UNIVERSITÄT DORTMUND
FACHBEREICH INFORMATIK

Daniel Vogtland

Untersuchung von Ionenmobilitätsspektrometriedaten auf Peaks von Substanzen in verschiedenen Konzentrationen unter Einsatz von Glättungs- und Fittingverfahren

Diplomarbeit

4. Mai 2007

INTERNE BERICHTE
INTERNAL REPORTS

Lehrstuhl Informatik I - Logik in der Informatik
Otto-Hahn-Str.16
44227 Dortmund

Gutachter:
Prof. Dr. Bernd Reusch
PD. Dr. Jörg Ingo Baumbach

GERMANY · D-44221 DORTMUND

Inhaltsverzeichnis

1. Einleitung

Bei der Ionenmobilitätsspektrometrie handelt es sich um ein Verfahren zur Analyse von Gasgemischen mit dem Ziel, Bestandteile bis in den Spurenbereich zu detektieren. Das grundsätzliche Prinzip besteht darin, im Gasgemisch enthaltene Moleküle zu ionisieren und anhand unterschiedlicher Geschwindigkeiten in einem elektrischen Feld bei einem gleichzeitig entgegenströmendem Gas zu unterscheiden. Durch eine Vortrennungseinheit treten unterschiedliche Moleküle zu unterschiedlichen Zeitpunkten in die Ionisierungsphase ein, so dass die entstehenden Daten zwei Dimensionen aufweisen.

In dieser Arbeit wurden einige Methoden entwickelt, diese Daten funktional beschreiben zu können und auf Basis dieser Beschreibung eine Abhängigkeitsanalyse bezüglich der Konzentration zu ermöglichen. Der Schwerpunkt lag dabei auf der Schaffung automatisierter Analysevorgänge bzw. die Unterstützung durch ein Computersystem. Auf diese Weise kann eine maschinelle Vorhersage durch ein Computerprogramm erfolgen, in welcher Konzentration eine relevante Substanz in einem unbekannten Gasgemisch vorliegt. Denkbare Anwendungen wären bekannte Einsatzgebiete wie etwa das Aufspüren von Kampfstoffen und illegalen Betäubungsmitteln, oder die Früherkennung von Krankheitsbildern wie beispielsweise Lungenkrebs.

Die Grundannahme bestand darin, dass eine bestimmte Substanz zu ein oder mehreren charakteristischen Peaks in den zweidimensionalen Daten führt und die Volumina dieser Peaks in einem engen Zusammenhang mit der Konzentration der untersuchten Substanz stehen. Da die Datenaufzeichnung in der Regel fehlerbelastet ist (Rauschen), wurden dazu zunächst Glättungsmethoden auf ihre Anwendbarkeit untersucht.

Im Kapitel 2 wird eine Einführung in die Ionenmobilitätsspektrometrie gegeben. Die Kapitel 3 und 4 beschreiben theoretische Grundlagen eingesetzter Algorithmen zur Glättung und zum Fitting von Peakmodellen und Konzentrationsabhängigkeitsmodellen.

Im Kapitel 5 werden die in dieser Arbeit verwendeten Daten und drei einfache Vorverarbeitungsschritte beschrieben. Die Datenglättung und die Erkennung von Peaks wird im Kapitel 6 vorgestellt. Dabei wird auch eine im Rahmen dieser Arbeit entwickelte Glättungsmethode vorgestellt. Die Peakerkennung erfolgt zunächst eindimensional, was dem Vorgehen ohne Einsatz einer Vortrennungseinheit entsprechen würde. Detektierte Peaks werden anschließend zu zweidimensionalen Peaks zusammengefasst. Neben der Interpolation wird dabei auch die Verwendung spezieller zweidimensionaler Funktionen betrachtet.

Die auf zweidimensionalen Peaks basierende Analyse von Konzentrationsabhängigkeiten wird im Kapitel 7 behandelt. Betrachtet werden hierbei Peakvolumina, wobei sich auf numerische Approximationen konzentriert wird. Eine Konzentrationsabhängigkeit wird als funktionale Beschreibung der Abhängigkeit zwischen der Konzentration einer untersuchten Substanz und den Volumina der zugehörigen Peaks aufgefasst. Grundlage ist die Auswahl einer Modellfunktion, bei deren Fitting einer algorithmische Unterstützung gegeben wird.

Auf die Implementierung der entwickelten Methoden wird im Kapitel 8 eingegangen. Dabei wurde eine PlugIn-Architektur entwickelt, die eine einfache Erweiterbarkeit beispielsweise durch neue Glättungsalgorithmen oder Modellfunktionen ermöglicht.

Im Kapitel 9 erfolgt eine Zusammenfassung der Ergebnisse dieser Arbeit. Darüber hinaus erfolgt ihre Diskussion und die Darlegung offener Fragestellungen.

2. Ionenmobilitätsspektrometrie

Bei der *Ionenmobilitätsspektrometrie* handelt es sich um ein Verfahren zur Analyse von Gasgemischen mit dem Ziel, Bestandteile bis in den Spurenbereich hinein zu detektieren. Ihre Entwicklung begann um 1970, ursprünglich unter dem Namen Plasma-Chromatographie [20, 34].

2.1. Methodische Grundlagen

Die grundsätzliche Methode besteht darin, Gasmoleküle bei Umgebungsdruck zu ionisieren und anhand ihrer Mobilität in einem elektrischen Feld bei einem entgegen strömenden *Driftgas* zu unterscheiden [3]. Dazu werden die zu analysierenden Moleküle (Analyte) mittels eines *Trägergases* (in der Regel Luft oder Stickstoff [30, 34]) in ein Analysesystem geführt.

Nach einer Ionisation bewegen sich Ionen der eingeleiteten Analyten im so genannten *Driftraum* unter dem Einfluss eines elektrischen Feldes. Sie werden durch dieses beschleunigt, gleichzeitig aber durch Stöße untereinander und insbesondere durch Kollisionen mit entgegen strömenden Driftgasmolekülen gebremst. Die Geschwindigkeit eines Moleküls wird also zum Einen durch seine Masse und zum Anderen durch seine räumliche Struktur bestimmt. So können unterschiedliche Moleküle bei gleicher Masse häufig trotzdem unterschieden werden. Nach kurzer Zeit stellt sich eine spezifische *mittlere Driftgeschwindigkeit* v (in cm/s) ein, die durch

$$v = K \cdot E \tag{2.1}$$

beschrieben werden kann [3, 30]. Die Konstante K (in $\mathrm{cm^2V^{-1}s^{-1}}$) wird dabei als *Mobilität* oder Mobilitätskoeffizient bezeichnet, E (in V/cm) ist die Stärke des elektrischen Feldes. Ist das elektrische Feld homogen, so lässt sich K mittels

$$K = \frac{L}{t \cdot E}, \tag{2.2}$$

berechnen [3, 30]. L (in cm) gibt die Länge des Driftraums, t (in s) die *Driftzeit* (die Zeit, die ein Molekül zur Durchquerung des Driftraums benötigt) an.

Da die Moleküldichte durch die Temperatur T (in K) und den Umgebungsdruck p (in kPa) beeinflusst wird, ist die Driftzeit auch von diesen Variablen abhängig [30]. Temperatur und Druck variieren von Messung zu Messung, jedoch lässt sich durch eine Normierung auf $T_0 = 273.15$ K und $p_0 = 101.325$ kPa [1] mittels

$$K_0 = K \cdot \frac{T_0}{T} \cdot \frac{p}{p_0} \tag{2.3}$$

die temperatur- und druckunabhängige *reduzierte Mobilität* K_0 (in $\mathrm{cm^2V^{-1}s^{-1}}$) berechnen [30]. K_0 kann als ein spezifisches Merkmal von Ionen aufgefasst werden, das die Detektion bestimmter Analyte ermöglicht.

Der Kern des hier betrachteten Analsysesystems ist das *Ionenmobilitätsspektrometer* (IMS). Neben dem Driftraum stellt der vorgeschaltete *Ionisationsraum* die wichtigste Komponente des IMS dar.

Die Analyte werden zunächst mit dem Trägergas über einen Einlass in den Ionisationsraum geführt. Hier findet mittels einer geeigneten *Ionisationsquelle* eine Ionisation der Moleküle statt. Auf diesen Vorgang wird im nächsten Abschnitt näher eingegangen. In den Abschnitten 2.3 und 2.4 wird nachfolgend der weitere Analyseverlauf beschrieben.

2.2. Ionisation

Das folgende Beispiel erläutert die Ionisation mittels einer radioaktiven ^{63}Ni-Strahlungsquelle in Stickstoff. Das Isotop ^{63}Ni sendet β-Teilchen mit einer begrenzten Energie aus. Treffen β-Teilchen auf Stickstoffmoleküle, geben sie Energie an diese ab und es wird ein Elektron freigesetzt [34]:

$$N_2 + \beta \rightarrow N_2^+ + \beta' + e^-. \tag{2.4}$$

Enthalten Träger- oder Driftgas Wassermoleküle (Feuchtigkeit), so leiten die Stickstoffmoleküle verschiedene Reaktionsketten ein [34]:

- Über die Reaktionen

$$\begin{aligned} N_2^+ + 2N_2 &\rightarrow N_4^+ + N_2 \\ N_4^+ + H_2O &\rightarrow 2N_2 + H_2O^+ \\ H_2O^+ + H_2O &\rightarrow H_3O^+ + OH \\ H_3O^+ + H_2O + N_2 &\rightarrow (H_2O)_2\, H^+ + N_2 \\ (H_2O)_2\, H^+ + H_2O + N_2 &\rightarrow (H_2O)_3\, H^+ + N_2 \end{aligned}$$

  entstehen $(H_2O)_x\, H^+$ Ionen mit $x \in \{0, \ldots, 7\}$.

- Bei der Beteiligung von Luft als Träger- oder Driftgas entstehen durch

$$\begin{aligned} N_2^+ + O_2 &\rightarrow O_2^+ + N_2 \\ O_2^+ + N_2 &\rightarrow NO^+ + NO \\ NO + N_2^+ &\rightarrow NO^+ + N_2 \\ NO^+ + H_2O + N_2 &\rightarrow (H_2O)\, NO^+ + N_2 \end{aligned}$$

  zusätzlich $(H_2O)_y\, NO^+$ Ionen mit $y \in \{0, 1\}$.

- Sind Spuren von Ammoniak vorhanden (beispielsweise durch Verunreinigungen), so übertragen die $(H_2O)_x\, H^+$ Moleküle Ionen zum Ammoniak (Protonentransfer) und es entstehen durch die Reaktionen

$$\begin{aligned} NH_3 + (H_2O)_3\, H^+ &\rightarrow NH_4^+ + 3H_2O \\ NH_4^+ + H_2O + N_2 &\rightarrow (H_2O)_2\, NH_4^+ + N_2 \end{aligned}$$

  Ionen der Form $(H_2O)_z\, NH_4^+$ mit $z \in \{0, 1\}$.

Die Erzeugung negativer Ionen erfolgt simultan, wobei die ursprünglich hochenergetischen β-Teilchen infolge fortwährender Stöße mit den Trägergasmolekülen laufend energieärmer und schließlich thermisch werden [34]:

- Wird Stickstoff als Träger- und Driftgas verwendet, sind thermische Elektronen e^- wesentliche Ladungsträger.

- Die Verwendung von Stickstoff als Driftgas und Luft als Trägergas hat die Erzeugung negativer O_2^- und $O\left(H_2O\right)_2^-$ Ionen zur Folge.
- Bei dem Einsatz von Luft als Träger- und Driftgas werden hauptsächlich $\left(H_2O\right)_n O_2^-$ Ionen gebildet, wobei n von der Feuchtigkeit der Luft abhängt. Liegt keine Luftfeuchte vor, so werden O_2^- Ionen gebildet (Elektronenanlagerung).

Die wichtigsten Ladungsträger stellen $\left(H_2O\right)_x H^+$ (positiv) und O_2^- (negativ) dar, die als *Reaktionsionen* bezeichnet werden [34]. Sie sind in aufgezeichneten Spektren oder IMS-Chromatogrammen deutlich als *Reaktionsionenpeak* (RIP) zu erkennen. Ihre Aufgabe besteht darin, Analyte entsprechend der jeweils gewählten Polarität zu ionisieren. Die Ionisation durch positive Reaktionsionen kann in fünf Reaktionen gegliedert werden:

Protonentransfer tritt auf, falls die Protonenaffinität eines Analyten A größer als die der Reaktionsionen ist, wobei diese Bedingung für die meisten funktionalen Gruppen (z.B. Alkohole, Aldehyde, Ketone, Carbonsäuren, Ester, Thiole, Sulfide, Nitrile und Armine) erfüllt ist [34]:

$$\left(H_2O\right)_x H^+ + A \rightarrow AH^+ + xH_2O.$$

Ladungstransfer tritt hauptsächlich zwischen $\left(H_2O\right)_y NO^+$ und Benzolderivanten A auf [34]:

$$\left(H_2O\right)_y NO^+ + A \rightarrow A^+ + yH_2O + NO.$$

Nukleophile Anlagerung mit A als Nukleophil wird beschrieben durch [34]:

$$\left(H_2O\right)_y NO^+ + A \rightarrow ANO^+ + yH_2O.$$

Hydrid- oder Hydroxid-Abstraktion wird beschrieben durch [34]:

$$\left(H_2O\right)_y NO^+ + C_nH_{2n+2} \rightarrow \left(C_nH_{2n+2}\right)^+ + HNO + yH_2O.$$

Dimerbildung kann bei höheren Konzentrationen eines Analyten A auftreten [30]:

$$AH^+ + A \rightarrow A_2H^+.$$

Weisen Analyte AB eine hohe Elektronenaffinität auf und stehen Ionen der Art $\left(H_2O\right)_n O_2^-$ bzw. thermischen Elektronen e^- zur Verfügung, so können auf drei Arten negative Ionen gebildet werden [34]:

Assoziativer Elektroneneinfang $AB + e^- \rightarrow AB^-$

Dissoziativer Elektroneneinfang $AB + e^- \rightarrow A + B^-$Durch diese Reaktion werden halogenierte Benzole ionisiert. Falls mehrere Halogen-Substituenten in Frage kommen, reagiert nur das Halogen mit der schwächsten Bindung zum Kohlenstoffatom. Dabei gilt die Reihenfolge $I > Br > Cl$.

Protonenabstraktion $ABH + \left(H_2O\right)_n O_2^- \rightarrow AB^- + (n+1)\, H_2O$

Ein anderes Beispiel für eine Ionisationsquelle ist UV-Licht. Ein Molekül A kann durch Bestrahlung mittels UV-Licht ionisiert werden, falls die Energie hv der von der UV-Quelle ausstrahlenden Lichtquadranten größer ist als das Ionisationspotential von A [34]:

$$A + hv \rightarrow A^+ + e^- \tag{2.5}$$

Die Moleküle AB können durch UV-Licht in einen angeregten Zustand versetzt werden, haben dann jedoch mehrere Möglichkeiten zu reagieren, so dass sie nicht notwendigerweise auch ionisiert werden [34]. Übliche maximale Photoenergien für UV-Quellen sind 9.5, 10.2, und 11.7 eV [20]. Der Unterschied zu einer radioaktiven Ionisationsquelle besteht hauptsächlich darin, dass die Ionisation von Analyten nicht mit Hilfe von Reaktionsionen verläuft und meist nur positive Ionen gebildet werden.

2.3. Driftraum

Der Driftraum ist eine Röhre, die direkt an den Ionisationsraum angeschlossen ist. Sie ist oft aus sich abwechselnden Schichten von Metall- und Isolatorringen (beispielsweise Glas, Keramik oder Teflon) aufgebaut [34]. An dem dem Ionisationsraum entgegen liegenden Ende befindet sich die *Faraday-Platte*. Sie erzeugt im Driftraum ein elektrisches Feld, dessen Polarität darüber entscheidet, ob positive oder negative Ionen von ihr angezogen werden (positiver oder negativer Betrieb). Die Röhre ist so konstruiert, dass das elektrische Feld im Driftraum möglichst homogen ist. Aus Richtung der Faraday-Platte strömt außerdem durch einen eigenen Einlass das Driftgas durch den Driftraum in den Ionisationsraum.

Ein ständiger Fluss nachströmender Trägergasmoleküle bringt die ionisierten Moleküle (Ionen) in den Einflussbereich des elektrischen Feldes. Die angezogenen Ionen müssen jedoch zuerst das sich periodisch öffnende *Schaltgitter* passieren. Es existieren zwei Varianten des Schaltgitters [30, 34]. Bei dem Bradbury-Nielsen-Gitter sind Gitterdrähte auf einer Ebene angeordnet, während das Tyndall-System aus zwei Gittern mit parallelen Drähten besteht, die im Abstand von einem Millimeter angeordnet sind. In beiden Fällen ist das Schaltgitter geschlossen, wenn zwischen den Gitterdrähten ein elektrisches Feld aufgebaut ist. Dieses ist senkrecht zu dem Feld des Driftraums gerichtet und verhindert ein Eindringen von Ionen in den Driftraum. Während einer kurzen Gitteröffnungszeit (in der Regel zwischen 10 μs und 1 ms [30]) dringen Ionen in den Driftraum und bewegen sich abhängig von ihrer Masse und räumlichen Struktur unterschiedlich schnell auf die Faraday-Platte zu. Ein längerer Zeitraum, in dem das Schaltgitter geschlossen ist, soll garantieren, dass erst alle Ionen den Driftraum durchquert haben, bevor das Schaltgitter erneut öffnet. Das entgegen strömende Driftgas bremst nicht nur die Ionen ab, sondern hindert außerdem neutrale Moleküle (die ja vom elektrischen Feld nicht angezogen werden) am Eintritt in den Driftraum bzw. trägt diese wieder aus ihm hinaus. Über einen Auslass am Ionisationsraum können Moleküle, die nicht in den Driftraum gelangen, aus dem IMS entweichen.

In der Faraday-Platte wird ein Strom erzeugt, der normalerweise von allen Ionen im Driftraum abhängig wäre. Aus diesem Grund befindet sich im Abstand von etwa 1 mm vor ihr das so genannte *Aperturgitter* [20]. Das Aperturgitter sorgt dafür, dass nur Ionen in dem Raum zwischen ihm und der Faraday-Platte die Stromstärke bestimmen. Ist der Abstand klein genug gewählt, befinden sich nur Ionen mit annähernd gleichen Driftzeiten in diesem Raum. Die erzeugte Stromstärke entspricht dann in etwa der Ladungssumme eintreffender Ionen, deren Ladung beim Kontakt mit der Faraday-Platte von dieser aufgenommen wird.

Der erzeugte Strom im nA bis pA-Bereich wird durch einen Verstärker geleitet, wo er üblicher Weise in eine Spannung mit einem typischen Wert von 0 bis 10 V umgewandelt wird [9]. Dieses Signal wird an eine Datenerfassungskarte geleitet, die als Analog-Digital-Wandler eine Aufzeichnung abgetasteter Signalwerte durch ein Computersystem ermöglicht.

Die Abbildung 2.1 stellt den Aufbau und die Wirkungsweise eines IMS schematisch dar. Es gibt auch erweiterte bzw. abgeänderte Modelle des IMS. Dies sind etwa IMS mit zwei Drifträumen zum parallelen Messen negativer und positiver Ionen, multiple IMS mit hintereinander geschalteten Drifträumen und FT-IMS (sie öffnen das Schaltgitter bereits, wenn noch

nicht alle Ionen den Driftraum durchquert haben) [8]. Hier wird jedoch nicht weiter darauf eingegangen.

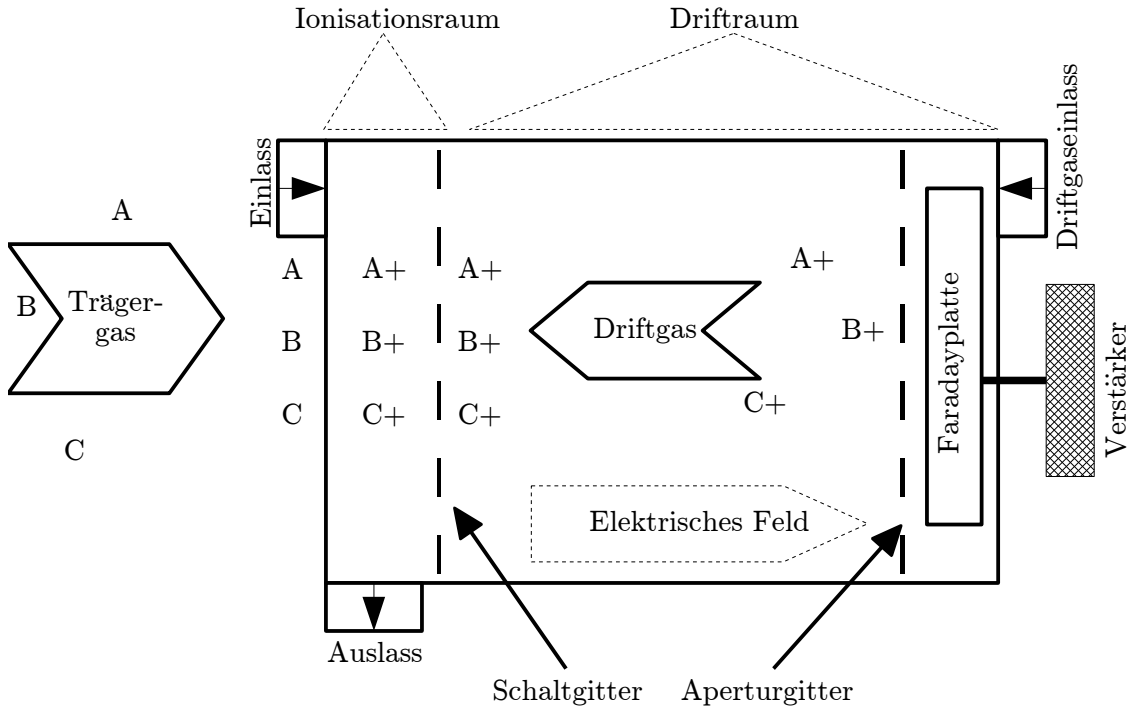

Abbildung 2.1.: Schematische Darstellung des Aufbaus und der Wirkungsweise eines IMS: A, B, C sollen neutrale Moleküle, A+, B+, C+ (positiv) geladene Ionen symbolisieren (nach [30, 34])

2.4. Vortrennung

Aufgezeichnete IMS-Daten können als abgetastete Zeitsignale aufgefasst werden. Die Daten zwischen zwei Schaltgitteröffnungen bilden dann ein *Spektrum*. Häufungen von Driftzeiten in diesen Spektren führen zu Peaks, die sich mit Hilfe der K_0-Transformation (Gleichung 2.3 auf Seite 3) mit Peaks aus Spektren anderer Messungen vergleichen lassen.

Die Interpretation der Daten wird jedoch durch die Überlappung von Peaks unterschiedlicher Ionen erschwert. Ein noch viel gravierenderes Problem stellen Molekülreaktionen während der Ionisation oder im Driftraum dar. In der Regel wünscht man sich, dass einer bestimmten Substanz spezifische Ionen zugeordnet werden können. Wird die Ionenbildung jedoch stark durch andere Substanzen beeinflusst, so ist dies fast nicht zu realisieren. Es bietet sich also an, einen zweistufigen Trennungsprozess zu durchlaufen, dessen Ziel darin besteht, verschiedene Analyte auch erst zu unterschiedlichen Zeitpunkten in das IMS gelangen zu lassen.

Eine Möglichkeit der substanzbezogen Vortrennung, besteht in der Verwendung einer *Polymermembran*. Dazu werden die Moleküle an der Membranoberfläche absorbiert, durchlaufen die Membran abhängig von ihrer Permeabilität in unterschiedlichen Geschwindigkeiten und werden nach dem Austreten an der Membranoberfläche verdampft [30].

Eine andere Möglichkeit ist die *Gas-Chromatographie* (GC). Es handelt sich um ein physikalisch-chemisches Trennverfahren, das die Verteilungseigenschaften von Molekülen bezüglich

ihrer nicht mischbaren (flüssigen oder festen) stationären Phase und (gasförmigen) mobilen Phase nutzt [34].

Ein Beispiel der GC stellt die in der früheren UdSSR entwickelte *Multikapillarsäule* (MCC) dar [30]. Etwa 1000 gleichartige Kapillare mit einem Innendurchmesser von je 43 μm und einer Länge von bis zu 30 cm (stabförmige Anordnung) oder bis zu 1 m (spiralförmige Anordnung) bilden eine Glassäule [34]. Multikapillarsäulen ermöglichen eine relativ schnelle Vortrennung und können sowohl bei Raumtemperatur als auch bis zu einer Temperaturen von 250 °C betrieben werden [34]. Die Säulentemperatur einen starken Einfluss auf die Durchlaufgeschwindigkeiten der Moleküle [30]. Dazu wurde der Vorschlag unterbreitet mit optimalen Parameter $k_T = 0.0206$, $k_q = 10^{-4}$ und der Temperaturabweichung ΔT mittels

$$t_r^{30} = t_r \cdot (1 + k_T \cdot \Delta T) + (t_r)^2 \cdot \Delta T \cdot k_q \tag{2.6}$$

eine Normierung der Austrittszeiten auf eine Säulentemperatur von 30°C vorzunehmen [1].

Wird eine Vortrennungseinheit vor das IMS geschaltet, werden pro Messung mehrere Spektren relevant. Der zeitliche Aufzeichungsstartpunkt eines Spektrums wird als *Retentionszeit* bezeichnet. In der Abbildung 2.2 sind ein kompletter Datensatz mit Vortrennung und ein einzelnes Spektrum dargestellt. Die Retentionszeit stellt eine zusätzliche Information dar. Neben charakteristischen reduzierten Mobilitäten können so auch zusätzlich charakteristische Retentionszeiten in Betracht gezogen werden.

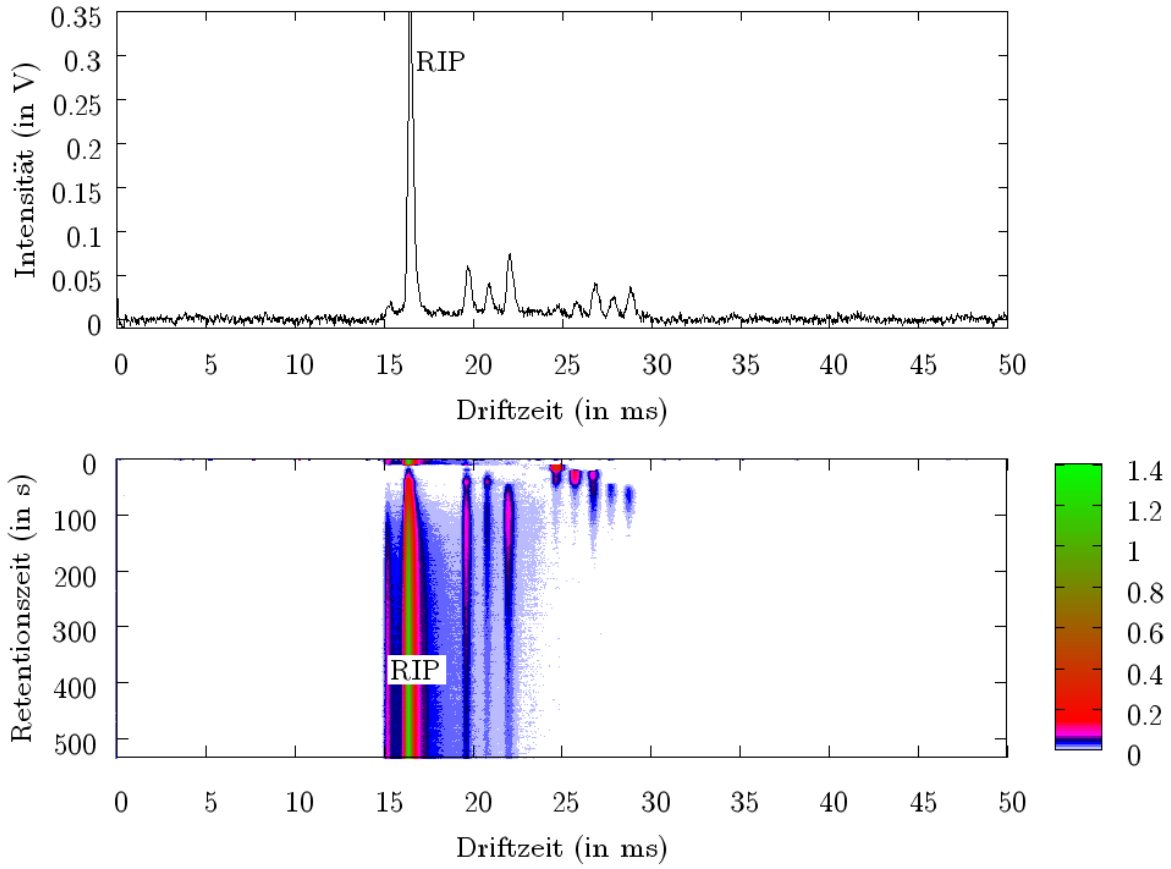

Abbildung 2.2.: Visualisierung eines einzelnen Spektrums (oben) und einer gesamten Messung mit Vortrennung (unten)

3. Glättung von Signalen

Aufgezeichnete Signale, wie beispielsweise Spektren, sind in der Regel durch Störeinflüsse verfälscht. Eine Möglichkeit, die ursprünglichen Signale approximativ zu rekonstruieren, besteht in der Glättung der Daten. In diesem Kapitel werden einige Verfahren vorgestellt, die zur Glättung eingesetzt werden können. Diese Verfahren stammen aus den Bereichen Bildverarbeitung, Zeitsignalanalyse so wie Funktionsapproximation und lassen sich auf Grund der Struktur der Daten auf diese anwenden.

3.1. Mittelwertfilter

Ein Teilbereich der digitalen Bildverarbeitung ist die Restaurierung verrauschter Bilddaten. Das Rauschen resultiert meist aus analogen Störeinflüssen bei der Datenaufzeichnung. Für gewöhnlich liegen die Bilddaten in Form einer $n \times m$ Matrix $\mathbf{A}$

$$\begin{pmatrix} x_{1,1} & \cdots & x_{n,1} \\ \vdots & & \vdots \\ x_{1,m} & \cdots & x_{n,m} \end{pmatrix}$$

vor. Im Folgenden wird davon ausgegangen, dass die $x_{i,j}$ Intensitäten auf einer Graustufenskala $[g_{min}, g_{max}]$ darstellen. Häufig wird g_{min} auf Null gesetzt und als Schwarz aufgefasst, während g_{max} auf Eins gesetzt und als Weiß interpretiert wird.

Angenommen $\tilde{x} \in [g_{min}, g_{max}]$ sei ein fehlerfreier Punkt eines gegebenen Bildes. Fehlerfrei soll dabei bedeuten, dass das Bild ohne Störeinflüsse aufgezeichnet worden wäre. Dann wirkt sich das Rauschen folgendermaßen auf den tatsächlich vorliegenden Punkt x aus:

$$x = \begin{cases} \min(\tilde{x} + E, g_{\max}) & \text{falls} \quad \mathrm{E} > 0 \\ \max(\tilde{x} + E, g_{\max}) & \text{sonst} \end{cases}$$

Der Fehlerwert E wird als einer bestimmten Wahrscheinlichkeitsverteilung unterworfen angesehen. Neben der Normalverteilung gibt es in der digitalen Bildverarbeitung für diese Verteilung auch speziellere Modelle [26].

Das *Mittelwertfilter* ist ein Werkzeug, dessen Aufgabe darin besteht, das Rauschen in Bilddaten zu mindern. Trotz seiner Einfachheit ist das Mittelwertfilter im Zeitbereich[1] sehr effektiv bezüglich der Reduktion von weißem Rauschen bei hoher Kantenschärfe [35].

Die grundsätzliche Idee ist folgende: Jeder Bildpunkt wird durch den Mittelwert eines rechteckigen Fensters (Ausschnitt) der Matrix neu berechnet. Die Länge des Rechtecks sei w und die Höhe h. Normalerweise bildet ein betrachteter Punkt $x_{s,t}$ den Mittelpunkt des Fensters bezüglich jeder der beiden Dimensionen. Dann fließen die Punkte $x_{i,j}$ mit $s - a \leq i \leq s + a$ und $t - b \leq j \leq t + b$ mit $a := \lfloor w/2 \rfloor$ und $b := \lfloor w/2 \rfloor$ in die Berechnung ein[2].

[1] Der Zeitbereich ist der untransformierte Bereich der Bilddaten, vgl. Abschnitt 3.2.

[2] Natürlich ergeben sich daraus für Randpunkte des Bildes Probleme. In Abschnitt 3.3.5 werden einge Möglichkeiten aufgezählt, diesem Problem zu begenen.

Der Mittelwert $y_{s,t}$ bezüglich dieser Fensterpunkte wird durch

$$y_{s,t} = \frac{1}{w \cdot h} \sum_{i=s-a}^{s+a} \sum_{j=t-b}^{t+b} x_{i,j} \tag{3.1}$$

berechnet. Sind alle Mittelwerte berechnet, werden die Matrixwerte durch diese ersetzt. Falls die $y_{s,t}$ systematisch (beispielsweise von links nach rechts, nächste Reihe, von rechts nach links, usw.) erzeugt werden, ist eine effiziente Berechnung möglich, da immer nur einzelne Elemente aus der Summe heraus fallen und durch andere Elemete ersetzt werden.

Eine Abwandlung des Mittelwertfilters ist der *Medianfilter*. Hierbei entspricht $y_{s,t}$ nicht dem Mittelwert, sondern dem Median. Die Vorteile sind eine bessere Rauschreduktion und die Vermeidung der Bildung neuer Intensitäten [37]. Denn der Mittelwertsfilter erzeugt auch Intensitäten, die nicht in dem zu filternden Bild vorhanden sind. Dies kann unerwünscht sein.

Der Berechnungsaufwand beim Medianfilter ist jedoch sehr viel höher als der beim Mittelwertsfilter. Für jeden Punkt müssen alle Fensterpunkte zur Berechnung des Medians neu sortiert werden. Einen Ausweg bietet die Nutzung der so genannten Histogrammmethode, die ohne ständige Neusortierungen auskommt und bei großen Fenstern den Berechnungsaufwand erheblich reduziert [37]. Eine andere Möglichkeit den Berechnungsaufwand für den Medianfilter zu reduzieren besteht besteht darin, nur einen Teil des Fensters zu berechnen. Ein Beispiel ist die Kreuzform [26]: Bei dem Fenstermittelpunkt $x_{s,t}$ werden nur die Punkte $x_{i,t}$ mit $s - a \leq i \leq s + a$ und $x_{s,j}$ mit $t - b \leq j \leq t + b$ betrachtet. Neben der Verringerung des Berechnungsaufwandes kann durch geschickte Wahl der Fensterform auch eine Berücksichtigung bestimmter Strukturen des zu analysierenden Bildes erzielt werden [37].

Eine weitere Abwandlung des Mittelwertfilters besteht darin, die Mittelwerte gewichtetet zu berechnen. Es gibt mehrere Möglichkeiten, diese Gewichtung zu realisieren. Eine Möglichkeit ist die so genannte *Multi-Pass-Methode*. Dazu wird das Mittelwertfilter mehrfach auf die Matrix $\mathbf{A}$ angwendet. Nach vier oder mehr Durchläufen ergibt sich eine Gausskurven-ähnliche Gewichtung [35]. Eine weitere Möglichkeit besteht darin, explizit *Gewichtskoeffizienten* anzugeben. Dazu wird das Fenster durch eine zweite Matrix $\mathbf{W}$ repräsentiert. Diese Matrix $\mathbf{W}$ enthält die Gewichtskoeffizienten $w_{i,j}$ mit $1 \leq i \leq w$ und $1 \leq j \leq h$, wobei in der Regel kein Gewichtskoeffizient größer als $w_{a+1,b+1}$ gewählt wird. Die gewichteten Mittelwerte $y_{s,t}$ werden dann durch

$$y_{s,t} = \frac{1}{Q} \sum_{i=s-a}^{s+a} \sum_{j=t-b}^{t+b} x_{i,j} w_{i+a-1,j+b-1} \quad \text{mit} \quad Q := \sum_{i=1}^{w} \sum_{j=1}^{h} w_{i,j} \tag{3.2}$$

berechnet [26].

3.2. Fourier-Transformation

Ein eindimensionales reellwertiges Signal definiert eine Funktion $f : \mathbb{R} \to \mathbb{R}, \quad t \mapsto y$, wobei die unabhängige t-Variable oft mit einem Zeitmaß (Sekunden, Minuten etc.) assoziiert wird und y einen abhängigen Funktionswert (Signalstärke, Intensität, etc.) darstellt. Ein zweidimensionales reellwertiges Signal führt entsprechend zu einer Funktion $f : \mathbb{R}^2 \to \mathbb{R}, \quad (x, y) \mapsto z$. Dieses Signal kann beispelsweise wie im Abschnitt 3.1 ein Bild darstellen, dann sind x und y die Koordinaten eines Bildpunktes und z seine Intensität[3]. Diese Darstellungsform von Signalen bzw. Funktionen (im Folgenden werden diese Begriffe analog verwendet) wird als

[3] Liegt ein farbiges Bild im RGB-Farbmodell vor, so kann man beispielsweise jede Farbkomponente isoliert betrachtet werden.

Zeitbereich bezeichnet. Der *Frequenzbereich* eines Signals beschreibt dagegen, welche Frequenz im Signal mit welcher Intensität vorhanden ist. Die *Fourier-Transformation* (FT) transformiert Funktionen aus dem Zeitbereich in den Frequenzbereich. Sie beruht auf der von Jean Baptiste Joseph Fourier formulierten Annahme, dass jedes kontinuierliche periodische Signal als die Summe von sinusförmigen Wellen, also durch eine Überlagerung von Sinus- und Kosinusfunktionen, dargestellt werden kann [35].

Die FT arbeitet mit Funktionswerten aus dem komplexen Zahlenraum. dies stellt jedoch keine Einschränkung dar, da eine reellwertige Zahl y auf eine komplexe Zahl $y + i \cdot 0$ (also mit einem Imaginäranteil von Null) abgebildet werden kann. Dabei ist i im Verlauf dieses Abschnitts als die bekannte Konstante $i = \sqrt{-1}$ definiert.

In Abschnitt 3.2.1 wird die kontinuierliche FT eingeführt. Abschnitt 3.2.2 behandelt die diskrete FT. Abschließend wird in Abschnitt 3.2.3 kurz auf eine Erweiterung der FT eingegangen, die auch Informationen aus dem Zeitbereich mit einbezieht.

3.2.1. Kontinuierliche Fourier-Transformation

Der Frequenzbereich wird bei der FT durch Funktionen der Form

$$b_\omega : \mathbb{R} \to \mathbb{C}, \quad t \mapsto e^{i2\pi\omega t} = \cos(2\pi\omega t) + i \cdot \sin(2\pi\omega t)$$

beschrieben. Diese Funktionen können als *Basisfunktionen* bezeichnet werden. Mit ihrer Hilfe wird die folgende kontinuierliche Transformation definiert [6].

Definition 3.1 (kontinuierliche Fourier-Transformation) *Die kontinuierliche Fourier-Transformation (Continuous Fourier Transform, CFT) einer Funktion $f : \mathbb{R} \to \mathbb{C}$ ist als*

$$\hat{f}(\omega) := \int_{-\infty}^{\infty} f(t) \cdot e^{-i2\pi\omega t} dt \tag{3.3}$$

definiert. Sie berechnet zu jeder Frequenz $\omega \in \mathbb{R}$ einen Koeffizienten $c_\omega \in \mathbb{C}$.

Die Funktion $\hat{f}$ wird als *Fouriertransformierte* oder *Spektralfunktion* von f bezeichnet [4]. Sie liefert die Information, welche Frequenzen in f vorkommen und wie stark diese ausgeprägt sind. Diese Informationen können ausgewertet und verändert werden. Beispielsweise kann der Einsatz digitaler Hoch-, Tief- und Bandpassfilter vollständig im Frequenzebereich erfolgen. Die Berechnung der Zeitbereichsdarstellung einer Funktion mit Hilfe der Koeffizienten c_ω erfolgt durch die inverse Transformation in der folgenden Definition [6].

Definition 3.2 (inverse kontinuierliche Fourier-Transformation) *Die inverse kontinuierliche Fourier-Transformation (ICFT) einer Spektralfunktion $\hat{f} : \mathbb{R} \to \mathbb{C}$ ist als*

$$f(t) := \int_{-\infty}^{\infty} \hat{f}(\omega) \cdot e^{i2\pi\omega t} d\omega \tag{3.4}$$

definiert.

Gilt $f \in L^1(\mathbb{R})$[4], so ist dies eine hinreichende aber nicht notwendige Bedingung zur Anwendbarkeit der CFT und der ICFT auf f [6], die aber für endliche Signale mit endlicher

[4] Der $L^1(\mathbb{R})$ ist ein Hilbertraum, der alle absolut einfach integrierbaren Funktionen enthält [12]:

$$f \in L^1(\mathbb{R}) \Leftrightarrow \int_{-\infty}^{\infty} |f(x)| \, dx < \infty$$

Signalhöhe stets erfüllt ist. Der *Faltungssatz* gilt für je zwei Funktionen f, g, deren Spektralfunktionen $\hat{f}, \hat{g}$ existieren. Dieser stellt eine Beziehung zwischer ihrer Faltung

$$f * g\,(t) := \int_{-\infty}^{\infty} f\,(s)\, g\,(t-s)\;\, ds$$

und den Spektralfunktionen durch

$$\widehat{(f * g)} \Leftrightarrow \hat{f} \cdot \hat{g} \tag{3.5}$$

her [27]. Die Faltung zweier Zeitbereichsfunktionen entspricht also dem Produkt ihrer Spektralfunktionen.

Leider sind CFT und ICFT in der Literatur nicht einheitlich definiert. Eine allgemeine Möglichkeit der Defnition des Transformationspaares ist durch

$$\hat{f}\,(\omega) := a_1 \int_{-\infty}^{\infty} f\,(t) \cdot e^{-i\alpha t} dt \qquad f\,(t) := a_2 \int_{-\infty}^{\infty} \hat{f}\,(\omega) \cdot e^{i\alpha t} d\omega$$

gegeben, wobei $\alpha = 2\pi\omega$ und die Vorfaktoren die Bedingung $a_1 \cdot a_2 = 1/\,(2\pi)$ erfüllen müssen [6]. Darüber hinaus ist eine Vertauschung der Vorzeichen in den Basisfunktionstermen möglich [27].

Der Übergang zur zweidimensionalen FT wird durch eine Quadrierung des Bildbereichs der FT bewerkstelligt. Es werden also Frequenzen in beide Dimensionsrichtungen analysiert. Die entsprechenden Formeln sind durch

$$\hat{f}\,(u,v) \;\; := \;\; \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f\,(x,y) \cdot e^{-i2\pi(ux+vy)} dx\; dy \tag{3.6}$$

$$f\,(x,y) \;\; := \;\; \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \hat{f}\,(u,v) \cdot e^{i2\pi(ux+vy)} du\; dv \tag{3.7}$$

gegeben [6]. Dabei verfügen die zweidimensionalen Basisfunktionen $e^{i2\pi(ux+vy)}$ über die Sinus - und Kosinuskomponenten $\cos\,(2\pi\,(ux+vy))$ und $\sin\,(2\pi\,(ux+vy))$. Die analytische Berechnung der Gleichung 3.6 kann auf Grund der Separierbarkeit auch durch die sukessive Berechung der beiden eindimensionalen Integrale mittels

$$\hat{f}\,(u,v) := \int_{-\infty}^{\infty} e^{-i2\pi vy} \left(\int_{-\infty}^{\infty} f\,(x,y) \cdot e^{-i2\pi ux} dx \right) dy \tag{3.8}$$

erfolgen, die Gleichung 3.7 kann analog berechnet werden.

3.2.2. Diskrete Fourier-Transformation

In der Praxis werden Signale nur in abgetasteter Form vorliegen. Im Folgenden wird sich auf eine konstante Abtastrate Δ beschränkt. Die abgetastete Funktion kann dann durch eine Kette von Funktionswerten $f_n = f\,(\Delta n)$ repräsentiert werden.

Das *Sampling Theorem* besagt, dass f durch die f_n vollständig definiert ist, falls das Signal durch $f_c = 1/\,(2\Delta)$ bandbreitenbeschränkt bezüglicher seiner anteiligen Frequenzbeträge ist [6, 27]: $\hat{f}\,(\omega) = 0$ für alle $|\omega| \geq f_c$. Dann lässt sich f in der Form

$$f(t) = \Delta \sum_{n=-\infty}^{\infty} f_n \frac{\sin\,(2\pi f_c\,(t - n\Delta))}{\pi\,(t - n\Delta)} \tag{3.9}$$

notieren [6, 27].

Die FT muss nur noch für die $\omega \in [-f_c, f_c]$ durchgeführt werden. In der Realität ist darüber hinaus jede Abtastpunktmenge endlich. Es gibt also nur m viele Funktionswerte $f_0, \ldots, f_{m-1}$. Somit kann die Spektralfunktion auch nur durch m Punkte approximiert werden [6, 27].

Die Diskretisierung besteht nun darin, die Integrale in den Gleichungen 3.3 und 3.4 auf Seite 11 durch Summen zu approximieren [27]. Es ergeben sich die Transformationen der folgenden beiden Definitionen [6]:

Definition 3.3 (diskrete Fourier-Transformation) *Die diskrete Fourier-Transformation (DFT) einer an den Punkten $f_n = f(\Delta n)$ abgetasteten Funktion $f : \mathbb{R} \to \mathbb{C}$ ist als*

$$\hat{f}\left(\frac{n}{m\Delta}\right) := \sum_{k=0}^{m-1} f_k \cdot e^{-i2\pi nk/m} \quad n = 0, \ldots, m-1 \tag{3.10}$$

definiert.

Definition 3.4 (inverse diskrete Fourier-Transformation) *Die inverse diskrete Fourier-Transformation (IDFT) zu einer abgetasteten Spektralfunktion $\hat{f}$ mit dem Ziel, eine abgetastete Funktion im Zeitbereich wie in Definition 3.3 zu erzeugen, ist als*

$$f_n := \frac{1}{m} \cdot \sum_{k=0}^{m-1} \hat{f}\left(\frac{k}{m\Delta}\right) \cdot e^{i2\pi nk/m} \quad n = 0, \ldots, m-1 \tag{3.11}$$

definiert.

Die DFT und die IDFT können leicht auf Computern implementiert werden. Doch die diskrete Variante bietet einen weiteren Vorteil: Sie ermöglicht die *schnelle Fourier-Transformation* (Fast Fourier Transform, FFT).

Bis zur Einführung der FFT wurde der Berechnungsaufwand mit $O\left(m^2\right)$ bemessen, durch die FFT sinkt der Aufwand auf $O\left(m \log_2 m\right)$ [27]. Dabei wird eine gerade Anzahl von Punkten in gerade und ungerade Indizes aufgeteilt und die DFT aus den beiden Teilmengen-DFTs berechnet [6, 23, 27]. Wenn man eine komplexe Konstante $W := e^{-i2\pi/m}$ definiert, so kann die DFT in der Gleichung 3.10 auch durch

$$\begin{aligned}
\hat{f}\left(\tfrac{n}{m\Delta}\right) \quad =: \quad \hat{f}_n \quad &= \sum_{k=0}^{m-1} f_k \cdot e^{-i2\pi nk/m} \\
&= \sum_{k=0}^{m/2-1} f_{2k} \cdot e^{-i2\pi n(2k)/m} \quad + \quad \sum_{k=0}^{m/2-1} f_{2k+1} \cdot e^{-i2\pi n(2k+1)/m} \\
&= \sum_{k=0}^{m/2-1} f_{2k} \cdot e^{-i2\pi nk/(m/2)} \quad + \quad W^n \sum_{k=0}^{m/2-1} f_{2k+1} \cdot e^{-i2\pi nk/(m/2)}
\end{aligned}$$

dargestellt werden [27]. Sie lässt sich also aus den DFTs der beiden Mengen bilden. Wegen $W^0 = -W^2$ [6], kann die Berechnung von $\hat{f}_n$ auch durch

$$\begin{aligned}
\hat{f}_n \quad &= \sum_{k=0}^{m/2-1} f_{2k} \cdot W^{2nk} \quad + \quad W^n \sum_{k=0}^{m/2-1} f_{2k+1} \cdot W^{2nk} \\
\hat{f}_{n+m/2} \quad &= \sum_{k=0}^{m/2-1} f_{2k} \cdot W^{2nk} \quad - \quad W^n \sum_{k=0}^{m/2-1} f_{2k+1} \cdot W^{2nk}
\end{aligned} \tag{3.12}$$

mit $n = 0, \ldots, m/2-1$ erfolgen [23]. Falls m eine Zweierpotenz ist, kann dieser Prozess rekursiv durchlaufen werden. Die Berechung der IDFT kann ebenfalls mit Hilfe der FFT erfolgen,

da $f_n = \hat{f}\left((1/m) \cdot \overline{\hat{f}_n}\right)$ [28]. Es gibt viele Varianten des FFT-Algorithmus, einige verarbeiten auch Datenvektoren beliebiger Länge. Für Details sei auf die entsprechende Literatur verwiesen [6, 12, 23, 27].

Die Berechnung der zweidimensionalen DFT kann auf Grund ihrer Separierbarkeit durch

$$\hat{f}\left(\frac{n_x}{m_x \Delta_x}, \frac{n_y}{m_y \Delta_y}\right) = \sum_{k_y=0}^{m_y-1} \left(\sum_{k_x=0}^{m_x-1} f_{k_x k_y} \cdot e^{-i2\pi n_x k_x / m_x} \right) \cdot e^{-i2\pi n_y k_y / m_y} \tag{3.13}$$

erfolgen [6]. Die zweidimensionale IDFT wird analog dazu mittels

$$f_{k_x k_y} = \frac{1}{m_y} \sum_{k_y=0}^{m_y-1} \left(\frac{1}{m_x} \sum_{k_x=0}^{m_x-1} f_{k_x k_y} \cdot e^{-i2\pi n_x k_x / m_x} \right) \cdot e^{-i2\pi n_y k_y / m_y} \tag{3.14}$$

berechnet [6]. Beide Transformationen können effizient durch FFTs berechnet werden. Eine Anwendung in der Bildverarbeitung ist die Detektion von Texturen (Seefläche, Acker, etc.) anhand ihrer charakteristischen Muster im Frequenzbereich [4]. Um diese Flächen mit Hilfe der zweidimensionalen DFT zu lokalisieren, kann das Bild dann in gleich große Blöcke aufgeteilt werden, die alle einzeln auf das gesuchte Muster untersucht werden.

3.2.3. Gefensterte Fourier-Transformation

Die *gefensterte Fourier-Transformation* (Windowed Fourier Transform, WFT) stellt eine Erweiterung der FT dar, die mittels einer *Fensterfunktion* neben spektraler auch zeitliche Information liefert. Dazu wird eine Fensterfunktion $g : \mathbb{R} \to \mathbb{R}_{\geq 0}$ fest gewählt. Diese sollte ein kompakter Träger sein oder ein ausgeprägtes Maximum bei Null besitzen (beispielsweise eine Normalverteilung mit Erwartungswert Null) [4].

Die *Fenstertransformierte* $Gf(\omega, s)$ untersucht dann eine Funktion f, indem in der FT die Substitution von $f(t)$ durch $f(t) \cdot g(t-s)$ erfolgt [4]. Das durch g definierte Fenster wird durch s verschoben und das Auftreten der zu ω korrespondierenden Basisfunktion entsprechend gewichtet erfasst.

Definition 3.5 (gefensterte Fourier-Transformation) *Die gefensterte Fourier-Transformation (Windowed Fourier Transform, WFT) einer Funktion $f : \mathbb{R} \to \mathbb{C}$ unter Verwendung einer Fensterfunktion $g : \mathbb{R} \to \mathbb{R}_{\geq 0}$ ist als*

$$Gf(\omega, s) := \int_{-\infty}^{\infty} f(t) \cdot g(t-s) \cdot e^{-i2\pi\omega t} dt \tag{3.15}$$

definiert.

Auf Grund der redundanten Information über f in Gf gibt es verschiedene inverse Transformationen [4]. Eine diskrete Version der WFT arbeitet in der Regel mit äquidistanten Unterteilungen der t- und der ω-Achse [4]. Die WFT bietet auf Grund der lokalereren Betrachtung neben der Spektralinformation auch Zeitinformation. Allerdings ist die durch g fest definierte Fenstergröße (bzw. Gewichtung) problematisch [4]: Einerseits könnten hohe Frequenzen schlecht lokalisert werden (Fenster zu breit), andererseits könnte das Fenster zu schmal sein, um nur eine einzige Vollschwingung einer niedrigeren Frequenzen zu erfassen.

3.3. Wavelet-Transformation

Die *Wavelet-Transformation* (WT) dient wie die FT einer Darstellung von Funktionen $f : \mathbb{R} \to \mathbb{C}$ mit Hilfe von Basisfunktionen. Während bei der FT sinusförmige Wellen eingesetzt werden, stellt bei der WT eine unter gewissen Randbedingungen frei wählbare Mutterfunktion den Ausgangspunkt für die Basisfunktionen dar. Diese wird als *Mutter Wavelet* [4] oder einfach *Wavelet* [4, 19] bezeichnet. Die Basisfunktionen sind verschobene und gestauchte Varianten der Mutterfunktion, auch sie werden manchmal als Wavelets bezeichnet [19]. Um zwischen Mutterfunktion und einer Basisfunktion bestimmter Ausprägung zu unterscheiden, werden hier die Bezeichnungen Wavelet und Waveletfunktion gewählt.

Ein Zeitsignal liefert die Summe aller in ihm enthaltenen Frequenzen zu einem bestimmten Zeitpunkt, seine Spektralfunktion dagegen liefert zu jeder Frequenz ihre Ausprägung über das gesamte Signal [16]. Jede der beiden Repräsentationen beinhaltet also genau eine der Informationskomponenten. Der Vorteil der WT liegt darin, dass sie sowohl Zeit- wie auch Frequenzinformation liefert, und im Gegensatz zur WFT (vgl. Abschnitt 3.2.3) eine variable Zeit-Frequenz-Auflösung ermöglicht.

Die WT findet in zahlreichen Gebieten Verwendung. Einige Beispiele sind [4, 12, 17, 39]: Filterung von Signalen (Glättung), die Kompression von Bilddaten (JPEG2000, MPEG-4, Fingerabdruckskartei des FBI), Auswertung von Bilddaten (Texturdetektion).

