Resource-Constrained Analysis of Ion Mobility Spectrometry Data

Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Technischen Universität Dortmund
an der Fakultät für Informatik
von

Dominik Kopczynski

Dortmund
2017

Tag der mündlichen Prüfung: 06. Februar 2017

Dekan: Prof. Dr. Gernot A. Fink

1. Gutachter: Prof. Dr. Sven Rahmann
2. Gutachter: Prof. Dr. Jörg Rahnenführer

Abstract

During the past decades numerous spectrometry devices, e.g. mass spectrometry or liquid or gas chromatography, have been engineered to measure the different properties of molecules, compounds or even complex structures. One device exploits the different mobilities of ionized analytes, the so-called ion mobility spectrometer. The advantages of this device are the low costs of production and maintenance (e.g. a high vacuum as in mass spectrometry is not required), the fast capture (a few milliseconds suffice) and the provision of a high resolution of up to parts per billion (ppb) by volume. An ion mobility spectrometer coupled with a multi-capillary column for pre-separation achieves a resolution higher by several magnitudes. Substantial research was done to investigate its feasibility for clinical or biotechnology applications, especially clinical diagnosis or live monitoring. Ongoing miniaturization provides devices of even mobile phone-size, allowing mobile applications. In critical places like main stations or sports stadiums, mobile devices are conceivable for the detection of drugs or explosives. Another application scenario can be a mobile device monitoring the breath of patients which can be used at home. For such scenarios it is inevitable that the data is analyzed directly at the device right after the capturing. The amount of data, the complexity of the two-dimensional spectra as well as time and device restrictions require analysis software specifically designed for this application.

The basis of MCC/IMS analysis is a representation of all high-intensity regions (peaks) in the measurement by using a few descriptive parameters per peak instead of the full measurement data, a process that we refer to as *peak extraction*. The position of peaks infers the corresponding analyte and its signal intensity delivers information about the concentration of the analyte. These peaks can hint at several features, e.g. diseases in clinical diagnostics. Previous work mainly concentrated on the extraction of the position of the peaks' highest signal intensity (mode). Using statistical distributions, we introduce a function which requires only seven descriptive parameters to approximate the complete shape of a peak. The straightforward nature of this function as well as the intuitive descriptors simplify and accelerate the methods estimating the descriptor set for every detected peak. Additional post-processing steps like comparison with a reference, or aligning or clustering a set of measurements further simplify and add precision to the provided peak model.

Having a measurement and the proposed peak model, the peaks have to be detected and the model descriptors have to be estimated automatically.

Here, we introduce two methods executing this task. The offline peak model estimation reduces one measurement automatically into a set of peak models but without any restrictions, i.e. the data is completely available during the whole analysis process and can be accessed as often as required. Furthermore, space and time restrictions were not taken into account. The idea of this method is to take its approaches as a basis and redesign them for an online analysis. Our second introduced method is referred to as online peak model estimation. The method is restricted to store only one or a small quantity of consecutive ion mobility spectra and discard the raw data directly after the analysis. Additionally, this analysis has a strict time restriction provided by the device itself (every 100 ms a new ion mobility spectrum is captured) and should even run in time on current embedded systems as the Raspberry Pi. Of course, this method should also provide a list of peak descriptors. For that purpose, we redesigned particular methods to satisfy these restrictions. This method is suitable for the application on mobile detection devices.

To find commonalities and differences among a set of measurements for further classification or timeline analysis, it is an inherent necessity to find and connect peaks provided by the same analyte. We refer to these clusters covering several peaks from different measurements as *consensus peaks*. Several clustering methods are already introduced in literature, but many have the disadvantage of requesting the number of clusters a priori. We introduce an enhanced method of the classic EM algorithm which dynamically determines the number of clusters. Additionally, we present the main ideas of efficient implementation to make the clustering method feasible on embedded systems as well. As the EM algorithm works with statistical models, the obtained information of the peak extraction step can be efficiently applied, providing a more precise clustering.

As an addition, a method is introduced to align either a peak list containing peak descriptors or consensus peak descriptors against a reference list with potentially previously discovered analytes and their parameters. This method also employs statistical models and statistical optimization methods.

Since all utilized statistical methods and models are rather expensive in terms of computation time and contain almost always the costly exponential function, we introduce an approximate exponential function as substitute. This function has the ability to compute an exponential value up to 4-6 times faster than exact functions of provided standard libraries with only a minimal loss of precision. The exploitation of the binary representation of floating point values within a processor makes this acceleration possible. These features are desirable for the application on embedded systems.

All methods and implementations will be evaluated in detail in terms of computation time, accuracy and reasonability.

Danksagung

In den fast fünf Jahren, die ich an der TU Dortmund als wissenschaftlicher Mitarbeiter gearbeitet habe, konnte ich nicht nur mir ein fundiertes Wissen im Bereich Bioinformatik aneignen, sondern hatte auch die Möglichkeit einen enormen Erfahrungsschatz zu sammeln. Deshalb geht mein größter Dank an meinen Doktorvater Sven Rahmann, der mir die Möglichkeit gegeben und sein Vertrauen in mich gesetzt hat mein Können unter Beweis zu stellen. Ich bin zutiefst dankbar für die Aufgaben und Freiheiten, die ich im Laufe der Zeit bekommen habe, die lehrreichen Diskussionen, die wir geführt haben und die Möglichkeit in einem spannenden und zukunftsträchtigen Projekt mitarbeiten zu können. Auch möchte ich meinem Betreuer Jörg Ingo Baumbach für die produktiven Unterhaltungen, die konstruktiven Kritiken bedanken und die Einführung meiner in das faszinierende und spannende Gebiet der Metabolomik. Der Firma B&S Analytik möchte ich sowohl für das Bereitstellen der vielen Datensätze als auch für die Nutzung ihrer BioScouts herzlich danken.

Ein großer Dank geht auch an alle aktuellen und ehemaligen Kolleginnen und Kollegen meiner Arbeitsgruppe Dr. Daniela Beißer, Dr. Christina Czeschik, Marianna D'Addario, Corinna Ernst, Nina Hesse, André Janowicz, Dr. Johannes Köster, Prof. Tobias Marschall, Dr. Marcel Martin, Christopher Schröder, Henning Timm, Mareike Vogel und Dr. Inken Wohlers für die vielen Momente in und außerhalb des Uni-Umfeldes und die immer ein offenes Ohr für mich hatten. Marianna gilt mein besonderer Dank für zwei Jahre unserer Zusammenarbeit am selben Projekt, die vielen Diskussionen und ihre unzähligen Kommentare, Tipps und Anregungen.

Auch beim Lehrstuhl 11 möchte ich mich ganz herzlich für eine Zeit bedanken, die angenehmer und interessanter nicht sein kann. Mein spezieller Dank gilt hier Gundel Jankord, die sich immer Zeit für mich genommen hat und mir stets bei meinen Verwaltungsbelangen geholfen hat und immer geduldig mit mir war. Die gemeinsamen thematischen und unthematischen Diskussionen mit Dominik Köppl werde ich in guter Erinnerung behalten.

Zu gut weiß ich, wie anstrengend es ist eine Arbeit fehlerfrei in jeglicher Hinsicht zu schreiben. Nicht weniger mühsam ist es auch eine Arbeit Korrektur zu lesen. Deswegen möchte ich mich auch bei allen fleißigen Korrekturlesern Marianna D'Addario, Nina Hesse, Salome Horsch, Johannes Köster, Elias Kuthe und Henning Timm bedanken für das Auffinden der kleinen und großen Rechtschreib-, Grammatik- und Logikfehler sowie die Anregungen und Kritiken.

