende Unterschiede festgestellt werden:

- die Entwicklung der Werte springt nicht so sehr; die Veränderungen in einer Iteration sind geringer
- das Intervall, in dem die Erwartungswerte der Komponenten zu finden sind, ist größer
- die Anordnung der Erwartungswerte der Komponenten auf der reellen Achse ändert sich nicht
- das Mischgewicht der ersten Komponente wird nahezu Null

Auffällig ist die Entwicklung der ersten Komponente ab Iteration Nr. 95: das zugehörige Mischgewicht fällt von ca. $\frac{17}{200}$ in Iteration 95 auf nur noch $\frac{0.5}{200}$ in Iteration 100. Bei 200 Trainingsdaten erklärt diese Komponente also nicht mal mehr ein einziges Trainingsmuster. Parallel dazu nimmt die Standardabweichung zu von 0.18 in Iteration 95 auf 0.54 in Iteration 100. Ebenso springt der Erwartungswert in den letzten drei Iterationen sehr stark. Die Parameter der drei anderen Komponenten verändern sich dagegen wenig; lediglich das Mischgewicht der benachbarten 3. Komponente wächst in dem Maße, in dem das Mischgewicht der ersten Komponente abnimmt. Abbildung 5.12 verdeutlicht diesen Prozess anhand der GMMs, die in verschiedenen Iterationen gesamplet wurden: während die erste Komponente in der Iteration 95 noch einen erkennbaren Beitrag zur Mischverteilung leistet, ist ihr Anteil in der 99. Iteration kaum mehr erkennbar.

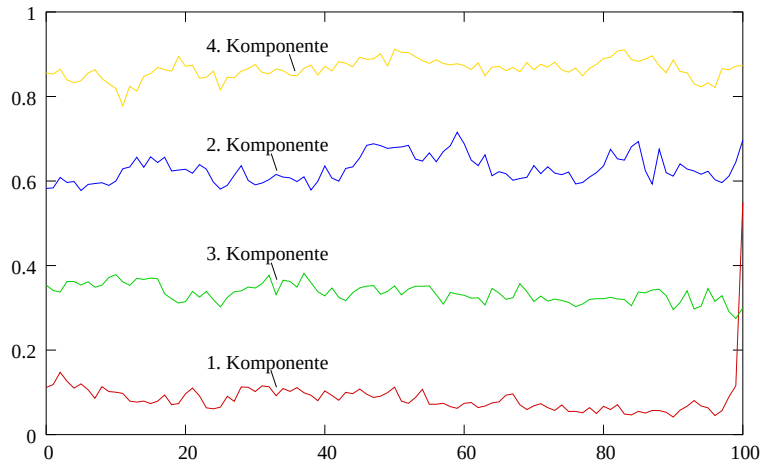

Abbildung 5.10: Entwicklung der Erwartungswerte eines GMM mit 4 Komponenten während der ersten 100 Iterationen Data Augmentation mit nichtinformativen Prioriverteilungen.

Einfluss der Prioriverteilungen Die Entwicklung der Erwartungswerte von Data Augmentation mit nichtinformativen Prioriverteilungen (siehe Abbildung 5.10) sowie die des EM-Algorithmus (siehe Abbildung 5.6) zeigen eine ähnliche Anordnung der Komponenten über dem gesamten Intervall zwischen ca. 0.1 und 0.9. Im Gegensatz dazu sind die Erwartungswerte im Fall von Data Augmentation mit echten Prioriverteilungen (siehe Abbildung 5.4) auf das Intervall von 0.2 bis 0.8 beschränkt.

Dieser Unterschied erklärt sich aus der Form der Prioriannahmen im Fall echter Prioriverteilungen: die Erwartungswerte werden als normalverteilt um 0.5 angenommen und die Varianz und der Erwartungswert einer Komponente werden so gekoppelt, dass kleine Varianzen nur bei Erwartungswerten im Bereich um 0.5 akzeptiert werden. Die gegebenen Trainingsdaten stammen jedoch aus einer Gleichverteilung, deren Dichte am Rand, hier also an den Stellen 0 und 1, Sprungstellen besitzt. Um diese Sprünge mit einem GMM anzunähern, müssen an den Rändern Komponenten mit kleiner Varianz und daher steilen Flanken angeordnet werden, die jedoch einerseits eine geringe Prioriwahrscheinlichkeit besitzen und denen andererseits wegen der geringen Varianz auch nur eine kleine Anzahl Trainingsmuster zugeordnet werden. Daher wird die Posterioriverteilung maßgeblich durch die Prioriannahmen bestimmt, die die Komponenten zum Wert 0.5 hin ziehen.

Im nichtinformativen Fall findet dagegen weder eine Kopplung von Erwartungswert und Varianz, noch eine Bevorzugung bestimmter Werte statt, so dass die Anordnung der Komponenten ausschließlich von den Trainingsdaten bestimmt wird. Dies kann sowohl im Fall von Data Augmentation als auch für den EM-Algorithmus beobachtet werden.

Der Vergleich des Lernverhaltens zeigt damit sehr deutlich den Einfluss der Prioriverteilungen. Selbst bei der Verwendung von Prioriverteilungen, die verhältnismäßig wenige Vorgaben machen und die gut zu den gegebenen Trainingsdaten passen – wie im ersten Experiment – werden die Ergebnisse stark beeinflusst. Auch der Vergleich der Likelihood auf Trainings- und Testdaten (siehe Abbildungen 5.7 und 5.13) verdeutlicht die Behinderung des Lernens durch die Prioriannahmen, da sowohl auf den Trainings- wie auch auf den Testdaten die Log-Likelihood bei Verwendung nichtinformativer Prioriannahmen höher ist.

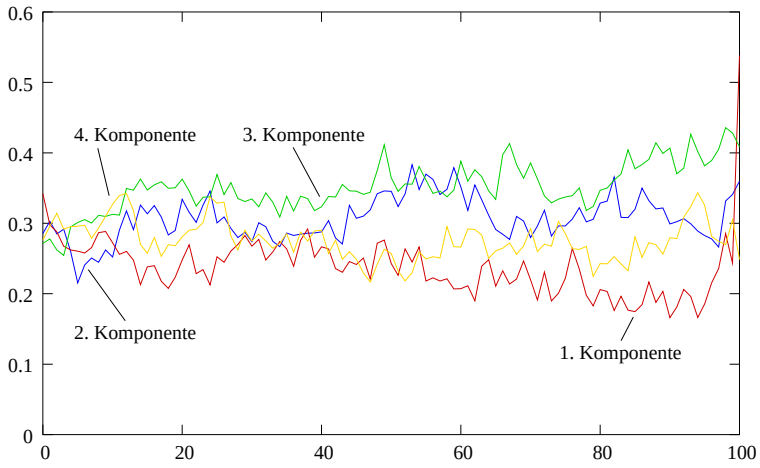

Abbildung 5.11: Entwicklung der Standardabweichungen eines GMM mit 4 Komponenten während der ersten 100 Iterationen Data Augmentation mit nichtinformativen Prioriverteilungen.

Das Verschwinden einer Komponente Eine Schwierigkeit bei der Arbeit mit nichtinformativen Prioriverteilungen ist die Möglichkeit, dass die Posterioriverteilungen unecht werden können, so dass ein Samplen nicht mehr möglich ist. Dieser Fall tritt ein, wenn einer Komponente im Imputation-step zu wenige oder gar keine Muster zugeordnet werden.

Vergleicht man die Entwicklung der Mischgewichte der jeweils ersten Komponente bei der Verwendung des EM-Algorithmus' (siehe Abbildung 5.5) und von Data Augmentation mit nichtinformativen Prioriverteilungen (siehe Abbildung 5.9) – in beiden Fällen die Komponente, die am weitesten links auf der reellen Achse angeordnet ist – stellt man fest, dass sich das Mischgewicht im Fall des EM-Algorithmus' etwa bei einem Wert von 0.1 stabilisiert, während es im Fall von Data Augmentation zunächst etwa 90 Iterationen lang um 0.1 schwankt, bevor es auf Null abfällt.

Die Konvergenz des Mischgewichts der ersten Komponente beim EM-Algorithmus lässt sich auf eine zunehmende Spezialisierung zurückführen: die Komponente konzentriert sich zunehmend auf die Muster, die am linken Rand der Gleichverteilung liegen. Ab etwa der 200. Iteration haben nur noch Muster in diesem Bereich große Zugehörigkeitswerte h_{ij} zu dieser

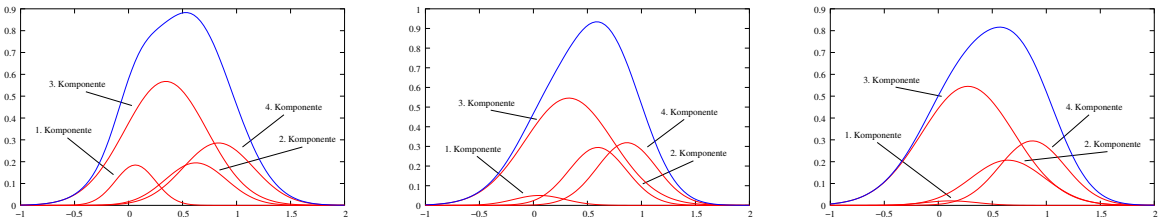

Abbildung 5.12: Dichte der GMMs, die von Data Augmentation mit nichtinformativen Prioriverteilungen in den Iterationen 95 (links), 97 (Mitte) und 99 (rechts) gesamplet wurden. Dargestellt ist die Dichte der Mischverteilung sowie die einzelnen Komponenten. Die erste Komponenten verschwindet zusehends.

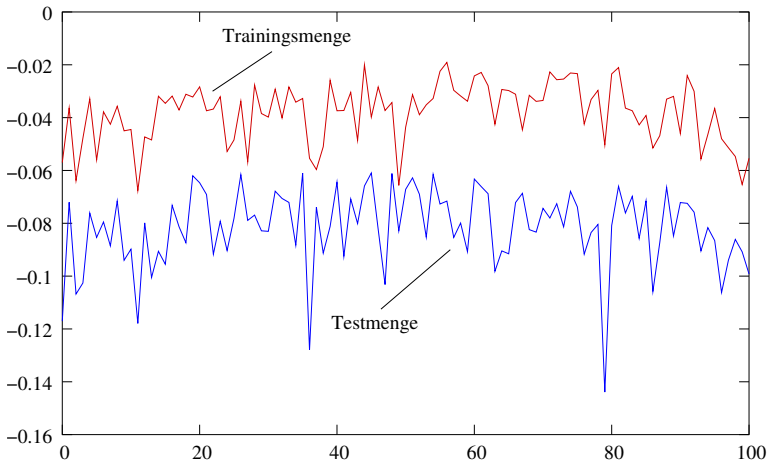

Abbildung 5.13: Entwicklung der Daten-Log-Likelihood auf der Trainings- (oben) und Testmenge (unten) während der ersten 100 Iterationen Data Augmentation mit nichtinformativen Prioriverteilungen

Komponente. Da die Berechnung der Modellparameter im M-step nach dem Maximum-Likelihood-Prinzip erfolgt, werden die Parameter der ersten Komponente so gewählt, dass sie die Daten im linken Bereich der Verteilung optimal beschreiben. Dadurch werden auch im darauffolgenden E-step hohe Zugehörigkeitswerte erreicht.

Das Data Augmentation Verfahren mit nichtinformativen Prioriverteilungen ist im Gegensatz dazu durch die stochastische Daten-Zuordnung und die randomisierte Parameterbestimmung gekennzeichnet. Im Fall einer Komponenten, der viele Daten zugeordnet werden, verhält sich der P-step sehr ähnlich dem M-step des EM-Algorithmus: bis auf einen kleinen Rauschanteil werden Parameter gesamplet, die den Maximum-Likelihood-Schätzern sehr ähnlich sind. Abbildung 5.14 verdeutlicht dies anhand der Wahrscheinlichkeiten, ähnliche Erwartungswerte und Varianzen zu samplen. Dieses Verhalten ist dadurch begründet, dass die Posterioriverteilungen bei vielen zugeordneten Daten ihre Wahrscheinlichkeitsmasse in einem kleinen Bereich um den Maximum-Likelihood-Schätzer herum konzentrieren (vgl. Abschnitt 4.2.2).

Anders dagegen verhält sich Data Augmentation, wenn einer Komponente wenige Muster zugeordnet werden. Dann ist die Wahrscheinlichkeitsmasse der Posterioriverteilungen auf einen großen Bereich des Parameterraums verteilt und daher die Wahrscheinlichkeit, dem Maximum-Likelihood-Schätzer ähnliche Parameter zu erzeugen, gering (siehe Abbildung 5.14).

Wird daraufhin im P-step eine große Varianz und ein Erwartungswert gesamplet, der vom Mittelwert der zugeordneten Daten weit entfernt liegt, so ist die Likelihood dieser Daten bezüglich der neuen Parameter, d.h. die Wahrscheinlichkeit $\mathcal{P}(x|\mu_j, \Sigma_j)$, gering. Dadurch wird im darauf folgenden I-step der Wert der Zugehörigkeitswahrscheinlichkeit h_{ij} für die bisher zugeordneten Muster klein, es sei denn, die Likelihood des Musters bezüglich aller anderen Komponenten der Mischverteilung ist ebenso verschwindend gering.

Es sind also zwei Fälle zu unterscheiden: a) eine Komponente, der wenige Muster zugeordnet werden, beschreibt überwiegend solche Daten, die von keiner anderen Komponente

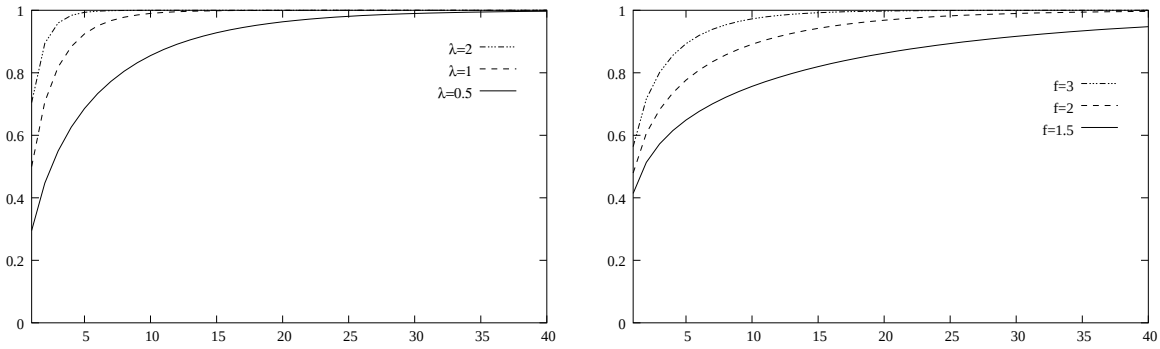

Abbildung 5.14: Links: Wahrscheinlichkeit, in Abhängigkeit von der Anzahl Daten in einem P-step einen Erwartungswert zu samplen, der nicht mehr als λ mal der Standardabweichung der Daten vom Mittelwert der Daten entfernt ist. Rechts: Wahrscheinlichkeit, in Abhängigkeit von der Anzahl Daten eine Varianz zu samplen, die nicht weiter als f mal größer als die empirische Varianz der Daten ist. Die Anzahl Daten ist jeweils auf der x-Achse abgetragen. Mit zunehmender Anzahl Daten wächst die Wahrscheinlichkeit deutlich, Parameter zu samplen, die den empirischen Größen ähnlich sind.

erklärt werden, z.B. weit außerhalb liegende Muster, oder b) die Komponente liegt in einem Bereich, in dem auch andere Komponenten eine deutlich von Null verschiedene Dichte besitzen. Im ersten Fall ist die Wahrscheinlichkeit hoch, dass der Komponente in den I-steps aufeinander folgender Iterationen die gleichen Muster zugeordnet werden und daher wird diese Komponente nicht verschwinden, d.h. ihr Mischgewicht wird von Null entfernt bleiben. Im zweiten Fall dagegen werden der Komponente immer weniger Muster zugeordnet, wodurch neu gesamplete Varianzen immer größer und Erwartungswerte immer weniger spezifisch bezüglich der noch zugeordneten Daten werden. Dieser Effekt verstärkt sich i.d.R., bis der Komponente irgendwann überhaupt keine Muster mehr zugeordnet werden. Abbildung 5.12 zeigt eine solche Entwicklung anhand der erzeugten GMMs. Aus den Abbildungen 5.10 und 5.11 kann man die beschriebene Entwicklung der Varianzen und Standardabweichungen ablesen.

5.3 Generalisierung und Overfitting

5.3.1 Anforderungen an ein Lernverfahren

Ein Kerngedanke des Maschinellen Lernens ist das Prinzip der *Generalisierung*, d.h. der Verallgemeinerung der Erkenntnis, die man aus einer Trainingsstichprobe erhalten hat. Dieser Sachverhalt wird in der Literatur als die Fähigkeit eines Modells oder Lernverfahrens definiert, nicht nur die Trainingsdaten gut zu beschreiben, sondern auch unbekannte Daten aus der selben Datenquelle. In [MacKay 03] heißt es im Kontext von überwachtem Lernen:

> *Generalisierung bedeutet es, von den beobachteten Daten auf den Wert t_{N+1} an einer neuen Stelle x_{N+1} zu schließen.*

Ähnlich heißt es im Bezug auf die Modellklasse der neuronalen Netze in [Mitchell 97]:

> *(...) die Generalisierungsgüte des Netzwerkes – die Güte, mit der es Beispiele jenseits der Trainingsdaten beschreibt.*

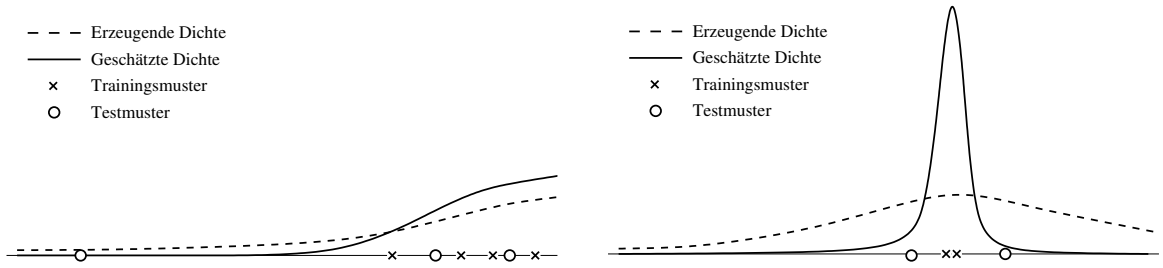

Abbildung 5.15: Typische Overfitting-Situationen, die bei univariater Dichteschätzung auftreten können. Links: die langschwänzige erzeugende Verteilung wird durch eine Dichte approximiert, die nach Außen hin rasch verschwindet; dadurch hat das Testmuster links außen bezüglich der gelernten Dichte eine extrem kleine Likelihood. Rechts: Die gelernte Dichte hat einen deutlichen Peak an einer Stelle, an der in der Trainingsmenge (durch Zufall) einige Trainingsmuster eng beieinander liegen und überschätzt damit die erzeugende Dichte deutlich. Die Likelihood auf der Trainingsmenge ist dadurch zu groß.

Das Generalisierungsprinzip wird ferner im Rahmen der statistischen Lerntheorie [Vapnik 95] axiomatisiert.

Eine schlechte Generalisierungsleistung wird auch als *Overfitting* bezeichnet. Im Rahmen der Dichteschätzung ist es charakterisiert durch Situationen, in denen die gegebene Trainingsmenge eine große Likelihood besitzt, eine davon unabhängige Testmenge aus der selben Datenquelle aber nur eine geringe Likelihood.

Es lassen sich bei der Dichteschätzung zwei extreme Formen von Overfitting beschreiben, die in Abbildung 5.15 exemplarisch skizziert sind. In beiden Fällen ist die geschätzte Dichte gegenüber der erzeugenden Dichte zu kompakt, was zu einer Überschätzung der Dichte in den Bereichen führt, in denen Trainingsdaten zu finden sind, und zu einer Unterschätzung in den anderen Bereichen. In beiden Fällen sind also die Varianzen des GMMs zu klein.

Die beiden in Abbildung 5.15 dargestellten Situationen zeigen aber auch, dass Overfitting dann auftritt, wenn die Trainingsstichprobe nicht aussagekräftig genug ist, um die erzeugende Verteilung vollständig zu beschreiben: in der linken Teilabbildung sind keinerlei Trainingsmuster im langschwänzigen Teil der erzeugenden Verteilung vorhanden, in der rechten Teilabbildung konzentrieren sich die Trainingsmuster auf den mittleren Teil der erzeugenden Verteilung, während die äußeren Bereiche unbesetzt bleiben. Overfitting entsteht also, wenn wenige Trainingsdaten untypisch angeordnet sind. Im Umkehrschluss ist die Generalisierung besser, wenn die erzeugende Verteilung durch eine größere Anzahl Trainingsdaten beschrieben wird, da untypische Anordnungen der Trainingsdaten dann seltener sind.

Ein Lernverfahren, das gute Generalisierung ermöglichen soll, muss also darauf achten, (i) Situationen mit zu wenigen Trainingsdaten zu vermeiden und (ii) falls das nicht möglich ist, eine vorsichtige Schätzung der Varianz vorzunehmen, d.h. eher eine zu große Varianz zu schätzen als eine zu kleine. Beim Lernen von GMMs kommt uns zugute, dass sich das GMM gemäß Abschnitt 2.4.3 in seine Komponenten zerlegen lässt, so dass sich die Anzahl Trainingsdaten pro Komponente durch Entfernen einer Komponente beeinflussen lässt.

Angenommen, wir möchten den Varianzparameter für eine Datenmenge bestimmen, deren empirische Varianz 1 ist. Die Posterioriverteilung der Varianzen bei nichtinformativen Prioriverteilungen ist eine inverse Gammaverteilung, deren Breite von der Anzahl Daten-

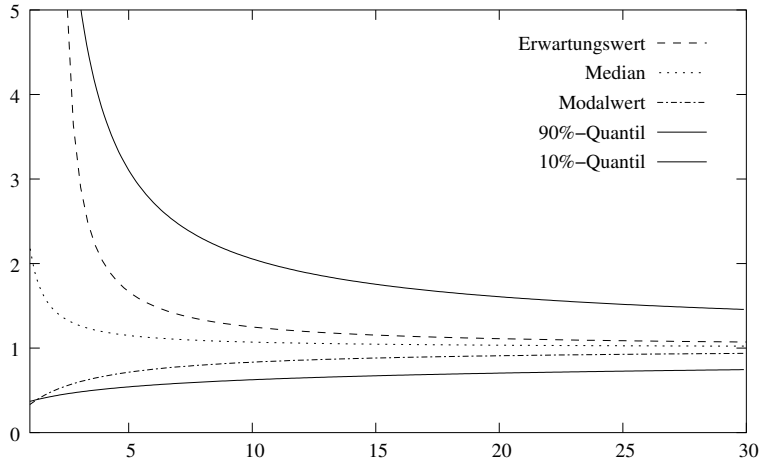

Abbildung 5.16: Erwartungswert, Median und Modalwert der Posteriori-Dichte für die Varianzen in Abhängigkeit von der Anzahl Trainingsdaten. Die empirische Varianz der Daten ist zu 1 angenommen worden. Beachte, dass der Erwartungswert für $n \leq 3$ nicht existiert.

punkte abhängt. Abbildung 5.16 stellt den Modalwert, den Median und den Erwartungswert dieser Verteilung in Abhängigkeit von der Anzahl Datenpunkte dar.

Für große Datenmengen ist die Posterioriverteilung eng um den tatsächlichen Wert von 1 konzentriert, für kleine Datenmengen dagegen ist die Verteilung weit über die positive Halbachse gestreut. Auffällig ist, dass der Modalwert für kleine Datenmengen an den linken Rand der Verteilung wandert, während Median und Erwartungswert große Werte annehmen. Der Modalwert ist daher gerade bei kleinen Trainingsmengen keine geeignete Größe zur Beschreibung der Posterioriverteilung. Dies ist der Grund, weshalb der Maximum-a-posteriori-Ansatz, der als Schätzer den Modalwert der Posterioriverteilung liefert, für kleine Trainingsmengen und nichtinformative Prioriverteilungen eine schlechte Generalisierung erzielt. Ein gegen Overfitting weniger anfälliges Lernverfahren muss also in einer Situation, in der nur wenige Daten zum Schätzen des Varianzparameters verfügbar sind, nicht nur den Modalwert der Posterioriverteilung betrachten, sondern die gesamte Breite der Verteilung.

5.3.2 Generalisierung und Data Augmentation

Im letzten Abschnitt wurden die Anforderungen an ein Lernverfahren beschrieben, um gute Generalisierung zu erreichen. Nun soll untersucht werden, inwieweit Data Augmentation diese Anforderungen erfüllt, insbesondere bei der Verwendung nichtinformativer Prioriverteilungen.

In Abschnitt 5.2.2 wurde bereits beschrieben, in welchen Situationen im Laufe von Data Augmentation mit nichtinformativen Prioriverteilungen eine Komponente verschwindet: dann, wenn der Komponente an sich bereits wenige Muster zugeordnet werden und gleichzeitig andere Komponenten die betreffenden Muster ebenfalls erklären könnten, sprich, andere Komponenten besitzen in dem betroffenen Bereich eine deutlich von Null verschiedene Dichte. Die verschwindende Komponente erklärt also keinen wesentlichen Teil der gesamten Verteilung. Durch das Verschwinden dieser Komponente wird einer Überanpassung der

Komponente auf einen unwesentlichen Teil der Verteilung vorgebeugt. Gleichzeitig verringert sich durch Entfernen dieser Komponente aus dem GMM die Modellkomplexität, was im Einklang mit dem philosophischen Prinzip von *Occam's razor*[3] steht.

Werden einer Komponente dagegen zwar nur wenige Trainingsmuster zugeordnet, diese aber von keiner anderen Komponente erklärt, bleibt die Komponente erhalten, da ihr stets Muster zugeordnet werden. Diese Komponente ist also – auch im Sinne von Occam's razor – ein notwendiger Teil des GMM. Gemäß Abbildung 5.16 ist die Posterioriverteilung der Varianzen aber breit gestreut und langschwänzig, so dass mit großer Wahrscheinlichkeit große Varianzen gesamplet werden. Eine Überanpassung als Folge zu kleiner Varianzen tritt also nur selten auf, da sowohl der Median als auch der Erwartungswert der Posterioriverteilung größer als die empirische Varianz der Daten sind und sie für kleine Trainingsmengen sogar noch zunehmen. Der Vergleich der Daten-Log-Likelihood auf den Trainings- und Testdaten in dem Fallbeispiel aus Abschnitt 5.2 (siehe Abbildung 5.13) bestätigt diese Aussage: die beiden Kurven laufen weitgehend parallel, ein Auseinanderdriften als typisches Anzeichen von Overfitting ist nicht zu erkennen.

5.3.3 Umgang mit Singularitäten

In einigen Fällen können Singularitäten auftreten, wenn die empirische Varianz der Daten, die einer Komponente zugeordnet werden, bereits singulär ist, d.h. wenn die Varianz/ Kovarianz-Matrix nicht vollen Rang hat. Ein derartiger Fall stellt eine extreme Form des Overfitting dar. Er tritt immer dann auf, wenn die d-variaten Daten, die einer Komponente zugeordnet werden, in einer $(d-1)$-dimensionalen Hyperebene liegen.

Bei univariaten Daten tritt dieser Fall auf, wenn entweder nur ein Muster zugeordnet wird, oder wenn alle zugeordneten Muster identisch sind. Zwar ist der Fall identischer Muster theoretisch fast sicher auszuschließen, da die Grundannahme ist, dass die Trainingsdaten aus einer Verteilung mit Dichte stammen (vergleiche Abschnitt 2.1), er tritt jedoch in der praktischen Anwendung aufgrund von Diskretisierungsfehlern und dem Arbeiten mit Daten, die aus einem möglicherweise zwar großen, aber doch diskreten Raum stammen, häufig auf.

Ist die empirische Varianz singulär und wurden nichtinformative Prioriverteilungen verwendet, ist auch die Posterioriverteilung unecht. Ein Samplen und eine stochastische Beschreibung ist daher nicht möglich. Eine Möglichkeit mit einer solchen Situation umzugehen wäre es, diesen Fall wie eine Komponente zu behandeln, der keine Daten zugeordnet wurden, d.h. diese Komponente zu entfernen. Eine solche Löschstrategie würde in vielen praktischen Anwendungen allerdings zu unangemessenen Ergebnissen führen, da das Zusammenfallen mehrerer Muster in einem niedrigdimensionalen Hyperraum auch eine typische Ausprägung der gegebenen Aufgabe sein kann.

Abbildung 5.17 zeigt exemplarisch eine Verteilung von Messwerten, die einer Studie für die *Bundesforschungsanstalt für Ernährung* entnommen ist [Lauer 01b]. Die dargestellte Verteilung weist zwei bevorzugte Stellen auf, während der sonstige Verlauf glatt ist. Die Verteilung ist also weder durch ein kontinuierliches Modell, noch durch ein diskretes beschreibbar.

Möchte man dennoch ein GMM für diese Verteilung trainieren, sollte dieses die Sprünge

[3] Occam's razor, *„Pluralitas non est ponenda sine neccesitate“* (Vielfalt darf nicht postuliert werden, außer wenn notwendig). Philosophisches Prinzip, einfachen Erklärungen den Vorzug vor komplizierteren zu geben, wenn immer möglich. Nach William von Ockham (ca. 1285–1349), Franziskaner, Philosoph.

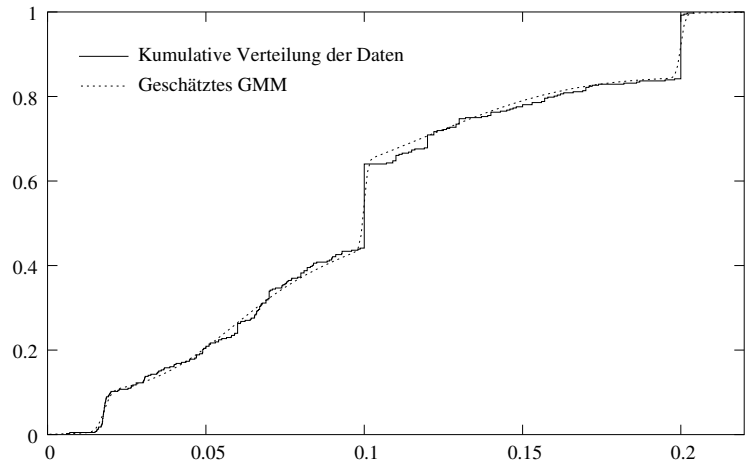

Abbildung 5.17: Kumulative Verteilung einer Messwertreihe zur radioaktiven Belastung von Lebensmitteln. Die Verteilung weist zwei bevorzugte Stellen (0.1 und 0.2) auf, ansonsten hat sie kontinuierlichen Charakter. Insgesamt wurden 392 Muster verwendet. Das gelernte GMM besteht aus fünf Komponenten.

in der kumulativen Verteilung durch je eine Komponente mit geringer Varianz modellieren. Andererseits dürfen derlei Komponenten mit geringer Varianz an keiner anderen Stelle auftreten, möchte man kein Overfitting riskieren.

Ein Ausweg aus dieser Situation wäre die Verwendung echter Prioriverteilungen, die auch bei Singularitäten in den Daten echte Posterioriverteilungen erzwingen. Eine massive Beeinflussung des Lernergebnisses, wie in Abschnitt 5.2.1 geschildert, ist allerdings unerwünscht. Die Posterioriverteilung (4.21) sollte kaum beeinflusst werden, wenn die empirische Varianz $\hat{S}$ regulär ist, aber dennoch im Fall einer singulären Größe $\hat{S}$ eine positiv definite Größe $\hat{\Lambda}$ besitzen.

Dies wird erreicht, indem man für Λ^{-1} eine positiv definite Matrix mit sehr kleinen Eigenwerten einsetzt: ist $\hat{S}$ positiv definit, so wird i.d.R. der Term $n\hat{S}$ die Formel (4.25) dominieren und der Einfluss von Λ^{-1} kann vernachlässigt werden, während bei einer singulären Matrix $\hat{S}$ durch die Addition mit der positiv definiten Matrix Λ^{-1} wiederum eine positiv definite Matrix entsteht.

Die Verwendung von Prioriverteilungen dieser Form führt dazu, dass im Fall einer Komponente, der nur identische Muster zugeordnet werden, Varianzen/Kovarianzmatrizen gesamplet werden die stets positiv definit sind. Das in Abbildung 5.17 eingetragene GMM verdeutlicht diesen Sachverhalt: die Sprungstellen in den Daten werden vom gelernten GMM leicht geglättet, sind aber als Sprung dennoch erkennbar. Die Stärke der Glättung ist von der Anzahl Trainingsdaten an den betreffenden Stellen abhängig.

5.4 Modifikationen von Data Augmentation

5.4.1 Motivation

In den vorangegangenen Abschnitten wurde das Verhalten von Data Augmentation bei Verwendung verschiedener Prioriverteilungen beschrieben. Es wurde deutlich, dass Data Augmentation im Gegensatz zu klassischen Lernalgorithmen, z.B. dem EM-Verfahren, nicht eine einzige Lösung liefert, sondern eine randomisierte Folge suboptimaler Lösungen. Ziel dieser Arbeit ist es jedoch, Data Augmentation zur Dichteschätzung einzusetzen, d.h. zur Berechnung eines einzigen möglichst guten Parametervektors, wie in Kapitel 2 beschrieben.

Dadurch wird es notwendig, die erzeugte Folge von Parametern weiterzuverarbeiten und eine Auswahl des besten Parametervektors durchzuführen. Hierbei sollen nicht nur GMMs einer vorgegebenen Größe betrachtet werden, sondern auch die Anzahl der Komponenten variiert werden. Dabei ist auf gute Generalisierung und die Vermeidung von Overfitting zu achten. In Abschnitt 5.3.2 wurde dargestellt, dass das Verhalten von Data Augmentation mit nichtinformativen Prioriverteilungen gerade im Hinblick auf die Vermeidung von Overfitting eine interessante Alternative zu klassischen Lernverfahren ist.

Den in [Lauer 02] dargestellten Ideen folgend soll in den nachfolgenden Abschnitten aus dem Data Augmentation Ansatz ein neues Verfahren entwickelt werden, das GMMs lernt im Sinne der Berechnung eines besten Parametersatzes.

- Wie kann aus vielen zufälligen, guten Lösungen der gesampleten Parameterfolge ein bester Schätzer konstruiert werden?
- Wie kann unter den vielen möglichen Lösungen einer ausgewählt werden?
- Wie kann eine geeignete Größe des GMM bestimmt werden?
- Wie sind die Prioriverteilungen zu wählen?

5.4.2 Prioriverteilungen und das Wegfallen von Komponenten

Aufgrund der in Abschnitt 5.3 diskutierten Aspekte der guten Generalisierung und der Vermeidung von Singularitäten wird hier die Wahl folgender Prioriverteilungen vorgeschlagen:

- nichtinformative Prioriverteilung für die Mischgewichte
- nichtinformative Prioriverteilung für die Erwartungswerte
- echte Prioriverteilungen für die Varianzen, die jedoch nur einen vernachlässigbaren Einfluss auf die Posterioriverteilungen besitzen. Zur Vereinfachung der Sprechweise sollen solche Prioriverteilungen im Weiteren als *wenig informative Prioriverteilungen* bezeichnet werden.