Im Abschnitt 3.3.1 wird zunächst die kontinuierliche WT vorgestellt. Der Abschnitt 3.3.2 behandelt einige Überlegungen zur Diskretisierung. Im Abschnitt 3.3.3 wird mit der Multi-Skalen-Analyse eine filterbasierte Umsetzung der diskreten WT vorgestellt. Der darauf folgende Abschnitt 3.3.4 stellt einen neueren Ansatz zur Berechnung einer WT vor. Die Abschnitte 3.3.5 und 3.3.6 gehen abschließend kurz auf die Behandlung von Randwertproblemen und die Technik des Thresholdings ein.

3.3.1. Kontinuierliche Wavelet-Transformation

Ein Wavelet ist eine Funktion $\psi : \mathbb{R} \to \mathbb{C}$. Sie muss folgende Bedingungen erfüllen [4]:

$$\psi \in L^2(\mathbb{R}) \tag{3.16}$$

$$\|\psi\| = 1 \tag{3.17}$$

$$0 < c_\psi := 2\pi \int_{-\infty}^{\infty} \frac{\left|\hat{\psi}(\alpha)\right|^2}{|\alpha|}\, d\alpha < \infty \tag{3.18}$$

Die Bedingung 3.16 bedeutet, dass ψ quadratisch integrierbar sein muss[5]. Gilt darüber hinaus noch $\psi \in L^1(\mathbb{R})$ (was auf alle praktisch vorkommenden Wavelets zutrifft [4]), so ist Bedingung 3.18 äquivalent zu der Anforderung $\hat{\psi}(0) = 0$ [19]. Das bedeutet, dass der

[5] Der $L^2(\mathbb{R})$ ist ein Hilbertraum, der alle quadratisch integrierbaren Funktionen beinhaltet. Zulässige Funktionen f sind dann mittels

$$f \in L^2(\mathbb{R}) \Leftrightarrow \int_{-\infty}^{\infty} |f(x)|^2\, dx < \infty$$

festgelegt [4, 12]. Norm und Skalarprodukt zweier Funktionen $f, g \in L^2(\mathbb{R})$ sind als

$$\|f\| := \sqrt{\int_{-\infty}^{\infty} |f(x)|^2\, dx} \qquad \langle f, g\rangle := \int_{-\infty}^{\infty} f(x)\, \overline{g(x)}\, dx$$

definiert [4, 12].

Mittelwert von $\psi(t)$ Null beträgt. Auf Grund der Normierungsbedingung 3.17 muss der Graph von $\psi(t)$ wellengleich teilweise oberhalb und teilweise unterhalb der t-Achse verlaufen. Damit wird der Begriff Wavelet (auf Deutsch: kleine Welle, „Wellchen“) intuitiv.

Die Strauchung eines ψ erfolgt mittels eines reellwertigen Parameters $a \neq 0$. Mit einem zusätzlichen Translationsparameters $b \in \mathbb{R}$ können die aus ψ resultierenden Waveletfunktionen definiert werden:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}} \cdot \psi\left(\frac{t-b}{a}\right) \tag{3.19}$$

Die Waveletfunktion $\psi_{a,b}$ wird vom Nullpunkt aus um $|a|$ verbreitert. Gilt $a < 0$, so wird sie an der t-Achse gespiegelt. Der Nullpunkt wird um b auf der t-Achse verschoben. Der Vorfaktor stellt die Normierungsbedingung (Gleichung 3.17 auf der vorherigen Seite) sicher [4]. Jede Waveletfunktion ist selbst also wieder ein legales Wavelet. Die folgende Definition gibt die Berechnung der *kontinuierlichen Wavelet-Transformation* an [4][6]:

Definition 3.6 (kontinuierliche Wavelet-Transformation) *Die kontinuierliche Wavelet-Transformation (Continuous Wavelet Transform, CWT) einer Funktion $f \in L^2(\mathbb{R}) : \mathbb{R} \to \mathbb{C}$ zu einem Wavelet ψ ist als*

$$Wf_\psi(a,b) := \langle f, \psi_{a,b} \rangle = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} f(t)\, \overline{\psi\left(\frac{t-b}{a}\right)}\, dt \tag{3.20}$$

definiert.

Der Ergebnisbereich ist eine Ebene mit den Achsen a und b, die einen Schnitt $\{(a,b)\,|a = 0,\ b \in \mathbb{R}\}$ aufweist. Oft wird nur die obere Halbhälfte für $a > 0$ betrachtet, wobei dann das Integral in der Gleichung 3.18 auf der vorherigen Seite anzupassen ist [4].

Je nach Beschaffenheit von ψ kann Wf_ψ als eine Art Zeit-Frequenz-Analyse von f dienen. Der Parameter b definiert einen Bezugspunkt auf der Zeitachse von f. Der Parameter a bestimmt die betrachtete „Frequenz“, d.h. a legt die Konzentration der Betrachtung auf der Zeitachse fest. Große a verkleinern den relevanten Zeitausschnitt („hohe Frequenz“), kleine a vergrößern ihn („niedrige Frequenz“). Dies kann auch als *Zoomeffekt* von a beschrieben werden [19].

Natürlich sollte das ursprüngliche Zeitsignal f auch wieder aus Wf_ψ rekonstruierbar sein. Eine naheliegende Möglichkeit besteht darin, wie im Fall der ICFT einfach den gesamten Definitionsbereich von Wf_ψ zur Rekonstruktion zu benutzen. Die folgende Definition beschreibt dann die *inverse kontinuierliche Wavelet-Transformation* [4][7].

Definition 3.7 (inverse kontinuierliche Wavelet-Transformation) *Die inverse kontinuierliche Wavelet-Transformation (ICWT) zu einem Wavelet ψ ist als*

$$f(t) = \frac{1}{c_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} Wf_\psi(a,b)\, \psi_{a,b}(t)\, \frac{da\, db}{|a|^2} \tag{3.21}$$

definiert.

[6] In der Literatur finden sich auch abweichende Definitionen ohne Konjugation [19,40]. Bei reellwertigen Wavelets hat das keine Auswirkungen, doch die konjugierte Schreibweise entspricht dem $L^2(\mathbb{R})$-Skalarprodukt.

[7] In der Literatur findet man auch eine Definition mit abweichendem Vorfaktor $1/\sqrt{c_\psi}$ statt $1/c_\psi$ [19].

3.3.2. Diskrete Wavelet-Transformation

Der Definitionsbereich von Wf_ψ ist $\mathbb{R}^{2*} := \{(a,b)\,|a \in \mathbb{R}\setminus\{0\}\,,\, b\}$ und somit (fast) quadratisch größer, als der von f. Die Betrachtung wird auf eine abzählbare Teilmenge $M := \{(a_m, b_{m,n})\,|m,n \in \mathbb{Z}\}$ der oberen Halbebene ($a > 0$) von $\mathbb{R}^{2*}$ eingeschränkt. Es werden ein Zoomschritt $\sigma > 1$ (meist $\sigma = 2$) und ein Grundschritt $\beta > 0$ festgelegt. Die M-Punkte werden durch die Substitutionen

$$a_m := \sigma^m \quad \text{und} \quad b_{m,n} := n\sigma^m\beta \tag{3.22}$$

definiert, wobei $m,n \in \mathbb{Z}$. Jeder Punkt $(a_m, b_{m,n})$ verweist auf ein Rechteck $R_{m,n}$ mit der Breite $\sigma^m\beta$ und und der Höhe $\sigma^m\sqrt{\sigma} - \sigma^m/\sqrt{\sigma}$ [4]. Die Halbebene wird durch das $R_{m,n}$-Gitter disjunkt in ein hierachisches Gitter zerlegt: Mit zunehmender „Frequenz“ ($m \to \infty$) liegen die $R_{m,n}$ dichter bei einander. Dies ist intuitiv berechtigt: Bei der Betrachtung eines Gemäldes aus der Ferne nimmt man gröbere Merkmale wie beispielsweise den goldenenen Schnitt wahr, jedoch keine feinen Einzelheiten. Tritt man sehr nah an das Gemälde heran, so sind diese gröberen Merkmale nicht mehr erfassbar, jedoch fällt vielleicht erst jetzt die Signatur des Künstlers auf.

Zu dem Gitter korrespondiert eine Familie [4, 19] $\psi_\bullet := \left\{\psi_{m,n}|\,(m,n) \in \mathbb{Z}^2\right\}$ von Waveletfunktionen

$$\begin{aligned} \psi_{m,n}(t) \;:=\; \psi_{\sigma^m, n\sigma^m\beta}(t) &= \sigma^{-\frac{m}{2}}\psi\left(\tfrac{t-n\sigma^m\beta}{\sigma^m}\right) \\ &= \sigma^{-\frac{m}{2}}\psi\left(\sigma^{-m}t - n\beta\right). \end{aligned} \tag{3.23}$$

Sie bilden zusammen mit σ, β einen *Wavelet-Frame* für $L^2(\mathbb{R})$, falls es Konstanten $A, B > 0$ gibt, so dass

$$A\,\|f\|^2 \le \sum_{m\in\mathbb{Z}}\sum_{n\in\mathbb{Z}} |\langle\psi_{m,n}, f\rangle|^2 \le B\,\|f\|^2 \tag{3.24}$$

gilt [19]. Ein Wavelet-Frame ist ein spezieller *Frame* mit dem zugehörigen *Frame-Operator* $Tf(m,n) := \langle f, \psi_{m,n}\rangle = Wf_\psi(a_m, b_{m,n})$ [4].

Ein Frame hat die besondere Eigenschaft, dass ein diskreter Satz von Punkten ausreicht, um den gesamten Raum zu berechnen. Die anderen Punkte stehen nämlich in Abhängigkeit zu diesen und sind daher redundant. Dieser Satz von Punkten ist im Fall des Wavelet-Frames durch $\psi_\bullet$ gegeben. Zu jedem Frame lässt sich dann ein *dualer Frame* $\tilde{\psi}_\bullet$ berechnen, mit dessen Hilfe im Fall des Frame-Wavelets ein Funktionswert von $f \in L^2(\mathbb{R})$ aus $\psi_\bullet$ zurückgewonnen werden kann [4,19,40]. Falls $A = B$ kann f auch ohne einen zusätzlichen dualen Frame mittels

$$f(t) = \frac{1}{A}\sum_{m\in\mathbb{Z}}\sum_{n\in\mathbb{Z}} Wf_\psi(m,n)\,\psi_{m,n}(t) \tag{3.25}$$

berechnet werden [19, 40]. Hierbei gelten wieder die in der Gleichung 3.22 definierten Substitutionen.

Diese Diskretisierung in beide Richtungen macht überhaupt erst eine Berechnung der WT auf einem Computersystem möglich. Allgemein spricht man von der *diskreten Wavelet-Transformation* (DWT). Für die Behandlung der Frage, welche Bedingungen ψ, σ und β erfüllen müssen, um einen gültigen Wavelet-Frame darzustellen, sei auf die entsprechende Literatur verwiesen [4, 16, 19].

3.3.3. Multi-Skalen-Analyse

Die *Multi-Skalen-Analyse* (MSA) erlaubt eine effiziente DWT-Berechnung. Sie geht auf S. Mallat und Y. Meyer zurück (nach [19]). Man wählt $\sigma = 2$ und $\beta = 1$ und beschränkt sich auf

Wavelets, deren Familie von Waveletfunktionen $\psi_\bullet$ eine orthonomierte Basis des $L^2(\mathbb{R})$ bilden (*Waveletbasis*) [4,19]. Mit den Substitutionen aus der Gleichung 3.22 auf der vorherigen Seite ist die entsprechende Bedingung durch

$$\int_{-\infty}^{\infty} \psi_{j,k}(t) \cdot \psi_{m,n}(t)\ dt = \begin{cases} 1 & \text{falls } j = m,\ k = n \\ 0 & \text{sonst} \end{cases} \tag{3.26}$$

gegeben [40]. Jedes $f \in L^2(\mathbb{R})$ kann dann durch

$$f = \sum_{j\in\mathbb{Z}} \sum_{k\in\mathbb{Z}} \langle f, \psi_{j,k} \rangle \psi_{j,k} \tag{3.27}$$

ausgedrückt werden [12,19]. $\psi_\bullet$ wird als *Wavelet-Basis* bezeichnet [4,19].

Bei der MSA werden abgeschlossene Unterräume V_j des $L^2(\mathbb{R})$ betrachtet, die eine Inklusionskette bilden. Diese Unterräume müssen folgende Bedingungen erfüllen [4,19]:

$$\ldots \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} \ldots V_j \subset V_{j-1} \subset \ldots \subset L^2(\mathbb{R}) \tag{3.28}$$

$$\bigcap_j V_j = \{0\} \tag{3.29}$$

$$\overline{\bigcup_j V_j} = L^2(\mathbb{R}) \tag{3.30}$$

$$f(t) \in V_j \Leftrightarrow f\left(2^j t\right) \in V_0 \tag{3.31}$$

Außerdem muss es eine *Skalierungsfunktion* $\phi \in L^2(\mathbb{R}) \cap L^1(\mathbb{R})$ geben, so dass die Funktionen $\phi(t-k), k \in \mathbb{Z}$ eine orthonormierte Basis von V_0 bilden[8] [4]. Die Skalierungsfunktion ist mit dem Wavelet vergleichbar, denn mit seiner Hilfe werden die verschobenen und skalierte Funktionen

$$\phi_{j,k}(t) := \frac{1}{\sqrt{2^j}} \phi\left(\frac{t - k \cdot 2^j}{2^j}\right) \tag{3.32}$$

erzeugt, deren Familie $(\phi_{j,k} | k \in \mathbb{Z})$ eine orthonormierte Basis von V_j bildet [4]. Der nächst niedrigfrequente Anteil einer Funktion $f \in V_{j-1}$ in V_j wird mittels einer orthogonalen Projektion $P_j f$ auf den kleineren Raum V_j beschrieben. Das orthogonale Komplement von V_j wird als W_j bezeichnet und die entsprechende Projektion $Q_j f$ beschreibt die verbleibenden hochfrequenten Anteile von f [4,19]. P_j und Q_j lassen sich als Tief- und Hochpassfilter für Funktionen aus V_{j-1} interpretieren. Die Detailstufe V_j wird als *Skala* j bezeichnet.

P_j lässt sich mittels

$$P_j f = \sum_{k=-\infty}^{\infty} \langle f, \phi_{j,k} \rangle \phi_{j,k} \tag{3.33}$$

auf eine beliebige Funktion $f \in L^2(\mathbb{R})$ anwenden [4]. Es gilt $V_{j-1} = V_j \oplus W_j$ (orthogonale Zerlegung) [4], doch die $\phi_{j,k}$ stellen nur eine orthonormierte Basis für V_j dar. Ein von ϕ abhängiges Wavelet ψ erezeugt eine Waveletbasis $\psi_\bullet$ von Wavletfunktionen

$$\psi_{j,k}(t) := \frac{1}{\sqrt{2^j}} \psi\left(\frac{t - k \cdot 2^j}{2^j}\right) = \frac{1}{\sqrt{2^j}} \psi\left(\frac{t}{2^j} - k\right), \tag{3.34}$$

[8] Für eine Methode zur Orthonormierung unzulässiger ϕ sei auf die entsprechende Literatur verwiesen [4,19].

die eine Anwendung von Q_j auf eine beliebige Funktion $f \in L^2(\mathbb{R})$ mittels

$$Q_j f = \sum_{k=-\infty}^{\infty} \langle f, \psi_{j,k} \rangle \psi_{j,k} \tag{3.35}$$

zulässt, wobei jede Unterfamilie mit festem j eine orthonormierte Basis von W_j darstellt [4].

Auf Grund der Abhängigkeit zwischen ϕ und ψ muss das Wavelet gar nicht bekannt sein. Der Schlüssel liegt in der so genannten *Skalierungsgleichung*: Für $V_0 \subset V_{-1}$ ist notwendig und hinreichend, dass eine Folge h_k, $k \in \mathbb{Z}$ reeller Zahlen existiert, so dass

$$\phi(t) = \sqrt{2} \sum_{k=-\infty}^{\infty} h_k \phi(2t - k) \tag{3.36}$$

für fast alle $t \in \mathbb{R}$ gilt [4, 19]. Mit ihrer Hilfe lassen sich nämlich die Funktionswerte von ψ mittels

$$\psi(t) = \sqrt{2} \sum_{k=-\infty}^{\infty} g_k \phi(2t - k), \quad g_k := (-1)^k h_{1+2l-k}, \; l \in \mathbb{Z} \tag{3.37}$$

berechnen [19]. Die h_k genügen also, um eine MSA durchzuführen, da sie sowohl die Skalierungsfunktion wie auch das Wavelet bestimmen[9]. Bei realen Anwendungen ist nur ein geringer Teil der $h_k \neq 0$. Insbesondere bilden die h_k dann eine endliche Kette. Beispielsweise sind die Werte $h_0 = h_1 = 1/\sqrt{2}$ [19] die Koeffizienten der Daubechies$_1$-Skalierungsfunktion.

Den Kern der MSA stellen die Gleichungen 3.36 und 3.37 dar. Aus der Skalierungsgleichung 3.36 folgt für beliebige $j, n \in \mathbb{Z}$

$$2^{-j/2} \phi\left(\frac{t}{2^j} - n\right) = 2^{-(j-1)/2} \sum_k h_k \phi\left(\frac{t}{2^j} - 2n - k\right),$$

was sich als die Rekursionsformel

$$\phi_{j,n} = \sum_k h_k \phi_{j-1,2n+k} \quad \forall j, \; \forall n \tag{3.38}$$

interpretieren lässt [4]. Analog kann eine zweite Rekursionsgleichung

$$\phi_{j,n} = \sum_k g_k \phi_{j-1,2n+k} \quad \forall j, \; \forall n \tag{3.39}$$

hergeleitet werden [4].

Ausgangspunkt sei eine maximale Detailstufe j, so dass V_j die zu transformierende Funktion $f \in L^2(\mathbb{R})$ enthält. Der Einfachheit halber sei dies $j = 0$. Zu dieser Skala liegen dann die Daten $a_{0,\cdot}$ vor. Streng genommen müssten sie mittels

$$a_{0,k} := \langle f, \phi_{0,k} \rangle := \int_{-\infty}^{\infty} f(t) \overline{\phi(t - k)} \, dt$$

berechnet werden, doch für den Fall, dass f nur als diskreter Datensatz $(f(i) \,|\, i \in \mathbb{Z})$ vorliegt, können sie auch einfach mittels $a_{0,k} := f(k)$ festgelegt werden [4].

[9] Natürlich fällt auf, dass das Wavelet nicht eindeutig ist. Legt man sich aber auf eine Belegung von l fest (beispielsweise $l = 0$), resultiert zu jeder Skalierungsfunktion ein eindeutiges Wavelet.

Der Tiefpassprojektor für V_0 ist bei Verwendung der $a_{0,k}$ formal als

$$P_0 f = \sum_k a_{0,k} \phi_{0,k}$$

definiert (vgl. Gleichung 3.33 auf Seite 18). Bei der MSA wird bei jedem Schritt, also $j-1 \to j$, die Wellenlänge (Frequenz) verdoppelt, ohne dabei explizit das Skalarprodukt ausrechnen zu müssen [4]:

$$P_{j-1} f = \sum_k \langle f, \phi_{j-1,k} \rangle = \sum_k a_{0,k} \phi_{0,k}. \tag{3.40}$$

Unter Verwendung von der Gleichung 3.38 auf der vorherigen Seite lassen sich die $a_{j-1,\cdot}$ *Approximationskoeffizienten* mittels

$$a_{j,n} = \langle f, \phi_{j,n} \rangle = \sum_k \overline{h_k} \langle f, \phi_{j-1,2n+k} \rangle = \sum_k \overline{h_k} a_{j-1,2n+k} \tag{3.41}$$

auch direkt über die $a_{j,\cdot}$ und die $h_\cdot$ berechnen [4]. Es gilt $V_{j-1} = V_0 \oplus W_0$ bzw. $P_{j-1} f = P_j f + Q_j f$, dabei enthält $P_j f$ alle Merkmale von f mit einer Ausdehnung von wenigstens 2^{j-1} und $Q_j f$ alle Merkmale um $2^j/\sqrt{2}$ auf der Zeitachse [4], so dass bei jedem Rekursionsschritt Details der Größe $\sim 2^j/\sqrt{2}$ von f abgespaltet werden.

Die $\psi_{j,\cdot}$ bilden eine orthonormierte Basis von W_j, weshalb sich $Q_j f$ als

$$Q_j f = \sum_k d_{j,k} \psi_{j,k}$$

ergibt. Unter Ausnutzung von der Gleichung 3.39 auf der vorherigen Seite und der Gleichung 3.40 lassen sich die $d_{j,\cdot}$ (*Approximationskoeffizienten*) mittels

$$d_{j,n} = \langle f, \psi_{j,n} \rangle = \sum_k \overline{g_k} \langle f, \psi_{j-1,2n+k} \rangle = \sum_k \overline{g_k} a_{j-1,2n+k} \tag{3.42}$$

aus den Daten des vorherigen Rekursionsschritts berechnen [4]. Der Faktor 2 auf in den Gleichungen 3.41 und 3.42 bewirkt eine Unterabtastung, so dass die Menge neu berechneter Approximations- und Detailkoeffizienten halbiert wird. Umfasst der anfägliche $a_{0,\cdot}$ Vektor umfasse $n = 2^m$ Elemente, dann lässt sich mit der Gleichung 3.41 und der Gleichung 3.42 ein schneller Algorithmus formulieren, der mit m Rekursionsschritten und einer Berechnungskomplexität von $O(n)$ auskommt [19].

Die Projektion $P_{j-1} f$ kann als

$$P_{j-1} f = P_j f + Q_j f = \sum_k a_{j,k} \phi_{j,k} + \sum_k d_{j,k} \psi_{j,k}$$

aufgeschrieben werden. Demnach gilt für ein $a_{j-1,\cdot}$

$$a_{j-1,n} = \langle P_{j-1} f, \phi_{j-1,n} \rangle = \sum_k a_{j,k} \langle \phi_{j,k}, \phi_{j-1,n} \rangle + \sum_k d_{j,k} \langle \psi_{j,k}, \phi_{j-1,n} \rangle,$$

wobei mit den beiden Rekursionsformeln (Gleichung 3.38 auf der vorherigen Seite und Gleichung 3.38 auf der vorherigen Seite)

$$\langle \phi_{j,k}, \phi_{j-1,n} \rangle = \sum_k h_{n-2k} \quad \text{und} \quad \langle \psi_{j,k}, \phi_{j-1,n} \rangle = \sum_k g_{n-2k}$$

folgt, so dass schließlich die Rekonstruktionsformel

$$a_{j-1,n} = \sum_k h_{n-2k} a_{j,k} + \sum_k g_{n-2k} d_{j,k} \tag{3.43}$$

entsteht [4]. Die MSA und die anschließende Rekonstruktion kann schematisch folgendermaßen dargestellt werden:

$$\begin{tikzcd}
a_{0,\cdot} \arrow[r, "\overline{h}"] \arrow[dr, "\overline{g}"] & a_{1,\cdot} \arrow[r, "\overline{h}"] \arrow[dr, "\overline{g}"] & a_{2,\cdot} \arrow[r, "\overline{h}"] \arrow[dr, "\overline{g}"] & \cdots \arrow[r, "h"] & a_{2,\cdot} \arrow[r, "h"] & a_{1,\cdot} \arrow[r, "h"] & a_{0,\cdot} \\
& d_{1,\cdot} & d_{2,\cdot} & \cdots \arrow[ur, "g"] & d_{2,\cdot} \arrow[ur, "g"] & d_{1,\cdot} \arrow[ur, "g"] &
\end{tikzcd}$$

Da die $a_{j-1,\cdot}$ wieder aus den $a_{j,\cdot}$ und den $d_{j,\cdot}$ rekonstruiert werden können, kann das Ergebnis einer MSA im Eingabearray etwa nach folgendem Schema gespeichert werden:

Eingabe:	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$	$a_{0,8}$
Ausgabe:	$d_{1,1}$	$d_{1,2}$	$d_{1,3}$	$d_{1,4}$	$d_{2,1}$	$d_{2,2}$	$d_{3,1}$	$a_{3,1}$

3.3.4. Lifting-Schema

Das *Lifting-Schema* stellt einen alternativen Ansatz dar, eine DWT durchzuführen. Es wurde 1994 eingeführt, doch seine Wurzeln reichen jedoch tiefer zurück [41]. Das Lifting-Schema ist explizit darauf ausgelegt, eine DWT „am Platze“ auszuführen und kommt in der Regel mit weniger Berechnungen als die klassische MSA aus. Dies liegt daran, dass das Extrahieren von Detailinformation einer abgetasteten Funktion nicht mittels der Anwendung von „Blackboxfunktionen“ (Skalierungsfunktion und Wavelet) innerhalb eines generellen mathematischen Verfahrens realisiert wird, sondern sehr implementierungsnah und unter Ausnutzung der Unterabtastung bewerkstelligt wird. Grundlage dieses Abschnitts stellt das Buch „Ripples in Mathematics“ dar [17].

Ausgangspunkt ist ein Datenvektor $(s_{0,1} \ \ldots \ s_{0,2n})$ mit $2n = 2^m$, der eine abgetastete reellwertige Funktion $f \in L^2(\mathbb{R})$ repräsentiert. Der Kern des Lifting-Schemas ist die Spaltung der Eingabesequenz und eine Annahme über den Verlauf der abgetasteten Funktion. Wie bei der FFT und der MSA werden m rekursive Iterationen durchlaufen. Bei jeder Iterationen werden wie bei der MSA unterabgetastet Detail- und Approximationskoeffizienten berechnet. Der einfachste Algorithmus ist das *Ein-Schritt-Lifting* (One Step Lifting). Pro Iteration werden drei auf einander folgende Schritte ausgeführt.

1. *Split*: Die Eingabe $s_{j,1}, \ldots, s_{j,2n}$ wird in gerade und ungerade Indizes aufgeteilt. Es resultieren die geraden Elemente $s_{j-1,1}, \ldots, s_{j-1,n}$ und die ungeraden Elemente $d_{j-1,1}, \ldots, d_{j-1,n}$.

2. *Prediction*: Jede Lifting-Schema-Analyse unterstellt dem untersuchten Signal eine bestimmte Struktur. Es wird also eine Korrelation zwischen einem Wert und seinen Nachbarwerten vermutet.

   Beim Ein-Schritt-Lifting werden nur die Werte $s_{j-1,i}$ und $d_{j-1,i}$ verglichen. Ein Operator $P(s_{j-1,i})$ liefert den Erwartungswert für $d_{j-1,i}$ in Abhängigkeit vom vorangehenden Wert $s_{j,i}$. Die Differenz zwischen Vorhersage und tatsächlichem Wert wird als Detailkoeffizient in $d_{j-1,i}$ gespeichert: $d_{j-1,i} \to d_{j-1,i} - P(s_{j-1,i})$.

3. *Update*: Es gilt mit Hilfe der berechneten Detailkoeffizienten (Abweichung von der Vorhersage) und den betrachteten $s_{j-1,\cdot}$ einen Updateoperator U auf die $s_{j-1,\cdot}$ anzuwenden, so dass sie eine Approximation unter Beachtung der vermuteten Korrelation darstellen. Die verwendete Operation ist stets eine Addition. Beim Ein-Schritt-Lifting bedeutet dies: $s_{j-1,i} \to s_{j-1,i} + U\left(d_{j-1,i}\right)$ Die $s_{j-1,i}$ werden als Eingabe für den nächste Iterationsschritt verwendet.

Ein Beispiel ist die (nicht normierte) Haar-Transformation. Hier wird das Signal als konstant angenommen. Die Detailkoeffizienten berechnen sich als Differenzen zwischen $s_{j-1,i}$ und $d_{j-1,i}$. Die Approximationskoeffizienten stellen Mittelwerte dar.

$$\begin{array}{ccccccccc} d_{j-1,i} & \to & d_{j-1,i} & - & s_{j-1,i} & = & s_{j,2i+1} & - & s_{j,2i} \\ s_{j-1,i} & \to & s_{j-1,i} & + & \frac{d_{j-1,i}}{2} & = & s_{j,2i} & - & \frac{s_{j,2i+1}}{2} \end{array}$$

Ein-Schritt-Lifting lässt nur eine sehr eingeschränkte Analyse zu. So beinhaltet beispielsweise die tatsächliche Haar-Transformation eine Normalisierung. Beim *generellen Lifting* können innerhalb eines Iterationsschrittes nach dem Aufspalten der Eingabe jedoch beliebig viele Prediction- und Update-, so wie zusätzlich Normalisierungsoperatoren eingesetzt werden. Ein Operator wird jedoch erst auf den kompletten Datensatz angewendet, bevor der nächste Operator zum Einsatz kommt. Die (normalisierte) Haar-Transformation berechnet sich dann wie zuvor durch

$$\begin{array}{ccccc} d_{j-1,i} & \to & d_{j-1,i} & - & s_{j-1,i} \\ s_{j-1,i} & \to & s_{j-1,i} & + & \frac{d_{j-1,i}}{2} \end{array}$$

gefolgt von zwei Normalisierungoperationen

$$\begin{array}{ccc} s_{j-1,i} & \to & \sqrt{2} \quad s_{j-1,i} \\ d_{j-1,i} & \to & \frac{1}{\sqrt{2}} \quad d_{j-1,i}. \end{array}$$

Ein Beispiel für ein Lifting, das nicht nur je zwei benachbarte Punkte miteinander vergleicht, ist die Daubechies$_4$-Transformation.

$$\begin{aligned} & s_{j-1,i} \to s_{j-1,i} + \sqrt{3} d_{j-1,i} \\ & d_{j-1,i} \to d_{j-1,i} - \frac{1}{4}\sqrt{3} s_{j-1,i} - \frac{1}{4}\left(\sqrt{3}-2\right) s_{j-1,i-1} \\ & s_{j-1,i} \to s_{j-1,i} - d_{j-1,i+1} \\ & s_{j-1,i} \to \frac{\sqrt{3}-1}{\sqrt{2}} s_{j-1,i} \\ & d_{j-1,i} \to \frac{\sqrt{3}+1}{\sqrt{2}} d_{j-1,i} \end{aligned}$$

Zur Rekonstruktion werden wie beim MSA-Algorithmus nur die Detailkoeffizienten und der letzte Approximationskoeffizient benötigt. Die inverse Transformation des Lifting-Schemas ist eine Umkehrung des eingesetzten Transformationsalgorithmus.

- Alle Operationen werden in umgekehrter Reihenfolge durchlaufen.
- Subtraktionen bei Predict-Operationen werden durch Additionen ersetzt.
- Additionen bei Update-Operationen werden durch Subtraktionen ersetzt.
- Normalisierungskonstanten werden invertiert.

Mit diesen einfachen Regeln ist beispielsweise die inverse Haar-Transformation durch

$$\begin{array}{lcllcl} d_{j-1,i} & \rightarrow & \sqrt{2}d_{j-1,i} & & \\ s_{j-1,i} & \rightarrow & \frac{1}{\sqrt{2}}s_{j-1,i} & & \\ s_{j,2i} & \rightarrow & s_{j-1,i} & - & \frac{d_{j-1,i}}{2} \\ s_{j,2i+1} & \rightarrow & d_{j-1,i} & + & s_{j,2i} \end{array}$$

gegeben.

Das Lifting-Schema bietet einen einfachen Zugang zur Entwicklung eigener „Wavelet-Transformationen“, wozu auf die entsprechende Literatur verwiesen sei [41].

3.3.5. Randwertprobleme

In allen bisher vorgestellten rekursiven Algorithmen FFT, MSA und Lifting wurde vorausgesetzt, dass die Menge der initialen Eingabewerte eine Zweierpotenz, also $n = 2^m$ ist. Grundsätzlich gibt es zwei Möglichkeiten, eine unzulässige Eingabenlänge auf eine Zweierpotenz zu bringen.

- Die Eingabe wird auf die nächst kleinere Zweierpotenz gekürzt.

- Die Eingabe wird mit Werten aufgefüllt, so dass die nächst größere Zweierpotenz erreicht wird.

Die erste Möglichkeit ist mit Sicherheit problematisch: Besteht die Eingabe beispielsweise aus 255 Elementen, so wird fast die Hälfte aller Werte verworfen. Außerdem ergibt sich um Zusammenhang mit komplexeren Liftings (wie beispielsweise die vorgestellte Daubechies$_4$-Transformation) das Problem, dass an den Rändern mehr Nachbarpunkte betrachtet werden, als zur Verfügung stehen. Hier werden nun drei Methoden vorgestellt, das Auffüllen der Eingabe bzw. die Bereitstellung von Werten nicht vorhandener Nachbarpunkte zu bewerkstelligen [17]:

- *Zero-Padding* Nicht vorhandene Werte werden auf Null gesetzt. Das Problem besteht natürlich darin, dass dadurch hohe Sprungstellen entstehen und das Ergebnis der verwendeten Transformation merklich beeinflussen können.

- *Periodisierung* Das Signal wird an benötigten Stellen als periodisch wiederholt angenommen. Bei konstanten oder periodischen Elementen in dem untersuchten Signal scheint dies eine möglicherweise sinnvolle Methode. Allerdings können auch bei dieser Methode ungewollte Sprungstellen auftreten.

- *Spiegelung* Das Signal wird an benötigten Stellen als sich gespiegelt fortsetzend angenommen. Unerwünschte Sprungstellen werden nicht erzeugt. Diese Methode lässt sich auch gut bei der Musikproduktion mit Hilfe eines Samplers einsetzen: Soll ein Sample wiederholt („geloopt“) werden, um eine möglichst lange Note spielen zu können, so verringert diese Methode gegenüber der Periodisierung das Risiko von so genannten „Clicks“, die sich in Form von Knacken bemerkbar machen. Allerdings kann auch hier der Funktionsverlauf sehr verfälscht werden, beispielsweise, falls das Signal eigentlich kontinuierlich steigend ist.

3.3.6. Thresholding

Die häufigste Anwendung der Wavelet-Transformation besteht darin, ein abgetastetes Signal zu transformieren, alle Detailkoeffizienten, die nicht einen bestimmten Schwellwert (Threshold) überschreiten, auf Null zu setzen und anschließend eine inverse Wavelet-Transformation durchzuführen. Dieses Verfahren wird als *Thresholding* bezeichnet. Seine Einsatzgebiete sind beispielsweise Rauschunterdrückung und Komprimierung.

Man unterscheidet *hartes Thesholding* (Hard Thresholding) und *weiches Thresholding* (Soft Thresholding). Beim harten Thresholding werden alle betroffenen Deteilkoeffizenten auf Null gesetzt, während beim weichen Thresholding zusätzlich von jedem verbleibendem Detailkoeffizient der Schwellwert subtrahiert wird. Da auch negative Detailkoeffizienten auftreten können, empfiehlt es sich natürlich mit Absolutbeträgen zu arbeiten. Für einen Deteilkoeffizent d ist bei gegebenem Schwellwert λ der resultierende Detailkoeffizient beim harten Thresholding durch

$$T_h(d,\lambda) := \begin{cases} 0 & \text{falls} \quad |d| \leq \lambda \\ d & \text{falls} \quad |d| > \lambda \end{cases} \tag{3.44}$$

und beim weichen Thresholding durch

$$T_s(d,\lambda) := \begin{cases} 0 & \text{falls} \quad |d| \leq \lambda \\ d - \operatorname{sign}(d) \cdot \lambda & \text{falls} \quad |d| > \lambda \end{cases} \tag{3.45}$$

definiert. Natürlich können auch nur Koeffizienten unterhalb des Schwellwerts auf Null gesetzt werden, die offiziellen Definitionen sind aber durch die beiden angegebenen Gleichungen gegeben [7].

Der Schwellwert λ kann mit Hilfe von Hintergrundwissen über die Daten manuell gewählt werden. Donoho and Johnstone entwickelten jedoch zur Rekonstruktion einer mit als normalverteilt angenommenen Rauschen überlagerten Funktion einen universellen Schwellwert

$$\lambda := \sigma\sqrt{2\log(n)}, \tag{3.46}$$

wobei n die Anzahl der Abtastpunkte und σ die Standardabweichung des Rauschens darstellen [7]. Sind $a_1, \ldots, a_n$ die Koeffizienten der höchsten Detailstufe und M der Median der a_i, dann kann eine Abschätzung $\sigma \approx \tilde{M}/0.6745$ erfolgen, wobei $\tilde{M}$ den Median der Werte $|a_i - M|$ bezeichnet [40].

3.4. Fuzzy-Transformation

Die *Fuzzy-Transformation* (F-Transformation[10]) ist eine relativ neue Transformation, die von Irina Perfilieva eingeführt wurde [24]. Sie dient in erster Linie der Approximation von Funktionen aus einem endlichen Teilbereich von $\mathbb{R}$ mit einem diskreten Satz von reellen Werten. Diese Diskretisierung wurde beispielsweise zur Überführung von Differentialgleichungen in algebraische Gleichungen benutzt [38]. Mit einer Erweiterung auf zwei Dimensionen kann die F-Transformation auch zur Komprimierung von Bilddaten eingesetzt werden [25].

In Abschnitt 3.4.1 wird deshalb zunächst auf den benötigten Begriff der „unscharfen Menge“ eingeführt. Abschnitt 3.4.2 behandelt dann die F-Transformation. Abschließend werden in Abschnitt 3.4.3 kurz einige Ideen zur Konstruktion von gut approximierenden F-Transformationen vorgestellt.

[10] Die Abkürzung F-Transformation sollte nicht mit der Abkürzung FT verwechselt werden, die hier der Fourier-Transformation vorbehalten ist.

3.4.1. Unscharfe Mengen

In der klassischen Logik gibt es zur Bewertung einer Aussage (aussagen- bzw. prädikatenlogische Formel unter einer gegebenen Belegung) nur zwei Möglichkeiten: Entweder die Aussage ist wahr, oder die Aussage ist falsch. Doch wie sieht es mit einer Aussage wie „eine Geschwindigkeit von 55.1 km/h ist schnell“ aus? Hier muss es nach den Regeln der Prädikatenlogik einen festen Grenzwert geben, ab dem eine Geschwindigkeit als schnell eingeschätzt wird. Angenommen die Grenze zwischen langsam und schnell liegt bei 55 km/h. Dann wäre eine Geschwindigkeit von 54.9 km/h langsam, während die Geschwindigkeit von 55.1 km/h als schnell einzustufen ist. Dies entspricht wohl kaum einem natürlichen Empfinden. Im Alltag werden Ausdrücke wie „relativ schnell“ oder „recht schnell“ gebraucht. Die Aussage ist also irgendwie „unscharf“.

Jede *scharfe Menge* A hat eine charakteristische *Zugehörigkeitsfunktion* μ_A, die durch

$$\mu_A(x) := \begin{cases} 1 & \text{falls} \quad x \in A \\ 0 & \text{sonst} \end{cases} .$$

gegeben ist [43]. Die Frage nach der Geschwindigkeit ließe sich also lösen, indem man die zwei Mengen $L := \{x|x \in [0, 55)\}$ und $S := \{x|x \geq 55\}$ festlegt und die Beantwortung der Frage durch die Zugehörigkeitsfunktion μ_S entscheiden lässt.

In der Fuzzy-Logik werden jedoch *unscharfe Mengen* (Fuzzy-Sets) betrachtet. Eine unscharfe Menge A auf dem Universum X hat eine Zugehörigkeitsfunktion [43]:

$$\mu_A : X \to [0, b] \tag{3.47}$$

Die Menge $A_\alpha := \{x \,|\mu_A(x) > \alpha\}\,,\ \alpha \in [0, 1)$ wird als *starker α-Schnitt*, die Menge $A_{\bar{\alpha}} := \{x \,|\mu_A(x) \geq \alpha\}\,,\ \alpha \in (0, 1]$ als *schwacher α-Schnitt* oder α-Menge bezeichnet [42]. Wird $b = 1$ gesetzt, heisst A *normal* [43].

Man kann nun relative Aussagen treffen, indem man entsprechende Zugehörigkeitsfunktionen festlegt. Für das Geschwindigkeitsbeispiel wären also μ_l und μ_S zu definieren. Man könnte es etwa so einrichten, dass $\mu_L(55.1) = 0$ und $\mu_S(55.1) = 0.7$.

3.4.2. Fuzzy-Transformation

Eine F-Transformation ist immer nur für einen endlichen Bereich $[a, b] \subset \mathbb{R}$ definiert, der als *Universum* bezeichnet wird. Es werden also Funktionen $f : [a, b] \to \mathbb{R}$ betrachtet. Zur Durchführung einer F-Transformation wird eine Partition des gewählten Universums bestimmt, die der folgenden Definition genügt [24]:

Definition 3.8 (Fuzzy-Partition) *Es sei eine Folge von Punkten $x_1, \ldots, x_n$ mit $n \geq 2$, $x_i \in [a, b]$ festgelegt. Zu den Punkten korrespondieren (normale) unscharfe Mengen*[11] *$A_1, \ldots, A_n$. Zusätzlich werden $x_{-1} := a$ und $x_{n+1} := b$ definiert. Eine* Fuzzy-Partition *über $[a, b]$ ist dann gegeben, falls die A_k folgenden Bedingungen erfüllen.*

1. *$A_k : [a, b] \to [0, 1]\,, A_k(x_k) = 1$.*
2. *$A_k(x) = 0$ falls $x \notin [x_{k-1}, x_{k+1}]$.*
3. *A_k ist stetig.*

[11] Zur Vereinfachung werden die Bezeichner der unscharfen Mengen auch für die entsprechenden Zugehörigkeitsfunktionen verwendet, also $A(x) := \mu_A(x)$.

4. *A_k ist monoton steigend in $[x_{k-1}, x_k]$.*

5. *A_k ist monoton fallend in $[x_k, x_{k+1}]$.*

6. $\sum_{k=1}^{n} A_k(x) = 1$ *für alle* $x \in [a, b]$.

Die A_k werden als *Basisfunktionen* bezeichnet. Eine Möglichkeit die A_k zu definieren besteht darin, Dreiecksfunktionen zu benutzen [24]. Die x_k können als Stützstellen interpretiert werden. Werden sie äquidistant gewählt, so dass $x_k = a + h(k-1)$ mit $h := (b-a)/(n-1)$, und gelten bei $n > 2$ die zusätzlichen Bedingungen

7. $A_k(x_k - x) = A_k(x_k + x)$ für alle $x \in [a, b]$, $1 < k < n$.

8. $A_{k+1}(x) = A_k(x - h)$ für alle $x \in [a, b]$, $1 < k < n - 1$.

wird die Fuzzy-Partition als *uniform* bezeichnet [24]. In der Abbildung 3.1 sind eine uniforme und eine nichtuniforme Fuzzy-Partition dargestellt.

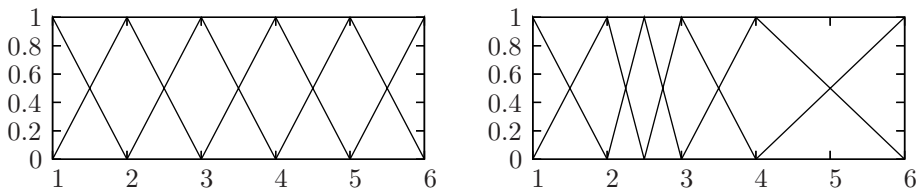

Abbildung 3.1.: Eine uniforme Fuzzy-Partition (links) und eine nichtuniforme Fuzzy-Partition (rechts) mit Dreiecksfunktionen

Das Bild der F-Transformation einer Funktion f ist ein endlicher reellwertiger Vektor. Die folgende Definition beschreibt die Berechnung der Transformation [24]:

Definition 3.9 (Fuzzy-Transformation) *Gegeben sei eine Fuzzy-Partition von $[a, b]$ mit n Basisfunktionen, $\mathbf{c}$ sei ein Vektor $(c_1, \ldots, c_n) \in \mathbb{R}^n$. Dann wird die (kontinuierliche) F-Transformation einer integrierbaren Funktion $f : [a, b] \to \mathbb{R}$ durch*

$$c_k := \frac{\int_a^b f(x) A_k(x) \, dx}{\int_a^b A_k(x) \, dx}$$

berechnet. Die $c_1, \ldots, c_n$ werden als Komponenten *der F-Transformation bezeichnet.*

Der obere Term verhält sich ähnlich wie eine WFT-Fensterfunktion mit ausgeprägtem Maximum. Allerdings werden hier keine Frequenzanteile betrachtet, sondern direkt der Bildbereich von f um die entsprechende Stützstelle x_k herum gewichtet analysiert. Auf Grund des Quotiententerms stellt c_k einen gewichteten Mittelwert im Bereich $[x_{k-1}, x_{k+1}]$ dar. Außerdem sorgt er dafür, dass dieses Ergebnis in Bezug zur Ausdehnung von A_k im Universum gesetzt wird. So können die Komponenten mit einander verglichen werden und ermöglichen so im Zusammenspiel eine Repräsentation von f. Diese Repräsentation ist allerdings nur approximativ. Folglich ist in der Regel keine exakte Rekonstruktion zu erwarten. Die *inverse Fuzzy-Transformation* wird in der folgenden Definition geliefert [24].

Definition 3.10 (inverse Fuzzy-Transformation) *Gegeben sei eine Fuzzy-Partition von* $[a, b]$ *mit* n *Basisfunktionen, so wie der aus einer Anwendung der entsprechenden F-Transformation auf eine Funktion* f *hervor gegangene Komponentenvektor* $\mathbf{c} \in \mathbb{R}^n$*. Dann ist die inverse Fuzzy-Transformation als*

$$f(x) = \sum_{k=1}^{n} c_k A_k(x) \quad , \; x \in [a, b]$$

definiert.

Die Informationen aus dem Komponentenvektor werden entsprechend der durch die Basisfunktionen gegebenen Gewichtungen wieder in den $[a, b]$-Raum von f transferiert. Die Rückrichtung ist also grundsätzlich diskret. Doch wie sieht es mit der Transformation von abgetasteten Funktionen aus? Da die F-Transformation selbst eine Approximation mittels Stützstellen vornimmt, liegt es nahe sie auch einfach auf die durch die Abtastung erhaltenen Funktionswerte anzuwenden. Zwar „fehlen“ die Werte zwischen den Abtastpunkten, doch die folgende Definition bezieht auch nur die definierten Werte nach dem gleichen Prinzip wie in Definition 3.9 auf der vorherigen Seite in die Analyse ein [24].

Definition 3.11 (diskrete Fuzzy-Transformation) *Gegeben sei eine Fuzzy-Partition von* $[a, b]$ *mit* n *Basisfunktionen,* $\mathbf{c}$ *sei der zu berechnende Komponentenvektor* $(c_1, \ldots, c_n) \in \mathbb{R}^n$*. Dann erfolgt die* Fuzzy-Transformation *einer abgetasteten Funktion* $f : \{s_1, \ldots, s_m\} \to \mathbb{R}$*,* $s_i \in [a, b]$ *durch*

$$c_k := \frac{\sum_{i=1}^{m} f(x_i) A_k(x_i)}{\sum_{i=1}^{m} A_k(x_i)}.$$

Dieses Verfahren lässt sich leicht für beliebig viele Dimensionen generalisieren [25]. Hier wird jedoch nur kurz auf die Berechnung der F-Transformation für den zweidimensionalen Fall eingegangen. Das betrachtete Universum U ist in diesem Fall $[a_x, b_x] \times [a_x, b_x] \subset \mathbb{R}^2$. Die zulässigen Funktionen sind entsprechenderweise der Form $f : U \to \mathbb{R}$. Für jede der beiden Dimension wird eine Fuzzy-Partition definiert. Die zugehörigen Basisfunktionen sind $A_1, \ldots, A_n$ und $B_1, \ldots, B_m$. Aus U resultiert eine $n \times m$ Komponentenmatrix $\mathbf{C}$, die die Komponenten $c_{x,y}$ enthält. Die kontinuierliche F-Transformation wird analog zu Definition 3.9 auf der vorherigen Seite durch

$$c_{k,i} := \frac{\int_{a_y}^{b_y} \int_{a_x}^{b_x} f(x, y) A_k(x) B_i(y) \; dx \, dy}{\int_{a_y}^{b_y} \int_{a_x}^{b_x} A_k(x) B_i(y) \; dx \, dy} \tag{3.48}$$

berechnet. Die Generalisierung von Definition 3.11 erfolgt unter der Voraussetzung eines $(m_x \cdot m_y)$-Abtastgitters[12] zu der diskreten F-Transformation

$$c_{k,i} := \frac{\sum_{q=1}^{m_y} \sum_{p=1}^{m_x} f(x_p, y_q) A_k(x_p) B_i(y_q)}{\sum_{q=1}^{m_y} \sum_{p=1}^{m_x} A_k(x_p) B_i(y_q)} \tag{3.49}$$

[12] Das Gitter muss dabei dicht genug bezüglich der Basisfunktionen sein. Die entsprechenden Bedingungen sind $(\forall k)(\exists p) A_k(x_p) > 0$ und $(\forall k)(\exists p) B_k(y_p) > 0$ [25].

und entsprechend die Generalisierung von Definition 3.10 auf der vorherigen Seite zu der inversen Transformation

$$f(x,y) = \sum_{k=1}^{n} \sum_{i=1}^{m} c_{k,i} A_k(x) B_i(y). \tag{3.50}$$

Die Generalisierung auf mehr Dimensionen erfolgt analog. Der folgende Abschnitt behandelt jedoch entsprechend der verwendeten Literatur nur die eindimensionale F-Transformation.

3.4.3. Nichtuniforme Fuzzy-Partitionen

Es ist offensichtlich, dass eine Funktion f mit steigender Anzahl der Basisfunktionen besser approximiert werden kann. Die Güte dieser Approximation hängt jedoch nicht nur von der Anzahl der Basisfunktionen ab, sondern auch davon, wie günstig die Stützstellen bezüglich f gewählt sind.

In der Diplomarbeit von M. Gaspar wurde folgender Annahme nachgegangen [13]: Eine begrenzte Menge von Basisfunktionen approximiert $f \in [a,b]$ genau dann optimal, wenn die Stützstellen $x_2, \ldots, x_{n-1}$ auf für den Kurvenverlauf von f wichtigen Punkten positioniert werden. Dazu muss man sich natürlich von uniformen Fuzzy-Partitionen lösen.

In der Arbeit wurden mehrere Algorithmen zur Bestimmung einer optimalen Fuzzy-Partition bezüglich der Approximation von abgetasteten Funktionen mit Hilfe einer Fehlerfunktion entwickelt, die im Folgenden kurz vorgestellt werden. Für Details sei auf die Originalliteratur verwiesen [13].

Der *Greedy Algorithmus* fügt sukzessiv Basisfunktionen in eine vorhandene Fuzzy-Partition ein. Ausgangspunkt sind die beiden vorgegebenen Randbasisfunktionen. Bei jedem Schritt wird eine zusätzliche Basisfunktion eingefügt. Dabei wird die Basisfunktion gewählt, die zu der Fuzzy-Partion mit dem geringsten Fehlerfunktionswert führt. Dies wird solange fortgesetzt, bis ein geeignetes Abbruchkriteriumerfüllt ist.

Das *informierte Einfügen mit lokaler Verbesserung* fügt ebenfalls sukzessiv Basisfunktionen ein. Allerdings wird nicht jede mögliche Basisfunktion getestet, sondern nur ein Maximum zufällig nach einer bestimmten Wahrscheinlichkeitsverteilung gewählt. Diese ist so eingerichtet, dass die Wahrscheinlichkeit einzelner Abtastpunkte von einer numerischen Approximation der ersten oder zweiten Ableitung von f bestimmt wird. Nur lokal um diesen initialen Wert wird dann wie beim Greedy Algorithmus optimiert.

Die *iterierte lokale Verbesserung* geht dagegen von einer initial gegebenen Anzahl von Basisfunktionen aus und versucht diese mittels numerisch approximierten Gradientenabstiegs die Fehlerfunktion zu minimieren. Da das Resultat dieses Verfahrens ist sehr abhängig von der initialen Wahl der x_k ist, wird von M. Gaspar die Nutzung seiner Wahrscheinlichkeitsfunktion empfohlen.

Beim *sukzessiven schwellwertgesteuerten Einfügen* wird versucht zu bestimmen, ob es auf einem Teilbereich des Universums notwendig ist, eine Basisfunktion hinzuzufügen, um eine bestimmte Approximationsgüte in diesem Teilbereich zu gewährleisten. Der Teilbereich wird als Fenster bezeichnet und erstreckt sich anfangs nur über die ersten beiden Abtastpunkte. Falls die Approximationsgüte nicht erreicht wird erfolgt das Einfügen einer neuen Basisfunktion und das Fenster korrigiert, andernfalls wird das Fenster vergrößert.

4. Fitting von Funktionen

Neben der Rekonstruktion eines Signals aus verfälschten Daten mit Hilfe von Glättung, kann auch der Versuch unternommen werden, durch gewisse Annahmen über den Verlauf des untersuchten Signals, den Fehler vom Signal zu trennen.

Diese Annahmen kann beispielsweise als Funktionsklasse ausgerückt werden. Im Abschnitt 4.1 wird diese Idee näher spezifiziert. Der Abschnitt 4.2 behandelt einen globaleren Optimierungsansatz, der versucht, Prinzipien der Natur zu simulieren.

4.1. Least-Squares Fitting

Bei der *Regressionsanalyse* handelt es sich um eine Methode der Statistik, die dazu dient, Abhängigkeiten zwischen Variablen zu ermitteln. Dabei werden Datenpunkte betrachet und angenommen, dass sie mit gewissen Abweichungen durch ein funktionales Modell beschrieben werden können. Die Datenpunkte werden als Beispielbelegungen der betrachteten Variablen aufgefasst. Ein mögliches Beispiel wäre die Fragestellung, ob eine lineare Abhängigkeit zwischen durchschnittlichem monatlichem Netto-Einkommen und den Kinobesuchen einer Person im Jahr 2007 besteht. Das Einkommen könnte dabei als x, die Anzahl der Kinobesuche als y Komponente von Datentupeln aufgefasst werden. Jede Testperson der Stichprobe würde dann einen Datenpunkt liefern. Das Modell würde diesen Datenpunkten eine lineare Abhängigkeit unterstellen und die Regressionsanalyse hätte das Ziel, die Parameter dieser Abhängigkeit zu bestimmen.