Zuletzt möchte ich mich noch bei meinen Eltern herzlich bedanken ohne deren Tritzen und Beharrlichkeit ich wohl nie diesen Weg eingeschlagen wäre.

Während meiner Arbeit wurde ich unterstützt vom Sonderforschungsbereich 876 - "Verfügbarkeit von Information durch Analyse unter Ressourcenbeschränkung" im Projekt TB1, siehe `http://sfb876.tu-dortmund.de`.

Dominik Kopczynski

Dortmund, Juli 2016

Contents

Glossary

Chapter comprehensive variables

c	index for component vector
d	drift time in unit of ms
d_{G}	grid opening time in unit of ms
D	list of equidistant drift time points
$\mathcal{D}$	number of dimensions
ϵ	threshold value
$\mathcal{E}$	relative parameter changing
f_{ims}	measurement-dependent proportion factor to convert a drift time into an inverse reduced mobility
g	probability density function for Inverse Gaussian distribution
h^*	height of a peak, i.e., $h^* = S_{r',t'}$ assuming peaks mode at index pair (r', t')
i	index for observation
j	index for cluster vector
J	Jacobian matrix
k	kth iteration
l	index for parameter vector
λ	shape parameter for Inverse Gaussian distribution
m	mode for Inverse Gaussian distribution
μ	mean for Inverse Gaussian distribution
μ'	corrected mean, i.e. $\mu' := \mu + o$ for Inverse Gaussian distribution
μ_{N}	mean of normally distributed background noise
μ_{G}	mean for Gaussian distribution
n	number of observations
$\mathcal{N}$	noise margin
o	offset for Inverse Gaussian distribution
ϕ	FWHM to standard deviation factor in Gaussian distribution
q	number of parameters, i.e. $q := \|\theta\|$
r	index for vectors or matrices according to a certain retention time
ρ	retention time in unit of ms
R	list of equidistant retention time points
R	retention time dimension
σ	standard deviation for Inverse Gaussian distribution
σ_{N}	standard deviation of normally distributed background noise
σ_{G}	standard deviation for Gaussian distribution
S	IMSC signal matrix

S_r	IM spectrum at index r
$S_{\cdot,t}$	chromatogram at index t
t	index for vectors or matrices according to a certain inverse reduced mobility (IRM)
τ	inverse reduced mobility in unit of $\mathrm{V\,s\,cm^{-2}}$
θ	parameter vector
T	list of equidistant inverse reduced mobility points
T	inverse reduced mobility dimension
v	volume in particular unit
w	width
ω	weight vector
W	membership matrix
z	number of components, i.e. $z := \lvert\mathcal{Z}\rvert$
$\mathcal{V}$	velocity in unit of $\mathrm{cm\,ms^{-1}}$
$\mathcal{Z}$	component vector

Variables for Chapter 1

E	field strength in unit of $\mathrm{V\,cm^{-1}}$
l	length in unit of cm
K	mobility in unit of $\mathrm{cm^2\,V^{-1}\,s^{-1}}$
$\mathcal{P}$	pressure in unit of bar
$\mathcal{P}_0$	absolute pressure for normalization
U	voltage in unit of V

Variables for Chapter 2

b	exponent with base 2
η	mantissa
$\mathcal{M}$	fourth-order polynomial
s	sign bit

Variables for Chapter 3

A	averaged IMSC signal matrix
$\mathcal{A}$	sorted integer list
B	Baseline spectrum
$\mathbf{D}$	diffusion coefficient
FN	false negative
FP	false positive
$\mathbf{G}$	gap for global alignment
G	Fowlkes-Mallows index
H	histogram
I	minimal signal intensity

$\mathcal{K}_{\mathrm{B}}$	Boltzmann constant
κ_{R}	expansion size in retention time (index unit)
κ_{T}	expansion size in IRM (index unit)
L_{D}	entropy of the data
L_{M}	entropy of the model
L_{U}	entropy of the uniform distribution
$\mathbf{P}_{\theta}$	function for peak surface with given parameter set θ
PPV	positive predictive value
$\mathcal{Q}$	electric charge
Q	relative size of entropy
SENS	sensitivity
V	derivative matrix
TP	true positive

Variables for Chapter 4

B	quadratic polynomial
$\mathbf{E}$	error function
γ	error threshold
$\mathcal{L}$	Huber loss function
P	set of model parameters of peaks for an IM spectrum
$\mathbb{P}$	chain of one-dimensional Inverse Gaussian models
$\mathcal{R}$	residual function
S_{R}	RIP-only spectrum
$\mathcal{S}$	single IM spectrum at a certain retention time
Z	scoring matrix for aligning two sets P, P^{+}
ζ	log-odds score for computing scoring matrix Z

Variables for Chapter 5

$\mathcal{C}$	sorted cluster list
$\mathbf{D}$	data table of n measurements $\times k$ consensus peaks

Variables for Chapter 6

$\mathcal{R}$	source point set
s_{R}	scale parameter in retention time
s_{T}	offset parameter in IRM dimension
$\mathcal{T}$	target point set

1 Introduction

An Ion mobility (IM) spectrometer (IMS) coupled with a multi-capillary column (MCC), MCC/IMS for short, is gaining importance for biotechnological and medical applications. With MCC/IM spectrometers, one can measure the presence and concentration of volatile organic compounds (VOCs) in the air or in exhaled breath with high sensitivity. In contrast to other technologies, such as mass spectrometry coupled with gas chromatography (GC/MS), an MCC/IMS works at ambient pressure and temperature. Several diseases can potentially be diagnosed at an early stage with MCC/IM spectrometry technology. These diseases include chronic obstructive pulmonary disease (described by Westhoff et al. (2010), Koczulla et al. (2011) and Bessa et al. (2011)), sarcoidosis as introduced by Bunkowski et al. (2009) or lung cancer as investigated by Westhoff et al. (2009) and Darwiche et al. (2011). The identification of bacteria was investigated by Jünger et al. (2012) and especially the potential of VOCs as biomarkers of patients suffering from an infection with Pseudomonas aeruginosa explored by Maddula et al. (2011). Keller et al. (1999) also used the MCC/IM spectrometers for the detection of drugs as well as Ewing et al. (2001) employed it to detect explosives. More advances in breath analysis are presented by Fink et al. (2014). The constant monitoring of VOC levels is of interest in biotechnology, e.g., for watching fermenters with yeast producing desired compounds as investigated by Kolehmainen et al. (2003) and Halbfeld et al. (2014). Monitoring also can be applied in medicine, e.g., monitoring propofol levels in the exhaled breath of patients during surgery as described by Kreuder et al. (2011) and Buchinger et al. (2013), monitoring sevoflurane levels described by Kunze et al. (2015) or monitoring the level of Neutrophil granulocyte in blood (detecting neutropenia) proved in clinical studies by Furtwängler et al. (2014). Furthermore, this technology has the potential to infer information by comparison with other analytical technologies as GC/MS as reported by Jünger et al. (2010) and Maurer et al. (2014).

IMS technology is moving towards miniaturization and small mobile devices. Research was done in this field by Teepe et al. (2001), Salleras et al. (2006), Zimmermann and Barth (2007), and Aguilera-Herrador et al. (2008). Nowadays, the hardware even reaches dimensions of mobile phones. This creates new challenges for data analysis: The analysis should be possible *within* the measuring device without the necessity of additional hardware like an external laptop or a compute server. The application of these mobile detectors can be e.g. searching for drugs and explosives in public areas with few or no security control stations, for instance central railway stations or sports stadiums.