Ein Beispiel für eine solche wenig informative Prioriverteilung erhält man bei der Wahl der Parameter $\eta = 0$, $q = 0$ und $\Lambda = 50 \cdot \hat{S}^{-1}$, wobei $\hat{S}$ die empirische Varianz aller Trainingsdaten ist, vorausgesetzt diese Varianz ist positiv definit[4]. Durch die Abhängigkeit

[4] Andernfalls lässt sich eine positiv definite Größe durch Addition von εI_d auf $\hat{S}$ für kleines $\varepsilon > 0$ erzielen

```
 1.   wähle initiale k, (w_1, ..., w_k, μ_1, ..., μ_k, Σ_1, ..., Σ_k)
 2.   wiederhole für t = 0, 1, 2, ...
 3.      wiederhole, bis keine Komponenten mehr gelöscht werden
            IMPUTATION-STEP (I-STEP):
 4.         wiederhole für jedes Trainingsmuster x_i
 5.            wiederhole für jede Komponente j
 6.               berechne h_ij gemäß (4.40)
 7.            sample Zuordnung z_i gemäß (4.41)
            DELETION-STEP (D-STEP):
 8.         falls Komponenten mit zu wenigen Zuordnungen existieren
 9.            lösche eine der Komponenten mit zu wenigen Zuordnungen
10.            verkleinere den Parametervektor entsprechend
11.            vergrößere die Mischgewichte der verbleibenden Komponenten
12.            verringere k um 1
         POSTERIOR-STEP (P-STEP):
13.      sample neue Mischgewichte (w_1, ..., w_k) gemäß (4.17)
14.      wiederhole für jede Komponente j
15.         sample neue Kovarianzmatrix Σ_j gemäß (4.21)
16.         sample neuen Erwartungswert μ_j gemäß (4.21)
```

Abbildung 5.18: Data Augmentation Algorithmus für GMMs, erweitert um das Löschen von Komponenten (D-step). Imputation- und Deletion-step werden so lange wiederholt, bis allen Komponenten ausreichend viele Muster zugeordnet werden. Erst anschließend werden neue Parameter gesamplet. Die Vergrößerung der Mischgewichte (Zeile 11) ist notwendig, damit die Summe der Mischgewichte wieder 1 ergibt.

zwischen Λ und $\hat{S}$ ist diese Prioriverteilung invariant gegenüber Skalierung der Trainingsdaten. Der Faktor 50 ist willkürlich, darf allerdings nicht zu klein gewählt werden, um keinen zu starken Einfluss der Prioriverteilungen zu riskieren. 50 hat sich in Experimenten als i.d.R. ausreichend groß herausgestellt.

Durch die Verwendung unechter Prioriverteilungen kommt es zwangsläufig zu Situationen, in denen eine Komponente alle Zuordnungen verliert. In Abschnitt 5.3 wurde beschrieben, dass eine solche Situation in Fällen auftritt, in denen das GMM größer als notwendig ist und dass zur Vermeidung von Overfitting die Entfernung dieser Komponente aus der Mischverteilung ratsam ist. Data Augmentation soll daher um eine Routine erweitert werden, die Komponenten mit zu wenigen Zuordnungen aus dem GMM entfernt. Da zur Schätzung einer d-variaten Kovarianzmatrix mindestens $d + 1$ Datenpunkte erforderlich sind, wird eine Komponente dann entfernt, wenn nicht mehr als d Datenpunkte zugeordnet werden.

Abbildung 5.18 zeigt die erweiterte Form von Data Augmentation, Abbildung 5.19 das zugehörige Ablaufdiagramm. Nach dem Imputation-step wird zunächst überprüft, ob allen Komponenten genug Daten zugeordnet wurden. Wenn ja, wird wie gehabt der Posterior-step durchgeführt. Andernfalls wird eine der Komponenten mit zu wenigen Zuordnungen aus dem GMM entfernt, d.h. der Parametervektor wird um das entsprechende Mischgewicht, den Erwartungswert und die Varianz/Kovarianzmatrix verkleinert. Außerdem müssen die

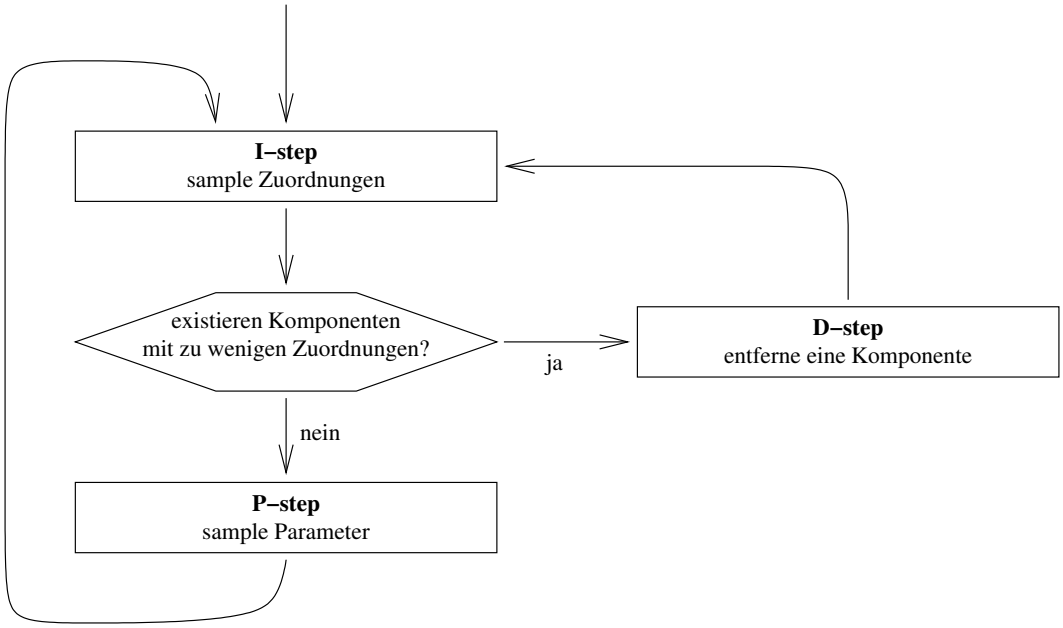

Abbildung 5.19: Ablaufdiagramm Data Augmentation erweitert um den D-step zum Entfernen einer Komponente mit zu wenigen Zuordnungen.

Mischgewichte der anderen Komponenten durch Multiplikation mit $\frac{1}{1-w_D}$ reskaliert werden, damit die Summe der Mischgewichte wieder 1 ergibt. Dabei bezeichnet w_D das vormalige Mischgewicht der entfernten Komponente. Anschließend wird erneut der Imputation-step durchgeführt. Dadurch erhalten auch die Daten, die zuvor der entfernten Komponente zugeordnet waren, wieder eine konsistente Zuordnung. Alternativ ist es möglich, die weiterhin konsistenten Zuordnungen beizubehalten und nur die inkonsisten gewordenen neu zu samplen.

Durch den D-step ist der Algorithmus in der Lage, überflüssige Komplexität aus dem Modell zu entfernen. Ein Ergänzen zusätzlicher Komponenten ist nicht möglich. Das bedeutet, dass das Verfahren mit einer großen, unter Umständen viel zu großen Mischverteilung gestartet wird und anschließend die Modellgröße monoton fällt. Dies bedeutet auch, dass eine fälschlicherweise gelöschte Komponente nicht wieder hergestellt werden kann. Der Algorithmus implementiert also keine Markov-Kette über verschiedene Modell-Größen, wie ein Reversible-jump-MCMC Ansatz. Lediglich in den Zeiten, in denen keine Komponente gelöscht wird, kann sein Verhalten durch eine Markov-Kette beschrieben werden. Das Entfernen einer Komponente entzieht sich der Markovschen Modellierung.

Andererseits garantiert die monoton fallende GMM-Größe, dass sich die Modellkomplexität nicht ständig ändert, sondern irgendwann eine untere Grenze erreicht wird, spätestens wenn nur noch eine Komponente vorhanden ist. Mit jeder Reduktion der Modellkomplexität verringert sich die Dynamik der Markov-Kette, da die gleiche Anzahl Daten auf weniger Komponenten verteilt wird. Dadurch konzentrieren die Posterioriverteilungen ihre Wahrscheinlichkeitsmasse auf einen engeren Bereich, so dass gesamplete Parameter sich mit größerer Wahrscheinlichkeit ähnlich sind. Somit sinkt auch die Wahrscheinlichkeit, dass

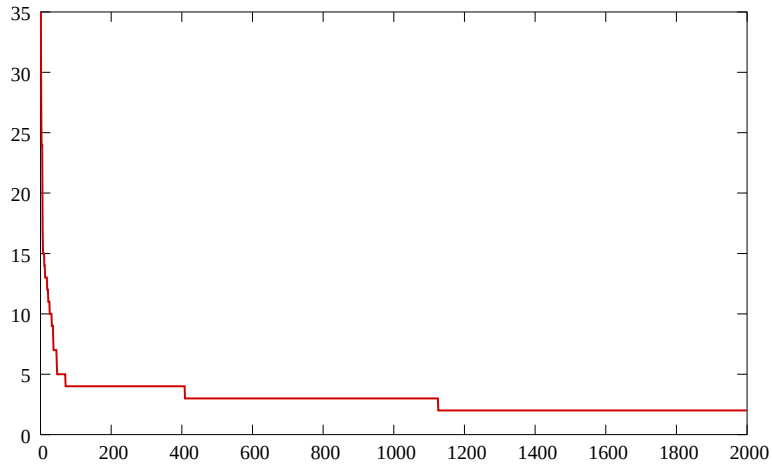

Abbildung 5.20: Entwicklung der GMM-Größe im Zuge von Data Augmentation mit Löschen von Komponenten. Auf der x-Achse sind die Anzahl Iterationen (Anzahl P-steps) aufgetragen, auf der y-Achse die Größe des GMMs. Gestartet wurde mit 200 zufällig angeordneten Komponenten. Trainiert wurde auf 200 gleichverteilten univariaten Trainingsdaten. Da innerhalb der ersten Iteration bereits die meisten Komponenten entfernt wurden, ist die y-Achse auf das Intervall von 0 bis 35 beschränkt.

weitere Komponenten verschwinden.

Abbildung 5.20 zeigt die Größenentwicklung bei einem Lauf Data Augmentation, der mit einem GMM aus 200 zufällig angeordneten Komponenten gestartet wurde. Nach wenigen Iterationen sind nur noch vier Komponenten übrig geblieben, die restlichen wurden wegen zu geringer Anzahl Zuordnungen entfernt. Die Trainingsdaten bestanden aus 200 im Intervall $[0, 1]$ gleichverteilten univariaten Daten, wie im Fallbeispiel aus Abschnitt 5.2. Abbildung 5.21 zeigt die Entwicklung der Erwartungswerte ab Iteration Nr. 47, ab der das GMM nur noch aus maximal fünf Komponenten besteht.

5.4.3 Optimierung der Einzellösungen

Die bisher beschriebene Vorgehensweise liefert uns eine Folge von GMMs, die gemäß der Posterioriwahrscheinlichkeit verteilt sind, nicht jedoch einen besten Schätzer. Die Posterioriverteilung bevorzugt zwar die guten Parameterbereiche, lässt allerdings auch andere Werte zu, so dass ein einzelner gesampleter Parameterwert – außer in seltenen Fällen – nicht optimal sein wird. Wie kann also aus der Abfolge gesampleter suboptimaler Parameter eine bessere Lösung berechnet werden?

Zunächst soll der P-step isoliert betrachtet werden unter der Annahme, dass jeder Komponente immer die selben Muster zugeordnet werden. Im Wesentlichen wird das Verhalten des Verfahrens in diesem Fall von der Form der Posterioriverteilungen bestimmt, wie sie in Abschnitt 4.2.2 beschrieben wurden.

Mischgewichte Mit nichtinformativen Prioriverteilungen ergibt sich die Posterioriverteilung für die Mischgewichte zu einer Dirichletverteilung $D(n_1, \ldots, n_k)$, wobei n_j die Anzahl

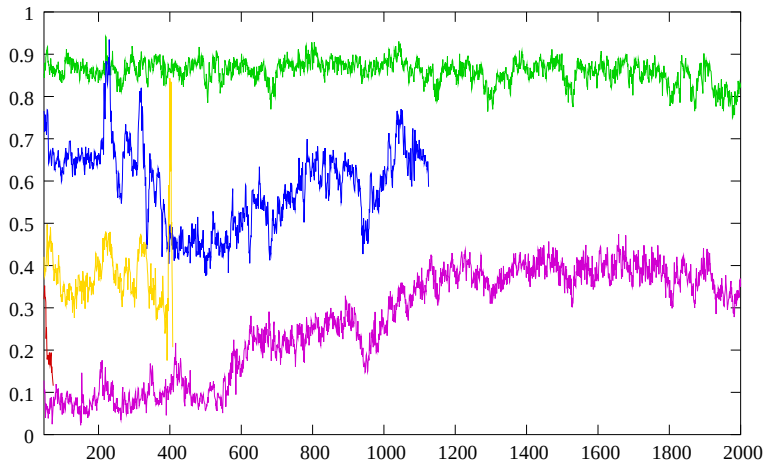

Abbildung 5.21: Entwicklung der Mittelwerte der verbliebenen fünf Komponenten im Zuge von Data Augmentation mit Löschen von Komponenten. Auf der x-Achse sind die Anzahl Iterationen (Anzahl P-steps) aufgetragen, beginnend mit Iteration Nr. 47, auf der y-Achse die Erwartungswerte der GMM-Komponenten. Trainiert wurde auf 200 gleichverteilten univariaten Trainingsdaten.

Daten bezeichne, die der j-ten Komponente zugeordnet sind. Der Erwartungswert dieser Verteilung ist (vgl. Anhang B) $(\frac{n_1}{n_1+\cdots+n_k}, \ldots, \frac{n_k}{n_1+\cdots+n_k})$. Dies ist zugleich der Maximum-Likelihood-Schätzer für die Auswahlwahrscheinlichkeiten. Liegt uns dagegen nicht die Posterioriverteilung als solche vor sondern nur eine Stichprobe daraus, so können wir den Erwartungswert durch den Mittelwert der Stichprobe approximieren.

Varianzen/Kovarianzmatrizen Die Posterioriverteilungen für die Varianzen bzw. Kovarianzmatrizen wurden in Abschnitt 5.3.2 bereits untersucht. Im univariaten Fall ist der Erwartungswert der Posterioriverteilung, sofern er überhaupt existiert, für eine kleine Anzahl zugeordneter Daten viel größer als der Maximum-Likelihood-Schätzer $\hat{s}^2$. Ebenso gilt im multivariaten Fall, dass die erwarteten Varianzen in jeder Koordinatenrichtung größer sind als der Maximum-Likelihood-Schätzer $\hat{S}$. Der Erwartungswert ist also nicht anfällig gegen Overfitting.

Wurden einer Komponente viele Daten zugeordnet, unterscheiden sich der Erwartungswert der Posterioriverteilung und der Maximum-Likelihood-Schätzer kaum, denn es gilt für $n \geq d + 2$:

$$E[\Sigma|x_1, \ldots, x_n] = \frac{1}{\hat{q} - d - 1}\hat{\Lambda}^{-1} \approx \frac{n}{n - d - 1}\hat{S} \underset{n\to\infty}{\longrightarrow} \hat{S} \tag{5.1}$$

In allen Fällen ist der Erwartungswert der Posterioriverteilung, sofern er existiert, ein geeigneter Schätzer für die Varianz/Kovarianzmatrix.

Erwartungswerte Die Posterioriverteilung der Erwartungswerte ist eine Verteilung, die nach Konstruktion symmetrisch um den Maximum-Likelihood-Schätzer $\bar{x}$ ist. Ihr Erwartungswert existiert bei kleiner Anzahl zugeordneter Muster nicht, sondern nur für größere

Anzahl. Im Fall der Existenz des Erwartungswertes ist dieser aus Symmetriegründen identisch mit $\bar{x}$.

Existiert der Erwartungswert der Posterioriverteilung, kann er durch den Mittelwert einer Stichprobe der Verteilung abgeschätzt werden. Ist die Anzahl der zugeordneten Muster dagegen klein, so ist die erwartete Varianz der Komponente nach vorangegangener Überlegung sehr groß, d.h. die Komponente entartet zu einer sehr flachen, breiten Wahrscheinlichkeitsdichte. In einem solchen Fall spielt der Erwartungswert der Komponente kaum eine Rolle, da die Dichte nahezu konstant ist. Der Nachteil durch einen ungünstig gewählten Erwartungswert ist also gering, so dass wir auch in diesem Fall willkürlich den Mittelwert eines Samples aus der Posterioriverteilung als Erwartungswert der Komponente setzen können.

Bei der ausschließlichen Betrachtung des P-steps können wir aus einem Sample von Parametern aus der Posterioriverteilung durch Mittelwertbildung einen i.d.R. besseren Parametervektor berechnen. Diese Ergebnisse konnten auch empirisch bestätigt werden. Siehe dazu Kapitel 7.2.

P- und I-step Durch die Verzahnung von P-step und I-step entsteht das letztendliche Verhalten von Data Augmentation. Dadurch wird eine Markov-Kette gebildet, deren stationäre Verteilung die Posterioriverteilung der Parameter gegeben die Daten bildet. Das bedeutet, die Markov-Kette hält sich in den Bereichen großer Posterioriwahrscheinlichkeit häufiger auf als in den Bereichen niedriger Wahrscheinlichkeit. Insbesondere ist die Wahrscheinlichkeit gering, einen Bereich mit großer Posterioriwahrscheinlichkeit zu verlassen, sobald sich die Markov-Kette in diesen Bereich hineinbewegt hat. Es werden also hintereinanderfolgend Parameter aus den selben Bereichen großer Wahrscheinlichkeit gezogen. Durch Anwendung eines gleitenden Durchschnitts über die gesampleten Parameter können wir somit einen besseren Schätzer erzielen als durch das Samplen alleine.

Die Abbildung 5.22 zeigt den Vergleich der Daten-Log-Likelihood auf Trainings- und Testmenge mit und ohne Mittelung der GMM-Parameter. Deutlich ist einerseits der glattere Verlauf bei Mittelung als auch die insgesamt höhere Likelihood zu erkennen, gleichermaßen auf der Trainings- wie auch der Testmenge. Bei der Abbildung ist zu beachten, dass während der Trainingsphase das GMM von 35 auf 2 Komponenten reduziert wurde. Dies erklärt die sprunghaften Veränderungen z.B. in Iteration 1125.

Abbildung 5.23 zeigt einen Vergleich der gemittelten Werte, wenn unterschiedlich lange gleitende Durchschnitte verwendet werden. Je kürzer die Länge des gleitenden Durchschnitts, d.h. über je weniger Iterationen gemittelt wird, desto stärker schwanken die Werte. Andererseits führen zu große Längen der gleitenden Durchschnitte dazu, dass mitunter nicht nur über Parameter aus einem Bereich hoher Wahrscheinlichkeit sondern aus mehreren Bereichen gemittelt wird, so dass die resultierenden Parameter unbrauchbar sind. Dies geschieht, wenn die Dynamik der Markov-Kette zu groß ist. Empirisch hat sich eine Mittelung über 50 Iterationen bewährt.

5.4.4 Auswahl einer besten Lösung

Da Data Augmentation kein Verfahren ist, das in monotoner Weise nach dem Optimum eines Gütekriteriums sucht, kann nicht davon ausgegangen werden, dass die letzten Parameterwerte vor dem Abbrechen die besten sind. Im Gegenteil: die Markov-Kette kann sich zum

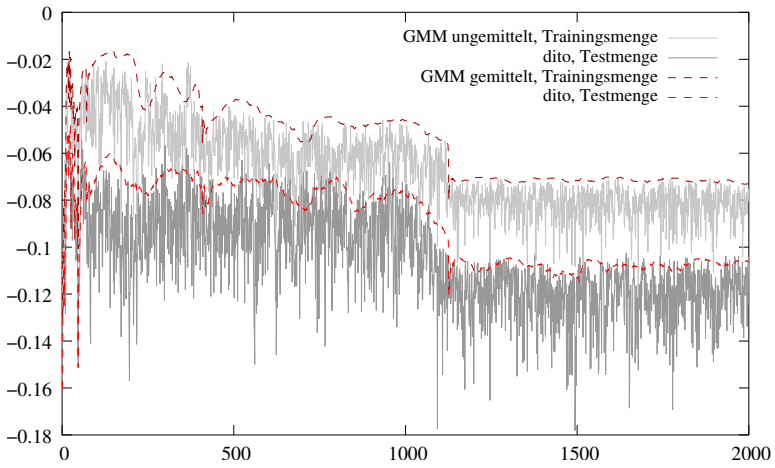

Abbildung 5.22: Vergleich der Daten-Log-Likelihood auf Trainings- und Testmenge mit und ohne Mittelung der GMM-Parameter. Die Mittelwertbildung erfolgte über je 50 Iterationen. Das Training erfolgte auf 200 gleichverteilten univariaten Daten.

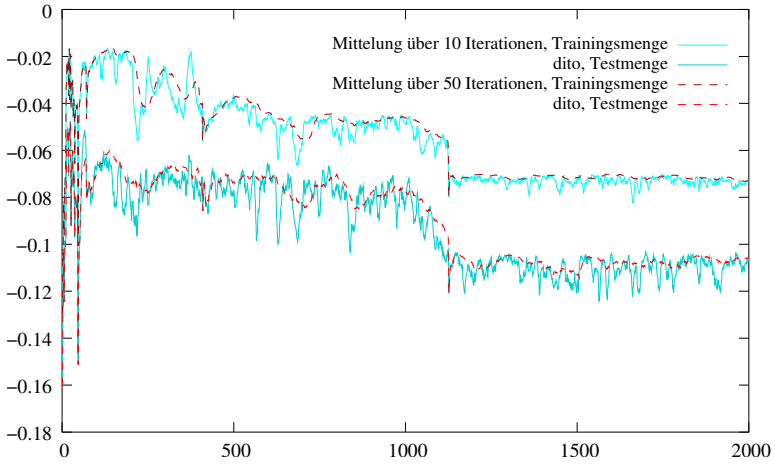

Abbildung 5.23: Vergleich der Daten-Log-Likelihood auf Trainings- und Testmenge mit Mittelung der GMM-Parameter. Verwendet wurde eine schwache Glättung durch Mittelung über 10 Iterationen und eine mäßig starke Glättung durch Mittelung über 50 Iterationen.

Abbruchzeitpunkt in gleicher Weise in einem besonders guten, aber auch in einem besonders schlechten Bereich des Parameterraums befinden. Außerdem kommt hinzu, dass das Löschen einer Komponente ein irreversibler Vorgang ist. Wurde zufällig eine Komponente zu viel entfernt, so kann die beste Lösung gar nicht mehr erreicht werden.

Abbildung 5.22 verdeutlicht diesen Sachverhalt anhand eines Beispiels. Mit dem Entfernen der drittletzten Komponente in Iteration Nr. 1125 bricht die Likelihood deutlich ein, sowohl auf der Test- wie auch auf der Trainingsmenge. Ein, wenn auch nicht so ausgeprägter, Einbruch ist in Iteration Nr. 408 zu erkennen, in der die viertletzte Komponente entfernt wird.

Wir benötigen also ein Kriterium, um festzustellen, welche der berechneten Lösungen als die beste anzusehen ist. Um eine möglichst breite Datengrundlage für das Training zur Verfügung zu haben, soll auf die Verwendung einer Validierungsmenge verzichtet werden. Damit entfällt die Möglichkeit einer empirischen Validierung, wie in Abschnitt 4.3.2 skizziert.

Wir machen uns zu Nutze, dass wir a) durch das zufällige Samplen von Parametern und b) durch die Bildung eines gleitenden Durchschnitts ein Overfitting auf den Trainingsdaten weitgehend ausgeschlossen haben. Wie in Abschnitt 5.3.2 diskutiert, können Parameterwerte, die zu Overfitting neigen, zwar auftreten, aber nur mit geringer Wahrscheinlichkeit, da Overfitting-Situationen zwar eine große Likelihood besitzen, sie sich aber auf sehr kleine Bereiche des Parameterraums beschränken. Daher ist die Wahrscheinlichkeitsmasse der Posterioriwahrscheinlichkeit in den Overfittig-Bereichen gering und ein Herausspringen der Markov-Kette aus einem solchen Bereich sehr wahrscheinlich. Außerdem sorgt die Mittelung dafür, dass einzelne Parameterwerte sich nicht zu stark auf die gemittelten Parameter auswirken, so dass derartige Ausprägungen durch Glättung entfernt werden.

Wir können also davon ausgehen, dass die Folge gemittelter Parameter kaum noch Neigung zum Overfitting beinhaltet. Die Likelihood auf der Trainingsmenge misst also tatsächlich die Anpassungsgüte bezüglich des Daten-erzeugenden Prozess'. Sie stellt damit ein geeignetes Auswahlkriterium dar. Die Darstellung in Abbildung 5.22 verdeutlicht dies anhand eines Fallbeispiels. Beide Kurven, die Likelihood auf der Trainingsmenge wie auch auf der Testmenge zeigen für die gemittelten Parameter einen nahezu parallelen Verlauf. Die besten Parameter auf der Trainingsmenge sind somit auch sehr gut auf der Testmenge.

Es fällt auf, dass in den ersten Iterationen, wenn das GMM viel zu groß ist, die gemittelten Werte noch sehr stark schwanken. Dies liegt daran, dass einerseits große GMMs eher zum Overfitting neigen als kleine, und zweitens, dass das fortwährende Entfernen von Komponenten die Mittelwertbildung stört. Es ist daher empfehlenswert, die ersten Iterationen bei der Modellauswahl nicht zu berücksichtigen. In den im Weiteren dargestellten Experimenten wurde jeweils eine Einschwingphase von 200 Iterationen ignoriert und erst danach die Likelihood der gemittelten Modelle berechnet.

Bei dieser Form der Modellauswahl kommt der Länge des gleitenden Durchschnitts die Rolle des Regularisierungsparameters zu. Je kürzer diese Länge gewählt wird, desto größer ist die Auswirkung einzelner Overfitting-Situationen und damit das Risiko von Overfitting. Eine zu große Länge dagegen mindert die Qualität der Lösung dadurch, dass ein Mittelwert über verschiedene Bereiche des Parameterraums berechnet wird.

Zusammenfassend wird die Mittelwertbildung und die Modellauswahl als Ergänzung zum Grundalgorithmus (Abbildung 5.18) in Abbildung 5.24 präsentiert.

```
GEGEBEN:  EINE FOLGE GESAMPLETER GMM-PARAMETER ϑ_1, ..., ϑ_ν,
          DIE LÄNGE DES GLEITENDEN DURCHSCHNITTS L,
          DIE LÄNGE DER EINSCHWINGPHASE e

 1.   wiederhole für t = e, ..., ν
 2.      stelle letzte Iteration t_p fest, zu der GMM-Größe verringert wurde
 3.      setzte t_m := max{t_p, t − L + 1}
 4.      berechne gemittelte Parameter ϑ̄_t := 1/(t−t_p+1) Σ_{t'=t_m}^{t} ϑ_{t'}
 5.      berechne Likelihood llh_t des Modells ϑ_t auf den Trainingsdaten
 6.   suche den Index des besten Modells t_b := arg max{llh_t | t = e, ..., ν}
 7.   liefere ϑ̄_{t_b}
```

Abbildung 5.24: Ansatz, um aus einer von Data Augmentation gesampleten Folge von Parametern durch Mittelung der Parameter und Vergleich der Likelihood auf der Trainingsmenge einen besten Parameter zu bestimmen.

5.5 Weitere Aspekte von Data Augmentation

5.5.1 Permutationsinvarianz

GMMs, wie sie hier betrachtet werden, besitzen Komponenten mit den gleichen Eigenschaften, d.h. die Dichteklasse (Normalverteilung), die Parametrisierung der Dichte sowie die Prioriverteilungen für die Parameter sind identisch. Dadurch entsteht eine Permutationsinvarianz bezüglich der Reihenfolge der Komponenten: zwei GMMs gleicher Größe $p(\cdot) = \sum_{j=1}^{k} w_j \varphi_{\mu_j,\Sigma_j}(\cdot)$ und $p'(\cdot) = \sum_{j=1}^{k} w'_j \varphi_{\mu'_j,\Sigma'_j}(\cdot)$ sind identisch, wenn es eine Permutation ϱ gibt, so dass für alle $j = 1, \ldots, k$ gilt: $w_j = w'_{\varrho(j)}$ und $\mu_j = \mu'_{\varrho(j)}$ und $\Sigma_j = \Sigma'_{\varrho(j)}$. Das heißt, die Komponenten sind vertauschbar.

Wegen der identischen Eigenschaften der Komponenten besitzt auch die Posterioriverteilung der Parameter die selbe Permutationsinvarianz. Das bedeutet, dass es beim Ausführen eines Samplingverfahrens wie Data Augmentation dazu kommen kann, dass die „Rolle“ der Komponenten vertauscht wird. Anhand Abbildung 5.4 kann dies beispielhaft demonstriert werden: die bevorzugten Positionen um die Werte 0.25 und 0.8 werden abwechselnd von verschiedenen Komponenten belegt. Da Data Augmentation aus der symmetrischen Posterioriverteilung samplet, ist ein solches Verhalten sogar zu erwarten. In der Literatur wird dieses Phänomen auch als *Label switching* bezeichnet.

Möchte man nicht die gesamte Parameterverteilung betrachten, sondern nur die Randverteilungen der einzelnen Parameter, so führt das Label switching Phänomen zu Schwierigkeiten: die Randverteilungen sind für alle Komponenten gleich, und damit auch die Modal- und Mittelwerte. Der Zusammenhang, wie die Komponenten zusammenwirken müssen, um eine möglichst optimale Beschreibung der Daten zu erhalten, geht verloren.

Dieses Problem wurde in der Literatur auf verschiedene Weisen gelöst: In [Richardson 97] wird für das univariate Schätzproblem eine feste Anordnung der Komponenten vorgeschrieben, d.h. die Reihenfolge der Mittelwerte (alternativ: Mischgewichte, Varianzen) auf der reellen Achse ist fest vorgegeben. Die Konstruktion der Markov-Kette nimmt darauf Rücksicht, indem Übergänge, die zu einer Verletzung der Reihenfolge führen würden, verworfen

werden. Diese Vorgehensweise besitzt allerdings den Nachteil, dass auch die Posterioriverteilung der Parameter durch diese Randbedingung verändert wird. Außerdem ist die Übertragung auf multivariate GMMs schwierig.

Ein Alternativvorschlag stammt aus [Stephens 97, Stephens 00]. Hierbei wird vorgeschlagen, nach einem kompletten Lauf des Verfahrens die gesampleten GMM-Parameter auf Ähnlichkeit zu untersuchen. Dadurch wird es möglich, eine Permutation zu bestimmen, unter der sich zwei Parametervektoren am ähnlichsten sind. Die auf diese Weise gefundene Permutation wird auf einen der beiden Parametervektoren angewendet und eventuell vorliegendes Label switching eliminiert. Die Methode zeigt in den beiden Arbeiten gute Ergebnisse, erfordert jedoch eine aufwändige Nachbearbeitung.

Im Kontext der in Abschnitt 5.4 beschriebenen Variante von Data Augmentation stellt sich Label switching als kein gravierendes Phänomen heraus. Tritt Label switching auf, so werden durch die Mittelung der Parametervektoren über mehrere Iterationen GMMs gebildet, die eine geringe Likelihood auf der Trainingsmenge haben und daher nicht ausgewählt werden. Eine einzelne Vertauschung von Komponenten führt also nicht dazu, dass der Algorithmus am Ende ein schlechtes Ergebnis liefert. Gleichzeitig wird aber durch die Beschränkung auf einen gleitenden Durchschnitt moderater Länge der Gefahr vorgebeugt, durch Mittelung über alle gesampleten Parameter Lösung zu erzeugen, die identische Parameter für alle Komponenten erzeugen.

Ferner ist zu berücksichtigen, dass durch die Verwendung nichtinformativer Prioriverteilungen und die Entfernung überflüssiger Komponenten das Phänomen des Label switching nur selten auftritt, da die Dynamik der Markov-Kette klein wird. Beispielsweise zeigt Abbildung 5.21, dass die Anordnung der Komponenten sich nach einer kurzen Einschwingphase nicht mehr ändert.

5.5.2 Initialisierung

Die Sätze über ergodische Markov-Ketten (siehe Abschnitt 3.2) besagen, dass die stationäre Verteilung unabhängig von der Initialisierung der Markov Kette angenommen wird. Diese Aussage gilt allerdings nicht mehr, wenn Data Augmentation mit unechten Prioriverteilungen verwendet wird. Außerdem sorgt die Entfernung von Komponenten für einen unumkehrbaren Eingriff in die Markov-Kette. Daher können gute Lösungen nur dann erzielt werden, wenn bereits die Initialisierung gut ist.

Zunächst ist unbekannt, wie die Struktur der Daten beschaffen ist. Daher kann am Anfang weder entschieden werden, wie groß das GMM sein muss, noch wie die Parameter initialisiert werden sollten. Lediglich grobe Aussagen sind durch Analyse der Trainingsdaten möglich, z.B. stellt die Anzahl Trainingsdaten eine obere Schranke für die Anzahl Komponenten dar; ferner geben Erwartungswert und Varianz der Trainingsstichprobe Auskunft über die Lage der Verteilung.

Da die in dieser Arbeit vorgestellte Variante von Data Augmentation lediglich Komponenten löschen, aber keine hinzufügen kann, ist es notwendig, das initiale GMM so groß zu wählen, dass die Anzahl benötigter Komponenten auf jeden Fall überschätzt wird. Gleichzeitig wirkt sich ein Zuviel an Modellkomplexität lediglich in einer längeren Rechenzeit aus, um die überzähligen Komponenten zu entfernen, nicht aber in einem schlechteren Lernergebnis. Dieses Verhalten ist grundlegend anders als beim EM-Algorithmus, bei dem zu viele Komponenten ein großes Overfitting-Risiko bedeuten.

Eine geeignete Initialisierung sollte dem Verfahren also eine Vielzahl möglicher Ausprägungen von Komponenten anbieten, aus denen sich Data Augmentation in den ersten Iterationen diejenigen auswählen kann, die für die gegebenen Trainingsdaten benötigt werden. Es hat sich in Testläufen herausgestellt, dass es günstig ist, mehr Komponenten anzubieten als Trainingsdaten vorhanden sind und die Parameter der Komponenten zufällig zu bestimmen. In den in Kapitel 7 vorgestellten Experimenten wurden jeweils doppelt so viele Komponenten verwendet, wie Trainingsdaten vorhanden waren. Die Initialisieung der Komponenten erfolgte zufällig durch Ziehen aus folgenden Verteilungen:

$$\mu_j \sim_{i.i.d.} N(\bar{x}, 4 \cdot \hat{S}) \tag{5.2}$$

$$\Sigma_j = \hat{S} \tag{5.3}$$

$$w_j = \frac{1}{k} \tag{5.4}$$

wobei $\bar{x}$ der Mittelwert der Trainingsstichprobe, $\hat{S}$ die empirische Kovarianzmatrix der Trainingsdaten und k die Anzahl Komponenten bezeichnet. Diese Initialisierung garantiert, dass der Datenraum ausreichend von Komponenten überdeckt wird. Außerdem ist diese Form der Initialisierung invariant gegen Reskalierung der Trainingsdaten.

Jede weitere Form der Vorbehandlung, z.B. die beim EM-Algorithmus übliche Verwendung von *k-means* ([Forgy 65, MacQueen 67], siehe auch [Bishop 95]) zur Clusterung der Trainingsdaten, hat sich als nicht ganz so gut erwiesen, da auf diese Weise bereits eine Vorstrukturierung erfolgt, die mitunter interessante Bereiche des Parameterraums bereits ausschließt. Statt dessen sollte die Auswahl der besten Komponenten dem Verfahren selbst überlassen bleiben. Lediglich bei zeitkritischen Anwendungen, in denen das Entfernen der vielen initialen Komponenten zu aufwändig ist, kann durch die Vorgabe eines kleineren GMMs, dessen Komponenten mit Hilfe des k-means Ansatzes initialisiert wurden, eine Verbesserung gegenüber der rein zufälligen Initialisierung eines kleinen GMM erreicht werden.

5.5.3 Überlappende Komponenten

Abschließend soll untersucht werden, wie sich Data Augmentation mit wenig informativen Prioriverteilungen verhält, wenn ein GMM mehrere ähnliche oder gleichartige Komponenten enthält. Hier spielt die randomisierte Arbeitsweise und die Wechselwirkung zwischen P-step und I-step eine wesentliche Rolle.

In einem Experiment wurde auf 200 normalverteilten univariaten Trainingsdaten ein GMM mit zwei Komponenten trainiert. Die Initialisierung der Komponenten war identisch, so dass im ersten I-step jedes Muster mit gleicher Wahrscheinlichkeit der einen oder anderen Komponente zugeordnet wurde. Nach einer, drei, 10 und 30 Iterationen wurden die Erwartungswerte der beiden Komponenten erfasst und durch wiederholte Ausführung des Experiments eine empirische Verteilung der Komponenten-Erwartungswerte ermittelt. Abbildung 5.25 zeigt die kumulative Verteilungsfunktion dieser Verteilungen.