Beschränkt man sich auf den reellwertigen Zahlenraum, so lässt sich jeder Datenpunkt durch einen Vektor $\mathbf{d} = (x_1, \ldots, x_l, y) \in \mathbb{R}^{l+1}$ beschreiben. Dabei gibt l die Anzahl der unabhängigen Variablen an. Die Menge aller Datenpunkte ist dann ein Vektor $(\mathbf{d}_1, \ldots, \mathbf{d}_m)$. Das Modell wird durch eine *Modellfunktion*

$$f : \mathbb{R}^{l+n} \to \mathbb{R}, \quad x_1, \ldots, x_l; p_1, \ldots, p_n \mapsto f\left(x_1, \ldots, x_l; p_1, \ldots, p_n\right)$$

beschrieben. Dabei stellt $\mathrm{p} = (p_1, \ldots, p_n)$ den zu bestimmenden Parametervektor da. Die Dimensionalität von f ist durch die Anzahl der unabhängigen Variablen, also l bestimmt. Der Prozess der Parameteroptimierung wird im Folgenden als *Fitting* bezeichnet.

Im Allgemeinen wird vorausgesetzt, dass die Datenpunkte durch eine Modellfunktion mit optimalen Parametern approximiert werden können. Dabei wird angenommen, dass jeder Datenpunkt mit einem Fehler E behaftet ist und durch

$$y = f\left(x_1, \ldots, x_l\right) + E$$

aus der Modellfunktion f mit fest gewählten (optimalen) Parametern hervor geht. E wird dabei in der Regel als normalverteilt angenommen [29].

Um das Fitting einer Modellfunktion f durchzuführen, muss definiert werden, wann eine Approximation der Datenpunkte als optimal angesehen wird. Beim *Least-Squares* (LS) wird dazu eine Fehlerfunktion χ festgelegt, die sich aus der Summe der quadrierten Fehler zwischen den Datenpunktn und den entsprechenden Modellfunktionswerten berechnet. Der Parametervektor der Modellfunktion ist dann optimal, falls der Fehlerfunktionswert minimal ist. In seiner

klassischen Variante werden beim LS nur eindimensionale Modellfunktionen betrachtet. Dann ergibt sich die folgende Definition [27, 36]:

Definition 4.1 (Least-Squares) *Gegeben sei eine eindimensionale Modellfunktion mit n Parametern $\mathbf{p}$ und m Datenpunkte $(x, y) \in \mathbb{R}^2$. Die LS-Fehlerfunktion χ ist dann durch*

$$\chi(\mathbf{p}) := \sum_{i=1}^{m} \left(y_i - f(x_i; p_1, \ldots, p_n)\right)^2 \tag{4.1}$$

definiert.

Soll das Fitting einer mehrdimensionalen Modellfunktion f mittels LS erfolgen, so kann χ als

$$\chi(\mathbf{p}) := \sum_{i=1}^{m} \left(y_i - f(x_{i,1}, \ldots, x_{i,l}; p_1, \ldots, p_n)\right)^2 \tag{4.2}$$

definiert werden. Mehrdimensionale Modellfunktionen werden jedoch in dieser Arbeit nicht weiter behandelt.

Oft wird die Fehlerberechnung in der Gleichung 4.1 noch um Datenpunkt-abhängige Gewichtungen σ_i erweitert. Dabei ist σ_i die (bekannte) Standardabweichung des Datenpunktes $\mathbf{d}_i$. Der LS-Fehlerfunktion χ ist in diesem Fall als

$$\chi(\mathbf{p}) := \sum_{i=1}^{m} \frac{\left(y_i - f(x_i; p_1, \ldots, p_n)\right)^2}{\sigma_i} \tag{4.3}$$

definiert [27]. Da im Rahmen dieser Arbeit Standardabweichungen von Datenpunkten als nicht bekannt gelten, wird die Fehlerberechnung nach der Gleichung 4.1 ohne die σ_i durchgeführt. Die Gleichungen aus der verwendeten Literatur sind entsprechend angepasst.

Im Folgenden werden drei auf LS basierende Fitting-Methoden vorgestellt. Abschnitt 4.1.1 behandelt das Fitting einer linearen Modellfunktion. In Abschnitt 4.1.2 wird das Fitting nach dem generalisierten LS vorgestellt. Das Fitting nichtlinearer Modellfunktionen wird abschließend in 4.1.3 thematisiert.

4.1.1. Lineares Least-Squares

Das einfachste LS-Fitting ist das *lineare Least-Squares* (LLS). Die Modellfunktion f beschreibt dabei eine Gerade

$$f(x; p_1, p_2) = p_1 + p_2 \cdot x.$$

mit den zu optimierenden Parametern $\mathbf{p} = (p_1, p_2)$. Dieses Fitting wird auch als *Ordinary Least-Squares* (OLS) bezeichnet [31].

Nach Definition 4.1 ist die Fehlerfunktion χ durch

$$\chi(\mathbf{p}) := \sum_{i=1}^{m} \left(y_i - (p_1 + p_2 \cdot x_i)\right)^2 \tag{4.4}$$

gegeben und der Parametervektor ist optimal gewählt, falls $\chi(\mathbf{p})$ minimal ist.

Die Fehlerfunktion sollte ein globales Minimum haben. Ein Maximum sollte dagegen nicht vorhanden sein, da die Parameter beliebig schlecht gewählt werden können. Man kann deshalb versuchen, die Fehlerfunktion nach jedem der beiden Parameter abzuleiten und diese Terme

gleich Null zu setzen. Da nur positive Fehlerwerte auftreten, ist ein optimaler Parametervektor $\mathbf{p}$ dann durch die Gleichungen

$$\begin{aligned} \frac{\partial \chi}{\partial p_1} &= -2 \sum_{i=1}^{m} (y_i - p_1 - p_2 \cdot x_i) &= 0 \\ \frac{\partial \chi}{\partial p_2} &= -2 \sum_{i=1}^{m} (y_i - p_1 - p_2 \cdot x_i) \cdot x_i &= 0 \end{aligned} \tag{4.5}$$

gegeben [21]. Diese lassen sich in ein lineares Gleichungssystem

$$\begin{aligned} \sum_{i=1}^{m} y_i &= p_1 \cdot m &+ p_2 \cdot \sum_{i=1}^{m} x_i \\ \sum_{i=1}^{m} (x_i \cdot y_i) &= p_1 \cdot \sum_{i=1}^{m} x_i &+ p_2 \cdot \sum_{i=1}^{m} (x_i)^2 \end{aligned} \tag{4.6}$$

überführen, das mittels

$$\begin{aligned} p_1 &= \frac{\left(\sum_{i=1}^{m} y_i\right)\left(\sum_{i=1}^{m} (x_i)^2\right) - \left(\sum_{i=1}^{m} x_i\right)\left(\sum_{i=1}^{m} x_i y_i\right)}{m \cdot \sum_{i=1}^{m} (x_i)^2 - \left(\sum_{i=1}^{m} x_i\right)^2} \\ p_2 &= \frac{m \cdot \sum_{i=1}^{m} x_i y_i - \left(\sum_{i=1}^{m} x_i\right)\left(\sum_{i=1}^{m} y_i\right)}{m \cdot \sum_{i=1}^{m} (x_i)^2 - \left(\sum_{i=1}^{m} x_i\right)^2} \end{aligned} \tag{4.7}$$

nach p_1 und p_2 aufgelöst werden kann [21, 36]. Alternativ können die Parameter auch mittels

$$\begin{aligned} p_1 &= \bar{y} - p_2 \cdot \bar{x} \\ p_2 &= \frac{\sum_{i=1}^{m} \tilde{x}_i \tilde{y}_i}{\sum_{i=1}^{m} (\tilde{x}_i)^2} \end{aligned}$$

berechnet werden, wobei $\bar{x}$ und $\bar{y}$ den Mittelwerten aller Punktkoordinaten entsprechen und $\tilde{x}_i = x_i - \bar{x}$ und $\tilde{y}_i = y_i - \bar{y}$ [31]. Die Gleichungen 4.6 werden als *Normalgleichungen* bezeichnet, das durch sie beschriebene Gleichungssystem als *Normalgleichungssystem* [21].

Mit Hilfe des LLS-Fitting lässt sich auch eine exponentielle Modellfunktion auf eine Menge von Datenpunkten anpassen. Dazu wird die Exponentialfunktion durch

$$y = p_1 {p_2}^x \rightarrow \log(y) = a_1 + a_2 x \tag{4.8}$$

in eine Gerade überführt [36]. Die Exponentialfunktion lässt sich dann bei gegebenen a_1, a_2 durch

$$y = \exp(a_1 + a_2 x) \tag{4.9}$$

berechnen. Abbildung 4.1 auf der nächsten Seite stellt das Fitting einer exponentiellen Modellfunktion mittels LLS dar.

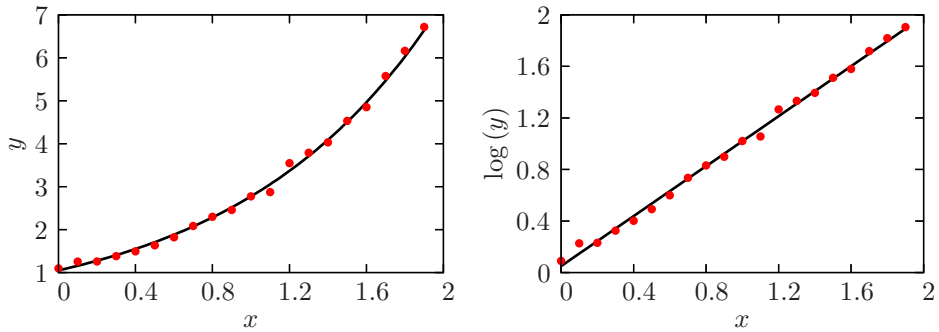

Abbildung 4.1.: Fitting einer Exponentialfunktion (links) mittels linearem Least-Squares (rechts)

4.1.2. Generalisiertes Linear Least-Squares

Die Bildung von Normalgleichungen mittels der Ableitungsgleichungen 4.5 und der Überführung in ein lineares Gleichungssystem wie in den Gleichungen 4.6 lassen sich auf beliebige polynomielle Modellfunktionen

$$f\left(x; p_1, \ldots, p_n\right) = \sum_{i=1}^{n} p_i \cdot x^{i-1}$$

mit $n > 0$ erweitern [36]. Das Fitting polynomieller Modellfunktionen ist aber nur ein Spezialfall des *generalisierten Linear Least Squares* (GLLS). Beim GLLS werden Funktionen betrachtet, die linear bezüglich ihrer Funktionsparameter sind. Eine entsprechende Modellfunktion f besteht beim GLLS aus einer Linearkombination von Basisfunktionen f_i [27]:

$$f\left(x; p_1, \ldots, p_n\right) = p_1 \cdot f_1\left(x\right) + \ldots + p_n \cdot f_n\left(x\right) \tag{4.10}$$

Der Fehlerfunktion χ berechnet sich nach Definition 4.1 auf Seite 30 folgendermaßen:

$$\chi\left(\mathbf{p}\right) := \sum_{i=1}^{m} \left(y_i - \sum_{j=1}^{n} p_j \cdot f_j\left(x_i\right) \right)^2 \tag{4.11}$$

In dieser Form lässt sich das GLLS Problem als Matrix $\mathbf{A}$ mit n Spalten (Basisfunktionen) und m Zeilen (Datenpunkte), so wie einem Vektor $\mathbf{b}$ der Länge m notieren [27, 29]:

$$\mathbf{A} = \begin{pmatrix} f_1\left(x_1\right) & \cdots & f_n\left(x_1\right) \\ \vdots & & \vdots \\ f_1\left(x_m\right) & \cdots & f_n\left(x_m\right) \end{pmatrix} \tag{4.12}$$

$$\mathbf{b} = \left(y_1, \ldots, y_m\right) \tag{4.13}$$

Die Gleichung 4.11 wird wie im Abschnitt 4.1.1 nach allen Parametern abgeleitet. Das Minimum von χ bezüglich der Parameter wird wieder durch Gleichsetzen aller Ableitungsterme mit Null bestimmt. Es resultieren n Gleichungen der Form

$$0 = \sum_{i=1}^{m} \left(y_i - \sum_{j=1}^{n} p_j \cdot f_j\left(x_i\right) \right) \cdot f_k\left(x_i\right) \quad k = 1, \ldots, n\ , \tag{4.14}$$

welche wieder die Normalgleichungen darstellen [27]. Sie können mittels

$$\sum_{j=1}^{n} \alpha_{kj} \cdot p_j = \beta_k$$

auch als Matrixgleichung formuliert werden, hierbei ist

$$\alpha_{kj} = \sum_{i=1}^{m} f_j(x_i) \cdot f_k(x_i) \quad \text{oder äquivalent dazu} \quad [\alpha] = \mathbf{A}^T \cdot \mathbf{A}$$

eine $n \times n$ Matrix und

$$\beta_k = \sum_{i=1}^{m} y_i \cdot f_k(x_i) \quad \text{oder äquivalent dazu} \quad [\beta] = \mathbf{A}^T \cdot \mathbf{b}$$

ein Vektor der Länge n [27]. Die Normalgleichungen des GLLS können außerdem in der Matrixform

$$[\alpha] \cdot \mathbf{p} = [\beta] \quad \text{oder äquivalent dazu} \quad \left(\mathbf{A}^T \cdot \mathbf{A}\right) \cdot \mathbf{p} = \mathbf{A}^T \cdot \mathbf{b}$$

notiert werden [27]. Um das Normalgleichungssystem für $\mathbf{p}$ direkt zu lösen, existiert eine Reihe von Methoden [27]: mittels LU Dekomposition und Rücksubstitution, Choleksy Dekomposition und Gauss-Jordan Elimintaion. Für eine Beschreibung dieser Methoden sei auf die entsprechende Literatur verwiesen [27]. Um der Über- und Unterdeterminiertheit der Gleichungssysteme zu begegnen, kann die *Singular Value Decomposition* (SVD) eingesetzt werden. Auch hier sei auf die entsprechende Literatur verwiesen [18, 27, 32].

4.1.3. Nichtlineares Fitting mittels des Levenberg-Marquardt Algorithmus

GLLS schränkt die Menge der zulässigen Modellfunktionen auf alle Linearkombinationen von Basisfunktionen wie in der Gleichung 4.10 auf der vorherigen Seite ein. Es gibt jedoch auch viele nicht lineare Modellfunktionen. Eine solche Modellfunktion f ist nicht einmal linear bezüglich ihrer Parameter $\mathbf{p}$.

Das Fitting kann aber trotzdem als Minimierung der Fehlerfunktion χ aus Definition 4.1 auf Seite 30 realisiert werden, wobei diese Minimierung auf Grund der Nichtlinearität der zu fittenden Funktionsparameter allerdings nur iterativ erfolgen kann [27]. Der Optimierungsprozess erzeugt also eine Kette von Parameterbelegungen $\mathbf{p}_t$. Ausgehend von einer initialen Parameterbelegung $\mathbf{p}_{t0}$ besteht die Aufgabe darin, sich Schritt für Schritt dem Minimum von χ zu nähern. Jeder Iterationsschritt kann dabei als ein Parameterbelegungsübergang $\mathbf{p}_t \rightarrow \mathbf{p}_{t+1}$ beschrieben werden.

Eine Methode dieser Optimierung stellt der *Levenberg-Marquardt Algorithmus* (LMA) dar. Er wurde von Donald Marquardt entwickelt und geht auf ein Idee von Kenneth Levenberg zurück [27]. Auch beim LMA erfolgt die Minimierung durch Nullsetzen der Ableitungen nach den Parametern. Allerdings muss sich dem Nullpunkt jeder Ableitung iterativ genähert werden. Der LMA vereinigt zwei Methoden, diesen Nullpunkt zu finden [27]: Eine Newton-Methode und Gradientenabstieg.

Um die Optimierung zu beschleunigen wird bei der Newton-Methode angenommen, dass sich χ um das Minimum herum durch eine quadratische Funktion

$$\chi(\mathbf{p}) \approx \tilde{\chi}(\mathbf{p}) \approx \gamma - \mathbf{h} \cdot \mathbf{p} + \frac{1}{2}\mathbf{p} \cdot \mathbf{H} \cdot \mathbf{p} \tag{4.15}$$

approximieren lässt [27]. Dabei ist $\mathbf{h}$ ein Vektor der Länge n und H eine $n \times n$ Matrix. Bei H handelt es sich um die Hessematrix, die die zweiten Ableitungen enthält.

Ist die Approximation gut, so kann diese den Minimierungsprozess sehr beschleunigen. Das Minimum der approximativen Fehlerfunktion kann nämlich mittels

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{P}^{-1} \cdot [-\nabla\chi(\mathbf{p}_t)] \tag{4.16}$$

durch einen einzigen Schritt vom aktuellen Parametervektor $\mathbf{p}_t$ aus erreicht werden [27].

Allerdings kann die Approximation an der betrachteten Stelle auch beliebig schlecht sein. In diesem Fall kann die Optimierung per Gradientenabstieg erfolgen. Dazu wird dem Gradienten in Abwärtsrichtung um eine bestimmte Schrittweite s gefolgt, wobei s klein genug gewählt sein muss um den Nullpunkt nicht zu überspringen [27]:

$$\mathbf{p}_{t+1} = \mathbf{p}_t - s \times \nabla\chi(\mathbf{p}_t) \tag{4.17}$$

Um die Gleichung 4.16 und die Gleichung 4.17 benutzen zu können, müssen der Gradient $\nabla\chi$ und die Hessematrix $\mathbf{H}$ bestimmt werden. Der Gradient entspricht des Ableitens von χ nach allen Parametern p_i. Es resultieren entsprechend der Parametervektorlänge n Terme [27]:

$$\frac{\partial e}{\partial p_k} = -2 \cdot \sum_{i=1}^{m} (y_i - f(x_i; p_1, \ldots, p_n)) \cdot \frac{\partial f(x_i)}{\partial p_k} \quad k = 1, \ldots, n \tag{4.18}$$

Die Elemente der Hessematrix P berechnen sich durch eine zweite partielle Ableitung [27]:

$$\frac{\partial^2 \chi}{\partial p_k \partial p_l} = 2 \cdot \sum_{i=1}^{m} \left(\frac{\partial f(x_i; \mathbf{p})}{\partial p_k} \cdot \frac{\partial f(x_i; \mathbf{p})}{\partial p_k} - (y_i - f(x_i; \mathbf{p})) \cdot \frac{\partial^2 f(x_i; \mathbf{p})}{\partial p_l \partial p_k} \right) \tag{4.19}$$

In beiden Fällen werden partielle Ableitungen der Modellfunktion benötigt. Da die Modellfunktion jedoch bekannt ist, sollten diese in der Regel vorliegen.

Nun wird der Faktor 2 in der Gleichung 4.18 und der Gleichung 4.19 mittels den Definitionen

$$\beta_k \equiv -\frac{1}{2} \cdot \frac{\partial \chi}{\partial p_k} \quad \alpha_{kl} \equiv \frac{1}{2} \cdot \frac{\partial^2 \chi}{\partial p_k \partial p_l} \tag{4.20}$$

eliminiert [27]. Mittels $[\alpha] = \frac{1}{2}\mathbf{H}$ kann dann die Gleichung 4.15 auf der vorherigen Seite in ein lineares Gleichungssystem für Parameteränderungen δp_i überführt werden [27]:

$$\sum_{l=1}^{n} \alpha_{kl} \cdot \delta p_l = \beta_k \quad k = 1, \ldots, n \tag{4.21}$$

Dieses Gleichungssystem kann dann mittels eines geeigneten Verfahrens nach den δp_i aufgelöst werden. Wird dagegen die Gleichung 4.17 (Gradientenabstieg) eingesetzt, so können die δp_i mittels

$$\delta p_i = s \times \beta_i \tag{4.22}$$

berechnet werden [27].

Der LMA vernachlässigt allerdings die zweite Ableitung. Statt wie in der Definition in der Gleichung 4.20 wird α_{kl} beim LMA mittels

$$\alpha_{kl} = \sum_{i=1}^{m} \frac{\partial f(x_i; p_1, \ldots, p_n)}{\partial p_k} \cdot \frac{\partial f(x_i; p_1, \ldots, p_n)}{\partial p_l} \tag{4.23}$$

berechnet [27]. Die zweiten Ableitungen entfallen also. Dies kann aus verschiedenen Günden gerechtfertigt werden [27]. Zum Einen ist der zweite Ableitungsterm im Vergleich zum Term

der ersten Ableitung in der Praxis oft sehr klein, zum Anderen kann der zweite Ableitungsterm bei „Ausreißern“ in den Daten oder einer schlecht passenden Modellfunktion sogar eine destabilisierende Wirkung haben. Darüber hinaus hat diese Vereinfachung keinen Einfluss auf den finalen Parametervektor $\mathbf{p}$, sondern ändert nur die Iterationskette der $\mathbf{p_i}$ dorthin.

Bis jetzt wurde die Schrittweite s noch nicht bestimmt. Jedes Parameter kann eine andere Dimension bzw. Größenordnung besitzen. Insbesondere können die Parameter mit unterschiedlichsten physikalischen Einheiten verknüpft sein, was wiederum beim Entwurf der Modellfunktion eine wichtige Rolle gespielt haben könnte. So hätte beispielsweise eine Änderung der Parameter der Modellfunktionen $p_1 + {p_2}^2$ und $p_1 + 1000 \cdot {p_2}^2$ bei gleicher Schrittweite unter Umständen sehr unterschiedlich starke Auswirkungen. Ein einzelner Gradient kann keine Dimensionsinformation geben, deshalb werden beim LMA die Diagonalelemente der Matrix $[\alpha]$ verwendet. Die Dimension von p_i wird bestimmt durch das Verhältnis von β_i und $\alpha_{i,i}$ [27]. Der resultierende Wert könnte aber noch zu groß sein, weshalb noch durch ein $\lambda > 0$ geteilt wird. Die Gleichung 4.22 auf der vorherigen Seite wird dann durch die folgende Gleichung beschrieben [27]:

$$\delta p_i = \frac{1}{\lambda \cdot \alpha_{i,i}} \cdot \beta_i \quad \text{äqivalent zu} \quad \lambda \cdot \alpha_{i,i} \cdot \delta p_i = \beta_i \tag{4.24}$$

Durch die Gleichung 4.23 auf der vorherigen Seite ist dabei sichergestellt, dass $\alpha_{i,i}$ positiv ist.

Der LMA fasst die Gleichung 4.23 auf der vorherigen Seite und die Gleichung 4.24 zusammen, indem eine neue Matrix $[\alpha']$ mittels

$$\alpha'_{i,k} = \begin{cases} \alpha_{i,i} \cdot (1+\lambda) & \text{falls} \quad i = k \\ \alpha_{i,k} & \text{falls} \quad i \neq k \end{cases} \tag{4.25}$$

definiert wird und die beiden Gleichungen durch

$$\sum_{i=1}^{n} \alpha'_{i,k} \cdot \delta p_i = \beta_k \quad k = 1, \ldots, n \tag{4.26}$$

ersetzt werden [27]. Der Pararameter λ entscheidet darüber, ob die Gleichung 4.23 auf der vorherigen Seite oder die Gleichung 4.24 zur Optimierung von $\mathbf{p}$ herangezogen wird. Ist λ groß, so wird die α' Matrix diagonal dominant und die Gleichung 4.26 geht in die Gleichung 4.24 über, ist λ dagegen klein (nahe Null), so geht die Gleichung 4.26 in die Gleichung 4.23 auf der vorherigen Seite über [27].

Ausgehend von einer initialen Parameterbelegung $\mathbf{p}$ und unter der Voraussetzung, dass die partiellen Ableitungen von f nach den p_i bekannt sind, kann nun der eigentliche Iterationsprozess als der Algorithmus 4.1.1 formuliert werden [27]. Dabei wird neben λ noch ein konstanter Parameter μ verwendet zur Steuerung der λ-Veränderung verwendet. Dieser Parameter sollte mit einem größeren Wert wie beispielsweise $\mu = 10$ belegt werden [27].

Das Abbruchkriterium wurde bisher noch nicht formuliert. Eine Konvergenz, selbst unter Einbeziehung der durch die maschinenabhängigen Beschränkungen (Fließkomma-Genauigkeit, Rundungsfehler) gegebenen Grenzen kann nicht erwartet werden. Dafür gibt es zwei Gründe [27]: Zum Einen stellt das Minimum der Fehlerfunktion nur ein statistisches Optimum der Parameterbelegung $\mathbf{a}$ dar. Zum Anderen kann die Suche um das Minimum zirkulieren, ohne es direkt zu treffen. Stattdessen wird die Optimierung abgebrochen, wenn die Verbesserung ein oder zweimal einen Grenzwert unterschreitet [27]. Eine zusätzliche Abbruchbedingung kann natürlich auch durch die Angabe einer maximalen Zahl von Iterationsschritten erfolgen.

Als Beispiel sei die folgende Modellfunktion g mit den Parametern p_1 und p_2 gegeben:

$$g(x) = p_1 \cdot (p_2 + x)^2 \tag{4.27}$$

Algorithmus 4.1.1 Levenberg-Marquardt-Algorithmus

1: Berechne $\chi(\mathbf{p})$.
2: Wähle einen kleinen initialen Startwert für λ, beispielsweise $\lambda = 0.001$.
3: **while** Abbruchkriterium nicht erfüllt **do**
4: Löse das durch die Gleichung 4.26 gegebene Gleichungssystem für $\delta\mathbf{p}$.
5: Berechne $\chi(\mathbf{p} + \delta\mathbf{p})$.
6: **if** $\chi(\mathbf{p} + \delta\mathbf{p}) \geq \chi(\mathbf{p})$ **then**
7: Setze $\lambda \to \lambda \cdot \mu$.
8: **else**
9: Setze $\lambda \to \lambda/\mu$.
10: Setze den neuen Parametervektor $\mathbf{p} \to \mathbf{p} + \delta\mathbf{p}$
11: **end if**
12: **end while**

Die partiellen Ableitungen lauten:

$$\frac{\partial g(x)}{\partial p_1} = (p_2 + x)^2 \quad \frac{\partial g(x)}{\partial p_2} = 2 \cdot p_1 \cdot (p_2 + x) \tag{4.28}$$

Abbildung 4.2 zeigt das Fitting ausgehend von dem initialen Parametervektor $(1, 0)$.

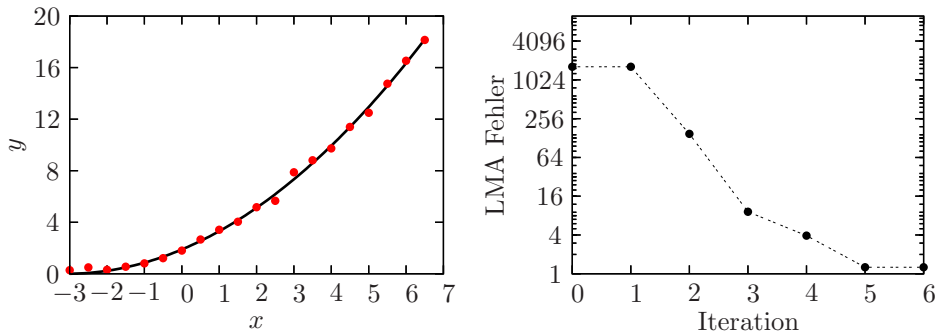

Abbildung 4.2.: Fitting einer quadratischen Funktion mittels des LMA (links) und der zugehörige Fehlerverlauf über die Iterationsschritte (rechts)

4.2. Evolutionäre Algorithmen

Die *evolutionären Algorithmen* (EA) gehören zur Gruppe der *randomisierten Algorithmen*. Randomisierung bedeutet, dass ein Algorithmus nicht deterministisch verläuft, sondern sein Verhalten von Zufallszahlen abhängt. Die Besonderheit von EAs besteht darin, dass sie sich an Prinzipien der Natur orientieren. Dies sind die Prinzipen Reproduktion, Muation und Selektion. Mit ihrer Hilfe soll sich eine virtuelle Population nach und nach den Umgebungsbedingungen anpassen.

Die Aufgabe eines EA besteht darin, ein Optimierungsproblem zu lösen. Das zu optimierende Problem gibt die „Umweltbedingungen“ vor. Jede Lösung wird dabei als ein Individuum innerhalb einer Population von Lösungen aufgefasst. Ausgehend von einer initialen Startpopulation werden dann so lange „Generationsübergänge“ durchlaufen, bis ein zu definierendes Abbruchkriterium erfüllt ist.

Aus evolutionstheoretischer Sicht gibt es zwei Gesichtspunkte, unter denen ein Individuum betrachtet werden kann [11]: Der *Genotyp* (genetische Erbinformation) und der *Phenotyp* (Verhalten in der Umwelt). Zwischen Genotyp und Phenotyp bestehen dabei verschiedene Beziehungen. So beeinflusst der Genotyp den Phenotyp, beispielsweise kann eine Maus in der Regel nicht fliegen. Umgekehrt kann ein ungünstiger Phenotyp zur Auslöschung der zugehörigen Genotypen führen. Phenotypen können im Rahmen der Selektion andere Phenotypen verdrängen. Ein Beispiel ist die Einführung nicht heimischer Tiere, für die keine natürlichen Feinde existieren, und die so nach und nach heimische Tiere verdrängen. Außerdem werden Genotypen bei der Fortpflanzung offensichtlich durch die Genotypen der erzeugenden Individuen beeinflusst.

Im Rahmen der EAs wird ein Individuum durch ein *Chromosom* repräsentiert. Ein Chromosom besteht aus *Genen*, die seine einzelnen Charakteristika darstellen [10]. Angenommen das Optimierungsziel bestehe darin, die Drehknöpfe eines unbekannten Synthesizers so einzustellen, dass ein möglichst unverfälschter Sinuswellenton entsteht. Dann kann jede Drehknopfstellung als ein Gen aufgefasst werden.

Elementarer Bestandteil jedes EA ist die *Fitnessfunktion*. Sie weist jedem Individuum abhängig von den aktuellen Umgebungsbedingungen eine numerische Bewertung zu. In der Regel spielt sie bei dem Selektionsprinzip eine wichtige Rolle. Der Funktionswert wird als *Fitness* des entsprechenden Individuums bezeichnet.

Eine wichtige Eigenschaft der Fitnessfunktion besteht darin, dass sie eigentlich gar nicht bekannt sein muss. Es genügt wenn sie als „Blackboxfunktion“ vorliegt. So könnte sich etwa im Fall des Synthesizerbeispiels die Fitness einer Drehknopfstellungsliste durch die Summe der Nicht-Null-Einträge der Fouriertransformierten einer Aufnahme des resultierenden Tons unter Einbeziehung eines bestimmten Schwellwerts berechnen.

Im Folgenden werden zwei Realisierungen von EAs vorgestellt: Evolutionsstrategien und genetische Algorithmen. Für weitere EA-Varianten sei beispielsweise auf entsprechende Literatur verwiesen [10].

4.2.1. Genetische Algorithmen

Ein *genetischer Algorithmus* (GA) simuliert genetische Evolution. Der Fokus liegt also auf dem Genotyp.

Jedes Individuum $\mathbf{I}$ wird durch einen Vektor $\mathbf{x}$ (Chromosom) von n Variablen (Genen) eines bestimmten Typs T_i repräsentiert. Die Chromosomen haben häufig die gleiche Länge und die $\mathbf{x}$ stellen die zu optimierenden Parametervektoren der Fitnessfunktion

$$f : T_1 \times \ldots \times T_n \to \mathbb{R}$$

dar. Beim klassischen GA ist jedes Gen eine Bitkette fester Länge (binäre Repräsentation), also $x_i \in \{0,1\}^{l_i}$. Es sind aber aber auch andere Informationsrepräsentationen wie reelle Zahlen möglich. Für binären Gene wird eine Repräsentation im so genannten *Graycode* bevorzugt. Diese Darstellung hat im Gegensatz zur gewöhnlichen Darstellung von binären Zahlen den Vorteil, dass sich zwei aufeinander folgende Zahlen nur in einem Bit unterscheiden.

Zu jeder Generation g wird die Population $P_g = \{\mathbf{I}_1, \ldots, \mathbf{I}_\mu\}$ betrachtet. Der generelle GA kann als der Algorithmus 4.2.1 notiert werden [10]. Populationsgröße, Elternzahl und Nachkommenzahl werden in der Regel fest gewählt. Das einfachste Abbruchkriterium ist das Erreichen einer maximalen Generationsanzahl. Andere Möglichkeiten sind die Definition eines minimalen Funktionswertes von f oder die Anforderung einer minimalen Verbesserung der Fitnesswerte von einer Generation zur Anderen.

Algorithmus 4.2.1 Genetischer Algorithmus

1: Starte mit initialer Population P_0, setze $g = 0$.
2: Berechne den Fitnesswert für jedes $I \in P_0$.
3: **while** Abbruchkriterium nicht erfüllt **do**
4: Wähle Eltern aus P_g (Selektion).
5: Rekombiniere gewählte Eltern um Nachkommen N_g zu erzeugen (Rekombination).
6: Mutiere die Nachkommen N_g (Mutation).
7: Gehe über zur nächsten Generation: $g = g + 1$.
8: Berechne den Fitnesswert für jedes $I \in P_g$.
9: Wähle neue Population P_g aus $P_{g-1} \cup N_{g-1}$ (Selektion).
10: **end while**

Die im Algorithmus auftauchenden Mechanismen Selektion (4,9), Rekombination (5) und Mutation (6) können als Operatoren aufgefasst werden. Der Rest dieses Abschnitts geht kurz auf einige Realisierungen dieser Operatoren ein.

4.2.1.1. Selektion

Die *Selektion* wählt aus einer Menge $M = \{I_1, \ldots, I_n\}$ m Individuen aus. Einige Möglichkeiten, diese Auswahl zu treffen, sind [10]:

Randomisierte Selektion Die m Individuen werden gleichverteilt zufällig gewählt. Diese Form der Selektion macht natürlich wenig Sinn für die Auswahl der nächsten Generation im Algorithmusschritt (9).

Proportionale Selektion Ein Individuum I_k wird mit der Wahrscheinlichkeit

$$1 - \frac{f(I_k)}{\sum\limits_{j=1}^{n} f(I_j)}$$

gewählt[1].

Ein Spezialform ist das so genannte *Rouletterad*. Jeder Fitnesswert wird mittels $1 - f(I_k)/max$ normiert. Der Wert max stellt hierbei den größten auftretenden Fitnesswert dar. Ein I_k wird dann nach dem Algorithmus 4.2.2 gewählt.

Algorithmus 4.2.2 Rouletterad-Wahl

1: Initialisiere eine Summe $s = f(I_1)$ und ein Zähler $z = 1$.
2: Bestimme eine gleichverteilte Zufallszahl $z \sim \mathcal{U}(0, 1)$.
3: **while** $s < u$ **do**
4: $z = z + 1$.
5: $s = s + f(I_z)$.
6: **end while**
7: Gebe I_z aus.

Contest Selektion Eine Gruppe von Individuen wird zufällig gewählt und das mit der besten Fitness innerhalb dieser Gruppe gewinnt. Im Fall von Cross-Over-Rekombination (s.u.) werden zwei Gruppen verwendet.

[1] In der Literatur wird ein Maximimierungsproblem behandelt, während hier ein Minimierungsproblem behandelt wird. Deshalb findet sich in der Literatur eine abweichende Formel.

Rangbasierte Selektion Die Wahrscheinlichkeit für die Wahl eines Individuums wird nicht durch den absoluten Fitnesswert, sondern durch die Position in der Rangliste bestimmt.

Elite Selektion Meist ist hiermit gemeint, dass die m besten Individuen ausgewählt werden.

4.2.1.2. Rekombination

Die *Rekombination* besteht beim GA darin, aus mindestens zwei Elternindividuen auf Basis der Chromosomrepräsentation Nachkommenindividuen zu erzeugen. Dabei wird versucht, das biologische Prinzip des *Cross-Over* zu simulieren.

Häufig werden aus zwei Eltern zwei Nachkommen erzeugt. Sind die Gene in Form von Bitketten gegeben, kann das biologische Prinzip des Cross-Over auf verschiedene Weise nachempfunden werden [10]:

Uniformes Cross-Over Es wird eine Cross-Over-Rate $p_c \in [0, 1]$ festgelegt. Pro Gen wird eine Bitkette der gleichen Länge erzeugt. Diese Bitkette wird als *Cross-Over-Maske* bezeichnet. Für jedes Bit dieser Maske wird eine Zufallszahl $z \sim \mathcal{U}(0, 1)$ berechnet. Falls $z \leq p_c$ wird das entsprechende Bit auf Eins, andernfalls auf Null gesetzt.

Die beiden Nachkommen sind zunächst Duplikate der Eltern. An jeder Maskenposition, die den Wert Eins hat, werden dann jedoch die entsprechenden Bits zwischen den Nachkommen vertauscht.

Ein-Punkt Cross-Over Für jedes Gen der Länge m wird eine Zufallszahl $z \sim \mathcal{U}(1, m-1)$ berechnet. Ein Nachkomme erhält für das jeweilige Gen die Bits $1, \ldots, z$ des ersten Elternindividuums und die Bits $z+1, \ldots, m$. Der andere Nachkomme erhält die nicht vergebenen Bitpositionen um seine Gene zu füllen.

Zwei-Punkt Cross-Over Es werden zwei Zufallszahlen $z_1, z_2 \sim \mathcal{U}(1, m)$ pro Gen der Länge m berechnet. Ein Nachkomme erhält die Bits an den Positionen $z_1, \ldots, z_2$ des ersten Elternindividuums und die Bits an den Positionen $i \notin [z_1, z2]$ des zweiten Elternindividuums. Der andere Nachkomme erhält die verbleibenden Bits.

Sind die Gene reellwertig, so ist eine einfache Rekombinationsmöglichkeit dadurch gegeben, sich auf die zufällige Verteilung ganzer Gene zu beschränken. Eine Möglichkeit von „Geninternem“ Cross-Over besteht darin, pro Gen zwei Zufallszahlen $z_1, z_2 \sim \mathcal{U}(1, 0)$ zu berechnen und das entsprechende Gen der Nachkommen auf folgende Weise zu festzulegen [10]:

$$n_1 := z_1 e_1 + (1 - z_1)\, e_2$$
$$n_2 := (1 - z_2)\, e_1 + z_2 e_2$$

Hierbei sind n_1, n_2 die Nachkommengene und e_1, e_2 die Elterngene. Es gibt zahlreiche andere Cross-Over-Techniken für reellwertige Gene, wozu jedoch auf die entsprechende Literatur verwiesen sei [14].

4.2.1.3. Mutation

Die Motivation für die *Mutation* besteht darin, Genmaterial in Individuen einzubringen, das in einer bestehende Population nicht vorhanden ist und sich auch nicht durch Rekombination bilden lässt. In GAs wird ein Mutationsparameter $p_m \in [0, 1]$ zur Steuerung der Mutationsrate benutzt. Der Wert sollte recht klein sein, damit „gute“ Gene nicht zu oft bei der Nachkommenerzeugung zerstört werden. Die Mutation wird immer pro Gen und pro Nachkomme durchgeführt.

Sind die Gene binär, so gibt es folgende Möglichkeiten der Mutation [10]:

Zufällige Mutation Für jedes Bit wird eine Zufallszahl $z \sim \mathcal{U}(0,1)$ berechnet. Gilt $z \leq p_m$, so wird das Bit invertiert.

Geordnete Mutation Es werden zwei Positionen des betrachteten gleichverteilt zufällig berechnet. Nur Bits in diesem Bereich werden wie wie bei der zufälligen Mutation behandelt.

Handelt es sich dagegen um reellwertige Gene, so wird zunächst wieder eine Zufallszahl $z_1 \sim \mathcal{U}(0,1)$ berechnet. Nur wenn $z_1 \leq p_m$ gilt, wird zusätzlich eine $\mathcal{N}\left(0,\sigma^2\right)$-verteilte Zufallszahl z_2 berechnet und zu dem aktuellen Genwert hinzu addiert [10]. In der Regel wird die Varianz der Normalverteilung von der Fitness des Individuums abhängig gemacht: Eine gute Fitness führt zu niedriger Varianz und umgekehrt [10].

4.2.2. Evolutionsstrategien

Eine *Evolutionsstrategie* (ES) verfolgt den darwinistischen Gedanken des „Survival of the Fitest“: Über mehrere Generationen hinweg setzen sich gerade die Individuen durch, die am besten an die Umweltbedingungen (Fitnessfunktion) angepasst sind. Die ES konzentriert sich im Vergleich mit dem GA stärker auf den Phenotyp, allerdings wird auch hier durch die Möglichkeit der Rekombination der Genotyp in Betracht gezogen.

Ein Individuum $\mathbf{I}$ kann formal als ein Vektor $(\mathbf{x}, \boldsymbol{\sigma})$ aufgefasst werden, wobei $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ und $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n) \in {\mathbb{R}_{>0}}^n$. Die x_i werden als *Objektparameter* bezeichnet und stellen die zu optimierenden Parameter von f dar. Die σ_i werden als *Strategieparameter* bezeichnet und können als „Mutationswilligkeit“ von $\mathbb{I}$ aufgefasst werden. Bei der ES werden Fitnessfunktionen der Form

$$f : \mathbb{R}^n \to \mathbb{R}$$

eingesetzt, die nur die Objektparameter $\mathbf{x}$ betrachten. Die Dimension des von der ES zu lösenden Optimierungsproblems ist demnach n.

Zu jeder Generation g wird die Population $P_g = \{\mathbf{I}_1, \ldots, \mathbf{I}_\mu\}$ (Eltern) betrachtet. Aus dieser Generation gehen λ neue Individuen (Nachkommen) hervor. Durch Selektion wird die Population P_{g+1} allerdings wieder auf μ Individuen begrenzt.

Man unterscheidet $(\mu + \lambda)$- und (μ, λ)-ES [33]. Bei der $(\mu + \lambda)$-ES können Eltern in die nächste Population P_{g+1} gelangen. In der (μ, λ)-ES ist ihre Lebensdauer jedoch auf eine Generation begrenzt. Im zweiten Fall muss natürlich $\lambda \geq \mu$ gelten. Der Übergang von P_g auf P_{g+1} verläuft nach folgende Schritten:

1. Aus den μ Eltern werden λ Nachkommen erzeugt. Bei einer (μ, λ)-ES erzeugt jedes Elternindividuum im Schnitt λ/μ Nachkommen [33]. Bei einer $(\mu + \lambda)$-ES können die Eltern zufällig gewählt werden [10]. Entweder die Nachkommen stellen zunächst nur Duplikate dar, oder es wird eine Rekombination (s.u.) durchgeführt. Die Menge der Nachkommen sei mit N bezeichnet.

2. *Mutation*: Die Objektparameter der Nachkommen werden gemäß der Strategieparameter (s.u.) zufällig geändert.

3. *Selektion*: Die Menge K enthält alle Kandidaten. Bei einer (μ, λ)-ES ist $K = N$, bei einer $(\mu + \lambda)$-ES ist $K = N \cup P_g$. Aus X werden mit Hilfe der Fitnessfunktion f die μ besten Individuen ausgewählt [33]. Sie bilden P_{g+1}.

Diese Schritte werden so lange wiederholt bis ein geeignetes Abbruchkriterium eintritt. Das Abbruchkriterium kann wie bei einem GA festgelegt werden.

Soll in Schritt 1 eine Rekombination erfolgen, gibt es zwei Möglichkeiten [10, 33]. Es kann entweder ein Cross-Over wie bei GAs mit reellwertigen Genen erfolgen, oder es werden alle ausgewählten Eltern in Betracht gezogen werden (Multi-Cross-Over). Sowohl Objekt- wie auch Strategieparameter können rekombiniert werden. Parametervektoren werden entweder aus unveränderten Elemente unterschiedlicher Eltern zusammengesetzt oder entstehen durch Mittelwertbildungen.

Die Mutation eines Objektparametervektors $\mathbf{x}$ in Schritt 2 erfolgt in der Regel mit Hilfe einer Normalverteilung und den Strategieparametern $\boldsymbol{\sigma}$ [11][2]:

$$x_i{}' := x_i + \mathcal{N}\left(0, \sigma_i{}^2\right) \tag{4.29}$$

Die Normalverteilung ist symmetrisch, also haben mutationsbedingte Änderungen eines Objektparameters für beide Richtungen die gleiche Wahrscheinlichkeit. Eine Alternative besteht darin, eine asymmetrische Verteilungsfunktion zu benutzen [15].

In theoretischen Modellen (Kugelmodell, Korridor) können die Strategieparameter nahezu optimal bestimmt werden [33]. Bei der Minimierung unbekannter Funktionen können sich konstante Belegungen aber sehr negativ auswirken. Sind die Varianzen zu niedrig, konvergiert die Suche nur sehr langsam und wird vielleicht viel zu früh abgebrochen. Sind sie zu hoch steigt das Risiko ein Optimum ständig zu überspringen.

Einen Ausweg bietet die 1/5-*Erfolgsregel*, die sich bei vielen praktischen Problemen als sehr effektiv erwies [33]. Dabei wird in bestimmten Zeitabständen die Erfolgsrate der Mutationen betrachtet. Wenn die Anzahl der Verbesserungen im Verhältnis zu der Anzahl der Mutationen 1/5 überschreitet, werden die Varianzen $\boldsymbol{\sigma}$ erhöht. Fällt die Erfolgsrate unter 1/5, werden die Varianzen verringert. Beträgt sie 1/5, so bleiben die Varianzen unverändert.

Eine andere Möglichkeit der Strategieparameteroptimierung bietet die so genannte *Selbstadaption*. Hierbei werden in Schritt 2 zunächst die Strategieparameter mutiert. Erst danach erfolgt eine Mutation der Objektparameter. Es gibt zwei Vorschläge, die Strategieparametermutation durchzuführen [10]. Der erste Vorschlag

$$\sigma_i{}' := \sigma_i e^{\sqrt{n} \cdot z} \tag{4.30}$$

stammt von Hans-Paul Schwefel, der zweite Vorschlag

$$\sigma_i{}' := \sigma_i + \left(1 + \sqrt{n} \cdot z\right) \tag{4.31}$$

von David B. Fogel. In beiden Fällen ist $z \sim \mathcal{N}(0, 1)$. Die Selbstadaption liegt im Vergleich zur 1/5-Erfolgsregel dichter an dem Ziel der „natürlichen Selbstregulierung“, da bis auf die Angabe von μ, λ, einer initialen Population und eines Abbruchkriteriums keine Außeneinwirkung mehr auf den Verlauf einer ES ausgeübt wird.

Eine Erweiterung stellt die Aufnahme eines zusätzlichen Strategieparametersatzes von $n(n-1)/2$ Lagewinkeln, die die Mutationen korrelieren sollen, dar. Für Details sei auf entsprechende Literatur verwiesen [10, 15]. An dieser Stelle wird dieses Kapitel, wie auch der Grundlagenteil dieser Arbeit abgeschlossen.

[2] Für den Fall, dass sich nur $\mathcal{N}(0, 1)$-verteilte Zufallszahlen erzeugen lassen (z.B. bei Standardbibliotheken einer Programmiersprache), kann man sich folgendermaßen behelfen [2]: Ist Z eine $\mathcal{N}(0, 1)$-verteilte Zufallszahl, so ist $X = \sigma Z + \mu$ eine $\mathcal{N}\left(\mu, \sigma^2\right)$-verteilte Zufallszahl.

5. Daten

Die in dieser Arbeit entwickelten Methoden wurden auf reale Daten angewendet. Diese Daten entstanden unter Verwendung eines ^{63}Ni-IMS (vgl. Abschnitte 2.2 und 2.3), wobei eine MCC als Vortrennungseinheit (vgl. Abschnitt 2.4) eingesetzt wurde. In diesem Kapitel werden die verwendeten Daten beschrieben und einige formale Definitionen der mit den Daten verbundenen Informationen eingeführt. Darüber hinaus werden in den Abschnitten 5.4, 5.5 und 5.6 drei einfache Vorverarbeitungsschritte vorgestellt.

5.1. Messungen

Das ISAS - Institute for Analytical Sciences stellte für diese Arbeit verschiedene Datensätze zur Verfügung. Diese *Rohdaten* bestehen aus einer Reihe aufgezeichneter Datenpunktsequenzen. Eine solche Sequenz wird im Folgenden als *Spektrum* bezeichnet. Allerdings stellt jedes Spektrum eine Mittelung von einer festen Anzahl „wirklicher“ Spektren dar. Dafür gibt es zwei Gründe. Zum Einen soll dadurch schon während der Aufzeichnung eine Rauschminderung erzielt werden. Zum Anderen können mit der zur Aufzeichnung eingesetzten Software nur bis zu 1000 Spektren pro Datensatz geschrieben werden, da es sich um eine DOS/WINDOWS 3.1-Anwedung handelt und jedes Spektrum in einer eigenen Datei mit einem dezimalen Index als Dateinamenerweiterung gespeichert wird[1]. In dieser Arbeit wird dieser Aspekt jedoch vernachlässigt und ein Spektrum als tatsächliche IMS-Analyse einer Schaltgitteröffnung aufgefasst.

In den Rohdaten liegt jedes Spektrum als Textdatei vor. Jede dieser Textdateien enthält neben (Driftzeit, Intensität)-Tupeln noch einen auf Sekunden gerundeten Datumseintrag, aus dem sich eine Retentionszeit für das entsprechende Spektrum ermitteln lässt[2]. Die Abtastrate und die Anzahl der Abtastpunkte bezüglich der Driftzeiten ist konstant. Bei allen verwendeten Messungen wurden pro Messung etwa 500 Spektren im Abstand von etwa einer Sekunde aufgezeichnet.

Zur Weiterverarbeitung werden die Rohdaten in ein kompakteres Datenformat, hier als Binärdaten bezeichnet, konvertiert. Dabei werden bei Bedarf schon die in den Abschnitten 5.4, 5.5 und 5.6 vorgestellten Transformationen dieser Daten vorgenommen. Außerdem werden unbrauchbare Spektren verworfen. In der Regel betrifft dies das erste Spektrum, da hier starke Störsignale auftreten. Der resultierende Datensatz wird im Folgenden als *Messung* bezeichnet und lässt sich formal definieren:

Definition 5.1 (Messung) *Eine Messung ist ein Tupel* $(\mathbf{D}, \mathbf{d}, \mathbf{r})$. *Hierbei ist* $\mathbf{D}$ *eine* $n \times m$

[1] Inzwischen wird eine neuere Software benutzt. Die in dieser Arbeit verwendeten Daten wurden jedoch alle durch die ältere Software aufgezeichnet.

[2] Damit ergibt sich das Problem, dass in Ausnahmefällen Spektren mit identischen Retentionszeiten auftreten. Dieses Problem wurde so gelöst, indem immer nur das erste Spektrum einer Folge von Spektren mit gleicher Retetionszeit in die Matrix aufgenommen wird und die anderen Spektren verworfen werden. Die Idee, das Modifikationsdatum der Textdatei an Stelle des ausgelesenen Datums zu verwenden, brachte keine Verbesserung. Tatsächlich ergaben sich sogar noch mehr identische Retentionszeiten.

Matrix

$$\begin{pmatrix} x_{1,1} & \cdots & x_{n,1} \\ \vdots & & \vdots \\ x_{1,m} & \cdots & x_{n,m} \end{pmatrix}$$

von Intensitäten $x_{i,j}$ (in V), $\mathbf{d}$ ein Vektor $(d_1, \ldots, d_n)$ von zugehörigen Driftzeiten (in ms) und $\mathbf{r}$ ein Vektor $(r_1, \ldots, r_m)$ von zugehörigen Retentionszeiten (in s). Jeder Intensitätsdatenpunkt $x_{i,j}$ korrespondiert zur Driftzeit d_i und zur Retentionszeit r_j.

Ein einzelnes Spektrum kann dann im Wesentlichen durch einen Zeilenvektor der Matrix beschrieben werden.

Definition 5.2 (Spektrum) *Sei $(\mathbf{D}, \mathbf{d}, \mathbf{r})$ eine Messung wie in Definition 5.1. Ein Spektrum ist dann ein Tupel $(\mathbf{x}, \mathbf{d}, r)$. Betrachtet sei das i-te Spektrum mit $1 \leq i \leq m$. Dann ist $\mathbf{x}$ der i-te Zeilenvektor von $\mathbf{D}$, $\mathbf{d}$ der Driftzeitenvektor der Messung und $r = r_i$ die zugehörige Retentionszeit.*

Zusätzlich wird der Begriff *Driftzeitschnitt* definiert. Hier werden Spaltenvektoren von $\mathbf{D}$ betrachtet.

Definition 5.3 (Driftzeitschnitt) *Sei $(\mathbf{D}, \mathbf{d}, \mathbf{r})$ eine Messung wie in Definition 5.1. Ein Driftzeitschnitt ist dann ein Tupel $(\mathbf{x}, \mathbf{r}, d)$. Betrachtet sei der i-te Driftzeitschnitt mit $1 \leq i \leq n$. Dann ist $\mathbf{x}$ der i-te Spaltenvektor von $\mathbf{D}$, $\mathbf{r}$ der Retentionszeitenvektor der Messung und $d = d_i$ die zugehörige Driftzeit.*

5.2. Messungsparameter

Es existieren eine Reihe von Variablen, die Einfluss auf die aufgezeichneten Daten haben. Deshalb wird jeder Messung außerdem ein Satz von *Messungsparamtern* zugeordnet. Diese waren leider nur in Form von handgeschriebenen „Logbüchern“ vorhanden[3]. Dieses Manko stellte sich aber in einer frühen Phase dieser Arbeit in gewisser Weise als vorteilhaft heraus, da eine anfängliche Nichtverfügbarkeit eine alternative Driftzeitennormalisierung notwendig machte. Ein Teil der Messungsparameter wurde in dieser Arbeit jedenfalls als elektronisch auslesbar vorausgesetzt. Dies wurde so gelöst, dass zu jeder (binären) Messdatendatei eine zusätzliche Messungsparameterdatei vorliegen muss.

Im Folgenden werden die wichtigsten Messungsparameter kurz vorgestellt und ihre typischen Belegungen für die zur Konzentrationsabhängigkeitanalyse verwendeten Messungen angegeben. Für weitere Informationen sei auf das Kapitel 2 verwiesen.

Gitteröffnungszeit Dies ist die Schaltgitteröffnungszeit (in μs). Alle verwendeten Messungen wurden mit einer Gitteröffnungszeit von 300 μs aufgezeichnet.

Gitterintervall Dies ist das Intervall (in ms) von einer Gitteröffnungszeit zur nächsten. Alle verwendeten Messungen wurden mit einem Gitterintervall von 100 ms aufgezeichnet. Allerdings umfasst jedes Spektrum nur den Bereich der ersten 50 ms.

Feldstärke Dies ist die elektrische Feldstärke (in V/cm) des Feldes im Driftraum. Alle verwendeten Messungen wurden mit einer gleich eingestellten Feldstärke aufgezeichnet.