Another possible scenario is a mobile device which monitors the health state of patients and can even be taken home. For these devices it is indispensable to detect a critical change of health states as early and reliably as possible. Thus, an analysis of the data must happen immediately at the device during the monitoring. To save as much energy as possible for the detector, the data should be processed on a small embedded chip or small device like a Raspberry Pi[1] or similar hardware with restricted resources. Algorithms in small mobile hardware face constraints, such as the need to use little energy (hence little random access memory), while maintaining prescribed time constraints.

1.1 Objectives

The focus of this thesis is on the automatic analysis of MCC/IM spectrometry data, especially in a resource-constrained context. Regions within these spectra with high signal intensity are called peaks and are of increased importance. These peaks hint at the contained analytes within a sample and can be potential biomarkers. Given a measurement in form of a two-dimensional signal matrix S and a peak description θ, a peak extraction aims to detect peaks within the matrix and reduces the information to its descriptive parameters per peak. Here, we distinguish between an offline and an online peak extraction. Given a signal matrix S, an offline peak extraction is allowed to access the whole matrix at any time during the complete analysis process without any restriction. Methods for offline peak extraction especially for MCC/IM spectrometry measurements have already been introduced, e.g. a k-means approach by Bader et al. (2005), a watershed method described by Bunkowski (2011) or a neighborhood region comparison searching for local maxima by D'Addario et al. (2014). More information about these methods is given in Section 3.2.

All these methods have in common that the complete signal matrix has to be available during the extraction. That makes an application for mobile detectors rather unsuitable, as they have constrained resources like limited memory or low processor power. On the other hand, an online peak extraction is allowed to access the data only sequentially and discard the raw data immediately after analysis. Since detectors like the MCC/IM spectrometry device provide chunks of the data at certain time intervals, the online peak extraction should also be able to analyze these chunks completely within the time range. Only few methods have been introduced for this topic, i.e. peak detection by slope analysis or Savitzky-Golay Laplace-operator filter thresholding regions by Egorov et al. (2013) (consider Section 4.1). Several descriptions of a peak within MCC/IM spectrometry measurements have been introduced. Bödeker and Baumbach (2009) described the theoretical surface of a peak using a product of two lognormal distributions (each for one dimension). This

[1] `https://www.raspberrypi.org/`

description requires nine parameters. Vogtland and Baumbach (2009) introduced another model, describing the surface by splitting the cross section of a peak in both dimensions into two parts. Here, the first part describes the peak fronting with a Gaussian distribution up to its mode whereas the tailing is described by either a Breit-Wigner or a log-Breit-Wigner distribution. Eight parameters are necessary to describe a peak. A detailed description of both models is given in Section 3.2. Both models have in common that the description of the surface makes it possible to determine the volume of a peak. But due to their rather complex formulas, the estimation of appropriate parameters becomes more difficult which results in longer computation time. In this thesis, we contribute to all three mentioned issues. First, we define a statistical model describing the surface of the peaks with only seven parameters. Accordingly, we introduce one offline and one online peak extracting method. The novelty of both methods is the provision of a peak list containing the seven parameters per peak. Other methods only extract the position and the highest intensity of the peaks. We also show that the online peak extraction keeps up with time restrictions even on devices with low processing power.

The second focus of the analysis contributes right after a peak extraction. For the exploration and characterization of analytes as potential markers for certain conditions (e.g. detection of diseases using human breath), time lines or data sets from the same experiment have to be aggregated and peaks occurring in several measurements caused by the same analyte have to be identified and clustered. We are given a set of observations (extracted peaks from different measurements) and a suggested parameterized probability density function $\mathbf{P}$ with parameter set θ for observations within a cluster. The clustering objective is to find a partition of the observations in a dynamically adjusting set of clusters and estimate the parameters for every θ of all clusters. Here, the likelihood is maximized that the observations are generated by all probability density P using the parameter sets θ. Previous work only focused on clustering with an unknown number of clusters like density-based spatial clustering of applications with noise introduced by Ester et al. (1996) or cluster editing described by Rahmann et al. (2007) and Böcker et al. (2011). Other methods only considered a clustering using a probability density function for the observations with a fixed and pre-defined number of clusters, e.g. the expectation-maximization algorithm introduced by Dempster et al. (1977). More details are given in Sections 1.5.3 and 5.2. We introduce a novel approach which takes advantage of applying a probability density function for observations as well as dynamically adjusts the number of clusters. Additionally, we develop the method into a tool capable of clustering tens of thousands of observations within a few seconds on devices with low processor power as well. Optionally, peak lists containing either single measurements or consensus peaks can be compared with a reference list including already known compounds (i.e. a peak list of already labeled peaks) and their parameters to infer their corresponding analytes. Given a set of two-dimensional reference observations and a set of target observations of the same dimension and a probability density function **P**, we search for a transformation set mapping the targets to the references. The likelihood is maximized that the targets are described by the references after transformation using **P**. This problem is referred to as point set registration and can be solved using the coherent point drift algorithm introduced by Myronenko and Song (2010). We describe an adaption of this method tailored for our application.

1.2 Outline

After describing the technology of MCC/IM spectrometers followed by a set of fundamental optimization methods within this section, an approximate function computing $\exp(x)$ is introduced in Section 2 which is up to six times faster than conventional builtin functions of the standard libraries. Section 3 illustrates a proposed statistical model describing the shape of two-dimensional peaks and introduces an offline method for extracting peaks and estimating parameters for this model. Another method for peak extraction utilizing an online approach is described in Section 4. Both methods provide a so-called peak list containing all important information of the peaks. A necessary step for continuing comparisons between peak lists is a clustering independent of a predefined cluster size. A new approach for an adaptive EM clustering is described in Section 5. The output of the clustering is a so-called consensus peak list. Having a peak list from a single measurement or a consensus peak list of a set of measurements provided by the clustering with yet unknown peaks, it is important to determine the compounds (analytes) the peaks are provided by. Such a peak list is referred to as a target peak list. Given a reference peak list with already labeled peaks, Section 6 illustrates a method to label peaks within the target peak list by aligning them to the reference peak list. Although both these problems of clustering and aligning appear to be similar, they need to be considered separately, as we assume during the clustering that all single peak lists originate from the same device, but can not claim this for the aligning as well. Caused by several features of the measuring devices, peaks can shift systematically in a measurement. The evaluation is split into several parts. Single method evaluations are described within their sections, whereas comparative evaluations are given in Section 10. Section 11 concludes and discusses the thesis.

All performance benchmarks were executed on two different platforms, (1) a desktop PC with Intel(R) Core(TM) i5 2.80GHz CPU, 8GB memory, with Ubuntu 12.04 (64bit) OS and (2) a Raspberry Pi[2] type B with ARM1176JZF-S 700MHz CPU, 512 MB memory, Raspbian Wheezy (32bit) OS. The Raspberry Pi was chosen because it is a complete credit card-sized low-cost single-board computer with weak CPU and low power consumption (3.5 W) which is suitable for embedded and mobile analysis measurement devices.

[2]`http://www.raspberrypi.org/`

1.3 MCC/IMS

For this work, a BioScout IMS (B&S Analytik, Dortmund, Germany) with a ^{63}Ni ß-ionization source is used which couples a multi-capillary column with an ion mobility spectrometer.