Erwartungsgemäß sind die Verteilungen symmetrisch um Null und sie werden umso breiter, je mehr Iterationen durchgeführt werden. Die Standardabweichungen der empirischen Verteilungen betrugen 0.1 nach einer Iteration, 0.2 nach drei Iterationen, 0.3 nach 10 Iterationen und 0.4 nach 30 Iterationen. Obwohl die Initialisierung der beiden Komponenten

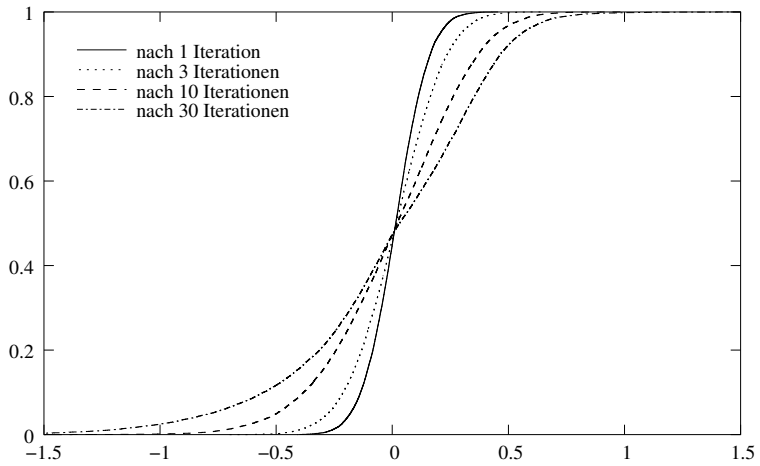

Abbildung 5.25: Entwicklung der empirischen kumulativen Verteilungsfunktionen der Erwartungswerte eines GMMs mit zwei Komponenten. Die Komponenten wurden mit identischen Parametern initialisiert. Als Trainingsdaten dienten 200 standardnormalverteilte Muster.

identisch war, streuen die Parameter der beiden Komponenten bereits nach wenigen Iterationen deutlich. Diese Streuung wird umso stärker, je weniger Trainingsdaten zur Verfügung stehen.

Alleine durch die randomisierte Vorgehensweise spezialisieren sich die Komponenten also bereits nach wenigen Iterationen auf verschiedene Bereiche des Datenraums. Eine ähnliche Beobachtung kann man auch bezüglich der Mischgewichte machen: durch den randomisierten I-step werden einer Komponente mehr Muster zugeordnet als der anderen. Dadurch wird im darauffolgenden P-step für die eine Komponente ein größeres Mischgewicht gesamplet als für die andere, so dass auch im darauf folgenden I-step die Zuordnung der Muster zu den Komponenten ungleichmäßig erfolgt.

Im Gegensatz dazu würde der streng deterministische EM-Algorithmus in dieser Situation beide Komponenten stets in der gleichen Weise verändern, so dass die Komponenten bei gleicher Initialisierung nie unterschiedliche Ausprägungen annehmen würden. Data Augmentation sorgt daher in weit größerem Maße als der EM-Algorithmus dafür, dass Komponenten möglichst unterschiedliche Aspekte der Daten abdecken, ihre Dichten sich also möglichst wenig überlappen. Ist dies der Fall, so werden bestimmte Trainingsmuster bestimmten Komponenten bevorzugt zugeordnet, so dass sich die Sampleverteilungen in aufeinanderfolgenden P-steps nicht sehr stark unterscheiden. Ist die Anzahl zugeordneter Muster außerdem noch groß genug, so konzentriert sich die Suchverteilung im P-step auf einen kleinen Bereich des Parameterraums, so dass wiederum ähnliche Parameter erzeugt werden (vergleiche die Diskussion zu Abbildung 5.14).

Aus dem Zusammenspiel zwischen I-step und P-step lassen sich daher Bedingungen ableiten, wann die Dynamik der Markov-Kette gering ist. Dies ist der Fall, falls

(i) jeder Komponenten viele Muster zugeordnet werden

(ii) die Dichte der Komponenten sich wenig überlappt, so dass jeder Komponente immer

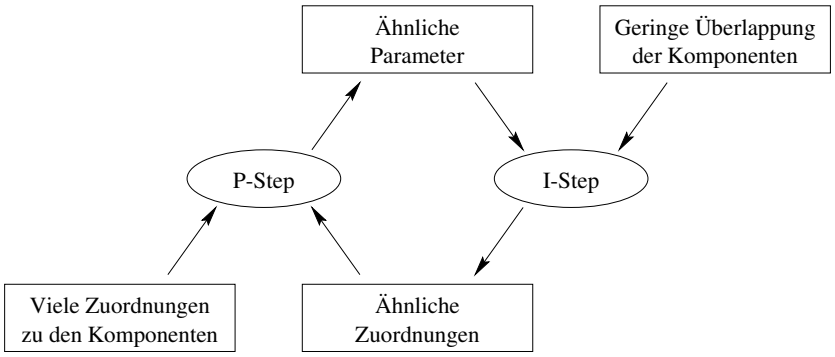

Abbildung 5.26: Bedingungen, die zu einer geringen Dynamik der Markov-Kette führen

wieder die selben oder ähnliche Muster zugeordnet werden.

Dieser Zusammenhang ist in Abbildung 5.26 dargestellt. Die Tatsache, dass Data Augmentation GMMs mit geringer Überlappung der Komponenten bevorzugt, ist im Sinne einer möglichst kompakten Beschreibung der Verteilung eine erwünschte Eigenschaft: zwei ähnliche Komponenten beschreiben die selben Daten und können daher durch eine einzelne Komponente ersetzt werden. Nur Komponenten, die einen eigenen Aspekt der Daten modellieren, bleiben erhalten. Redundanz wird somit vermieden.

Kapitel 6

Kombinationsverfahren

6.1 Deterministische und randomisierte Lernverfahren

In den vorangegangenen Kapiteln wurden deterministische und randomisierte Lernverfahren streng voneinander getrennt betrachtet: der EM-Algorithmus als – bis auf die Initialisierung – deterministische Vorgehensweise, um ein Maximum der Likelihoodfunktion zu finden, sowie Data Augmentation als randomisierter Algorithmus, um aus der Daten-Posterioriverteilung zu samplen.

Das Ziel des Lernens von GMMs ist es jedoch, wie in Kapitel 2 beschrieben, eine möglichst gute Beschreibung für den datengenerierenden Prozess zu finden, aus dem die Trainingsdaten stammen. Es werden also GMM-Parameter gesucht, die die Trainingsdaten gut beschreiben, andererseits aber auch eine gute Generalisierung ermöglichen.

Wie in Abschnitt 5.3 geschildert, stellt das Maximum-Likelihood-Prinzip für GMMs nur eine Annäherung an dieses Ziel dar, da aufgrund der unbeschränkten Likelihoodfunktion Overfitting auftritt und damit die Forderung nach guter Generalisierung nicht gewährleistet werden kann. Insofern ist auch der EM-Algorithmus, der auf dem Maximum-Likelihood-Prinzip basiert, kein optimales Lernverfahren.

Ebenso ist auch die eigentliche Zielsetzung von Data Augmentation eine andere als die geschilderte Lernaufgabe, da nicht die Daten-Posterioriwahscheinlichkeiten, sondern optimale GMM-Parameter gesucht sind. In Abschnitt 5.4 wurde daher eine Abwandlung des Originalverfahrens entwickelt, die stärker auf die eigentliche Lernaufgabe hin ausgerichtet ist. Hierbei wird Data Augmentation als eine zielgerichtete randomisierte Suche interpretiert, die in geschickter Weise Parametersätze erzeugt. Die Posterioriverteilungen werden lediglich als geeignete Suchverteilungen verstanden, um gute Parameter zu finden. Sie werden dagegen an keiner Stelle im Sinne des Bayesschen Sampling verwendet, d.h. es soll kein analytisch nicht berechenbarer Integralausdruck durch eine randomisierte Approximation ersetzt werden wie dies z.B. beim Importance Sampling (siehe z.B. [Andrieu 03]) der Fall ist.

Eine vielversprechende Idee ist es daher, deterministische und randomisierte Verfahren zu kombinieren, um von den Vorteilen beider Ansätze zu profitieren und damit dem Erreichen der eigentlichen Lernaufgabe näher zu kommen. Es geht also darum, Data Augmentation stärker auf die wichtigen Bereiche des Parameterraums zu konzentrieren. Ansatzpunkte hierzu sind die Veränderung der Samplingverteilung im P-step sowie der Übergang von eindeutigen Klassenzugehörigkeiten im I-step zu graduellen Zugehörigkeitsgraden. In einer umgekehrten Sichtweise, vom EM-Algorithmus aus gesehen, besteht die Aufgabe darin, E-

		Parameterbestimmung	
		deterministisch	randomisiert
Zuordnungen	deterministisch	EM	REM
	randomisiert	SEM SAEM MCEM	Data Augmentation

Tabelle 6.1: Übersicht über die Kombinationsverfahren

und M-step zu randomisieren.

Betrachten wir hierzu die Arbeitsweise des EM-Algorithmus' sowie von Data Augmentation. Beide Verfahren arbeiten iterativ, wobei in jeder Iteration zwei Einzelschritte durchgeführt werden. Am Ende einer Iteration steht jeweils ein vorläufiger GMM-Parametersatz. Als Zwischenziel wird ebenfalls bei beiden Verfahren eine Art Zuordnung der Daten zu den Komponenten vorgenommen.

Eine in der Literatur vorgeschlagene Kombination ist die Verbindung von randomisierter Bestimmung der Klassenzugehörigkeiten und deterministischen Parameterbestimmung. Eine Reihe von Ansätzen basiert auf dieser Idee, insbesondere *stochastic EM* (SEM) [Broniatowski 83], *stochastic annealing EM* (SAEM) [Celeux 92] sowie *Monte-Carlo EM* (MCEM) [Wei 90].

Neben einer kurzen Diskussion dieser Verfahren soll hier in erster Linie eine neuartige Möglichkeit zur Kombination von randomisierten und deterministischen Elementen entwickelt werden – genannt randomisiert EM (REM) – die vom Verhalten her dem Data Augmentation Verfahren stärker ähnelt als dem EM-Algorithmus. Die Idee besteht hierbei in der Kombination von deterministisch berechneten graduellen Klassenzugehörigkeiten und randomisierter Parameterbestimmung.

Einen Überblick über den Zusammenhang der verschiedenen Verfahren zeigt Tabelle 6.1.

6.2 SEM, SAEM und MCEM

Ausgehend vom EM-Algorithmus setzen diese drei Verfahren in unterschiedlicher Weise an der Randomisierung des E-steps an. Eine detailliertere Darstellung der Verfahren findet sich in [Celeux 95].

Der Grundgedanke des SEM Verfahrens besteht in der Kombination des I-steps mit dem M-step. Es wird also nach Berechnung der Zugehörigkeitswahrscheinlichkeiten h_{ij} jeder Datenpunkt x_i einer Komponente z_i zufällig zugeordnet. Dabei wird aus einer Multinomialverteilung mit den Wahrscheinlichkeiten $h_{i1}, h_{i2}, \ldots, h_{ik}$ gezogen, genauso wie im I-step von Data Augmentation. Mit den zufällig gezogenen Klassenzugehörigkeiten zerfällt das GMM in ein Klassenmodell, so dass im nachfolgenden M-step lediglich die Maximum-Likelihood-Schätzer der einzelnen Komponenten bzw. der Maximum-Likelihood-Schätzer der Mischgewichte berechnet werden müssen.

Durch die Zufallsauswahl im E-step ergibt sich ein randomisiertes Verhalten des Algorithmus'. Der Grad der Zufälligkeit hängt dabei von der Anzahl Datenpunkte ab: bei wenigen Daten variieren die erzeugten Parameter sehr stark, bei einer großen Anzahl dagegen wirkt sich der Zufallseinfluss in geringerem Maße aus. Durch diese Randomisierung soll das Problem lokaler Maxima der Likelihoodfunktion überwunden werden und daher eine vorzeitige Konvergenz des Verfahrens in einem suboptimalen Bereich vermieden werden.

Nachteilig wirkt sich die Randomisierung allerdings im Bezug auf Generalisierung und die Vermeidung von degenerierten Komponenten aus: vor allem bei zu großen GMMs oder bei wenigen Datenpunkten ist die Wahrscheinlichkeit, dass einer Komponente nur ein einzelner Datenpunkt zugeordnet wird, sehr groß. Dadurch degeneriert die entsprechende Komponente im darauffolgenden M-step zwangsläufig.

Eine Konvergenz des Verfahrens im Sinne des Findens eines besten Schätzers ist ebenfalls nur dann zu erwarten, wenn auf viele Datenpunkte zurückgegriffen werden kann, so dass der Zufallseinfluss gering bleibt. Diese Schwierigkeit vermeidet SAEM, das SEM um eine Annealing schedule erweitert, ähnlich dem *Simulated Annealing* [Kirkpatrick 83].

In jeder Iteration wird hierzu der EM- sowie der SEM-Ansatz verwendet um zwei neue Parameter $\vartheta_{EM}^{(t+1)}$ sowie $\vartheta_{SEM}^{(t+1)}$ zu erhalten. Der SAEM-Algorithmus berechnet daraus ein gewichtetes Mittel $\vartheta_{SAEM}^{(t+1)} = (1-\zeta_{t+1})\vartheta_{EM}^{(t+1)} + \zeta_{t+1}\vartheta_{SEM}^{(t+1)}$, wobei der Annealing Parameter $\zeta_t \in [0,1]$ langsam von 1 auf 0 abgesenkt wird. Am Anfang verhält sich das Verfahren daher wie der SEM-Algorithmus und sucht den Parameterraum großräumig ab, während mit kleiner werdenden ζ_t das Verhalten immer mehr dem EM-Algorithmus gleicht, d. h. der Zufallseinfluss wird geringer und damit die Konvergenz ermöglicht.

Eine ähnliche Idee verfolgt der MCEM-Ansatz. Dieser basiert allerdings auf dem *Multiple Imputation*-Prinzip [Rubin 87]. Hierbei wird die Zufallsauswahl des SEM-Ansatzes mehrfach hintereinander durchgeführt und somit mehrere Sätze an Zuordnungen $z_{i,1}, z_{i,2}, \ldots, z_{i,m}$ ermittelt. Anschließend werden durch Mittelung wieder graduelle Zuordnungen h'_{ij} wiefolgt berechnet:

$$h'_{ij} = \frac{\#\{l \in \{1,\ldots,m\} | z_{i,l} = j\}}{m} \tag{6.1}$$

Die h'_{ij} stellen also eine erwartungstreue stochastische Approximation der h_{ij} dar. Durch Variation der Anzahl Wiederholungen m kann die Streuung der h'_{ij} gesteuert werden. Für $m = 1$ verhält sich MCEM wie SEM, für $m \to \infty$ wie der EM-Algorithmus. Daher wird m als Annealing Parameter im Laufe der Zeit zunehmend vergrößert. MCEM verhält sich also sehr ähnlich wie der SAEM-Ansatz, allerdings ist der Rechenaufwand, vor allem bei großen Werten von m, sehr groß.

Die Verwendung von SEM, SAEM oder MCEM dient in erster Linie der Vermeidung lokaler Optima der Likelihoodfunktion. Eine Verbesserung der Generaliserungsleistung kann aufgrund der Konstruktion der Algorithmen nicht erwartet werden, im Wesentlichen gleicht sie der des EM-Algorithmus'.

Ein interessanter Aspekt von SAEM und MCEM ist die Möglichkeit, den Zufallseinfluss mit Hilfe des Annealing Parameters gezielt zu kontrollieren. Statt, wie bei Data Augmentation, die Samplingverteilung so zu wählen, dass die entstehende Markov-Kette eine bestimmte stationäre Verteilung besitzt, wird die Samplingverteilung heuristisch so eingestellt, dass das Verfahren möglichst gute Parameter erzeugt, oder sie wird mit einer Annealing schedule versehen, um die Stärke des zufälligen Einflusses im Laufe des Trainings zu verändern.

6.3 Randomisiert EM

6.3.1 Herleitung

In Abschnitt 5.3.2 wurden die günstigen Eigenschaften von Data Augmentation im Bezug auf die Generalisierungsleistung beschrieben. Diese beruhen in erster Linie auf der zufälligen Parameterwahl im P-step. Daher soll in diesem Abschnitt untersucht werden, in welcher Weise es möglich ist, den P-step mit dem E-step des EM-Algorithmus' zu kombinieren und ob eine weitere Verbesserung der Ergebnisse durch eine Veränderung der Samplingverteilung im P-step möglich ist. Das resultierende Verfahren soll als *randomisiert EM* (REM) bezeichnet werden.

Der P-step, wie in Abschnitt 4.2 definiert, geht von klassifizierten Daten aus, d.h. jedes Trainingsmuster x_i ist genau einer Komponente z_i zugeordnet. Der E-step dagegen erzeugt keine eindeutigen Zuordnungen, sondern graduelle Zuordnungswahrscheinlichkeiten h_{ij}. Daher erfordert die Kombination von E- und P-step eine Anpassung des P-steps an graduelle Zugehörigkeiten. Diese Modifikation soll als *randomisierter M-step*[1] bezeichnet werden.

Als Randbedingungen zur Konstruktion des randomisierten M-steps sollen folgende Bedingungen gelten:

- bei eindeutigen Zurdnungen, d.h. $h_{ij} \in \{0, 1\}$ soll sich der randomisierte M-step wie der P-step verhalten
- je größer die Zugehörigkeit eines Musters x_i zu einer Komponente j wird, d.h. je größer h_{ij}, desto größer soll der Einfluss des Musters x_i auf die Posterioriverteilung der j-ten Komponente sein (Monotonie)

Wie aus den Gleichungen (4.17) und (4.22)–(4.27) hervorgeht, gehen die Trainingsmuster in die Parameter der Posterioriverteilungen additiv ein. Eine mögliche Verallgemeinerung dieses Ansatzes besteht daher darin, die jeweiligen Trainingsmuster anteilig gemäß ihren h_{ij}-Werten eingehen zu lassen. Dabei wird ausgenutzt, dass gilt: $\sum_{j=1}^{k} h_{ij} = 1$. Es ergeben sich damit die folgenden Samplingverteilungen:

$$\begin{aligned}
(w_1, \ldots, w_k) | \left(h_{ij}\right)_{i=1,\ldots,n;\, j=1,\ldots,k} &\sim D(\gamma_1 + n_1, \ldots, \gamma_k + n_k) \\
\Sigma_j | x_1, \ldots, x_n, \left(h_{ij}\right)_{i=1,\ldots,n;\, j=1,\ldots,k} &\sim W^{-1}(\hat{q}_j, \hat{\Lambda}_j) \\
\mu_j | \Sigma_j, x_1, \ldots, x_n, \left(h_{ij}\right)_{i=1,\ldots,n;\, j=1,\ldots,k} &\sim N(\hat{m}_j, \frac{1}{\hat{\eta}_j}\Sigma_j)
\end{aligned} \tag{6.2}$$

mit:

$$\hat{\eta}_j = \eta + n_j \tag{6.3}$$

$$\hat{q}_j = q + n_j \tag{6.4}$$

$$\hat{m}_j = \frac{n_j}{n_j + \eta}\bar{x}_j + \frac{\eta}{n_j + \eta}m \tag{6.5}$$

$$\hat{\Lambda}_j = \left(\Lambda^{-1} + n_j\hat{S}_j + \frac{n_j\eta}{n_j + \eta}(\bar{x}_j - m)(\bar{x}_j - m)^T\right)^{-1} \tag{6.6}$$

[1] Die Bezeichnung *randomisierter M-step* erfolgt lediglich in Anlehnung an den EM-Algorithmus. Tatsächlich erfolgt keine Maximierung der Likelihood sondern eine zufällige Erzeugung von Parametern.

$$n_j = \sum_{i=1}^{n} h_{ij} \tag{6.7}$$

$$\bar{x}_j = \frac{1}{n_j} \sum_{i=1}^{n} h_{ij} x_i \tag{6.8}$$

$$\hat{S}_j = \frac{1}{n_j} \sum_{i=1}^{n} h_{ij} (x_i - \bar{x}_j)(x_i - \bar{x}_j)^T \tag{6.9}$$

Offensichtlich ergeben sich bei diskreten Zugehörigkeitswahrscheinlichkeiten $h_{ij} \in \{0, 1\}$ die ursprünglichen Posterioriverteilungen des P-step, d.h. die erste Konstruktionsbedingung ist erfüllt. Ferner gehen die Trainingsmuster gewichtet mit ihrem jeweiligen Zugehörigkeitsgrad ein, so dass auch die Monotoniebedingung erfüllt ist.

Berechnet man die Erwartungswerte der Samplingverteilungen mit:

$$\gamma = 0, \quad q = d + 1, \quad \Lambda^{-1} \to 0, \quad \eta = 0, \quad m \text{ beliebig} \tag{6.10}$$

erhält man die Berechnungsvorschrift des klassischen M-steps:

$$E[w_j] = \frac{n_j}{n} \tag{6.11}$$

$$E[\Sigma_j] = \hat{S}_j \tag{6.12}$$

$$E[\mu_j] = \bar{x}_j \tag{6.13}$$

Der REM-Algorithmus besteht somit aus der alternierenden Ausführung des E-steps und des randomisierten M-steps. Zusätzlich kann, wie beim Data Augmentation Algorithmus beschrieben, ein Lösch-Schritt zum Entfernen überzähliger Komponenten hinzukommen. Analog zu Abbildung 5.18 ist das algorithmische Schema von REM in Abbildung 6.1 wiedergegeben.

6.3.2 Verhalten

Das Verhalten von REM wird durch das Zusammenwirken von E-step und randomisiertem M-step bestimmt. An dieser Stelle soll nur der Fall mit nichtinformativen Prioriverteilungen gemäß Abschnitt 5.4.2 betrachtet werden.

Im Vergleich des REM-Ansatzes mit Data Augmentation stellt die Zuordnung der Muster zu den Komponenten den einzigen Unterschied dar. Im Fall nahezu eindeutiger Zuordnungen, d.h. für $h_{ij} \approx 1$ für eine bestimmte Komponente j verhalten sich beide Ansätze gleichartig, da der I-step von Data Augmentation nahezu deterministisch wird.

Im Fall unklarer Zuordnungen dagegen unterscheiden sich beide Ansätze voneinander. Zur Untersuchung dieses Sachverhaltes wurde der Extremfall betrachtet, in dem für alle Datenpunkte die Wahrscheinlichkeiten, zu einer von zwei Komponenten zu gehören, identisch sind. Hierzu wurden auf einem Datensatz von 200 normalverteilten univariaten Daten sowohl das REM-Verfahren als auch Data Augmentation einige Iterationen trainiert. Die initialen GMMs bestanden aus zwei identischen Komponenten, so dass im ersten I-step/E-step für alle Datenpunkte x_i und alle Komponenten j galt: $h_{ij} = 0.5$.

Nach je einer und je 10 Iterationen der Trainingsverfahren wurden die Erwartungswerte der Komponenten erfasst. Abbildung 6.2 zeigt die resultierende empirische Verteilungsfunktion der Erwartungswerte nach einem und nach 10 Schritten der verglichenen Lernverfahren. Deutlich ist zu erkennen, dass die Streuung der Erwartungswerte im Fall von REM

```
 1.  wähle initiale k, (w_1, ..., w_k, μ_1, ..., μ_k, Σ_1, ..., Σ_k)
 2.  wiederhole für t = 0, 1, 2, ...
 3.     wiederhole, bis keine Komponenten mehr gelöscht werden
           EXPECTATION-STEP (E-STEP):
 4.        wiederhole für jedes Trainingsmuster x_i
 5.           wiederhole für jede Komponente j
 6.              berechne h_ij gemäß (4.40)
           DELETION-STEP (D-STEP):
 7.        falls Komponenten mit zu kleinem n_j existieren
 8.           lösche eine der Komponenten mit zu wenigen Zuordnungen
 9.           verkleinere den Parametervektor entsprechend
10.           vergrößere die Mischgewichte der verbleibenden Komponenten
11.           verringere k um 1
        RANDOMISIERTER M-STEP:
12.     sample neue Mischgewichte (w_1, ..., w_k) gemäß (6.2)
13.     wiederhole für jede Komponente j
14.        sample neue Kovarianzmatrix Σ_j gemäß (6.2)
15.        sample neuen Erwartungswert μ_j gemäß (6.2)
```

Abbildung 6.1: REM, erweitert um das Löschen von Komponenten (D-step). Expectation- und Deletion-step werden so lange wiederholt, bis allen Komponenten ausreichend große n_j besitzen. Erst anschließend werden neue Parameter gesamplet.

kleiner ist als im Fall von Data Augmentation. Dies spiegelt den geringeren Zufallseinfluss durch die Verwendung des deterministischen E-steps an Stelle des zufälligen I-steps wieder. Eine interessante Beobachtung ist ferner, dass das SEM-Verfahren eine noch geringere Streuung erzeugt, d.h. die Randomisierung der Zuordnungen im E-/I-step wirkt sich weniger stark aus als die Randomisierung des M-/P-steps.

REM erzeugt eine ergodische Markov-Kette. Wie zuvor argumentiert, ist deren Dynamik aber geringer als bei Data Augmentation, d.h. REM hält sich längere Zeit in dem selben Bereich des Parameterraums auf als Data Augmentation. Es springt nicht so sehr. Dadurch werden die interessanten Bereiche des Parameterraums längere Zeit am Stück besucht und daher die Berechnung guter Parametersätze durch die Mittelung über mehrere Iterationen begünstigt. Die experimentellen Ergebnisse in Kapitel 7 untermauern diese Aussage.

Ebenso wirkt sich die geringere Dynamik auf die Entwicklung der GMM-Größe aus, wenn GMMs mit zu wenigen Zuordnungen (bzw. zu kleinem n_j) entfernt werden. So konnte in oben geschildertem Experiment mit 200 Datenpunkten und identisch initialisierten Komponenten beobachtet werden, dass bei Verwendung von Data Augmentation in 380 von 10000 Läufen innerhalb der ersten 30 Iterationen eine Komponente entfiel, während dieser Fall bei REM nur in 12 von 10000 Läufen auftrat. Dies bedeutet jedoch nicht, dass überzählige Komponenten bei REM nicht entfernt würden, sondern nur, dass es etwas länger dauert als bei Data Augmentation. Insgesamt kann nicht beobachtet werden dass REM größere GMMs bevorzugt als Data Augmentation (vergleiche Abschnitt 7.3).

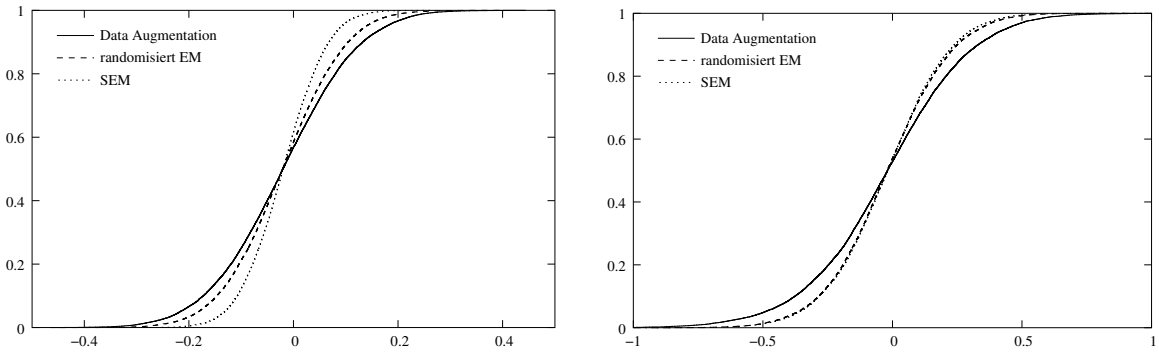

Abbildung 6.2: Vergleich der empirischen kumulativen Verteilungsfunktionen der Erwartungswerte eines GMMs mit zwei Komponenten nach einer Iteration (links) Data Augmentation, REM bzw. SEM und nach 10 Iterationen (rechts). Die Komponenten wurden mit identischen Parametern initialisiert. Als Trainingsdaten dienten 200 standardnormalverteilte Muster.

6.3.3 Veränderung der Samplingverteilung

Bereits durch den Übergang von Data Augmentation zu REM wurde die exakte Bayessche Modellierung verlassen. Die stationäre Verteilung von REM ist wegen des veränderten Übergangskerns nicht mehr die Daten-Posteriori-Verteilung, sondern sie weicht von dieser ab. Es stellt sich daher die Frage, ob durch eine gezielte Veränderung der Samplingverteilung im randomisierten M-step weitere Verbesserungen erzielt werden können.

Die Konstruktion des randomisierten M-steps in Abschnitt 6.3.1 stellt eine Verallgemeinerung des P-steps für graduelle Klassenzugehörigkeiten dar. Soll der Zufallseinfluss weiter verringert werden, so kann durch Veränderung der Suchverteilung eine stärkere Konzentration auf einen kleinen Bereich um den Maximum-Likelihood-Schätzer erreicht werden. Hierzu soll der randomisierte M-step um einen zusätzlichen Parameter f erweitert werden, der den Grad der Konzentration der Suchverteilung um den Maximum-Likelihood-Schätzer bestimmt.

Der bisherige randomisierte M-step soll prinzipiell beibehalten werden, allerdings wird die Konstruktion so abgeändert, dass die Anzahl Datenpunkte scheinbar um den Faktor f erhöht wird, so dass sich die Posterioriverteilungen stärker um einen bestimmten Wert herum konzentrieren.

Die Suchverteilungen im randomisierten M-step können somit aus (6.2)–(6.9) gewonnen werden durch Ersetzen der Größe n_j in (6.2)–(6.6) durch n'_j:

$$n'_j = f \cdot \sum_{i=1}^{n} h_{ij} \tag{6.14}$$

Es ist zu beachten, dass die Formeln für den Daten-Mittelwert (6.8) sowie die empirische Varianz (6.9) unverändert bleiben.

Mit den Parametern aus (6.10) erhält man für die Erwartungswerte der Suchverteilungen wiederum die Berechnungsvorschriften des deterministischen M-Steps (6.11)–(6.13). Die Streuung der Suchverteilungen wird umso kleiner, je größer f gewählt wird, für $f = 1$ ergibt

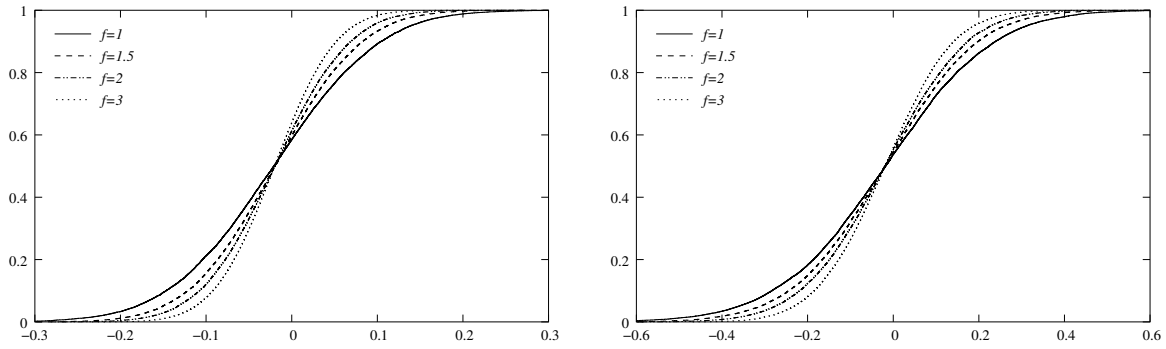

Abbildung 6.3: Vergleich der empirischen kumulativen Verteilungsfunktionen der Erwartungswerte eines GMMs mit zwei Komponenten nach einer Iteration (links) REM und nach 10 Iteration (rechts) für verschiedene Einstellungen von f. Die Komponenten wurden mit identischen Parametern initialisiert. Als Trainingsdaten dienten 200 standardnormalverteilte Muster.

sich die ursprüngliche Formulierung gemäß (6.2)–(6.9). Die Abhängigkeit der Streuung von f kann exemplarisch Abbildung 6.3 entnommen werden.

Mit dem Parameter f kann man somit steuern, ob der Algorithmus eher in einer kleinen lokalen Umgebung um den Maximum-Likelihood-Schätzer suchen soll (großes f) oder ob die Suche über einen größeren Bereich gehen soll (kleines f). Experimentelle Ergebnisse bezüglich einer günstigen Einstellung von f werden in Abschnitt 7.3 vorgestellt.

6.4 Komitees

6.4.1 Prinzip der Komiteebildung

Neben den bisher diskutierten Ansätzen, das Lernverfahren selbst zu optimieren, stellt Komiteebildung eine weitere Möglichkeit zur Verbesserung der Lernergebnisse dar. Hierbei werden in einem dem eigentlichen Lernen nachgeordneten Schritt mehrere gelernte Modelle zu einem größeren Modell kombiniert. Dieses Modell selbst wird nicht mehr weiter eingelernt.

Die Vorteile der Komiteebildung sind vor allem aus dem Bereich der nichtlinearen Regression bekannt [Ragg 00]. Bei derlei Aufgabenstellungen soll ein funktionaler Zusammenhang zwischen einer (mehrdimensionalen) Eingangsgrößen e und einer Zielgröße t gelernt werden, d.h. eine Funktion $f : \mathbb{R}^d \to \mathbb{R}$, für die der Fehler auf vorgegebenen Trainingsdaten $(e_1, t_1), \ldots, (e_n, t_n)$ möglichst gering wird im Sinne von:

$$\textit{minimiere} \sum_{i=1}^{n} (f(e_i) - t_i)^2 \tag{6.15}$$

Wurden mehrere Regressionsmodelle trainiert, z.B. mehrere verschiedene mehrschichtige neuronale Netze oder RBF-Netze $f_1, \ldots, f_\nu$, so kann daraus ein Komitee gebildet werden:

$$f_{Komitee}(e) := \frac{1}{\nu} \sum_{i=1}^{n} f_i(e) \tag{6.16}$$

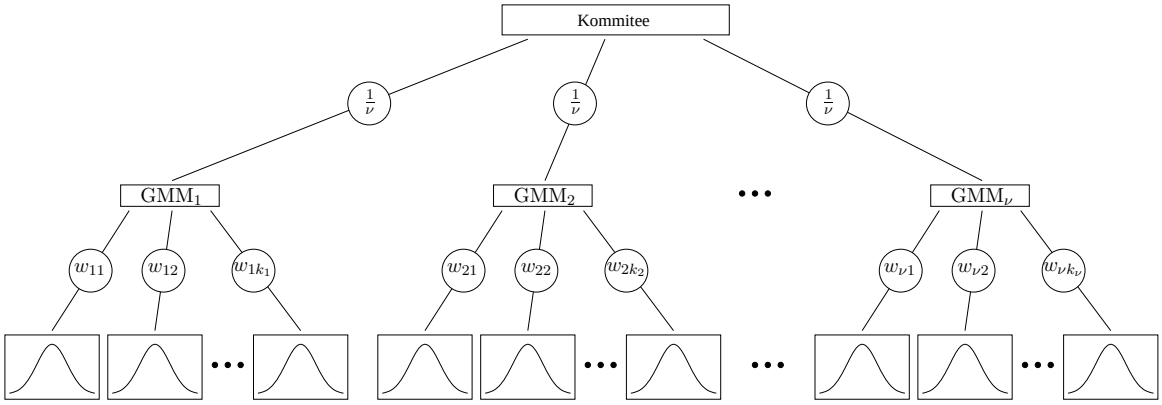

Abbildung 6.4: Komitee aus ν GMMs. w_{ij} bezeichne das Mischgewicht der j-ten Komponente des i-ten GMM. k_i bezeichne die Größe des i-ten Komiteemitglieds.

Das Komitee stellt wiederum ein Modell der Daten dar. Es entsteht durch Mittelung über die trainierten Modelle. Es kann gezeigt werden, dass

(a) der Erwartungswert des Prognosefehlers von $f_{Komitee}$ kleiner oder gleich dem mittleren Fehler der einzelnen Modelle f_i ist

(b) bei paarweise unkorrelierten Fehlern der f_i der Erwartungswert des Prognosefehlers von $f_{Komitee}$ um den Faktor $\frac{1}{\nu}$ kleiner ist als der mittlere Fehler der einzelnen Modelle f_i

Herleitung, Verfeinerung und Beweis dieser beiden Aussagen siehe [Ragg 00]. Gemäß Aussage (a) riskiert man durch Komiteebildung also zumindest im Erwartungswert keine Verschlechterung der Prognose. Andererseits kann man unter bestimmten Bedingungen sogar erwarten, dass sich der Prognosefehler bei Komiteebildung verringert. Dazu ist es notwenig, dass jeder Teilnehmer f_i des Komitees einen geringen Prognosefehler aufweist, gleichzeitig aber auch die Prognosefehler der Komiteeteilnehmer möglichst stochastisch unabhängig voneinander sind.