[3] Nicht alle hier vorgestellten Messungsparameter sind Bestandteil der Logbücher. Allerdings verfügt jedes IMS über ein eigenes Logbuch. Inzwischen wird der gesamte Datenbestand inklusive der Messungsparameter in einer Datenbank aufgenommen.

Zunächst wurde diese mit 326 V/cm angegeben. Dieser Wert wurde jedoch auf 330 V/cm korrigiert.

Polarität Sie gibt den Modus des IMS-Betriebs an: Positiver oder negativer Betrieb. Alle verwendeten Messungen wurden bei positiver Polarität aufgezeichnet.

Driftraumlänge Die Driftraumlänge (in cm) legt die Länge des zeitlichen Analysebereichs fest. Bei allen verwendeten Messungen wurde ein 12 cm langer Driftraum eingesetzt.

Mittelung Sie gibt die Anzahl der „wirklichen“ Spektren an, deren Mittelung ein aufgezeichnetes Spektrum ergibt. Dabei werden die Mittelwerte immer disjunkt gebildet, d.h. jeder Datenpunkt wird nur zur Bildung eines aufgezeichneten Spektrums benutzt. In allen verwendeten Messungen wurden je 10 Datenpunktreihen gemittelt.

Abtastpunkte Dies ist die Anzahl der aufgezeichneten Abtastpunkte pro Spektrum. Alle verwendeten Messungen wurden mit 2000 Abtastpunkten pro Spektrum aufgezeichnet.

Abtastfrequenz Sie gibt die Abtastpunktabstände an. Der Abstand wird mittels Gitterintervall/Abtastpunkte berechnet. Bei allen verwendeten Messungen wurde eine Abtastfrequenz von 4000 eingesetzt. Dies führt zusammen zu einem Abtastpunktabstand von 0.025 ms und dem aufgezeichneten Bereich von 50 ms.

Driftgas Dies bezeichnet das verwendete Driftgas, das den Ionen im Driftraum entgegen strömt. Bei allen verwendeten Messungen wurde Luft als Driftgas eingesetzt.

Driftgasdurchfluss Dies ist die Durchflussgeschwindigkeit (in mL/min) des Driftgases. Alle verwendeten Messungen wurden bei einem Driftgasfluss von 100 mL/min aufgezeichnet.

Trägergas Dies bezeichnet das verwendete Trägergas, das die Analyte in das IMs einführt. Bei den verwendeten Messungen wurde entweder Luft oder Stickstoff als Trägergas eingesetzt.

Trägergasdurchfluss Dies ist die Durchflussgeschwindigkeit (in mL/min) des zu analysierenden Gasgemisches an. Alle verwendeten Messungen wurden bei einem Trägergasfluss von 100 mL/min aufgezeichnet.

Druck Dies ist der Umgebungsdruck (in hPa). In den Logbüchern waren leider nur auf Meerespiegel normierte Werte angegeben, die in der Regel zwischen 985 und 1015 hPa lagen. Allerdings wurde von einer Mitarbeiterin des ISAS - Institute for Analytical Sciences (Sabine Bader) die Umrechnungsformel

$$p = p_N \cdot \exp\left(\frac{-9.81 \cdot 1.225 \cdot h}{p \cdot 100}\right)$$

geliefert, wobei p_N den normierten Druck und h die Höhe in Metern (beim ISAS - Institute for Analytical Sciences beträgt diese etwa 90 m) bezeichnet. Die Metereinheit wird ignoriert.

Temperatur Dies ist die Temperatur (in °C) im Driftraum und sollte in etwa mit der Umgebungstemperatur übereinstimmen (die bei den Messungen auch verwendet wurden). Die verwendeten Messungen wurden bei Temperaturen zwischen 20 und 25 °C aufgezeichnet.

Vortrennung Die Vortrennungseinheit bestimmt im Wesentlichen Trennungsgüte und Retentionszeiten von spezifischen Molekülen. Bei allen verwendeten Messungen wurde die gleiche MCC mit einer konstanten Säulentemperatur von 30 °C eingesetzt.

IMS Das eingesetzte IMS bestimmt u.a. Ionisation und Driftraumlänge. So findet sich beispielsweise in den Daten eines UV-IMS kein RIP. Alle verwendeten Messungen wurden mit dem gleichen ^{63}Ni-IMS durchgeführt.

Verstärker Der eingesetzte Verstärker hat einen starken Einfluss auf die entstehenden Daten. Insbesondere kann er spezifische Störsignale unter die IMS-Daten mischen. Alle verwendeten Messungen wurden mit dem gleichen Verstärker durchgeführt.

Datenerfassungskarte Die eingesetzte Datenerfassungskarte kann der Abtastgenauigkeit Grenzen setzen, dies betrifft den minimalen zeitlichen Abstand von Abtastpunkten und die Diskretisierung des analogen Spannungsignals. Außerdem kann eine Datenerfassungskarte ebenfalls Störsignale produzieren. Alle verwendeten Messungen wurden mit der gleichen Datenerfassungskarte durchgeführt. Diese wandelt abgetastete Werte des Spannungssignals mit einer Auflösung von 12 Bit in digitale Werte um, wobei allerdings auf Grund eines symmetrischen Spannungsbereichs und der Einbeziehung zu großer Spannungen nicht alle Bits genutzt werden[4].

Aufzeichnungssoftware Die verwendete Software kann ebenfalls Grenzen setzen. So lässt die bei den Messungen eingesetzte Software nur maximal 1000 spektren pro Messung zu und unterstützt keine automatische Aufzeichnung von Druck und Temperatur.

Computersystem Natürlich muss das eingesetzte Computersystem auf die Datenerfassungskarte und Software optimiert sein. Diese drei Messungsparameter stehen in engen Zusammenhang. Alle Messungen wurden mit dem gleichen Computer durchgeführt.

Für diese Arbeit wurden nur die Messungsparameter verwendet, die unter dem Gesichtspunkt der Driftzeitnormierung als sinnvoll und notwendig erachtet wurden. Dies sind Temperatur, Druck, Driftraumlänge, Gitteröffnungszeit und Feldstärke.

5.3. Konzentrationsdaten

Da in dieser Arbeit das Ziel einer automatisierten Ermittlung von Konzentrationsabhängigkeiten der Analyte verfolgt wurde, sind natürlich zwei weitere Typen von Variablen pro Messung von Bedeutung. Zunächst muss festgelegt werden, welche *Substanz* überhaupt untersucht wird. Außerdem ist selbstverständlich die *Konzentration* (in μg/L) einer Substanz in Beispielmessungen anzugeben.

Es sind zwei Typen von Daten zu unterscheiden:

Messungsreihe Die *Messungsreihe* beschreibt eine Sequenz von Messungen. In der Regel ist hier nur eine Substanz möglichst isoliert in den Messungen enthalten. Diese Substanz ist dann der Messungsreihe zugeordnet, während jeder Messung eine entsprechende Konzentration zugeordnet wird.

Einzelmessung Die *Einzelmessung* stellt einzelne (unabhängig betrachtete) Messung dar, die in der Regel mehrere Substanzen enthält (Gasgemisch).

Daten können natürlich abhängig von der Verwendung auch beiden Typen zugeordnet werden. So stellt eine Messungsreihe eine Sammlung von Einzelmessungen dar. Auf der anderen Seite wurden verschiedene Testgasgemische, die als Einzelmessungen eingesetzt werden sollten, als Messungsreihen mit mehreren Substanzen realisiert.

[4] Inzwischen wird eine neue Datenerfassungskarte eingesetzt, die eine höhere Auflösung bietet und alle Bits ausschöpft.

Um verschiedene Konzentrationen zu erzeugen wurde die Methode der *exponentiellen Verdünnung* eingesetzt. Hierbei wird eine Flasche mit einem Gasgemisch gefüllt. Durch nachströmendes Trägergas verringert sich die Konzentration enthaltener Substanzen exponentiell. Bei Ausgangskonzentration c_0 (in μg/L) einer bestimmten Substanz, dem Verdünnungsdurchfluss v (in mL/H) und dem Flaschenvolumen V (in L), lässt sich die Konzentration c (in μg/L) nach der Zeit t (in Minuten) durch die folgende, von einer Mitarbeiterin des ISAS - Institute for Analytical Sciences (Luzia Seifert) gelieferten Formel berechnen (vgl. auch [30]):

$$c = c_0 \cdot \exp\left(-\frac{v}{V} \cdot t\right).$$

Zwei Substanzen standen zur Konzentrationsabhängigkeitsbestimmung zur Verfügung: 2-Heptanon und 2-Octanon. Die Gasgemische bestehen aus den Substanzen 2-Heptanon, 2-Hexanon und 2-Octanon. Alle zur Konzentrationsabhängigkeitsanalyse verwendeten Messungen sind im Anhang A aufgelistet.

5.4. Intensitätsinvertierung

Die Spektren der vom ISAS - Institute for Analytical Sciences gelieferten Messungen haben eine negative Ausdehnung bezüglich der Intensitätsdimension. So erscheint der RIP als negativer Peak. Positive Peaks (die nicht auf Fehler wie beispielsweise Rauschen zurückzuführen sind) treten nicht auf.

Diese Repräsentation der vom IMS gelieferten Information wurde in dieser Arbeit von Anfang an als „unästhetisch“ betrachtet. Deshalb werden alle eingelesenen Intensitäten zunächst invertiert. Eine eingelesene Intensität $\tilde{x}_{i,j}$ wird also auf eine Intensität $x_{i,j} = (-1) \cdot \tilde{x}_{i,j}$ abgebildet.

Unter dem Gesichtspunkt einer automatisierten Analyse ist es natürlich uninteressant, ob nun eine positive oder negative Ausdehnung vorliegt. Es muss nur klar sein, welche der beiden Situationen vorliegt. Eine Fallunterscheidung wurde in dieser Arbeit jedoch umgangen und stattdessen festgelegt, dass nur „positive Spektren“ auftreten dürfen.

5.5. Basislinienkorrektur

Ein „leeres“ Spektrum ist dann gegeben, wenn während des betrachteten Schaltgitteröffnungsintervalls keine Ionen in den Driftraum gelangten. Man würde in diesem Fall ein annähernd konstantes Null-Signal bezüglich der Intensitäten des Spektrums erwarten, also $x_i \approx 0$. Annähernd deshalb, da jede tatsächliche Intensität $\tilde{x}_i$ durch einen Rauschfehler E_i verfälscht wird. Dies lässt sich in der bekannten Form

$$x_i = \tilde{x}_i + E_i$$

notieren. Unterstellt werden hier normalverteilte Fehler, also $E_i \sim \mathcal{N}\left(0, \sigma^2\right)$, $\sigma > 0$. Dabei wird weiterhin angenommen, dass σ klein genug ist, um bereits bei einer verhältnismäßig geringen Menge von Datenpunkten $x_1, \ldots, x_n$ ein Mittelwert von Null resultiert. Es können auch andere Wahrscheinlichkeitsverteilungen zu Grunde gelegt werden, so lange diese die Bedingung einer schnellen Konvergenz des Null-Mittelwertes erfüllen.

Da in dieser Arbeit verwendeten Messungen ausschließlich durch die Ionisation mittels einer ^{63}Ni-Strahlungsquelle erfolgte, beinhalten sie natürlich keine wirklich leeren Spektren: Zumindest der RIP hebt sich deutlich hervor. In diesem Fall kann man sich auf die Betrachtung

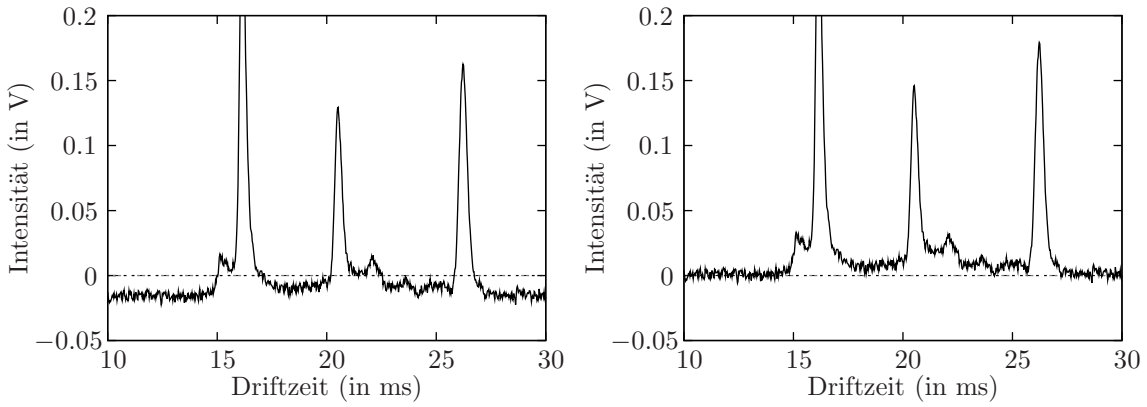

Abbildung 5.1.: Ein Spektrumsausschnitt vor (links) und nach (rechts) der Basislinienkorrektur

von leeren Teilbereichen beschränken, im Folgenden als *Rauschbeispiel*[5] bezeichnet. In einem Rauschbeispiel sollten nur Intensitäten $x_i = E_i$ vorkommen, außerdem muss der Bereich zusammenhängend und groß genug für die Mittelwertberechnung sein. Bei den in dieser Arbeit betrachteten Messungen hat sich der Driftzeitenbereich $[1, 10]$ (in ms) bewährt.

Eine Konsequenz ist, dass alle zulässigen Rauschbeispiele den gleichen Mittelwert bezüglich der (verfälscht) abgetasteten Intensitäten haben. Dieser Mittelwert a definiert mittels $f(x) = a$ die *Basislinie* des Spektrums.

Wie bereits erwähnt, sollte $a = 0$ gelten. Allerdings trifft dies in der Regel nicht zu. Zumindest weisen alle verwendeten Messungen Spektrenmittelwerte $a \neq 0$ auf. Die *Basislinienkorrektur* verschiebt die Basislinie auf die Driftzeitenachse. Dazu wird ein geeignetes Rauschbeispiel gewählt und die mittlere Intensität a berechnet. Alle Spektrenintensitäten x_i berechnen sich dann aus den in den eingelesenen Daten $\tilde{x}_i$ mittels $x_i = \tilde{x}_i - a$. Die transformierten Spektren haben dann alle eine Basislinie von Null. Der Effekt der Basislinienkorrektur ist in der Abbildung 5.1 dargestellt.

5.6. Faltenausgleich

Viele ^{63}Ni-IMS-Messungen weisen im oberen Bereich der Retentionszeiten eine Art Falte auf. Am deutlichsten ist diese im RIP zu erkennen. Dieser Effekt scheint im Zusammenhang mit hoher Feuchtigkeit in dem zu analysierendem Gasgemisch zu stehen. Auch wenn dieses Phänomen in den verwendeten Messungen nicht auftritt, so sollte hier natürlich trotzdem eine Lösung angestrebt werden.

Betrachtet man eine solche Falte genauer, so scheint es sich um eine positive Verschiebung für das gesamte Spektrum zu handeln. Unter dieser Annahme muss für jedes Spektrum s im Einflussbereich der Falte eine korrigierende Translation

$$t_s : \mathbb{R}^n \to \mathbb{R}^n,\ (d_1, \ldots, d_n) \mapsto (d_1 + \lambda, \ldots, d_n + \lambda)$$

gefunden werden. Daraus ergibt sich natürlich das Problem, dass sich die Messung anschließend vielleicht nicht mehr wie in der Definition 5.1 durch zwei Vektoren und eine Matrix

[5] Eigentlich wird die Bezeichnung „Rauschsample“ bevorzugt. Dies stellt jedoch eine unerwünschte Sprachmischung dar. Der englischsprachige Begriff „Noise Sample“ scheint eben so unangebracht, da ansonsten in dieser Arbeit der deutschsprachige Begriff „Rauschen“ verwendet wird.

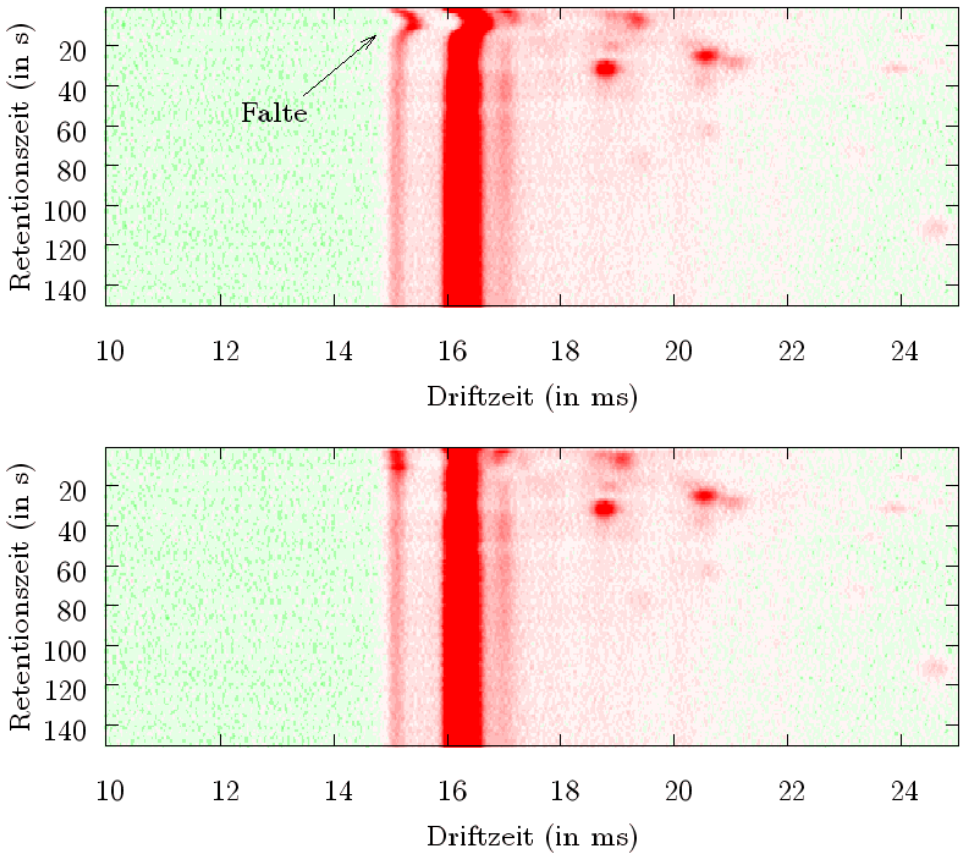

Abbildung 5.2.: Ein Messungsausschnitt vor (oben) und nach (unten) Faltenausgleich

beschreiben lässt. Allerdings sind die Abtastpunkte äquidistant. Bechränkt man sich auf die Genauigkeit der Abtastrate, so kann die Translation durch ein Verschieben der Indizes innerhalb der Spektren erfolgen:

$$t_s : \mathbb{R}^n \to \mathbb{R}^n,\ (d_1, \ldots, d_n) \mapsto (d_{1+a}, \ldots, d_{n+a})\,.$$

Auf diese Weise „fällt“ ein Teil der Daten aus der Matrix heraus, während ein Teil der Matrix unbestimmt ist. Im Grunde lassen sich alle Methoden aus dem Abschnitt 3.3.5 anwenden, um diesen Bereich mit Werten zu füllen. Hier wurden sich in diesem Fall aber auf Zero-Padding beschränkt.

Nun gilt es, eine Translation für jedes Spektrum zu bestimmen. In den meisten Spektren stellt der RIP das absolute Maximum dar. Mit zunehmender Anzahl von Analyten kann der RIP natürlich auch diese Eigenschaft verlieren, denn die Reaktionsionen geben ihre Energie an die Analyte ab. Beschränkt man sich auf den Fall, dass das Maximum stets durch den RIP gegeben ist, so kann man alle Spektren über den RIP-Index miteinander vergleichen.

Ist die Maximumeigenschaft nicht erfüllt, so kann man zumindest annehmen, dass in vielen Spektren die Maximumeigenschaft erfüllt ist und wahrscheinlich sogar das globale Maximum über alle Spektren auf den RIP verweist. Mit Hilfe dieser Informationen können dann lokale Maxima in den anderen Spektren ermittelt werden, beispielsweise indem immer das Maximum mit dem kleinsten Indexabstand im Vergleich zum globalen Maximum ermittelt wird.

Sind alle RIP-Indizes ermittelt, kann der kleinste Index als Referenzindex benutzt wer-

den. Dies ist dadurch gerechtfertigt, da ja nur positive Verschiebungen angenommen werden. Möchte man diesen Referenzindex durch mehr als ein Spektrum bestimmen lassen, so kann man eine maximale Abweichung festlegen und den Referenzindex als (gerundeten) Mittelwert aller durch die Abweichung tolerierten Indizes definieren.

Die Translationen im Intensitätsraum der einzelnen Spektren berechnen sich nun ganz einfach als

$$t_s : \mathbb{R}^n \to \mathbb{R}^n,\ x_i \mapsto \begin{cases} x_{i+(R_s-R)} & \text{falls } i \leq n - (R_s - R) \\ 0 & \text{sonst} \end{cases}$$

wobei R den Referenzindex und R_s den RIP-Index des entsprechenden Spektrums bezeichnet. Der Effekt des Faltenausgleichs ist in der Abbildung 5.2 dargestellt.

6. Peakerkennung

Das Ziel dieser Arbeit bestand in der Entwicklung einer automatisierten Methode zur Bestimmung von Konzentrationsabhängigkeiten. Eine Konzentrationsabhängigkeit kann dabei als eine mathematische Beschreibung der Beziehung zwischen aus Messungen extrahierter Informationen und den Messungen zugeordneten Konzentrationen einer spezifischen Substanz beschrieben werden. Dazu müssen zunächst vergleichbare Informationen festgelegt werden.

Messungen zeigen in der Regel ausgeprägte Peaks, also lokale Maxima in „intuitiv“ geglätteten Messungsdaten. Damit sind diese nicht eindeutig. Beispielsweise kann ein lokales Maximum in einem Spektrum als ein Effekt von Rauschen oder als Peak aufgefasst werden. Eine algorithmische Lösung sollte in diesem Fall natürlich möglichst die richtige Entscheidung treffen. Dieser Prozess gliedert sich in zwei Schritte. Zunächst muss das Signal geglättet werden. Danach müssen eindimensionale Peaks in den Spektren ermittelt werden.

Jeder eindimensionale Peak sollte auf ein bestimmtes Molekül verweisen. Damit bestehen Beziehungen zwischen Spektrumpeaks benachbarter Spektren, denn das Auftreten von Analyten sollte auf ein Intervall von Retentionszeiten eingeschränkt sein und die Messungsparameter im Wesentlichen für die gesamte Messung gelten. Das Ergebnis sind zweidimensionale Peaks in den Messungsdaten. Eine wichtige Voraussetzung dieser Arbeit besteht darin, dass eindimensionale Peaks durch Funktionen beschrieben werden können und sich diese Funktionen recht ähnlich sind. Eine weitere Annahme ist, dass sich zusammengehörige Peaks durch eine zweidimensionale Funktion beschreiben lassen, die aus den eindimensionalen Peaks gewonnen werden kann. Dieser Prozess kann ebenfalls in zwei Schritte untergliedert werden: Die Bildung von Ketten zusammengehöriger eindimensionaler Peaks und die Erzeugung zweidimensionaler Peaks aus den Ketten.

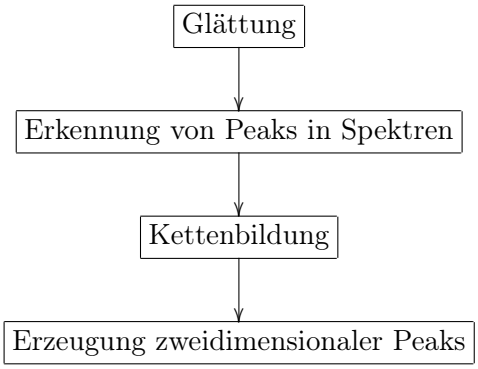

Abbildung 6.1.: Die Peakerkennungspipeline

Es ergibt sich eine feste Folge von Verarbeitungsschritten zur Ermittlung der Messungspeaks, die in der Abbildung 6.1 dargestellte *Peakerkennungspipeline*. Im Folgenden werden in dieser Arbeit entwickelte Lösungen für die einzelnen Verarbeitungsschritte vorgestellt. Ausgangspunkt ist eine Messung, die die in den Abschnitten 5.4, 5.5 und eventuell 5.6 vorgestellten Vorverarbeitungsschritte bereits durchlaufen hat.

6.1. Glättung

Unter Glättung kann man sich informell die Überführung einer Funktion in eine Version mit weniger Details vorstellen. Dabei sollten unwichtige oder störende Merkmale „weg geglättet“ werden, während die Funktion ansonsten unangetastet bleibt. Für diese Arbeit ist das Ziel einer Glättung recht einfach zu formulieren. Es besteht darin, das ursprüngliche Signal eines Spektrums zu rekonstruieren[1], wobei mit Rekonstruktion die Entfernung von Störeinflüssen wie Rauschen gemeint ist.

Ausgangspunkt für diesen Verarbeitungsschritt ist eine Messung $(\mathbf{D}, \mathbf{d}, \mathbf{r})$ mit m Spektren und n Driftzeitschnitten. Im mathematischen Sinne stellt die Glättung eine Transformation T_G der Intensitätsmatrix $\mathbf{D}$ dar, also

$$T_G : \begin{pmatrix} x_{1,1} & \cdots & x_{n,1} \\ \vdots & & \vdots \\ x_{1,m} & \cdots & x_{n,m} \end{pmatrix} \mapsto \begin{pmatrix} y_{1,1} & \cdots & y_{n,1} \\ \vdots & & \vdots \\ y_{1,m} & \cdots & y_{n,m} \end{pmatrix},$$

bzw. $T_G(x_{i,j}) = y_{i,j}$.

Die in dieser Arbeit untersuchten Glättungsverfahren werden im Folgenden vorgestellt. Es zeigte sich, dass die zur Konzentrationsabhängigkeitsanalyse verwendeten Messungen weniger mit Rauschen belastet sind, als frühere Messungen. Deshalb wurde eine ältere Messung zur Demonstration der Glättungen ausgewählt.

6.1.1. Mittelwertfilter

Die Datenmatrix D kann als eine Rastergrafik aufgefasst werden. Dann lässt sich das Mittelwertfilter wie im Abschnitt 3.1 anwenden. Dabei wird ein Fenster durch die ganzzahligen Parameter $a \geq 0$ und $b \geq 0$ festgelegt. Randwertprobleme werden durch Zero-Padding gelöst, dazu sei

$$f(x_{i,j}) := \begin{cases} x_{i,j} & \text{falls} \quad 1 \leq i \leq n \ \wedge \ 1 \leq j \leq m \\ 0 & \text{sonst} \end{cases}$$

definiert. Ein transformierter Intensitätswert wird dann mittels

$$y_{s,t} = \frac{1}{(2a+1)(2b+1)} \sum_{i=s-a}^{s+a} \sum_{j=t-b}^{t+b} f(x_{i,j}) \tag{6.1}$$

berechnet.

Falls $a = b = 0$, wird nicht geglättet. Wird nur einer der Parameter größer Null gewählt, erfolgt die Glättung nur in einer Dimension. Da sich Maxima aber in beide Dimension ausbreiten, ist eine zweidimensionale Transformation vorzuziehen.

Je größer a oder b gewählt sind, desto mehr Datenpunkte fließen in die Transformation ein und das Rauschen wird zunehmend reduziert. Allerdings werden auch die Peaks ihrer Umgebung angepasst. Die Abbildungen 6.2 und 6.3 zeigen die Auswirkung unterschiedlicher Fenster. Bei genauer Betrachtung der Abbildung 6.3 zeigt sich, dass das gezeigte Spektrum durch ein größeres b schlechter approximiert wird.

6.1.2. Fourier-Transformation

Im Abschnitt 3.2 wurde die Fourier-Transformation beschrieben. Sie liefert die Information welche Frequenz mit welcher Intensität im Signal vorhanden ist.

[1] Bei den betrachteten Messungen ist dies die Überlagerung von 10 Signalen.

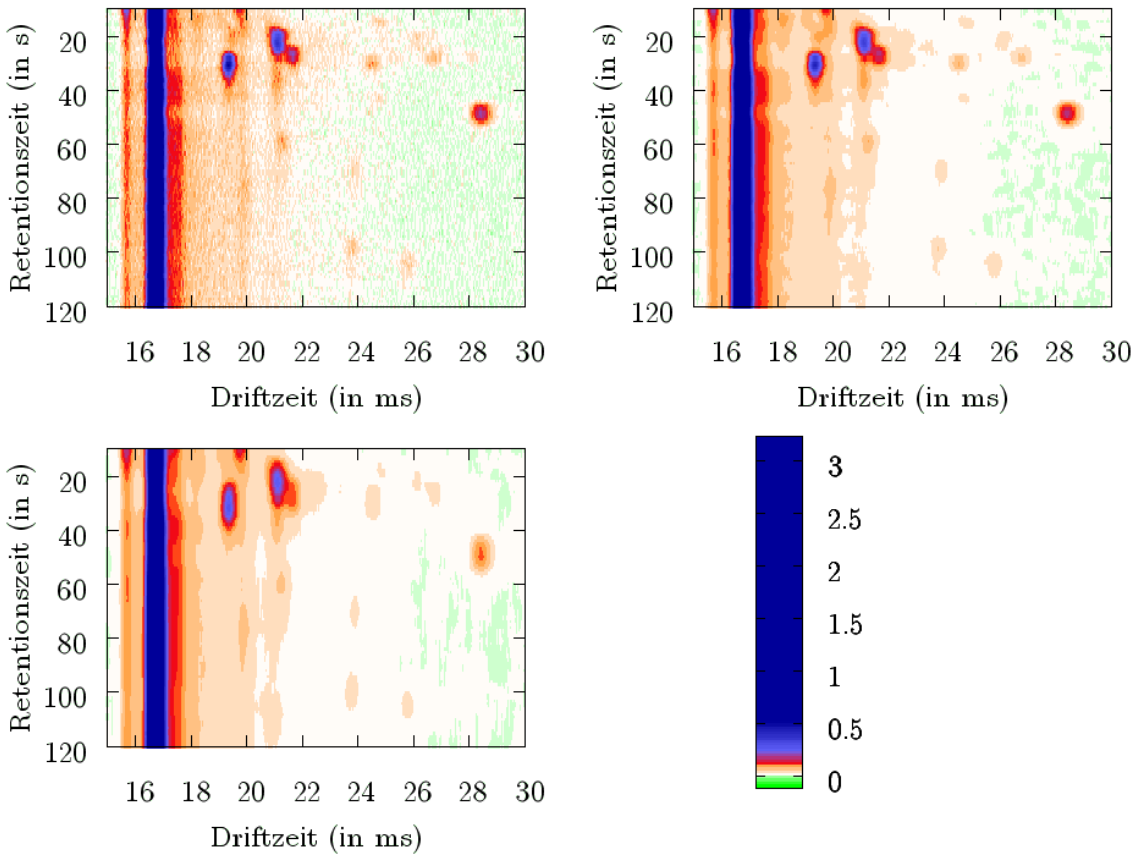

Abbildung 6.2.: Ein Ausschnitt einer Messung (oben, links) nach Anwendung eines Mittelwertfilters mit $x = 5, y = 1$ (oben, rechts) und eines Mittelwertfilters mit $x = y = 5$ (unten, links), die Intensitätsskala ist in Volt angegeben

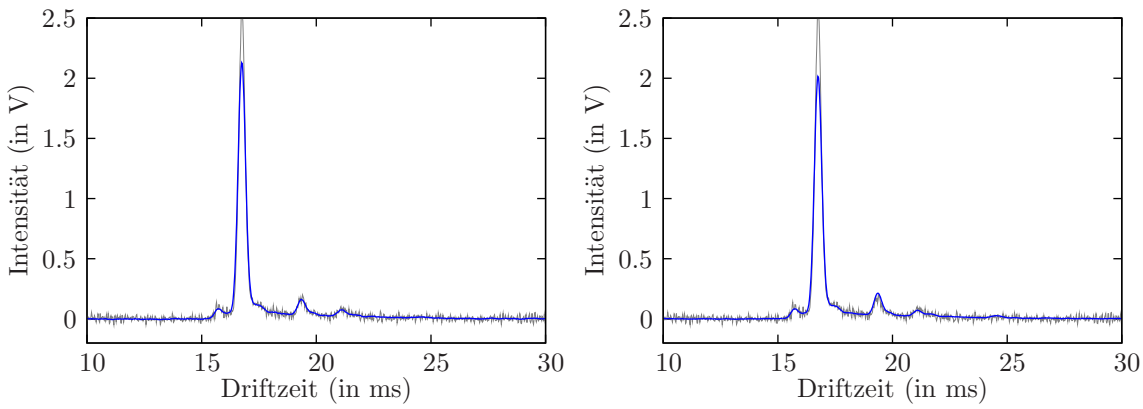

Abbildung 6.3.: Ein Ausschnitt eines Spektrums (grau) nach Anwendung eines Mittelwertfilters mit $x = 5, y = 1$ (links) und eines Mittelwertfilters mit $x = y = 5$ (rechts)

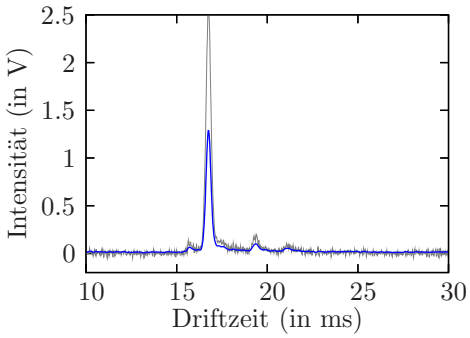

Abbildung 6.4.: Ein Ausschnitt eines Spektrums (grau) nach Löschen der hochfrequenten Anteile in der Spektralfunktion und anschließender Rücktransformation mittels der IFT (blau)

In dieser Arbeit wurde dem Rauschen im Signal unterstellt, dass es sich im Wesentlichen durch hohe Frequenzen auszeichnet. Da das Rauschen im Gegensatz zu speziellen Signalausprägungen im gesamten Abtastungsbereich vorhanden sein sollte, müssten die entsprechenden Frequenzen relativ stark ausgeprägt sein.

Um das Rauschen aus der Spektralfunktion heraus zu filtern, ist es dann nahe liegend, die entsprechenden Frequenzintensitäten auf Null zu setzen und mittels der IFT ein geglättetes Signal zu erzeugen. Dies entspricht einem Tiefpassfilter.

Zur Anwendung der FFT muss die Signallänge auf eine Zweierpotenz vergrößert werden. Zero-Padding und Spiegelung (vgl. Abschnitt 3.3.5) zeigten dabei keine nennenswerten Unterschiede im Ergebnis. Unter Angabe einer maximalen Frequenz α lässt sich die Transformation der Spektralfunktion $\hat{f}$ als ein Tiefpassfilter

$$\hat{g}(\omega) := \begin{cases} \hat{f}(\omega) & \text{falls} \quad \omega \geq \alpha \\ 0 & \text{sonst} \end{cases}$$

definieren, wobei $\hat{g}$ die transformierte Spektralfunktion bezeichnet. Die Abbildung 6.4 zeigt den Effekt dieser Glättung. Deutliche Merkmale des Signals bleiben zwar erhalten, aber die Approximation ist sehr schlecht. Die Erweiterung auf eine zweidimensionale FT-Analyse brachte auch keine Vorteile, während die Berechnungszeit stark anstieg. Auch der Einsatz eines Multi-Bandpassfilters, also das Nullsetzen beliebiger Frequenzintervalle, führte zu keinem brauchbaren Ergebnis.

Eine weitere Idee bestand darin, zuerst ein Rauschbeispiel mittels Spiegelung auf die benötigte (auf eine Zweierpotentz verlängerte) Spektrumgröße zu bringen. Mit den absoluten Werten der entsprechenden Spektralfunktion wurde danach ein weiches Thresholding (vgl. Abschnitt 3.3.6) auf die Spektralfunktion des entsprechenden Spektrumsignals durchgeführt. Das Ziel bestand darin, die Frequenzanteile des Rauschens exakt zu bestimmen und aus dem zu glättenden Signal zu löschen. Dieser Versuch schlug gänzlich fehl, da eine Rauschreduzierung praktisch nicht zu erkennen war. Nach diesen Negativresultaten wurde eine FT-Glättung in dieser Arbeit nicht weiter untersucht.

6.1.3. Wavelet-Transformation

Im Abschnitt 3.3 wurde die Wavelet-Transformation vorgestellt. Die Berechnung der WT erfolgt dabei in dieser Arbeit mittels des Lifting-Schemas (vgl. Abschnitt 3.3.4). Randwertprobleme werden durch Spiegelung gelöst.

Wünschenswert ist eine Berechnung „am Platze“. Dies bedeutet, dass die Implemtierung nur auf einem initial gegebenen Array arbeitet. Neben einem geringeren Speicherbedarf, zeichnen sich Am-Platze-Algorithmen vor Allem durch eine beschleunigte Berechnung aus. Damit die Berechnung am Platze erfolgen kann, ist eine spezielle Kodierung der Daten notwendig.

Ausgangspunkt ist ein Array der Länge der Länge 2^m mit $m > 0$. Initial ist dieses Array mit dem (gegebenenfalls verlängerten) Intensitätsvektor eines Spektrums gefüllt. In der Lifting-Notation ist dies $(s_{0,0}, \ldots, s_{0,n-1})$, wobei zur Vereinfachung der im Folgenden vorgestellten Kodierung die Indizes auf die Arrayindexmenge $\{0, \ldots, n-1\}$ abgebildet wurden.

Es sind genau sind $m+1$ Skalen $V_0, \ldots, V_m$ möglich, wobei V_0 den initialen Intensitätsvektor darstellt. Es sei

$$S_i := 2^i \quad \text{und} \quad L_i := \frac{n}{2^i}. \tag{6.2}$$

Zu jeder Skala V_i gehören dann die geraden Elemente $s_{i,0}, \ldots, s_{i,L_i-1}$ und falls $i > 0$ die ungeraden Elemente $d_{i,0}, \ldots, d_{i,L_i-1}$, also die entsprechenden Approximations- und Detailkoeffizienten. Jedem Koeffizienten muss ein Arrayindex zugewiesen werden. Dazu werden zwei Abbildungen I_s und I_d als

$$I_s : (i,j) \mapsto j \cdot S_i = j \cdot 2^i \tag{6.3}$$

$$I_d : (i,j) \mapsto I_s\,(i,j) + \frac{S_i}{2} = j \cdot 2^i + 2^{i-1} \tag{6.4}$$

definiert. Approximationskoeffizienten $s_{i,j}$ ist der Arrayindex $I_s\,(i,j)$ und Detailkoeffizienten $d_{i,j}$ der Arrayindex $I_d\,(i,j)$ zugeordnet.

Bei jedem Analyseschritt, also dem Übergang von V_i zu V_{i+1}, können wie bei der MSA die Approximationskoeffizienten $s_{i,\cdot}$ überschrieben werden. So resultieren nach dem vollständigen Lifting $n-1$ Deatilkoeffizienten und ein Approximationskoeffizient. Die Kodierung sei am folgenden Beispiel verdeutlicht.

Skala 0:	$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$	$s_{0,4}$	$s_{0,5}$	$s_{0,6}$	$s_{0,7}$
Skala 1:	$s_{1,0}$	$d_{1,0}$	$s_{1,1}$	$d_{1,1}$	$s_{1,2}$	$d_{1,2}$	$s_{1,3}$	$d_{1,3}$
Skala 2:	$s_{2,0}$	$d_{1,0}$	$d_{2,0}$	$d_{1,1}$	$s_{2,1}$	$d_{1,2}$	$d_{2,1}$	$d_{1,3}$
Skala 3:	$s_{3,0}$	$d_{1,0}$	$d_{2,0}$	$d_{1,1}$	$d_{3,0}$	$d_{1,2}$	$d_{2,1}$	$d_{1,3}$

Es gibt zwei einfache Verfahren, mit Hilfe der WT eine Rauschreduzierung durchzuführen: Thresholding und die Transformation eines Signals auf eine detailärmere Skala. Der Erfolg beider Methoden hängt von dem gewählten Wavelet ab. In dieser Arbeit wurde sich auf den Vergleich des Haar- und des Daubechies$_4$-Wavelets eingeschränkt.

Die Reduktion auf eine Skala mit weniger Details ist in der Abbildung 6.5 dargestellt. Es zeigt sich, dass die Detailinformationen der entsprechenden Wavelets nicht mit dem Rauschen übereinstimmen.

Thresholding (vgl. Abschnitt 3.3.6) liefert bessere Ergebnisse. Dabei kann der Grenzwert entweder manuell festgelegt oder automatisch bestimmt werden. Im zweiten Fall kann die Standardabweichung des Rauschens entweder wie im Abschnitt 3.3.6 abgeschätzt oder durch die Analyse eines Rauschbeispiels berechnet werden. In der Abbildung 6.6 wurde der automatische Grenzwert und das Daubechies$_4$-Wavelet verwendet. Im Vergleich der Thresholdingmethoden bringt hartes Thresholding signifikante Signalausprägungen besser zur Geltung, während weiches Thresholding in der Regel ein „glatteres“ Signal liefert (s. Abbildung 6.6).

Vergleicht man die beiden Wavelets, so fällt beim Haar-Wavelet vor Allem die „klotzige“ Rekonstruktion modifizierter Koeffizienten auf. Aus der Lifting-Schema-Berechnung folgt außerdem, dass das Daubechies$_4$-Wavelet im Vergleich weniger lokal analysiert. Somit stellt das Daubechies$_4$-Wavelet in dieser Arbeit das bevorzugte WT-Analysewerkzeug dar.

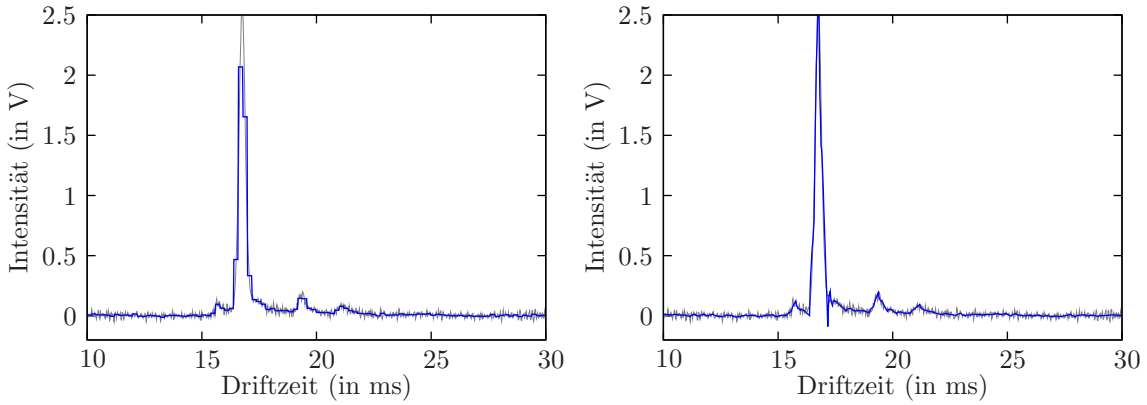

Abbildung 6.5.: Ein Ausschnitt eines Spektrums (grau) und seine Approximation in der Skala V_3 bei Verwendung des Haar-Wavelets (links) und des Daubechies$_4$-Wavelets (rechts)

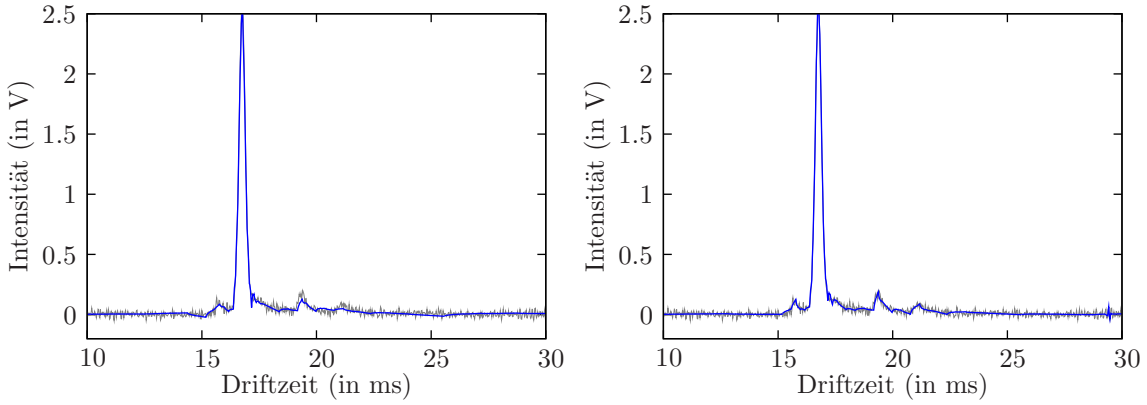

Abbildung 6.6.: Ein Ausschnitt eines Spektrums (grau) nach weichem (links) und hartem (rechts) Thresholding

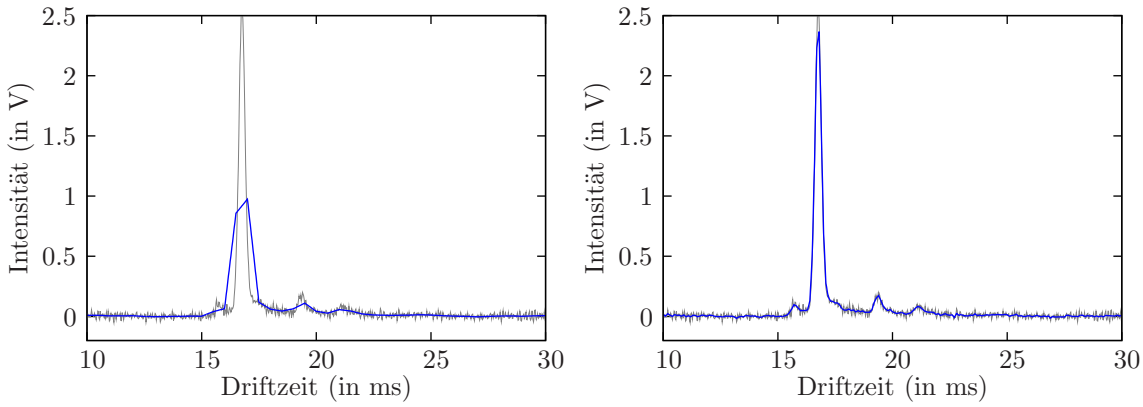

Abbildung 6.7.: Glättung eines Spektrumausschnitts (grau) durch Anwendung einer uniformen F-Transformation mit 100 (links) und 500 (rechts) Intervallen

6.1.4. Fuzzy-Transformation

Im Abschnitt 3.4.2 wurde die Fuzzy-Transformation vorgestellt. Sie lässt sich in gewisser Weise mit dem Mittelwertfilter vergleichen, da beide Methoden Mittelungen von benachbarten Punkten vornehmen. Allerdings wird bei der F-Transformation die Gewichtung an Hand von Stützstellen festgelegt und somit nicht jeder Datenpunkt gleich behandelt. Auf der anderen Seite ist die F-Transformation einer bloßen Interpolation von Stützstellen überlegen, weil dazwischen liegende Datenpunkte berücksichtigt werden.

Uniforme Fuzzy-Partitionen können gute Glättungstransformationen erzeugen, falls die Anzahl der Basisfunktionen optimal gewählt ist. Zu wenige Basisfunktionen approximieren das transformierte Signal nur unzureichend. Zu viele Basisfunktionen führen dagegen zu einer niedrigen Rauschreduzierung. Die Abbildung 6.7 zeigt eine F-Transformation mit uniformer Fuzzy-Partition.

Die F-Transformation stellt unter allen hier vorgestellten Glättungsmethoden die Rechenintensivste dar, falls eine größere Anzahl von Basisfunktionen eingesetzt wird. Dabei wurde die Berechnung schon optimiert, da ja nur maximal zwei Intervalle für die Berechnung einer Komponente oder Rekonstruktion eines Datenpunktes relevant sind. Nichtuniforme Fuzzy-Partition bieten zum Einen den Vorteil, die Anzahl der Basisfunktionen zu reduzieren. Zum Anderen - und dies ist der wichtigere Nutzen - lassen sich die Stützstellen auf signifikante Merkmale des zu transformierenden Signals legen. Der Greedy-Algorithmus (vgl. Abschnitt 3.4.3) lieferte sehr gute Resultate. Allerdings ist der hohe Berechnungsaufwand nicht tolerierbar. So wurden bei der Nutzung von 300 Basisfunktionen für wenige Spektren schon mehrere Minuten Rechenzeit benötigt. Stattdessen wurde in dieser Arbeit eine andere Methode zur Bestimmung der Stützstellen entwickelt, die im folgenden Abschnitt vorgestellt wird.

6.1.5. Fuzzy-Transformation mit Wavelet-Transformation

Die WT liefert sowohl zeitliche wie auch spektrale Information. Diese Information lässt sich dazu benutzen, wichtige Merkmale des Funktionsverlaufs eines zu glättenden Signals zeitlich zu lokalisieren. Die F-Transformation hat dagegen eine sehr gute Glättungseigenschaft in dem Sinn, dass lokale Minima und Maxima geringer Ausprägung (insbesondere zwischen Stützstellen) eliminiert werden.

In dieser Arbeit wurde der Versuch unternommen, diese beiden Stärken zu kombinie-

ren. Es macht wenig Sinn, zuerst die F-Transformation und dann die WT anzuwenden, denn die WT kann nur Informationen extrahieren, die im untersuchten Signal auch tatsächlich vorliegen. Das umgekehrte Vorgehen, also zuerst mittels der WT zu glätten und dann die F-Transformation anzuwenden, scheint hoffnungsvoller. Allerdings weist ein Waveletrücktransformiertes Signal mit unter stärkere (unerwünschte) Funktionsmerkmale als das Ausgangssignal auf (vgl. Abbildung 6.6). Zudem muss die Auswahl der Stützstellen noch erfolgen.

Der hier vorgeschlagene Weg besteht darin, die Informationen der WT zur Bestimmung der Stützstellen zu benutzen und das Signal anschließend nur durch die F-Transformation zu glätten. Die Stützstellenbestimmung erfolgt dabei direkt aus den Detailkoeffizienten der WT. Als wichtige Merkmale werden gerade die Detailkoeffizienten festgelegt, die nach einem harten Thresholding nicht Null sind. Mit Hilfe von der Gleichung 6.3 lässt sich der zu einem bestimmten Detailkoeffizienten korrespondierende Zeitbereich ermitteln. Es wird nun genau in der Mitte eines jeden solchen Bereiches eine Basisfunktion in die Fuzzy-Partition eingefügt, falls an diesem Index nicht bereits eine Basisfunktion existiert.

Es sei $S = (s_{0,0}, \ldots, s_{0,n-1})$ der Intensitätsvektor des zu transformierenden Spektrums. Die verwendeten Basisfunktionen der F-Transformationen seien Dreiecksfunktionen, die korrespondierende Fuzzy-Partition F enthalte initial nur die beiden Randbasisfunktionen. Das verwendete Wavelet sei im Folgenden das Daubechies$_4$-Wavelet, der verwendete Schwellwert λ sei automatisch ermittelt. Der Algorithmus 6.1.1 beschreibt dann das Auffüllen von F mit Basisfunktionen.

Algorithmus 6.1.1 Basisfunktionen durch WT-Thresholding

1: Gegeben sei eine Fuzzy-Partition F, ein Schwellwert λ und ein Datenvektor S der Länge n.
2: Erweitere S durch Spiegelung auf einen Vektor $\tilde{S}$ der Länge 2^m.
3: Führe Lifting auf $\tilde{S}$ durch.
4: **for** $i = 1; i \leq m; i++$ **do**
5: **for** $j = 0; j < L_i; j++$ **do**
6: **if** $\left|\tilde{d}_{i,j}\right| \leq \lambda$ **then**
7: Berechne Arrayindex $p = \lfloor (I_s(i,j) + I_d(i,j))/2 \rfloor$.
8: **if** $0 \leq p < n$ und an p noch keine Basisfunktion vorhanden **then**
9: Füge Basisfunktion an Index p in F ein.
10: **end if**
11: **end if**
12: **end for**
13: **end for**
14: **return** F.

Die Abbildung 6.8 zeigt eine Anwendung dieser Glättung. Dabei fällt auf, dass manchmal noch eine stärkere Berücksichtigung von Signalmerkmalen wünschenswert wäre. Von besonderer Bedeutung sind stark ausgeprägte lokale Maxima. Allerdings wäre die Erzeugung von Basisfunktionen an den lokalen Maxima wohl kaum erfolgversprechend, denn die meisten lokalen Minima und Maxima werden durch das Rauschen erzeugt. Die Lösung besteht darin, den Intensitätsraum eines Spektrums „gröber“ zu betrachten. Nur ausgeprägte Maxima sollen Basisfunktionen erzeugen.

Algorithmisch lässt sich dies mit einem Toleranzparameter $\epsilon > 0$ realisieren. Das Kriterium für ein Minimum oder Maximum ist unter der Annahme, dass keine dieser Extremstellen über mehr als einen Datenpunkt verläuft, durch einen Vorzeichenwechsel in der ersten Ableitung gegeben. Dieses Kriterium wird für ausgeprägte Minima oder Maxima, im Folgenden als ϵ-

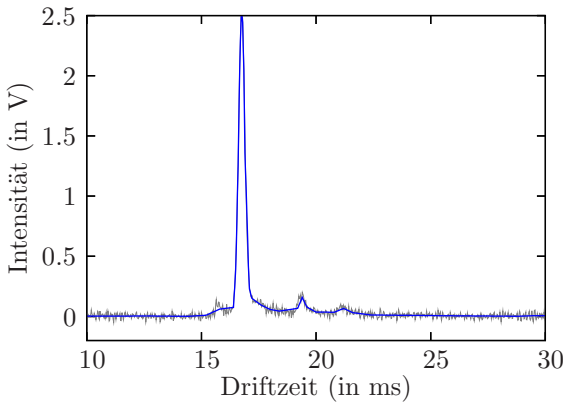

Abbildung 6.8.: Ein Ausschnitt eines Spektrums (grau) und die Anwendung der F-Transformation mit einer aus WT-Thresholding gewonnenen Fuzzy-Partition (blau)

Minima bzw. *ϵ-Maxima* bezeichnet, dahin gehend verschärft, dass die absolute Differenz des Extremstellenpunktes zu den umgebenden ausgeprägten Extremstellenpunkten mindestens ϵ beträgt. Die Randpunkte werden gesondert behandelt.