Ion Mobility Spectrometer

IM spectrometers separate molecules by their properties like mass, polarizability and structure. A classic IMS is divided into two parts: an ionization chamber and a drift tube. In contrast to other analysis devices like mass-spectrometry (MS), IM spectrometers do not need a high vacuum. Since the devices work with ambient pressure, construction and maintenance are cheaper. Neutral analyte compounds are injected by either nitrogen or synthetic air as carrier gas into the ionization chamber and ionized by radioactive nickel as an ionization source. Here, no additional power supply is needed. Negatively as well as positively charged ions can be produced. In positive mode, reactant ions are captured emerging from the carrier gas. Since in typical measurement processes the amount of carrier gas exceeds the amount of all remaining analytes, a reactant ion peak is resulting as the highest peak within the spectra. It also provides an upper bound for the number of ionizations, since the ionization source emits constantly. Hence, comparing two spectra their integral should be almost equal. By collisions with the molecules, the reactant ions generate product ions. An ion shutter that separates the ionization chamber from the drift tubes opens periodically. Around the drift tube, drift rings are mounted in equidistant intervals providing an electric field which forces the product ions to drift through the tube towards a Faraday plate mounted at the opposite end of the tube. Additionally, drift gas is injected with an adjustable gas flow from the opposite direction to function as an obstacle. By collisions with the drift gas and acceleration through the electric field, a separation of the product ions takes place. Finally, the ions hit the Faraday plate and transfer their charge, causing a low current which can be measured after amplification. Caused by the motion, not all ions of the same analyte hit the plate at the same time but appear at continuous drift time with different concentrations, providing the typical bell-shaped peaks within the spectrum with a certain characteristic mean drift time for any analytes. A schematic setup of an IMS is illustrated in Figure 1.1. After the collision with the plate, the ions become neutral again. Finally, the molecules are pushed out of the IMS by the drift gas. In general, these IM spectrometers provide a high resolution measured in parts per billion by volume.

The measurement process takes about 100 milliseconds and the signal intensity (voltage change) is captured at each point in time with 250 kHz. Due to the analog/digital converter (12 bit for BioScout), the signals obtain discrete values. Since the BioScout works with both positive and negative mode, the

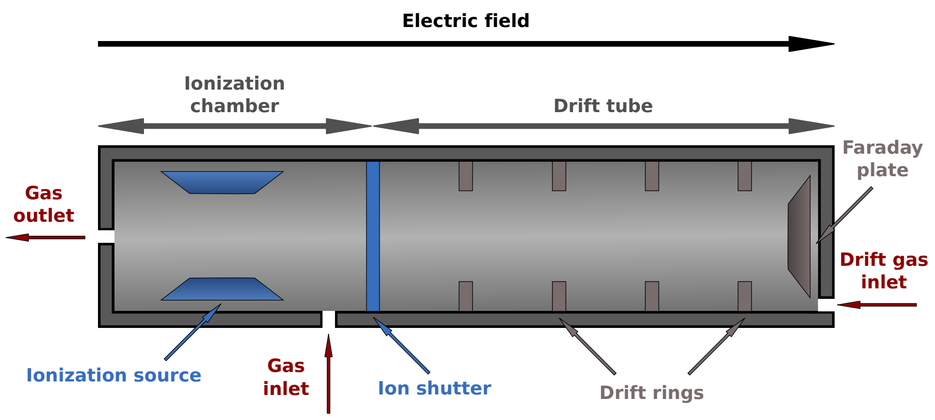


Figure 1.1: Schematic cross section of an IM spectrometer presented by D'Addario et al. (2014). Analyte compounds are ionized after injection into the ionization chamber. The ions are accelerated by an electric field and move through the drift tube. They cause a voltage change when colliding with the Faraday plate; this is the measured signal.

digital values range between 0 (full deflection in positive mode), 2048 (no signal) and 4095 (full deflection in negative mode). After amplification, a full deflection corresponds to either +5 V or −5 V, depending on the mode.

For a comparable measurement independent of external conditions (temperature, pressure), the drift times are converted into reduced inverse mobilities. Using the electric field strength $E = U/l$ with voltage U and the characteristic drift velocity $\mathcal{V} = l/d$ with drift tube length l in centimeter and drift time d in seconds, the mobility K can be computed by the relation $\mathcal{V} = K \cdot E$. To be independent of temperature $\mathcal{T}$ and pressure $\mathcal{P}$ within the tube, the mobility is reduced (i.e. normalized) by absolute pressure $\mathcal{P}_0 = 1.013\,25$ bar and temperature $\mathcal{T}_0 = 293.15$ K as described by Baumbach and Eiceman (1999). According to the ideal gas law $\mathcal{P}\, v = \mathcal{N}_\mathrm{a}\, \mathcal{R}_c\, \mathcal{T}$ (with volume v, amount of substance of gas $\mathcal{N}_\mathrm{a}$, universal gas constant $\mathcal{R}_c$), the reduced mobility K_0 is

$$K_0 = \frac{\mathcal{T}_0\, \mathcal{P}}{\mathcal{T}\, \mathcal{P}_0} \cdot \frac{l^2}{U\, d}. \tag{1.1}$$

For convenience, the inverse reduced mobility (IRM) $\tau = 1/K_0$ in units of V s cm^{-2} is used with the advantage of being proportional to the drift time. Let T be the set of (equidistant) IRMs where a measurement is made and D be the corresponding set of equidistant drift time points (each $1/250\,000$ sec-

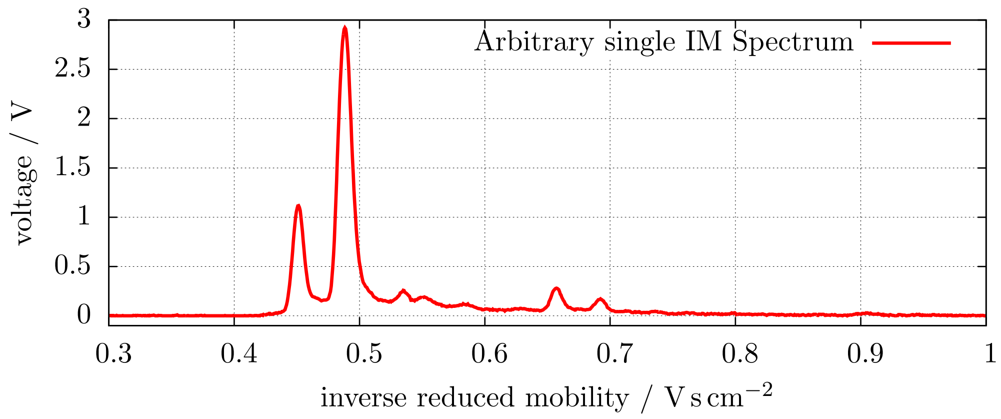


Figure 1.2: An exemplary single IM spectrum containing the reactant ion peak (RIP) with mode at 0.48 V s cm^{-2} and at least two peaks caused by analytes (mode at 0.66 and 0.69 V s cm^{-2}).

ond for 50 ms, i.e. 12 500 time points). Since almost all measures are constant within a measurement process, there exists a constant proportional factor

$$f_{\text{ims}} = \frac{\mathcal{T}\,\mathcal{P}_0}{\mathcal{T}_0\,\mathcal{P}} \cdot \frac{U}{l^2} > 0$$

such that $\tau = f_{\text{ims}} \cdot d$. An exemplary single IM spectrum is illustrated in Figure 1.2.