6.4.2 Komitees von GMMs

Im unüberwachten Lernen von GMMs ist das Ziel, die unbekannte Dichte der Trainingsdaten zu bestimmen. Im Gegensatz zur Aufgabenstellung bei Regression kann aber nicht der quadratische Fehler zwischen tatsächlicher Dichte und dem Modell minimiert werden, da die tatsächliche Dichte nicht bekannt ist, auch nicht punktweise an den Stellen der Trainingsdaten. Statt dessen wird die Likelihood der Trainingsdaten als Gütemaß verwendet. Daher ist eine direkte Übertragung der theoretischen Aussagen über die Eigenschaften der Komiteebildung nicht möglich. Dennoch kann die Grundidee auch bei der Dichteschätzung eingesetzt werden [Ormoneit 95].

Ein Komitee von GMMs ist, analog dem Regressionsfall, das arithmetische Mittel aus den Komiteemitgliedern. Bei ν Komiteemitgliedern, gegeben durch ihre Dichte p_i $(i = 1, \ldots, \nu)$, ergibt sich die Dichte des Komitees zu:

$$p_{Komitee}(x) = \frac{1}{\nu} \sum_{i=1}^{\nu} p_i(x) \tag{6.17}$$

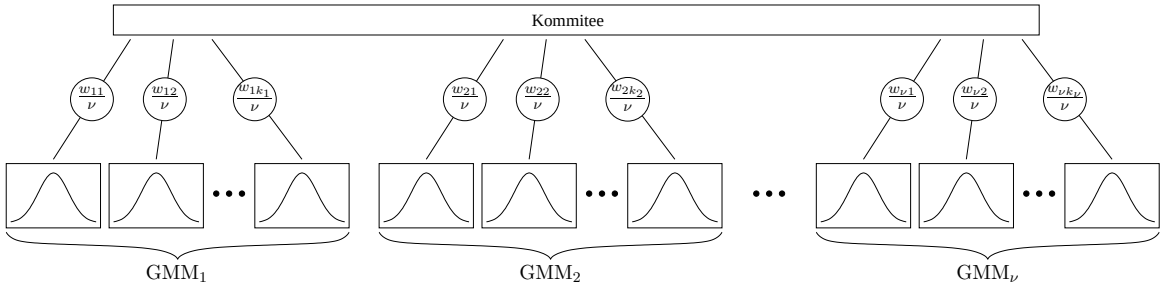

Abbildung 6.5: Komitee aus ν GMMs in flacher Form. w_{ij} bezeichne das Mischgewicht der j-ten Komponente des i-ten GMM. k_i bezeichne die Größe des i-ten Komiteemitglieds.

Ein Komitee von GMMs ist also selbst eine Mischverteilung, deren Komponenten nun aber selbst GMMs sind. Abbildung 6.4 verdeutlicht diesen Aufbau. Kombiniert man den Faktor $\frac{1}{\nu}$ mit den Mischgewichten der einzelnen GMMs kann man – äquivalent – das Komitee von GMMs als ein großes GMM darstellen, wie in Abbildung 6.5 dargestellt.

Die Güte des Komitees hängt auch für GMMs von zwei Faktoren ab: (a) der Güte der Komiteemitglieder sowie (b) der Unabhängigkeit der Komiteemitglieder. Vor allem im Bezug auf den zweiten Aspekt sollen die beiden Lernverfahren *EM-Algorithmus* und *Data Augmentation/REM* verglichen werden.

Der EM-Algorithmus ist ein deterministischer Ansatz, d.h. bei identischen Trainingsdaten und identischer Initialisierung wird immer das selbe GMM gelernt. Zudem maximiert der EM-Algorithmus die Likelihoodfunktion, d.h. es findet vornehmlich lokale Optima der Likelihoodfunktion. Um die Erzeugung verschiedenartiger Komiteeteilnehmer zu ermöglichen, verbleiben somit nur noch drei Möglichkeiten: die Variation der Trainingsmenge, eine andere Initialisierung sowie eine Variation über die Größe der GMMs.

Bei der Variation der Trainingsmenge wird jeder Komiteeteilnehmer auf einer Teilmenge der Trainingsdaten trainiert. In [Ormoneit 95] werden jeweils Teilmengen durch Ziehen ohne Zurücklegen gebildet, die im Umfang 70% der gesamten Trainingsmenge umfassen. Durch diese Vorgehensweise wird jedes Komiteemitglied auf einer etwas veränderten Datenbasis trainiert, so dass sich zwangsläufig unterschiedliche Ausprägungen ergeben. Der Nachteil dieser Vorgehensweise ist allerdings, dass jedes Komiteemitglied suboptimal bleiben muss, da eine Verkleinerung der Trainingmenge stets mit einer Verschlechterung des gelernten Modells verbunden ist. Der Komitee-Effekt wird dadurch zumindest teilweise kompensiert.

Unterschiedliche initiale GMMs sollen dazu dienen, unterschiedliche lokale Optima der Likelihoodfunktion zu finden. Allerdings garantieren unterschiedliche Initialisierungen dieses Ziel nicht: unterschiedliche Initialisierungen können zur Konvergenz im selben lokalen Optimum führen. Ferner ist zu berücksichtigen, dass zur Berechnung möglichst guter GMMs mit dem EM-Algorithmus bereits gute Initialisierungen erforderlich sind. Andernfalls konvergiert das Verfahren nur sehr langsam und die gefundenen Optima sind eher schlecht. Daher werden initiale GMMs oft vorverarbeitet, z.B. mit dem *k-means*-Clustering, das i.d.R. gute Initialisierungen ermöglicht. Eine derartige Vorgehensweise verringert andererseits die Diversität der initialen GMMs. Es ist also schwierig, nur durch verschiedenartige Initialisierung sowohl Unterschiedlichkeit der gelernten GMMs als auch Optimalität zu erreichen.

Durch die Variation über die GMM-Größe ergibt sich eine weitere Möglichkeit, ver-

schiedenartige Komiteemitglieder zu erhalten. Da die Bestimmung der richtigen GMM-Größe mit dem EM-Algorithmus jedoch an sich bereits schwierig und aufwändig ist, erfordert eine solche Variation zusätzlichen Aufwand. Zu berücksichtigen ist ferner, dass der EM-Algorithmus bei zu großen GMMs zum Overfitting neigt und degenerierte Komponenten erzeugt, so dass die Modellgüte des einzelnen Komiteemitglieds sinkt.

Komiteebildung auf der Grundlage des EM-Algorithmus ist also möglich, aber durch die nur mit Einschränkungen erreichbare Unabhängigkeit der Komiteemitglieder bleibt der Komitee-Effekt begrenzt. Der deterministische Ansatz behindert die Entstehung von großer Diversität und erzeugt ähnliche Komiteemitglieder.

Im Gegensatz hierzu eignen sich Data Augmentation und REM besser zur Komiteebildung, da der Zufallseinfluss bei der Parameterberechnung die Vielfalt gelernter GMMs eher fördert. Durch das Samplen der Parameter werden in jedem Lauf unterschiedliche Lösungen erzwungen. Die Mittelung der Parameter über mehrere Iterationen verringert zwar den Zufallseinfluss, schaltet ihn aber nicht vollständig aus. Ferner ist die Variation der GMM-Größen unproblematischer als beim EM-Verfahren und wird vom Lernalgorithmus selbst automatisch vorgenommen, d.h. in verschiedenen Läufen des Verfahrens werden mitunter unterschiedlich große GMMs gelernt.

In Abschnitt 7.6 werden die Effekte der Komiteebildung mit den verschiedenen Lernverfahren anhand von experimentellen Ergebnissen dargestellt. Es zeigt sich bei allen Lernverfahren eine Verbesserung der Ergebnisse, wobei sich der Grad der Unterschiedlichkeit der Komiteemitglieder deutlich auf die Ergebnisse auswirkt.

Kapitel 7

Experimenteller Vergleich

7.1 Übersicht

In diesem Kapitel sollen die in den vorangegangenen Kapiteln entwickelten Techniken experimentell anhand verschiedener Datensätze verglichen werden. Bei den einzelnen Untersuchungen wurden die randomisierten Ansätze *Data Augmentation* und *REM* untereinander sowie mit mehreren Varianten des EM-Algorithmus' verglichen, um die Aussagen der vorangegangenen Kapitel auch experimentell zu bestätigen.

Bei der Durchführung der Experimente wurde darauf Wert gelegt, die Ergebnisse nachvollziehbar und in Zahlen fassbar zu halten sowie auf eine breite Datenbasis zu stellen. Das heißt, die einzelnen Experimente wurden stets auf verschiedenartigen Datensätzen durchgeführt, um die Ergebnisse verallgemeinern zu können. Durch die vielfache Wiederholung der Experimente mit unterschiedlichen Initialisierungen konnten die Schlussfolgerungen darüberhinaus mit stochastischen Testmethoden (insbesondere t-Test) abgesichert werden. Als Gütemaß für die Qualität einer Dichteschätzung wird stets die Log-Likelihood auf unabhängigen Testmengen verwendet.

Im Gegensatz zu bisher veröffentlichten experimentellen Untersuchungen über MCMC-basierte Lernverfahren für GMMs [Diebolt 94, Celeux 95, Stephens 00] erlauben die in diesem Kapitel vorgestellten Experimente erstmals einen umfassenden und systematischen Vergleich der Leistungsfähigkeit der randomisierten Vorgehensweise. Die Verwendung der Likelihood auf Testmengen als Gütemaß vermeidet die Bewertung der Anpassungsgüte anhand unscharfer oder vom individuellen Eindruck des Betrachters abhängiger Kriterien, wie sie in bisherigen Veröffentlichungen verwendet wurden. Dadurch wird die Objektivität und Vergleichbarkeit der Ergebnisse gewährleistet.

In Abschnitt 7.2 werden zunächst die beiden Grundprinzipien *Samplen von Parametern* und *Maximum-Likelihood* gegenübergestellt. Um den Einfluss unterschiedlicher GMM-Größen zu vermeiden, wird lediglich ein GMM mit einer Komponente, d.h. eine einfache Normalverteilung gelernt. Dadurch entfällt sowohl die Bestimmung der GMM-Größe als auch die Bestimmung von Daten-Zuordnungen, d.h. der I- bzw. E-step. Verglichen wird daher im Wesentlichen der P-step von Data Augmentation mit dem Maximum-Likelihood-Schätzer entsprechend dem M-step des EM-Algorithmus'. Es zeigt sich, dass bereits für diesen einfachen Fall das Samplen von Parametern bessere Ergebnisse liefert als die Maximum-Likelihood-Methode, insbesondere bei kleinen Trainingsmengen und höherdimen-

sionalen Daten.

Im darauffolgenden Abschnitt 7.3 werden die verschiedenen Samplingalgorithmen miteinander verglichen, d.h. Data Augmentation und REM mit verschiedenen Initialisierungen und Samplingverteilungen. Es stellt sich heraus, dass REM etwas bessere Ergebnisse liefert als Data Augmentation und dass eine Veränderungen der Samplingverteilung keine Verbesserung bringt. In den weiteren Untersuchungen wird daher nur noch der REM-Algorithmus als Vertreter der Samplingalgorithmen verwendet.

Der Abschnitt 7.4 stellt den zentralen experimentellen Vergleich von REM mit dem EM- und SAEM-Ansatz dar. Hierbei werden die Verfahren auf der Gesamtaufgabe *Bestimme ein bestes GMM für eine gegebene Datenmenge unbekannter Verteilung* miteinander verglichen, d.h. es sind sowohl die Modellgröße als auch die Modellparameter zu lernen. Es zeigt sich, dass REM auf nahezu allen Datensätzen mindestens so gut arbeitet wie das EM- und SAEM-Verfahren und darüberhinaus auf vielen Datensätzen signifikant bessere Ergebnisse liefert. Die Verbesserungen werden sowohl durch eine bessere Generalisierung als auch eine bessere Exploration erzielt. Die Ergebnisse dokumentieren den großen Fortschritt, der durch den Einsatz von REM beim Lernen von Mischverteilungen erreicht werden kann.

In den weiteren Abschnitten dieses Kapitels steht die Untersuchung einzelner Aspekte der Lernverfahren im Mittelpunkt: der experimentelle Vergleich von EM und REM bei bekannter Modellgröße in Abschnitt 7.5, die Auswirkungen von Kommiteebildung in Kombination mit dem REM- und EM-Algorithmus in Abschnitt 7.6, die maximal lernbare Modellgröße im Verhältnis zur Größe der Trainingsmenge in Abschnitt 7.7 sowie Rechenzeitaspekte in Abschnitt 7.8.

Eine ausführliche Beschreibung der verwendeten Datensätze sowie die Dokumentation der experimentellen Ergebnisse im Detail ist in Anhang A zu finden.

7.2 Vergleich P-step und Maximum-Likelihood-Schätzer

In einer ersten Versuchsreihe war das Ziel, die randomisierte Parameterbestimmung an sich mit dem deterministischen Maximum-Likelihood-Prinzip zu vergleichen. Das bedeutet, bei diesen Experimenten sollte der P-step mit dem M-step verglichen werden.

Um störende Effekte durch unterschiedliche GMM-Größen, Komponenten ohne Zuordnungen sowie Diskretisierungsprobleme zu vermeiden, wurde in dieser Versuchsreihe eine einzelne Normalverteilung trainiert, d.h. das GMM wurde auf eine einzelne Komponente reduziert. Dadurch werden stets alle Trainingsmuster dieser einen Komponente zugeordnet, so dass der E- und I-step irrelevant werden.

Um möglichst gut nachvollziehbare und stochastisch sichere Ergebnisse zu bekommen, wurden die Experimente auf künstlich erzeugten Datensätzen durchgeführt, deren Verteilung bekannt war, so dass die Anpassungsgüte auf sehr großen Testmengen ermittelt werden konnte. Hier wurden Testmengen mit je 10000 Datenpunkten verwendet.

Verglichen wurden vier verschiedene Schätzer:

(I) $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$
Der Maximum-Likelihood-Schätzer, entspricht dem M-step des EM-Algorithmus'

(II) $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n} x_i x_i^T$
Erwartungstreue Variante von (I)

Dimensionalität der Daten	Größe der Trainingsmenge 10	20	50	100
univariat	I	I	I	II
bivariat	III	III	III	II
trivariat	III	III	III	III
9-variat	–	III	III	III

Tabelle 7.1: Beste Varianzschätzer für normalverteilte Daten.

(III) $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\hat{\Sigma} = \frac{1}{n-d-1}\sum_{i=1}^{n} x_i x_i^T$
Erwartungswert der Daten-Posterioriverteilung bei Verwendung nichtinformativer Prioriverteilungen, entspricht dem erwarteten Parametersatz des P-steps bei Mittelung über alle gesampleten Parameter

(IV) $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\hat{\Sigma} = \frac{1}{n+d+1}\sum_{i=1}^{n} x_i x_i^T$
Maximum-a-posteriori-Schätzer bei Verwendung nichtinformativer Prioriverteilungen

Zunächst wurden Experimente mit uni-, bi-, tri- und 9-variaten normalverteilten Daten durchgeführt, wobei die Trainingsmengen zwischen 10 und 100 Datenpunkte groß waren. Tabelle 7.1 führt die jeweils besten Schätzer auf, Tabelle A.2 zeigt darüberhinaus die mittlere Log-Likelihood auf den Testdaten.

Der Maximum-Likelihood-Schätzer (I) stellte sich nur auf univariaten Daten als das beste Verfahren heraus. Dagegen erzielte der Erwartungswert der Posterioriverteilung (III) auf höherdimensionalen Daten fast durchweg die besten Ergebnisse der verglichenen Schätzer. Die Größe der Trainingsmenge hat offensichtlich nur einen geringen Einfluss auf die Rangfolge der vier betrachteten Schätzer. Überraschender Weise erwies sich der erwartungstreue Schätzer (II) nur auf großen Trainingsmengen im uni- und bivariaten Fall als überlegen.

Um das Verhalten der Schätzer auch auf nicht-normalverteilten Daten zu testen, wurden in weiteren Versuchsreihen Daten aus einer Verteilung mit dreiecksförmiger Dichte sowie gleichverteilte Daten verwendet. Die Tabellen A.3 und A.4 dokumentieren die Ergebnisse. Sowohl bei univariaten als auch bei trivariaten Daten stellte sich der Erwartungswert der Posterioriverteilung (III) als der beste Schätzer heraus.

Die drei bisher betrachteten Verteilungen liegen sehr kompakt in einem kleinen Bereich des Datenraums. Um das Verhalten der Schätzer für Daten aus langschwänzigen[1] Verteilungen zu untersuchen, wurden in weiteren Versuchsläufen Daten aus einer Laplaceverteilung (Tabelle A.5) sowie aus einer t-Verteilung mit 5 Freiheitsgraden (Tabelle A.6) verwendet. Beide Verteilungen streuen stärker auch in weit vom Erwartungswert entfernten Bereichen als die Normalverteilung.

Während für die t_5-verteilten Daten wiederum der Erwartungswert der Posterioriverteilung (III) sowohl für uni-, bi, wie auch trivariate Daten die besten Ergebnisse lieferte, erwies sich dieser Schätzer bei der Laplaceverteilung nur für tri- und 9-variate Datensätze als überlegen. Diese Eigenschaft ist sehr ähnlich dem Verhalten auf normalverteilten Daten.

Zusammenfassend kann festgestellt werden, dass sich der Erwartungswert der Daten-Posteriori-Verteilung bei Verwendung nichtinformativer Prioriverteilungen bereits für einfache Normalverteilungsmodelle bewährt. Insbesondere für multivariate Daten schlägt er

[1] englisch: *heavy tailed*

den Maximum-Likelihood-Schätzer genauso wie den Maximum-a-posteriori Ansatz. Diese Beobachtung stützt die Verwendung des P-steps in Kombination mit der Bildung gleitender Durchschnitte über die gesamplete Folge von Parametern, da diese Vorgehensweise den Erwartungswertschätzer approximiert.

7.3 Vergleich von Data Augmentation und REM

Nun sollen die Samplingverfahren Data Augmentation und REM in verschiedenen Varianten untereinander auf den Datensätzen A1–A6 verglichen werden. Hierzu wurden mit den verschiedenen Verfahren auf jedem Datensatz je 100 GMMs trainiert, wobei stets unterschiedliche Trainings- und Testmengen verwendet wurden. Die verwendeten Datensätze sind in Anhang A.1 beschrieben.

Data Augmentation wurde in der Weise verwendet, wie es in Kapitel 5 beschrieben wurde, d.h. mit nichtinformativen Prioriverteilungen, Elimination überzähliger Komponenten, Parameteroptimierung durch Bildung gleitender Durchschnitte über 50 Iterationen sowie Modellselektion durch Analyse der Likelihood auf der Trainingsmenge. Das initiale GMM wurde zufällig gewählt, die Anzahl Komponenten war doppelt so groß wie die Größe der Trainingsmenge.

Das REM-Verfahren wurde in vier Varianten untersucht: (a) die Grundform mit dem vom P-step abgeleiteten randomisierten M-step, nichtinformativen Prioriverteilungen, Initialisierung wie bei Data Augmentation, gleitende Durchschnitte über die Parameter der letzten 50 Iterationen und Modellselektion durch Vergleich der Likelihood auf der Trainingsmenge. Variante (b) unterscheidet sich in einer anderen Initialisierung: statt eines sehr großen GMM wurde ein kleines GMM verwendet, dieses aber mit dem k-means-Verfahren vorverarbeitet. Dadurch verringert sich der Rechenaufwand. Zwei weitere Varianten (c) und (d) unterscheiden sich von (b) in einer veränderten Samplingverteilung gemäß Abschnitt 6.3.3 mit dem Parameter f zu 1.5 bzw. 2 gesetzt.

Tabelle 7.2 führt für jeden Benchmark das jeweils beste Verfahren auf. In den Tabellen A.7 und A.8 können die detaillierten Ergebnisse nachgeschlagen werden.

Bei Vergleich der Ergebnisse fällt auf, dass die Veränderung der Samplingverteilung mit Hilfe des Parameters f keine positiven Auswirkungen auf die Ergebnisse hat. Im Gegenteil, durch die geringere Dynamik der Markov-Kette konzentriert sich das Verfahren auf ungeeignete Bereiche. Ferner wird das Wegfallen von Komponenten erschwert, so dass die gelernten GMMs deutlich größer sind als bei $f = 1$.

Der Vergleich von Data Augmentation und REM deutet darauf hin, dass REM etwas bessere Ergebnisse liefert als Data Augmentation. Ein Vergleich der Test-log-likelihood mit dem t-Test (siehe z.B. [Bosch 93]) liefert für die Datensätze A1, A5 und A6 signifikant bessere Ergebnisse für REM (5% Signifikanzniveau), während für A2 und A4 signifikant schlechtere Ergebnisse erreicht werden. Berücksichtigt man ferner das Ergebnis auf den Datensätzen B1–B7 (siehe Tabelle A.8), so stellt sich REM für B1, B2 und B3 als signifikant besser dar, während für B4, B5 und B6 keine signifikanten Unterschiede festgestellt werden können. Im Ganzen zeichnen sich also etwas bessere Ergebnisse für REM als für Data Augmentation ab.

Beim Vergleich der unterschiedlichen Initialisierungen des randomisierten EM-Verfahrens zeigt sich ein unklares Bild: auf den Datensätzen A1 und A3 sind beide Varianten gleich gut, auf A2 und A4 ist die Zufallsinitialisierung eines sehr großen GMMs besser, aber mit einem P-Wert von 18% bzw. 23% des t-Tests nicht signifikant. Auf Datensatz A6 ist die

Benchmarksatz	bestes Verfahren
A1	REM/k-means
A2	Data Augmentation
A3	REM/k-means
A4	Data Augmentation
A5	REM/k-means
A6	REM
B1	REM/k-means
B2	REM/k-means
B3	REM/k-means
B4	Data Augmentation
B5	REM/k-means
B6	REM/k-means

Tabelle 7.2: Beste Samplingverfahren auf den Benchmarkdatensätzen. *REM/k-means* steht für die Variante von REM mit kleinen initialen GMMs, die mit Hilfe von k-means vorverarbeitet wurden

Zufallsinitialisierung signifikant (P-Wert 1.1%) besser, auf Datensatz A5 dagegen signifikant schlechter (P-Wert 0.00% bei umgekehrtem Vergleich).

Ein Vorteil der Initialisierung der Verfahren mit einer kleineren Mischverteilung besteht in dem geringeren Rechenaufwand. Während bei Initialisierung mit einem sehr großen Modell zunächst viele E- und D-steps benötigt werden, um deplatzierte Komponenten zu entfernen und das Modell auf eine den Daten angemessene Größe zu reduzieren, entfällt dieser Aufwand bei kleineren initialen GMMs. Zu beachten ist hierbei auch, dass in jedem D-step nur eine einzige Komponente entfernt werden darf, so dass sich die Verkleinerung des GMMs über viele Iterationen von E- und D-steps hinziehen kann.

Allerdings ist zu beachten, dass die anfängliche Anordnung der Komponenten eine wichtige Rolle für das resultierende GMM spielt, da bei ungünstiger Anordnung zu viele Komponenten frühzeitig entfernt werden. Die Initialisierung mit einem sehr großen GMM wirkt dem entgegen, da durch die Vielfalt der initialen Komponenten der Parametrraum sehr gut abgedeckt ist. Einen ähnlichen Effekt erhält man durch die Vorverarbeitung des initialen GMMs mit Hilfe von k-means Clustering, da mit dieser Technik die Komponenten bereits an guten Ausgangspositionen angeordnet sind.

Aufgrund des geringeren Rechenaufwandes wird in den weiteren Experimenten REM stets mit der kleineren, vorverarbeiteten Initialisierung verwendet.

7.4 Vergleich randomisiertes und deterministisches Lernen

In diesem Abschnitt sollen das Verhalten von deterministisch arbeitenden und randomisierten Lernverfahren für GMMs verglichen werden. Dies erfolgt anhand der Datensätze A1–A6 und B1–B6 aus Anhang A.1. Verglichen werden der deterministische EM-Algorithmus, der SAEM-Algorithmus sowie REM. Zur Bestimmung der Modellgröße wurden AIC, BIC, Evidenz und Kreuzvalidierung eingesetzt. Die detaillierten Ergebnisse dieser Untersuchungen befinden sich in Anhang A.2.

Datensatz	bestes Vergleichsverfahren	Signifikante Veränderungen
A1	SAEM Evidenz	▲
A2	SAEM BIC	▼
A3	SAEM Evidenz	
A4	SAEM AIC	▲
A5	SAEM BIC	▲
A6	SAEM cross-validation	▼
B1	SAEM AIC	▲
B2	SAEM AIC	▲
B3	SAEM AIC	▲
B4	EM BIC	
B5	EM AIC	
B6	SAEM BIC	

Tabelle 7.3: Vergleich von REM und verschiedenen EM-Varianten auf den Benchmarkdaten. Ein ▲ in der dritten Spalte signalisiert ein signifikant besseres Abschneiden von REM gegenüber der besten EM-Variante, ein ▼ signalisiert ein signifikant schlechteres Abschneiden. Die detaillierten Ergebnisse sind in den Tabellen A.9 und A.10 zu finden.

Für jeden Datensatz und jedes untersuchte Lernverfahren wurde auf jedem der 100 Trainingsmengen ein Modell trainiert und auf den zugehörigen Testmengen die Log-Likelihood gemessen. Anschließend wurden die 100 Einzelergebnisse gemittelt und die Standardabweichung berechnet.

Der EM- und SAEM-Algorithmus wurden wiefolgt eingesetzt. Zu gegebener GMM-Größe wurde ein initiales GMM mit Hilfe von k-means-Clustering bestimmt und dieses mit dem EM (SAEM)-Algorithmus 300 Iterationen lang trainiert. Um die geeignete Modellgröße zu bestimmen, wurden diese Experimente für GMM-Größen von einer Komponente bis zu maximal 10 Komponenten wiederholt und anschließend das Modellselektionskriterium berechnet, z.B. AIC. Als resultierendes Modell wurde das kleinste Modell gewählt, bei der das Modellselektionskriterium einen besseren Wert annahm als für die nächst größere Modellkomplexität, d.h. es wurde mit kleinen Modellen begonnen, bis sich der Wert des Modellselektionskriteriums verschlechterte.

Abweichend hiervon wurde bei der Kreuzvalidierung vorgegangen. Hierbei wurden die Trainingsmengen zunächst in je fünf Teilmengen unterteilt, und Modelle für die Vereinigung von jeweils vier der fünf Teilmengen trainiert. Durch Auswerten der Likelihood auf der fünften Menge sowie Mittelung über alle möglichen Aufteilungen der fünf Mengen wurde der Wert für das Modellselektionskriterium ermittelt (fünffache Kreuzvalidierung). Anschließend wurde der beste Wert unter den 10 möglichen GMM-Größen ermittelt und für diese Modellgröße ein Modell auf allen Trainingsdaten gelernt.

REM wurde mit kleinen (10 Komponenten), mit k-means-Clustering vorverarbeiteten GMMs initialisiert und 1000 Iterationen trainiert. Die ausführlichen Ergebnisse des experimentellen Vergleichs sind in Tabelle A.9 und A.10 angegeben. Tabelle 7.3 vergleicht in knapper Form den REM-Ansatz mit der jeweils besten Variante des EM-Algorithmus'.

Der Vergleich des EM- bzw. SAEM-Algorithmus einerseits und von REM andererseits zeigt überwiegend bessere Ergebnisse für REM. Auf den Datensätzen A1, A4 und A5 arbeitet

das Verfahren signifikant besser, für A2, A3 und A6 sind die Ergebnisse uneinheitlich: überwiegend leichte Verbesserungen, aber teilweise auch signifikante Verschlechterungen.

Bei Betrachtung der Datensätze B1–B6 hingegen ergeben sich fast nur Verbesserungen durch den Einsatz von REM. Lediglich für B6, kombiniert mit dem BIC zur Modellselektion zeigt sich REM leicht, aber nicht signifikant, unterlegen.

Zu beachten ist ferner die Anzahl Läufe, in denen der EM- bzw. SAEM-Algorithmus GMMs erzeugt hat, die einzelne Testdatenpunkte überhaupt nicht beschreiben konnten, d.h. für die die Dichte des gelernten GMMs numerisch Null war. Diese Fälle wurden bei der Berechnung der Mittelwerte, Standardabweichungen und Teststatistiken gar nicht berücksichtigt. Sie stellen, obwohl nur selten aufgetreten, ein großes Problem dar, da sie auf Overfitting hinweisen. Offensichtlich sind die Komponenten dieser GMMs so stark degeneriert, dass bereits in geringem Abstand vom Erwartungswert der Komponenten die Modelldichte praktisch Null ist. Das bedeutet, zumindest ein Eigenwert der Kovarianzmatrix ist nahezu Null. Nicht verwunderlich ist, dass derartige Fälle hauptsächlich auf dem Datensatz B2 auftraten, der sehr stark ausgeprägte diskrete Strukturen aufweist. Dagegen traten derlei Effekte bei REM nicht auf.

Auffällig sind einige sehr große Standardabweichungen der Test-log-Likelihood von EM und SAEM auf den Datensätzen B2 und B3 für das Evidenzkriterium und für Kreuzvalidierung. Diese Werte sind auf degenerierte GMMs zurückzuführen: da die Testmengen ebenso wie die Trainingsmengen diskrete Datenpunkte aufweisen, führen degenerierte Komponenten mitunter zu einer sehr großen Test-log-Likelihood, je nachdem an welchen Stellen die Degenerierung erfolgt. Im Prinzip wäre hier dann die Test-log-Likelihood unendlich. Da aber die verwendete Implementierung des EM-Ansatzes aus numerischen Gründen die Determinante der Kovarianzmatrizen nach unten begrenzt, werden stets endliche Werte angenommen. Die Werte selbst sind also Artefakte der Implementierung. Man beachte, dass degenerierte Komponenten vermieden werden sollten, da sie sich extrem an die Trainingsdaten überangepasst haben.

Eine überraschende Beobachtung ist die Tatsache, dass REM für viele Datensätze nicht nur auf den Testdaten, sondern auch auf den Trainingsdaten eine größere Likelihood aufweist (siehe Tabelle A.11). Dies verwundert, da der EM-Ansatz als Maximum-Likelihood-Methode gerade die Maximierung dieser Größe als Ziel besitzt, während die Samplingverteilung von REM vollkommen unabhängig von der Daten-Likelihood ist. Lediglich bei der Auswahl des besten der gemittelten GMMs wird die Daten-Likelihood beachtet. Dies verdeutlicht, dass REM den Parameterraum besser exploriert als der EM-Ansatz und dadurch schneller bessere Bereiche des Parameterraums findet, während der EM-Algorithmus in suboptimalen Maxima der Likelihoodfunktion verharrt.

Bessere Ergebnisse von REM auf den Testdaten lassen sich jedoch nicht ausschließlich auf eine bessere Exploration des Parameterraums zurückführen. So weist REM eine durchweg signifikant schlechtere Likelihood auf den B4-Trainingsdaten auf als EM und SAEM, während auf den Testdaten durchweg bessere Ergebnisse erzielt werden. Hier zeigt sich eine bessere Generalisierungsleistung und ein geringerer Trend zum Overfitting.

Insgesamt erweist sich also im direkten experimentellen Vergleich REM als robuster gegen Overfitting und Abweichungen von der Annahme kontinuierlich verteilter Daten und ist vielfach in der Lage, deutlich bessere Ergebnisse zu erzielen als der EM- oder SAEM-Ansatz. Nichtsdestotrotz gibt es einzelne Datensätze, für die der EM-Algorithmus bessere Ergebnisse erbringt.

Verfahren	Test-log-Likelihood	Trainings-log-Likelihood	P-Wert
REM	-26104 ± 243.9	-1529 ± 28.17	
Data Augmentation	-26175 ± 246.3	-1536 ± 28.33	2.22% ▲
EM	-26442 ± 359.3	-1540 ± 32.15	0.00% ▲
SAEM	-26439 ± 353.7	-1540 ± 32.12	0.00% ▲

Tabelle 7.4: Vergleich der Test-log-Likelihood auf Datensatz A4 bei vorgegebener GMM-Größe von fünf Komponenten. Die Spalte *P-Wert* stellt den P-Wert des einseitigen t-Tests zum Vergleich der Test-log-Likelihood von REM und dem jeweiligen Vergleichsverfahren dar, das Zeichen ▲ weist auf eine signifikante Verbesserung hin.

7.5 Vergleich bei bekannter GMM-Größe

Während die Experimente in Abschnitt 7.4 von einer vollkommen unbekannten GMM-Größe ausgingen soll in diesem Abschnitt an Datensatz A4 untersucht werden, inwiefern die besseren Ergebnisse für REM auf die Größenbestimmung zurückzuführen sind.

Da die Daten von Datensatz A4 (siehe Anhang A.1) künstlich aus einem GMM mit fünf Komponenten erzeugt wurden, sollten die Lernverfahren in der Lage sein, bei Kenntnis der korrekten GMM-Größe bessere Ergebnisse zu erzielen als im allgemeinen Fall, in dem die Größe mitbestimmt werden muss.

Hierzu wurden REM, Data Augmentation, der EM-Algorithmus sowie SAEM mit einem GMM aus fünf Komponenten auf den A4-Daten trainiert und die Likelihood auf den Testmengen bestimmt. Die initialen GMMs wurden mit *k-means*-Clustering vorverarbeitet. Tabelle 7.4 führt die Ergebnisse dieser Versuche auf.

Es wird deutlich, dass das bessere Abschneiden von REM gegenüber EM und SAEM in Abschnitt 7.4 nicht nur auf die Bestimmung der GMM-Größe zurückzuführen ist, sondern auch auf eine bessere Parameterschätzung. Zwar verbessern sich die Lernergebnisse für den EM-Ansatz bei Vorgabe der passenden Größe leicht gegenüber den besten Ergebnissen mit variabler Größe, allerdings bleibt REM weiterhin deutlich überlegen. Wiederum schneidet REM auch bei Vergleich der Trainings-log-Likelihood besser ab als EM und SAEM, obwohl es an sich kein echter Maximum-Likelihood-Ansatz ist.

Auffällig ist auch die Tatsache, dass sich REM bei Vorgabe der GMM-Größe leicht verschlechtert gegenüber automatischer Größenanpassung. Dies lässt sich auf eine bessere Initialisierung bei Verwendung zusätzlicher Komponenten zurückführen. Zu beachten ist auch, dass bei selbständiger Wahl der GMM-Größe REM im Durchschnitt 5.9 Komponenten wählt, d.h. die gefundene GMM-Größe liegt etwas über der Anzahl in den Daten enthaltener Komponenten.

7.6 Komitee-Effekt

Die in Abschnitt 6.4 dargestellte Möglichkeit der Komiteebildung soll an dieser Stelle durch experimentelle Ergebnisse untermauert werden. Hierzu wurden Komitees verschiedener Größe auf den Datensätzen A1, A2 und A6 trainiert und die Likelihood auf den Testmengen verglichen.

Abbildung 7.1 zeigt die Ergebnisse in Diagrammform. Es wurden Komitees aus bis zu 20

Mitgliedern gebildet, wobei die einzelnen Komiteemitglieder mit dem REM-Verfahren bzw. mit dem EM-Algorithmus trainiert wurden. Beim EM-Algorithmus wurde die Modellgröße der Komiteemitglieder mit dem AIC-, BIC- bzw. Evidenzansatz bestimmt, wobei zwei Varianten zu unterscheiden sind: zum einen wurden Komitees gebildet, bei denen alle Komiteemitglieder die gleiche GMM-Größe besaßen (im Diagramm mit durchgezogener Linie dargestellt), zum anderen Komitees mit Komiteemitgliedern unterschiedlicher Größe (gestrichelte Linien). Die in Abbildung 7.1 dargestellten Werte wurden durch Mittelung über 100 Läufe mit unterschiedlichen Trainingsmengen bestimmt.

Im Rahmen der Experimente konnten folgende Beobachtungen gemacht werden:

- Sowohl für REM als auch für den EM-Algorithmus kann ein Komitee-Effekt beobachtet werden. Die Test-log-Likelihood wächst mit zunehmender Anzahl Komiteemitglieder an. Dies bestätigt die Erwartung, dass ein Komitee-Effekt nicht nur im überwachten Lernen sondern auch bei der Dichteschätzung auftritt.

- Die Verbesserung der Ergebnisse durch Hinzunahme eines weiteren Komiteemitglieds werden umso geringer, je größer die Komitees werden. Auch dieser Effekt deckt sich mit dem erwarteten Verhalten.