Es resultieren zwei Suchprozesse: Das Auffinden von ϵ-Minima und das Auffinden von ϵ-Maxima. Der Algorithmus 6.1.2 realisiert die Suche des nächsten ϵ-Minimums von einem initialen Index $i_I \in \{0, \ldots, n-1\}$ aus. Damit wird immer eine Minimumposition m zurück-

Algorithmus 6.1.2 ϵ-Minimum Suche

1: Gegeben sei ein initialer Index i_I, ein Toleranzparameter ϵ und ein Datenvektor $S = (s_0, \ldots, s_{n-1})$.
2: Setze Minimumindex $m = i_I$.
3: **if** $i_I = n - 1$ **then**
4: **return** $(m, e = m)$.
5: **end if**
6: Setze aktuellen Index $i = i_I + 1$.
7: **while** $(i < n)$ **do**
8: **if** $s_i < s_m$ **then**
9: $m = i$.
10: **else**
11: **if** $s_i - s_m > \epsilon$ **then**
12: **return** $(m, e = i - 1)$.
13: **end if**
14: **end if**
15: $i++$.
16: **end while**
17: **return** $(m, e = i - 1)$.

geliefert, selbst dann, wenn die Datenwerte konstant sind. Dies ist leicht zu überprüfen, da dann für das Ende des betrachteten Intervalls $e = n - 1$. Analog dazu dient der Algorithmus 6.1.3 zum Auffinden von ϵ-Maxima. Diese beiden Algorithmen lassen sich dann zur Bestimmung aller ϵ-Maxima abwechselnd anwenden. Das Resultat ist der Algorithmus 6.1.4.

Algorithmus 6.1.3 ϵ-Maximum Suche

1: Gegeben sei ein initialer Index i_I, ein Toleranzparameter ϵ und ein Datenvektor $S = (s_0, \ldots, s_{n-1})$.
2: Setze Maximumindex $m = i_I$.
3: **if** $i_I = n - 1$ **then**
4: **return** $(m, e = m)$.
5: **end if**
6: Setze aktuellen Index $i = i_I + 1$.
7: **while** $(i < n)$ **do**
8: **if** $s_i > s_m$ **then**
9: $m = i$.
10: **else**
11: **if** $s_m - s_i > \epsilon$ **then**
12: **return** $(m, e = i - 1)$.
13: **end if**
14: **end if**
15: $i++$.
16: **end while**
17: **return** $(m, e = i - 1)$.

Algorithmus 6.1.4 ϵ-Maxima Suche

1: Gegeben sei ein Toleranzparameter ϵ und ein Datenvektor $S = (s_0, \ldots, s_{n-1})$.
2: Initialisiere eine leere Liste L von Indizes.
3: Setze Extremstellenindex $m = 0$ und Startindex $i = 0$.
4: **loop**
5: Rufe Algorithmus 6.1.3 zur Suche des nächsten ϵ-Maximums mit m, i auf und erhalte neue m, i.
6: **if** $i = n - 1$ **then**
7: **return** L.
8: **end if**
9: Setze $L = L \cup \{i\}$.
10: Rufe Algorithmus 6.1.2 zur Suche des nächsten ϵ-Minimus mit $m = i, i$ auf und erhalte neue m, i.
11: **if** $i = n - 1$ **then**
12: **return** L.
13: **end if**
14: **end loop**

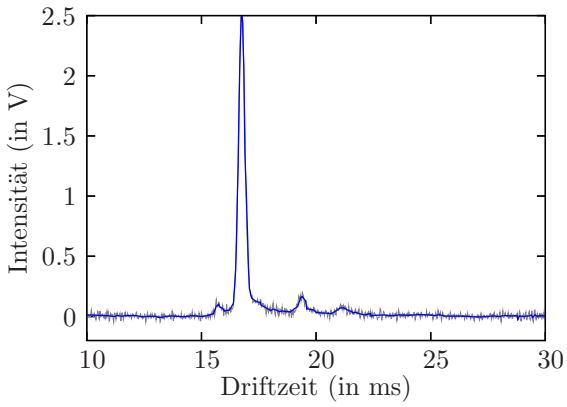

Abbildung 6.9.: Ein Ausschnitt eines Spektrums (grau) und die Anwendung der F-Transformation mit einer aus WT-Thresholding und ϵ-Maxima gewonnenen Fuzzy-Partition (blau)

Es resultiert eine Liste L von Indizes gefundener ϵ-Maxima. An jedem Index wird, falls nicht schon vorhanden, eine neue Basisfunktion in die Fuzzy-Partition F eingefügt. Eine Beispiel dieser Glättung wird in der Abbildung 6.9 gezeigt. Es ist offensichtlich, dass ϵ einen sehr sensiblen Parameter darstellt. Als günstige Parameter haben sich $\epsilon \geq 0.01$ erwiesen, in der Abbildung 6.9 wurde $\epsilon = 0.035$ gewählt. Eine Automatisierung kann beispielsweise mittels $\epsilon := \beta \cdot a$ erfolgen, wobei β den maximalen Intensitätsabstand in einem Rauschbeispiel und a einen Faktor wie Zwei bezeichnen.

Die Spektren einer Messung lassen sich unabhängig von einander durch die hier vorgestellten Kombination von WT und F-Transformation glätten. Eine Erweiterung auf zwei Dimensionen wie im Abschnitt 3.4.2 lässt sich nicht realisieren, da jedem Spektrum eine eigene Fuzzy-Partition zugewiesen wird. Allerdings kann eine zweidimensionale Glättung durch eine gewichtete Mittelung zwischen den Resultaten der Glättung in jede der beiden Dimensionen erzielt werden. Dazu wird zusätzlich jeder Driftzeitschnitt geglättet. Es resultieren dann zwei geglättete Matritzen $\mathbf{S}$ (Spektrumglättung) und $\mathbf{D}$ (Driftzeitschnittglättung). Mit einem zusätzlichen Parameter $\alpha \in [0, 1]$ kann dann jeder final geglättete Datenpunkt $y_{i,j}$ durch

$$y_{i,j} = (1 - \alpha) \cdot s_{i,j} + \alpha \cdot d_{i,j} \tag{6.5}$$

berechnet werden. Der Parameter α legt also das Gewichtungsverhältnis beider Dimensionen fest, gilt beispielsweise $\alpha = 0$, so wird nur nach Spektren geglättet. Eine Glättung mit $\epsilon = 0.035$ und $\alpha = 0.5$ ist in der Abbildung 6.10 dargestellt. Man erkennt eine Rauschminderung bei Erhaltung wichtiger Merkmale.

6.2. Erkennung eindimensionaler Peaks

Angenommen, es würde nur eine einzige Molekülart ionisiert. Dann kann man vermuten, dass diese Ionen ungefähr zur gleichen Driftzeit die Faraday-Platte erreichen und das resultierende Spektrum einen einzigen schmalen Impuls enthält. Allerdings ist diese Annahme zu ideal, denn die Driftzeiten konzentrieren sich zwar in Form eines Intensitätspeaks um eine spezifische Driftzeit herum, dieser hat jedoch einen gausskurvenähnlichen Verlauf. Gelangen viele der Ionen in den Driftraum, so dass der Peak entsprechend hoch ist, fällt ein ausgeprägtes *Tailing* auf. Dies gilt insbersondere für den RIP (s. Abbildung 6.11).

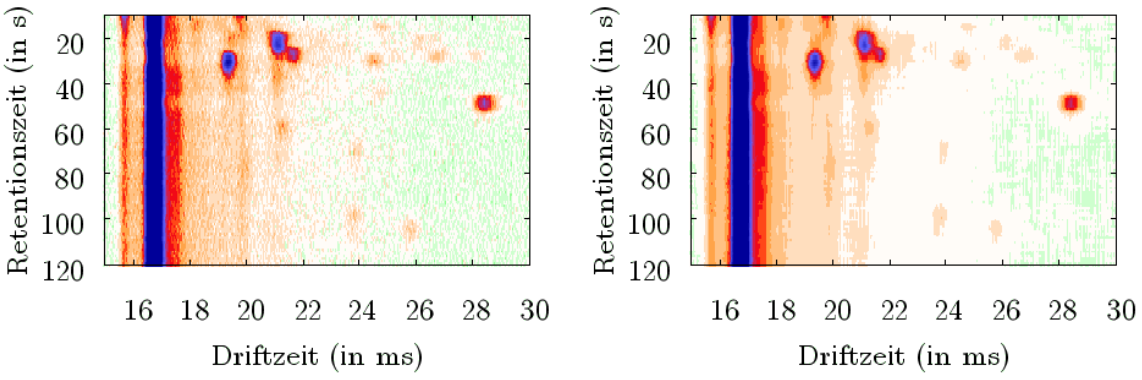

Abbildung 6.10.: Ein Ausschnitt einer Messung (links) und die Anwendung der F-Transformation mit einer aus WT-Thresholding und ϵ-Maxima gewonnenen Fuzzy-Partition in zwei Dimensionen (rechts), die Intensitätsskala entspricht der von Abbildung 6.2

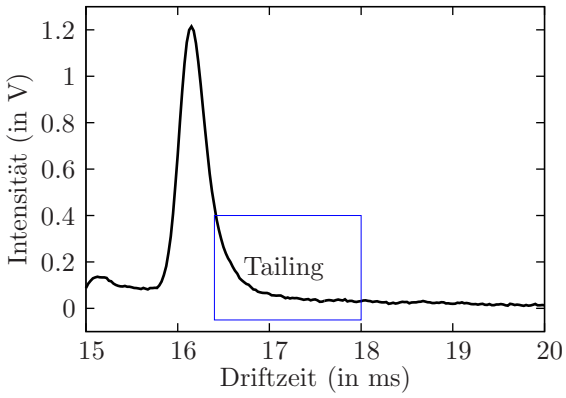

Abbildung 6.11.: Tailing des RIPs

Es ist zu vermuten, dass der Kurvenverlauf der Peaks von einer Wahrscheinlichkeitsfunktion bestimmt wird. Dann ist (zumindest approximativ) eine funktionale Beschreibung möglich. Dazu sei die *Peakfunktion* definiert:

Definition 6.1 (Peakfunktion) *Eine Peakfunktion ist eine Funktion*

$$f : (x; x_0, h, a_1, \ldots, a_n) \mapsto y,$$

die mit den reellwertigen Parametern x_0 (Position des Maximums) und h (Höhe des Maximums), so wie gegebenenfalls weiteren reellwertigen Parametern $a_1, \ldots, a_n$, einen reellen Wert x auf einen reellen Wert y abbildet und die folgenden Bedingungen erfüllt.

1. *$f(x_0) = h > 0$ stellt das einzige Maximum dar.*
2. *$\forall x : f(x) \geq 0$.*

Gilt $f(x_0 + b) = f(x_0 - b)$ für alle $b \in \mathbb{R}$, so wird f als symmetrisch *bezeichnet. Die Klasse von Peakfunktionen zu einer Peakfunktion f mit beliebigen zulässigen Parameterbelegungen wird als $[f]$ notiert.*

Jedes Spektrum entsteht durch eine Überlagerung von Peakfunktionen und Störeinflüssen wie beispielsweise Rauschen. Die Störeinflüsse können mit einer der im Abschnitt 6.1 vorgestellten Methoden reduziert werden, so dass im Idealfall nur das Signal der Peakfunktionen vorliegt. Eine Peakfunktion sollte so gewählt werden, dass sich möglichst alle im Spektrum vorkommenden Peaks durch sie beschreiben lassen. Weitere Parameter müssen so konstruiert sein, dass sie allein durch Änderung von h Peaks mit gleichem Driftzeitmaximum aber anderer Peakhöhe beschreiben.

Jeder eindimensionale Peak ist dann durch eine Peakfunktion im Driftzeit-Intensität-Raum definiert. Beide Begriffe können synonym verwendet werden. Es sei F die Menge aller Peakfunktionen. Die Funktion $Ret : F \to \mathbb{R}$ bildet eine ermittelte Peakfunktion auf die Retentionszeit des korrespondierenden Spektrums ab.

Eine Peakfunktion mit freien Parametern stellt eine Modellfunktion im Sinn vom Abschnitt 4.1 dar, die einen Peak eines bestimmten Ionentyps im Spektrum beschreibt. Das Resultat eines entsprechenden Fittingprozesses ist eine Belegung der Peakfunktionsparameter. Im einfachsten Fall reicht es aus, ϵ-Maxima zu finden und auf diese Weise x_0, h so wie spezifische Parameter für jede Peakfunktion zu bestimmen. Allerdings wird dies durch die Überlagerung verschiedener Peakfunktionen erschwert, denn Überschneiden sich zwei Peakfunktionen f und g, so berechnet sich die Intensität y zur Driftzeit x durch $y = f(x) + g(x)$.

In der Regel werden Maxima großer Peaks nur verhältnismäßig wenig durch kleinere Peaks beeinflusst. Es bietet sich also an, vom einem globalen Maximum sukzessiv nach Peaks zu suchen. Ist eine gefundene Peakfunktion eine gute Approximation des durch die entsprechenden Ionen erzeugten Signals, so kann die sie dazu verwendet werden, dieses Signal mittels Subtraktion von Funktionswerten aus dem Spektrum zu eliminieren. Danach kann dieses Verfahren erneut auf das modifizierte Spektrum angewendet werden. Dies wird so lange wiederholt, bis ein bestimmter Schwellwert $\lambda \geq 0$ für globale Maxima im modifizierten Spektrum erreicht wird.

Da sowohl die Approximation, wie auch die zuvor auf das Spektrum angewendete Glättung, in der Regel nicht optimal sind, können zu große Werte subtrahiert werden. Daher wird die Elimination bezüglich einer Driftzeit x und der korrespondierenden Intensität y mittels $y \to \max(y - f(x), 0)$ durchgeführt. Außerdem ist es ratsam, Spektrumbereiche, die nur durch einen Ionentyp bestimmt werden, von der weiteren Betrachtung auszuschließen. Zum

Einen wird so die Berechnung beschleunigt, zum Anderen kann durch die Angabe von Toleranzparametern unter Umständen die Bildung von unerwünschten Peaks durch Glättungs- und Approximationsfehler vermieden werden. Im Algorithmus 6.2.1 sind die Schritte zusammen gefasst. Die in dieser Arbeit entwickelten Realisierungen der Algorithmuszeilen 9, 10 und 11 werden im Folgenden vorgestellt. Die zur Demonstration verwendeten Daten wurden dabei für das Fitting mit der Methode aus dem Abschnitt 6.1.5 geglättet. In den Abbildungen werden aber zur besseren Abschätzung der Approximsationsgüte bezüglich der realen Spektren die nicht geglätteten Daten gezeigt.

Algorithmus 6.2.1 Fitting von Peakfunktionen

1: Gegeben seien die Intensitäten $y_1, \ldots, y_n$, die zugehörigen Driftzeiten $x_1, \ldots, x_n$, eine Peakfunktionsklasse $[f]$ und ein Schwellwert λ.
2: Erzeuge leere Liste L von Peakfunktionen.
3: Erzeuge leere Liste von Indexmarkierungen.
4: **loop**
5: Suche Index i des ersten globalen Maximums bezüglich nicht markierter Intensitäten.
6: **if** $y_i \leq \lambda$ **then**
7: **return** L.
8: **end if**
9: Erzeuge Peakfunktion $f \in [f]$.
10: Markiere gegebenenfalls eine Indexmenge um i.
11: Subtrahiere Peaksignal vom Intensitätsvektor.
12: Setze $L \to L \cup \{f_p\}$.
13: **end loop**

6.2.1. Lokales Gauss-Breit-Wigner Fitting

Bis auf das Tailing weisen Peaks einen gausskurvenähnlichen Verlauf auf. Die *Gausskurve* der Normalverteilung ist als Dichtefunktion

$$\varphi\left(x;\mu,\sigma^2\right) := \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad \text{mit Maximum} \quad \varphi\left(\mu;\mu,\sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}}$$

definiert [2]. Der Erwartungswert μ kann als der Parameter x_0 einer Peakfunktion interpretiert werden. Eine Skalierung des Maximums auf den Parameter h kann durch

$$\varphi_1\left(x;x_0,h,\sigma^2\right) := h \cdot e^{-\frac{(x-x_0)^2}{2\sigma^2}}$$

erfolgen. Die modifizierte Gausskurve φ_1 stellt bereits eine gültige Peakfunktion dar. Es fehlt jedoch noch eine sinnvolle Transformation des Parameters σ^2. Dazu sei der Begriff der *Halbwertsbreite* definiert:

Definition 6.2 (Halbwertsbreite) *Es sei f eine symmetrische Peakfunktion mit den Parametern $x_0, h, \ldots$. Dann ist die Halbwertsbreite w bezüglich f als*

$$f\left(x_0 + w\right) = \frac{h}{2}$$

definiert. Ist f nicht symmetrisch, so bezeichnet w_+ mit $f\left(x_0 + w_+\right) = h/2$ die positive *und w_- mit $f\left(x_0 - w_-\right) = h/2$ die* negative *Halbwertsbreite.*

Die Halbwertsbreite ist ein intuitiv verständlicher Parameter. Beinhaltet ein Spektrum $(\mathbf{y}, \mathbf{x}, r)$ nur einen einzelnen Peak mit dem Maximum (x_0, h), so können w_+ und w_- außerdem mittels

$$w_+ = x_i - x_0 \qquad \text{mit} \qquad i = \min\left(i \,|x_i > x_0 \wedge y_i \leq h/2\right) \tag{6.6}$$

$$w_- = x_0 - x_i \qquad \text{mit} \qquad i = \max\left(i \,|x_i < x_0 \wedge y_i \leq h/2\right) \tag{6.7}$$

leicht abgeschätzt werden. Um die Halbwertsbreite w als Parameter aufzunehmen, kann mittels

$$h \cdot e^{-\frac{w^2}{2\sigma^2}} = \frac{h}{2} \Leftrightarrow \sigma = \pm\frac{w}{\sqrt{2\log(2)}}$$

eine weitere gausskurvenähnliche Peakfunktion

$$g\left(x; x_0, h, w\right) := h \cdot e^{-\frac{\log(2)(x-x_0)^2}{w^2}} = h \cdot 2^{-\frac{(x-x_0)^2}{w^2}} \tag{6.8}$$

definiert werden. Allerdings wird der Funktionsverlauf eines Peaks im Tailingbereich durch g nur schlecht approximiert. Die *Breit-Wigner-Funktion*, die als Dichtefunktion

$$\sigma\left(x; x_0, \Gamma\right) := \frac{1}{2\pi} \cdot \frac{\Gamma}{\left(x - x_0\right)^2 + \left(\frac{\Gamma}{2}\right)^2}$$

definiert ist [5], liefert eine bessere Beschreibung des Tailings. Unter Verwerfung des Vorfaktors lässt sich mittels

$$a\frac{\Gamma}{\left(\frac{\Gamma}{2}\right)^2} = h \Leftrightarrow a = \frac{h \cdot \Gamma}{4}$$

eine Skalierung auf den Parameter h durch

$$\sigma_1\left(x; x_0, h, \Gamma\right) := \frac{h \cdot \Gamma}{4} \cdot \frac{\Gamma}{\left(x - x_0\right)^2 + \left(\frac{\Gamma}{2}\right)^2} = h \cdot \frac{\Gamma^2}{4 \cdot \left(x - x_0\right)^2 + \Gamma^2}$$

erreichen. Auch hier lässt sich Γ mittels

$$h \cdot \frac{\Gamma^2}{4w^2 + \Gamma^2} = \frac{h}{2} \Leftrightarrow \Gamma = \pm 2w$$

durch die Halbwertsbreite w ersetzen. Es resultiert eine modifizierte Breit-Wigner-Funktion

$$bw\left(x; x_0, h, w\right) := h \cdot \frac{w^2}{\left(x - x_0\right)^2 + w^2}. \tag{6.9}$$

Auch sie ist eine Peakfunktion. Werden g und bw zusammengefasst, so entsteht die im Folgenden als *Gauss-Breit-Wigner-Funktion* bezeichnete Peakfunktion

$$f_{gbw}\left(x; x_0, h, w_g, w_{bw}\right) := \begin{cases} h & \text{falls} \quad x = x_0 \\ g\left(x; x_0, h, w_g\right) & \text{falls} \quad x < x_0 \\ bw\left(x; x_0, h, w_{bw}\right) & \text{falls} \quad x > x_0 \end{cases} \tag{6.10}$$

Dabei wird für jede der beiden Teilfunktionen eine eigene Halbwertsbreite definiert. In der Abbildung 6.12 ist der Funktionsverlauf, so wie ein manuelles Fitting von f_{gbw} dargestellt. Für diese Peakfunktion wurde eine lokale Fittingmethode entwickelt, die im Folgenden vorgestellt wird.

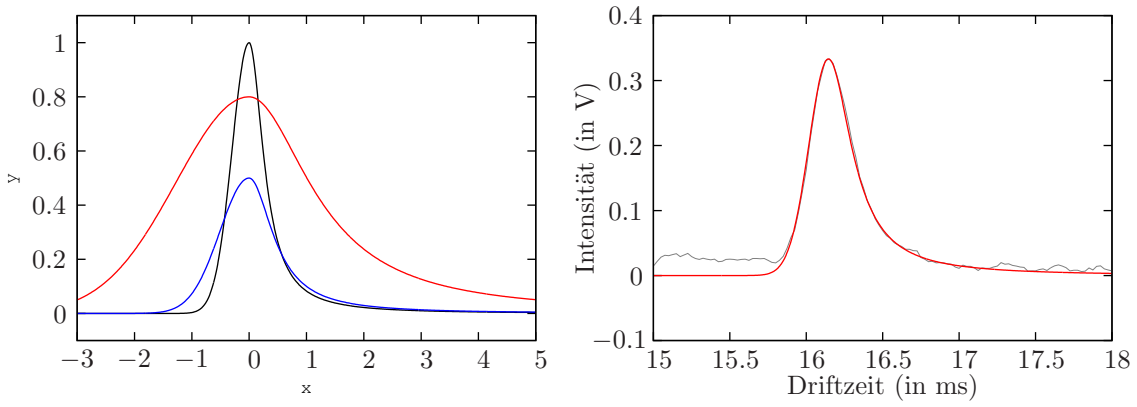

Abbildung 6.12.: Die Peakfunktion f_{gbw} mit verschiedenen Parametern (links) und ein manuelles Fitting (rechts)

Bei stark überlagerten Peaks lässt sich eine Abschätzung wie in den Gleichungen 6.6 und 6.7 nicht vornehmen, da beispielsweise zu der Driftzeit $x_0 - w_g$ das Maximum eines anderen Peaks liegen könnte. Ausgehend vom Maximum muss die Bestimmung von w_g und w_{bw} also iterativ erfolgen. Da $g(x; x_0, h, w) = g(x; x_0, h, -w)$ lässt sich zu einem Punkt (x, y) mit $x < x_0$ und $0 < y < h$ die negative Halbwertsbreite w_g mittels

$$h \cdot 2^{-\frac{(x-x_0)^2}{{w_g}^2}} = y \Leftarrow w_g = \sqrt{\frac{(x-x_0)^2}{\log_2\left(\frac{h}{y}\right)}}$$

berechnen. Analog kann zu einem Punkt (x, y) mit $x > x_0$ und $0 < y < h$ mittels

$$h \cdot \frac{{w_{bw}}^2}{(x-x_0)^2 + {w_{bw}}^2} \Leftarrow w_{bw} = \frac{\sqrt{(x-x_0)^2 \cdot y}}{\sqrt{h-y}}$$

die zugehörige negative Halbwertsbreite w_{bw} berechnet werden.

Beide Peakseiten, also die Bereiche $x < x_0$ und $x > x_0$, werden unabhängig von einander betrachtet. Zuerst wird ein Startfenster der Länge $a \in \mathbb{N}$ festgelegt. Bei den in dieser Arbeit verwendeten Daten hat sich beispielsweise $a = 10$ bewährt. Es sei m der Index des Maximums, dann wird eine mittlere negative Halbwertsbreite $w_g := (w_g(x_{m-1}, y_{m-1}) + \ldots + w_g(x_{m-a}, y_{m-a}))/a$ berechnet. Jeder weitere Datenpunkt (x_i, y_i) mit $i < m - a$ aktualisiert diesen Mittelwert mittels $w_g \to (w_g + w_g(x_i, y_i))/(m - i)$.

Natürlich sollte diese Suche abgebrochen werden, so bald der Kurvenverlauf des Spektrums auch noch durch andere Peaks bestimmt wird. Zu diesem Zweck wurde mit $\xi_g > 1$ ein reellwertiger Parameter eingeführt, der mittels

$$w_g(x_i, y_i) < \xi_x \cdot w_g$$

eine Abbruchbedingung definiert. Die Bestimmung von w_{bw} verläuft mit einem Parameter $\xi_{bw} > 1$ analog. Als günstige Belegungen erwiesen sich $\xi_g = 1.1$ und $\xi_{bw} = 1.3$.

Es fällt auf, dass nur positive Abweichungen berücksichtigt werden. Dies ist damit zu begründen, dass weitere Peaks nur zu größeren $w_g(\cdot, \cdot)$ führen können. Der Grund für die Definition eines Faktors an Stelle eines absoluten Grenzwertes besteht darin, dass auf diese Weise

die Größenordnung der Halbwertsbreite berücksichtigt wird. Da w_g bzw. w_{bw} eine Mittelung aller betrachteten Datenpunkthalbwertsbreiten darstellen, haben die letzten Werte bei einer genügend großen Anzahl von Iterationsschritten kaum Gewicht. Je isolierter der Peak ist, desto besser wird die erzielte Approximation. Insbesondere ist es notwendig, dass der Datenpunktbereich $i-a,\ldots,i+a$ nur durch den betrachteten Peak bestimmt wird.

Der Schritt 9 des Algorithmus 6.2.1 ist somit realisiert. Um den Schritt 10 durchzuführen, können zunächst alle für die Halbwertsbreitenbestimmung verwendeten Datenpunkte als nicht mehr zu betrachten markiert werden. Eine alternative Abbruchbedingung zeigte jedoch bessere Resultate. Die Berechnung eines $x < x_0$ zu einem y mit $0 < y < h$ erfolgt durch

$$h \cdot 2^{-\frac{(x-x_0)^2}{{w_g}^2}} = y \Leftarrow x = x_0 - w_g\sqrt{\log_2\left(\frac{h}{y}\right)}.$$

Die Berechnung eines $x > x_0$ zu einem y mit $0 < y < h$ erfolgt durch

$$h \cdot \frac{{w_{bw}}^2}{(x-x_0)^2 + {w_{bw}}^2} \Leftarrow x = \frac{x_0 \cdot y + \sqrt{{w_{bw}}^2\,(h-y)\,y}}{y}.$$

Mit einem Toleranzparameter $\lambda_x \geq 0$ lassen sich dann ausgehend vom Maximum (exklusive diesem) alle Indizes i markieren, bis die Bedingung

$$y_i - {f_{gbw}}^{-1}(x_i) < \lambda_x$$

verletzt ist. Hierbei bezeichnet ${f_{gbw}}^{-1}$ den zu y_i korrespondierenden x-Wert. Es werden wieder nur positive Abweichungen berücksichtigt.

Im Schritt 11 des Algorithmus kann mit Hilfe reellwertiger Parameter $s_h \geq 1$, $s_{w_g} \geq 1$ und $s_{w_{bw}} \geq 1$ außerdem noch erreicht werden, dass eine größere Peakkurve vom Spektrum subtrahiert wird, als tatsächlich ermittelt wurde. Jede Intensität wird dann mittels

$$y_i \to \max\left(0, f_{gbw}\left(x_i; x_0, s_h \cdot h, s_{w_g} \cdot w_g, s_{w_{bw}} \cdot w_{bw}\right)\right)$$

aktualisiert. Die drei zuletzt genannten Parameter sind allerdings mit Vorsicht zu genießen, da sie unter Umständen wichtige Informationen für benachbarte Peaks eliminieren.

Ein Fitting mit $s_h = s_{w_g} = s_{w_{bw}} = 1$ ist in der Abbildung 6.13 dargestellt. Es stellte sich heraus, dass diese Form des Fittings (bei sorgfältiger Wahl der Parameter) akzeptable Ergebnisse liefert. Allerdings wirkt es recht störend, dass f_{gbw} eine zusammengesetzte Funktion ist. Wünschenswert wäre eine nicht zusammengesetzte Peakfunktion, die dann wohl eher einer Verteilungsfunktion entspricht.

6.2.2. Fitting mit der Lognormalverteilung

Zunächst wurde die Dichtefunktion der *Lognormalverteilung*, die nach WIKIPEDIA[2] als

$$f_{lognorm}(x; m, s) := \frac{1}{\sqrt{2\pi} \cdot s \cdot x} \cdot e^{-\frac{1}{2}\left(\frac{\log(x)-m}{s}\right)^2} \quad \text{mit Maximum} \quad y = \frac{1}{\sqrt{2\pi} \cdot s} \cdot e^{\frac{s^2}{2}-m}$$

definiert ist, mittels

$$f_{ln}(x; x_0, h, m, s) := \begin{cases} 0 & \text{falls} \quad x - x_0 + e^{m-s^2} \leq 0 \\ \frac{h \cdot e^{m-\frac{s^2}{2}}}{x-x_0+e^{m-s^2}} \cdot e^{-\frac{1}{2}\cdot\left(\frac{\log\left(x-x_0+e^{m-s^2}\right)-m}{s}\right)^2} & \text{falls} \quad x - x_0 + e^{m-s^2} > 0 \end{cases}$$

[2] http://www.wikipedia.de

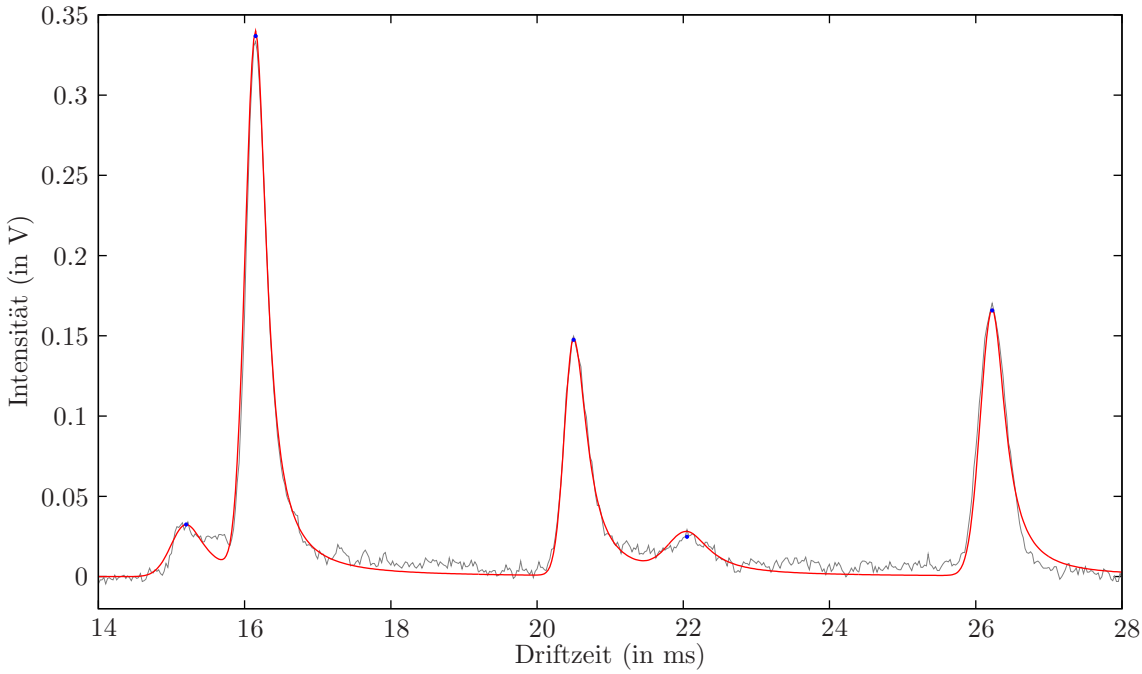

Abbildung 6.13.: Ermittlung eindimensionaler Peaks mit Hilfe der Peakfunktion f_{gbw} und lokalem Fitting

auf x_0 und h parameterisiert. Die Peakfunktion f_{ln} ließ sich aber bei großen Peaks mit ausgeprägtem Tailing nur schlecht anpassen. Eine Teilung wie bei f_{gbw} schien erfolgversprechender, denn bestimmte Belegungen von s scheinen die Tailing-Hälfte gut zu approximieren, während andere Belegungen die verbleibende Hälfte besser beschreiben. Die resultierende Peakfunktion ist als

$$f_{lnln}\left(x; x_0, h, m_1, s_1, m_2, s_2\right) := \begin{cases} 0 & \text{falls} \quad x - x_0 + e^{m_1 - {s_1}^2} \leq 0 \\ f_{ln}\left(x; x_0, h, m_1, s_1\right) & \text{falls} \quad x - x_0 + e^{m_1 - {s_1}^2} < x < x_0 \\ h & \text{falls} \quad x = x_0 \\ f_{ln}\left(x; x_0, h, m_2, s_2\right) & \text{falls} \quad x > x_0 \end{cases} \tag{6.11}$$

definiert. Dabei wurde für die $x < x_0$ Hälfte $s_1 = 0.04$ und für die $x > x_0$ Hälfte $s_2 = 0.4$ gewählt. Um für beide Peakhälften jeweils den Parameter m_1 bzw. m_2 zu bestimmen, wurde sich ein bestimmtes Verhalten der Peaks zu Nutze gemacht: Mit zunehmender Driftzeit des Maximums werden die Peaks immer breiter. Für die Parameter m_1 und m_2 eine lineare Abhängigkeit der Parameter m_1 und m_2 bezüglich der Driftzeit unterstellt. Das Ergebnis war akzeptabel, jedoch schlechter als beim lokalen Fitting mit f_{gbw}. Aus diesem Grund wurde ein Fitting mit der Peakfunktion f_{lnln} nicht weiter untersucht. In der Abbildung 6.14 ist ein Fittingresultat mit $m_1 = 0.6127 + 0.034917 \cdot x_0$ und $m_2 = -1.2 + 0.017 \cdot x_0$ dargestellt, wobei vergrößerte Peaksignale vom Spektrum subtrahiert wurden.

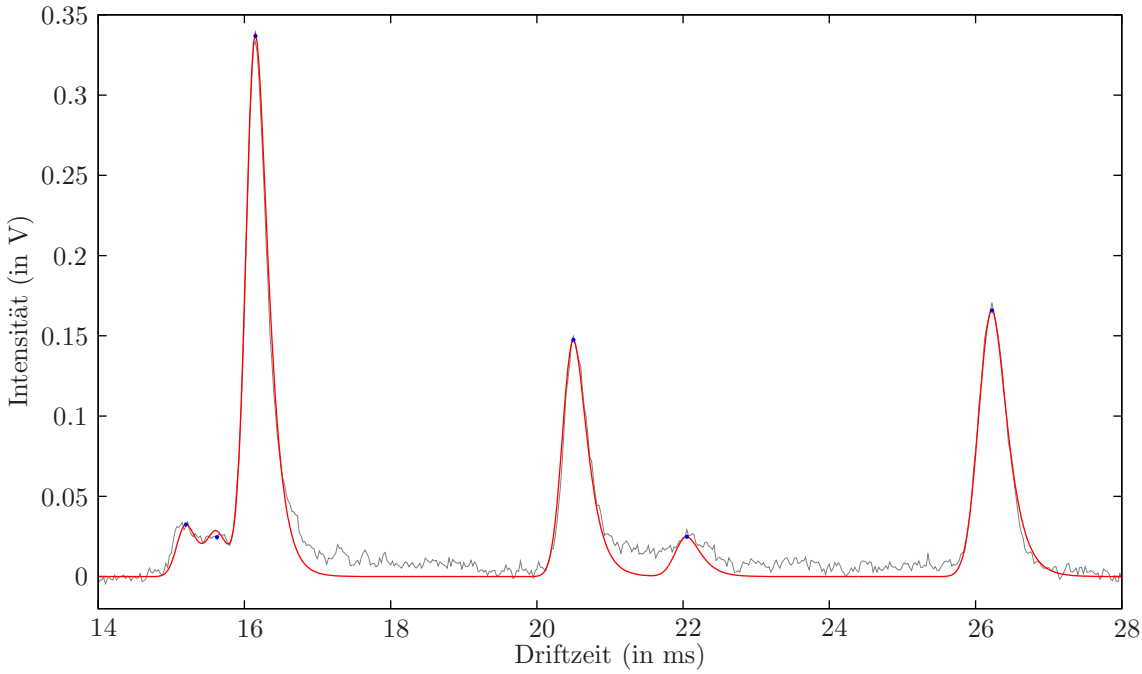

Abbildung 6.14.: Ermittlung von eindimensionalen Peaks mit Hilfe der Peakfunktion f_{lnln}

6.2.3. Log-Cauchy Fitting

Aus der Beschäftigung mit der Lognormalverteilung entstand eine weitere Idee. Die Dichtefunktion der *Cauchy-Verteilung* ist als

$$f_{cauchy}(x; s, t) := \frac{1}{s\pi} \cdot \frac{1}{1 + \left(\frac{x-t}{s}\right)^2}$$

definiert [22]. Die Idee zur Konstruktion einer Peakfunktion bestand nun darin, den Vorfaktor durch h zu ersetzen, das Maximums auf x_0 zu verschieben und motiviert durch die Lognormalverteilung mittels

$$f_{lc1}(x; x_0, h, s) := h \cdot \frac{1}{1 + \left(\frac{\log(x-x_0+1)}{s}\right)^2}$$

einen Logarithmustherm einzubringen. Die Funktion f_{lc1} hat die Eigenschaft, dass der Bereich der $x < x_0 \wedge f_{lc1}(x) > 0$ auf das Intervall $(x_0 - 1, x_0)$ festgelegt ist. Um dieses Intervall skalieren zu können, wird mittels

$$f_{lc2}(x; x_0, h, a, s) := h \cdot \frac{1}{1 + \left(\frac{\log\left(\frac{x-x_0+a}{a}\right)}{s}\right)^2}$$

ein reellwertiger Parameter $a > 0$ eingeführt. Der Parameter s kann mittels

$$h \Big/ \left(1 + \left(\frac{\log\left(\frac{w+a}{a}\right)}{s}\right)^2\right) = \frac{h}{2} \Leftrightarrow s = \pm \log\left(\frac{w+a}{a}\right)$$

von einer positiven Halbwertsbreite w abhängig gemacht werden. Es resultiert die im Folgenden als *Log-Cauchy-Funktion* bezeichnete Peakfunktion

$$f_{lc}\left(x; x_0, h, a, w\right) := \begin{cases} \frac{h}{1+\left(\frac{\log\left(\frac{x-x_0+a}{a}\right)}{\log\left(\frac{w+a}{a}\right)}\right)^2} & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases} . \tag{6.12}$$

Es ergibt sich das Problem, dass im Tailingbereich f_{lc} oft zu langsam abfällt. Eine Lösung besteht darin, den Exponenten im x_0-Abstand-Term auf 3.0 zu erhöhen. Die resultierende Funktion ist durch

$$f_{lce3}\left(x; x_0, h, a, w\right) := \begin{cases} \frac{h}{1+\left|\frac{\log\left(\frac{x-x_0+a}{a}\right)}{\log\left(\frac{w+a}{a}\right)}\right|^3} & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases}$$

gegeben. Diese Funktion führt zu dem Problem, dass das Maximum sehr „rund“ ausfällt, während hohe Peaks in den Spektren zum Maximum hin eher spitz zulaufen. Als Ausweg wurde eine Interpolation zwischen beiden Peakfunktionen mit einem reellwertigen Parameter $\alpha \in [0, 1]$ gewählt. Die entsprechende Peakfunktion ist dann als

$$f_{lcie2+3}\left(x; x_0, h, a, w, \alpha\right) := \begin{cases} (1-\alpha) \cdot f_{lc}\left(x; x_0, h, a, w\right) \\ \quad +\alpha \cdot f_{lce3}\left(x; x_0, h, a, w\right) & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases} \tag{6.13}$$

definiert. In der Abbildung 6.15 sind die drei Funktionen vergleichend dargestellt.

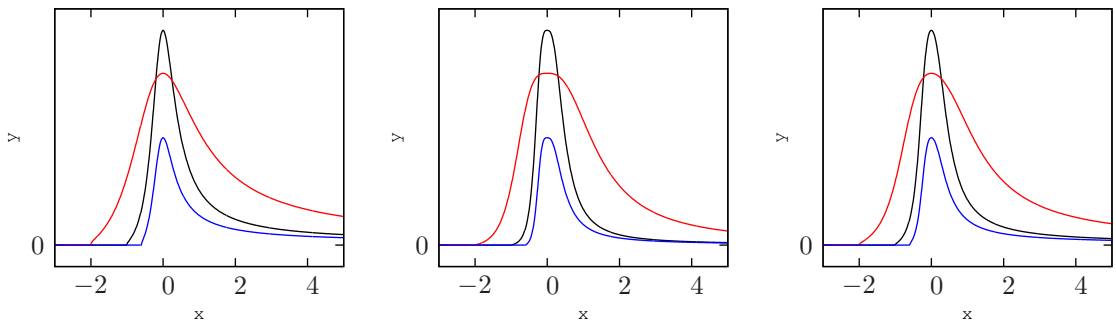

Abbildung 6.15.: Funktionsverläufe der Peakfunktionen f_{lc} (links), f_{lce3} (mittig) und $f_{lcie2+3}$ mit $\alpha = 0.5$ (rechts)

Es wurde eine logarithmische Abhängigkeit zwischen Driftzeit des Maximums und den Parametern a und w unterstellt. Zur Indexmarkierung kann beispielsweise auf Grund der schwereren analytischen Handhabbarkeit das Kriterium

$$y_i - f_{lcie2+3}\left(x_i; x_0, h, a, w, \alpha\right) < \lambda_y$$

genutzt werden, wobei $\lambda_y > 0$. Außerdem werden wie bei f_{gbw} Parameter s_h, s_a und s_w definiert, die einen skalierten Peak vom Spektrum subtrahieren. Ein Fitting mit $\alpha = 0.5$, $a = 0.387 \cdot \log\left(x_0\right)$ und $w = 0.0645 \cdot \log\left(x_0\right)$ ist in der Abbildung 6.16 dargestellt.

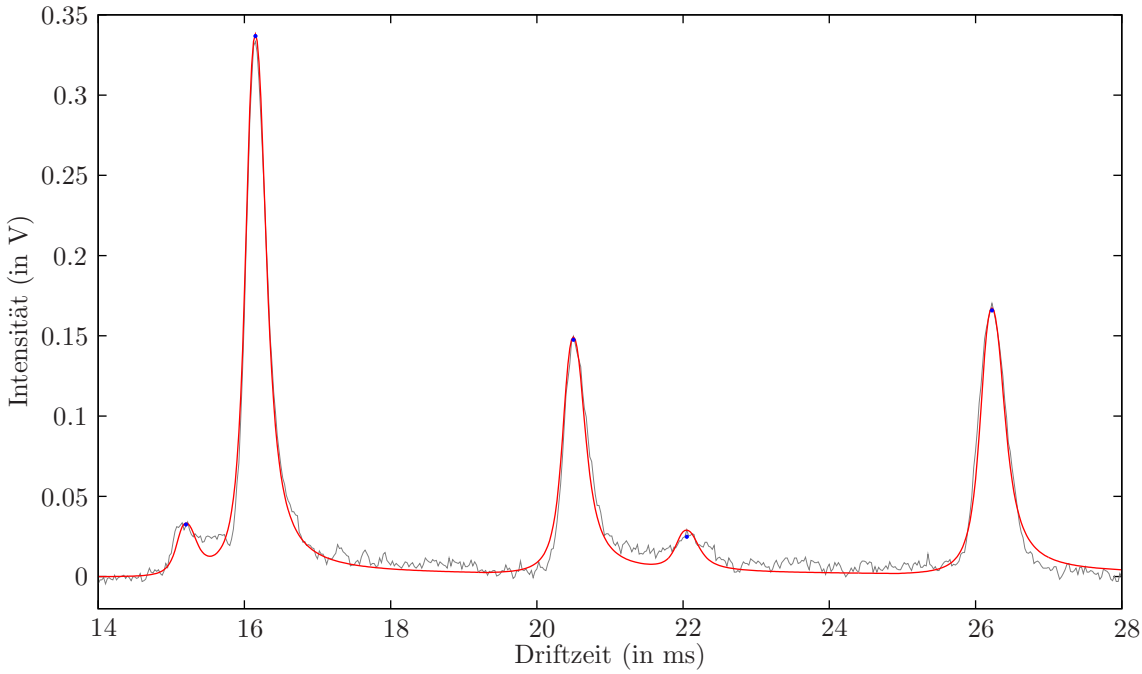

Abbildung 6.16.: Fitting mit der Peakfunktion $f_{lcie2+3}$

6.2.4. Log-Breit-Wigner Fitting

Analog zu f_{lc} kann die Breit-Wigner-Funktion logarithmiert werden. Die nach einer Vereinfachung resultierende Peakfunktion ist durch

$$f_{lbw}\left(x; x_0, h, a, w\right) := \begin{cases} \frac{h \cdot \log\left(\frac{a+w}{a}\right)^2}{\log\left(\frac{a+w}{a}\right)^2 + \log\left(\frac{a+x-x_0}{a}\right)^2} & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases} \tag{6.14}$$

gegeben. Eine Ersetzung des quadratischen Exponenten durch 3.0 führt zu

$$f_{lbwe3}\left(x; x_0, h, a, w\right) := \begin{cases} \frac{h \cdot \left|\log\left(\frac{a+w}{a}\right)\right|^3}{\left|\log\left(\frac{a+w}{a}\right)\right|^3 + \left|\log\left(\frac{a+x-x_0}{a}\right)\right|^3} & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases} \tag{6.15}$$

und $f_{lbwie2+3}$ bezeichnet wieder die Interpolation zwischen f_{lbw} und f_{lbwe3} mit Hilfe des Parameters $\alpha \in [0, 1]$. Analog zu $f_{clie2+3}$ wurde den Parametern a und w eine logarithmische Abhängigkeit bezüglich x_0 unterstellt. In der Abbildung 6.17 ist ein Fitting mit $a = 0.357083 \cdot \log\left(x_0\right)$, $w = 0.07 \cdot \log\left(x_0\right)$ und $\alpha = 0.5$ dargestellt. Das Resultat weicht von dem in der Abbildung 6.16 nur wenig ab, was wohl auf die Ähnlichkeit der verwendeten Peakfunktionen zurückzuführen ist.

Zusammenfassend lässt sich feststellen, dass keine Peakfunktion ideal ist. Insbesondere wenn zwei benachbarte Peaks nur geringen Abstand haben, wird oft ein „dritter Peak“ zwischen diesen ermittelt. Dies lässt sich durch Indexmarkierungen aber in vielen Fällen umgehen. In dieser Arbeit wurde außerdem der Fall vernachlässigt, dass zwei Peaks so nah nebeneindander liegen, dass sie durch die Abtastung als ein Peak erscheinen. Dieser Fall lässt sich wahrscheinlich mit Driftzeitabhängigkeiten lösen, war jedoch für diese Arbeit nicht relevant und bleibt einer anderen Arbeit überlassen. Von den entwickelten Peakfunktionen stellt f_{gbw} die Bevorzugte da,

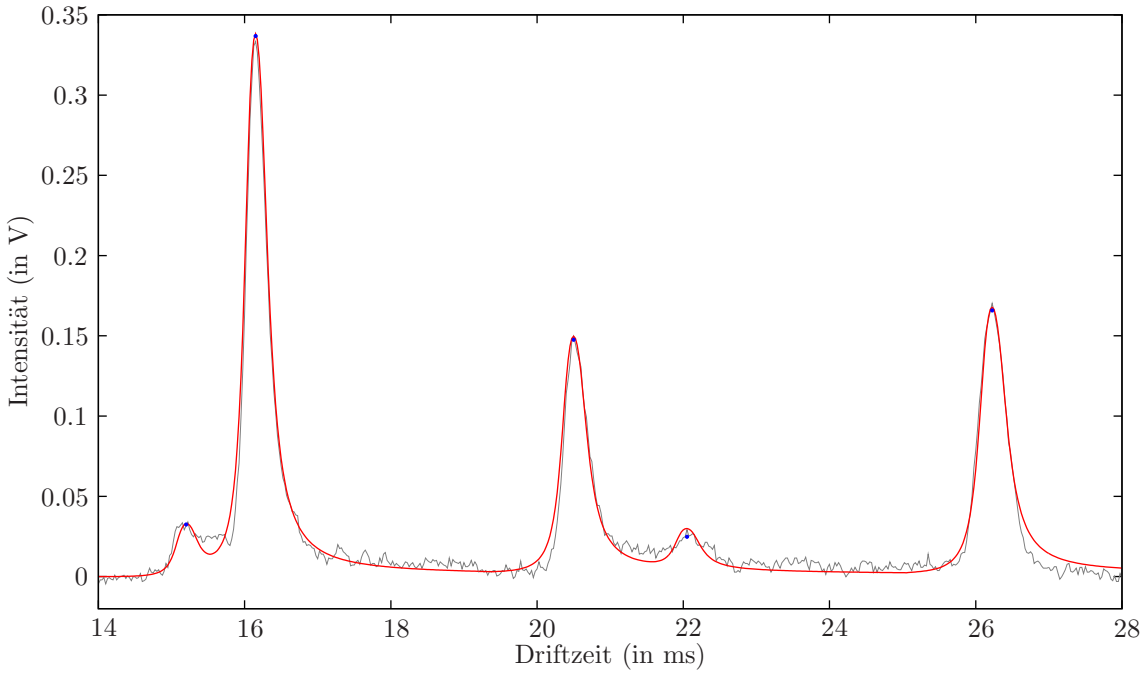

Abbildung 6.17.: Fitting mit der Peakfunktion $f_{lbwie2+3}$

allerdings zeigen die Beispiele, dass keine der hier vorgestellten Funktion absolut „unbrauchbar“ ist. Es bleibt weiteren Arbeiten überlassen, die Suche nach einer idealen Funktion und einer Fittingmethode innerhalb von Spektren fortzusetzen. In dieser Arbeit wurde sich mit der beispielsweise durch das lokale f_{gbw}-Fitting erreichten Approximationsgenauigkeit begnügt, da die verwendeten Daten selbst nach einer Glättung ebenfalls Ungenauigkeiten aufweisen. Die lokale Fittingmethode in Verbindung mit f_{gbw} bietet den Vorteil, dass nicht im Vorfeld Abhängigkeiten zu x_0 ermittelen sind und somit eine universelleres Fitting eindimensionaler Peaks ermöglicht wird.

6.3. Kettenbildung

Nachdem die eindimensionalen Peaks einer Messung $(\mathbf{X}, \mathbf{d} \in \mathbb{R}^n, \tilde{\mathbf{r}} \in \mathbb{R}^m)$ ermittelt wurden, resultieren die zu den Spektren korrespondierenden Peaklisten $L_1, \ldots, L_m$. Die enthaltenen Peakfunktionen werden als $f_{i,j}$ mit $1 \leq i \leq m$ und $1 \leq j \leq n_i$ notiert. Es wird vorausgesetzt, dass die Peaks innerhalb dieser Listen aufsteigend nach Maximumdriftzeit x_0 geordnet sind. Leere Listen sind zulässig. Da Peaks gleicher Ionen auch in verschiedenen Spektren gleiche x_0 aufweisen sollten, liegt es nahe, die entsprechenden Peakfunktionen als eine *Kette* zusammenzufassen:

Definition 6.3 (Kette) *Eine Kette der Länge k ist eine Liste $(f_1), \ldots, (f_k)$ von Peakfunktionen. Die f_i sind aufsteigend nach Retentionszeiten geordnet: $Ret(f_i) < Ret(f_{i+1})$.*

Dazu wird eine zusätzliche Liste L_A erzeugt und alle Peakfunktionen in diese absteigend nach Peakhöhe h geordnet eingefügt. L_A gibt die Bearbeitungsreihenfolge vor. Die zu Grunde liegende Idee besteht darin, dass die x_0 größerer Peaks wahrscheinlich weniger fehleranfällig als die kleinerer Peaks sind.

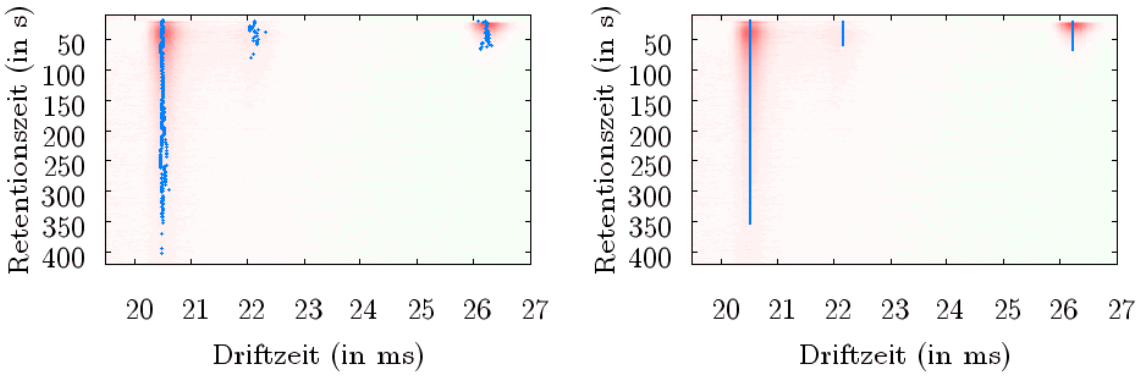

Abbildung 6.18.: Bildung von Ketten (rechts) aus erkannten eindimensionalen Peaks (links)

Es sei $f_{s,t}$ die erste Peakfunktion in L_A. Sie wird aus L_A und L_s eliminiert und eine neue Kette K so wie eine Variable $\tilde{x}_0$ für die mittlere Maximumdriftzeit angelegt. Initial gilt $K = \{f_{s,t}\}$ und $\tilde{x}_0 = x_{0s,t}$. Zur Vereinfachung sei $1 < s < m$, f_C bezeichne die aktuell betrachtete Peakfunktion, zunächst $f_C = f_{s,t}$. Zusätzlich wird ein „Lückenindexzähler“ $g = 0$ initialisiert und eine maximal zulässige Indexlücke mit dem ganzzahligem Parameter $g_M \geq 0$ festgelegt. Auf diese Weise können kleine Peaks, die durch ungünstige Glättung oder Approximation nicht zu Peakfunktionen führten, „übersprungen“ werden. Darüber hinaus wird ein Zähler $u = 1$ initialisiert. Seine Aufgabe besteht darin, die Anzahl der zusammenhängenden Spektrenindizes seit der letzten Lücke festzuhalten. Ein ganzzahliger Parameter $u_M \geq 1$ gibt die minimale Größe einer zusammenhängenden Teilkette an.