Pre-separation

It is impossible to distinguish different compounds with identical mean drift time, since the width of both peaks is identical and thus an overlay of both yields one high peak only. Especially in complex biological samples like human breath it is inevitable that these samples consist of several different compounds. Therefore the IMS is coupled with a multi-capillary column (MCC) which separates the compounds before they enter the IMS. Both MCC and a gas chromatograph (GC) have in common that the analytes are pushed by a carrier gas with an adjustable flow through a column coated with a gel from the inside. The state of molecules sticking to the gel is referred to as stationary phase whereas the state of molecules flowing through the column is called mobile phase. In contrast to GCs, the MCC is constructed of several dozen to several hundred columns, e.g. the BioScout uses an MCC with 1200 columns. Therefore, the separation is faster in an MCC than in a GC. Hence a complete measurement process takes only a few minutes at most for an MCC in comparison to a GC which can take about an hour for the same separation. According to the strength of molecules interacting with the matrix of the MCC, different compounds are retained for different periods of time in the MCC. The time the molecules need to pass the MCC is called retention time.

Table 1.1: Parameters of the BioScout that were used for almost all captured measurements for evaluation. Differing adjustments are marked.

Parameter	*Value*
MCC	
Column type	OV5
Column diameter	0.04 mm
Temperature $\mathcal{T}$	40 °C
column length l	20 cm
Column number	1200
Carrier gas	synthetic air
Carrier gas flow	150 mL min^{-1}
Sample Loop	
Duration	10 s
Volume	10 mL

Parameter	*Value*
IMS	
Ionization source	^{63}Ni
Drift gas	synthetic air
Drift gas flow	100 mL min^{-1}
Voltage U	4380 V
Length l	12 cm
Grid opening time d_{G}	0.3 ms
Capturing period	100 ms
Polarity	positive

Coupling both devices

Before the analytes are injected into the MCC, a sampling captures the air to be analyzed for an adjustable time period. Afterwards, a certain concentration of the sample is injected into the MCC. After a specific retention time ρ, several molecules of a compound pass the MCC and are directly injected into the IMS device. Thus an IM spectrometry measurement is executed periodically. For a sufficient measurement, the BioScout captures about $10 - 12$ minutes (assuming a certain set of adjustments listed in the following paragraph) to ensure that all volatile organic compounds pass the column.

Device parameters

As mentioned, for this thesis the MCC/IMS BioScout was used as stated by Bödeker et al. (2008). Table 1.1 summarizes all relevant parameters and properties. If not stated different, these parameters hold for all measurement captures for the evaluation sections within the following chapters.

1.4 Data and Peaks

Let R be the finite set of retention time points at which an IM spectrometry measurement is taken, and let T be the finite set of measured inverse reduced mobility (IRM) points. We obtain a two-dimensional spectrum $S = (S_{r,t})$ indexed by $r \leq |R|$ and IRMs $t \leq |T|$ at retention time R_r and IRM T_t. It can be visualized as a two-dimensional heat map (Fig. 1.3). For convenience, a column $S_{\cdot,t}$ with fixed IRM index t is called a *chromatogram*, a row S_r

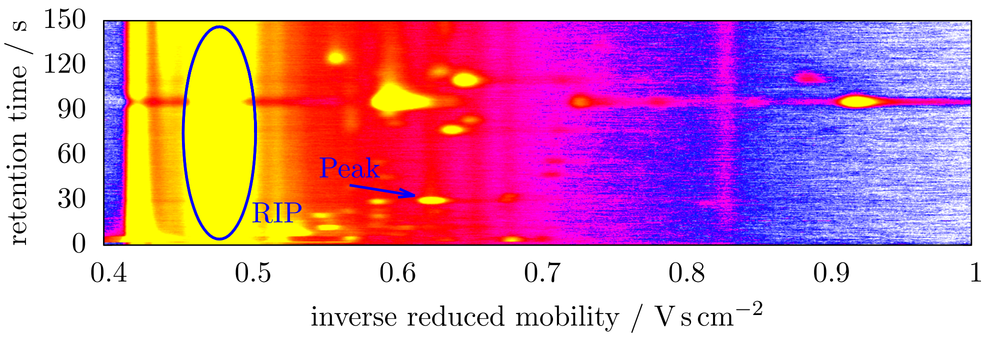


Figure 1.3: Visualization of a raw measurement (IMSC) as a heat map as presented by Kopczynski and Rahmann (2014); signal color: white (lowest) < blue < purple < red < yellow (highest). The constantly present reactant ion peak (RIP) with mode at $0.48\,\mathrm{V\,s\,cm^{-2}}$ and one exemplary VOC peak are annotated.

with fixed retention time index is referred to as an *IM spectrum*, the whole matrix S is called an *IM spectrometry chromatogram (IMSC)* and a cell of the matrix $S_{r,t}$ with both fixed retention time and IRM is called a *signal intensity*. In practice it is common not to analyze the full spectra, but aggregated ones, in which a set of consecutive values within an IM spectrum and a set of consecutive IM spectra are averaged. It is convenient to aggregate with a 5×5 non-overlapping field in which the mean of 25 values is used instead. The consequences are a decrease of the background noise level due to the "smoothing" as well as a decrease of the number of values. Since the peaks have a certain minimal width in both dimensions that are provided by the physics of the device, every peak is wider than the aggregation kernel so that none of them disappears by the smoothing.

Regions with high signal intensities within an IMSC are denoted as *peaks*. The retention time and IRM of a peak discriminate a certain compound which generates the peak and the intensity of a peak infers the concentration. The presence or absence of peaks as well as their intensity can determine biomarkers which may hint at certain diseases. The width in both retention time and IRM is proportional with the corresponding dimensions. In the IRM dimension, the peaks are almost normally distributed whereas the full width at half maximum (FWHM) depends on the grid opening time d_{G} and increases negligibly. The computation of FWHM in the IRM dimension is described in Section 3.3. We denote the standard deviation in the IRM dimension as $\Delta\tau$ in unit of $\mathrm{V\,s\,cm^{-2}}$ and in retention time $\Delta\rho$ in unit of s, respectively. Furthermore, the peaks produce stronger tailings with increasing retention time.

Challenges of IMSC Analyses

Several properties of the described measuring technique provide characteristic features which makes automated analyses difficult. Due to the little current flow within the IMS (ranges in pA), the amplification causes an additive background noise overlaying the entire spectrum. The reactant ion peak appears in every IM spectrum and does not provide any important information about the desired peaks. Since the ions are separated by collisions with the drift gas, they do not cause a single signal at only one discrete point in time (as they do in mass spectrometry), but hit the Faraday plate during a time interval causing a log-concave signal over time. Measuring in either positive or negative mode provides different measurements in which the peak amounts and locations differ. When the amount of ions of a certain analyte exceeds a certain threshold, dimers and even polymers are formed, causing narrowed monomers. This narrowing occurs because the quantity of ionizations is fixed and thus the peaks in an IM spectrum only offer information about the relative concentration of an analyte instead of an absolute concentration. Furthermore, a so-called tailing function is caused by the device, appears in every IM spectrum and needs to be handled; for details consider Bader et al. (2008). The high number of data points requires fast analysis methods. For example, in breath gas analysis a data acquisition takes about ten minutes with ten IM spectra per second and 12 500 data points per IM spectrum which results in $6\,000 \times 12\,500 = 75\,000\,000$ data points. Finally, since the location of peaks caused by the same analyte differs slightly between individual measurements, robust methods have to cope with this feature.