- Beim EM-Algorithmus spielt die Frage des verwendeten Modellselektionskriterium eine wichtige Rolle für das Ausmaß des Komitee-Effektes: das BIC-Kriterium – obwohl für einzelne Läufe besser als AIC und Evidenz – erlaubt nur einen geringen Komitee-Effekt, selbst bei großen Komitees. Dieses Verhalten lässt sich darauf zurückführen, dass BIC kleinere GMMs bevorzugt, diese aber aufgrund der Konvergenzeigenschaften des EM-Algorithmus' ähnlicher sind als große GMMs. Dadurch ist die stochastische Unabhängigkeit der Komiteemitglieder beeinträchtigt. AIC und Evidenz dagegen wählen größere GMMs aus (vergleiche auch Tabelle A.9) und erlauben damit die Herausbildung unterschiedlicher Modellausprägungen. Dementsprechend ist auch der Komitee-Effekt größer.

- Die Verwendung von Komiteemitgliedern gleicher Modellgröße beim EM-Verfahren verringert den Komitee-Effekt gegenüber der Verwendung unterschiedlich großer Modelle. Insbesondere mit dem AIC- und Evidenzkriterium zur Modellselektion sind die Unterschiede eklatant. Der Grund liegt in der größeren Unterschiedlichkeit der Komiteemitglieder bei Verwendung unterschiedlich großer GMMs.

- Beim Vergleich von REM und dem EM-Ansatz liegt der Komitee-Effekt ungefähr auf gleichem Niveau bei Verwendung unterschiedlich großer Komiteemitglieder und AIC bzw. Evidenz zur Modellselektion. Mit dem BIC oder bei Verwendung von Komiteemitgliedern gleicher Größe dagegen ist der Komitee-Effekt von REM deutlich größer als bei EM. Zu beachten ist allerdings, dass REM selbst stets unterschiedlcih große Komiteemitglieder verwendet.

Aus den Diagrammen wird deutlich, dass bei sehr großen Komitees der zusätzliche Trainingsaufwand in keinem günstigen Verhältnis mehr zur Verbesserung der Ergebnisse steht. Daher wurden in den nachfolgend dargestellten Untersuchungen jeweils Komitees mit maximal 10 Komiteemitgliedern betrachtet. Tabelle A.12 stellt Testergebnisse bei Komiteebildung auf den Datensätzen A1–A6 und B1–B6 den Testergebnissen für einzelne Läufe von

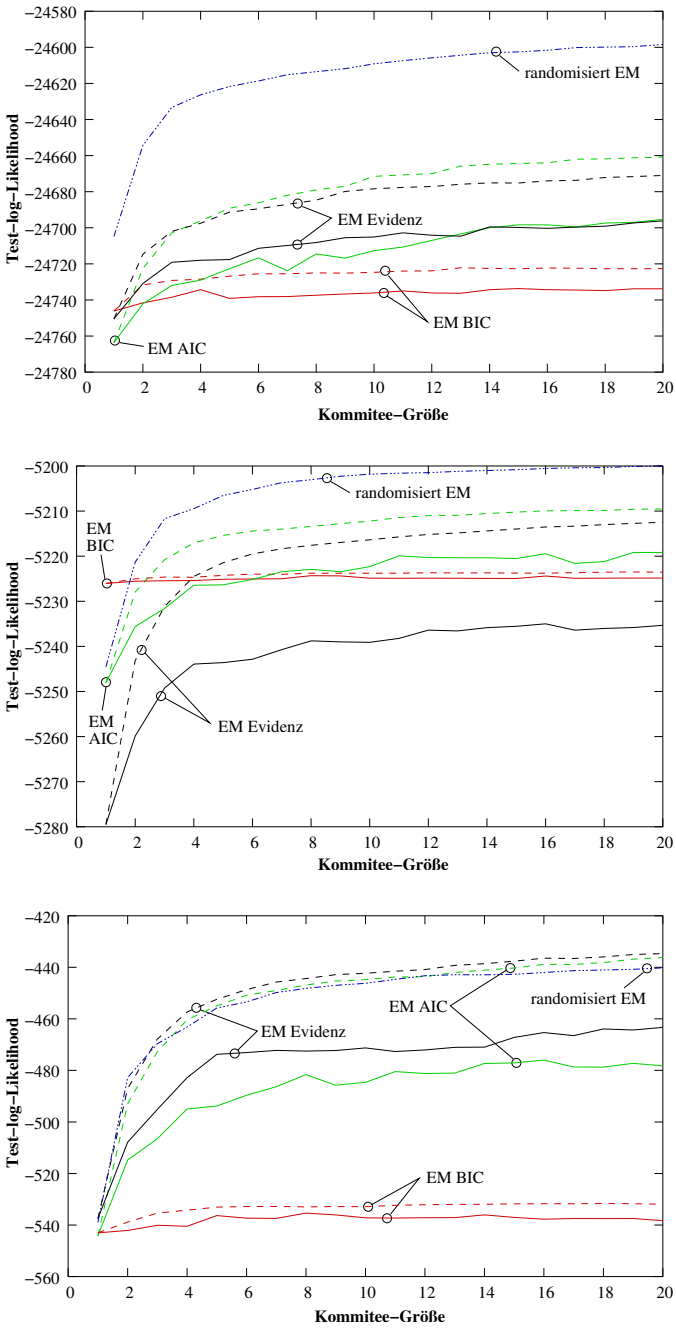

Abbildung 7.1: Darstellung des Komitee-Effektes für Komitees mit maximal 20 Mitgliedern auf den Datensätzen A1 (oben), A2 (Mitte) und A6 (unten). Auf der Abszisse ist die Komitee-Größe aufgetragen, auf der Ordinate die mittlere Test-log-Likelihood. Die Ergebnisse von REM sind mit der Strich-Punkt-Punkt-Punkt-Linie eingezeichnet, die des EM-Verfahrens mit gestrichelter bzw. durchgezogener Linie. Beim EM-Algorithmus wurde nach den drei Modellselektionskriterien AIC, BIC und Evidenz unterschieden. Ferner wurde differenziert danach, ob alle Komiteemitglieder die selbe GMM-Größe besitzen (durchgezogene Linien) oder ob für jedes Komiteemitglied eine eigenständige Bestimmung der Modellgröße stattgefunden hat (gestrichelte Linien).

Datensatz	bestes Vergleichsverfahren	Signifikante Veränderungen
A1	SAEM cross-validation	▲
A2	SAEM cross-validation	▲
A3	EM Evidenz	
A4	SAEM cross-validation	▲
A5	SAEM AIC	▲
A6	SAEM cross-validation	
B1	SAEM AIC	▲
B2	SAEM Evidenz	▲
B3	EM Evidenz	▼
B4	SAEM BIC	▲
B5	SAEM Evidenz	▼
B6	SAEM AIC	

Tabelle 7.5: Vergleich von REM mit Kommiteebildung und verschiedenen EM-Varianten mit Kommiteebildung auf den Benchmarkdaten. Ein ▲ in der dritten Spalte signalisiert ein signifikant besseres Abschneiden von REM gegenüber der besten EM-Variante, ein ▼ signalisiert ein signifikant schlechteres Abschneiden. Die detaillierten Ergebnisse sind in den Tabellen A.13 und A.14 zu finden.

REM gegenüber. Deutlich erkennbar ist die höhere Test-log-Likelihood sowie die geringere Standardabweichung der Ergebnisse auf allen betrachteten Datensätzen.

Die Tabellen A.13 und A.14 vergleichen die Ergebnisse bei Komiteebildung zwischen REM und dem EM-Ansatz, wobei beim EM-Algorithmus Komitees mit Mitgliedern gleicher Modellgröße verwendet wurden. Deutlich ist die Verbesserung der Ergebnisse auf den Datensätzen A1–A6 zu erkennen, so dass REM fast durchweg signifikant bessere Ergebnisse erzielt als EM. Tabelle 7.5 stellt den Vergleich zwischen REM und der jeweils besten EM-Variante bei Kommiteebildung in Kurzform dar.

Ebenso konnte REM auf den Benchmark-Datensätzen B1, B2, B4 und B6 den Abstand zum EM-Verfahren durch Komiteebildung vergrößern. Allerdings zeigen die Benchmark-Datensätze B3 und B5, dass im Einzelfall der Komitee-Effekt auch für den EM-Ansatz größer sein kann als für REM. Charakteristischer Weise ist allerdings auch bei diesen Benchmark-Daten das BIC als Modellselektionskriterium bei Komiteebildung ungeeignet.

7.7 Curse of Dimensionality

Unter dem *Fluch der Dimensionalität*[2] versteht man im Bereich der Stochastik und des maschinellen Lernens die Schwierigkeit, mit einer begrenzten Menge höherdimensionaler Trainingsdaten zuverlässig Modelle schätzen zu können. Während viele Modelle mit univariaten Daten oder niedrigdimensionalen Daten gute Ergebnisse liefern, scheitern sie für höherdimensionale Daten, da zur Bestimmung der größeren Anzahl Modellparameter viele Trainingsmuster nötig sind, die oft nicht in ausreichender Anzahl zur Verfügung stehen.

Als Erklärung für dieses ungünstige Skalierungsverhalten für höherdimensionale Daten

[2] englisch: *curse of dimensionality*

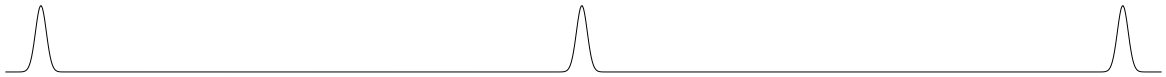

Abbildung 7.2: Beispiel für die Dichte einer Trainingsmenge mit 3 univariaten Clustern

wird vielfach angegeben, dass zum gleichmäßigen Ausfüllen eines höherdimensionalen Volumens exponentiell mehr Datenpunkte erforderlich sind als für niedrigdimensionale Volumina gleicher Kantenlänge. Vergleiche hierzu auch die Argumentation in [Bishop 95]. In diesem Abschnitt soll untersucht werden, welches Verhalten REM für höherdimensionale Daten aufweist und welcher Zusammenhang zwischen Modellgröße, Dimensionalität der Trainingsdaten sowie der Größe der Trainingsmenge besteht.

Um zu prüfen, ob die Größenbestimmung von REM in der Lage ist, prinzipiell mit n Trainingsdaten der Dimensionalität d ein GMM mit k Komponenten zu bestimmen, wurde künstlich eine Trainingsmenge mit n d-dimensionalen Trainingspunkten erzeugt, die in k normalverteilten Clustern angeordnet waren. Die einzelnen Cluster waren sehr weit voneinander entfernt angeordnet, so dass sich keinerlei Überlappungen zwischen den einzelnen Clustern ergaben. Abbildung 7.2 zeigt beispielhaft die Dichte einer derartigen univariaten Trainingsmenge mit drei Clustern.

Anschließend wurde mit REM ein GMM auf dieser Trainingsmenge trainiert, wobei als initiales GMM genau die Mischverteilung verwendet wurde, die zum Erzeugen der Trainingsmuster diente. Nach 2000 Iterationen REM wurde geprüft, ob alle k Komponenten des GMM noch vorhanden waren oder ob ein oder mehrere Komponenten in der Zwischenzeit entfernt worden waren. In ersterem Fall erlaubt REM also mit der gegebenen Trainingsmenge prinzipiell die Berechnung von GMMs mit k Komponenten, ansonsten ist die Menge Trainingsdaten zu gering, um ein GMM dieser Größe zu schätzen. Die Untersuchung sagt nichts darüber aus, wie viele Komponenten bei Trainingsmengen mit anderer Struktur gefunden werden. Dies hängt von der konkreten Verteilung der Trainingsdaten und der Initialisierung des Lernalgorithmus' ab. Aber die Untersuchung erlaubt eine Abschätzung der maximal möglichen Modellgröße, da sowohl die Struktur der Datenverteilung als auch die Initialisierung optimal auf die Berechnung möglichst großer Modelle abgestimmt sind.

Abbildung 7.3 zeigt den Zusammenhang zwischen der Dimensionalität der Trainingsdaten sowie der maximal möglichen Anzahl Komponenten, gemittelt über je fünf Läufe des Experiments und differenziert nach der Gesamtgröße der Trainingsmenge. Wie zu erwarten, fällt die maximale Größe schätzbarer GMMs mit zunehmender Dimensionalität der Daten. Ferner hängt die maximal mögliche GMM-Größe auch von der Gesamtzahl Trainingsmuster ab, wobei der Zusammenhang zwischen GMM-Größe und Anzahl Daten sublinear ist, d.h. bei doppelt so großer Trainingsmenge können nicht doppelt so große GMMs trainiert werden.

Berücksichtigt man die Größe der Trainingsmenge und bildet den Quotienten aus Anzahl Trainingsdaten und maximaler GMM-Größe ergibt sich das Schaubild 7.4. Deutlich ist der Effekt zu erkennen, dass bei höherdimensionalen Daten mehr Trainingsmuster pro GMM-Komponente erforderlich sind. Dies entsprich der Erwartung gemäß dem *Curse of dimensionality*. Allerdings ist – zumindest für maximal 15-variate Daten – kein exponentieller Anstieg dieses Quotienten aus den experimentellen Ergebnissen heraus interpretierbar. Der große Sprung von 15 auf 30 für die Trainingsmenge mit 30 Datenpunkten beruht auf einer Modellverkleinerung von zwei auf eine Komponente; die Höhe des Sprungs ergibt sich also

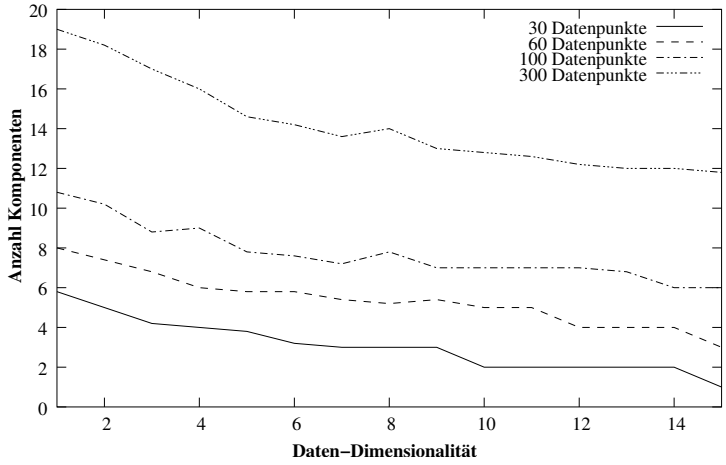

Abbildung 7.3: Zusammenhang zwischen der Dimensionalität der Trainingsdaten und der maximal möglichen GMM-Größe in Abhängigkeit von der Größe der Trainingsmenge.

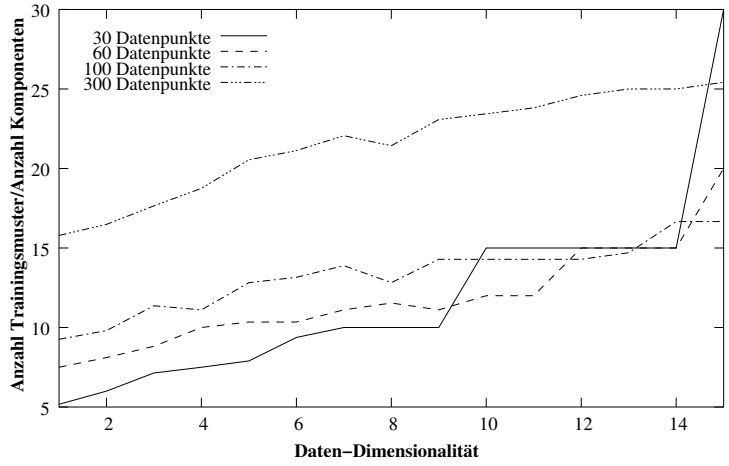

Abbildung 7.4: Zusammenhang zwischen der Dimensionalität der Trainingsdaten und der minimal erforderlichen Anzahl Trainingsdaten pro GMM-Komponente in Abhängigkeit von der Gesamtgröße der Trainingsmenge.

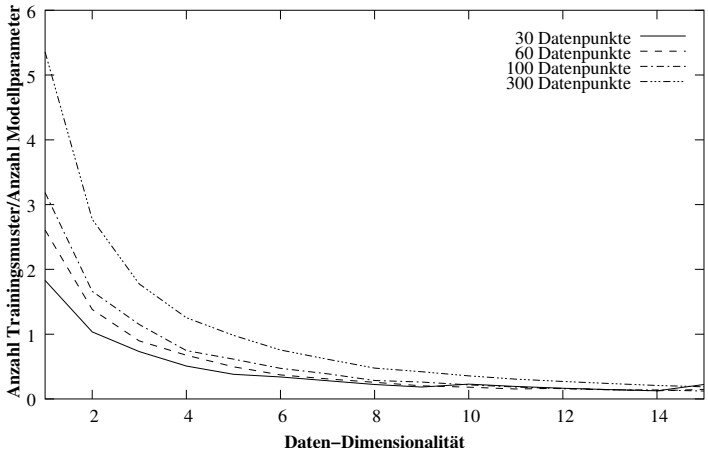

Abbildung 7.5: Zusammenhang zwischen der Dimensionalität der Trainingsdaten und der minimal erforderlichen Anzahl Trainingsdaten pro geschätztem Modellparameter in Abhängigkeit von der Gesamtgröße der Trainingsmenge.

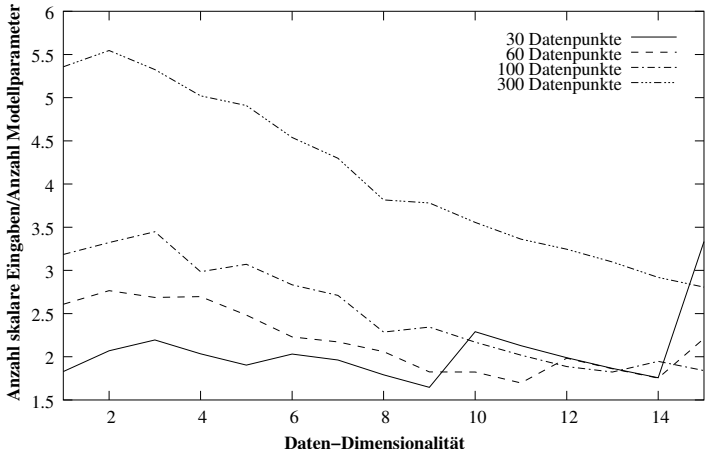

Abbildung 7.6: Zusammenhang zwischen der Dimensionalität der Trainingsdaten und der minimal erforderlichen Anzahl skalarer Einträge in den Trainingsdaten pro geschätztem Modellparameter in Abhängigkeit von der Gesamtgröße der Trainingsmenge. Unter der Anzahl skalarer Einträge in den Trainingsdaten wird hier das Produkt aus der Anzahl Trainingsmuster und der Dimensionalität der Trainingsdaten verstanden.

aufgrund der nur diskret wählbaren Modellgrößen.

Ein weiteres interessantes Phänomen kann man beobachten, wenn man nicht die maximale Anzahl Komponenten betrachtet, sondern die zugehörige Anzahl Modellparameter, da die Anzahl der Modellparameter nicht linear sondern quadratisch mit der Dimension der Daten wächst (vgl. Abschnitt 4.3.3). Abbildung 7.5 zeigt den Quotienten aus der Größe der Trainingmenge und der maximalen Anzahl Modellparameter in Abhängigkeit von der Daten-Dimensionalität und der Gesamtgröße der Trainingsmenge. Überraschender Weise fällt dieser Quotient für höherdimensionale Daten rasch ab und erreicht asymptotisch einen Wert um 0.2, d.h. die Anzahl Modellparameter kann die Anzahl Trainingsdaten um etwa das fünffache übersteigen.

Dieser Effekt bleibt qualitativ auch erhalten, wenn man statt des Quotienten aus Anzahl Trainingsmuster und Modellparameter den Quotienten aus der Anzahl skalarer Einträge in den Trainingsmustern und der Anzahl Modellparameter bildet. Diesen Zusammenhang zeigt das Diagramm 7.6. Man berücksichtigt also, dass ein höherdimensionales Muster mehr Information enthält als ein niedrigdimensionales. Es fällt auf, dass auch dieser Quotient mit zunehmender Daten-Dimensionalität fällt und sich asymptotisch bei einem Wert um 2 einschwingt.

Die Ergebnisse dieser Experimente lassen keine Schlussfolgerungen über die Allgemeingültigkeit des *Fluch der Dimensionalität* zu. Allerdings verdeutlichen die Resultate, dass die Modellkomplexität gemessen in der Anzahl Modellparameter mit zunehmender Dimensionalität der Daten nicht abnimmt, sondern – im Gegenteil – tendenziell sogar etwas zunimmt. Insbesondere ist kein exponentieller Zusammenhang zwischen der Dimensionalität der Daten und der Anzahl Trainingsdaten pro GMM-Komponente feststellbar. Mit REM trainierte GMMs skalieren also gut mit höherdimensionalen Daten.

Die Ergebnisse aus den vorangegangenen Untersuchungen unterstützen diese Aussage nicht nur im Hinblick auf die Modellgröße sondern auch die Anpassungsgüte. So konnte in Abschnitt 7.2 experimentell gezeigt werden, dass das dem P-step/randomisierten M-step zugrundeliegende Schätzprinzip gerade für höherdimensionale Datensätze besser geeignet ist als der Maximum-Likelihood-Ansatz. Auch die Tabellen A.9 und A.10 weisen vergleichsweise sehr gute Ergebnisse auf den höherdimensionalen Datensätzen A5, B2, B3 und B5 auf.

7.8 Rechenzeit

Abschließend soll die Rechenzeit betrachtet werden, die die Ausführung von REM benötigt. Die Experiemente wurden auf einem handelsüblichen PC mit Intel Pentium 4 Prozessor und 2.6GHz Taktrate durchgeführt. Die Software wurde in C++ geschrieben und mit dem gcc-3.3 unter Linux übersetzt. Da es sich um Entwicklungssoftware handelte, war sie nicht auf minimiale Laufzeit hin optimiert. Die Ergebnisse können also nur eine grobe Einschätzung der benötigten Rechenzeit liefern.

Tabelle 7.6 stellt die benötigte mittlere Rechenzeit auf den Datensätzen A1–A6 und B1–B6 dar. Es wurden zwei Varianten untersucht: REM mit kleinen initialen GMMs, die mit k-means-Clustering vorverarbeitet wurden sowie REM mit großen zufälligen initialen GMMs.

Im Fall kleiner initialer GMMs ist die Rechenzeit sehr gering und liegt im Bereich bis maximal 17 Sekunden, selbst auf den höherdimensionalen Datensätzen A5, B2 und B3. Im Fall großer initialer GMMs wird dagegen sehr viel Zeit benötigt, um zunächst einmal die

Datensatz	Rechenzeit in Sekunden	
	kleine Initialisierung	große Initialisierung
A1	6.862 ± 0.69	298.703 ± 6.90
A2	6.101 ± 0.90	113.967 ± 2.54
A3	1.571 ± 0.24	17.029 ± 0.50
A4	3.901 ± 0.21	183.985 ± 6.19
A5	15.342 ± 2.50	430.504 ± 5.79
A6	4.388 ± 0.36	28.189 ± 0.56
B1	6.461 ± 0.60	163.205 ± 16.31
B2	11.717 ± 1.25	93.665 ± 1.67
B3	16.607 ± 1.69	318.482 ± 4.92
B4	2.315 ± 0.32	15.913 ± 0.31
B5	4.157 ± 0.56	53.967 ± 0.67
B6	1.430 ± 0.20	5.218 ± 0.23

Tabelle 7.6: Mittlere Rechenzeit für 1000 Iterationen von REM bei kleiner Initialisierung (10 Komponenten, vorverarbeitet mit k-means) und großer Initialisierung (initiales GMM hat doppelt soviele Komponenten wie Trainingsdaten vorhanden sind, Zufallsinitialisierung).

GMM-Größe zu reduzieren, so dass sich erheblich längere Rechenzeiten ergeben. Hierbei ist zu beachten, dass nach jedem Löschen einer Komponente erneut ein E-step mit dem verbleibenden GMM durchgeführt werden muss, so dass zur Entfernung von k' Komponenten k' E-steps mit großen GMMs durchgerechnet werden müssen, d.h. es müssen $\frac{nk'(k'+1)}{2}$ Zuordnungswahrscheinlichkeiten h_{ij} berechnet werden.

Ein quantitativer Vergleich der Rechenzeit mit dem EM-Algorithmus soll an dieser Stelle nicht geschehen, da die Vergleichbarkeit der Ansätze nicht gegeben ist. Es kann davon ausgegangen werden, dass eine einzelne Iteration EM etwas schneller ist als eine Iteration REM, da das Samplen der Parameter entfällt. Allerdings ist zu beachten, dass die Bestimmung einer geeigneten GMM-Größe beim EM-Algorithmus mehrere Läufe mit unterschiedlich großen GMMs erfordert, so dass die Rechenzeit eines einzelnen Laufs nichts über die Gesamtzeit verrät, die benötigt wird.

Kapitel 8

Bewertung der Ansätze

8.1 EM-Algorithmus, REM und Varianten

In den Kapiteln 5, 6 und 7 wurde das Verhalten randomisierter Lernverfahren für GMMs untersucht und mit dem EM-Ansatz verglichen. Die Ergebnisse verdeutlichen, dass randomisierte Lernverfahren für GMMs sehr gut geeignet sind und den EM-Ansatz in vielen Fällen schlagen.

In Abschnitt 7.4 wurden die Algorithmen auf Benchmarkdaten verglichen. Es stellte sich heraus, dass der EM-Ansatz zwar auf einigen Benchmarks etwas besser als REM arbeitet, auf den meisten anderen allerdings schlechtere Ergebnisse liefert. Das bessere Verhalten von REM trat auch auf, wenn die GMM-Größe im Vorhinein bekannt war (Abschnitt 7.5) oder wenn das GMM bis auf eine einzige Komponente reduziert wurde (Abschnitt 7.2). Die besseren Ergebnisse sind also nicht nur auf die Bestimmung der GMM-Größe zurückzuführen, sondern auch auf die randomisierte Arbeitsweise als algorithmisches Konstruktionsprinzip.

Dabei zeigte sich sowohl aus der theoretischen Analyse heraus (Abschnitt 5.3) als auch in der experimentellen Validierung (Abschnitt 7.4) eine bessere Generalisierungsfähigkeit der gelernten GMMs. Overfitting und die damit einhergehende Degenerierung einzelner Komponenten können wirksam vermieden werden, wobei die Anpassung an bevorzugt auftretende Werte, z.B. bei semikontinuierlichen Verteilungen, weiterhin möglich bleibt (vergleiche Abbildung 5.17).

Aber nicht nur die Generalisierung, sondern auch die Exploration des Parameterraums werden durch die randomisierte Vorgehensweise gefördert. So erreicht REM auf den meisten Benchmarks sogar auf den Trainingsdaten eine größere Likelihood als der EM-Ansatz, obwohl dieser speziell auf die Maximierung der Likelihood ausgelegt ist, während die Parametergenerierung von REM vollkommen unabhängig von der Daten-Likelihood ist.

Ein weiterer Aspekt ist die Größenbestimmung des GMM. Durch die Integration von Parameterschätzung und Größenbestimmung bei REM und Data Augmentation ergeben sich nicht nur Vorteile bei der Rechenzeit, sondern auch bei der Qualität der Ergebnisse. So stellt das verbleibende GMM nach Löschen einer Komponente bereits ein sehr gut initialisiertes Modell für den weiteren Lernverlauf dar. Im Gegensatz dazu wird beim EM-Ansatz für jede Modellgröße das GMM neu initialisiert, so dass die geschilderten Synergie-Effekte nicht auftreten können.

Das Rechenzeitverhalten von REM wurde in Abschnitt 7.8 betrachtet. Für beide

Initialisierungsvarianten ergeben sich akzeptable Rechenzeiten, für die kleine Initialisierung sogar sehr gute. Der Vergleich mit dem EM-Ansatz ist nur bedingt möglich: zwar ist ein einzelner Lauf des EM-Ansatzes bei gleicher Anzahl Iterationen und gleicher Modellgröße schneller, allerdings muss die Gesamtaufgabe, nämlich Parameter- *und* Größenbestimmung betrachtet werden. Während REM für beide Aufgaben lediglich einen einzigen Lauf benötigt, erfordert die Größenbestimmung mit dem EM-Ansatz mehrere Läufe mit unterschiedlich großen GMMs, so dass der Rechenzeitvorteil eines einzelnen Laufs nicht ins Gewicht fällt. REM und Data Augmentation sind also auch im Hinblick auf die benötigte Rechenzeit konkurrenzfähig.

8.2 REM, Data Augmentation und SAEM

Beim Vergleich der randomisierten Algorithmen untereinander spielt vor allem das Ausmaß der Randomisierung und die Stelle, an der die Randomisierung ansetzt, eine Rolle. Die Zuordnung der Daten zu den Komponenten wird beim SAEM-Ansatz und bei Data Augmentation randomisiert, die Parametererzeugung bei Data Augmentation und REM. Durch eine zusätzlich mögliche Feineinstellung der Samplingverteilung lässt sich zudem bei REM der Grad der Zufälligkeit steuern, ebenso bei SAEM.

Die theoretische Analyse (vergleiche Abbildung 6.2) verdeutlicht den unterschiedlichen Grad der Randomisierung: ausgehend vom deterministischen EM-Algorithmus nimmt der stochastische Anteil bei SAEM über REM bis Data Augmentation zu. Zusätzlich kann der stochastische Anteil bei REM durch andere Einstellung des Parameters f reduziert werden (siehe Abbildung 6.3).

Der experimentelle Vergleich der randomisierten Verfahren (siehe Abschnitt 7.3) weist etwas bessere Ergebnisse für REM gegenüber Data Augmentation auf, wobei die Ergebnisse auf einigen Datensätzen abweichen. Eine zusätzliche Verringerung des stochastischen Einfluss bringt dagegen keine weitere Verbesserung.

Beim experimentellen Vergleich fällt ferner auf, dass die Randomisierung der Muster-Zuordnungen zu den Komponenten, wie sie bei SAEM und Data Augmentation stattfindet, einen geringeren Effekt besitzt als die Randomisierung der Parametergenerierung. So unterscheiden sich auch im experimentellen Vergleich Data Augmentation und REM sehr viel deutlicher vom EM-Ansatz und SAEM als Data Augmentation von REM bzw. SAEM vom EM-Algorithmus. Dieser Befund wird auch durch die Diskussion zur Generalisierungsfähigkeit (vergleiche Abschnitt 5.3) gedeckt: nur mit einer randomisierten Parameter-Erzeugung wird Overfitting vorgebeugt und damit die Generalisierungsleistung verbessert, nicht jedoch mit einer randomisierten Musterzuordnung.

8.3 Komitee-Effekt

Der Komitee-Effekt konnte experimentell sowohl für REM als auch für den EM-Algorithmus nachgewiesen werden (siehe Abschnitt 7.6).

Die Größe des Komitee-Effektes hängt beim EM-Verfahren ganz entscheidend davon ab, welches Modellselektionskriterium verwendet wird und ob die Modellgrößen der einzelnen Komiteemitglieder variieren dürfen oder nicht. So verhindert das BIC einen merklichen Komitee-Effekt, da die Auswahl verhältnismäßig kleiner GMMs die Vielfalt der Komitee-

mitglieder reduziert. Für AIC, Evidenz und Kreuzvalidierung liegt der Komitee-Effekt dagegen in einer ähnlichen Größenordnung wie für REM.

Die Verbesserungen durch die Hinzunahme weiterer Komiteemitglieder nehmen mit zunehmender Komiteegröße merklich ab. Ab etwa 10 Mitgliedern sind die Verbesserungen nur mehr marginal und der zusätzliche Lernaufwand nur bei extrem kritischen Anwendungen zu rechtfertigen.

8.4 Modellgröße

Die experimentellen Ergebnisse zur maximal möglichen Modellgröße (siehe Abschnitt 7.7) wurden nur für REM durchgeführt.

Die Ergebnisse zeigen mit zunehmender Dimensionalität der Daten eine bessere Ausnutzung der Information der Trainingsdaten, da der Quotient aus der Größe der Trainingsmenge und der Anzahl Modellparameter fällt. Ein exponentielles Anwachsen dieser Größe, wie aus dem *Curse of dimensionality* zu erwarten gewesen wäre, konnte in keiner Hinsicht beobachtet werden. Das bedeutet, dass GMMs als Modellklasse und REM als Lernverfahren auch für höherdimensionale Daten gut geeignet sind.

Ein weiterer Effekt ist die Abhängigkeit der Modellgröße von der Größe der Trainingsmenge. Wie die Ergebnisse zeigen ist dieser Zusammenhang sublinear, d.h. eine doppelt so große Trainingsmenge erlaubt nicht das Lernen doppelt so großer Modelle, selbst wenn die zusätzlichen Muster in eigenen, wohl abgegrenzten Clustern liegen. Eine gegenseitige Beeinflussung der Muster aus verschiedenen Clustern lässt sich also bei GMMs im Gegensatz zu Klassenmodellen nicht ganz vermeiden.

Kapitel 9

Varianten und Erweiterungen

9.1 GMMs mit radialsymmetrischen Kovarianzmatrizen

An dieser Stelle sollen mögliche Erweiterungen und Anwendungen der in den Kapiteln 4–6 entwickelten Verfahren *Data Augmentation* und *randomisiert EM* exemplarisch vorgestellt werden. Damit sollen die Potentiale des Samplingansatzes für GMMs verdeutlicht und mögliche zukünftige Entwicklungslinien der Verfahren skizziert werden. Eine ausführliche Ausarbeitung der vorgestellten Varianten und Erweiterungen bleibt dagegen zukünftigen Arbeiten vorbehalten. Als erstes soll der Fall von GMMs mit radialsymmetrischen Kovarianzmatrizen diskutiert werden.

In manchen Fällen ist es üblich, die Freiheitsgrade einer Normalverteilung einzuschränken, um damit Degenerierungen zu vermeiden oder die Berechnung der Verteilung zu vereinfachen. Eine einfache und relativ weit verbreitete Möglichkeit ist die Beschränkung auf radialsymmetrische Kovarianzmatrizen.

Eine Kovarianzmatrix Σ heißt radialsymmetrisch, wenn sie die Form $\Sigma = \sigma^2 I_d$ besitzt für geeignetes $\sigma^2 > 0$. Die Varianzen dieser Normalverteilungen sind also in allen Richtungen gleich σ^2, die Kovarianzen sind 0. Dadurch ergibt sich eine radialsymmetrische Dichte um den Erwartungswert μ. Degenerierungserscheinungen, die darauf beruhen, dass die Varianz in einer Richtung sehr klein und in einer anderen sehr groß wird, sind somit ausgeschlossen.

Aus radialsymmetrischen Normalverteilungen lassen sich wiederum Mischverteilungen aufbauen. Wenn μ_j den Erwartungswert der j-ten Komponente, $\sigma_j^2 > 0$ die Varianz der j-ten Komponente und w_j ein Mischgewicht bezeichnen, dann ist die Dichte des GMM mit k radialsymmetrischen Komponenten gegeben durch:

$$p_{GMMradial}(x) := \sum_{j=1}^{k} \frac{w_j}{\sqrt{2\pi\sigma_j^2}^d} \exp\{-\frac{1}{2\sigma_j^2}(x-\mu_j)^T(x-\mu_j)\} \tag{9.1}$$

wobei für die Mischgewichte die üblichen Bedingungen gelten wie in Abschnitt 2.4 geschildert.

Analog der Vorgehensweise in Abschnitt 4.2.2 lässt sich auch für radialsymmetrische GMMs eine Bayessche Modellierung entwerfen, wobei sich für die Mischgewichte keinerlei Veränderungen gegenüber der dortigen Darstellung ergibt. Als Prioriverteilungen für die

Erwartungswerte und Varianzen können nun die konjugierten Verteilungen gewählt werden:

$$\sigma^2 \sim \Gamma^{-1}(\beta, \alpha) \tag{9.2}$$

$$\mu|\sigma^2 \sim N(m, \frac{\sigma^2}{\eta} I_d) \tag{9.3}$$

Wie sich durch Ausrechnen leicht nachvollziehen lässt, ergeben sich als Posterioriverteilungen somit:

$$\begin{aligned} \sigma^2|x_1, \ldots, x_n &\sim \Gamma^{-1}(\hat{\beta}, \hat{\alpha}) \\ \mu|\sigma^2, x_1, \ldots, x_n &\sim N(\hat{m}, \frac{\sigma^2}{\hat{\eta}} I_d) \end{aligned} \tag{9.4}$$

mit:

$$\hat{\beta} = \beta + \frac{1}{2}(n\hat{s}^2 + \frac{n\eta}{n+\eta}(\bar{x} - m)^T(\bar{x} - m)) \tag{9.5}$$

$$\hat{\alpha} = \alpha + \frac{dn}{2} \tag{9.6}$$

$$\hat{s}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^T(x_i - \bar{x}) \tag{9.7}$$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{9.8}$$

$$\hat{\eta} = n + \eta \tag{9.9}$$

$$\hat{m} = \frac{n}{n+\eta}\bar{x} + \frac{\eta}{n+\eta}m \tag{9.10}$$

Nichtinformative Prioriverteilungen ergeben sich für $\alpha \to 0$, $\beta \to 0$ und $\eta \to 0$. In diesem Fall kann m beliebig gewählt werden.