Nun wird die Liste L_{s-1} betrachtet und (falls vorhanden) die Peakfunktion f_N mit dem geringsten x_0 Abstand zu $\tilde{x}_0$ ermittelt. Dabei sind die L_i nach x_0 geordnet, so dass zur Beschleunigung eine binäre Suche ermöglicht wird. Falls mit einem reellwertigen Toleranzparameter $\lambda_d \geq 0$ die Bedingung

$$|x_0 - \tilde{x}_0| \leq \lambda_d \tag{6.16}$$

erfüllt ist, erfolgt ein Aktualisierung von K mittels $K \to K \cup \{f_N\}$ und einer Neuberechnung von $\tilde{x}_0$ als Mittelwert der x_0 aller nun in K enthaltenen Peakfunktionen. Die Peakfunktion f_N wird aus L_A und L_{s-1} gelöscht. Außerdem werden $g \to 0$, $p_C \to f_N$ und $u \to u + 1$ gesetzt. Diese Prozedur wird dann für die verbleibenden $L_{s-2}, \ldots, L_1$ fortgesetzt.

Ist die Bedingung 6.16 verletzt oder es existiert keine weitere Peakfunktion, so wird der Lückenindexzähler mittels $g \to g + 1$ aktualisiert. Gilt $g \leq g_M$ und $u \geq u_M$, so wird wie beim Erfüllen der Bedingung 6.16 verfahren, allerdings werden $g \to g + 1$ und $u \to 0$ gesetzt. Andernfalls wird die Suche in diese Richtung abgebrochen und die L_i mit $i > s$ betrachtet. Initial werden dabei $f_C = p_{s,t}$, $g = 0$ und u entsprechend des mit $p_{s,t}$ zusammenhängenden Teilbereichs von K gesetzt. Das Verfahren verläuft analog.

Ist die Suche in beide Richtungen terminiert, so kann noch mittels eines reellwertigen Parameters $r_R > 0$ die Bedingung

$$Ret\,(f_k) - Ret\,(f_1) \geq r_R$$

überprüft werden. Ist sie verletzt, so wird die gefundene Kette verworfen.

Die Kettenbildung erfolgt so lange, bis L_A leer ist. In der Abbildung 6.18 ist eine Kettenbildung dargestellt. Der wichtigste Parameter ist λ_d. Er sollte nicht größer als eine Gitteröff-

nungszeit gewählt werden. Als günstige Parameter stellten sich außerdem $1 \leq g_M \leq 6$ und $g_M < u_M$ heraus. Dabei besteht die Aufgabe von g_M allerdings nur darin, kleine Lücken auf Grund schlechter Glättung oder Approximation zu behandeln. Ein anderes Problem stellen Lücken dar, die aus tatsächlichen Unterbrechungen eines Peaks resultieren. Beispiele sind die Spektren, in denen alle Ionen des RIPs „verbraucht“ sind oder in denen ein Dimer auftritt, das einem Monomer alle Ionen entzieht. Dann müssen Ketten mit (nahezu) gleichen $\tilde{x}_0$ verbunden werden, wenn zu den Retentionszeiten der Lücke Peaks einer anderen Kette auftreten, die ansonsten zumindest weniger ausgeprägt sind.

6.4. Erzeugung zweidimensionaler Peaks

Nachdem zu einer Messung Ketten $K_1, \ldots K_n$ gebildet wurden, muss eine funktionale Beschreibung der so entstandenen Information erfolgen. Allgemein ist dazu eine Funktion

$$f : (x, y) \mapsto 0 \leq f(x, y; x_0, y_0, h, \ldots) \leq f(x_0, y_0) = h$$

zu definieren. Hierbei geben $h > 0$ die Intensität am Maximum und x_0, y_0 die Drift- bzw. Retentionszeit des Maximums an. Im Folgenden sei eine einzelne Kette K betrachtet.

Die einfachste Defintion eines zweidimensionalen Peaks p ist durch die Interpolation der Spektrumfunktionen in K gegeben:

$$p_{SiIn}(x, y) := \begin{cases} 0 & \text{falls } y \notin [Ret(f_1), Ret(f_n)] \\ f_i(x) & \text{falls } y = Ret(f_i) \\ f_{INT}(x) & \text{sonst} \end{cases} \tag{6.17}$$

Die Peakfunktion f_{INT} wird benötigt, falls y zwischen den Retentionszeiten der Peaks f_i und f_j liegt. Jedes ihrer Parameter $a \in \{x_0, h, a_1, \ldots, a_m\}$ ist als

$$a = a_{f_i} + \frac{a_{f_j} - a_{f_i}}{Ret(f_j) - Ret(f_i)}$$

definiert. Die Parameter x_0, y_0 und h entsprechen x_0, h und $Ret(f_k)$ eines eindimensionalen Peaks maximaler Höhe.

Es bezeichne $\langle f \rangle \subset [f]$ die Klasse der Peakfunktionen mit h als einzig freiem Parameter. Außerdem sei eine reellwertige Funktion $f : f(x; \ldots, h, \ldots) \mapsto y$ als *Höhenfunktion* bezeichnet, falls $0 \leq h(x) \leq h$ für alle x gilt. $\langle h \rangle$ Von Peakfunktionen wurde gefordert, dass sie Peaks mit gleichem x_0 aber unterschiedlichem h auch nur mittels eine Änderung von h darstellen können[3]. Deshalb kann ein zweidimensionaler Peak über eine Peakfunktionsklasse $\langle f \rangle$ und eine Höhenfunktion definiert werden:

Definition 6.4 (zweidimensionale Peakfunktion) *Eine zweidimensionale Peakfunktion f zu einer Peakfunktionsklasse $\langle f_x \rangle$ und einer Höhenfunktion f_y ist als*

$$f : (x, y; x_0, y_0, h, \langle f_x \rangle, f_y) \mapsto f_x(x; x_0, f_y(y; \ldots, h, \ldots), \ldots) \tag{6.18}$$

definiert. Dabei muss die Bedingung

$$\forall x, y : 0 \leq f(x, y) \leq f(x_0, y_0) = h$$

erfüllt sein.

[3] Alle im Abschnitt 6.2 vorgestellten Funktionen scheinen diese Forderung zu erfüllen.

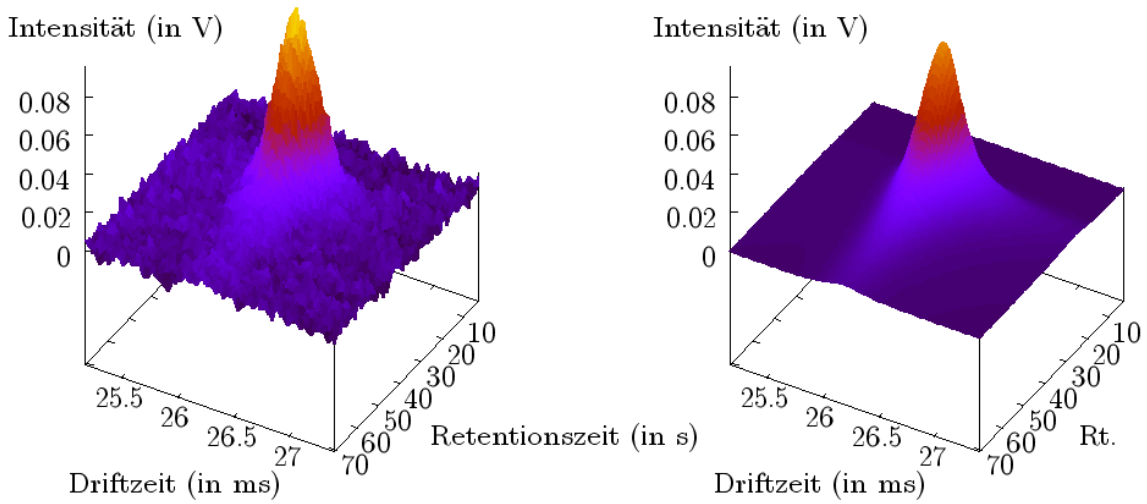

Abbildung 6.19.: Fitting eines zweidimensionalen Peaks (links) mit f_{gbw} als Peak- und Höhenfunktion (rechts)

Eine einfache Höhenfunktion ist durch die Interpolation der Liste von $(Ret\,(f_i)\,, h_i)$-Tupeln gegeben, wobei die f_i die in K enthaltenen Peakfunktionen bezeichnen. Diese Liste kann durch Extrapolation um zwei Randpunkte $(r_0, 0)$ und $(r_{n+1}, 0)$ erweitert werden um ein abruptes Springen der Funktionswerte auf Null bei $ret\,(f_1)$ und $ret\,(f_n)$ zu vermeiden. Eine Extraploation kann mittels der Approximation des Funktionsverlaufs im Randbereich durch Geraden erfolgen. Diese Höhenfunktion wird im Folgenden als f_{In} bezeichnet.

Verbleibende Parameter von $\langle f_x \rangle$, insbesondere x_0, können durch den Median oder ein gewichtetes Mittel festgelegt werden. Sei k der Listenindex des höchsten Spektrumpeaks. Dann kann der gewichtete Mittelwert eines Parameters a analog zur Rouletterad-Wahl (Algorithmus 4.2.2) mittels

$$a = \sum_{i=1}^{n} \frac{h_i}{h_k} \cdot a_i$$

berechnet werden. Auf diese Weise werden hohe (eindimensionale) Peaks, die in der Regel weniger fehlerbelastet sind, stärker berücksichtigt. Um eine ermittelte x_0-Abhängigkeit anderer Parameter zu gewährleisten, können formal die entsprechenden Berechungen die Parameter in den Peakfunktionen ersetzen. Alternativ kann für jede Peakfunktionsklasse eine individuelle Erzeugung aus K definiert werden.

Insbesondere stellt jede Peakfunktion eine gültige Höhenfunktion dar. Natürlich lassen sich nicht alle zweidimensionalen Peaks auf diese Weise darstellen. So ist beispielsweise der RIP, falls keine anderen Ionen in den Driftraum gelangen, in jedem Spektrum in etwa gleich stark ausgeprägt. Analyte sind aber nicht davon betroffen, denn die Vortrennungseinheit sollte die entsprechenden Moleküle nach einer Wahrscheinlichkeitsfunktion um ein Maximum im Retentionszeitraum herum verteilt in das IMS gelangen lassen. In der Abbildung 6.19 ist ein Fitting eines solchen bezüglich des Retentionszeitraums lokalen Peaks mit f_{gbw} als Höhenfunktion dargestellt.

Eine Höhenfunktion f_{gbw} führt jedoch bei größeren Peaks zu einer schlechten Approximation im Tailing-Bereich. Das Tailing über die Retentionszeiten ist viel ausgeprägter, als es

durch die Breit-Wigner-Funktion beschrieben werden kann. Auch keine der anderen Peakmodellfunktionen schien universell geeignet. Eine bessere Approximation boten generalisierte Versionen der Funktionen f_{lc} und f_{lbw}:

$$f_{lce}\left(x; x_0, h, a, w, p\right) := \begin{cases} \frac{h}{1+\left|\frac{\log\left(\frac{x-x_0+a}{a}\right)}{\log\left(\frac{w+a}{a}\right)}\right|^p} & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases} \tag{6.19}$$

$$f_{lbwe}\left(x; x_0, h, a, w, p\right) := \begin{cases} \frac{h \cdot \left|\log\left(\frac{a+w}{a}\right)\right|^p}{\left|\log\left(\frac{a+w}{a}\right)\right|^p + \left|\log\left(\frac{a+x-x_0}{a}\right)\right|^p} & \text{falls} \quad x - x_0 + a > 0 \\ 0 & \text{sonst} \end{cases} \tag{6.20}$$

Die Generalisierung besteht in der freien Wahl des Exponenten, als günstig erwies sich $1 \leq p \leq 3$.

Mit diesen Funktionen können aus Ketten zweidimensionale Peakfunktionen erzeugt werden indem ein eindimensionales Fitting auf den $(Ret\left(f_i\right), h_i)$-Tupeln der eindimensionalen Peaks durchgeführt wird. Es bietet sich die Verwendung eines evolutionären Algorithmus an, wobei die Wahl auf eine $(\mu + \lambda)$-ES fiel ohne Rekombination und mit Selbstadaption nach Hans-Paul Schwefel(vgl. Abschnitt 4.2.2) fiel. Dabei darf die initiale Population auch kleiner als μ sein, eine größere Individuenzahl wird allerdings durch Bestenauswahl auf μ begrenzt. Jeder Parametervektor einer Höhenfunktion stellt ein Individuum dar. Als Fehlerfunktion wird ein gewichteter Least-Squares-Fehler eingesetzt. Die Gewichtung erfolgt mittels $a, b > 0$ durch

$$E\left(\mathbf{r}, \mathbf{h}, \mathbf{p}_{f_y}\right) := \sum_{i=1}^{n} w \cdot \left(h_i - f_y\left(r_i\right)\right)^2 \quad \text{mit} \quad w := \begin{cases} a & \text{falls } h_i - f_y\left(r_i\right) < 0 \\ b & \text{sonst} \end{cases}$$

Dabei stellen $\mathbf{r}$ und $\mathbf{h}$ die Retentionszeiten bzw. Peakhöhen von K und $\mathbf{p}_{f_y}$ die zu optimierenden Funktionsparameter dar. Der Grund, auf eine Rekombination zu verzichten, liegt darin, dass Halbwertsbreiten bei unterschiedlichem Maximum nicht miteinander vergleichbar sind. Unzulässige Parameter werden auf den maximalen Fehlerwert ∞ abgebildet. Für die initiale Population werden zunächst das Maximum und die positive Halbwertsbreite wie im Abschnitt 6.2.1, so wie der Parameter a über die erste Retentionszeit von K abgeschätzt. Aus diesen Informationen werden Individuen mit $p = 1, 2, 3$ erzeugt. Die Suche wird nach dem Erreichen einer maximalen Generationenzahl abgebrochen.

Diese Methode kann auch dazu verwendet werden, auf folgendes Problem einzugehen. Manchmal sind Peaks „beschädigt“. Dies betrifft vor Allem Monomere, wenn ein Dimer auftritt und die Moleküle entzieht. Das Ergebnis ist ein ausgeprägtes lokales Minimum zu den entsprechenden Retentionszeiten, im Extemfall sogar das Erreichen der Basislinie. Ist ein Peak nicht zu stark beschädigt, so besteht die Hoffnung ihn rekonstruieren zu können. Dazu werden ϵ-Minima (vgl. Abschnitt 6.1.5) gesucht und die entsprechenden Intervalle aus der Liste entfernt. Gleichzeitig wird aber versucht an den Intervallgrenzen durch Geraden zu extrapolieren. Der Schnittpunkt zweier solcher Geraden wird als Maximum interpretiert und a, w abgeschätzt. Die Individuen mit unterschiedlichen Exponenten werden der initialen Population hinzugefügt. Ein weiterer Satz von initialen Individuen entsteht durch die Multiplikation der im vorherigen Schritt geschätzten Peakhöhen mit einen konstanten Faktor wie beispielsweise Drei. Das Verfahren ist leider nicht immer zuverlässig, kann jedoch zu einer Verbesserung führen. Die Abbildung 6.20 zeigt ein Ergebnis einer $(10 + 10)$-ES bei 50 Generationen. Der zugehörige Peak ist in der Abbildung 6.21 dargestellt.

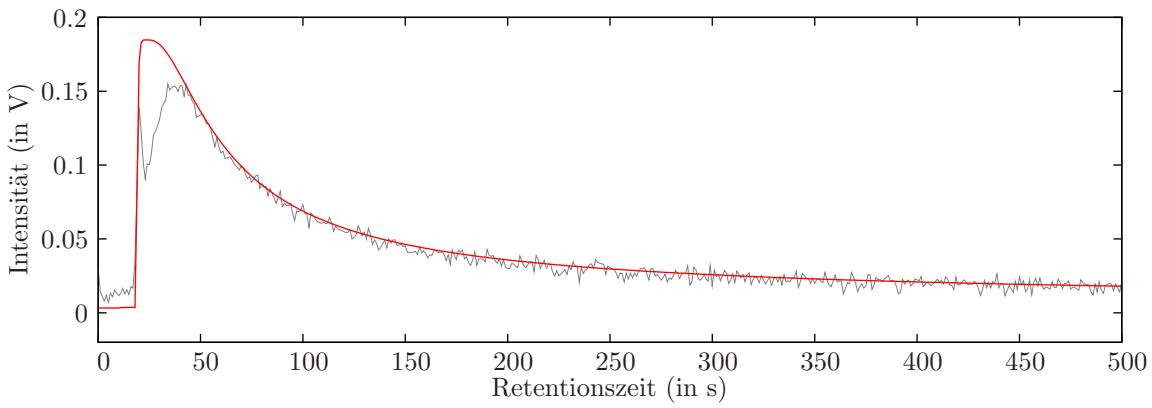

Abbildung 6.20.: Rekonstruktion eines beschädigten Peaks durch Anwendung einer ES auf eine Höhenfunktion f_{lce}

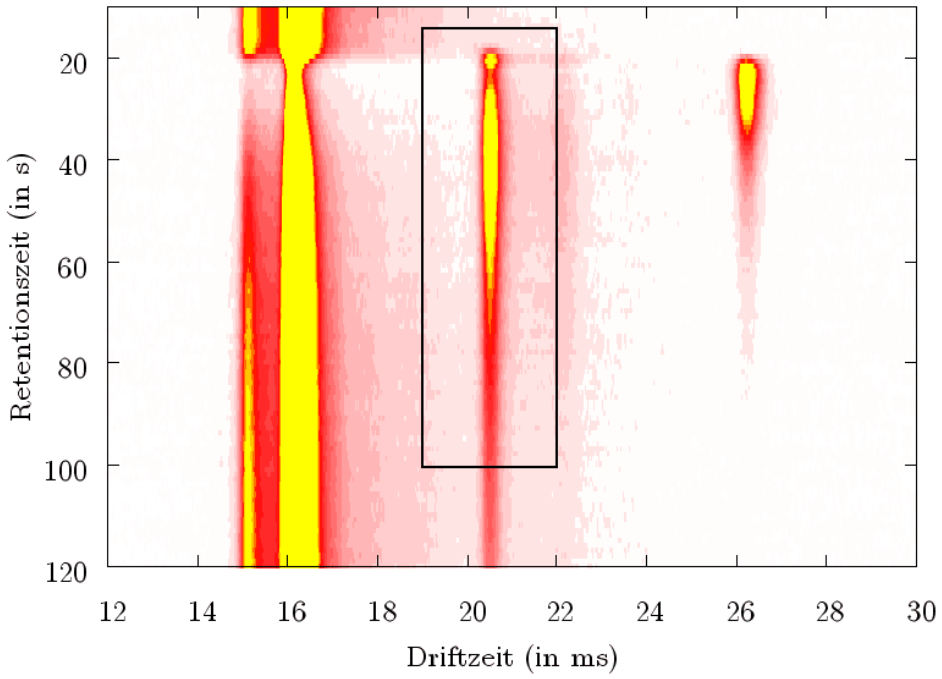

Abbildung 6.21.: Der beschädigte Peak zur Abbildung 6.20

7. Konzentrationsabhängigkeitsanalyse

Das eigentliche Ziel dieser Arbeit bestand in der Entwicklung einer Methode, mit der eine Abhängigkeit von Substanzkonzentrationen und Messungsinformationen untersucht werden kann. Ist die Konzentration (hier in μg/L) einer Substanz größer Null, so durchlaufen ihre Moleküle die Vortrennungseinheit und gelangen konzentriert um eine bestimmte Retentionszeit in das IMS. Dort werden diese durch eine geeignete Ionisationsquelle oder durch Reaktionsionen ionisiert, gelangen in den Driftraum und beeinflussen das von der Datenerfassungskarte aufgezeichnete Signal (vgl. Kapitel 2). Wird die resultierende Messung mit den im vorherigen Kapitel vorgestellten Methoden erfolgreich analysiert, so korrespondieren ein oder mehrere spezifische Peaks in Form zweidimensionaler Peakfunktionen zu der Substanz in der gegebenen Konzentration. Dies können beispielsweise Monomer und Dimer sein. Es liegt also nahe, die Untersuchung einer Konzentrationsabhängigkeit auf der Basis erkannter Peaks durchzuführen.

Ausgangspunkt einer Konzentrationsabhängigkeitsanalyse ist eine Messungsreihe. Zu jeder Messung korrespondiert eine als bekannt vorausgesetzte Konzentrationsangabe. In das IMS geleitete Gasgemische sollten dabei möglichst nur die betrachtete Substanz beinhalten, was aber schon allein wegen der Verwendung von Träger- und Driftgas nicht gewährleistet werden kann. Zur Einfachheit ist jede Messungsreihe aufsteigend nach Konzentrationen geordnet, was einer umgekehrten Sortierung der Messungen einer exponentiellen Verdünnung entspricht.

Damit Peaks unterschiedlicher Messungen vergleichbar gemacht werden können, muss eine *Normalisierung* erfolgen. Sie wird im Abschnitt 7.1 vorgestellt. Anschließend werden zusammengehörige Peaks verschiedener Messungen ermittelt. Mit Hilfe der Konzentrationsangaben entstehen so *Peakverläufe*. Jeder Peakverlauf ist dabei eine geordnete Liste von Peaks und korrespondierenden Konzentrationen, so wie Bezugskoordinaten im (normalisierten) Driftzeit-Retentionszeit-Raum. Eine Lösung wird im Abschnitt 7.2 vorgestellt.

Jeder Peak ist zweidimensional. Es liegt also nahe, eine Konzentrationsabhängigkeitsanalyse mit Hilfe von Peakvolumina zu realisieren. Der Abschnitt 7.3 geht auf die *Volumenberechnung* ein. Nachdem für jeden Peakverlauf die Volumina berechnet wurden, kann schließlich die eigentliche *Konzentrationsabhängigkeitsanalyse* erfolgen. Dieser Schritt wird im Abschnitt 7.4 vorgestellt.

Das zur Konzentrationsabhängigkeitsanalyse komplementäre Verfahren ist die *Konzentrationsbestimmung*. Dies bedeutet, dass mit Hilfe einer ermittelten Konzentrationsabhängigkeit die Konzentration einer Substanz in einer Einzelmessung „vorhergesagt“ wird. Im Abschnitt 7.5 wird darauf eingegangen.

Abschließend werden im Abschnitt 7.6 Ergebnisse von konkreten Konzentrationsabhängigkeitsuntersuchungen vorgestellt. Betrachtet wurden die Substanzen 2-Heptanon und 2-Octanon.

7.1. Normalisierung

Ein allgemeiner Messungspeak verfügt über die drei Parameter x_0, y_0 und h, wobei Letzterer eine eher quantitative Information liefert. Ein (x_0, y_0)-Tupel ist ein Bezugspunkt im Driftzeit-Retentionszeit-Raum, der allerdings durch verschiedenste Einflüsse bestimmt wird

(vgl. Kapitel 2). Selbst unter Verwendung des gleichen Analysesystems führen unterschiedliche Druck- und Temperaturbedingungen zu unterschiedlichen Koordinaten. Damit sie vergleichbar werden muss eine Normalisierung erfolgen.

Eine etablierte Driftzeitnormalisierung bezüglich Druck und Temperatur ist durch die K_0-Berechnung in der Gleichung 2.3 auf Seite 3 gegeben. Die Messungsparameter Temperatur T und Druck p sind in den abweichenden Einheiten °C statt K und hPa statt kPa angegeben. Die Einheiten der Feldstärke E (V/cm) und Driftraumlänge L (cm) stimmen mit denen der K_0-Definition überein. Driftzeiten liegen in ms statt s vor. Außerdem wird am ISAS - Institute for Analytical Sciences noch von jeder Driftzeit x die halbe Gitteröffnungszeit g (in μs) subtrahiert. Somit wird die Gleichung 2.3 zu

$$K_0\left(x\right) = \frac{1000 \cdot L}{\left(x - \frac{g}{2 \cdot 1000}\right) \cdot E} \cdot \frac{273.15}{T + 273.15} \cdot \frac{0.1 \cdot p}{101.325}. \tag{7.1}$$

Damit die Reihenfolge von Driftzeiten erhalten bleibt, wird hier jedoch eine normalisierte Driftzeit als

$$Norm_{K_0}\left(x\right) := \frac{1}{K_0\left(x\right)} \tag{7.2}$$

definiert.

Es kann vorkommen, dass Messungsparameter ungenau oder schlichtweg nicht vorhanden sind. Mittels

$$b := \frac{1000 \cdot L}{E} \cdot \frac{273.15}{T + 273.15} \cdot \frac{0.1 \cdot p}{101.325}$$

lässt sich die K_0-Formel jedoch zu $b \cdot 1/\left(x - g/2000\right)$ zusammenfassen. Mit $a := 1/b$ ist dann die entsprechende Normalisierung als

$$Norm_{Ref}\left(x\right) = a \cdot \left(x - \frac{g}{2000}\right) \tag{7.3}$$

definiert. Um a zu ermitteln genügt ein einziger Referenzpeak mit einer Maximumdriftzeit x_0 und einem normierten Wert $\tilde{x}_0$, wobei $\tilde{x}_0$ ein standardisierter $1/K_0$-Wert sein sollte. Dann kann a mittels

$$a = \frac{\tilde{x}_0}{x_0 - \frac{g}{2000}} \tag{7.4}$$

bestimmt werden. Bis auf die Gitteröffnungszeit sind keine Messungsparameter notwendig. Entstand eine Messung unter Verwendung einer radioaktiven Ionisationsquelle, so lässt sich der RIP als Referenzpeak verwenden. In diesem Fall kann $\tilde{x}_0 := 1/2.06 \approx 0.4854$ gesetzt werden. Das RIP x_0 lässt sich am Besten in den ersten Spektren einer Messung ermitteln, da hier in der Regel noch keine Analytpeaks auftreten, die das RIP-Maximum unter ihr eigenes Maximum bringen könnten.

Eine Normalisierung von Retentionszeiten kann beispielweise mittels der Gleichung 2.6 auf Seite 8 durchgeführt werden. In dieser Arbeit wurde die Retentionszeit jedoch nicht normalisiert. Stattdessen wurde mit Daten gearbeitet, die möglichst gleiche Vortrennungsbedingungen erfüllen.

Die Driftzeitnormalisierung am RIP kann unter Umständen auch zur Normalisierung der Peakhöhe h dienen. Unter der Annahme, dass nur lineare Skalierungen vorkommen, kann eine Normalisierung

$$Norm_{RefH}\left(h\right) = \frac{\tilde{h}_r}{h_r} \cdot h \tag{7.5}$$

erfolgen. Herbei bezeichnen $\tilde{h}_r$ einen Referenzwert wie beispielsweise 1.3 und h_r die ermittelte Höhe des RIPs in einem Spektrum ohne Analytpeaks. Normalerweise tritt in einem solchen

Spektrum auch ein Feuchtepeak auf. Dann sollte seine Höhe stets im selben Verhältnis zu der des RIPs steht. Leider zeigte sich jedoch, dass dies nicht der Fall ist. Dennoch kann möglicherweise durch diese Normalisierung eine Verbesserung bezüglich der Vergleichbarkeit von Messungen, die unter Einsatz verschiedener Verstärker[1] entstanden, erzielt werden.

Es sei F die Menge aller zweidimensionalen Peakfunktionen. Dann beschreiben allgemein Abbildungen $N_x : F \to \mathbb{R}$, $N_y : F \to \mathbb{R}$ und $N_h : F \to \mathbb{R}$ die Driftzeit-, Retentionszeit- und Höhennormalisierungen. Werden Retentionszeiten oder Peakhöhen nicht normalisiert, so können die Definitionen $N_y : f(x, y; x_0, y_0, h, \ldots) \mapsto y_0$ und $N_y : f(x, y; x_0, y_0, h, \ldots) \mapsto h$ vorgenommen werden.

7.2. Peakverläufe

Mit Hilfe von Normalsierungen N_x und N_y können zusammengehörige Peaks zwischen zwei Messungen verglichen werden. Zwei Peaks, durch zweidimensionale Peakfunktionen f_1 und f_2 definiert, aus unterschiedlichen Messungen gelten als zusammengehörig, wenn sie durch den gleichen Ionentyp erzeugt wurden. Beide Peaks können durch ein gemeinsames Tupel (x, y) von *Referenzkoordinaten* identifiziert werden. In der Regel werden dabei kleine Abweichungen zwischen $N_x(f_1)$ und $N_x(f_2)$ sowie $N_y(f_1)$ und $N_y(f_2)$ auftreten, so dass die Referenzkoordinaten gemittelte Werte darstellen. Dann erfolgt die Zuordnung von Peaks zu Referenzkoordinaten mittels eines Toleranzbereichs.

Werden alle Messungen sequentiell durchlaufen, so können zu jedem Referenzkoordinatenpaar Peaks und Konzentrationsangaben der entsprechenden Messung in Form von Listen erzeugt werden. Monomere und Reaktionsionen werden dabei in der Regel bei jeder Konentration vorhanden sein, während Dimere erst ab einer Mindestkonzentration auftreten. Ein Referenzkoordinatentupel und seine zugehörige Liste werden im Folgenden als Peakverlauf bezeichnet:

Definition 7.1 (Peakverlauf) *Ein Peakverlauf V ist ein Tupel $(L, (x, y))$. Dabei bezeichnet (x, y) das Referenzkoordinatenpaar und $L = \{(c_1, f_1), \ldots, (c_n, f_n)\}$ eine Verlaufsliste von zweidimensionalen Peakfunktionen f_i und zugehörigen Konzentrationen c_i, wobei $n \geq 1$. Zusätzlich darf das Hilfssymbol θ an Stelle von f_i auftreten. Ein Tupel (θ, c_i) bedeutet dabei, dass kein Peak der Messung zur Konzentration c_i den Referenzkoordinaten zugeordnet werden konnte.*

Angenommen eine Messung enthält die zweidimensionalen Peakfunktionen $f_1, \ldots, f_m$. Dann muss zu einem Referenzkoordinatenpaar $(\tilde{x}, \tilde{y})$ entweder ein oder kein Peak ausgewählt werden. Dazu wird das Referenzkoordinatenpaar mit einer unscharfen Menge assoziiert und deren Zugehörigkeitsfunktion mit Hilfe von zwei Toleranzparametern $\lambda_x, \lambda_y > 0$ als

$$z(x, y) := \max\left(1 - \frac{|\tilde{x} - x|}{\lambda_x}, 0\right) \cdot \max\left(1 - \frac{|\tilde{y} - y|}{\lambda_y}, 0\right) \tag{7.6}$$

definiert. Es wird der Peak zugeordnet, für dessen zweidimensionale Peakfunktion f die beiden Bedingungen

1. $z(N_x(f), N_y(f)) > 0$ (f liegt im starken z-Schnitt).
2. $z(N_x(f), N_y(f))$ ist minimal.

[1] Beispielsweise können zwei Verstärker zur gleichen Stromstärke signifikant unterschiedliche Spannungen liefern.

gelten. Trifft dies auf mehrere Peaks zu, so wird der erste gefundene Peak gewählt. Sind die Bedingungen für keinen Peak erfüllt, so wird auch kein Peak zugeordnet. Durch die Wahl von λ_x und λ_y kann eine gewichtete Abweichungswertung erzielt werden. Dies ist sinnvoll, da eine Retentionszeitabweichung von einer Sekunde weniger drastisch als eine Driftzeitabweichung von einer halben Millisekunde ist. Allgemein lässt sich für eine Liste L von zweidimensionalen Peakfunktionen und ein Referenzkoordinatenpaar $(\tilde{x}, \tilde{y})$ ein *Matching-Operator*

$$M(L, \tilde{x}, \tilde{y}) := \begin{cases} f & \text{falls } f \in L \text{ die Bedingungen bezüglich } \tilde{x} \text{ und } \tilde{y} \text{ erfüllt} \\ \theta & \text{sonst} \end{cases}$$

definieren.

Algorithmus 7.2.1 Erzeugung von Peakverläufen

1: Gegeben seien Listen $F_1, \ldots, F_n$ zweidimensionaler Peakfunktionen, die zugehörigen Konzentrationen $c_1, \ldots, c_n$ und ein Matching-Operator M.
2: Erzeuge für jedes $f \in F_1$ einen Peakverlauf $(\{(f, c_1)\}, (N_x(f), N_y(f)))$.
3: **for** $i = 2; i \leq n; i++$ **do**
4: Es sei $V = \{V_1, \ldots, V_m\}$ die Liste aller bisher erzeugten Peakverläufe.
5: **for** $j = 1; j \leq m; j++$ **do**
6: Es seien x, y die Referenzkoordinaten des Peakverlaufs V_j.
7: Berechne $f = M(F_i, x, y)$.
8: **if** $f \neq \theta$ **then**
9: Füge (f, c_i) zur Liste von V_j hinzu.
10: Setze x, y als Mittelwerte der $N_x(f_a), N_y(f_a)$ aller in V_j vorhandenen Peakfunktionen f_a.
11: **else**
12: Füge (θ, c_i) zur Liste von V_j hinzu.
13: **end if**
14: **end for**
15: **for** $\forall f \in F_i : f$ wurde keinem Peakverlauf zugeordnet **do**
16: Erzeuge einen Peakverlauf $(\{(f, c_i)\}, (N_x(f), N_y(f)))$.
17: **end for**
18: **end for**
19: **return** V.

Referenzkoordinaten sind zu Beginn der Analyse einer Messungsreihe nicht unbedingt bekannt. Der Algorithmus 7.2.1 erzeugt sowohl Referenzkoordinaten als auch komplette Peakverläufe indem nicht zugeordnete Peaks zu neuen Peakverläufen führen. Dabei werden auch Peakverläufe berücksichtigt, die sich nur über eine einzige Messung erstrecken. Entsprechende Peaks sind in der Regel auf Fehler bei der Peakerkennung zurückzuführen. Mit Angabe eines Schwellwerts können diese aussortiert werden. Nicht relevante Peakverläufe (wie der des RIP) können an Hand von Kriterien wie Peakhöhe und Kettenlänge identifiziert werden. Um eine Fehlinterpretation auszuschließen wurde jedoch stattdessen in der letztendlichen Implementierung die Festlegung eines Fensters im normalisierten Driftzeit-Retentionszeit-Raum vorgezogen. Dann werden nur Referenzkoordinatenpaare innerhalb des Fensters berücksichtigt. In der Regel kann das Fenster einfach so gewählt werden, dass nur Peaks mit höheren Driftzeiten als die des RIPs betrachtet werden. Weitere Kenntnisse über auftretende Peaks sind nicht notwendig.

Ist dagegen eine Folge $(x_1, y_1), \ldots, (x_m, y_m)$ von Referenzkoordinatenpaaren bekannt, so genügt es, für jedes Tupel einen Peakverlauf anzulegen und entsprechend der Messungen zu

füllen. Diese Methode ist als der Algorithmus 7.2.2 formuliert. Der Vorteil liegt darin, dass nur sehr lokal auf Peaks getestet wird. Der Nachteil ist natürlich, dass Peaks im Vorfeld manuell interpretiert werden müssen, wobei hierbei eine Unterstützung durch die erste Methode erfolgen kann.

Algorithmus 7.2.2 Erzeugung von Peakverläufen mit Vorgabe

1: Gegeben seien Listen $F_1, \ldots, F_n$ zweidimensionaler Peakfunktionen, die zugehörigen Konzentrationen $c_1, \ldots, c_n$, Matching-Operator M und eine Folge $(x_1, y_1), \ldots, (x_m, y_m)$ von Referenzkoordinatenpaaren.
2: **for** $k = 1; k \leq m; k++$ **do**
3: Berechne $f = M(F_1, x_k, y_k)$.
4: Erzeuge Peakverlauf $V_k = (\{(f, c_1)\}, (x_k, y_k))$.
5: **if** $f \neq \theta$ **then**
6: Eliminiere f aus F_1.
7: **end if**
8: **end for**
9: **for** $i = 2; i \leq n; i++$ **do**
10: **for** $k = 1; k \leq m$ **do**
11: Berechne $f = M(F_i, x_k, y_k)$.
12: Füge zu Verlaufsliste von V_k f hinzu.
13: **if** $f \neq \theta$ **then**
14: Eliminiere f aus F_i.
15: **end if**
16: **end for**
17: **end for**
18: **return** $V = \{V_1, \ldots, V_m\}$.

7.3. Volumenberechnung

Um das Volumen einer zweidimensionalen Peakfunktion zu berechnen sei zunächst die Fläche eines Peaks in einem einzelnen Spektrum mit Peakhöhe h betrachtet. Sie berechnet sich durch die Integration der entsprechenden (eindimensionalen) Peakfunktion. Im Fall von f_{gbw} kann sie analytisch bestimmt werden, denn f_{gbw} wurde aus zwei Dichtefunktionen, die also jeweils einen normierten Flächeninhalt von Eins haben, konstruiert (vgl. Abschnitt 6.2.1). Für g berechnet sich demnach der Flächeninhalt A_g durch

$$\frac{A_g}{\sqrt{2\pi\left(\frac{w}{2\cdot\log(2)}\right)^2}} = h \Leftrightarrow A_g = h \cdot w \cdot \sqrt{\frac{\pi}{\log(2)}},$$

und für bw berechnet sich der Flächeninhalt A_{bw} durch

$$A_{bw} \cdot \frac{1}{2\pi} = \frac{h \cdot (2 \cdot w)}{4} \Leftrightarrow A_{bw} = h \cdot \pi \cdot w.$$

Da jede der beiden Funktionen eine Peakhälfte beschreibt und beide Funktionen symmetrisch sind lässt sich dann der Flächeninhalt einer Peakfunktion f_{gbw} durch

$$A_{gbw} = \frac{A_g + A_{bw}}{2} \tag{7.7}$$

berechnen.

Allerdings kann eine beliebige Peakfunktion f analytisch schwer handhabbar sein. Aus diesem Grund wurde eine numerische Approximation betrachtet. Mit einer Schrittlänge $s_x > 0$ kann diese mittels des Algorithmus 7.3.1 berechnet werden. Als Abbruchkriterium kann $x \notin [x_{min}, x_{max}]$ mit zwei Parametern $x_{min} < x_0$ und $x_{max} > x_0$ oder $f(x) \leq \lambda$ mit einem Parameter $\lambda > 0$ dienenen.

Algorithmus 7.3.1 Approximation einer Peakfunktionsfläche

1: Gegeben sei eine Peakfunktion f mit den Parametern $x_0, h, \ldots$ und eine Schrittlänge s_x.
2: Initialisiere Variable $A = h$ für Flächeninhalt.
3: Initialisiere Varibale $x = x_0 - s_x$;
4: **while** Abbruchbedingung nicht erfüllt **do**
5: Setze $A \to A + f(x)$.
6: Setze $x \to x - s_x$.
7: **end while**
8: Setze $x = x_0 + s_x$.
9: **while** Abbruchbedingung nicht erfüllt **do**
10: Setze $A \to A + f(x)$.
11: Setze $x \to x + s_x$.
12: **end while**
13: Setze $A \to A \cdot s_x$.
14: **return** A.

Dieses Verfahren lässt sich mit Angabe einer zweiten Schrittlänge $s_y > 0$ leicht zur Berechnung des Volumens einer zweidimensionalen Peakfunktion f erweitern. Das Resultat ist der Algorithmus 7.3.2. Dabei ergibt sich ein zusätzliches Abbruchkriterium $y \notin [y_{min}, y_{max}]$ mit zwei Parametern $y_{min} < y_0$ und $y_{max} > y_0$.

Algorithmus 7.3.2 Approximation eines Peakvolumens

1: Gegeben sei eine zweidimensionale Peakfunktion f mit den Parametern $x_0, y_0, h, \langle f_x \rangle, f_y$ und Schrittlängen s_x, s_y.
2: Rufe Algorithmus 7.3.1 für $f_x(\ldots; x_0, h, \ldots)$ und s_x auf (Ergebnis: A).
3: Initialisiere Variable $V = A$ für Volumen.
4: Initialisiere Varibale $y = y_0 - s_y$;
5: **while** Abbruchbedingung nicht erfüllt **do**
6: Rufe Algorithmus 7.3.1 für $f_x(\ldots; x_0, f_y(y), \ldots)$ und s_x auf (Ergebnis: A).
7: Setze $V \to V + A$.
8: Setze $y \to y - s_y$.
9: **end while**
10: Setze $y = y_0 + s_y$.
11: **while** Abbruchbedingung nicht erfüllt **do**
12: Rufe Algorithmus 7.3.1 für $f_x(\ldots; x_0, f_y(y), \ldots)$ und s_x auf (Ergebnis: A).
13: Setze $V \to V + A$.
14: Setze $y \to y - s_y$.
15: **end while**
16: Setze $V \to V \cdot s_y$.
17: **return** V.

Falls eine Normalisierung der Peakhöhe h einer zweidimensionalen Peakfunktion f vorgenommen wird, ist im Algorithmus 7.3.2 eine modifizierte Höhenfunktion zu verwenden.

Handelt es sich um eine (eindimensionale) Peakfunktion, so kann h einfach durch $N_h(f)$ ersetzt werden. Für eine beliebige Höhenfunktion g, wie beispielsweise die im Abschnitt 6.4 eingeführte Interpolation zwischen Peakmaxima f_{In}, kann eine Höhenfunktion

$$g_n(x) := \frac{N_h(f)}{h} \cdot g(x)$$

definiert werden.

7.4. Konzentrationsabhängigkeitsanalyse

Nachdem für eine Messungsreihe Peakverläufe ermittelt (vgl. Abschnitt 7.2) und die zugehörigen Volumnia approximiert (vgl. Abschnitt 7.3) wurden, liegen zu m Referenzkoordinatenpaaren *Volumenverläufe* $V_1, \ldots, V_m$ vor. Tritt in einem Peakverlauf das Symbol θ auf, so ist dieses als ein Volumen von Null zu interpretieren. Jeder Volumenverlauf V_i besteht aus Tupeln $(c_1, v_1), \ldots, (c_{n_i}, v_{n_i})$, wobei jedes Tupel eine Konzentration c_j (in μg/L) und ein Volumen v_j (in ms·s·V) beinhaltet.

Bei einem einzigen Referenzkoordinatenpaar beschreibt die *Konzentrationsabhängigkeit* eine Beziehung zwischen Konzentration und Volumina. Allerdings tritt bei vielen Substanzen ab einer gewissen Konzenzentration neben dem Monomer zusätzlich ein Dimer auf. Es sind also zwei Volumenverläufe relevant. Theoretisch können sogar noch weitere Peaks auftreten. Es gibt grundsätzlich zwei Möglichkeiten mit mehreren Volumenverläufen umzugehen:

1. Die Volumenverläufe werden unabhängig von einander betrachtet.

2. Die Volumenverläufe werden kombiniert betrachtet.

Natürlich ist eine funktionale Beschreibung dieser Abhängigkeit wünschenswert. Im ersten Fall resultiert dann für jeden Volumenverlauf eine eigene funktionale Beschreibung. Die Konzentrationsabhängigkeit wird folglich durch eine Menge von Funktionen und die zugehörigen Referenzkoordinatenpaaren beschrieben. In dieser Arbeit wurde hauptsächlich dieser Ansatz verfolgt.

Im zweiten Fall beschreibt neben den Referenzkoordinatenpaaren nur eine einzige Funktion die Konzentrationsabhängigkeit. Liegt lediglich ein Volumenverlauf vor, so sind beide Fälle identisch. Sind jedoch mehrere Volumenverläufe relevant, so muss eine Möglichkeit gefunden werden, diese zu kombinieren. Im Abschnitt 2.2 wurde die Bildung von Dimeren aufgeführt. Es werden je zwei Analyte zusammengefasst. Demnach sind Monomer- und Dimervolumen nicht gleichberechtigt zu betrachten. Angenommen, die Volumenverläufe $V_1, \ldots, V_n$ sind in der Reihenfolge Monomer, Dimer, Trimer, ... geordnet und es treten Konzentrationen $c_1, \ldots, c_m$ auf. $V_i(c)$ liefert das Volumen zur Konzentration c im Volumenverlauf V_i. Beinhaltet V_i c nicht, so wird $V_i(c) = 0$ definiert. Um zur Konzentration c ein kombiniertes Volumen V_c zu erhalten, wurde hier motiviert durch die Dimerbildung eine gewichtete Berechnung

$$V_c = \sum_{i=1}^{n} i \cdot V_i(c) \tag{7.8}$$

verwendet. Wird versucht beschädigte Peaks zu rekonstruieren, ergibt sich allerdings das Problem, dass Ionen „doppelt gezählt“ werden. Angenommen, es werden ein Monomer- und ein Dimerpeak betrachtet. Dann muss die Berechnung mittels

$$V_c = V_1 - 2 \cdot V_2 + 2 \cdot V_2 = V_1$$

erfolgen, denn die Rekonstruktion weist die Moleküle des Dimers wieder dem Monomer zu. Es wird also nur das Monomer berücksichtigt. Aus diesem Grund macht eine kombinierte Betrachtung nur für die interpolierende Höhenfunktion f_{In} (ohne Rekonstruktion) Sinn. Es folgt außerdem eine wichtige Erkenntnis: Bei einer fehlerfreien Rekonstruktion von beschädigten Peaks genügt es, das Monomer zu betrachten.

In beiden Fällen ergibt sich das Problem für eine Menge $\mathbf{x} \in {\mathbb{R}_{\geq 0}}^n$ von Konzentrationen und eine Menge $\mathbf{y} \in {\mathbb{R}_{\geq 0}}^n$ von Volumen eine funktionale Beschreibung der Form $f : {\mathbb{R}_{\geq 0}}^n \rightarrow {\mathbb{R}_{\geq 0}}^n$ zu ermitteln. Im ersten Fall ist dies für jeden Volumenverlauf zu erledigen.

Wie ist f zu definieren? Ein trivialer Ansatz besteht in der Interpolation zwischen den Datenpunkten (x_i, y_i). Um auch Konzentrationen behandeln zu können, die nicht Teil der Datenpunkmenge sind, kann eine Extrapolation erfolgen. Allerdings existieren diverse Fehlerquellen:

- Die Glättung kann Peaksignale verändern oder im Extremfall eliminieren.
- Die Glättung kann im Einzelfall Störsignale ausprägen anstatt sie zu reduzieren.
- Verwendete ein- und zweidimensionale Peakfunktionen können schlechte Approximationen der Peaksignale darstellen.
- Die Kettenbildung kann Peaks vernachlässigen oder auf Grund zu großer Toleranzeinstellungen zu falschen Peaks führen.
- Die betrachteten Daten können inkonsistent sein: Zu einer höheren Konzentration tritt ein geringeres Volumen als zu einer geringeren Konzentration auf.
- Die Konzentrationsangaben zu einer Messungsreihe können fehlerhaft sein.

Eine Interpolation ist nicht in der Lage, mögliche Fehler zu reduzieren. Außerdem liegt es nahe, die Abhängigkeit zwischen Konzentration und Volumen durch eine konkrete Modellfunktion zu beschreiben. Ist diese günstig gewählt, so können unter Umständen Fehler korrigiert werden und die Abweichungen zwischen den Funktionswerten und den y_i einen Hinweis auf die Fehlerbelastung der Datenpunkte liefern. Allgemein lässt sich eine reellwertige *Konzentrationsabhängigkeitsfunktion* definieren:

Definition 7.2 (Konzentrationsabhängigkeitsfunktion) *Eine Funktion*

$$f : \mathbb{R} \rightarrow \mathbb{R}$$

ist eine Konzentrationsabhängigkeitsfunktion, falls sie die folgenden Bedingungen erfüllt:

1. $\forall x \leq 0 : f(x) = 0.$
2. $\forall x > 0 : f(x) \geq 0.$
3. $\exists x > 0 : f(x) > 0.$
4. $f(x) > 0 \Rightarrow f(x) < f(x + a)$, *mit* $a > 0$.
5. f *verfügt über Parameter* $p_1, \ldots, p_n$ *mit* $n \geq 1$.

Die Modellfunktion von f mit freien Parametern wird als $\langle f \rangle$ notiert und bezeichnet alle Funktionen, die durch unterschiedliche Wahl des Parametervektors entstehen.

Diese Definitionn ist recht frei, doch für einen Großteil reellwertiger Funktionen sind die Bedingungen nicht erfüllt. Sind für eine Funktion g nur die Bedingungen 1 und 2 verletzt, so kann mittels

$$f(x) := \begin{cases} 0 & \text{falls } x \leq 0 \\ \max(0, g(x)) & \text{sonst} \end{cases} \tag{7.9}$$

eine Funktion f definiert werden, die diese Bedingungen auf einfachste Weise erfüllt.

Besteht eine lineare Abhängigkeit, so kann ein Volumenverlauf durch die Konzentrationsabhängigkeitsfunktion

$$f_{lin}(x; p_1, p_2) := \max(p_1 + p_2 \cdot x, 0) \tag{7.10}$$

beschrieben werden, wobei $p_1 \leq 0$ und $p_2 > 0$.

Es zeigte sich jedoch, dass Volumenverläufe eher einen logarithmischen Charakter haben. Eine entsprechende Konzentrationsabhängigkeitsfunktion lässt sich durch

$$f_{log}(x; p_1, p_2, p_3) := \begin{cases} p_2 \cdot (\log(e + p_3 \cdot (x - p_1)) - 1) & \text{falls } x \geq p_1 \\ 0 & \text{sonst} \end{cases} \tag{7.11}$$

definieren, wobei $p_1 \geq 0$ und $p_2, p_3 > 0$. Die Funktion wurde so gewählt, dass p_1 die niedrigste Konzentration angibt, bei der es zur Ionenbildung und Eintritt dieser in den Driftraum kommt. Sie berücksichtigt nur $\log(e + x)$ Terme mit $x \geq 0$. Ab diesem Punkt steigt die Logarithmusfunktion immer weniger an, was mit Beobachtungen von Volumenverläufen übereinstimmte.

Mit einer Konzentrationsabhängigkeitsfunktion f kann ein Fitting der entsprechenden Modellfunktion auf einen Volumenverlauf mittels des LMA erfolgen (vgl. Abschnitt 4.1.3). Der realiserte Algorithmus basiert auf Implementierungen von Janne Holopaine[2] und J. P. Lewis[3]. Entsprechend den beiden Vorlagen wird zur Lösung von Gleichungssystemen das freie JAMA-Package[4] verwendet. Der Algorithmus wurde so abgewandelt, dass unzulässige Parameter wie ein höherer Fehlerwert behandelt werden. Diese Änderung führte jedoch zu schlechterer Konvergenz und wurde wieder verworfen. Eine mögliche Erklärung könnte darin bestehen, dass während des Gradientenabstiegs auch unzulässige Bereiche im Parameterraum durchlaufen werden können (wobei der Suchpfad des LMA anschließend wieder aus diesen heraus führt).

Um den LMA durchzuführen, sind die Ableitungen nach den einzelnen Parametern notwendig. Gradienten können natürlich auch numerisch approximiert werden, doch im Fall der beiden eingeführten Konzentrationsabhängigkeitsfunktionen ist eine analytische Bestimmung möglich. So ergeben sich unter der Voraussetzung, dass nur $x \geq -p_1/p_2$ auftreten, für f_{lin} die partiellen Ableitungen

$$\frac{\partial f_{lin}(x)}{\partial p_1} = 1 \quad \text{und} \quad \frac{\partial f_{lin}(x)}{\partial p_2} = x.$$

Für $x < -p_1/p_2$ liefern beide Ableitungen den Wert Null.

Im Fall von f_{log} liefern $x < p_1$ ebenfalls Null-Gradienten. Für $x \geq p_1$ ergeben sich die partiellen Ableitungen

$$\begin{aligned} \frac{\partial f_{log}(x)}{\partial p_1} &= -\frac{p_2 \cdot p_3}{p_3 \cdot (x - p_1) + e}, \\ \frac{\partial f_{log}(x)}{\partial p_2} &= \log(p_3 \cdot (x - p_1) + e) - 1 \quad \text{und} \\ \frac{\partial f_{log}(x)}{\partial p_3} &= \frac{p_2 \cdot (x - p_1)}{p_3 \cdot (x - p_1) + e}. \end{aligned}$$

[2] http://users.utu.fi/jaolho/java/lma.zip
[3] http://www.idiom.com/∼zilla/Computer/Javanumeric/LM.java
http://www.idiom.com/∼zilla/Computer/Javanumeric/LMfunc.java
[4] http://math.nist.gov/javanumerics/jama/Jama-1.0.2.jar

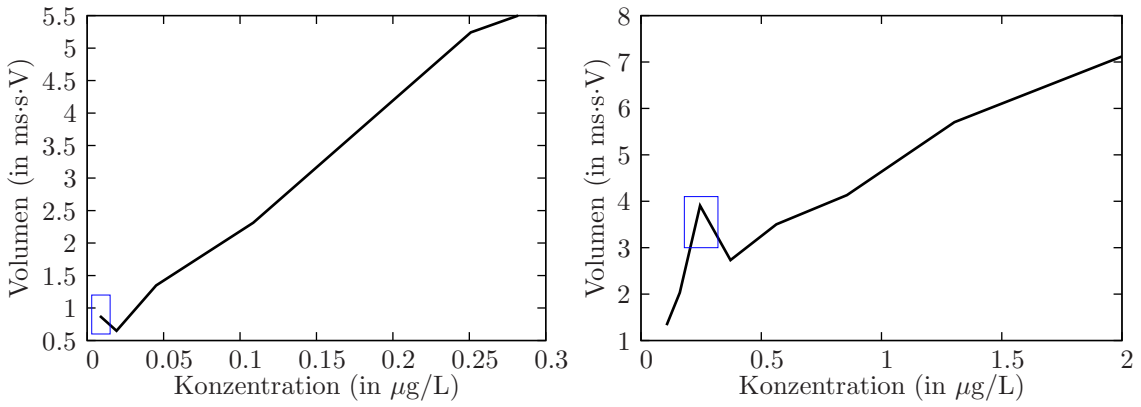

Abbildung 7.1.: Ausreißer in Volumenverläufen

Zusätzlich muss zu Beginn ein „Vorschlag“ in Form eines initialen Parametervektors erfolgen. Oftmals führen Standardparameter zu einem brauchbaren Ergebnis, allerdings kann dies nicht garantiert werden. Zur Not kann über einen kleinen visuellen Editor, der mittels Schiebereglern Parameteränderungen gestattet, ein passenderer Vorschlag geliefert werden. Der gleiche Editor kann auch zum „Nachbessern“ genutzt werden, falls beispielsweise Parameterbedingungen nach dem Fitting verletzt sind.