Our focus within this thesis is to detect peaks and not analytes. We concentrate on measurements in positive mode in which we extract peaks (i.e. reduce a measurement into a set of parameters describing the detected peaks), including preprocessing to clean the measurements. The detection of dimers and additional polymers and their connection and the inferring of the analytes concentrations is not within the scope of the research project. The main objective is to develop robust methods for the complete processing pipeline which can run sufficiently on embedded systems.

1.5 Optimization Methods

The algorithms in the following sections utilize several unconstrained minimizing methods, i.e. the least square methods and expectation-maximization algorithm. Especially the methods in Sections 3 and 4 make extensive use of these optimization methods. These algorithms are partially used as introduced in the literature. For some cases, the algorithms are adopted to solve particular problems. In the following, the original methods are summarized briefly. The adoptions are described in the sections in which they are utilized.

1.5.1 Linear Least Square Method

We are given a set of input variables $X = (X_{i,l})$, a set of observations $y = (y_i)$ with $i = 1, \dots, n$ and $l = 1, \dots, q$ and a linear relationship between input variables and observations $y = X\theta$, where θ is a q-dimensional vector of unknown parameters. To infer these parameters, we solve the following optimization problem:

$$\min_{\theta} ||X\theta - y||_2^2.$$

After expansion, we obtain $\min_\theta(\theta^{\mathrm{T}} X^{\mathrm{T}} X\theta - \theta^{\mathrm{T}} X^{\mathrm{T}} y - y^{\mathrm{T}} X\theta + y^{\mathrm{T}} y)$, where $^{\mathrm{T}}$ denotes the transposed vector or matrix, respectively. Since $\theta^{\mathrm{T}} X^{\mathrm{T}} y$ and $y^{\mathrm{T}} X\theta$ are scalar values, it holds $\theta^{\mathrm{T}} X^{\mathrm{T}} y = y^{\mathrm{T}} X\theta$. After derivation with respect to θ and setting to zero, we get $0 = 2X^{\mathrm{T}} X\theta - 2X^{\mathrm{T}} y$. Solving for θ, we obtain the final equation

$$\widehat{\theta} = (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} y \tag{1.2}$$

which is referred to as the normal equation. A detailed description is given by (Björck, 1996, Chapter 1).

Polynomial Regression A use case for linear least squares is the polynomial regression. Having a polynomial of degree q with $y_i = \theta_q x_i^q + \cdots + \theta_0 x_i^0$, we get the input set

$$X = \begin{bmatrix} \dfrac{\partial f(x_1)}{\partial \theta_0} & \cdots & \dfrac{\partial f(x_1)}{\partial \theta_q} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f(x_n)}{\partial \theta_0} & \cdots & \dfrac{\partial f(x_n)}{\partial \theta_q} \end{bmatrix} = \begin{bmatrix} x_1^q & x_1^{q-1} & \cdots & x_1 & 1 \\ \vdots & & \ddots & & \vdots \\ x_n^q & x_n^{q-1} & \cdots & x_n & 1 \end{bmatrix}$$

and obtain the coefficient vector θ by solving Equation (1.2).

1.5.2 Non-linear Least Square Method

The non-linear least square (NLLS) method is an iterative method to estimate parameters $\theta = (\theta_1, \dots, \theta_q)$ of a supposed parametric function f having a set of n observed data points $(x_1, y_1), \dots, (x_n, y_n)$ with $i = 1, \dots, n$. The idea is to minimize the quadratic error between the function and the observed data, thus $\min \sum_i^n e_i^2$ for all residual errors $e_i = y_i - f(x_i \,|\, \theta)$. Taking the first derivative with respect to every parameter and setting it to zero, we obtain $2 \sum_i^n e_i \cdot \partial e_i / \partial \theta_l = 0$ for all $l = 1, \dots, q$. Since the gradient functions do not have a closed formula, the parameters must be approximated by choosing starting parameters and improving them in every iteration, thus $\theta_l \approx \theta_l^{k+1} = \theta_l^k + \Delta\theta$ where k denotes the kth iteration and $\Delta\theta$ is a shift operator. Additionally, a Jacobian matrix $J = (J_{i,l})$ is set up, which is defined as follows: $J_{i,l} := -\partial e_i / \partial \theta_l$. To obtain $\Delta\theta$, the first order Tailor-series expansion is used, thus:

$$f(x_i \,|\, \theta) \approx f(x_i \,|\, \theta^k) + \sum_l^q \frac{\partial f(x_i \,|\, \theta_l^k)}{\partial \theta_l^k}(\theta_l - \theta_l^k) = f(x_i \,|\, \theta^k) + \sum_l^q J_{i,l}\Delta\theta_l.$$

Substituting $\Delta y_i = y_i - f(x_i \,|\, \theta^k)$ and setting all into the gradient functions, we obtain

$$-2\sum_i^n J_{i,l}(\Delta y_i - \sum_{l'}^q J_{i,l'}\Delta\theta_{l'}) = 0$$

for all $l \leq q$. Rearranging and writing in matrix notation leads to $(J^{\mathrm{T}}J)\Delta\theta = J^{\mathrm{T}}\Delta y$. Having a function in which the derivative with respect to a parameter depends on the remaining parameters, the following steps must be repeated until convergence is reached:

$$\begin{aligned}\Delta y &= y - f(x \,|\, \theta^k),\\ \Delta\theta &= (J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\Delta y,\\ \theta^{k+1} &= \theta^k + \Delta\theta.\end{aligned}$$

Details and different algorithms for NLLS can be found in the literature (Nocedal and Wright, 2006, Chapter 10).

Non-Linear Loss Minimization

In some cases it is not important, whether some observed values y_i exceed the function $f(x_i \,|\, \theta)$ significantly, as long as the residual errors for the remaining values are acceptably low. Baseline fitting is a reasonable use case for that scenario. Mazet et al. (2004) propose an approach based on NLLS using a non-symmetric error function. If the residual error $(y_i - f(x_i \,|\, \theta))^2$ exceeds a given threshold γ, γ is used instead of the error. Another scenario for using a modified error function is curve fitting, in which both positive and negative outliers are ignored. Here, the error function is truncated in both directions. However, since the non-symmetric error function was designed to fit baselines using polynomial functions, we introduce another error function which is more appropriate for statistical distributions. In Chapter 4, we describe both symmetric and non-symmetric error functions and how these are applied for non-linear loss minimization.

1.5.3 The EM Algorithm for Mixture Models with Heterogeneous Components

In the subsequent sections, variations of the EM algorithm are utilized as introduced by Dempster et al. (1977) for mixture model deconvolution. Here, we summarize the algorithm and describe the E-step common for all variants. In each of the following sections, we describe the specific model, the initial values for the parameters, and the specific maximum likelihood parameter estimators of the M-step. A variation of the EM algorithm is described in Section 5.

A fundamental idea of the EM algorithm is that the observed data x with $n = |x|$ data points is viewed as a sample of a mixture of probability distributions

$$f(x \,|\, \theta) = \sum_{c}^{z} \omega_c \, f_c(x \,|\, \theta_c),$$

where c indexes the z different component distributions f_c, where θ_c denotes all parameters of distribution f_c, and $\theta_c = (\theta_{c,1}, \ldots, \theta_{c,q})$ is the collection of all parameters. The mixture coefficients ω_c satisfy $\omega_c \geq 0$ for all c, and $\sum_c^z \omega_c = 1$.

We point out that, in contrast to most applications, in our case the probability distributions f_c are of different types, e.g., a uniform and a Gaussian one.

The goal of mixture model analysis is to estimate the mixture coefficients $\omega = (\omega_c)$ and the individual model parameters $\theta = (\theta_c)$, whose number and interpretation depends on the parametric distribution f_c.