Mit Hilfe der Bayesschen Modellierung lässt sich Data Augmentation analog Abschnitt 4.2 entwickeln. Der I-step bleibt unverändert, in den P-step geht die oben angegebene Bayessche Modellierung ein. Auch randomisiert EM kann analog aus der Bayesschen Modellierung durch Einführung gradueller Zugehörigkeit aus (9.4) entwickelt werden.

9.2 Poisson-Mischverteilungen

Neben der Möglichkeit, Mischverteilungen mit normalverteilten Komponenten zu bilden, werden in der Literatur auch andere Verteilungen zur Bildung von Mischverteilungen verwendet, z. B. Poisson-Verteilungen. Daher soll an dieser Stelle exemplarisch die Modellierung von Poisson-Mischverteilungen und die Abwandlung von Data Augmentation für diese Verteilungsfamilie demonstriert werden.

Die Poissonverteilung ist eine univariate, diskrete Verteilung. Sie besitzt einen Parameter ϑ, der sowohl den Erwartungswert als auch die Streuung der Verteilung kontrolliert (vgl. Anhang B). Die Zähldichte der Poissonverteilung ist:

$$x \sim \frac{\vartheta^x}{e^\vartheta x!} \quad \text{für } x \in \mathbb{N}_0 \tag{9.11}$$

Poisson-Mischungen bestehen analog zu GMMs aus k Poisson-verteilten Komponenten $\pi(\vartheta_j)$, die mit Mischgewichten w_j zu einer Linearkombination verknüpft werden. Die Mischgewichte müssen wiederum nicht-negativ sein und sich zu 1 aufsummieren. Die Zähldichte einer Poisson-Mischung mit k Komponenten hat somit folgende Form:

$$x \sim \sum_{j=1}^{k} \frac{w_j (\vartheta_j)^x}{e^{\vartheta_j} x!} \quad \text{für } x \in \mathbb{N}_0 \tag{9.12}$$

Als konjugierte Prioriverteilung der Poissonverteilung dient die Gammaverteilung [Robert 94]. Durch Einsetzen und ausrechnen erhält man die Posterioriverteilung:

$$\begin{aligned} \vartheta &\sim \Gamma(\beta, \alpha) \\ \vartheta | x_1, \ldots, x_n &\sim \Gamma(\beta + n, \alpha + \sum_{i=1}^{n} x_i) \end{aligned} \tag{9.13}$$

Der Modalwert der Posterioriverteilung liegt bei $\frac{\alpha + \sum_{i=1}^{n} x_i - 1}{\beta + n}$, der Maximum-Likelihood-Schätzer der Poissonverteilung ist $\vartheta_{ML} = \frac{1}{n} \sum_{i=1}^{n} x_i$. Durch Vergleich von Maximum-Likelihood- und Maximum-a-posteriori-Schätzer errechnet sich eine nichtinformative Prioriverteilung zu $\alpha = 1$, $\beta \to 0$. Dagegen ergibt Jeffreys Invarianzprinzip die nichtinformative Prioriverteilung $\alpha = \frac{1}{2}$, $\beta \to 0$

Der Data Augmentation Ansatz für Poisson-Mischungen basiert auf der beschriebenen Bayesschen Modellierung. Der I-Step erfolgt wie bei GMMs durch Auswerten der Zähldichten, Berechnung gradueller Zugehörigkeiten h_{ij} der Datenpunkte zu den einzelnen Komponenten sowie durch Samplen aus der zugehörigen Multnomialverteilung. Im P-step werden die Mischgewichte aus einer Dirichletverteilung gezogen, genauso wie für GMMs in Abschnitt 4.2 geschildert. Die Parameter ϑ_j der einzelnen Komponenten stammen aus der o.g. Posterioriverteilung, sind also Gamma-verteilt.

Die Verallgemeinerung des P-steps zum randomisierten M-step, d.h. die Berücksichtigung gradueller Zugehörigkeiten, erfolgt in gleicher Weise wie für GMMs, indem die einzelnen Trainingsmuster anteilig in die Berechnung der Posterioriverteilung eingehen.

Neben der Normal- und Poissonverteilung lassen sich auch andere Verteilungsfamilien zu Mischverteilungen ausbauen. Die Vorgehensweise erfolgt jeweils analog der hier beschriebenen Systematik. Es ist ferner möglich, Mischverteilungen auch für vektorwertige Daten zu bilden, deren Attribute teilweise kontinuierliche und teilweise diskrete Struktur aufweisen. Hierzu können geeignete univariate Verteilungen miteinander kombiniert werden, siehe auch [Schafer 97].

9.3 Data Augmentation für lückenhafte Daten

In vielen Anwendungen tritt das Problem unvollständig erfasster Daten auf, d.h. vektorwertige Trainingsdaten sind lückenhaft, einige Attribute sind nicht bekannt. Unter bestimmten Umständen ist jedoch eine Analyse derartiger Daten unumgänglich, z.B. weil es kaum vollständige Datensätze gibt oder weil ein Ignorieren der lückenhaften Muster zu einer Verzerrung der Ergebnisse führen würde. Ein Überblick über diese Thematik findet sich in [Schafer 97] sowie [Ghahramani 94].

Grundsätzlich sind drei Fälle zu unterscheiden:

CMAR Das Fehlen einzelner Werte ist rein zufällig und stochastisch unabhängig von den fehlenden Werten selbst als auch von den vorhandenen Werten. Dieser Fall wird als *completely missing at random* (CMAR) bezeichnet und ist insofern unkritisch, als ein Ignorieren der lückenhaften Muster nicht zu einer Verzerrung der Ergebnisse führen würde.

MAR Das Fehlen einzelner Werte ist stochastisch unabhängig von den fehlenden Werten selbst, nicht jedoch von den vorhandenen Werten. Dieser Fall wird als *Missing at random* (MAR) bezeichnet. Ein Ignorieren der lückenhaften Muster führt zu einer Verzerrung der Ergebnisse, allerdings lässt sich durch Analyse des Fehlens einzelner Werte diese Verzerrung herausrechnen, d.h. die Trainingsdaten enthalten genug Information, um das ursprüngliche erzeugende Modell der Daten rekonstruieren zu können.

zensiert Alle anderen Fälle werden als *zensierte Daten* bezeichnet, da das Fehlen einzelner Werte von den fehlenden Werten selbst abhängt. Eine Rekonstruktion des Datenerzeugenden Modells ist nur dann möglich, wenn die Zensurbedingung bekannt ist.

An dieser Stelle soll gezeigt werden, wie man Data Augmentation für GMMs mit allgemeinen Kovarianzmatrizen für den Fall lückenhafter Muster erweitern kann, sofern die lückenhaften Trainingsdaten die MAR- oder CMAR-Bedingung erfüllen.

Das grundlegende Konstruktionsprinzip von Data Augmentation ist die Idee, im I-step Lücken in den Trainingsdaten auszufüllen und im anschließenden P-step neue Parameter zu samplen (vgl. Abschnitt 4.2.1). Im Fall von GMMs bestanden die Lücken aus den Zuordnungen der Daten zu den einzelnen Komponenten. Weisen die Trainingsdaten darüber hinaus selbst fehlende Einträge auf, so muss der I-step so erweitert werden, dass sowohl die Lücken in den Trainingsdaten als auch die Zuordnungen gesamplet werden. Der P-step dagegen kann unverändert übernommen werden.

Betrachte nun ein einzelnes, lückenhaftes Trainingsmuster x_i. O.B.d.A. zerfalle es in einen bekannten Anteil u_i mit d' Einträgen sowie einen unbekannten, fehlenden Anteil v_i mit $d - d'$ Einträgen:

$$x_i = \begin{pmatrix} u_i \\ v_i \end{pmatrix} \tag{9.14}$$

Der Wert von v_i ist nicht bekannt. Die Aufteilung in bekannten und fehlenden Anteil mögen für jedes Trainingsmuster individuell sein. Analog zum Muster x_i lassen sich auch die Erwartungswerte und Kovarianzmatrizen der einzelnen GMM-Komponenten wiefolgt aufteilen:

$$\mu_j = \begin{pmatrix} \mu_j^u \\ \mu_j^v \end{pmatrix} \tag{9.15}$$

$$\Sigma_j = \begin{pmatrix} \Sigma_j^{uu} & (\Sigma_j^{uv})^T \\ \Sigma_j^{uv} & \Sigma_j^{vv} \end{pmatrix} \tag{9.16}$$

mit $\mu_j^u \in \mathbb{R}^{d'}$, $\mu_j^v \in \mathbb{R}^{d-d'}$, $\Sigma_j^{uu} \in \mathbb{R}^{d' \times d'}$ symmetrisch, positiv definit, $\Sigma_j^{uv} \in \mathbb{R}^{(d-d') \times d'}$ und $\Sigma_j^{vv} \in \mathbb{R}^{(d-d') \times (d-d')}$ symmetrisch, positiv definit. Man beachte, dass der obere Index u und v hier nicht als Potenzierung zu verstehen ist, sondern lediglich der Kennzeichnung der verschiedenen Anteile des Erwartungswertvektors bzw. der Kovarianzmatrix dient.

Im Imputation-step muss nun, gegeben ein GMM ϑ_{GMM} für die vollständigen Daten, der Wert von v_i und z_i gesamplet werden. Betrachte hierzu die Verteilung:

$$\begin{aligned}\mathcal{P}(v_i, z_i|u_i, \vartheta_{GMM}) &= \mathcal{P}(v_i|z_i, u_i, \vartheta_{GMM}) \cdot \mathcal{P}(z_i|u_i, \vartheta_{GMM}) \\ &\propto \mathcal{P}(v_i|z_i, u_i, \vartheta_{GMM}) \cdot \mathcal{P}(u_i|z_i, \vartheta_{GMM}) \cdot \mathcal{P}(z_i|\vartheta_{GMM})\end{aligned} \tag{9.17}$$

In (9.17) zerfällt die Wahrscheinlichkeit $\mathcal{P}(v_i, z_i|u_i, \vartheta_{GMM})$ in drei Teile, die sich berechnen lassen:

- der a-priori-Wahrscheinlichkeit einer Komponente $\mathcal{P}(z_i|\vartheta_{GMM})$
- der Randverteilung des bekannten Anteils des Datenvektors u_i: $\mathcal{P}(u_i|z_i, \vartheta_{GMM})$
- der bedingten Verteilung des fehlenden Anteils des Datenvektors gegeben der bekannte Anteil $\mathcal{P}(v_i|z_i, u_i, \vartheta_{GMM})$

Die Formeln zur Berechnung der bedingten Verteilung und der Randverteilung einer Normalverteilung können [Muirhead 82] entnommen werden. Es gilt:

wenn

$$\begin{pmatrix} u \\ v \end{pmatrix} \sim N(\begin{pmatrix} \mu^u \\ \mu^v \end{pmatrix}, \begin{pmatrix} \Sigma^{uu} & (\Sigma^{uv})^T \\ \Sigma^{uv} & \Sigma^{vv} \end{pmatrix})$$

dann

$$u \sim N(\mu^u, \Sigma^{uu}) \tag{9.18}$$

$$v|u \sim N(\mu^v + \Sigma^{uv}(\Sigma^{uu})^{-1}(u - \mu^u), \Sigma^{vv} - \Sigma^{uv}(\Sigma^{uu})^{-1}(\Sigma^{uv})^T) \tag{9.19}$$

Die Implementierung des I-steps erfolgt in zwei Schritten. Zunächst wird die Zuordnung z_i des Musters x_i zu einer Komponente bestimmt und anschließend der fehlende Anteil v_i des Musters ergänzt. Der erste Schritt erfordert ein Samplen aus der Multinomialverteilung gegeben durch den Term $\mathcal{P}(u_i|z_i, \vartheta_{GMM}) \cdot \mathcal{P}(z_i|\vartheta_{GMM})$. Die einzelnen Wahrscheinlichkeiten berechnen sich zu:

$$\mathcal{P}(z_i = j|u_i, \vartheta_{GMM}) = \frac{w_j \cdot \varphi_{\mu_j^u, \Sigma_j^{uu}}(u_i)}{\sum_{l=1}^{k} w_l \cdot \varphi_{\mu_l^u, \Sigma_l^{uu}}(u_i)} \tag{9.20}$$

Im zweiten Schritt wird v_i aus der bedingten Verteilung gesamplet, die sich gemäß (9.19) aus der Normalverteilung $N(\mu_{z_i}, \Sigma_{z_i})$ und dem Wert u_i ergibt.

Diese Vorgehensweise wird für jedes Trainingsmuster x_i wiederholt, so dass am Ende des I-steps alle Trainingsdaten vervollständigt sind, indem jedes Muster einer Komponenten zugeordnet und die Lücken innerhalb der Trainingsmuster ausgefüllt wurden. Damit können im P-step Modellparameter für das komplette-Daten-Modell erzeugt werden. Abbildung 9.1 zeigt die Arbeitsweise von Data Augmentation für lückenhafte Daten in schematischer Weise.

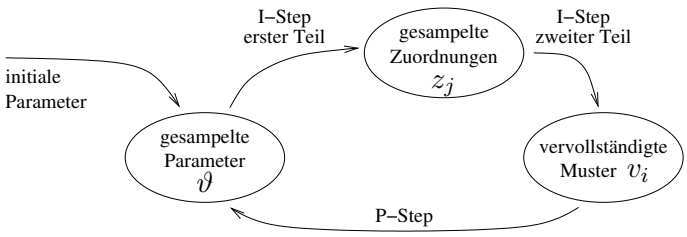

Abbildung 9.1: Schematische Darstellung der Arbeitsweise des Data Augmentation Algorithmus' für Mischverteilungen mit lückenhaften Daten.

9.4 Überwachtes Lernen mit GMMs

9.4.1 RBF-Netze und GMMs

GMMs werden in erster Linie als Dichteschätzer verwendet. Dennoch gibt es eine enge Verbindung zu Methoden des überwachten Lernens, insbesondere zu den sogenannten RBF-Netzen[1].

Bei RBF-Netzen handelt es sich um parametrisierte Funktionsapproximatoren, die aus einer Reihe von RBF-Neuronen sowie einem Ausgabeneuron bestehen. In den RBF-Neuronen wird aus dem Abstand der Eingabevektoren und einem internen Referenzvektor eine Aktivierung berechnet. Das Ausgabeneuron kombiniert die Aktivierungen der verschiedenen RBF-Neuronen linear und berechnet damit die Ausgabe des RBF-Netzes. Abbildung 9.2 zeigt die grundlegende Struktur dieses Modells. Eine ausführliche Beschreibung von RBF-Netzen würde den Rahmen dieser Arbeit sprengen, daher soll an dieser Stelle auf die Literatur verwiesen werden, z.B. [Hertz 91].

Beim überwachten Lernen soll ein Zusammenhang zwischen einem Eingabemuster e und einem Zielwert t berechnet werden. Je nach Art des Zielwertes unterscheidet man Regressionsaufgaben (reellwertiger Zielwert) sowie Klassifikation (Zielwert aus einer endlichen, meistens kleinen Menge). Als Eingabemuster werden Vektoren reeller Zahlen verwendet, d.h. $e \in \mathbb{R}^d$ mit geeignetem d.

Um GMMs für überwachte Aufgabenstellung einsetzen zu können wird die Menge der Paare aus Eingabe- und Zielwerten betrachtet, d.h. die Menge $\mathcal{X} := \{(e_1, t_1), \ldots, (e_n, t_n)\}$ für gegebene Trainingsdaten (e_i, t_i). Die Dichte dieser $d+1$-variaten Punktwolke wird mit einem GMM geschätzt. Um anschließend für unbekannte Eingaben e eine Prognose zu stellen, kann nun der bedingte Erwartungswert der Zielwerte gegeben das Eingabemuster e berechnet und dieser als Vorhersagewert verwendet werden.

Ähnlich wie in (9.17) kann hier die bedingte Verteilung der Zielwerte gegeben die Ein-

[1] RBF: *radial basis function*

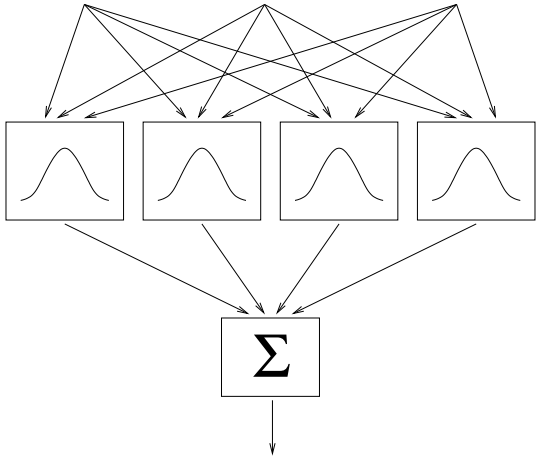

Abbildung 9.2: Schematische Darstellung eines RBF-Netzes mit vier RBF-Neuronen. Die Informationsverarbeitung erfolgt von oben nach unten. In der mittleren Schicht sind die RBF-Neuronen angeordnet, in der unteren Schicht das Ausgabeneuron.

gabewerte des GMMs berechnet werden. Es gilt:

$$\begin{aligned}\mathcal{P}(t|e,\vartheta_{GMM}) &= \sum_{j=1}^{k}(\mathcal{P}(t|z=j,e,\vartheta_{GMM})\cdot\mathcal{P}(z=j|e,\vartheta_{GMM}))\\ &= \frac{\sum_{j=1}^{k}(\mathcal{P}(t|z=j,e,\vartheta_{GMM})\cdot\mathcal{P}(e|z=j,\vartheta_{GMM})\cdot\mathcal{P}(z=j|\vartheta_{GMM})}{\mathcal{P}(e|\vartheta_{GMM})}\\ &= \frac{\sum_{j=1}^{k}(\mathcal{P}(t|z=j,e,\vartheta_{GMM})\cdot\mathcal{P}(e|z=j,\vartheta_{GMM})\cdot\mathcal{P}(z=j|\vartheta_{GMM}))}{\sum_{j=1}^{k}(\mathcal{P}(e|z=j,\vartheta_{GMM})\cdot\mathcal{P}(z=j|\vartheta_{GMM}))}\end{aligned} \tag{9.21}$$

Bezeichnet man mit ρ_j die Größe $\mathcal{P}(e|z=j,\vartheta_{GMM})\cdot\mathcal{P}(z=j|\vartheta_{GMM})$, so ergibt sich der bedingte Erwartungswert dieser Verteilung zu:

$$\begin{aligned}E[t|e,\vartheta_{GMM}] &= \int_{-\infty}^{\infty} t\cdot\mathcal{P}(t|e,\vartheta_{GMM})\,dt\\ &= \frac{\sum_{j=1}^{k}(\rho_j\int_{-\infty}^{\infty} t\cdot\mathcal{P}(t|z=j,e,\vartheta_{GMM})\,dt)}{\sum_{j=1}^{k}\rho_j}\end{aligned} \tag{9.22}$$

Lässt sich der Erwartungswertvektor der j-ten Komponenten aufspalten in $\mu_j = \begin{pmatrix}\mu^e\\ \mu^t\end{pmatrix}$, so vereinfacht sich der Integralausdruck in (9.22) unter Zuhilfenahme von (9.19) für radialsymmetrische GMMs zu μ^t. Für GMMs mit allgemeinen Kovarianzmatrizen erhält man mit Hilfe von (9.19) einen etwas komplizierteren Ausdruck.

Für radialsymmetrische GMMs stellt Ausdruck (9.22) eine Funktion der Form dar, wie sie von RBF-Netzen realisiert wird [Moody 89]. Die Aktivierungen der RBF-Neuronen werden durch die Dichte der normalverteilten Komponenten simuliert, die Gewichte, mit denen die einzelnen RBF-Neuronen zusammengeführt werden, durch die Werte μ_j^t. Dies verdeutlicht den engen Zusammenhang zwischen GMMs einerseits und RBF-Netzen andererseits.

9.4.2 Regressionsaufgaben

Die Einsatzmöglichkeiten von GMMs für Regressionsaufgaben wurden im Wesentlichen bereits in Abschnitt 9.4.1 entwickelt. Die Kernidee besteht darin, ein GMM auf den Paaren von Eingabe- und Zielwerten zu lernen und zur Vorhersage die bedingte Verteilung der Zielwerte gegeben die Eingabewerte zu betrachten. Daher soll die Arbeitsweise hier nur noch einmal schematisch dargestellt werden.

Training Trainiere ein GMM auf dem gemeinsamen Raum der Paare von Eingabe- und Zielwerten. Verwendet werden kann ein beliebiges Lernverfahren für GMMs. Es können radialsymmetrische Komponenten oder auch Komponenten mit allgemeinen Kovarianzmatrizen verwendet werden.

Vorhersage Gegeben ein unbekanntes Eingabemuster e berechne den bedingten Erwartungswert der Zielwerte gegeben e gemäß (9.22).

9.4.3 Klassifikationsaufgaben

Für Klassifikationsaufgaben kann prinzipiell die selbe Idee verwendet werden wie für Regressionsaufgaben, nämlich ein Training auf dem gemeinsamen Raum von Eingaben und Zielwerten. Allerdings ist zu beachten, dass die Klassenzugehörigkeiten diskret sind und daher ein GMM eher ungeeignet ist, um die Verteilung der Zielwerte zu beschreiben. Degenerierte GMMs wären unweigerlich die Folge und die Interpretation des gelernten Modells wäre sehr problematisch.

Es bietet sich daher an, das Modell der Struktur der Zielwerte anzupassen, indem die Zielwerte nicht durch eine Normalverteilung sondern durch eine Multinomialverteilung beschrieben werden. Das bedeutet, man beschreibt die Klassenzugehörigkeiten durch explizite Wahrscheinlichkeiten und modelliert die Verteilungen der einzelnen Klassen als GMMs. Damit ergibt sich folgende Vorgehensweise:

Training Teile die Trainingsdaten gemäß ihrer Klassenzugehörigkeiten auf. Trainiere für jede Klasse ein eigenes GMM. Berechne außerdem die Klassenhäufigkeiten aus den Trainingsdaten.

Modell Das gesamte Modell besteht aus je einem GMM für jede Klasse, hier bezeichnet durch seine Dichte $p_j(\cdot)$ sowie aus einer relativen Klassenhäufigkeit π_j für jede Klasse, wobei sich die Klassenhäufigkeiten zu 1 addieren.

Prognose Gegeben ein unbekanntes Eingabemuster e, führe eine Bayessche Klassifikation durch. Berechne dafür für jede Klasse j die Größe $\pi_j \cdot p_j(e)$. Ordne das Muster derjenigen Klasse j zu, für die $\pi_j \cdot p_j(e)$ maximal ist.

Es kann gezeigt werden [Duda 73], dass diese Art der Klassifikation den Fehler durch Missklassifikation minimiert, sofern die Verteilungen $p_j(\cdot)$ und die Klassenwahrscheinlichkeiten π_j die wahren Verteilungen bzw. Wahrscheinlichkeiten sind.

Sowohl Regression als auch Klassifikation mit GMMs stehen in Konkurrenz zu explizit auf überwachtes Lernen hin ausgelegte Algorithmen, z.B. Multilayer-Perzeptrons, Support-Vector-Machines, Entscheidungsbäume und andere. Im Gegensatz zu diesen Verfahren, die

nur eine Ein-/Ausgabefunktion lernen, lernen GMMs zusätzlich die Verteilung der Eingabemuster.

Dieser Unterschied hat Vor- und Nachteile. Zum einen ermöglicht die Kenntnis der Eingabeverteilung auch eine Auswertung des Modells in umgekehrter Richtung, d.h. es ist möglich, zu gegebenem Zielwert die Verteilung der möglichen Eingabewerte zu berechnen. Ferner ist es möglich, lückenhafte Trainingsmuster zu verarbeiten, solange diese die MAR-Bedingung erfüllen.

Andererseits ist das Lernen der Eingabeverteilung ein zusätzlicher Aufwand, der häufig nicht notwendig ist. Zudem können daraus fehlerhafte Modelle entstehen, die sich negativ auf die Prognoseleistung niederschlagen. Beispielsweise stellen stark diskretisierte Eingabemuster ein Problem dar, selbst wenn die Zielwerte kontinuierlich verteilt sind. Dieses Problem tritt bei alternativen Verfahren nicht oder nicht in dem Maße auf wie bei GMMs.

9.5 Ausreißerdetektion

Viele Datensätze enthalten Datenpunkte, die ungewöhnlich oder fragwürdig sind. Häufige Ursache ist eine fehlerhafte Datenerfassung oder Datenaufbereitung. Diese Ausreißer können spätere Datenanalysen verfälschen oder gar vollkommen unbrauchbar machen [Barnett 78, Rousseeuw 87]. GMMs können erfolgreich eingesetzt werden, um solche Datenpunkte zu finden und damit fehlerhafte Datenanalysen zu vermeiden. Der hier vorzustellende Ansatz wurde in [Lauer 01a] entwickelt.

Der Kerngedanke des Ansatzes besteht darin, die Dichte der gegebenen Daten zu bestimmen und solche Daten als Ausreißer zu klassifizieren, deren Dichte gering ist. Dabei ergibt sich jedoch die Schwierigkeit, dass die Trainingsdaten selbst Ausreißer enthalten können und diese die Schätzung der Dichte beeinflussen. Das bedeutet, dass an den Stellen der Ausreißer in den Trainingsdaten eine größere Dichte geschätzt wird als tatsächlich dem Daten-generierenden Prozess entspricht. Stünde eine Ausreißer-freie Trainingsmenge zur Verfügung, entfiele dieses Problem.

Das Problem der Ausreißerdetektion stellt sich als schwierig heraus, da zwei Aspekte sich gegenseitig bedingen:

- wäre ein korrektes Modell des datengenerierenden Prozess' bekannt, so könnte man mit dessen Hilfe die Ausreißer erkennen
- wären die Ausreißer in den Trainingsdaten bekannt, so könnte man sie entfernen und aus der verbleibenden Menge korrekter Datenpunkte ein Modell des datengenerierenden Prozess' schätzen.

Da jedoch weder klassifizierte Trainingsdaten noch ein Modell des datengenerierenden Prozess' bekannt sind, müssen beide Teilaufgaben gemeinsam gelöst werden.

Zur Modellierung der Ausreißer sollen verschiedene Annahmen getroffen werden:

(i) Ausreißer können in allen Bereichen des Datenraums auftreten. Sie bevorzugen keinen Teilbereich.

(ii) Der Anteil Ausreißer in den Trainingsdaten ist bekannt oder kann abgeschätzt werden. Dieser Anteil werde mit λ ($\lambda \in [0, 1]$) bezeichnet.

(iii) Die anderen Daten stammen aus einer Verteilung mit Dichte $p(\cdot)$.

Da nach Annahme (i) Ausreißer keinen Bereich des Datenraums bevorzugen, kann die Verteilung der Ausreißer p_A als eine ganz flache, breite Verteilung angenommen werden, z.B. eine Normalverteilung mit sehr großen Varianzen oder eine Gleichverteilung über den gesamten Datenraum[2].

Sind p, λ und p_A bekannt, kann die Verteilung der Trainingsdaten, so wie sie tatsächlich vorliegen, beschrieben werden. Sie setzt sich zusammen als Mischung der Ausreißer und der Originaldaten, jeweils mit λ und $1 - \lambda$ gewichtet:

$$x \sim (1 - \lambda) \cdot p(x) + \lambda \cdot p_A(x) \tag{9.23}$$

Die Verteilung der vorliegenden Daten nimmt also die Form einer Mischverteilung mit zwei Komponenten an. Die Verteilung p kann selbst wiederum ein GMM sein.

Außerdem kann ein gegebenes Muster x danach klassifiziert werden ob es Ausreißer ist oder nicht. Im Sinne einer Bayesschen Klassifikation werden die Posterioriwahrscheinlichkeiten der beiden Klassen *Ausreißer* und *kein Ausreißer* miteinander verglichen, so dass sich folgende Klassifikationsregel ergibt:

$$\begin{gathered} x \text{ wird als Ausreißer klassifiziert} \\ \text{genau dann wenn} \\ (1 - \lambda) \cdot p(x) < \lambda \cdot p_A(x) \end{gathered} \tag{9.24}$$

Ist λ und p_A bekannt oder kann es approximiert werden, so kann auch p aus einer Ausreißer-behafteten Trainingsmenge geschätzt werden, da es die einzige Unbekannte auf der rechten Seite von (9.23) ist. Da die Verteilung (9.23) eine Mischverteilung ist, können zu ihrer Schätzung der EM-Ansatz oder Data Augmentation verwendet werden. Der einzige Unterschied zur allgemeinen Vorgehensweise für GMMs (siehe Kapitel 4) besteht darin, dass eine der Komponenten, p_A, keine freien Parameter besitzt und dass auch das Mischgewicht dieser Komponenten fest vorgegeben ist.

Daher reduziert sich der M- bzw. P-step auf die Bestimmung neuer Parameter für die erste Komponente, p, während die Mischgewichte und die zweite Komponente unverändert bleiben. p kann selbst wiederum eine Mischverteilung sein, z.B. ein GMM. In diesem Fall findet also ein verschränktes Lernen zweier Mischverteilungen statt, nämlich der Mischverteilung p, die die Originaldaten beschreibt, sowie der Mischverteilung, die den Ausreißereinfluss einbindet.

In [Lauer 01a] werden einige empirische Beispiele gezeigt, wie dieser Ansatz zur Ausreißererkennung arbeitet, in [Lauer 01b] wird dieser Ansatz für Messdaten angewendet. In diesem Kontext zeigt sich auch, dass der Data Augmentation Algorithmus für eine derartige Anwendung besser geeignet ist als der EM-Ansatz. Der Grund liegt darin, dass die gesamte Daten-log-Likelihood durch die breite und feste Ausreißerverteilung p_A nach unten beschränkt ist. Dadurch werden die Probleme, die mit der lokalen Optimierung des EM-Verfahrens verbunden sind, verstärkt und führen zu schlechten Anpassungen. Im Gegensatz dazu vermeidet die randomisierte Suche von Data Augmentation diese Probleme. Eine detaillierte Beschreibung der Problematik findet sich in [Lauer 01a].

[2] diese Verteilung kann auch unecht sein

Kapitel 10

Zusammenfassung

10.1 Ergebnisse der Arbeit

Ausgehend von einem Monte-Carlo Samplingverfahren wurde in dieser Arbeit der REM-Algorithmus entwickelt, mit dem es möglich ist, Modellgröße und Parameter einer Gauß-Mischverteilung automatisch zu bestimmen. Dazu wurde das Verhalten des Monte-Carlo-Samplers untersucht und daraus ein neues Lernverfahren entwickelt, um optimale Modellparameter bestimmen zu können.

Sowohl experimentell als auch durch Analyse der Verfahren konnte nachgewiesen werden, dass eine randomisierte Parameterbestimmung in der Lage ist, die Beschränkungen klassischer Lernverfahren zu überwinden:

- die geschätzten Modelle besitzen eine große Anpassungsgüte an die Trainingsdaten, das randomisierte Lernen überwindet die Probleme lokaler Optima und ungenügender Exploration des Parameterraums.

- die gelernten Modelle besitzen eine gute Generalisierungfähigkeit. Überanpassung wird vermieden, da sich die Samplingverfahren nicht an der Likelihoodfunktion orientieren und somit Overfittingsituationen nicht bevorzugen. Zusätzliche Regularisierungstechniken sind nicht notwendig.

- Degenerierung der GMMs mit singulären Kovarianzmatrizen wird durch die randomisierte Parameterbestimmung ebenfalls vorgebeugt, da degenerierte Modellausprägungen zwar eine große Likelihood besitzen, nicht aber sehr wahrscheinlich sind im Bezug auf die Posterioriverteilungen.

- die randomisierte Vorgehensweise erlaubt nicht nur die Bestimmung von Modellparametern bei gegebener Modellgröße, sondern auch die Bestimmung der Modellgröße selbst ohne zusätzlichen Rechenaufwand.

- für die Einstellung der Parameter der Lernverfahren (Prioriverteilungen, Anzahl Iterationen, Initialisierung) konnten Empfehlungen entwickelt werden, mit denen gute Ergebnisse erzielt werden können.

- der Rechenaufwand der randomisierten Algorithmen ist gering.

Durch die Vermeidung von Overfitting und Degenerierungserscheinungen, die automatische Größenbestimmung und die unkritische Wahl der Parameter des Lernverfahrens wird es möglich, die Monte-Carlo-Algorithmen auch ohne ständige menschliche Überwachung einzusetzen. Die vielfältigen Möglichkeiten zur Erweiterung und Anpassung der Methodik an verschiedene Anforderungen und Aufgabenstellungen wie lückenhafte Daten, Ausreißeranalyse, Klassifikation und Regression unterstreichen die Potentiale der in dieser Arbeit entwickelten Verfahren.

Mit Hilfe der systematischen experimentellen Untersuchung konnte die Überlegenheit des REM-Verfahrens über die klassischen Methoden EM und SAEM auch empirisch unter Beweis gestellt werden. Die Experimente verdeutlichten die bereits analytisch festgestellten Vorteile der randomisierten Vorgehensweise:

- bessere Exploration des Parameterraums
- Vermeiden von Overfitting
- bessere Bestimmung der Modellgröße

Die beiden Algorithmen *Data Augmentation* und *REM* ermöglichen damit die Ablösung des problembehafteten EM-Ansatzes im Bereich der Dichteschätzung mit GMMs.

10.2 Besonderheiten des Ansatzes

Die Überwindung der Probleme klassischer Lernalgorithmen für GMMs durch *Data Augmentation* und *REM* kann auf ein grundlegend unterschiedliches Konstruktionsprinzip zurückgeführt werden.

Klassische Lernverfahren basieren auf einem Gütemaß für die Anpassung des Modells an die Trainingsdaten, z.B. der Likelihood auf den Trainingsdaten. Durch lokale Veränderung der Modellausprägungen wird versucht, dieses Gütemaß zu maximieren und damit eine beste Modellausprägung zu erhalten. Für hinreichend komplexe Modelle ist bei einer derartigen Vorgehensweise Overfitting vorprogrammiert und der Einsatz von Regularisierungstechniken unumgänglich.

Die Einführung randomisierter Modellanpassungen wie bei SAEM oder bei evolutionären Algorithmen hilft, vorzeitige Konvergenz in lokale Optima zu vermeiden; allerdings verbessert sie die Generalisierungsfähigkeit nicht, so lange mit Hilfe von Annealing Schemata oder durch Verwerfen von Modellausprägungen, die die Gütefunktion verschlechtern, eine ständige Ausrichtung des Lernverfahrens am Gütekriterium erfolgt.

Erst durch die Loslösung vom Gütekriterium, wie es der Gibbs-Sampler realisiert, kann die Gefahr des Overfitting gebannt werden. Da in die Samplingverteilung die Likelihood auf den Trainingsdaten nicht eingeht, ist Overfitting ausgeschlossen. Durch die Wahl einer geeigneten Samplingverteilung und die Nachbearbeitung der gesampleten Parameter kann dennoch eine gute Anpassung an die Trainingsdaten erreicht werden. Erst in einem nachgelagerten Selektionsschritt wird die Likelihood auf den Trainingsdaten verwendet. Eine zunehmende Überanpassung durch lokale Optimierung bleibt somit ausgeschlossen.

Die randomisierte Parameterbestimmung ermöglicht in Kombination mit der für GMMs typischen Aufteilung des Datenraums in verschiedene Bereiche, auf die sich je eine Komponente spezialisiert, die in dieser Arbeit realisierte Bestimmung der Modellkomplexität. Durch

Analyse der Dynamik der durch den Samplingalgorithmus erzeugten Markov-Kette lassen sich überzählige Komponenten entdecken und entfernen, da derartigen Komponenten nach und nach keine Datenpunkte mehr zugeordnet werden. Modelle mit einer angemessenen Komplexität zeichnen sich dadurch aus, dass jede Komponente einen eigenen Aspekt der Daten beschreibt und dieser Aspekt durch ausreichend viele Trainingsdaten begründet ist. Ein derartiges GMM besitzt also weder redundante Komponenten noch fragwürdige, nur unzureichend durch die Daten erklärte Modellausprägungen.