Erzeugte Volumenverläufe weisen oft „Ausreißer“ auf (s. Abbildung 7.1). Es empfiehlt sich diese vor Durchführung des LMA aus dem entsprechenden Volumenverläufen zu entfernen, denn der LMA versucht auch diesen gerecht zu werden. Eine einfache automatisierte Erkennung von Ausreißern wie in der Abbildung 7.1 links ist durch Verfolgung der Datenpunkte möglich. Bis zur ersten Steigung können alle Punkte aussortiert werden. Allerdings würden auch geringe Fehler zum Aussortieren führen. Eine Abhilfe könnte die Einführung von Toleranzparametern schaffen. Die vorkommenden Volumenverlauflängen waren jedoch so klein, dass es als sinnvoller erachtet wurde, sich auf eine manuelle Auswahlmöglichkeit zu beschränken. Es stellt sich natürlich die Frage, ob solche Ausreißer nur auf Fehler in der Peakerkennung zurückzuführen sind. Der Graph in der Abbildung 7.1 rechts zeigt den Volumenverlauf des 2-Heptanon Monomers in der Messungsreihe MR4. Dieser Ausreißer tritt in der Messung ls101219 mit der Konzentration 0.2458 μg/L auf. Der Peak ist zwar höher als der in Messung ls101218 mit der Konzentration 0.37217 μg/L, hat jedoch eine andere Form. Allerdings ergab eine Aufsummierung der Intensitäten im umschließenden Fenster (20 ms, 10 s), (21.2 ms, 450 s) für ls101219 den Wert 238.481, während sich für ls101218 der Wert 195.186 ergibt. Daten können also tatsächlich inkonsistent sein.

7.5. Konzentrationsbestimmung

Nachdem eine Konzentrationsabhängigkeit ermittelt wurde, ist es wünschenswert, mit ihrer Hilfe die Konzentration der betreffenden Substanz in einer beliebigen Messung „vorherzusagen“. Dafür muss zunächst die Peakerkennungspipeline durchlaufen werden. Danach erfolgt eine Normalisierung der resultierenden Peaks mittels den gleichen N_x und N_y, die bei der Konzentrationsabhängigkeitsanalyse eingesetzt wurden. Es genügt nun wie im Abschnitt 7.2 mit einem Matching-Operator M gezielt nach den durch die Konzentrationsabhängigkeit vorgegebenen Referenzkoordinatenpaaren zu suchen. Für jeden gefundenen Peak wird das Volumen approximiert, wobei das Hilfsymbol θ wieder auf Null abgebildet wird.

Für m Referenzkoordinatenpaare werden so die Volumina $y_1, \ldots, y_m$ erzeugt. Wurden bei der Konzentrationsabhängigkeitsanalyse die Peakverläufe kombiniert, so müssen die Volumen nach der Gleichung 7.8 auf Seite 85 zu einem einzigen y gewichtet summiert werden. Andernfalls werden die y_i direkt ausgewertet, wobei jedem y_i eine entsprechende Konzentrationsabhängigkeitsfunktion f_i zugeordnet ist.

Es sei f eine Konzentrationsabhängigkeitsfunktion und y ein (eventuell gewichtetes) Volumen. Dann besteht die Aufgabe darin, die Konzentration x zu finden, so dass gilt $f(x) = y$. Eine besondere Beachtung benötigt dabei bei unabhängig betrachteten Referenzkoordinatenpaaren der Fall $y = 0$. Ein Dimer wird erst zu einer höheren Konzentration als ein Monomer auftreten. Folglich existieren beliebig viele x. Relevant ist aber nur die höchste Konzentration, für die $f(x) = 0$ gilt. Deshalb sei die Funktion $Inv_f : \mathbb{R} \to \mathbb{R}$ mittels

$$Inv_f(y) := \begin{cases} f^{-1}(y) & \text{falls } y > 0 \\ \max\{x \,|\, f(x) = 0\} & \text{sonst} \end{cases} \tag{7.12}$$

definiert.

Umfasst die Konzentrationsabhängigkeit nur eine einzelne Konzentrationsabhängigkeitsfunktion f, so ist die geschätzte Konzentration durch $Inv_f(y)$ gegeben, wobei y das ermittelte Volumen darstellt. Sind dagegen mehrere Konzentrationsabhängigkeitsfunktionen $f_1, \ldots, f_n$ zu betrachten, so resultieren geschätzte Konzentrationen $x_1 = Inv_{f_1}(y_1), \ldots, x_n = Inv_{f_n}(y_n)$. Aus diesen muss eine Gesamtschätzung x erfolgen.

Mittelwert und Median können nicht verwendet werden. Als Beispiel mögen die drei Volumen Null, Null und Null dienen. In anderen Worten: Es wurde kein passender Peak gefunden. Nun könnte aber gelten: $x_1 = 0, x_2 = 0.3, x_3 = 0.8$. Bei Verwendung von Mittelwert oder Median würde $x > 0$ gelten, während das erwünschte Ergebnis $x = 0$ wäre. Einen Ausweg bietet eine modifizierte Mittelwertberechnung die nur $y_i = 0$ berücksicht, deren $Inv_{f_i}(y_i)$ über dem Mittelwert aller $Inv_{f_j}(y_j)$ mit $y_j > 0$ liegt. Die Berechnung wird im Algorithmus 7.5.1 demonstriert.

Der Vorteil einer unabhängigen Betrachtung Referenzkoordinatenpaaren liegt darin, dass neben der Gesamtschätzung x auch Einzelschätzungen x_i zur Verfügung stehen. Auf der anderen Seite berücksichtigt die modifizierte Mittelwertsberechnung alle Einzelschätzungen in gleicher Weise. Durch Einsatz eines EA lässt sich möglicherweise eine Optimierung erzielen. Jedes Individuum stellt dabei eine Gesamtschätzung x dar. Eine Fehlerfunktion ist durch

$$\chi(x) := \sum_{i=1}^{n} (f_i(x) - y_i)^2$$

gegeben. Gewählt wurde auch in diesem Fall eine $(\mu + \lambda)$-ES, wobei hier ein bestimmter Prozentsatz von Nachkommen durch Rekombination über Mittelwerte erzeugt werden kann.

7.6. Untersuchungsergebnisse

Die folgenden beiden Abschnitte beinhalten Ergebnisse, die im Zusammenhang mit den untersuchten Substanzen 2-Heptanon und 2-Octanon entstanden. Angegebene x Referenzkoordinaten beziehen sich auf die Normalisierung am RIP und entsprechen $1/\mathrm{K}_0$ Werten, angegebene y Referenzkoordinaten stellen gemittelte Retentionszeiten von Peakmaxima dar. Es wurden die Funktionen f_{gbw}, $f_{lbwie2+3}$, f_{In} und f_{lbwe} eingesetzt.

Algorithmus 7.5.1 Modifizierter Mittelwert

1: Gegeben seien die Konzentrationsschätzungen $x_1, \ldots, x_n$ und die zugehörigen Volumina $y_1, \ldots, y_n$.
2: Initialisiere zwei leere Listen A, B.
3: **for** $i = 1; i \leq n; i++$ **do**
4: **if** $y_i > 0$ **then**
5: Setze $A \rightarrow A \cup \{x_i\}$.
6: **else**
7: Setze $B \rightarrow B \cup \{x_i\}$.
8: **end if**
9: **end for**
10: **if** $A = \emptyset$ **then**
11: **return** $x = 0$.
12: **end if**
13: Berechne $\tilde{x}$ als Mittelwert aller $x_i \in A$.
14: **for** $\forall x_i \in B$ **do**
15: **if** $x_i < \tilde{x}$ **then**
16: Setze $A \rightarrow A \cup \{x_i\}$.
17: **end if**
18: **end for**
19: Berechne x als Mittelwert aller $x_i \in A$.
20: **return** x.

7.6.1. 2-Heptanon

Für die Substanz 2-Heptanon standen vier isolierte Messungsreihen zur Verfügung: MR1 - MR4. Neben RIP und Feuchte konnten in jeder dieser Messungsreihen drei Peaktypen ermittelt werden. Ermittelt wurden das Monomer mit $x \approx 0.617$ und $y \approx 25$, das Dimer mit $x \approx 0.792$ und $y \approx 24$, so wie ein dritter Peak mit $x \approx 0.667$ und $y \approx 27$. Letzterer wurde nach Rücksprache mit einer Mitarbeiterin des ISAS - Institute for Analytical Sciences (Luzia Seifert) als nicht dem 2-Heptanon zugehörig klassifiziert und nicht weiter betrachtet[5]. Alle Peaks sind examplarisch in der Abbildung 7.2 dargestellt.

Betrachtet man die Volumenverläufe der vier Messungsreihen, so liegen die entstehenden Kurven nicht übereinander. Es stellt sich natürlich die Frage, ob die Unterschiede auf Fehler während der Peakerkennung oder tatsächlich auf die numerischen Messungsdaten zurückzuführen sind. Aus diesem Grund wurden einfach die Intensitäten in einem das Monomer enthaltenden Fensters summiert. Das Ergebnis ist in der Abbildung 7.3 dargestellt und zeigt, dass auch die direkte Verwendung der Messungsdaten zu unterschiedlichen Kurven führt. Damit ist die wichtigste Annahme dieser Arbeit verletzt: Gleiche Konzentrationen führen keinesfalls zu gleichen Volumina.

Eine Erklärung könnte das hier als *Ionenverlust* bezeichnete Phänomen sein, dessen Untersuchung von einem Mitarbeiter des ISAS - Institute for Analytical Sciences (Dr. Wolfgang Vautz) angeregt wurde. Betrachtet man einen eindimensionalen Peak, so sollte unter der Voraussetzung, dass stets nur ein Gasgemisch gleicher Zusammensetzung in das IMS gelangt, die Fläche dieses Peaks unabhängig von der Driftzeit des Maximums immer gleich sein. Indem die Feldstärke geändert wird, können Peaks auf der Driftzeitachse verschoben werden, weil die auf die Ionen wirkende Beschleunigung im elektrischen Feld des Driftraums verändert wird.

[5] Möglicherweise ist der dritte Peak auf eine Verunreinigung in der 2-Heptanon Probe zurückzuführen. In Gemischen konnte er nicht nachgewiesen werden.

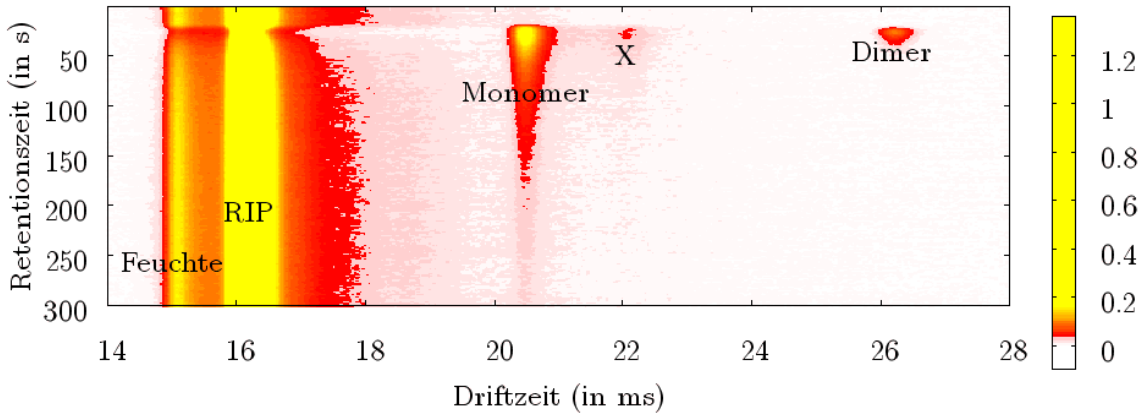

Abbildung 7.2.: Monomer und Dimer der Substanz 2-Heptanon am Beispiel der Messung ls081706, außerdem sind RIP und Feuchte, so wie ein unbekannter weiterer Peak X gekennzeichnet

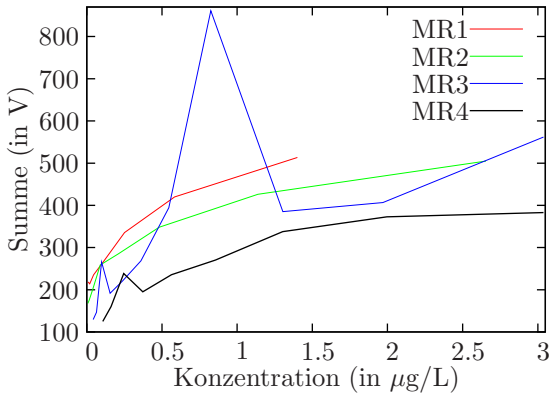

Abbildung 7.3.: Summe der Intensitäten in den Messungsreihen MR1 - MR4 im Fenster des 2-Heptanon Monomers

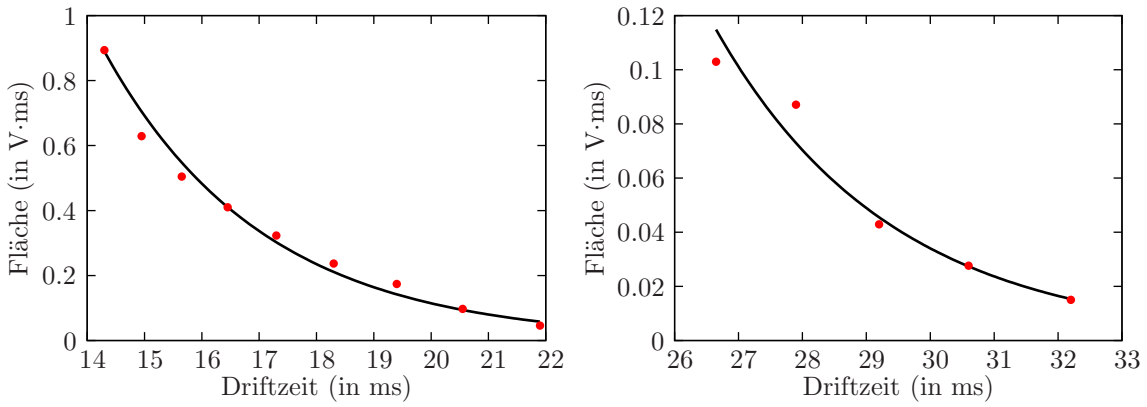

Abbildung 7.4.: Ionenverlust beim RIP (links) und beim 2-Nonanon Monomer bei gleicher Konzentration (rechts)

Es zeigt sich allerdings, dass die Fläche exponentiell bezüglich der Driftzeit des Peakmaximums abnimmt. Dazu wurde ein Peakfitting mit f_{gbw} durchgeführt und die entsprechenden Flächen in Abhängigkeit zu x_0 als Datenpunkte für das im Abschnitt 4.1.1 vorgestellten Fitting einer exponentiellen Modellfunktion aufgefasst. Zwei Resultate sind in der Abbildung 7.4 dargestellt. Wahrscheinlich gehen dabei Ionen nach einer bestimmten Wahrscheinlichkeitsverteilung am Driftraumrand „verloren“.

Eine Idee bestand darin, die Normalisierung von Peakhöhen (vgl. Abschnitt 7.1) durchzuführen. Leider ergab sich keine nennenswerte Verbesserung. In der Abbildung 7.5 wurden die Monomerpeaks durch Verwendung der Peakfunktion f_{gbw} und der interpolierenden Höhenfunktion f_{In} beschrieben. Dies könnte natürlich auch damit erklärt werden, dass die Höhennormalisierung schlichtweg ungeeignet für eine Normalisierung von Volumina ist. Allerdings ergeben sich mittlere Driftzeiten der Monomerpeakmaxima von 20.484 ms für MR1, 20.388 ms für MR2, 20.783 ms für MR3 und 20.727 ms für MR4. Dies entspricht nicht den Erwartungen, wenn man Ionenverlust als Ursache zu Grunde legt. Eine genauere Untersuchung zwischen Volumen und Konzentration bleibt offen. Eigentlich könnte die Untersuchung an dieser Stelle abgebrochen werden, es werden aber dennoch einige Resultate gezeigt, die sich in Zukunft durch neue Kenntnisse vielleicht verbessern lassen.

Man kann sich die Frage stellen, warum der Volumenverlauf zur Messungsreihe MR2 bei der Konzentration 0.4813 μg/L einen Ausreißer aufweist, der in der Abbildung 7.3 nicht vorhanden ist. Dies liegt daran, dass die Extrapolation der Höhenfunktion f_{In} nur schlecht approximiert: Die tatsächliche Kurve fällt immer langsamer ab, doch die lineare Extraploation hat notwendigerweise einen konstanten Gradienten. Dieses Problem könnte durch geeignete Wahl des Schwellwerts bei der Erkennung eindimensionaler Peakfunktionen und einer höheren Toleranz bei der Kettenbildung vermindert werden. Die wahl wäre jedoch abhängig von einer einzelnen Messung, wodurch die Vorteile der in dieser Arbeit angestrebten Automatisierung sehr vermindert würden. Aus diesem Grund wurde für jede verwendete Kombination von Peak- und Höhenfunktion, so wie Substanz stets die gleichen Peakerkennungsparameter benutzt. Einen Ausweg stellt die Verwendung einer rekonstruierenden Höhenfunktion wie beispielsweise f_{lbwe} dar, denn sie bietet eine bessere Extrapolation. Allerdings wird dies bei der in dieser Arbeit realisierten EA-Lösung zum Preis des Nichtdeterminismus erkauft, der unter ungünstigen Umständen zu schlechteren Resultaten als bei der Verwendung von f_{In} führt. Zwei Resultate unter Einsatz dieser Höhenfunktion sind in der Abbildung 7.6 dargestellt.

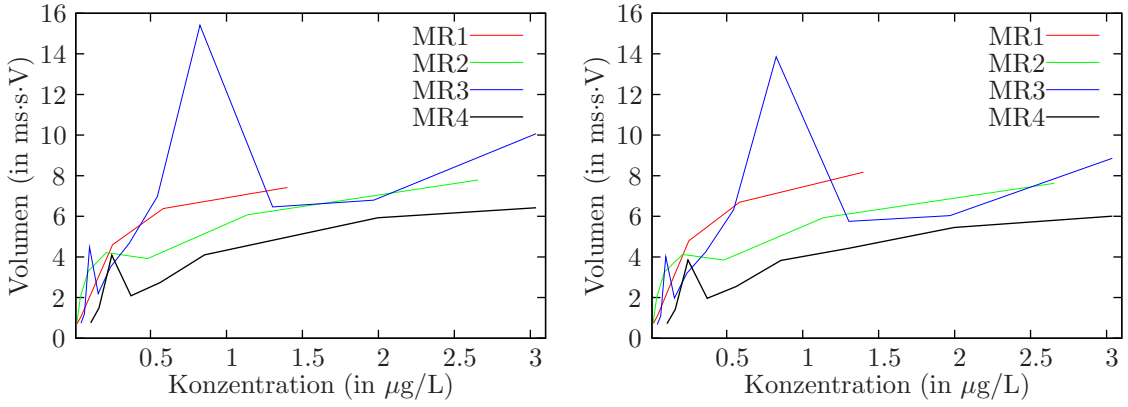

Abbildung 7.5.: Volumenverläufe zum 2-Heptanon Monomer in den Messungsreihen MR1 - MR4 mit Peakfunktion f_{gbw} und Höhenfunktion f_{In}, ohne (links) und mit (rechts) Peakhöhennormalisierung

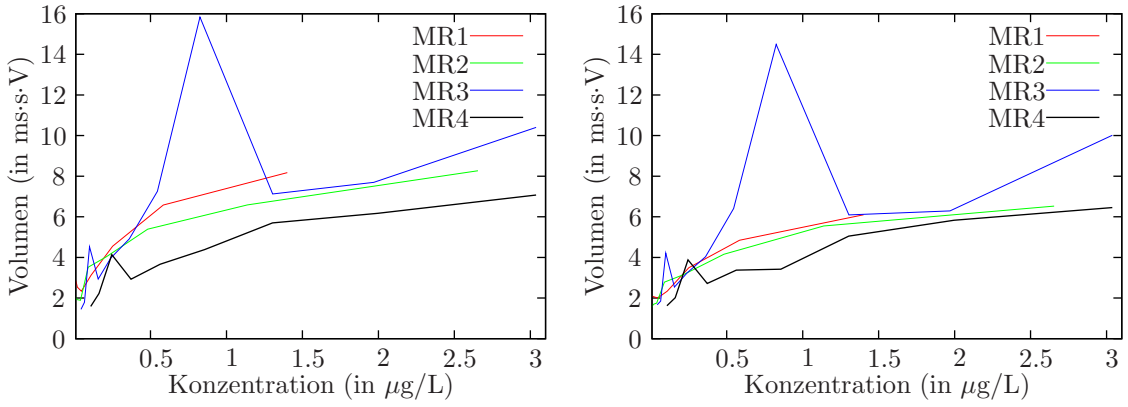

Abbildung 7.6.: Volumenverläufe zum 2-Heptanon Monomer in den Messungsreihen MR1 - MR4 mit Peakfunktion f_{gbw} (links), $f_{lbwie2+3}$ (rechts) und Höhenfunktion f_{lbwe}

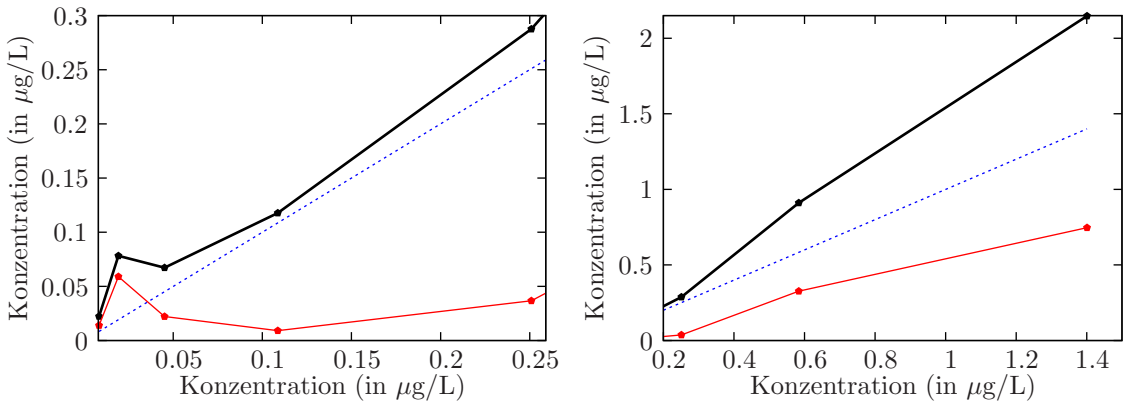

Abbildung 7.7.: Ermittelte Konzentration (schwarz), angegebene Konzentration (blau) und resultierender Fehler (rot) der für die Messungsreihe MR2 gefundene Konzentrationsabhängigkeit mit Peakfunktion f_{gbw} und Höhenfunktion f_{lbwe} in der Messungsreihe MR1

Da unterschiedliche funktionale Peakbeschreibungen zu unterschiedlichen Volumenverläufen führen, macht es nur Sinn, gleiche Peak-/Höhenfunktionskombinationen zu betrachten. Wie bereits erwähnt zeigte es sich, dass die logarithmische Konzentrationsabhängigkeitsfunktion f_{log} in der Regel eine bessere Approximation von Volumenveräufen bietet, als die lineare Konzentrationsabhängigkeitsfunktion f_{lin}. Zunächst wurden dabei offensichtliche Außreißer entfernt. Anschließend wurde der Vorschlag des LMA manuell optimiert, wobei insbesondere die Bedingung $p_1 = 0$ sichergestellt werden musste. In vielen Fällen lieferte der LMA bereits ein gutes Fitting, in Ausnahmefällen versagte er jedoch auch.

Für die Messungsreihe MR2 mit Peakfunktion f_{gbw} und Höhenfunktion f_{lbwe} ergaben sich beispielsweise für das Monomer die Parameter $p_1 = 0$, $p_2 = 1.733$ und $p_3 = 114.171$, für das Dimer die Parameter $p_1 = 0.01$, $p_2 = 1.3713$ und $p_3 = 4.1279$. Die Abbildungen 7.7 und 7.8 zeigen die Konzentrationsbestimmungen bezüglich der Messungsreihen MR1 und MR6. Eine Auflistung ermittelter Konzentrationsabhängigkeiten ist im Anhang B zu finden. Es wurde nicht der Anspruch gestellt, möglichst ähnliche Parametervektoren zu erzeugen. Im Anhang C sind, mit Ausnahme der als zu fehlerbelastet eingestuften Messungsreihe MR3, gemittelte Konzentrationsbestimmungsfehler aufgeführt.

7.6.2. 2-Octanon

Für die Substanz 2-Octanon stand die isolierte Messungsreihe MR10 zur Verfügung. Ermittelt wurden zwei Referenzkoordinatenpaare, das Monomer mit $x \approx 0.652$ und $y \approx 54$, so wie das Dimer mit $x \approx 0.852$ und $y \approx 53$. Sie sind examplarisch in der Abbildung 7.9 dargestellt.

Werden in den Drift-/Retentionszeitfenstern $x \in [21.5, 22.7], y \in [30, 450]$ und $x \in [28, 30], y \in [30, 230]$ die Intensitäten aufsummiert, so ergeben sich für das Monomer und das Dimer, bis auf jeweils einen Ausreißer bei der Messung ls121805, logarithmische Kurven (Abbildung 7.10). Dimerpeaks wurden (wahrscheinlich auf Grund zu geringer Größe) nur unzureichend erkannt, weshalb sich auf das Monomer beschränkt wurde. Als Beispiel seien die mit der Peakfunktion f_{gbw} und der Höhenfunktion f_{lbwe} ermittelte f_{log}-Parameter $p_1 = 0$, $p_2 = 1.9137$ und $p_3 = 20.7594$ betrachtet. Weitere Konzentrationsabhängigkeiten sind im Anhang B zu finden. Die Abbildung 7.12 stellt die Konzentrationsbestimmungen zur Mes-

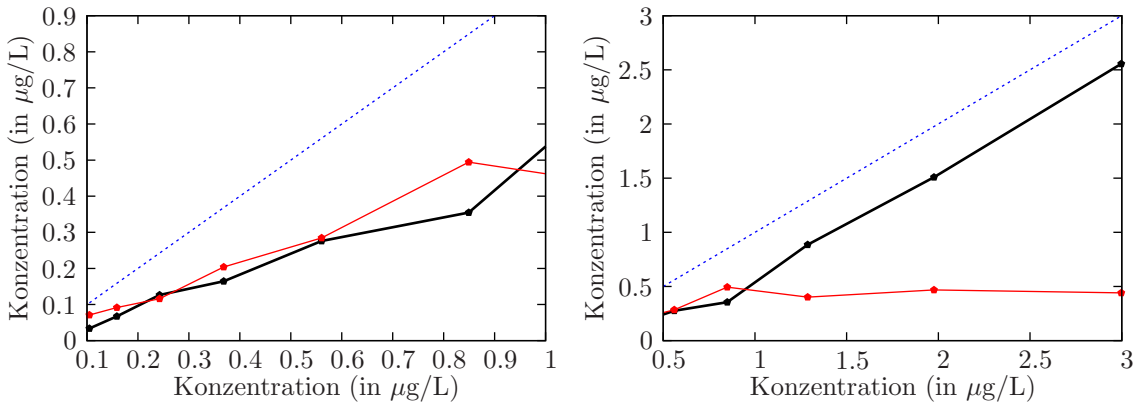

Abbildung 7.8.: Ermittelte Konzentration (schwarz), angegebene Konzentration (blau) und resultierender Fehler (rot) der für die Messungsreihe MR2 gefundene Konzentrationsabhängigkeit mit Peakfunktion f_{gbw} und Höhenfunktion f_{lbwe} in der Messungsreihe MR6

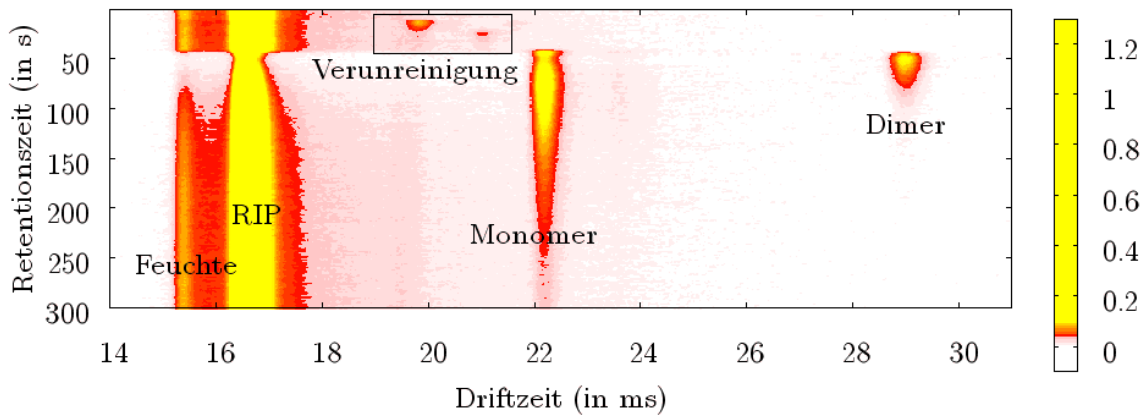

Abbildung 7.9.: Monomer und Dimer der Substanz 2-Octanon am Beispiel der Messung ls121801, außerdem sind RIP, Feuchte und Verunreinigungen gekennzeichnet

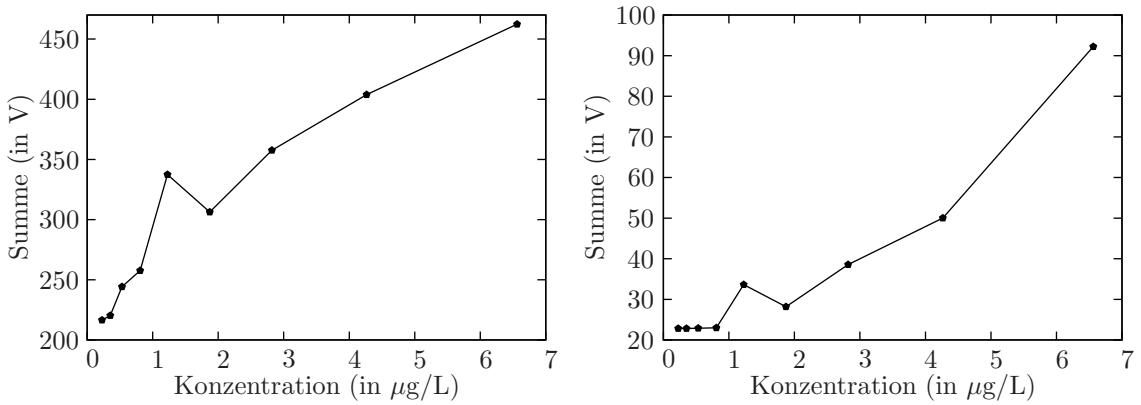

Abbildung 7.10.: Summierte Intensitäten im Bereich des Monomers (links) und des Dimers (rechts) in der Messungsreihe MR10

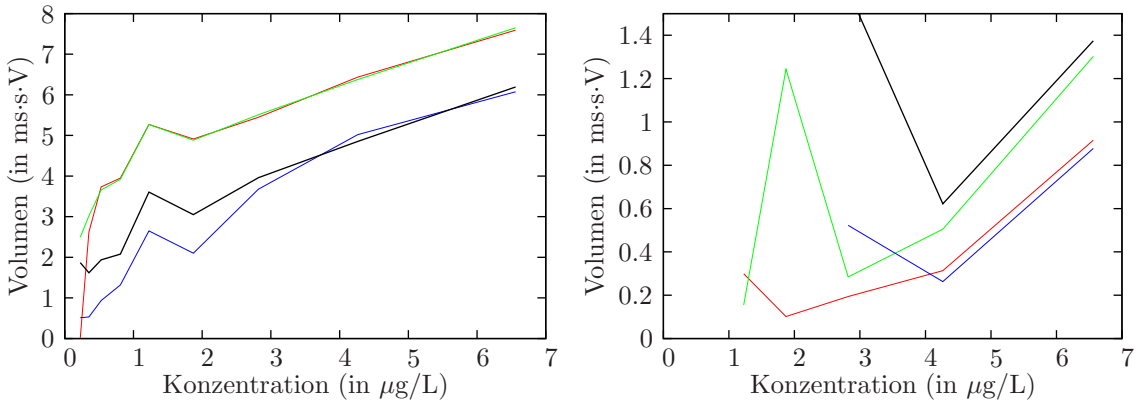

Abbildung 7.11.: Volumenverläufe des Monomers (links) und des Dimers (rechts) zu den Peak-/Höhenfunktionskombintionen f_{gbw}/f_{In} (rot), f_{gbw}/f_{lbwe} (grün), $f_{lbwie2+3}/f_{In}$ (blau) und $f_{lbwie2+3}/f_{lbwe}$ (schwarz)

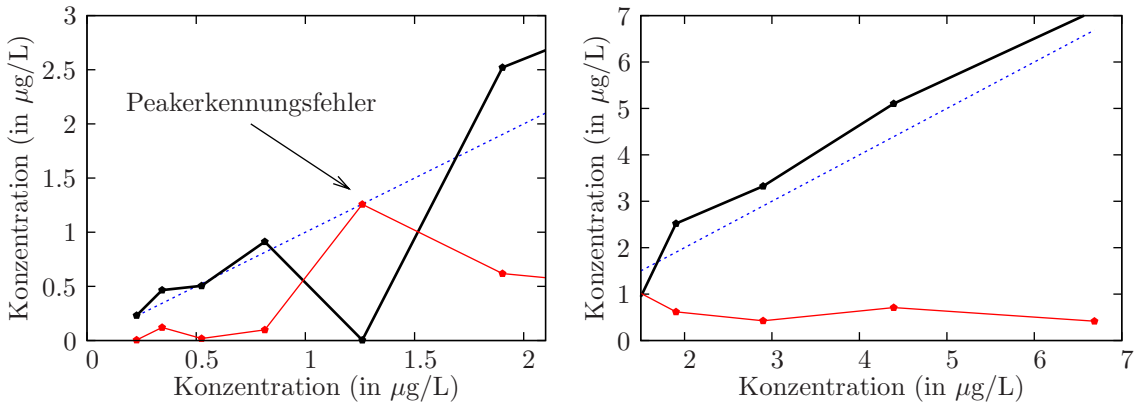

Abbildung 7.12.: Ermittelte Konzentration (schwarz), angegebene Konzentration (blau) und resultierender Fehler (rot) der für die Messungsreihe MR10 gefundene Konzentrationsabhängigkeit mit Peakfunktion f_{gbw} und Höhenfunktion f_{lbwe} in der Messungsreihe MR8

sungsreihe MR8 dar. Hierbei ist auch ein Peakerkennungsfehler markiert. Weitere gemittelte Konzentrationsbestimmungsfehler sind im Anhang C aufgelistet.

8. Implementierung

Die in den Kapiteln 6 und 7 vorgestellten Lösungen wurden in der Programmiersprache Java[1] (JDK 1.5) implementiert. Java bietet vor Allem den Vorteil der Plattformunabhängigkeit. Ein weiterer Vorteil von Java liegt in der relativ einfachen objektorientierten Programmierschnittstelle. Als nützlich erwiesen sich die Dokumentation der Java-API[2] und das Online-Book „Java ist auch eine Insel“[3]. Das Resultat der Implementierung ist das Programm *Codean* (Concentration-Dependence-Analyser). Es entstand eine GUI-Anwendung, wobei im Verlauf der Entwicklung einige als nicht relevant eingestufte Teile, beispielsweise die FT-Glättung, verworfen wurden.

Die einzigen verwendeten Libraries sind JAMA und JDOM[4]. Letzteres dient dem Schreiben und Lesen von XML-Dateien. Mit Ausnahme von Messungen (sie beinhalten zu viele Daten) wurden alle Serialisierungen[5] in Form von XML-Dateien vollzogen. XML-Dateien bieten den Vorteil, dass sie in der Regel mit jedem Texteditor zu öffnen sind und ihr Inhalt oft selbsterklärend ist.

8.1. Dateien

Eine einzelne Messung kapselt im Wesentlichen die Intensitätsmatrix, so wie den Driftzeiten- und den Retentionszeitenvektor. Bei der Überführung der Rohdaten in Binärdateien werden dabei schon bei Bedarf die im Kapitel 5 vorgestellten Transformation Intensitätsinvertierung, Basislinienkorrektur und Faltenausgleich vollzogen. Messungsparameter werden in einer eigenen Klasse wie auch Datei gespeichert. Der Grund liegt darin, dass die beide Datentypen in unterschiedlichen Formaten gepeichert werden.

Messungsreihen werden in einem Projekt organisiert. Für jedes Projekt wird während dessen Erzeugung ein neues Verzeichnis angelegt. Wird eine Messungen hinzugefügt, so werden die entsprechenden Daten und die zugehörigen Messungsparameter in dieses kopiert. Falls nicht vorhanden, wird eine neue Messungsparameterdatei angelegt. Zu jeder Messung wird außerdem eine Konzentrationsangabe gespeichert.

Neben diesen Ausgangsdaten und Konzentrationsabhängigkeiten werden auch Zwischenresultate in Dateien gespeichert. Es existieren verschiedene Beziehungen zwischen Dateien. Diese Beziehungen werden durch den Dateinamen definiert. So sind die Messungsparameter zur Messungsdatei „Messung1.mes“ in der Datei „Messung1.ma“, und die ermittelten zweidimensionalen Peaks in der Datei „Messung1.pks“ gespeichert. Eine Auflistung aller Dateitypen ist in der Tabelle 8.1 gegeben.

[1] http://www.sun.com/java/
[2] http://java.sun.com/j2se/1.5.0/docs/api/
[3] http://www.galileocomputing.de/openbook/javainsel6/ (Es wurde die ältere Version 1.4 benutzt)
[4] http://www.jdom.org/
[5] Serialisierung wird hier als die Möglichkeits des Speicherns/Lesens von/aus Dateien verstanden.

Tabelle 8.1.: Die verschiedenen von Codean verwendeten Dateitypen

Dateiname:	Inhalt:	Javaklasse:
<Messung>.mes	Messung	Measurement
<Messung>.ma	Messungsparameter	MeasurementAttributes
<Messung>.pks	Zweidimensionale Peaks	Peak2dList
<Projekt>.prj	Projektbeschreibung: Substanzinformation, Messungsnamen und zugehörige Konzentrationen, Algorithmenparameter	Project
<Projekt>.ar	Volumenverläufe	AnalyseResult
<...>.cdi	Konzentrationsabhängigkeiten und verwendete Algorithmenparameter	ConcentrationDependence, CombinedConcentrationDependence
<...>.set	Allgemeine Einstellungen, Algorithmenparametervorgaben, Visualisierungseinstellungen	ApplicationSettings

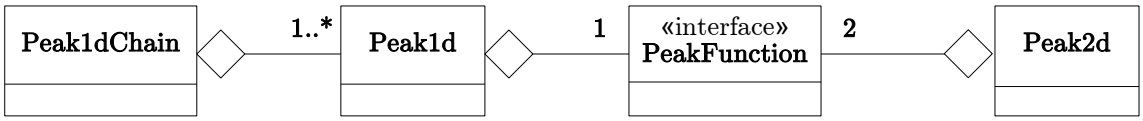

Abbildung 8.1.: Peakklassen

8.2. Peakerkennung

Die im Kapitel 6 vorgestellte Peakerkennungspipeline wurde in der Implementierung leicht abgeändert. So wird die im Abschnitt 7.1 eingeführte Normalisierung bereits zum Abschluß der Pipeline durchgeführt. Eindimensionale Peaks werden durch die Klasse *Peak1d* beschrieben, Ketten durch die Klasse *Peak1dChain* und zweidimensionale Peaks durch die Klasse *Peak2d*. Letztere entspricht einer zweidimensionalen Peakfunktion, wobei zusätzlich die normalisierten Referenzkoordinaten gehalten werden und die Menge aller detektierten Peaks in einer bestimmten Messung zur Serialisierung in der Klasse *Peak2dList* gekapselt wird.

Für beide Peaktypen steht das Interface *PeakFunction* zur Verfügung, das nur die Methode *calculate (double x, double x0, double h)* mit reellwertigem Rückgabewert definiert. Es bleibt natürlich dem Programmierer überlassen, die in der Definition 6.1 geforderten Bedingungen bei der Implementierung zu erfüllen. So wurde die interpolierende Höhenfunktion f_{In} als Realisierung von PeakFunction programmiert. Die Beziehung der Peakklassen ist in der Abbildung 8.1 als UML-Klassendiagramm dargestellt.

Für jeden Abschnitt der Pipeline ist (bis auf eine Ausnahme) eine separate Berechnungseinheit in Form eines Interfaces zuständig. Diese werden im Folgenden vorgestellt:

Smoother Erzeugt für eine Messung eine geglättete Intensitätsmatrix.

Peak1dDetector Erzeugt zu jedem Spektrum eine Liste von eindimensionalen Peaks (Klasse: Peak1d). Außerdem wird definiert, wie aus einer Kette eindimensionaler Peaks eine gemeinsam beschreibende Peakfunktion zu bilden ist.

Merger Fasst eindimensionale Peaks zu Ketten zusammen.

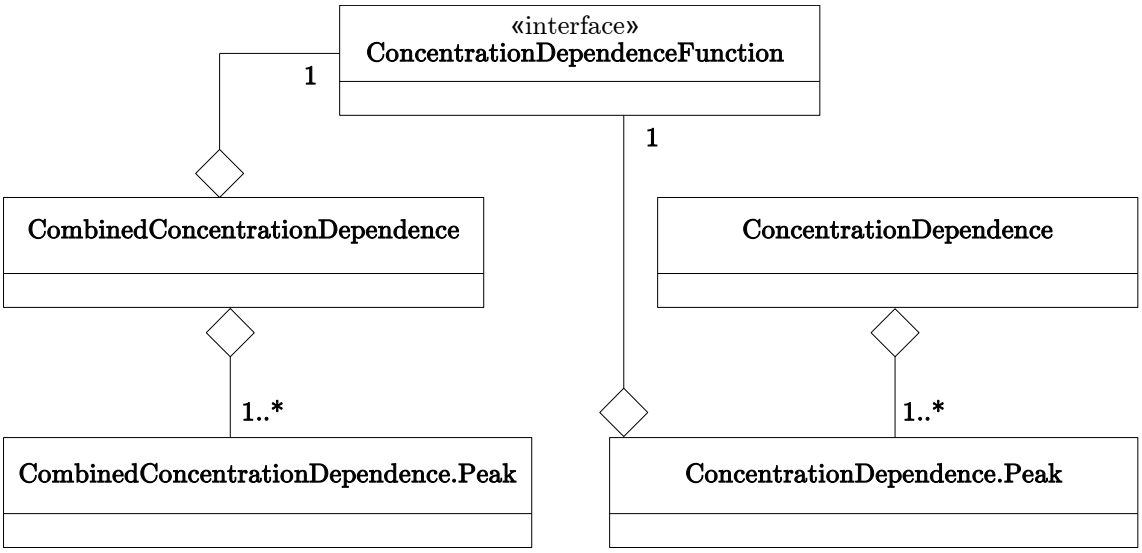

Abbildung 8.2.: Konzentrationsabhängigkeitsklassen

Peak2dDetector Erzeugt für eine Liste von (Retentionszeit, Intensität)-Tupeln und eine durch einen Peak1dDetector erzeugte Peakfunktion einen zweidimensionalen Peak.

Normaliser Berechnet für eine Liste zweidimensionaler Peaks Normalisierungen.

8.3. Konzentrationsabhängigkeitsberechnung

In der Implementierung wurden die Erzeugung von Peakverläufen und Berechnung der entsprechenden Volumenverläufe zusammengefasst. Dazu sind die folgenden beiden Interfaces definiert:

Matcher Liefert den Index des zweidimensionalen Peaks in einer Liste zurück, der am besten zu zwei gegebenen Koordinaten passt. Erfüllt kein Peak die geforderten Kriterien, so wird der Wert -1 zurückgeliefert.

VolumeApproximator Approximation von Peakvolumina.

Die zu ermittelnden Konzentrationsabhängigkeiten werden durch zwei Klassen beschrieben: *ConcentrationDependence* (uanabhängige Betrachtung von Volumenverläufen) und *CombinedConcentrationDependence* (kombinierte Betrachtung von Volumenverläufen). Beide nutzen das Interface *ConcentrationDependenceFunction*, das neben der Funktionswertberechnung auch inverse Funktionswertberechnung und den Parametervektor (für den LMA) liefert. Die Beziehungen dieser Komponenten sind in der Abbildung 8.2 als UML-Klassendiagramm dargestellt.

8.4. PlugIn-Architektur

Es stellte sich die Frage, wie die Berechnungskomponenten zu implementieren sind. Der Benutzer sollte Auswahlmöglichkeiten haben. Außerdem wäre eine schnelle Erweiterung mit neuen Algorithmen wünschenswert.

Die Konsequenz war die Bereitstellung einer PlugIn-Architektur. Alle PulgIns mit Ausnahme von ConcentrationDependenceFunction sind Interfaces, die das Interface *PugIn* erweitern.

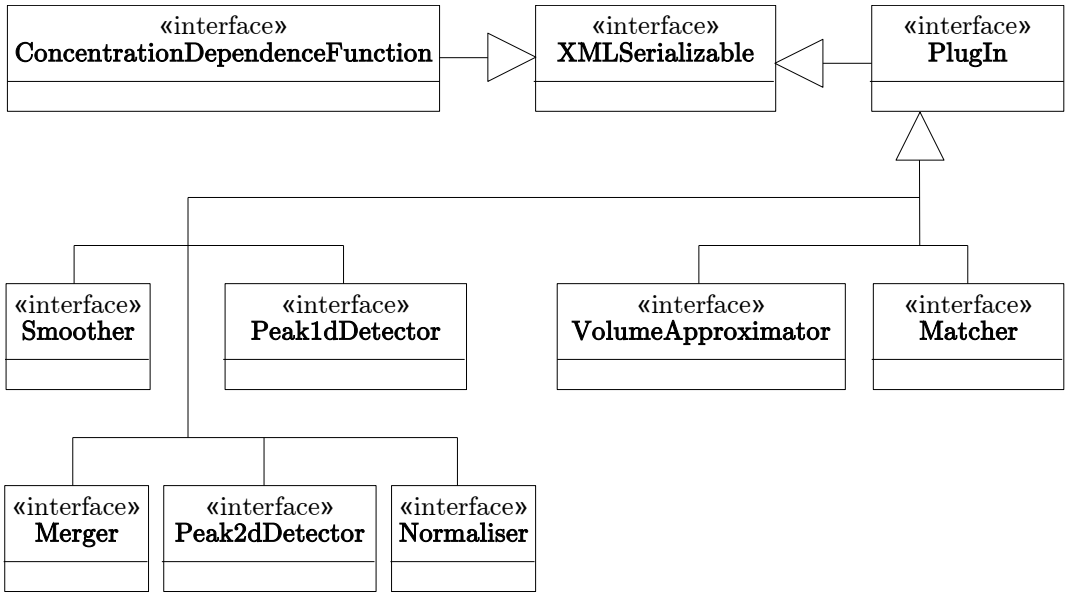

Abbildung 8.3.: PlugIn-Architektur

Das Interface ConcentrationDependenceFunction dient als eigenständiges PlugIn-Interface. Eine Übersicht ist in der Abbildung 8.3 dargestellt. Alle Instanzen von PlugIn werden in der Klasse *DetectionPlugIns* als eine PlugIn-Konfiguration verwaltet. Beispielsweise enthält ein DetectionPlugIns-Objekt $n > 0$ Peak1dDetector-Instanzen und den Index der ausgewählten Instanz.

Das durch PlugIns erweiterte Interface *XMLSerializable* ermöglicht die Serialisierung über XML-Dateien. Dazu muss für jede Implementierung neben einer Methode *settingsToXML(Element element)*, die alle Attribute in ein gegebenes XML-Element schreibt, auch ein Konstruktor *<Klassenname>(Element element)* definiert werden, der aus einem XML-Element ein neues Objekt des PlugIn-Typs erzeugt.

PlugIns sollten dynamisch einzubinden sein. Bei Codean sind dazu verschiedene (relative) Verzeichnisse definiert, die zu bestimmten Zeitpunkten nach class-Dateien durchsucht werden. Innere Klassen werden ignoriert. Jeder PlugIn-Typ hat dabei ein spezifisches Verzeichnis. Um ein neues PlugIn zu installieren genügt es, die class-Datei in das entsprechende Verzeichnis zu kopieren und Codean zu einem Update zu veranlassen:

- Funktionsabhängigkeitsfunktionen werden geladen, sobald ein Dialog zur Erzeugung einer Konzentrationsabhängigkeit geöfffnet wird.
- Sonstige PlugIns werden beim Start von Codean geladen.

Voraussetzung ist, dass ein argumentloser Standardkonstruktor definiert ist. Der folgende Quelltextausschnitt demonstriert das Laden von Smoother-PlugIns beim Start von Codean.

```
ClassLoader classLoader=ClassLoader.getSystemClassLoader();

String rootFile=
                new File(classLoader.getResource
                ("codean/detection/peak/plugin/PlugIn.class")
                        .toURI())
                .getParent()
```

```
                +File.separatorChar;
String rootDomain="codean.detection.peak.plugin.";

FilenameFilter filter=new FilenameFilter() {

    public boolean accept(File dir, String name) {
        return ((name!=null)&&(name.endsWith(".class"))
                &&(!name.contains("$")));
    }

};

String name="smoothers";

File dir=new File(rootFile+name);
File[] files=dir.listFiles(filter);

smoothersIndex=0;
smoothers=new Smoother[files.length];

for (int i=0;i<smoothers.length;i++) {

    smoothers[i]=(Smoother)classLoader.loadClass(
                rootDomain+name+"."
                +FileUtils.cutSuffix(files[i].getName(),".class"))
                .getConstructor(new Class[]{})
                        .newInstance(new Object[]{});
    if (smoothers[i].getClass().getName()
                .equals(WaveletFuzzySmoothing
                .class.getName())) // predefined
        smoothersIndex=i;

}
```

Offwensichtlich werden Projektdateien ungültig, falls ein PlugIn nicht mehr zur Verfügung steht. Das XML-Format erlaubt jedoch eine einfache Korrektur.

8.5. Applikation

Ursprünglich sollte die Konstruktion der Klassenstruktur nach dem MVC-Prinzip erfolgen, dieses Bestreben wurde jedoch wieder verworfen. So verfügt das PlugIn-Interface über eine Methode, die eine GUI-Komponente zur Bearbeitung von Parametern liefert. In der Regel werden Parameter wie auch der mit dem PlugIn verbundene Algorithmus Teil der Klasse sein, so dass ein Durchsetzen der MVC-Trennung eher umständlich erscheint. Als Umgebungssprache wurde Englisch gewählt.

Kern der Applikation ist die Klasse *Application*. Sie verwaltet als Singleton-Instanz einen einzigen *MainFrame*, gegebenenfalls ein Projekt (SDI) und verschiedene globale Parameter (u.a. DetectionPlugIns-Vorgaben).

Die wichtigste GUI-Komponente ist die Projektsicht (s. Abbildung 8.4). Hier lassen sich

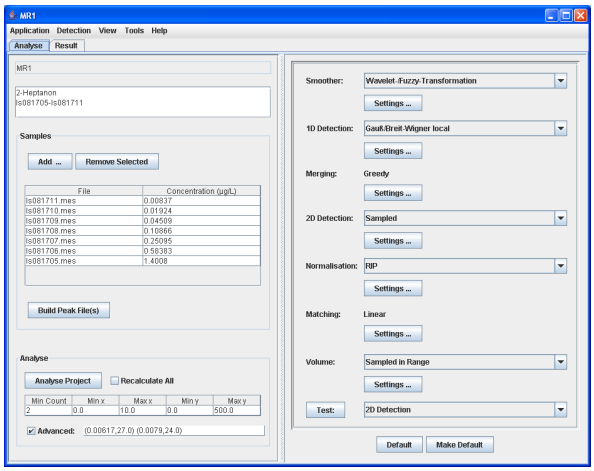

Abbildung 8.4.: Screenshot einer Projektansicht

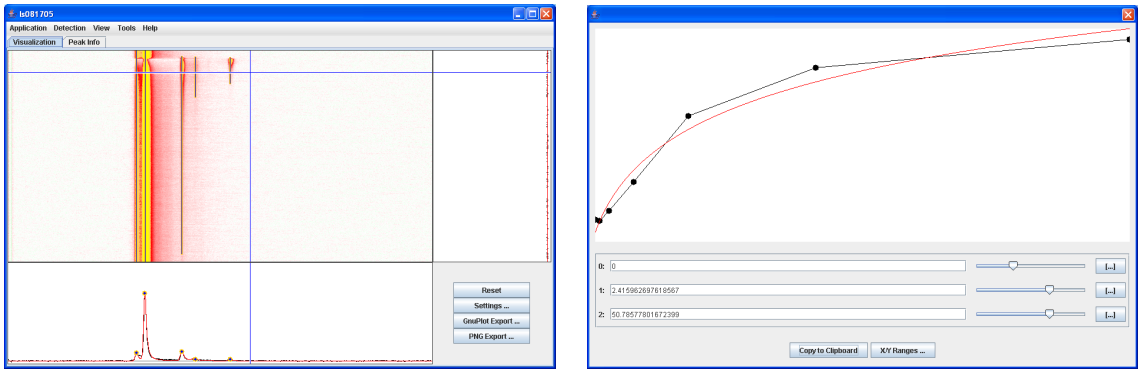

Abbildung 8.5.: Screenshots einer Peakvisualisierung (links) und des Editors zum manuellen Anpassen einer Konzentrationsabhängigkeitsfunktion (rechts)

Messungen zu einem Projekt hinzufügen oder aus diesem entfernen und der Analyseprozess starten. Resultierende Volumenverläufe lassen sich bearbeiten und aus ihnen mittels des LMA und eines grafischen Editors Konzentrationsabhängigkeiten bilden. In erster Linie ist die Benutzeroberfläche dabei für eine unabhängige Betrachtung von Referenzkoordinatenpaaren ausgelegt, dient jedoch auch der kombinierten Betrachtung.