Since the resulting maximum likelihood parameter estimation problem is non-convex, iterative locally optimizing methods such as the Expectation Maximization (EM) algorithm are frequently used. The EM algorithm consists of two repeated steps: The E-step (expectation) first estimates the expected membership of each data point in each component and then the component weights ω, given the current model parameters θ. The M-step (maximization) estimates maximum likelihood parameters θ_c for each parametric component f_c individually, using the expected memberships as hidden variables which decouple the model. As the EM algorithm converges towards a local optimum of the likelihood function, it is crucial to choose reasonable starting parameters for θ.

E-Step

The E-step is independent of the specific component distribution types and always proceeds in the same way. To estimate the expected membership $W_{i,c}$ of data point i in each component c, the component's relative probability at that data point is computed, i.e.

$$W_{i,c} = \frac{\omega_c \, f_c(x_i \,|\, \theta_c)}{\sum_{c'}^{z} \omega_{c'} \, f_{c'}(x_i \,|\, \theta_{c'})}, \tag{1.3}$$

such that $\sum_c^z W_{i,c} = 1$ for all $i = 1, \ldots, n$. Then the new component weight estimates ω_c^+ are the averages of $W_{i,c}$ across all data points

$$\omega_c^+ = \frac{1}{n} \sum_i^n W_{i,c}, \tag{1.4}$$

where n is the number of data points.

Convergence

After each M-step of an EM cycle, we compare $\theta_{c,l}$ (old parameter value) and $\theta_{c,l}^+$ (updated parameter value), where l indexes the elements of θ_c, the parameters of component c. Let

$$\mathcal{E}(a,b) := \begin{cases} 0 & \text{if } a = b = 0, \\ \frac{|a-b|}{\max(|a|,|b|)} & \text{else} \end{cases} \tag{1.5}$$

be the relative error. We say that the algorithm has converged when $\mathcal{E}(\theta_{c,l}^+, \theta_{c,l})$ drops below the threshold ε for all $c = 1, \ldots, z$ and $l = 1, \ldots, q$. Empirically, we determined a reasonable threshold $\varepsilon := 0.001$, corresponding to 0.1% precision. For convenience, when not mentioned otherwise, this threshold will be taken for the convergence tests within all methods described in the following. Since the threshold will be an adjustable parameter within the following methods, we denote ε as `convergence_threshold`.

2 Approximating the Exponential Function

The exponential function, or more specifically, the natural exponential function is defined as $x \mapsto \mathrm{e}^x$ with $\mathrm{e} = 2.718\,281\,828\ldots$ (which is the Euler number) as base and is commonly abbreviated with $x \mapsto \exp(x)$. The function is widely used within mathematics and especially within statistics. Several distribution functions like the Gaussian distribution, Inverse Gaussian distribution or Poisson distribution utilize the exponential function.

2.1 Background

The following methods which are described in the Chapters 3, 4, 5 and 6 utilize both the Gaussian distribution as well as the Inverse Gaussian distribution within their models. The offline as well as the online peak estimation method execute the exp function at least $|R| \cdot |T|$ times, analyzing an IMSC with dimension $|R| \times |T|$, whereas the clustering executes the function overall $n^2 \cdot \iota$ time, where n is the number of data points to be clustered and ι the number of iterations. For minor measurements, the exponential function is executed several thousand up to millions of times and the number increases accordingly for the measurements with high resolution. All generic math libraries provide an exact exp function. Using the knowledge about the representation of floating points within processors, it is possible to build a function approximating exp which accelerates the computation up to $4 - 6$ times with hardly and loss of precision.

2.2 Related Work

In the following, we summarize three approximation methods already known from literature, namely fastexp and fasterexp from the fastapprox-v0.3.2-library and EXP_makro.

```
static inline float fastexp (float p){
    p *= 1.442695040f;
    float offset = (p < 0) ? 1.0f : 0.0f;
    float clipp = (p < -126) ? -126.0f : p;
    int w = clipp;
    float z = clipp - w + offset;

    union {int i; float f;} v = {int((1 << 23) \
    * (clipp + 121.2740575f + 27.728023f / (4.8425256f - \
    z) - 1.49012907f * z))};
    return v.f;
}

static inline float fasterexp (float p){
    p *= 1.442695040f;
    float clipp = (p < -126) ? -126.0f : p;
    union {int i; float f;} v = {int ((1 << 23) \
    * (clipp + 126.94269504f))};
    return v.f;
}
```

Listing 2.1: C++ code of fastexp and fasterexp approximation provided by fastapprox-v0.3.2-library. The code was slightly adapted for better legibility.

The fastapprox-v0.3.2-library

The *fastapprox-v0.3.2*-library[1], implemented and maintained by Paul Mineiro, provides two methods: the fastexp and the fasterexp function. Both methods exploit the binary representation of floating points and omit the expensive computation of the exponential value by approximating its binary representation with atomic arithmetic operations. Given a negative value p, the method fastexp computes $\exp(p)$ using an hyperbolic fitting function, whereas fasterexp omits some arithmetic operations employing a constant value to fit the integer representation. The code is shown in Listing 2.1. Due to the limited number of operations, fasterexp is faster but provides results with a higher mean and maximum error.

The EXP function

Introduced by Schraudolph (1999), the EXP function also exploits the representation of floating points within a processor. By interpreting a float bit field as an integer bit field, the complete computation can be reduced to few integer operations. Listing 2.2 presents the complete code.

[1] `https://code.google.com/p/fastapprox/`

```
static union { float f; int i; } eco;
#define EXP_A (8388608 / M_LN2)
#define EXP_C 60801
#define EXP(y) (eco.i = EXP_A * (y) + \
    (1065353216 - EXP_C), eco.f)
```

Listing 2.2: C++ code of EXP function. For better comparability, the code was adapted to float type (the original EXP function was coded for double type).

2.3 Approximation Approach

As mentioned, the Gaussian distribution and the Inverse Gaussian distribution both utilize the exp function. By the definition of both distributions, the exponent is always negative or zero. Thus, the exp approximation will be tailored for this domain. The natural exponential function can be rewritten into another exponential function with any other base, for example $e^x = 2^{x/\log(2)}$. Thus, the property can be exploited that the floating points are represented as $s \cdot 2^b \cdot \eta$ as defined in the IEEE-754-1985 standard by the IEEE Computer Society (1985), where $s \in \{-1, 1\}$ is the sign, $b \in \mathbb{N}_0$ the exponent with an offset of 127 and $\eta \in [1; 2)$ the normalized mantissa. Now the result can be expressed as the product of two floats, namely let $\exp(x) := f_1 \cdot f_2$, where $f_1 := s_1 \cdot 2^{b_1} \cdot \eta_1$ with $s_1 := 1, \eta_1 := 1$ and $b := \lceil x/\log(2) \rceil$. Furthermore, let $f_2 := 2^{(x/\log(2)) - \lceil x/\log(2) \rceil}$. Here, f_2 ranges between $(0.5; 1]$ and $-1 < (x/\log(2)) - \lceil x/\log(2) \rceil \leq 0$. Let

$$\mathcal{M}^*(x) := 0.006935931x^4 + 0.053403572x^3 + 0.239547871x^2 + 0.693075817x + 1$$

be a fourth order polynomial, which fits 2^x almost perfectly within the range $(-1; 0]$. The residual error between 2^x and $\mathcal{M}$ is at most $5 \cdot 10^{-6}$, consider Figure 2.1. Furthermore, the maximum relative error between 2^x and $\mathcal{M}$ in the range $(-1; 0]$ is less than $1.5 \cdot 10^{-5}\%$. Hence, the relative error between $\exp(x)$ and the approximation is also at most $1.5 \cdot 10^{-5}$ for all $x < 0$, since all x are reduced to approximate 2^x within the range $(-1; 0]$. To reduce the number of arithmetical operations and redundant multiplications of x order, the function is as factorized as possible, let

$$\mathcal{M} := \big((0.006935931x + 0.019890581)x + 0.143440676\big)(x + 4.83179411)x + 1. \tag{2.1}$$

The coefficients are computed by solving

$$\min \sum_{j=0}^{n-1} \left(2^{x_j} - \sum_{i=0}^{4} c_i x_j^i\right)^2$$

where $n = 10^8$ and $x_j := -j/n$ for $j = 0, \ldots, n-1$. The equation can be solved by the linear least squares method, which is described in Section 1.5.2.