Das Prinzip, Parameter zu samplen anstatt sie durch lokale Optimierung zu berechnen, hat sich als sehr fruchtbar erwiesen, um Probleme klassischer Ansätze zu überwinden. Es ist daher geeignet, auch für andere Lernaufgaben als die der Dichteschätzung Ansätze zur Optimierung einer Gütefunktion erfolgreich zu ersetzen.

Anhang

Anhang A

Dokumentation der experimentellen Ergebnisse

A.1 Beschreibung der Testdatensätze A1–A6 und B1–B6

Für die in Kapitel 7 vorgestellten Experimente wurden die Benchmarkdatensätze A1–A6 und B1–B6 verwendet, die in diesem Anhang im Hinblick auf ihre Herkunft und Charakteristika beschrieben werden sollen.

Die sechs Datensätze A1–A6 sind künstlich erzeugt worden, wobei versucht wurde, die Verteilungen so zu wählen, dass ähnliche Verhältnisse auftreten wie in echten Anwendungen. Die Anzahl Attribute sowie die Größe von Trainings- und Testmengen sind in Tabelle A.1 angegeben. Für alle Verteilungen wurden je 100 verschiedene Trainings- und Testmengen erzeugt, so dass die Experimente 100 mal mit unterschiedlichen Daten aus der selben Datenquelle wiederholt werden konnten. Diese Vorgehensweise ermöglicht zuverlässigere Schlussfolgerungen als ein einzelnes Experiment auf einer einzelnen Trainings- und Testmenge.

Datensatz A1 Datensatz A1 stellt einen typischen Vertreter einer kontinuierlichen Verteilung dar. Abbildung A.1 zeigt die Testmuster in der zweidimensionalen Ebene. Die Verteilung besitzt sowohl Bereiche, in denen viele Datenpunkte kompakt angeordnet sind, als auch einzelne sehr weit gestreute Muster. Wie zu erkennen ist, bilden die Bereiche großer Dichte keinen konvexen Bereich, sondern umschließen einen Bereich geringerer Dichte Halbring-förmig. Die Modellierung mit GMMs erfordert also eine Anordnung mehrerer Komponenten im Halbrund.

Datensatz A2 Der kleiner Datensatz A2 besitzt eine diskrete Struktur. Dazu wurden zunächst aus einer kontinuierlichen Verteilung Zufallsstichproben gezogen und die Werte anschließend gerundet. Allerdings treten im Verhältnis zur Größe der Trainingsmenge verhältnismäßig viele verschiedene Werte auf, so dass sich die diskrete Natur der Daten nur wenig auswirkt. Insgesamt nehmen die vier Attribute maximal 160, 295, 168 bzw. 77 verschiedene Werte an, allerdings treten in einem einzelnen Trainings-Datensatz mit 300 Mustern davon nur ca. 90, 105, 75 bzw. 50 auf.

Nr.	Bezeichnung	Größe Trainingm.	Größe Testmenge	Dimension	Verteilungsart
A1		500	4000	2	kontinuierlich
A2		300	300	4	diskret
A3		200	1000	1	semikontinuierlich
A4		300	5000	2	kontinuierlich
A5		500	5000	8	kontinuierlich
A6		200	1000	3	kontinuierlich
B1	banana	400	4900	2	kontinuierlich
B2	breast-cancer	200	77	9	
B3	diabetes	268	300	8	
B4	thyroid	140	75	5	
B5	bupa	200	145	6	
B6	iris	100	50	4	

Tabelle A.1: Übersicht über die verwendeten Test-Datensätze.

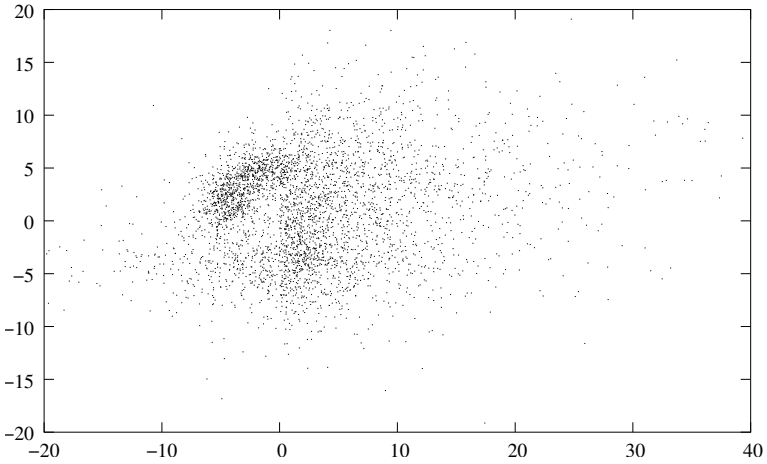

Abbildung A.1: Die Testdaten von A1 in der zweidimensionalen Ebene. Einige weit gestreute Datenpunkte liegen außerhalb des dargestellten Ausschnitts.

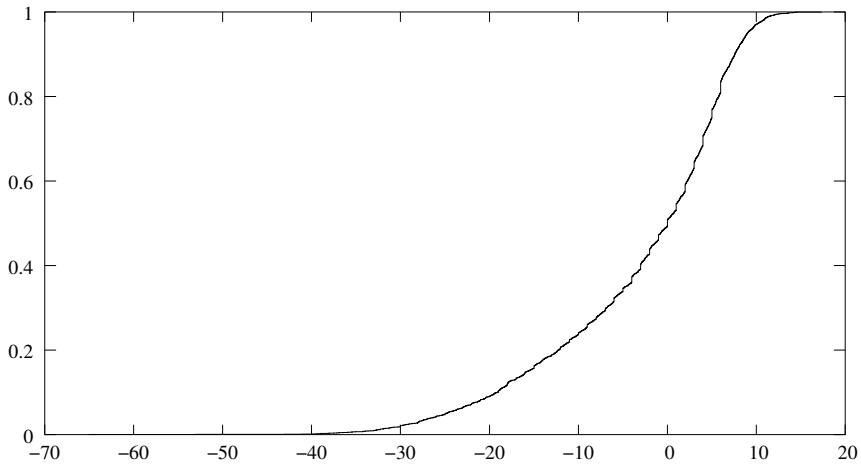

Abbildung A.2: Kummulative Verteilungsfunktion des Datensatzes A3. Die Sprungstellen in der Verteilungsfunktion sind erkennbar, aber nicht stark ausgeprägt.

Datensatz A3 Der Datensatz A3 ist aus einer univariaten semikontinuierlichen Verteilung gezogen worden. Abbildung A.2 zeigt die kummulative Verteilungsfunktion.

Zwei Besonderheiten dieses Datensatzes fallen auf: zum einen ist die Verteilung deutlich links-schief, zum anderen weist sie an den ganzzahligen Positionen bevorzugte Stellen auf, erkennbar an den Sprüngen in der kummulativen Verteilungsfunktion. Im Verhältnis zur geringen Größe der Trainingsmenge mit nur 200 Datenpunkten wirkt sich diese Bevorzugung allerdings nur wenig auf die Lernverfahren aus.

Datensatz A4 Der bivariate Datensatz A4, dargestellt in Abbildung A.3, wurde aus einem fünfkomponentigen GMM gesamplet. Neben den beiden Komponenten mit vertikaler und einer Komponente mit horizontaler Orientierung gehen eine kompakte, radialsymmtrische, am Ursprung angeordnete Komponente, sowie eine breit gestreute Normalverteilung ein. Dieser Datensatz eignet sich besonders für Untersuchungen, ob die ursprünglich vorgegebenen Komponenten auch von den Lernverfahren wiedergefunden werden.

Datensatz A5 Mit acht Attributen und 500 Datenpunkten in den Trainingsmengen stellt A5 den größten Datensatz dar. Die Daten wurden aus einer kontinuierlichen Verteilung gesamplet und sind um eine Hyperkugel herum gestreut, bilden also eine Anordnung, die einer Ringstruktur im Zweidimensionalen oder einer Hohlkugel im Dreidimensionalen entspricht.

Datensatz A6 Der Datensatz A6 wird durch eine trivariate Gleichverteilung beschrieben. Schwierigkeiten bereitet hierbei vor allem die Modellierung der Sprungstellen der Dichte.

Die sechs Benchmark-Datensätze B1–B6 dienen zum Ergebnisvergleich. Bis auf den künstlich erzeugten Datensatz B1 stammen alle Daten aus statistischen Erhebungen, so dass die Datenbasen sehr ähnlich zu Aufgabenstellungen in der Praxis sind, d.h. das Maß an Diskretisierung der Daten, die Größe der Datensätze und die Struktur der Verteilungen

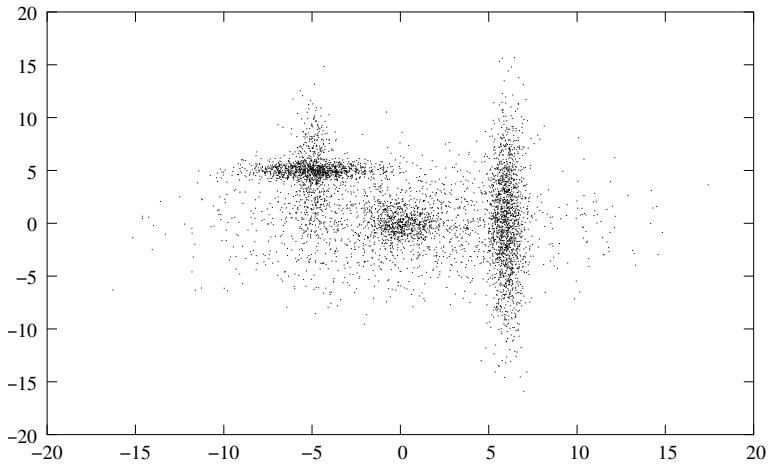

Abbildung A.3: Die Testdaten von A4 in der zweidimensionalen Ebene. Deutlich sind die fünf Komponenten zu erkennen, aus denen diese Verteilung zusammengesetz ist.

entsprechen den Anforderungen der Wirklichkeit. Die Gesamtmenge der zur Verfügung stehenden Daten wurde in Trainings- und Testdaten aufgeteilt, wobei für jeden Datensatz 100 zufällige Aufteilungen in Trainings- und Testdaten erfolgten, so dass die experimentellen Ergebnisse eine größere Sicherheit besitzen.

Datensatz B1 *„banana“* Der Datensatz „banana“ ist ein künstlich erzeugter bivariater Datensatz. Ursprünglich handelt es sich um eine Klassifikationsaufgabe, wobei die beiden Klassen Bananen-förmige, ineinander greifende Strukturen besitzen (siehe Abbildung A.4). Die Verteilung der Daten ist kontinuierlich, die Daten sind nicht diskretisiert oder gerundet. In den Untersuchungen dieser Arbeit werden die Klassenzugehörigkeiten ignoriert, d.h. die Aufgabe besteht darin, die Verteilung aller Daten unabhängig von ihrer Klassenzugehörigkeit zu bestimmen. Der Datensatz B1 sowie die Aufteilung in Trainings- und Testdaten entstammt der Benchmarksammlung [Rätsch 99].

Datensatz B2 *„breast-cancer“* Bei Datensatz B2 handelt es sich um einen Datensatz aus der Analyse von Brustkrebs. Als Attribute dienen Messwerte von Patienten, z.B. das Alter und die Größe des Tumors. Alle Werte sind nominal oder sehr stark diskretisiert. Die Anzahl verschiedener Werte, die die Attribute annehmen können, variiert zwischen 2 und 11. In dieser Arbeit wurden die Aufteilungen in Trainings- und Testmenge aus [Rätsch 99] verwendet. Unvollständig gegebene Datenvektoren wurden nicht verwendet.

Datensatz B3 *„diabetes“* Datensatz B3 entstammt einer medizinischen Untersuchung der Krankheit *Diabetes mellitus*. Die Attribute sind diskretisiert, allerdings nicht so stark wie bei Datensatz B2. Das am stärksten diskretisierte Attribut nimmt immerhin noch 16 verschiedene Werte an. Die Aufteilungen in Trainings- und Testmenge wurde wiederum [Rätsch 99] entnommen.

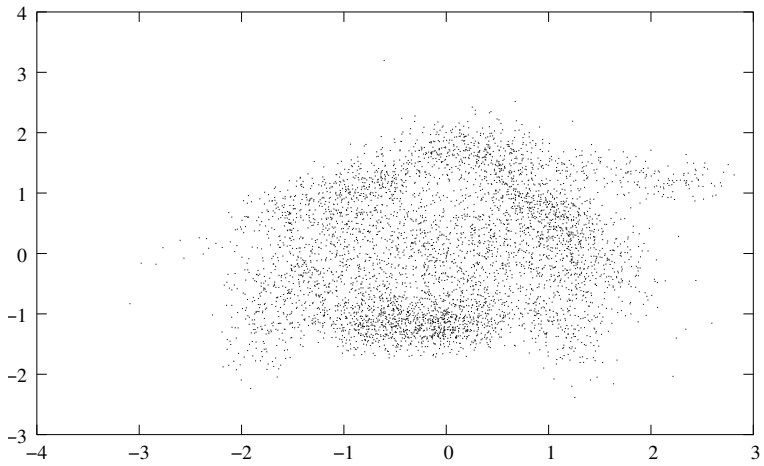

Abbildung A.4: Der Datensatz B1 in der zweidimensionalen Ebene.

Datensatz B4 *„thyroid“* Der Datensatz B4 beschreibt medizinische Zusammenhänge bei Patienten mit Schilddrüsen-Fehlfunktion. Die Attribute sind diskretisiert, nehmen jedoch mindestens 41 verschiedene Werte an, so dass der Diskretisierungseffekt sich bei 140 Trainingsdaten nur in geringem Maße bemerkbar macht. Die Aufteilungen in Trainings- und Testmenge wurde aus [Rätsch 99] übernommen.

Datensatz B5 *„bupa“* Der Datensatz B5 wurde [Blake 98] entnommen und beschreibt medizinische Kenngrößen von Patienten mit Erkrankungen der Leber. Die Attribute sind diskretisiert und nehmen zwischen 16 und 94 verschiedene Werte an.

Datensatz B6 *„iris“* Datensatz B6 beschreibt die geometrische Form von Blütenblättern von Blumen der Gattungen *Iris setosa*, *Iris versicolor* und *Iris virginica*. Ursprünglich als einen Klassifikationsaufgabe definiert, ist die Zielsetzung in dieser Arbeit, wie bei allen anderen Datensätze auch, die Bestimmung der Dichte der Daten, unabhängig von ihrer Klassenzugehörigkeit. Die Attribute sind diskretisiert, nehmen aber dennoch mindestens 22 verschiedene Werte an. Der Iris-Datensatz stammt aus [Blake 98].

Da die Zielsetzung der experimentellen Untersuchungen auf dem Vergleich verschiedener GMM-Schätzer liegt, wurden die Datensätze B1–B6 ohne weitere Datenvorverarbeitung verwendet, d.h. es hat weder eine Merkmalsauswahl, noch eine Transformation der Daten stattgefunden. Zweifellos können derartige Vorverarbeitungsschritte die Güte der gelernten Modelle beträchtlich verbessern, allerdings ist dies nicht Teil der Untersuchungen dieser Arbeit. Vielmehr stellen die unvorverarbeiteten Daten mit z.T. sehr vielen Attributen bei kleiner Trainingsmenge besonders große Anforderungen an die Leistungsfähigkeit und Robustheit der Lernverfahren. Diese Eigenschaften sind insbesondere dann von Bedeutung, wenn Lernverfahren im automatischen Betrieb auf unbekannten Daten eingesetzt werden sollen.

A.2 Dokumentation der Ergebnisse aus Kapitel 7

Nachfolgend aufgeführte Tabellen A.2–A.14 dokumentieren die in Kapitel 7 beschriebenen experimentellen Ergebnisse.

Dimensionalität	Größe Trainingsmenge	Schätzer I	II	III	IV
univariat	10	**-1.56066**	-1.56299	-1.57153	-1.57021
	20	**-1.48228**	-1.48409	-1.48728	-1.48231
	50	**-1.45066**	-1.45086	-1.45127	-1.45085
	100	-1.44089	**-1.44085**	-1.44087	-1.44110
bivariat	10	-3.29752	-3.25255	**-3.20321**	-3.48614
	20	-3.01986	-3.01189	**-3.00460**	-3.05788
	50	-2.90417	-2.90303	**-2.90201**	-2.90994
	100	-2.87025	**-2.87013**	-2.87019	-2.87121
trivariat	10	-5.38926	-5.25304	**-4.97843**	-6.06162
	20	-4.61755	-4.59554	**-4.55646**	-4.73988
	50	-4.37054	-4.36802	**-4.36432**	-4.38638
	100	-4.31362	-4.31300	**-4.31205**	-4.31760
9-variat	20	-16.8513	-16.5780	**-14.9290**	-20.0682
	50	-13.5606	-13.5350	**-13.3993**	-13.9056
	100	-13.0814	-13.0760	**-13.0501**	-13.1579

Tabelle A.2: Vergleich der mittleren Test-Log-Likelihood für verschiedene Varianzschätzer (I-IV) auf normalverteilten Daten. Die jeweils besten Schätzer sind durch Fettdruck herausgehoben.

Dimensionalität	Größe Trainingsmenge	Schätzer I	II	III	IV
univariat	20	-0.583036	-0.576572	**-0.571495**	-0.599602
trivariat	20	-1.893990	-1.867000	**-1.813000**	-2.03622

Tabelle A.3: Vergleich der mittleren Test-Log-Likelihood für verschiedene Varianzschätzer (I-IV) auf Daten, die aus einer Verteilung mit dreiecksförmiger Dichte gezogen wurden. Die jeweils besten Schätzer sind durch Fettdruck herausgehoben.

Dimensionalität	Größe Trainingsmenge	Schätzer I	II	III	IV
univariat	20	-0.223089	-0.221625	**-0.221549**	-0.229655
trivariat	20	-0.790805	-0.771827	**-0.741847**	-0.900996

Tabelle A.4: Vergleich der mittleren Test-Log-Likelihood für verschiedene Varianzschätzer (I-IV) auf gleichverteilten Daten. Die jeweils besten Schätzer sind durch Fettdruck herausgehoben.

Dimensionalität	Größe Trainingsmenge	Schätzer I	II	III	IV
univariat	10	-2.01201	-2.03949	-2.07318	**-1.97125**
	20	-1.88717	-1.89723	-1.90867	**-1.87070**
	50	-1.81184	-1.81397	-1.81632	**-1.80815**
bivariat	10	-4.36936	-4.31212	**-4.23823**	-4.59479
	20	-3.83142	**-3.82747**	-3.82821	-3.85739
	50	-3.63634	**-3.63612**	-3.63695	-3.63935
trivariat	10	-6.78080	-6.63325	**-6.32468**	-7.38202
	20	-5.84115	-5.81683	**-5.77081**	-5.97273
	50	-5.45935	-5.45688	**-5.45335**	-5.47498
9-variat	20	-21.1975	-20.8559	**-18.5927**	-25.0968
	50	-16.8379	-16.8102	**-16.6561**	-17.2033

Tabelle A.5: Vergleich der mittleren Test-Log-Likelihood für verschiedene Varianzschätzer (I-IV) auf Laplaceverteilten Daten. Die jeweils besten Schätzer sind durch Fettdruck herausgehoben.

Dimensionalität	Größe Trainingsmenge	Schätzer I	II	III	IV
univariat	20	-1.81606	-1.80461	**-1.79454**	-1.84259
bivariat	20	-3.68551	-3.65917	**-3.61512**	-3.77865
trivariat	20	-5.67842	-5.62981	**-5.51096**	-5.86386

Tabelle A.6: Vergleich der mittleren Test-Log-Likelihood für verschiedene Varianzschätzer (I-IV) auf t_5-verteilten Daten. Die jeweils besten Schätzer sind durch Fettdruck herausgehoben.

Verfahren	Test-log-Likelihood	mittlere GMM-Größe
	Datensatz A1	
Data Augmentation	-24779.1 ± 192.54	9.57 ± 2.7722
REM	-24692.6 ± 193.028	6.8 ± 1.28062
REM/kmeans	$\mathbf{-24688.7 \pm 182.684}$	7.31 ± 1.59182
REM/kmeans/f=1.5	-24826 ± 215.729	8.51 ± 1.38921
REM/kmeans/f=2	-24908 ± 250.533	9.1 ± 1.34536
	Datensatz A2	
Data Augmentation	$\mathbf{-5227.19 \pm 43.2185}$	4.71 ± 1.68104
REM	-5230.51 ± 45.7256	5.16 ± 1.06508
REM/kmeans	-5236.76 ± 50.1576	6.06 ± 1.41294
REM/kmeans/f=1.5	-5321.62 ± 64.1193	9.02 ± 0.927146
REM/kmeans/f=2	-5343.78 ± 70.7311	9.59 ± 0.679632
	Datensatz A3	
Data Augmentation	-3641.82 ± 38.1962	4.89 ± 2.23112
REM	-3635.52 ± 36.5907	3.51 ± 0.685493
REM/kmeans	$\mathbf{-3635.18 \pm 36.5539}$	3.51 ± 0.768049
REM/kmeans/f=1.5	-3644.72 ± 39.4109	5.23 ± 1.2398
REM/kmeans/f=2	-3652.69 ± 41.5289	6.2 ± 1.37113
	Datensatz A4	
Data Augmentation	$\mathbf{-25984.5 \pm 182.677}$	5.1 ± 0.640312
REM	-26027.8 ± 178.712	5.9 ± 0.932738
REM/kmeans	-26046.9 ± 178.09	5.93 ± 0.897274
REM/kmeans/f=1.5	-26315.3 ± 301.056	8.03 ± 1.01445
REM/kmeans/f=2	-26592 ± 434.755	9.12 ± 0.790949
	Datensatz A5	
Data Augmentation	-24545.5 ± 1934.33	3.82 ± 0.669029
REM	-24506.6 ± 1480	3.96 ± 0.733757
REM/kmeans	$\mathbf{-23546.5 \pm 1211.17}$	5.02 ± 0.87155
REM/kmeans/f=1.5	-24514.8 ± 1527.93	4.99 ± 0.877439
REM/kmeans/f=2	-25182.3 ± 2098.29	5.77 ± 1.18199
	Datensatz A6	
Data Augmentation	-554.549 ± 98.1876	6.82 ± 2.29076
REM	$\mathbf{-507.425 \pm 75.1365}$	6.66 ± 1.02196
REM/kmeans	-532.852 ± 80.0554	7.38 ± 1.12054
REM/kmeans/f=1.5	-716.572 ± 111.819	9.61 ± 0.630793
REM/kmeans/f=2	-792.355 ± 114.228	9.91 ± 0.286182

Tabelle A.7: Vergleich der Test-log-Likelihood und der GMM-Größen für die diskutierten Samplingverfahren auf den Datensätzen A1–A6. *Data Augmentation* bezeichnet den Algorithmus mit randomisiertem I- und P-step aus Kapitel 5, *randomisiert EM* die Mischform mit deterministischem E-step und randomisiertem M-step. Bei den Varianten, die mit *kmeans* markiert sind, wurden kleinere, mit dem k-means-Algorithmus vorverarbeitete initiale GMMs verwendet. Der Parameter f bezeichnet den Parameter zur gezielten Beeinflussung der Samplingverteilung aus Abschnitt 6.3.3. Das jeweils beste Verfahren ist durch Fettdruck herausgehoben.

Verfahren	Test-log-Likelihood	mittlere GMM-Größe
	Datensatz B1	
Data Augmentation	-13137.2 ± 135.211	9.86 ± 2.49808
REM/kmeans	$\mathbf{-13013 \pm 98.7721}$	7.52 ± 1.09982
	Datensatz B2	
Data Augmentation	-428.318 ± 89.4631	3.82 ± 1.00379
REM/kmeans	$\mathbf{-296.413 \pm 128.729}$	3.07 ± 0.724638
	Datensatz B3	
Data Augmentation	-2316.07 ± 128.966	4.3 ± 0.818535
REM/kmeans	$\mathbf{-2255.65 \pm 106.91}$	4.25 ± 0.726292
	Datensatz B4	
Data Augmentation	$\mathbf{-226.056 \pm 40.1212}$	3.82 ± 0.817068
REM/kmeans	-227.222 ± 43.3753	3.17 ± 0.633325
	Datensatz B5	
Data Augmentation	-3098.34 ± 78.3729	2.94 ± 0.797747
REM/kmeans	$\mathbf{-3086.78 \pm 60.5401}$	2.69 ± 0.730685
	Datensatz B6	
Data Augmentation	-92.5973 ± 16.3509	2.71 ± 0.73885
REM/kmeans	$\mathbf{-90.8874 \pm 15.1448}$	2.58 ± 0.619355

Tabelle A.8: Vergleich der Test-log-Likelihood und der GMM-Größen für die diskutierten Samplingverfahren auf den Datensätzen B1–B6. Die Spalten haben die selbe Bedeutung wie in Tabelle A.7.

Verfahren	Log-Likelihood	GMM-Größe	Fail	P-Wert	
Datensatz A1					
REM	-24689 ± 182.7	7.31 ± 1.59	0		
EM AIC	-24764 ± 204.1	5.43 ± 1.27	0	0.3%	▲
EM BIC	-24746 ± 185.0	3.43 ± 0.68	0	1.5%	▲
EM Evidenz	-24749 ± 196.5	4.25 ± 0.99	0	1.4%	▲
SAEM AIC	-24779 ± 214.1	5.55 ± 1.40	0	0.08%	▲
SAEM BIC	-24744 ± 179.0	3.43 ± 0.62	0	1.7%	▲
SAEM Evid.	-24742 ± 187.7	4.24 ± 0.96	0	2.3%	▲
SAEM CV5	-24744 ± 171.6	4.48 ± 1.20	1	1.5%	▲
Datensatz A2					
REM	-5237 ± 50.16	6.06 ± 1.41	0		
EM AIC	-5248 ± 51.98	4.84 ± 1.08	0	6.1%	
EM BIC	-5226 ± 36.40	2.36 ± 0.48	0	95.6%	▼
EM Evidenz	-5280 ± 64.91	6.13 ± 1.69	0	0.00%	▲
SAEM AIC	-5251 ± 55.10	5.00 ± 1.25	0	2.9%	▲
SAEM BIC	-5226 ± 36.73	2.36 ± 0.48	0	95.7%	▼
SAEM Evid.	-5282 ± 70.13	6.38 ± 1.87	0	0.00%	▲
SAEM CV5	-5227 ± 37.10	2.65 ± 0.67	0	94.6%	
Datensatz A3					
REM	-3635 ± 36.55	3.51 ± 0.77	0		
EM AIC	-3644 ± 40.01	2.85 ± 0.79	0	5.1%	
EM BIC	-3639 ± 33.29	2.09 ± 0.29	0	23.1%	
EM Evidenz	-3639 ± 33.31	2.10 ± 0.30	0	24.3%	
SAEM AIC	-3645 ± 39.25	2.86 ± 0.75	0	3.9%	▲
SAEM BIC	-3639 ± 33.32	2.09 ± 0.29	0	22.7%	
SAEM Evid.	-3639 ± 33.30	2.11 ± 0.31	0	24.6%	
SAEM CV5	-3647 ± 52.68	3.29 ± 2.16	0	3.2%	▲

Verfahren	Log-Likelihood	GMM-Größe	Fail	P-Wert	
Datensatz A4					
REM	-26047 ± 178.1	5.93 ± 0.90	0		
SAEM AIC	-26451 ± 369.8	4.51 ± 1.81	0	0.00%	▲
SAEM BIC	-26576 ± 328.4	3.15 ± 0.65	0	0.00%	▲
SAEM Evid.	-26483 ± 344.8	3.65 ± 1.10	0	0.00%	▲
SAEM CV5	-26491 ± 414.8	5.27 ± 1.61	0	0.00%	▲
Datensatz A5					
REM	-23547 ± 1211	5.02 ± 0.87	0		
SAEM AIC	-25977 ± 2580	6.17 ± 1.69	1	0.00%	▲
SAEM BIC	-25550 ± 2025	4.00 ± 0.85	0	0.00%	▲
SAEM Evid.	-27663 ± 3508	7.28 ± 2.09	4	0.00%	▲
SAEM CV5	-26833 ± 3753	4.02 ± 1.01	0	0.00%	▲
Datensatz A6					
REM	-532.9 ± 80.05	7.38 ± 1.12	0		
EM AIC	-543.4 ± 82.68	4.64 ± 1.49	0	18.2%	
EM BIC	-543.0 ± 33.76	1.19 ± 0.39	0	12.2%	
EM Evidenz	-536.7 ± 80.06	4.48 ± 1.34	0	36.8%	
SAEM AIC	-539.6 ± 79.02	4.56 ± 1.43	0	27.7%	
SAEM BIC	-543.1 ± 33.75	1.19 ± 0.39	0	12.2%	
SAEM Evid.	-531.6 ± 69.64	4.42 ± 1.27	0	54.7%	
SAEM CV5	-515.6 ± 58.69	2.98 ± 1.15	0	95.7%	▼

Tabelle A.9: Ergebnisse von REM, EM und SAEM-Algorithmus auf den Testdatensätzen A1-A6. Die Spalte *Fail* gibt die Anzahl Läufe (von 100) an, in denen das gelernte GMM so schlcht war, dass es bestimmte Testdaten überhaupt nicht beschreiben konnte. Die Spalte *P-Wert* gibt den P-Wert des einseitigen t-Tests beim Vergleich der Test-log-Likelihood mit dem REM-Verfahren an. Signifikante Verbesserungen (5%-Niveau) durch den Einsatz von REM sind mit dem Symbol ▲ markiert, signifikante Verschlechterungen mit ▼. Die Abkürzung *CV5* steht für fünffache Kreuzvalidierung.

Verfahren	Log-Likelihood	GMM-Größe	Fail	P-Wert	
Datensatz B1					
REM	-13013 ± 98.77	7.52 ± 1.10	0		
EM AIC	-13164 ± 129.3	5.97 ± 1.66	0	0.00%	▲
EM BIC	-13359 ± 198.6	2.94 ± 1.06	0	0.00%	▲
EM Evidenz	-13208 ± 152.7	4.29 ± 1.19	0	0.00%	▲
SAEM AIC	-13160 ± 129.4	5.85 ± 1.55	0	0.00%	▲
SAEM BIC	-13362 ± 205.5	2.93 ± 1.05	0	0.00%	▲
SAEM Evid.	-13210 ± 165.7	4.28 ± 1.18	0	0.00%	▲
SAEM CV5	-13199 ± 110.7	4.15 ± 0.85	0	0.00%	▲
Datensatz B2					
REM	-296.4 ± 128.7	3.07 ± 0.72	0		
EM AIC	-526.4 ± 252.9	2.66 ± 1.23	3	0.00%	▲
EM BIC	-516.3 ± 207.8	2.43 ± 0.99	1	0.00%	▲
EM Evidenz	-896.0 ± 1889	3.13 ± 1.43	9	0.1%	▲
SAEM AIC	-507.7 ± 197.2	2.58 ± 1.13	5	0.00%	▲
SAEM BIC	-512.5 ± 195.1	2.41 ± 0.96	4	0.00%	▲
SAEM Evid.	-985.7 ± 2013	3.11 ± 1.43	10	0.04%	▲
SAEM CV5	-1035 ± 2260	3.71 ± 1.21	12	0.07%	▲
Datensatz B3					
REM	-2256 ± 106.9	4.25 ± 0.73	0		
EM AIC	-2473 ± 555.2	2.65 ± 1.17	0	0.01%	▲
EM BIC	-2471 ± 555.2	2.51 ± 0.91	0	0.01%	▲
EM Evidenz	-3311 ± 953.8	3.51 ± 1.25	0	0.00%	▲
SAEM AIC	-2462 ± 578.1	2.71 ± 1.27	0	0.03%	▲
SAEM BIC	-2479 ± 573.4	2.47 ± 0.89	0	0.01%	▲
SAEM Evid.	-3270 ± 975.9	3.61 ± 1.44	0	0.00%	▲
SAEM CV5	-2842 ± 749.5	6.44 ± 1.95	0	0.00%	▲

Verfahren	Log-Likelihood	GMM-Größe	Fail	P-Wert	
Datensatz B4					
REM	-227.2 ± 43.38	3.17 ± 0.63	0		
EM AIC	-345.9 ± 248.3	5.00 ± 1.51	0	0.00%	▲
EM BIC	-233.7 ± 47.27	3.02 ± 0.14	0	15.7%	
EM Evidenz	-409.8 ± 271.7	6.13 ± 1.75	0	0.00%	▲
SAEM AIC	-335.1 ± 218.3	4.95 ± 1.66	0	0.00%	▲
SAEM BIC	-234.9 ± 49.43	3.04 ± 0.20	0	12.2%	
SAEM Evid.	-429.2 ± 280.2	6.15 ± 1.86	0	0.00%	▲
SAEM CV5	-291.8 ± 134.9	4.50 ± 1.00	1	0.00%	▲
Datensatz B5					
REM	-3087 ± 60.54	2.69 ± 0.73	0		
EM AIC	-3117 ± 326.0	4.74 ± 1.35	0	18.5%	
EM BIC	-3119 ± 89.25	2.03 ± 0.17	0	0.16%	▲
EM Evidenz	-3175 ± 382.6	6.66 ± 1.78	0	1.3%	▲
SAEM AIC	-3143 ± 244.2	4.84 ± 1.24	0	1.4%	▲
SAEM BIC	-3126 ± 46.57	2.05 ± 0.22	0	0.00%	▲
SAEM Evid.	-3173 ± 394.4	6.88 ± 1.86	0	1.6%	▲
SAEM CV5	-3257 ± 272.5	6.82 ± 1.80	0	0.00%	▲
Datensatz B6					
REM	-90.89 ± 15.14	2.58 ± 0.62	0		
EM AIC	-97.42 ± 22.66	3.32 ± 0.84	0	0.90%	▲
EM BIC	-90.08 ± 15.87	2.03 ± 0.17	0	64.3%	
EM Evidenz	-111.1 ± 30.31	4.66 ± 1.67	0	0.00%	▲
SAEM AIC	-97.03 ± 23.64	3.27 ± 0.89	0	1.5%	▲
SAEM BIC	-90.08 ± 15.87	2.03 ± 0.17	0	64.3%	
SAEM Evid.	-104.0 ± 26.98	4.15 ± 1.30	0	0.00%	▲
SAEM CV5	-98.80 ± 31.55	3.31 ± 1.95	0	1.3%	▲

Tabelle A.10: Ergebnisse von REM, EM und SAEM-Algorithmus auf den Benchmarkdatensätzen B1-B6. Die Spalten haben die selbe Bedeutung wie in Tabelle A.9.