Viele Daten können visualisiert werden: Messungen, Peaks, Volumenverläufe und Konzentrationsabhängigkeitsfunktionen. Die Visualisierung ist recht simpel gehalten (s. Abbildung 8.5), verfügt jedoch über die Möglichkeit des Hineinzoomens und der Rücksetzung. Koordinaten werden als Tooltips unter dem Mauszeiger angezeigt. Für Messungs- und Peakvisualisierungen können zusätzlich die Farbdarstellungen und angezeigte Elemente festgelegt werden. In der Regel können Daten für eine professionellere Visualisierung als GnuPlot[6]-Dateien exportiert werden. GnuPlot ist eine freie Datenvisualisierungssoftware, mit der auch alle in dieser Arbeit enthaltenen Plots (mit Ausnahme von Screenshots) entstanden.

[6] http://www.gnuplot.info

9. Zusammenfassung und Ausblick

Abschließend erfolgt nun eine Zusammenfassung der in dieser Arbeit enstandenen Resultate und deren Diskussion. Erwartungsgemäß machten dabei die im Kapitel 6 behandelten Probleme der Datenglättung und der Peakerkennung einen Großteil des Arbeitsaufwands aus. Die Reihenfolge der im Folgenden zusammengefassten Aspekte ist an die Kapitelreihenfolge angelehnt.

Mit der im Abschnitt 6.1.5 eingeführten waveletbasierten F-Transformation wurde eine universelle Methode der Datenglättung entwickelt, die bei zufälligem Rauschen gute Resultate liefert. Der Preis ist ein hoher Berechnungsaufwand, der in erster Linie auf die Berechnung der F-Transformation und ihrer Inversen zurückzuführen ist. Im Hinblick auf die Überlegenheit gegenüber den anderen betrachteten Methoden ist dieser jedoch vertretbar. Es darf jedoch nicht übersehen werden, dass keinesfalls der Anspruch einer perfekten Rekonstruktion verfälschter Signale gestellt werden kann. Interessant wäre eine vergleichende Untersuchung dieses Glättungsalgorithmus auf Basis verschiedener Daten, die ein weites Spektrum an Rauscheinflüssen aufweisen.

Die im Abschnitt 6.2 vorgestellte Methode zur Ermittlung funktional beschriebener Peaks durch (partielles) Fitting einer gewählten Peakmodellfunktion stellt eine gute Lösung zur automatisierten Interpreatation von IMS-Spektren dar. Sie ist mit Sicherheit einer numerischen Beschreibung überlegen, da grundsätzlich Überlagerungen im Tailingbereich entschlüsselt werden können. Keine Berücksichtigung fand der Fall, dass zwei Peaks nahezu deckungsgleich sind. Untersuchungen in diese Richtung sind interessant, unter der Voraussetzung einer funktionierenden Vortrennung sollte dieses Problem jedoch nicht auftreten. Die eingeführten Peakfunktionen sind alle in der Lage, Peaks in IMS-Spektren zu beschreiben. Keine Peakfunktion führt jedoch zu einer perfekten Nachbildung aller Peakverläufe. Hier lassen sich mit Sicherheit bei entsprechendem Untersuchungsaufwand bessere Peakfunktionen finden. Insbesondere driftzeitabhängige Parameter versprechen eine optimierte Detektion. Auf diese Weise kann auch eine Vereinfachung des hier ausgeblendeten Problems von (nahezu) deckungsgleichen Peaks angenommen werden.

Die im Abschnitt 6.3 vorgestellte Kettenbildung erwies sich als anwendbar. Sie war anfänglichen Versuchen mit dem Einsatz von Self-Organising-Maps (SOM) sogar überlegen, wobei in diese Richtung nicht weiter untersucht wurde. Es bleibt noch die Verbindung unterbrochener Ketten zu optimieren, was für die in dieser Arbeit betrachteten Daten schlicht nicht notwendig war.

Die Beschreibung von zweidimensionalen Peaks durch zwei eindimensionale Funktionen führt automatisch zu der Frage, ob nicht zweidimensionale Funktionen grundsätzlich besser zur Peakbeschreibung geeignet sind. Hier wird die These aufgestellt, das dies nicht der Fall ist. Der Grund liegt darin, dass die zweite Dimension den eigentlichen Funktionstypen für die Driftzeitachse nicht ändern dürfte. Schließlich handelt es sich um eine sequentiell aufgezeichnete Kette von Spektren, von denen jedes Einzelne als vollständige Messung aufgefasst werden kann. Das soll heißen, dass die Ionenbewegung nicht von der Retentionszeit abhängt und die entsprechende funktionale Beschreibung separiert für jedes Spektrum zu betrachten ist. Andererseits liegt es nahe, dass die Verteilungsgesetze bezüglich der Driftzeit konstant sind. Somit sollte sich ein zweidimensionaler Peak tatsächlich durch zwei eindimensionale

Funktionen beschreiben lassen. Klammert man Spezialfälle wie den RIP aus, so kann die Retentionszeitenfunktion, wie im Abschnitt 6.4 gezeigt, eine Rekonstruktion von „beschädigten“ Peaks ermöglichen. Die in dieser Arbeit entwickelte Methode kann zwar auf Grund ihrer Unzuverlässigkeit nicht überzeugen, zeigt jedoch, dass eine Rekonstruktion grundsätzlich möglich ist. Eine gezielte Optimierung wird sich mit hoher Wahrscheinlichkeit als fruchtbar erweisen. Es ist noch zu untersuchen, ob es tatsächlich universelle Funktionen für beide Dimensionen gibt. Ist dies der Fall (wovon hier ausgegangen wird), so ist eine Retententionszeitabhängigkeit von Höhenfunktionen wünschenswert.

Die Vergleichbarkeit von Driftzeiten sollte eigentlich durch die K_0-Berechnung gegeben sein. Auf Grund ungenauer Messungsparameter ist jedoch die im Abschnitt 7.1 vorgestellte Normalisierung am RIP zu bevorzugen. Allerdings ergeben sich auch hier geringfügige Abweichungen. Von einem Mitarbeiter des ISAS - Institute for Analytical Sciences (Dr. Wolfgang Vautz) wurde zudem der Verdacht geäußert, dass das elektrische Feld im Driftraum nicht homogen sei. Falls dies der Fall ist, sind hier Untersuchungen der Auswirkung auf die Vergleichbarkeit von Resultaten notwendig. Die beiden im Abschnitt 7.2 vorgestellten Verfahren zur Bildung von Peakverläufen und den mit Hilfe der im Abschnitt 7.3 vorgestellten Volumenapproximation daraus resultierenden Volumenverläufen führten jedoch zu brauchbaren Ergebnissen.

Bedauerlicherweise wurde im Abschnitt 7.6 nachgewiesen, dass eine Abhängigkeit über Peakvolumina nur ein unzureichendes Kriterium darstellt. Hier ist eine gezielte Untersuchung unter gesicherten Messbedingungen notwendig. Wahrscheinlich müssen auch sehr viel mehr Messungsreihen betrachtet werden, was natürlich einen erheblichen Zeitaufwand bedeutet. Es ist aber wichtig, dass Urachen für Ausreißer und nicht deckende Volumenverläufe geklärt werden. Man könnte sich zunächst auf eine Substanz beschränken. Im Hinblick auf eine nachfolgende Automatisierung wird die aus dieser Arbeit resultierende Implementierung dabei eine hilfreiche Unterstützung bieten. Außerdem ist eine Untersuchung des Ionenverlusts (vgl. Abschnitt 7.6.1) notwendig, die eine Normaliserung von Volumina bzw. Flächen zum Ziel hat.

Eine durchaus nützliche Erkenntnis wurde im Abschnitt 7.4 aufgeführt. Werden beschädigte Analytpeaks rekonstruiert, so kann eine Konzentrationsabhängigkeit allein vom Monomer abhängig gemacht werden. Die Rekonstruktion ist beim Auftreten höherer Konzentrationen, vielleicht sogar beim Eintreten vieler Ionen in den Driftraum, ein notwendiges Verfahren. Während der Bearbeitung dieser Arbeit wurde von Mitarbeitern des ISAS - Institute for Analytical Sciences (PD. Dr. Jörg-Ingo Baumbach, Dr. Luzia Seifert, Dr. Wolfgang Vautz) immer wieder explizit auf die Beachtung von Dimeren hingewiesen. Nach den simplen Überlegungen wird hier jedoch die These aufgestellt, dass die Betrachtung von rekonstruierten Monomerpeaks alle notwendigen Informationen liefert.

Die entstandene Implementierung, das Programm Codean, stellt in gewisser Weise ein Framework dar. Mit der im Kapitel 8 vorgestellten PlugIn-Architektur können auf einfache Weise neue Algorithmen zur Peakerzeugung getestet werden. Die Anbindung an eine Datenbank sollte keine Probleme bereiten. Ein zweidimensionales Fitting von Peaks ist nicht vorgesehen, allerdings können mit Sicherheit Komponenten von Codean für dieses Vorhaben wiederverwendet werden. Kurzeitig wurde überlegt, eine generellere Peakbeschreibung zu implementieren, allerdings stellte sich die Peakerkennungspipeline als nicht kompatibel heraus. Außerdem wurde ja bereits angeführt, dass eine separierte funktionale Beschreibung nicht zu schlechteren Resultaten führen sollte.

Letztendlich wurde eine Reihe von automatisierten Dateninterpretationen entwickelt, wobei im Wesentlichen alle Berechnungskomponenten leicht austauschbar sind. Bei den untersuchten Daten war leider nur ein ungenauer Zusammenhang zwischen Konzentration und Peakvolumina feststellbar. Es konnte aber gezeigt werden, dass Konzentrationsabhängigkeiten scheinbar einen logarithmischen Verlauf aufweisen. Sollte sich herausstellen, dass Peakvolumina auch

unter absolut kontrollierten Bedingungen kein stabiles Kriterium zur Konzentrationsvorhersage darstellen, so könnten andere Systeme, wie gekoppelte IMS (s. beispielsweise [9]), bessere Resultate liefern. Eine weitere Forschung in diese Richtung macht auf jeden Fall Sinn. Dafür sprechen zwei Gründe. Zum Einen konnte innerhalb einer Messungsreihe (bis auf gelegentliche Ausreißer) sehr wohl eine Abhängigkeit festgestellt werden. Zum Anderen lässt der physikalische/chemische Hintergrund der Ionenbildung eine funktionale Abhängigkeit zwischen Flächen bzw. Volumina von Peaks und der Konzentration einer Substanz vermuten.

Danksagung

Ich danke Herrn Prof. Dr. Bernd Reusch und Herrn PD. Dr. Jörg Ingo Baumbach für die Betreuung dieser Diplomarbeit und die hilfreichen Ratschläge. Herrn Dr. Lars Hildebrand danke ich für seine Unterstützung und die Kontaktherstellung mit dem ISAS - Institute for Analytical Sciences. Außerdem danke ich den Mitarbeitern des Projektbereichs „Metabolomics“ des ISAS - Institute for Analytical Sciences für die Hilfsbereitschaft. Neben Herrn PD. Dr. Jörg Ingo Baumbach, der stets bemüht war, Fragen zu theoretischen Grundlagen der Ionenmobilitätsspektrometrie zu beantworten, danke ich dabei insbesondere Frau Luzia Seifert für die Durchführung eines Großteils der verwendeten Messungen und Herrn Dr. Wolfgang Vautz für die Messungsplanung.

A. Daten

Tabelle A.1.: Die Messungsreihe MR1: Einzelsubstanz 2-Heptanon (Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:
ls081705	1.40080
ls081706	0.58383
ls081707	0.25095
ls081708	0.10866

Messung:	2-Heptanon:
ls081709	0.04509
ls081710	0.01924
ls081711	0.00837

Tabelle A.2.: Die Messungsreihe MR2: Einzelsubstanz 2-Heptanon (Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:
ls081713	2.65520
ls081714	1.13796
ls081715	0.48130
ls081716	0.20902

Messung:	2-Heptanon:
ls081717	0.08998
ls081718	0.03925
ls081719	0.01610
ls081720	0.00688

Tabelle A.3.: Die Messungsreihe MR3: Einzelsubstanz 2-Heptanon (Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:
ls101201	3.03800
ls101202	1.96904
ls101203	1.30419
ls101204	0.82537
ls101205	0.54550
ls101206	0.36026

Messung:	2-Heptanon:
ls101207	0.23827
ls101208	0.15567
ls101209	0.09866
ls101210	0.06506
ls101211	0.04300

Tabelle A.4.: Die Messungsreihe MR4: Einzelsubstanz 2-Heptanon(Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:
ls101213	3.03800
ls101214	1.99627
ls101215	1.30324
ls101216	0.85636
ls101217	0.56312

Messung:	2-Heptanon:
ls101218	0.37217
ls101219	0.24580
ls101220	0.16151
ls101221	0.10628

Tabelle A.5.: Die Messungsreihe MR5: Gemisch aus 2-Heptanon, 2-Hexanon und 2-Octanon (Konzentrationsangaben in μg/L, 2-Octanon unbekannt)

Messung:	2-Heptanon:	2-Hexanon:
ls101802	3.03000	2.76000
ls101803	1.96528	1.79015
ls101804	1.26551	1.15274
ls101805	0.83277	0.75856
ls101806	0.54840	0.49953
ls101807	0.36166	0.32943
ls101808	0.23851	0.21725

Messung:	2-Heptanon:	2-Hexanon:
ls101809	0.15729	0.14327
ls101810	0.10313	0.09394
ls101811	0.07016	0.06391
ls101812	0.04443	0.04047
ls101813	0.02768	0.02521
ls101814	0.00055	0.00050

Tabelle A.6.: Die Messungsreihe MR6: Gemisch aus 2-Heptanon, 2-Hexanon und 2-Octanon (2-Heptanon Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:
ls102003	2.99700
ls102004	1.97646
ls102005	1.28752
ls102006	0.84909
ls102007	0.56036

Messung:	2-Heptanon:
ls102008	0.36821
ls102009	0.24248
ls102010	0.15876
ls102011	0.10462

Tabelle A.7.: Die Messungsreihe MR7: Gemisch aus 2-Heptanon, 2-Hexanon und 2-Octanon (2-Heptanon Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:
ls110208	2.96600
ls110209	1.95884
ls110210	1.29275
ls110211	0.85501

Messung:	2-Heptanon:
ls110212	0.56549
ls110213	0.37401
ls110214	0.24683

Tabelle A.8.: Die Messungsreihe MR8: Gemisch aus 2-Heptanon, 2-Hexanon und 2-Octanon (Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:	2-Hexanon:	2-Octanon:
ls111701	2.90260	5.70080	6.68780
ls111702	1.90592	3.74329	4.39138
ls111703	1.25782	2.47041	2.89812
ls111704	0.82592	1.62213	1.90298
ls111705	0.54704	1.07441	1.26043
ls111706	0.35328	0.69385	0.81398
ls111707	0.22765	0.44712	0.52453
ls111708	0.14938	0.29338	0.34417
ls111709	0.09851	0.19348	0.22697

Tabelle A.9.: Die Messungsreihe MR9: Gemisch aus 2-Heptanon, 2-Hexanon und 2-Octanon (Konzentrationsangaben in μg/L)

Messung:	2-Heptanon:	2-Hexanon:	2-Octanon:
ls112801	2.86440	5.60510	6.68780
ls112802	1.89174	3.70179	4.41684
ls112803	1.13733	2.22554	2.65543
ls112804	0.73981	1.44768	1.72732
ls112805	0.48930	0.95748	1.14243
ls112806	0.32222	0.63052	0.75232
ls112807	0.21234	0.41552	0.49578
ls112808	0.13893	0.27185	0.32437
ls112809	0.09169	0.17941	0.21407
ls112810	0.06064	0.11866	0.14158
ls112811	0.03988	0.07803	0.09310
ls112812	0.02626	0.05138	0.06131
ls112813	0.01713	0.03352	0.04000

Tabelle A.10.: Die Messungsreihe MR10: Einzelsubstanz 2-Octanon (Konzentrationsangaben in μg/L)

Messung:	2-Octanon:
ls121801	6.56200
ls121802	4.26540
ls121803	2.81904
ls121804	1.87123
ls121805	1.22515

Messung:	2-Octanon:
ls121806	0.80913
ls121807	0.53091
ls121808	0.35216
ls121809	0.22595

B. Konzentrationsabhängigkeiten

Tabelle B.1.: Die Parameter zu den in Abbildung B.1 und B.2 dargestellten Konzentrationsabhängigkeiten von 2-Heptaon mit f_{log}, gerundet auf vier Nachkommastellen

Messungsreihe:	Peak:	Peak-/Höhenfunktion:	p_1 :	p_2 :	p_3 :
MR1	Monomer	f_{gbw}, f_{In}	0.0088	2.2169	63.7261
MR1	Dimer	f_{gbw}, f_{In}	0.1544	8.5629	0.2825
MR2	Monomer	f_{gbw}, f_{In}	0.0026	1.3739	255.1663
MR2	Dimer	f_{gbw}, f_{In}	0.0299	1.8372	1.9354
MR3	Monomer	f_{gbw}, f_{In}	0.0127	2.4726	32.9649
MR3	Dimer	f_{gbw}, f_{In}	0.1969	1474.2694	0.0012
MR4	Monomer	f_{gbw}, f_{In}	0.0134	2.5563	11.3011
MR4	Dimer	f_{gbw}, f_{In}	0.3319	0.5491	4.7131
MR1	Monomer	f_{gbw}, f_{lbwe}	0	2.2386	76.5227
MR1	Dimer	f_{gbw}, f_{lbwe}	0.2045	0.8538	9.4618
MR2	Monomer	f_{gbw}, f_{lbwe}	0	1.733	114.171
MR2	Dimer	f_{gbw}, f_{lbwe}	0.01	1.3713	4.1279
MR3	Monomer	f_{gbw}, f_{lbwe}	0	2.6253	33.5963
MR3	Dimer	f_{gbw}, f_{lbwe}	0.0323	69.5151	0.03
MR4	Monomer	f_{gbw}, f_{lbwe}	0	2.1744	23.3471
MR4	Dimer	f_{gbw}, f_{lbwe}	0.4523	0.3821	23.7784
MR1	Monomer	$f_{lbwie2+3}$, f_{In}	0	1.7367	54.8901
MR1	Dimer	$f_{lbwie2+3}$, f_{In}	0.2014	2.0243	1.7126
MR2	Monomer	$f_{lbwie2+3}$, f_{In}	0	1.7945	40.1994
MR2	Dimer	$f_{lbwie2+3}$, f_{In}	0.0699	1.9707	2.1736
MR3	Monomer	$f_{lbwie2+3}$, f_{In}	0	4.0314	8.8223
MR3	Dimer	$f_{lbwie2+3}$, f_{In}	0.269	2.8843	1.2817
MR4	Monomer	$f_{lbwie2+3}$, f_{In}	0.0662	1.6945	28.7217
MR4	Dimer	$f_{lbwie2+3}$, f_{In}	0.3176	0.8413	3.245
MR1	Monomer	$f_{lbwie2+3}$, f_{lbwe}	0	1.7967	64.7249
MR1	Dimer	$f_{lbwie2+3}$, f_{lbwe}	0.1190	2.0777	2.4146
MR2	Monomer	$f_{lbwie2+3}$, f_{lbwe}	0	1.6315	67.2067
MR2	Dimer	$f_{lbwie2+3}$, f_{lbwe}	0.0656	1.1895	8.1706
MR3	Monomer	$f_{lbwie2+3}$, f_{lbwe}	0	3.2396	15.9754
MR3	Dimer	$f_{lbwie2+3}$, f_{lbwe}	0.1006	15.4773	0.1992
MR4	Monomer	$f_{lbwie2+3}$, f_{lbwe}	0	2.0733	21.6921
MR4	Dimer	$f_{lbwie2+3}$, f_{lbwe}	0.3563	0.5502	17.0176

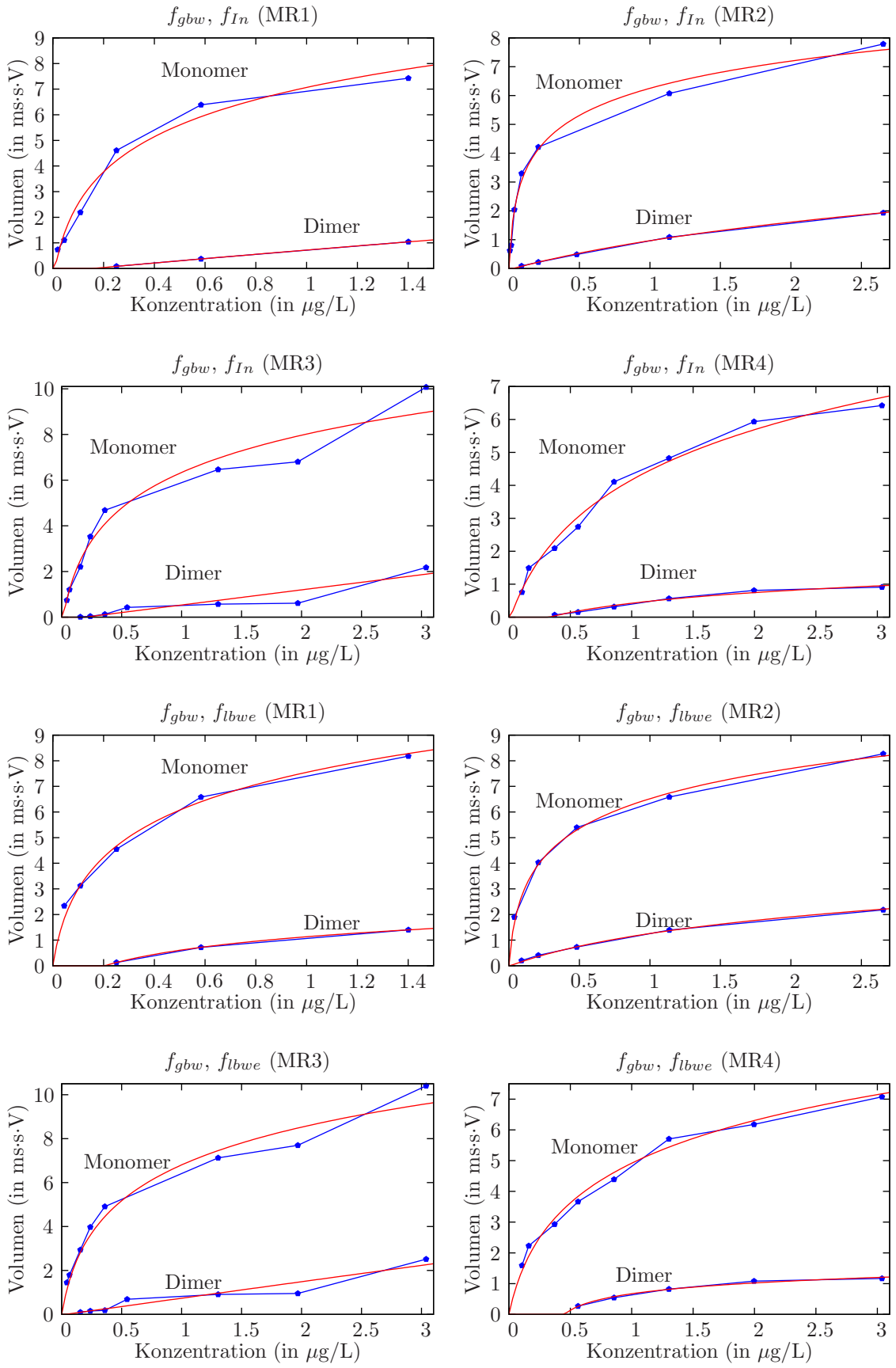

Abbildung B.1.: Konzentrationsabhängigkeitsfunktionen f_{log} für 2-Heptanon

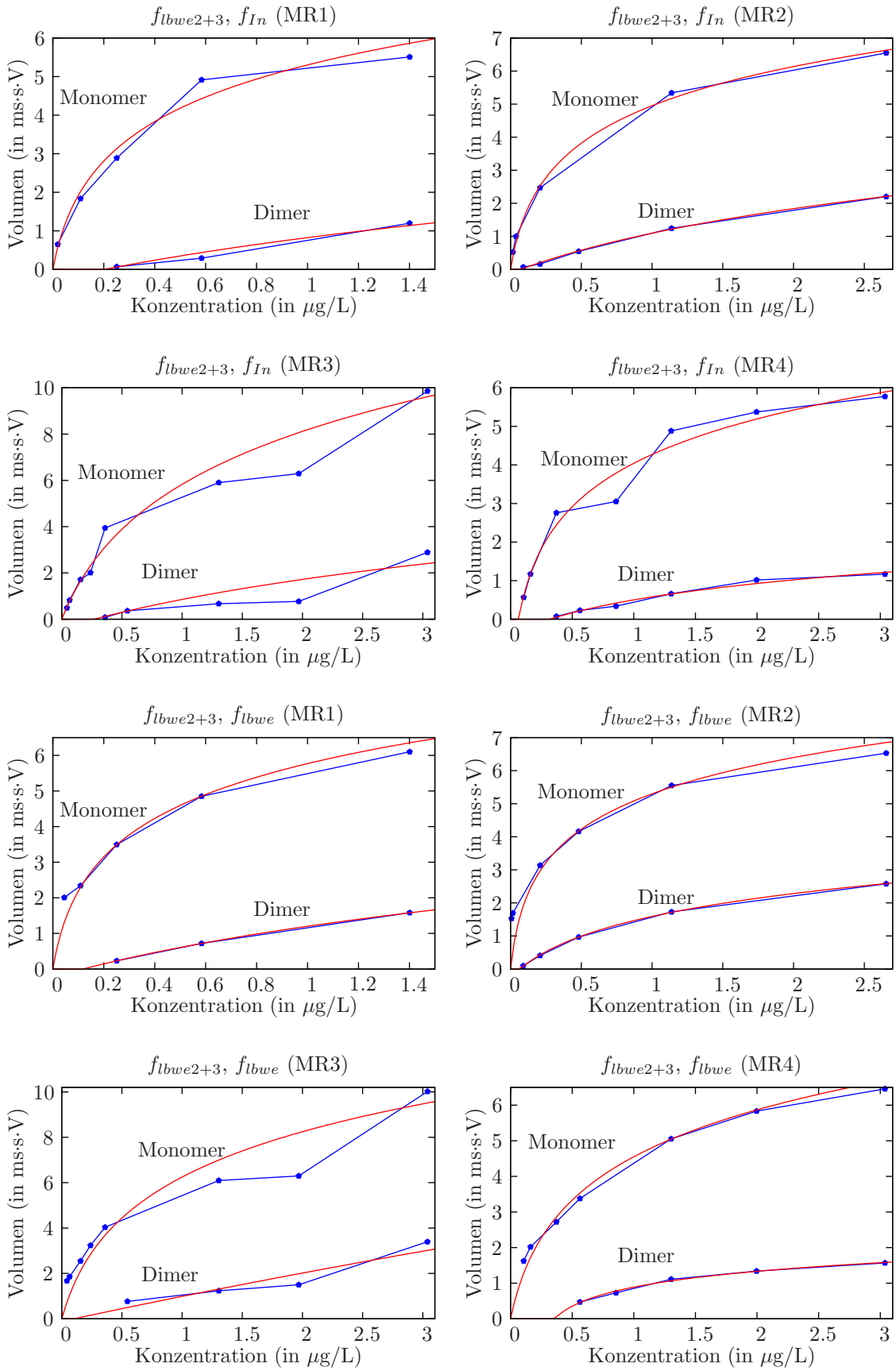

Abbildung B.2.: Konzentrationsabhängigkeitsfunktionen f_{log} für 2-Heptanon

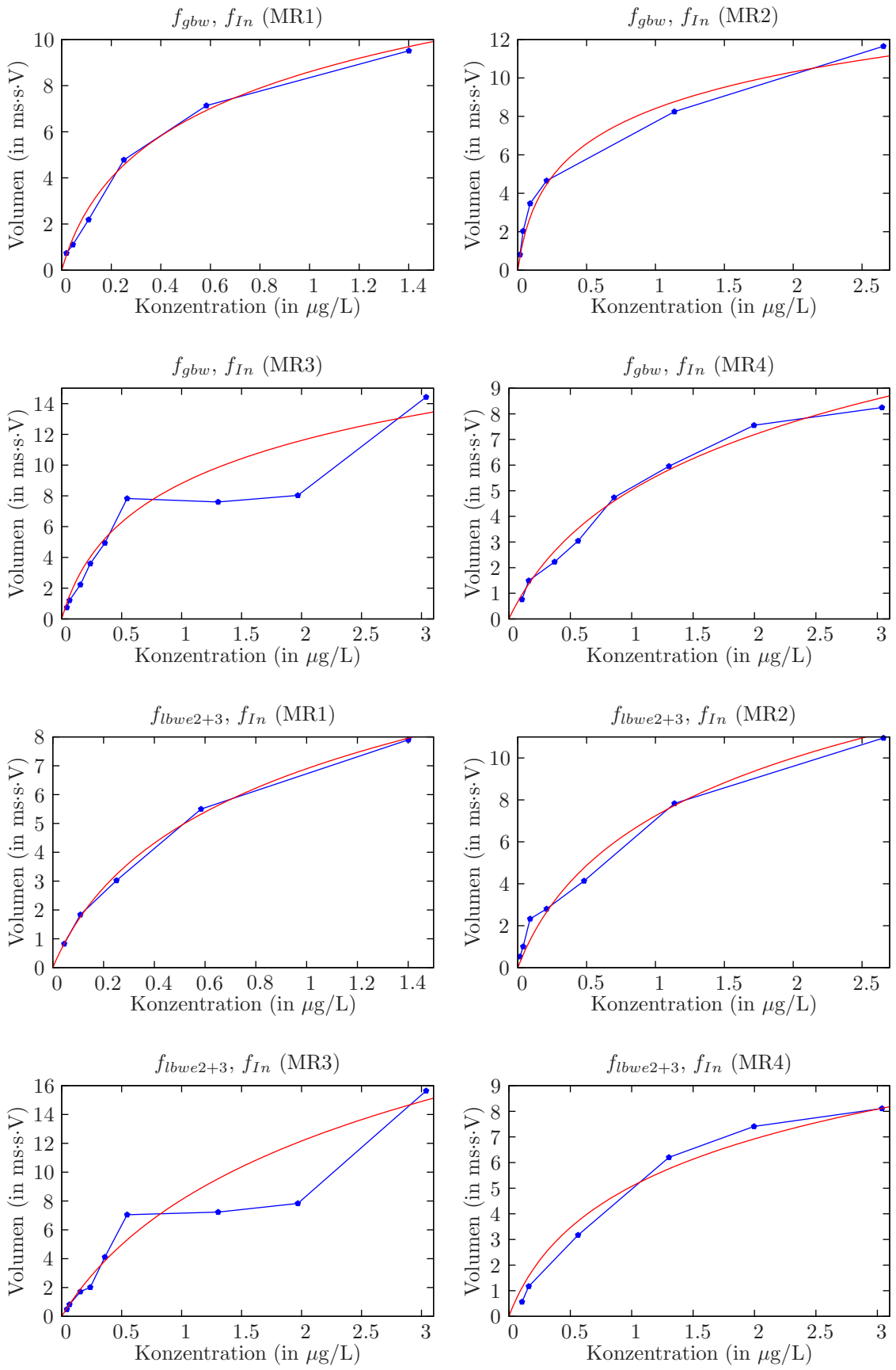

Abbildung B.3.: Kombinierte Konzentrationsabhängigkeiten mit f_{log} für 2-Heptanon

Tabelle B.2.: Die Parameter zu den in Abbildung B.3 dargestellten Konzentrationsabhängigkeiten von 2-Heptanon mit f_{log}, gerundet auf vier Nachkommastellen

Messungsreihe:	Peak-/Höhenfunktion:	p_1 :	p_2 :	p_3 :
MR1	f_{gbw}, f_{In}	0	3.4946	29.1283
MR2	f_{gbw}, f_{In}	0	2.8538	49.0614
MR3	f_{gbw}, f_{In}	0	4.5125	16.4309
MR4	f_{gbw}, f_{In}	0	4.0045	6.8284
MR1	$f_{lbwie2+3}$, f_{In}	0	3.6031	15.7226
MR2	$f_{lbwie2+3}$, f_{In}	0	4.7916	9.5978
MR3	$f_{lbwie2+3}$, f_{In}	0	8.52	4.3059
MR4	$f_{lbwie2+3}$, f_{In}	0	3.1527	10.8625

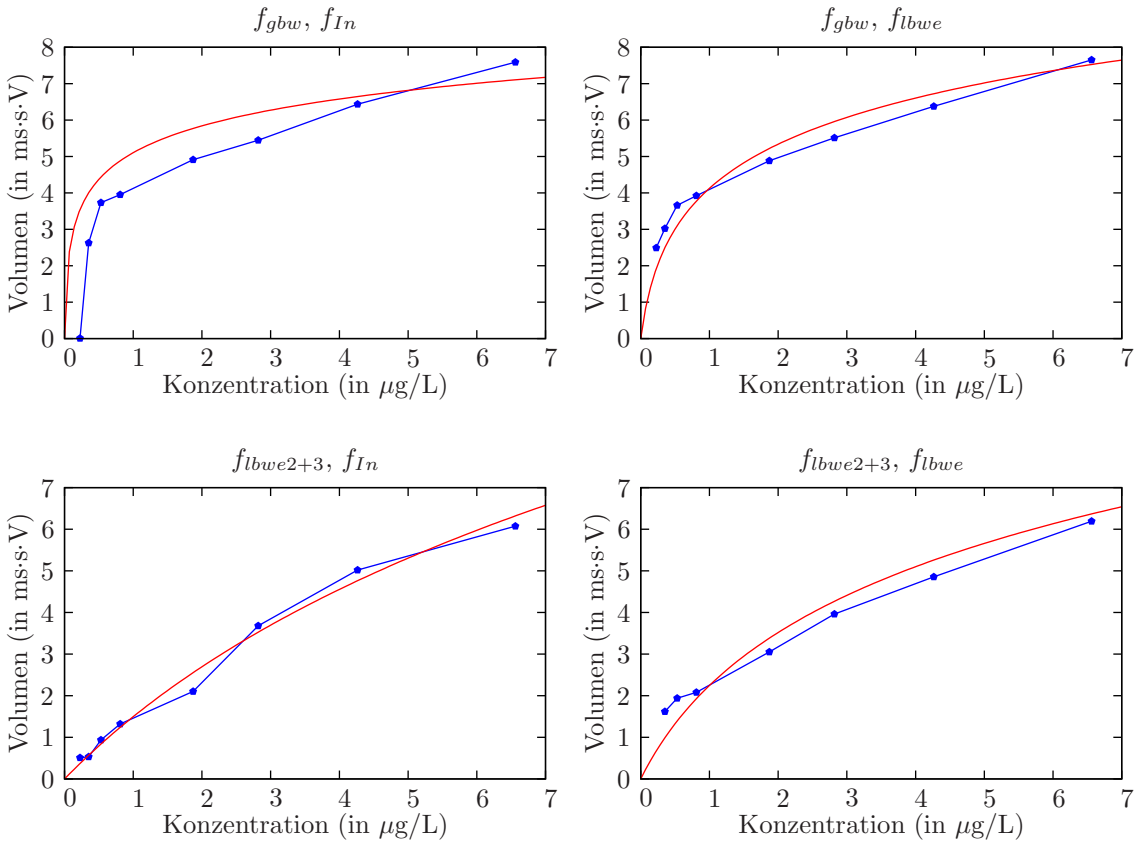

Abbildung B.4.: Konzentrationsabhängigkeiten des 2-Octanon Monomers mit verschiedenen Peakfunktionen

Tabelle B.3.: Die Parameter zu den in Abbildung B.4 dargestellten Konzentrationsabhängigkeiten von 2-Octanon mit f_{log}, gerundet auf vier Nachkommastellen

Peakfunktion:	Höhenfunktion:	p_1 :	p_2 :	p_3 :
f_{gbw}	f_{In}	0	1.0685	320.2809
f_{gbw}	f_{lbwe}	0	1.9137	20.7594
$f_{lbwie2+3}$	f_{In}	0	6.0587	0.7615
$f_{lbwie2+3}$	f_{lbwe}	0	3.0197	2.9994

C. Konzentrationsbestimmungsfehler

Im Folgenden sind gemittelte Fehler über Messungsreihen bei den verschiedenen in Konzentrationsabhängigkeiten aufgeführt. Ausreißer wurden nicht aussortiert. Die Messungsreihe MR3 wurde auf Grund ihres problematischen Volumenverlaufs ignoriert. Bei den Werten in Tabelle C.5 ist zu beachten, dass bei 2-Octanon im Vergleich zu 2-Heptanon viel höhere Konzentrationen auftreten.

Tabelle C.1.: Mittlere Fehler von Konzentrationsbestimmungen in μg/L, gerundet auf vier Nachkommastellen

MR1 (f_{gbw}, f_{In}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.0622	0.028
MR2	0.2333	0.0654
MR4	0.6837	0.45
MR5	0.7492	0.4309
MR6	0.4428	0.2663
MR7	0.818	0.4977
MR8	0.8447	0.4604
MR9	0.5572	0.1704

MR1 (f_{gbw}, f_{In}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.0283	0.022
MR2	0.0929	0.0686
MR4	0.6488	0.4335
MR5	1.0537	0.4152
MR6	0.4112	0.3118
MR7	0.4475	0.4044
MR8	0.7844	0.4417
MR9	0.5125	0.1713

MR2 (f_{gbw}, f_{In}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.1253	0.034
MR2	0.0942	0.0192
MR4	0.6508	0.4924
MR5	1.1076	0.3579
MR6	0.2611	0.2308
MR7	0.7004	0.4822
MR8	0.873	0.4979
MR9	0.5799	0.1935

MR2 (f_{gbw}, f_{In}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.0315	0.0189
MR2	0.1359	0.0415
MR4	0.6595	0.4576
MR5	1.389	0.4397
MR6	0.3562	0.3441
MR7	0.3672	0.2941
MR8	0.8043	0.4627
MR9	0.525	0.1812

MR4 (f_{gbw}, f_{In}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.8816	0.2301
MR2	0.7431	0.5457
MR4	0.1739	0.0857
MR5	2.2835	1.5008
MR6	0.8335	0.3682
MR7	0.4512	0.4434
MR8	0.6061	0.176
MR9	0.3883	0.0436

MR4 (f_{gbw}, f_{In}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.7059	0.1808
MR2	0.974	0.4717
MR4	0.1528	0.0765
MR5	2.6749	0.8133
MR6	0.6598	0.2139
MR7	1.0009	0.6008
MR8	0.4657	0.0693
MR9	0.2736	0.0632

Tabelle C.2.: Mittlere Fehler von Konzentrationsbestimmungen in μg/L, gerundet auf vier Nachkommastellen

MR1 (f_{gbw}, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.1048	0.0203
MR2	0.224	0.0464
MR4	0.6722	0.4161
MR5	0.7753	0.3872
MR6	0.5185	0.3322
MR7	0.8225	0.5103
MR8	0.8526	0.4812
MR9	0.5746	0.1476

MR1 (f_{gbw}, f_{lbwe}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.0984	0.0298
MR2	0.1278	0.0563
MR4	0.6263	0.3878
MR5	1.0947	0.3267
MR6	0.4108	0.3634
MR7	0.4944	0.3554
MR8	0.7364	0.4362
MR9	0.5377	0.1381

MR2 (f_{gbw}, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.1735	0.0366
MR2	0.0421	0.0467
MR4	0.5624	0.3905
MR5	1.2325	0.3573
MR6	0.2858	0.2844
MR7	0.6968	0.4045
MR8	0.8417	0.4772
MR9	0.5745	0.1458

MR2 (f_{gbw}, f_{lbwe}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.0758	0.0324
MR2	0.0449	0.0439
MR4	0.5876	0.3763
MR5	1.3795	0.3877
MR6	0.3059	0.3494
MR7	0.3729	0.2913
MR8	0.7213	0.431
MR9	0.5357	0.1355

MR4 (f_{gbw}, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.9107	0.2637
MR2	0.7596	0.4247
MR4	0.1215	0.0916
MR5	2.6303	1.1092
MR6	0.5178	0.1546
MR7	0.4785	0.4183
MR8	0.6937	0.3209
MR9	0.5292	0.0837

MR4 (f_{gbw}, f_{lbwe}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.7246	0.2111
MR2	0.9485	0.4209
MR4	0.1304	0.0683
MR5	3.3961	1.0199
MR6	0.7101	0.0826
MR7	0.8851	0.7235
MR8	0.4574	0.0995
MR9	0.4453	0.0811

Tabelle C.3.: Mittlere Fehler von Konzentrationsbestimmungen in μg/L, gerundet auf vier Nachkommastellen

MR1 ($f_{lbwie2+3}, f_{In}$):

Messungsreihe:	Mittelwert:	Median:
MR1	0.1228	0.0157
MR2	0.0611	0.035
MR4	0.5058	0.3764
MR5	1.0109	0.5827
MR6	0.3009	0.2506
MR7	0.797	0.4785
MR8	0.8164	0.4808
MR9	0.5617	0.1825

MR1 ($f_{lbwie2+3}, f_{In}$) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.0927	0.0238
MR2	0.1522	0.067
MR4	0.4602	0.3193
MR5	1.1746	0.7858
MR6	0.3037	0.2304
MR7	0.6059	0.4366
MR8	0.665	0.4353
MR9	0.49	0.1682

MR2 ($f_{lbwie2+3}, f_{In}$):

Messungsreihe:	Mittelwert:	Median:
MR1	0.054	0.0105
MR2	0.0579	0.0696
MR4	0.45	0.3249
MR5	1.0543	0.6708
MR6	0.2482	0.219
MR7	0.7223	0.4083
MR8	0.7825	0.4593
MR9	0.5447	0.1732

MR2 ($f_{lbwie2+3}, f_{In}$) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.0463	0.0235
MR2	0.0561	0.0301
MR4	0.4921	0.2976
MR5	0.903	0.8487
MR6	0.3821	0.2113
MR7	0.6341	0.4243
MR8	0.6584	0.4182
MR9	0.4847	0.1595

MR4 ($f_{lbwie2+3}, f_{In}$):

Messungsreihe:	Mittelwert:	Median:
MR1	0.322	0.1424
MR2	0.5651	0.1749
MR4	0.1575	0.1667
MR5	2.8083	1.9436
MR6	0.5122	0.1576
MR7	0.4441	0.1866
MR8	0.641	0.3494
MR9	0.4708	0.0998

MR4 ($f_{lbwie2+3}, f_{In}$) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.3294	0.0889
MR2	0.9272	0.1834
MR4	0.1383	0.0912
MR5	2.8032	1.2061
MR6	0.4529	0.1294
MR7	0.1087	0.0567
MR8	0.4497	0.3549
MR9	0.3854	0.1381

Tabelle C.4.: Mittlere Fehler von Konzentrationsbestimmungen in μg/L, gerundet auf vier Nachkommastellen

MR1 ($f_{lbwie2+3}$, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.1258	0.0717
MR2	0.1276	0.0542
MR4	0.4953	0.3194
MR5	1.1289	0.5913
MR6	0.3524	0.2777
MR7	0.768	0.436
MR8	0.7764	0.4792
MR9	0.5186	0.1427

MR1 ($f_{lbwie2+3}$, f_{lbwe}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.1279	0.095
MR2	0.1115	0.0723
MR4	0.477	0.2538
MR5	1.2027	0.9274
MR6	0.2996	0.1788
MR7	0.3989	0.2469
MR8	0.5432	0.3669
MR9	0.4385	0.1188

MR2 ($f_{lbwie2+3}$, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.0503	0.0504
MR2	0.0795	0.0557
MR4	0.4409	0.2912
MR5	1.4323	0.5869
MR6	0.2709	0.2598
MR7	0.6993	0.3528
MR8	0.7342	0.4603
MR9	0.5016	0.1348

MR2 ($f_{lbwie2+3}$, f_{lbwe}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.0691	0.0556
MR2	0.0293	0.0346
MR4	0.513	0.3353
MR5	1.5754	0.9415
MR6	0.2881	0.2171
MR7	0.4166	0.3297
MR8	0.6021	0.4237
MR9	0.4717	0.1519

MR4 ($f_{lbwie2+3}$, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR1	0.3413	0.2098
MR2	0.3182	0.2758
MR4	0.1395	0.0422
MR5	2.2944	1.7216
MR6	0.1417	0.0928
MR7	0.4724	0.1154
MR8	0.624	0.384
MR9	0.4315	0.0453

MR4 ($f_{lbwie2+3}$, f_{lbwe}) kombiniert:

Messungsreihe:	Mittelwert:	Median:
MR1	0.3363	0.1758
MR2	0.6498	0.2501
MR4	0.1372	0.1185
MR5	3.106	1.1929
MR6	0.5355	0.1362
MR7	0.4492	0.3839
MR8	0.2622	0.2053
MR9	0.301	0.0648

Tabelle C.5.: Mittlere Fehler von 2-Octanon Konzentrationsbestimmungen in μg/L, gerundet auf vier Nachkommastellen

MR10 (f_{gbw}, f_{In}):

Messungsreihe:	Mittelwert:	Median:
MR8	1.0641	0.3361
MR9	1.0524	0.3812
MR10	1.0602	0.7696

MR10 (f_{gbw}, f_{lbwe}):

Messungsreihe:	Mittelwert:	Median:
MR8	0.408	0.4162
MR9	0.6351	0.1583
MR10	0.5482	0.4446

MR10 ($f_{lbwie2+3}, f_{In}$):

Messungsreihe:	Mittelwert:	Median:
MR8	0.574	0.4086
MR9	0.4132	0.1416
MR10	0.237	0.1609

MR10 ($f_{lbwie2+3}, f_{lbwe}$):

Messungsreihe:	Mittelwert:	Median:
MR8	0.4324	0.3119
MR9	0.3481	0.1589
MR10	0.4467	0.3046

Literaturverzeichnis

[1] Bader, Sabine: *Atemluftüberwachung mittels mikrostrukturierter Ionenbeweglichkeitsspektrometrie: Statistische Analyse zum Auffinden von Biomarkern für Lungenkrebs*, Universität Dortmund, Diplomarbeit, 2005

[2] Bartsch, Hans-Jochen: *Taschenbuch Mathematischer Formeln.* 19. Carl Hanser, München (u.a.), 2001. – ISBN 3–446–21792–4

[3] Baumbach, Jörg I. ; Eiceman, Gary A.: Ion Mobility Spectrometry: Arriving On Site and Moving Beyond a Low Profile. *Applied Spectroscopy* 53 (1999), Nr. 9, S. 338–335

[4] Blatter, Christian: *Wavelets - Eine Einführung.* Vieweg, Braunschweig, Wiesbaden, 1998. – ISBN 3–528–06947–3

[5] Bock, R. K. ; Krischer, W.: *The Data Analysis BriefBook.* – URL: http://rkb.home.cern.ch/rkb/titleA.html (11.2.2007)

[6] Brigham, Elbert O.: *The Fast Fourier Transform and its Applications.* Prentice-Hall Internat. , Englewood Cliffs-New Jersey, 1988. – ISBN 0–13–307547–8

[7] Dragotti, Pier L. ; Vetterli, Martin: Shift-Invariant Gibbs Free Denoising Algorithm based on Wavelet Transform Footprints. *Proceedings of SPIE, the International Society for Optical Engineering* 4119 (2000), S. 821–830

[8] Eiceman, Gary A.: Advances in Ion Mobility Spectrometry: 1980-1990. *Critical Reviews in Analytical Chemistry* 22 (1991), Nr. 1 & 2, S. 17–36

[9] Eiceman, Gary A. ; Karpas, Zeev: *Ion Mobility Spectrometry.* 2. CRC Press, Boca Raton-FL (u.a.), 2005. – ISBN 0–8493–2247–2

[10] Engelbrecht, Andries P.: *Computational Intelligence: An Introduction.* Wiley, Chichester (u.a.), 2003. – Nachdruck. – ISBN 0–470–84870–7

[11] Fogel, David B.: *Evolutionary Computation.* IEEE Press, Piscataway-New Jersey, 1995. – ISBN 0–7803–1038–1

[12] Frazier, Michael W.: *An Introduction to Wavelets through Linear Algebra.* Springer, New York (u.a.), 1999. – ISBN 0–387–98639–1

[13] Gaspar, Marcel: *Entwicklung von Fuzzy-Transformationen mit adaptiven Basisfunktionen als Werkzeug zur Datenreduktion und Merkmalserzeugung*, Universität Dortmund, Diplomarbeit, 2006

[14] Herrera, F. ; Lozano, M. ; Verdegay, JL: Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. *Artificial Intelligence Review* 12 (1998), Nr. 4, S. 265–319

[15] Hildebrand, Lars: *Asymmetrische Evolutionsstrategien*, Universität Dortmund, Diss., 2001

[16] Holschneider, Matthias: *Wavelets - An Analysis Tool.* Clarendon Press, Oxford, 1995. – ISBN 0–19–853481–7

[17] Jensen, Arne ; Cour-Harbo, Anders L.: *Ripples in Mathematics - The Discrete Wavelet Transform.* Springer, Berlin (u.a), 2001. – ISBN 3–540–41662–5

[18] Lawson, Charles L. ; Hanson, Richard J.: *Solving Least Squares Problems.* Prentice-Hall, Englewood Cliffs-New Jersey, 1974 (Prentice-Hall Series in Automatic Computation). – ISBN 0–13–822585–0

[19] Louis, Alfred K. ; Maass, Peter ; Rieder, Andreas: *Wavelets.* 2. Teubner, Stuttgart, 1998. – ISBN 3–519–12094–1

[20] Louis, Robert H. S. ; Herbert H. Hill, Jr.: Ion Mobility Spectrometry in Analytical Chemistry. *Critical Reviews in Analytical Chemistry* 21 (1990), Nr. 5, S. 321–355

[21] Mager, Harry: *Moderne Regressionsanalyse.* Salle (u.a), Frankfurt am Main (u.a.), 1982 (Texte zur Chemie und Chemietechnik). – ISBN 3–7935–5524–0

[22] NIST/SEMATECH: *e-Handbook of Statistical Methods.* – URL: http://www.itl.nist.gov/div898/handbook/ (5.1.2006)

[23] Nussbaumer, Henri J.: *Fast Fourier Transform and Convolution Algorithms.* 2. Springer, Berlin (u.a.), 1982. – ISBN 3–540–11825–X

[24] Perfilieva, Irina: Fuzzy Transform: Application to Reef Growth Problem. *Fuzzy Logic in Geology, Academic Press, Amsterdam* (2003), S. 275–300

[25] Perfilieva, Irina: Fuzzy transforms: Theory and applications. *Fuzzy Sets and Systems* 157 (2006), Nr. 8, S. 993–1023

[26] Pratt, William K.: *Digital Image Procesing.* 2. Wiley, New York (u.a.), 1991. – ISBN 0–471–85766–1

[27] Press, William H. ; Teukolsky, Saul A. ; Vetterling, William T. ; Flannery, Brian P.: *Numerical Recipes in C: The Art of Scientific Computing.* 2. Cambridge University Press, Cambridge (u.a), 1992. – ISBN 0–521–43108–5

[28] Rockmore, Daniel N.: The FFT: An Algorithm the Whole Family Can Use. *Computing in Science and Engineering* 2 (2000), Nr. 1, S. 60–64

[29] Rousseeuw, Peter J. ; Leroy, Annick M.: *Robust Regression and Outlier Detection.* Wiley, New York (u.a.), 1987 (Wiley Series in Probability and Mathematical Statistics : Applied Probability and Statistics). – ISBN 0–471–85233–3

[30] Ruzsányi, Veronika: *Analyse flüchtiger Metaboliten von der Ausatemluft mittels Inonenmobilitätsspektrometer*, Universität Dortmund, Diss., 2005

[31] Salvatore, Dominick: *Theory and Problems of Statistics and Econometrics.* McGraw-Hill, New York (u.a), 1982 (Schaum's Outline Series). – ISBN 0–07–054505–7

[32] Scharf, Louis L.: *Statistical Signal Processing: Detection, Estimation and Time Series Analysis.* Addison-Wesley, München (u.a.), 1991. – ISBN 0–201–19038–9

[33] Schwefel, Hans-Paul: *Evolution and Optimum Seeking.* Wiley, New York (u.a.), 1995. – ISBN 0–471–57148–2

[34] Sielemann, Stefanie: *Detektion flüchtiger organischer Verbindungen mittels Ionenmobilitätsspektromonie und deren Kopplung mit Multi-Kapillar-Gas-Chromatographie*, Universität Dortmund, Diss., 1999

[35] Smith, Steven W.: *The Scientist and Engineer's Guide to Digital Signal Processing.* 2. California Technical Publishing, San Diego, 1999. – ISBN 0–9660176–6–8

[36] Spiegel, Murray R.: *Theory and Problems of Statistics.* 2. McGraw-Hill, New York (u.a.), 1988 (Schaum's Outline Series in Mathematics). – ISBN 0–07–060234–4

[37] Steinbrecher, Rainer: *Bildverarbeitung in der Praxis.* Oldenbourg, München, 1993. – ISBN 3–489–22372–0

[38] Štěpnička, Martin ; Válašek, Radek: Numerical solution of partial differential equations with help of fuzzy transform. *Fuzzy Systems, 2005. FUZZ'05. The 14th IEEE International Conference on* (2005), S. 1104–1109

[39] Stollnitz, Eric J. ; DeRose, Tony D. ; Salesin, David H.: *Wavelets for Computer Graphics.* Morgan Kaufmann, San Francisco, 1996. – ISBN 1–55860–375–1

[40] Strobel, Siehyun: *Einsatz von Support Vector Machines und Wavelet Transformation zur Analyse von Tiefenprofilen*, Universität Dortmund, Diplomarbeit, 2006

[41] Sweldens, Wim ; Schröder, Peter: Building your own Wavelets at Home. *Wavelets in Computer Graphics* 87 (1996)

[42] Tanaka, Kazuo: *An Introduction to Fuzzy Logic for Practical Applications.* Springer, New York (u.a.), 1997. – ISBN 0–387–94807–4

[43] Tilli, Thomas: *Fuzzy-Logik: Grundlagen, Anwendungen, Hard- und Software.* 3. Franzis, München, 1993. – ISBN 3–7723–4323–6

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Eidesstattliche Versicherung

Ich erkläre an Eides Statt, dass ich die vorliegende Arbeit selbständig und nur unter Benutzung der angegebenen Literatur angefertigt habe. Aus anderen Werken übernommene Inhalte sind durch Quellenangaben kenntlich gemacht. Diese Arbeit hat noch keiner Prüfungsbehörde vorgelegen.

Dortmund, den 4. Mai 2007 .....

(Unterschrift)