Verfahren	Log-Likelihood	P-Wert
Datensatz A1		
REM	-3016 ± 46.94	
EM AIC	-3021 ± 40.21	23.5%
EM BIC	-3045 ± 41.83	0.00% ▲
EM Evidenz	-3034 ± 42.61	0.26% ▲
SAEM AIC	-3020 ± 40.55	29.4%
SAEM BIC	-3045 ± 42.08	0.00% ▲
SAEM Evid.	-3034 ± 41.27	0.25% ▲
Datensatz A2		
REM	-5083 ± 35.32	
EM AIC	-5097 ± 34.29	0.34% ▲
EM BIC	-5164 ± 33.34	0.00% ▲
EM Evidenz	-5073 ± 44.39	96.84% ▼
SAEM AIC	-5093 ± 35.64	2.31% ▲
SAEM BIC	-5164 ± 33.60	0.00% ▲
SAEM Evid.	-5069 ± 45.45	99.31% ▼
Datensatz A3		
REM	-718.8 ± 12.36	
EM AIC	-718.3 ± 13.09	61.4%
EM BIC	-722.3 ± 12.37	2.57% ▲
EM Evidenz	-722.3 ± 12.41	2.63% ▲
SAEM AIC	-718.1 ± 13.00	65.0%
SAEM BIC	-722.2 ± 12.48	3.00% ▲
SAEM Evid.	-722.2 ± 12.40	2.86% ▲
Datensatz A4		
REM	-1517 ± 27.06	
SAEM AIC	-1544 ± 43.76	0.00% ▲
SAEM BIC	-1566 ± 33.79	0.00% ▲
SAEM Evid.	-1556 ± 37.86	0.00% ▲
Datensatz A5		
REM	-2037 ± 246.0	
SAEM AIC	-1908 ± 291.0	99.96% ▼
SAEM BIC	-2182 ± 252.2	0.00% ▲
SAEM Evid.	-1829 ± 313.5	100% ▼
Datensatz A6		
REM	-3.429 ± 15.92	
EM AIC	-33.53 ± 24.07	0.00% ▲
EM BIC	-97.96 ± 15.67	0.00% ▲
EM Evidenz	-35.35 ± 22.81	0.00% ▲
SAEM AIC	-34.23 ± 23.45	0.00% ▲
SAEM BIC	-97.96 ± 15.67	0.00% ▲
SAEM Evid.	-35.94 ± 21.83	0.00% ▲
Datensatz B1		
REM	-1015 ± 17.13	
EM AIC	-1023 ± 21.70	0.29% ▲
EM BIC	-1066 ± 27.47	0.00% ▲
EM Evidenz	-1042 ± 26.49	0.00% ▲
SAEM AIC	-1023 ± 21.23	0.13% ▲
SAEM BIC	-1066 ± 27.76	0.00% ▲
SAEM Evid.	-1042 ± 26.01	0.00% ▲
Datensatz B2		
REM	-628.3 ± 327.4	
EM AIC	-1098 ± 514.0	0.00% ▲
EM BIC	-1151 ± 494.0	0.00% ▲
EM Evidenz	-1946 ± 4461	0.19% ▲
SAEM AIC	-1087 ± 480.8	0.00% ▲
SAEM BIC	-1133 ± 469.0	0.00% ▲
SAEM Evid.	-2185 ± 5418	0.24% ▲
Datensatz B3		
REM	-3282 ± 170.1	
EM AIC	-3697 ± 920.9	0.00% ▲
EM BIC	-3711 ± 916.3	0.00% ▲
EM Evidenz	-4999 ± 1487	0.00% ▲
SAEM AIC	-3672 ± 962.7	0.01% ▲
SAEM BIC	-3722 ± 940.1	0.00% ▲
SAEM Evid.	-4893 ± 1532	0.00% ▲
Datensatz B4		
REM	-332.8 ± 53.69	
EM AIC	-246.9 ± 52.53	100% ▼
EM BIC	-312.4 ± 21.15	99.97% ▼
EM Evidenz	-216.4 ± 53.50	100% ▼
SAEM AIC	-249.5 ± 58.94	100% ▼
SAEM BIC	-311.7 ± 20.49	99.98% ▼
SAEM Evid.	-214.7 ± 60.63	100% ▼
Datensatz B5		
REM	-4159 ± 84.91	
EM AIC	-3912 ± 428.0	100% ▼
EM BIC	-4217 ± 123.8	0.01% ▲
EM Evidenz	-3758 ± 450.6	100% ▼
SAEM AIC	-3973 ± 348.8	100% ▼
SAEM BIC	-4225 ± 38.61	0.00% ▲
SAEM Evid.	-3770 ± 445.1	100% ▼
Datensatz B6		
REM	-126.4 ± 19.81	
EM AIC	-104.4 ± 22.691	100% ▼
EM BIC	-135.2 ± 13.6051	0.02% ▲
EM Evidenz	-81.02 ± 33.299	100% ▼
SAEM AIC	-106.3 ± 23.87	100% ▼
SAEM BIC	-135.2 ± 13.61	0.02% ▲
SAEM Evid.	-92.49 ± 27.36	100% ▼

Tabelle A.11: Vergleich der Log-Likelihood auf den jeweiligen Trainingsmengen. Die Spalte *P-Wert* gibt den P-Wert des einseitigen t-Tests zum Vergleich der Likelihood von REM und dem jeweiligen Vergleichsverfahren an, die Symbole ▲ und ▼ kennzeichnen signifikante Verbesserungen oder Verschlechterungen.

Datensatz	REM einzelner Lauf	REM Komitee
A1	-24688.7 ± 182.684	-24599.1 ± 150.182
A2	-5236.76 ± 50.1576	-5201.64 ± 35.8738
A3	-3635.18 ± 36.5539	-3631.45 ± 35.3509
A4	-26046.9 ± 178.09	-25976.6 ± 127.898
A5	-23546.5 ± 1211.17	-21972.9 ± 787.159
A6	-532.852 ± 80.0554	-441.057 ± 50.0116
B1	-13013 ± 98.7721	-12949.9 ± 68.6626
B2	-296.413 ± 128.729	-139.342 ± 53.2041
B3	-2255.65 ± 106.91	-2106.12 ± 49.0683
B4	-227.222 ± 43.3753	-212.003 ± 25.5552
B5	-3086.78 ± 60.5401	-3035.25 ± 37.3246
B6	-90.8874 ± 15.1448	-86.1315 ± 13.0731

Tabelle A.12: Vergleich der Test-log-Likelohood von REM ohne Komiteebildung und mit Komitees aus 10 Mitgliedern.

Verfahren	Test-log-Likelihood	P-Wert	
Datensatz A1			
REM	-24599 ± 150.2		
EM AIC	-24712 ± 191.7	0.00%	▲
EM BIC	-24736 ± 174.6	0.00%	▲
EM Evidenz	-24705 ± 182.6	0.00%	▲
SAEM AIC	-24702 ± 185.9	0.00%	▲
SAEM BIC	-24725 ± 174.5	0.00%	▲
SAEM Evidenz	-24702 ± 180.0	0.00%	▲
SAEM CV5	-24701 ± 165.6	0.00%	▲
Datensatz A2			
REM	-5202 ± 35.87		
EM AIC	-5222 ± 36.89	0.00%	▲
EM BIC	-5225 ± 35.80	0.00%	▲
EM Evidenz	-5239 ± 43.00	0.00%	▲
SAEM AIC	-5221 ± 35.77	0.01%	▲
SAEM BIC	-5224 ± 35.70	0.00%	▲
SAEM Evidenz	-5241 ± 41.11	0.00%	▲
SAEM CV5	-5219 ± 37.29	0.04%	▲
Datensatz A3			
REM	-3631.45 ± 35.35		
EM AIC	-3644.49 ± 39.88	0.8%	▲
EM BIC	-3638.83 ± 33.29	6.6%	
EM Evidenz	-3638.60 ± 33.32	7.2%	
SAEM AIC	-3644.06 ± 38.22	0.8%	▲
SAEM BIC	-3638.86 ± 33.30	6.5%	
SAEM Evidenz	-3638.64 ± 33.33	7.1%	
SAEM CV5	-3643.81 ± 46.28	1.8%	▲

Verfahren	Test-log-Likelihood	P-Wert	
Datensatz A4			
REM	-25976 ± 127.9		
SAEM AIC	-26317 ± 304.2	0.00%	▲
SAEM BIC	-26508 ± 277.3	0.00%	▲
SAEM Evidenz	-26381 ± 300.4	0.00%	▲
SAEM CV5	-26257 ± 276.1	0.00%	▲
Datensatz A5			
REM	-21972 ± 787.2		
SAEM AIC	-23046 ± 1296	0.00%	▲
SAEM BIC	-23411 ± 1569	0.00%	▲
SAEM Evidenz	-23736 ± 1088	0.00%	▲
SAEM CV5	-24543 ± 3711	0.00%	▲
Datensatz A6			
REM	-441.1 ± 50.01		
EM AIC	-486.0 ± 64.29	0.00%	▲
EM BIC	-537.2 ± 43.30	0.00%	▲
EM Evidenz	-472.3 ± 61.98	0.01%	▲
SAEM AIC	-483.4 ± 65.94	0.00%	▲
SAEM BIC	-537.2 ± 43.30	0.00%	▲
SAEM Evidenz	-470.7 ± 63.62	0.02%	▲
SAEM CV5	-447.7 ± 54.22	18.6%	

Tabelle A.13: Ergebnisse der Dichteschätzung mit Komitees von GMMs. Jedes Komitee besteht aus 10 Komiteeteilnehmern. Die Spalte *P-Wert* gibt den P-Wert des einseitigen t-Tests beim Vergleich der Test-log-Likelihood mit dem REM-Verfahren an. Signifikante Verbesserungen (5%-Niveau) durch den Einsatz von REM sind mit dem Symbol ▲ markiert, signifikante Verschlechterungen mit ▼. Die Abkürzung *CV5* steht für fünffache Kreuzvalidierung.

Verfahren	Test-log-Likelihood	P-Wert
Datensatz B1		
REM	-12950 ± 68.66	
EM AIC	-13051 ± 91.29	0.00% ▲
EM BIC	-13322 ± 218.8	0.00% ▲
EM Evidenz	-13127 ± 160.9	0.00% ▲
SAEM AIC	-13049 ± 93.63	0.00% ▲
SAEM BIC	-13309 ± 218.2	0.00% ▲
SAEM Evidenz	-13119 ± 165.0	0.00% ▲
SAEM CV5	-13128 ± 132.9	0.00% ▲
Datensatz B2		
REM	-139.3 ± 53.20	
EM AIC	-528.9 ± 317.4	0.00% ▲
EM BIC	-530.8 ± 316.1	0.00% ▲
EM Evidenz	-302.6 ± 198.7	0.00% ▲
SAEM AIC	-482.4 ± 354.9	0.00% ▲
SAEM BIC	-484.4 ± 355.3	0.00% ▲
SAEM Evidenz	-246.4 ± 213.0	0.00% ▲
SAEM CV5	-345.2 ± 145.6	0.00% ▲
Datensatz B3		
REM	-2106 ± 49.07	
EM AIC	-2290 ± 501.9	0.02% ▲
EM BIC	-2348 ± 515.1	0.00% ▲
EM Evidenz	-2008 ± 251.0	100% ▼
SAEM AIC	-2225 ± 512.9	1.2% ▲
SAEM BIC	-2295 ± 535.9	0.03% ▲
SAEM Evidenz	-2012 ± 291.1	99.9% ▼
SAEM CV5	-2041 ± 248.2	99.4% ▼

Verfahren	Test-log-Likelihood	P-Wert
Datensatz B4		
REM	-212.0 ± 25.56	
EM AIC	-268.3 ± 63.11	0.00% ▲
EM BIC	-231.7 ± 44.77	0.01% ▲
EM Evidenz	-275.6 ± 66.90	0.00% ▲
SAEM AIC	-263.6 ± 58.28	0.00% ▲
SAEM BIC	-229.7 ± 44.11	0.03% ▲
SAEM Evidenz	-281.1 ± 75.32	0.00% ▲
SAEM CV5	-235.8 ± 48.55	0.00% ▲
Datensatz B5		
REM	-3035 ± 37.32	
EM AIC	-2700 ± 371.1	100% ▼
EM BIC	-3112 ± 114.8	0.00% ▲
EM Evidenz	-2563 ± 334.3	100% ▼
SAEM AIC	-2731 ± 369.0	100% ▼
SAEM BIC	-3112 ± 112.9	0.00% ▲
SAEM Evidenz	-2615 ± 367.5	100% ▼
SAEM CV5	-2835 ± 360.0	100% ▼
Datensatz B6		
REM	-86.13 ± 13.07	
EM AIC	-92.45 ± 20.41	0.5% ▲
EM BIC	-89.64 ± 15.47	4.3% ▲
EM Evidenz	-99.26 ± 22.27	0.00% ▲
SAEM AIC	-85.70 ± 25.75	56%
SAEM BIC	-89.64 ± 15.47	4.3% ▲
SAEM Evidenz	-87.08 ± 38.00	40.7%
SAEM CV5	-90.71 ± 19.36	2.6% ▲

Tabelle A.14: Ergebnisse der Dichteschätzung mit Komitees von GMMs für die Benchmark-Datensätze B1–B6. Die Spalten haben die selbe Bedeutung wie in Tabelle A.13.

Anhang B

Wahrscheinlichkeitsverteilungen

Univariate, kontinuierliche Verteilungen

Univariate Gleichverteilung $U(a, b)$

Parameter	$a, b \in \mathbb{R}, a < b$
Trägermenge	$\mathbb{R}$
Dichte	$\begin{cases} \frac{1}{b-a} & \text{falls } a < x < b \\ 0 & \text{sonst} \end{cases}$
Erwartungswert	$\frac{a+b}{2}$
Varianz	$\frac{(b-a)^2}{12}$

Univariate Normalverteilung $N(\mu, \sigma^2)$

Parameter	Lageparameter $\mu \in \mathbb{R}$, Streuparameter $\sigma^2 \in \mathbb{R}_{>0}$
Trägermenge	$\mathbb{R}$
Dichte	$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\}$
Erwartungswert	μ
Modalwert	μ
Varianz	σ^2

Gammaverteilung $\Gamma(\beta, \alpha)$

Parameter	Formparameter $\alpha \in \mathbb{R}_{>0}$, Skalierungsparameter $\beta \in \mathbb{R}_{>0}$
Trägermenge	$\mathbb{R}_{>0}$
Dichte	$\frac{\beta^\alpha}{\Gamma(\alpha)} \frac{x^{\alpha-1}}{\exp\{\beta x\}}$
Erwartungswert	$\frac{\alpha}{\beta}$
Modalwert	$\frac{\alpha-1}{\beta}$
Varianz	$\frac{\alpha}{\beta^2}$

Inverse Gammaverteilung $\Gamma^{-1}(\beta, \alpha)$

Parameter	Formparameter $\alpha \in \mathbb{R}_{>0}$, Skalierungsparameter $\beta \in \mathbb{R}_{>0}$

Trägermenge	$\mathbb{R}_{>0}$
Dichte	$\frac{\beta^\alpha}{\Gamma(\alpha)} \frac{1}{x^{\alpha+1} \exp\{\frac{\beta}{x}\}}$
Erwartungswert	$\frac{\beta}{\alpha-1}$ (existiert nur für $\alpha > 1$)
Modalwert	$\frac{\beta}{\alpha+1}$
Varianz	$\frac{\beta^2}{(\alpha-1)^2(\alpha-2)}$ (existiert nur für $\alpha > 2$)
Zusammenhänge	ist $X \sim \Gamma(\beta, \alpha)$, so ist $\frac{1}{X} \sim \Gamma^{-1}(\beta, \alpha)$

χ^2-Verteilung $\chi^2(n)$

Parameter	Formparameter $n \in \mathbb{R}_{>0}$
Trägermenge	$\mathbb{R}_{>0}$
Erwartungswert	n
Varianz	$2n$
Zusammenhänge	$\chi^2(n) = \Gamma(\frac{1}{2}, \frac{n}{2})$ ist $X_1, \ldots, X_n \sim_{i.i.d.} N(0,1)$, so ist $\sum_{i=1}^n X_i^2 \sim \chi^2(n)$

t-Verteilung $t(n, \mu, \sigma^2)$

Parameter	Formparameter $n \in \mathbb{R}_{>0}$, Lageparameter $\mu \in \mathbb{R}$, Skalierungsparameter $\sigma^2 \in \mathbb{R}_{>0}$
Trägermenge	$\mathbb{R}$
Dichte	$\frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})\sqrt{n\pi\sigma^2}}(1 + \frac{(x-\mu)^2}{n\sigma^2})^{-\frac{n+1}{2}}$
Erwartungswert	μ (für $n \geq 2$)
Varianz	$\frac{n\sigma^2}{n-2}$ (für $n \geq 3$)

Beta-Verteilung $B(\alpha, \beta)$

Parameter	Formparameter $\alpha \in \mathbb{R}_{>0}$, $\beta \in \mathbb{R}_{>0}$
Trägermenge	$[0, 1]$
Dichte	$\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$
Erwartungswert	$\frac{\alpha}{\alpha+\beta}$
Modalwert	$\frac{\alpha-1}{\alpha+\beta-2}$ (für $\alpha, \beta > 1$)
Varianz	$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$

Univariate, diskrete Verteilungen

Binomialverteilung $b(n, p)$

Parameter	$n \in \mathbb{N}$, $p \in [0, 1]$
Trägermenge	$\{0, 1, \ldots, n\}$
Zähldichte	$\binom{n}{x} p^x (1-p)^{n-x}$
Erwartungswert	np
Varianz	$np(1-p)$

Poissonverteilung $\pi(\alpha)$

Parameter	$\alpha \in \mathbb{R}_{>0}$
Trägermenge	$\mathbb{N}_0$
Zähldichte	$\frac{\alpha^x}{e^\alpha x!}$
Erwartungswert	α
Varianz	α

Multivariate Verteilungen

d-variate Normalverteilung $N(\mu, \Sigma)$

Parameter	Lageparameter $\mu \in \mathbb{R}^d$, Streuparameter $\Sigma \in \mathbb{R}^{d\times d}$ symmetrisch, positiv definit
Trägermenge	$\mathbb{R}^d$
Dichte	$\frac{1}{\sqrt{2\pi}^d\sqrt{\lvert\Sigma\rvert}} \exp\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\}$
Erwartungswert	μ
Varianz/Kovarianz	Σ

$d \times d$-variate Wishartverteilung $W(q, \Lambda)$

Parameter	$q \in \mathbb{R}$, $q \geq d$, $\Lambda \in \mathbb{R}^{d\times d}$, symmetrisch, positiv definit
Trägermenge	$\mathbb{R}^{d\times d}$, symmetrisch, positiv definit
Dichte	$\propto \lvert X\rvert^{\frac{q-d-1}{2}} e^{-\frac{1}{2}Spur(\Lambda^{-1}X)}$
Erwartungswert	$q\Lambda$
Zusammenhänge	für $d=1$ gilt: $W(q,(\lambda)) = \Gamma(\frac{1}{2\lambda}, \frac{q}{2})$

$d \times d$-variate Inverse Wishartverteilung $W^{-1}(q, \Lambda)$

Parameter	$q \in \mathbb{R}$, $q \geq d$, $\Lambda \in \mathbb{R}^{d\times d}$, symmetrisch, positiv definit
Trägermenge	$\mathbb{R}^{d\times d}$, symmetrisch, positiv definit
Dichte	$\propto \lvert X\rvert^{-\frac{q+d+1}{2}} e^{-\frac{1}{2}Spur(\Lambda^{-1}X^{-1})}$
Erwartungswert	$\frac{1}{q-d-1}\Lambda^{-1}$ (existiert nur für $q \geq d+2$)
Modalwert	$\frac{1}{q+d+1}\Lambda^{-1}$
Zusammenhänge	ist $X \sim W(q,\Lambda)$, so ist $X^{-1} \sim W^{-1}(q,\Lambda)$
	für $d=1$ gilt: $W^{-1}(q,(\lambda)) = \Gamma^{-1}(\frac{1}{2\lambda}, \frac{q}{2})$

d-variate Dirichletverteilung $D(\alpha_1, \ldots, \alpha_d)$

Parameter	$\alpha_j \in \mathbb{R}_{>0}$
Trägermenge	$\{(x_1,\ldots,x_d) \in [0,1]^d \mid \sum_{j=1}^d x_j = 1\}$ (d-dimensionaler Simplex)
Dichte	$\frac{\Gamma(\alpha_1+\cdots+\alpha_d)}{\Gamma(\alpha_1)\cdot\cdots\cdot\Gamma(\alpha_d)} \prod_{j=1}^d x_j^{\alpha_j-1}$
Erwartungswert	$(\frac{\alpha_1}{\sum_{j=1}^d \alpha_j}, \ldots, \frac{\alpha_d}{\sum_{j=1}^d \alpha_j})$
Modalwert	$\frac{1}{\sum_{j=1}^k \alpha_j - k} \cdot (\alpha_1 - 1, \alpha_2 - 1, \ldots, \alpha_k - 1)$ (für $\alpha_j > 1$ für alle j)

Zusammenhänge	ist $(x_1, \ldots, x_d) \sim D(\alpha_1, \ldots, \alpha_d)$, so ist $x_k \sim B(\alpha_k, \sum_{j=1}^{d} \alpha_j - \alpha_k)$

d-variate Multinomialverteilung $\mathcal{M}(n, p_1, \ldots, p_d)$

Parameter	$p_j \in [0,1]$, $\sum_{j=1}^{d} p_j = 1$
Trägermenge	$\{(x_1, \ldots, x_d) \in \{0, 1, \ldots, n\}^d \mid \sum_{j=1}^{d} x_j = n\}$
Dichte	$\frac{n!}{x_1! \cdot \ldots \cdot x_d!} \prod_{j=1}^{d} p_j^{x_j}$
Erwartungswert	$(np_1, \ldots, np_d)$
Zusammenhänge	ist $(x_1, \ldots, x_d) \sim \mathcal{M}(p_1, \ldots, p_d)$, so ist $x_k \sim b(n, p_k)$

Anhang C

Zufallszahlengeneratoren

Abhandlungen und Beispiele für Zufallszahlengeneratoren, die gleichverteilte Pseudo-Zufallszahlen erzeugen, finden sich z.B. in [Knuth 69, Ripley 87, Robert 99]. Mit deren Hilfe lassen sich Pseudo-Zufallszahlen für alle beschriebenen Verteilungen erzeugen.

Die Erzeugung univariater normalverteilter Zufallszahlen kann mit der Methode aus [Box 58] erfolgen. Sie beruht auf folgendem Zusammenhang:

$$\begin{aligned} &u_1, u_2 \sim_{i.i.d.} U(0,1) \quad \Rightarrow \quad v_1, v_2 \sim_{i.i.d.} N(0,1) \\ &\text{mit } v_1 := \sqrt{-2\log u_1}\cos(2\pi u_2),\ v_2 := \sqrt{-2\log u_1}\sin(2\pi u_2) \end{aligned} \tag{C.1}$$

d-variat normalverteilte Zufallsvektoren können erzeugt werden durch Zurückführen auf d stochastisch unabhängige, univariate, standard-normalverteilte Zufallszahlen. Dazu wird der Zusammenhang verwendet:

$$Z \sim N(0, I_d) \text{ und } L \cdot L^T = \Sigma \quad \Rightarrow \quad X := LZ + \mu \sim N(\mu, \Sigma) \tag{C.2}$$

Eine geeignete Matrix L ist beispielsweise der Cholesky-Faktor von Σ.

Γ-verteilte Zufallszahlen können mit der Methode aus [Ahrens 74] aus gleichverteilten Zufallszahlen mit Hilfe eines Accept-/Reject-Ansatzes gewonnen werden, ebenso Γ^{-1}-verteilte Zufallszahlen. Die χ^2-Verteilung ergibt sich als Spezialfall der Γ-Verteilung.

Dirichlet-verteilte Zufallsvektoren lassen sich aus Γ-verteilten Zufallszahlen unter Zuhilfenahme des folgenden Zusammenhangs erzeugen:

$$X_j \sim \Gamma(1, \alpha_i) \text{ unabhängig} \quad \Rightarrow \quad \left(\frac{X_1}{\sum_{j=1}^d X_j}, \ldots, \frac{X_d}{\sum_{j=1}^d X_j}\right) \sim D(\alpha_1, \ldots, \alpha_d) \tag{C.3}$$

Beta-verteilte Zufallsvariablen ergeben sich als Spezialfälle der Dirichletverteilung.

Binomial- und Multinomial-verteilte Zufallsgrößen lassen sich direkt durch Simulation des zugehörigen Urnenmodells erhalten.

Die Erzeugung Wishart- und Invers-Wishart-verteilter Matrizen kann in zwei Schritten auf die Erzeugung unabhängiger normal- und χ^2-verteilter Zufallszahlen zurückgeführt werden [Muirhead 82, Schafer 97]:

$$Z \sim W(q, I_d) \text{ und } L \cdot L^T = \Lambda \quad \Rightarrow \quad LZL^T \sim W(q, \Lambda) \tag{C.4}$$

sowie (Bartlett-Zerlegung):

$$\left.\begin{array}{l} T = (t_{ij}) \text{ ist obere Dreiecksmatrix mit} \\ t_{ii} \geq 0 \\ t_{ii}^2 \sim \chi^2(q+1-i) \\ t_{ij} \sim N(0,1) \text{ für } i < j \\ t_{ij} \text{ unabhängig für alle } i,j \end{array}\right\} \quad \Rightarrow \quad TT^T \sim W(q, I_d) \tag{C.5}$$

Anhang D

Mathematische Notation

Die nachfolgende Tabelle enthält einige der verwendeten Bezeichner und Formelsymbole, sofern sie nicht in der allgemein üblichen Notation verwendet wurden. Weitere Bezeichner sind jeweils dort erklärt, wo sie verwendet werden. Bei reellen Zahlen in Dezimalschreibweise wird als Trennzeichen zwischen dem ganzzahligen Anteil und dem Bruchanteil der Dezimalpunkt statt dem Komma verwendet, z.B. 1.5 für die Zahl $\frac{3}{2}$, nicht $1,5$.

i.i.d.	"independently identically distributed": unabhängig, aber gemäß der gleichen Verteilung verteilte Zufallsgrößen
$X \sim K$	die Größe (Zufallsvariable) X ist verteilt gemäß Verteilung K
$\mathbb{R}$	reelle Zahlen
$\mathbb{R}_{>0}$	positive reelle Zahlen
$\mathbb{R}_{\geq 0}$	nicht-negative reelle Zahlen
$\mathbb{N}$	natürliche Zahlen ohne Null
$\mathbb{N}_0$	natürliche Zahlen einschließlich Null
$\mathbb{R}^d$	Menge der d-dimensionale, reellwertige Vektoren (i.d.R. Spaltenvektoren)
$\mathbb{R}^{c \times d}$	Menge der reellwertige Matrizen mit c Zeilen und d Spalten
I_d	Einheitsmatrix der Dimension d
$\lvert M \rvert$	Determinante der Matrix M
M^T	Transponierte der Matrix (des Vektors) M
$\#S$	Kardinalität einer Menge S
$\Gamma(\cdot)$	Gamma-Funktion, $\Gamma(x) := \int_0^\infty t^{x-1} e^{-t} dt$
$\mathcal{P}(\cdot)$	Wahrscheinlichkeit (generisch)
$E[X]$	Erwartungswert der Zufallsvariable X
$\delta(\cdot)$	Dirac-Stoß (Kammfunktion)
$\log x$	natürlicher Logarithmus von x
$\mathbb{I}_{[bed]}$	Indikatorfunktion: liefert 1, falls *bed* wahr ist, 0 sonst

Folgende Buchstaben werden i.d.R. mit nachfolgend aufgeführter Bedeutung verwendet:

d	Dimensionalität der Daten

i	Indexvariable für die Datenpunkte
j	Indexvariable für die Komponenten eines GMM
k	Größe eines GMM, Anzahl Komponenten
n	Anzahl Datenpunkte, Größe der Trainingsmenge
p, q	Dichten von Verteilungen
x_i	i-ter Datenpunkt der Trainingsmenge
ϑ	Parametervektor eines Modells
$\mathcal{H}$	Hypothesenraum

Literaturverzeichnis

[Ahrens 74] J. H. Ahrens & U. Dieter. Computer methods for sampling from gamma, beta, poisson and binomial distributions. Computing, Band 12, S. 223–246, 1974.

[Akaike 73] H. Akaike. Information theory and an extension of the maximum likelihood principle. In: B. N. Petrov & F. Csaki (Herausgeber), Second International Symposium on Information Theory, S. 267–281, 1973.

[Andrieu 03] Christophe Andrieu, Nando de Freitas, Arnaud Doucet & Michael I. Jordan. An introduction to MCMC for machine learning. Machine Learning, Band 50, Nr. 1-2, S. 5–43, 2003.

[Barnett 78] Vic Barnett & Toby Lewis. *Outliers in Statistical Data*. Wiley, 1978.

[Ben-David 97] Shai Ben-David & Michael Lindenbaum. Learning distributions by their density levels: A paradigm for learning without a teacher. Journal of Computer and System Sciences, Band 55, S. 171–182, 1997.

[Bishop 95] Christopher M. Bishop. *Neural Networks for Pattern Classification*. Oxford University Press, 1995.

[Blake 98] C.L. Blake & C.J. Merz. UCI repository of machine learning databases, 1998.

[Bosch 93] Karl Bosch. *Statistik-Taschenbuch*. R. Oldenbourg Verlag, 1993.

[Box 58] G.E.P. Box & M. Muller. A note on the generation of random normal variates. Annals of the Institute of Statistical Mathematics, Band 29, S. 610–611, 1958.

[Bozdogan 84] Hamparsum Bozdogan & Stanley L. Sclove. Multi-sample cluster analysis unsing Akaike's information criterion. Annals of the Institute of Statistical Mathematics, Band 36, S. 163–180, 1984.

[Broniatowski 83] M. Broniatowski, G. Celeux & J. Diebolt. Reconnaissance de mélanges de densités par un algorithme d'apprentissage probabiliste. Data Analysis and Informatics, Band 3, S. 359–374, 1983.

[Burnham 98] Kenneth P. Burnham & David R. Anderson. *Model selection and inference: a practical information-theoretic approach*. Springer, 1998.

[Celeux 92] G. Celeux & J Diebolt. A stochastic approximation type EM algorithm for the mixture problem. Stochastics and Stochastics Report, Band 41, S. 119–134, 1992.

[Celeux 95] Gilles Celeux, Didier Chauveau & Jean Diebolt. On stochastic versions of the EM algoritm. Forschungsbericht RR-2514, Institut national de recherche en informatique et en automatique (INRIA) Rhône-Alpes, 1995.

[Dempster 77] A. P. Dempster, N. M. Laird & D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society Series B, Band 39, S. 1–38, 1977.

[Devroye 87] Luc Devroye. *A Course in Density Estimation*. Birkhäuser, 1987.

[Diebolt 94] Jean Diebolt & Christian P. Robert. Estimation of finite mixtures through Bayesian sampling. Journal of the Royal Statistical Society Series B, Band 56, Nr. 2, S. 363–375, 1994.

[Duda 73] Richard O. Duda & Peter E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[Feller 68] William Feller. *An Introduction to Probability Theory and its Applications*, Band 1. Wiley, 1968.

[Forgy 65] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. Biometrics, Band 21, Nr. 3, S. 768, 1965.

[Fraley 98] C. Fraley & A. E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. Forschungsbericht 329, Department of Statistics, University of Washington, 1998.

[Geman 84] S. Geman & D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restauration of images. IEEE Transactions in Pattern Analysis and Machine Intelligence, Band 6, S. 721–741, 1984.

[Ghahramani 94] Zoubin Ghahramani & Michael I. Jordan. Learning from incomplete data. Forschungsbericht 1509, MIT Artificial Intelligence Laboratory, 1994.

[Ghosh 01] J. K. Ghosh & Tapas Samanta. Model selection - an overview. Current Science, Band 80, Nr. 9, S. 1135–1144, 2001.

[Gilks 96] W. R. Gilks, S. Richardson & D. J. Spiegelhalter (Herausgeber). *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

[Green 95] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Biometrika, Band 82, S. 711–732, 1995.

[Hastings 70] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. Biometrika, Band 57, S. 97–109, 1970.

[Hertz 91] John Hertz, Anders Krogh & Richard G. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley, 1991.

[Heuser 86] Harro Heuser. *Lehrbuch der Analysis*, Band 2. Teubner, 1986.

[Jeffreys 61] Harold Jeffreys. *Theory of Probability*. Clarendon Press, 1961.

[Kaufman 90] Leonard Kaufman & Peter J. Rousseeuw. *Finding Groups in Data*. Wiley, 1990.

[Kirkpatrick 83] S. Kirkpatrick, C. D. Gelatt & M. P. Vecchi. Optimization by simulated annealing. Science, Band 220, S. 671–680, 1983.

[Knuth 69] Donald E. Knuth. *The Art of Computer Programming, Part 2: Seminumerical Algorithms*. Addison-Wesley, 1969.

[Lauer 01a] Martin Lauer. A mixture approach to novelty detection using training data with outliers. In: Proceedings of the 12th European Conference on Machine Learning, S. 300–311, 2001.

[Lauer 01b] Martin Lauer, Wolfram Menzel & Martin Riedmiller. Abschlussbericht zum Forschungsprojekt: Erkennung unplausibler Messwerte in Kernstrahlungs-Datenbanken. Forschungsbericht, Universität Karlsruhe, Institut für Logik, Komplexität und Deduktionssysteme, 2001.

[Lauer 02] Martin Lauer. Sampling parameters to estimate a mixture distribution with unknown size. In: Artificial Neural Networks – ICANN 2002, S. 414–419, 2002.

[MacKay 03] David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[MacQueen 67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkley Symposium on Mathematical Statistics and Probability, Band I, Statistics, S. 281–297, 1967.

[McLachlan 88] Geoffrey J. McLachlan & Kaye E. Basford. *Mixture Models – Inference and Applications to Clustering*. Dekker, 1988.

[McLachlan 00] Geoffrey McLachlan & David Peel. *Finite Mixture Models*. Wiley, 2000.

[Metropolis 53] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller & E. Teller. Equations of state calculations by fast computing machines. The Journal of Chemical Physics, Band 21, S. 1087–1092, 1953.

[Mitchell 97] Tom M. Mitchell. *Machine learning*. McGraw-Hill, 1997.

[Moody 89] J. Moody & C. J. Darken. Fast learning in networks of locally-tuned processing units. Neural Computing, Band 1, S. 281–294, 1989.

[Muirhead 82] Robb J. Muirhead. *Aspects of Multivariate Statistical Theory.* Wiley, 1982.

[Ormoneit 95] Dirk Ormoneit & Volker Tresp. Improved Gaussian mixture density estimates using Bayesian penalty terms and network averaging. In: Advances in Neural Processing Systems 8, S. 542–548, 1995.

[Parzen 63] Emanuel Parzen. On estimation of a probability density function and mode. The Annals of Mathematical Statistics, Band 33, S. 1065–1076, 1963.

[Ragg 00] Thomas Ragg. *Problemlösung durch Komitees neuronaler Netze.* Dissertation, Universität Karlsruhe, Fakultät für Informatik, 2000.

[Rätsch 99] Gunnar Rätsch. IDA benchmark repository, 1999.

[Redner 84] Richard A. Redner & Homer F. Walker. Mixture densities, maximum likelihood and the EM algorithm. SIAM Review, Band 26, Nr. 2, S. 195–239, 1984.

[Richardson 97] Sylvia Richardson & Peter J. Green. On Bayesian analysis of mixtures with an unknown number of components. Journal of the Royal Statistical Society Series B, Band 59, S. 731–792, 1997.

[Ripley 87] Brian D. Ripley. *Stochastic Simulation.* Wiley, 1987.

[Robert 94] Christian P. Robert. *The Bayesian Choice.* Springer, 1994.

[Robert 99] Christian P. Robert & George Casella. *Monte Carlo Statistical Methods.* Springer, 1999.

[Roberts 98] Stephen J. Roberts, Dirk Husmeier, Iead Rezek & William Penny. Bayesian approaches to Gaussian mixture modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence, Band 20, Nr. 11, S. 1133–1142, 1998.

[Roeder 95] Kathryn Roeder & Larry Wasserman. Practical Bayesian density estimation using mixtures of normals. Forschungsbericht 633, Department of Statistics, Carnegie Mellon University, 1995.

[Rousseeuw 87] Peter J. Rousseeuw & Annick M. Leroy. *Robust Regression and Outlier Detection.* Wiley, 1987.

[Rubin 87] D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys.* Wiley, 1987.

[Schafer 97] J. L. Schafer. *Analysis of Incomplete Multivariate Data.* Chapman & Hall, 1997.

[Schmidt 03] Volker Schmidt. Markov-Ketten und Monte-Carlo-Simulation. Skriptum, Universität Ulm, Abteilung Stochastik, 2003.

[Schwarz 78] G. Schwarz. Estimating the dimension of a model. The Annals of Statistics, Band 6, S. 461–464, 1978.

[Silverman 86] Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[Stahel 95] Werner A. Stahel. *Statistische Datenanalyse*. Vieweg, 1995.

[Stephens 97] Matthew Stephens. *Bayesian Methods for Mixtures of Normal Distributions*. Dissertation, Magdalen College, Oxford, 1997.

[Stephens 00] Matthew Stephens. Dealing with label-switching in mixture models. Journal of the Royal Statistical Society Series B, Band 62, S. 795–809, 2000.

[Tanner 87] Martin A. Tanner & Wing Hung Wong. The calculation of posterior distributions by data augmentation. Journal of the American Statistical Society, Band 82, Nr. 398, S. 528–550, 1987.

[Tax 99] David M. J. Tax & Robert P. W. Duin. Data domain description using support vectors. In: Proceedings of the European Symposium on Artificial Neural Networks, S. 251–257, 1999.

[Titterington 85] D. M. Titterington, A. F. M. Smith & U. E. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley, 1985.

[Ueda 98] Naonori Ueda, Ryohei Nakano, Zoubin Ghahramani & Geoffrey E. Hinton. SMEM algorithm for mixture models. In: Advances in Neural Information Processing Systems 11, S. 599–605, 1998.

[Vapnik 95] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[Vapnik 00] Vladimir N. Vapnik & Sayan Mukherjee. Support vector method for multivariate density estimation. In: Advances in Neural Information Processing Systems 12, S. 659–665, 2000.

[Wei 90] G. C. G. Wei & M. A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithm. Journal of the American Statistical Association, Band 85, S. 699–704, 1990.