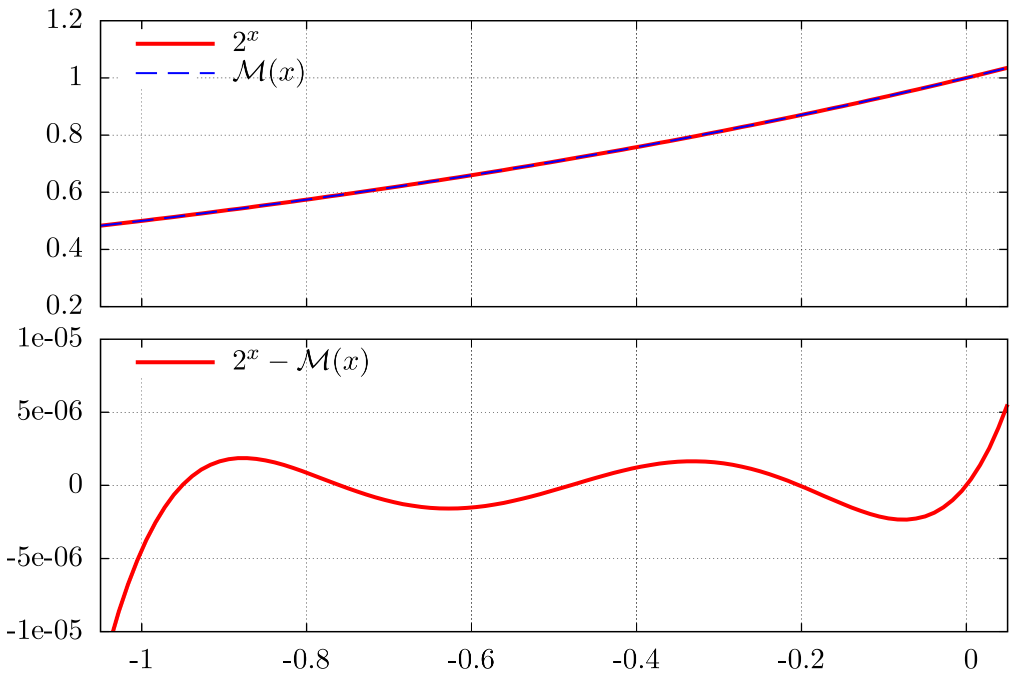


Figure 2.1: (top) Comparison between function 2^x and polynomial $\mathcal{M}$ within the X-axis range $(-1; 0]$, both functions run almost equally; (bottom) residual error between function 2^x and $\mathcal{M}$, the maximum absolute error is about $5 \cdot 10^{-6}$.

The starting coefficients are computed as follows $c_i := \log(2)^i/i!$. Several different approximations for $\mathcal{M}$ are examined, like $(1+\log(2)\cdot x/n)^n$ where n is a power of two e.g. 32, 64, 128 or polynomials of higher order, but either the relative error or the computation time was higher for all of these approaches.

2.4 Implementation

The computation of $\exp(x)$ takes place in three steps, consider C++ code in Listing 2.3. 1) The values f_1 and f_2 are prepared and initially filled, line $9-12$. 2) Value f_1 is computed, line $14-15$, more details in consecutive paragraph. 3) Value f_2 is computed according to Equation (2.1), line $17-22$.

Computing f_1 To provide a fast computation for f_1, the floating point representation is exploited. Within a 32 bit register, the bits $[23; 30]$ are coding the exponent. The exponent is biased with offset 127, making it possible to compare floats by treating them as integers. Let $f_{1,\mathrm{i}}$ be the integer interpretation of bit field f_1 and $f_{1,\mathrm{f}}$ for floating interpretation, respectively. Value x is casted into an integer, whereat implicitly a round up operation is performed

```
const float coeff_4 = 0.006935931, coeff_3 = 0.019890581;
const float coeff_2 = 0.143440676, coeff_1 = 4.831794110;
const float coeff_0 = 1., onebylog2 = 1.442695041;
const int offset_127 = 127, shift_23 = 23;
const float min_x = -88.;

inline float exp_approx(float x){
    if (x > min_x){
        x *= onebylog2;
        union {int i; float f;} f1 = {(int)x};
        x -= f1.i;
        float f2 = x, x_tmp = x;

        f1.i += offset_127;
        f1.i <<= shift_23;

        f2 *= coeff_4; x_tmp += coeff_1;
        f2 += coeff_3; x_tmp *= x;
        f2 *= x;
        f2 += coeff_2;
        f2 *= x_tmp;
        f2 += coeff_0;

        return f1.f * f2;
    }
    return 0.;
}
```

Listing 2.3: C++ code of exp_approx for negative numbers.

(remark that x is negative), let $f'_{1,\mathrm{i}} := (\mathrm{int})x$. Subsequently, the bias is added and afterwards shifted by 23 bits, let $f_{1,\mathrm{i}} := (f'_{1,\mathrm{i}} + 127) \ll 23$.

2.5 Enhancement under Resource Constraints

The following methods are used to allow the compiler to produce more optimized code:

- Using *const* for every constant, the compiler does not need to care about variables being overwritten.
- Splitting Equation (2.1) into its atomic operations allows the compiler to rearrange the order of computations (without a change of the logic). Since the equation contains several parallel computations, the compiler can produce even faster code for processors with the ability of parallel computation, e.g. single instruction multiple data (SIMD) technology.
- The rounding of x for computing f_1 happens implicitly by casting to int. Using the *ceil* function, the computation significantly slows down.

- To avoid costly casts between *int* and *float*, a *union* container is utilized.
- When x drops below -88, the exponent would be less than -127. Hence, the computation will be omitted and zero will be returned. This inaccuracy is acceptable for the further algorithms. Especially computing or checking for valid values will be omitted by returning zero.

2.6 Evaluation

The evaluation of the exponential function approximation contains a computation time benchmark as well as an accuracy benchmark. Since the exp approximation was designed for negative numbers, the benchmark contains four sets $\mathcal{S}_1 = (\mathcal{S}_{1,j}), \mathcal{S}_2 = (\mathcal{S}_{2,j}), \mathcal{S}_3 = (\mathcal{S}_{3,j}), \mathcal{S}_4 = (\mathcal{S}_{4,j})$ with $j := 1, \ldots, 10^8$, where $\mathcal{S}_{1,j}$ is an independent and identically distributed negative random variable within the range $(-1; 0]$, $\mathcal{S}_{2,j}$ within $(-40; 0]$, $\mathcal{S}_{3,j}$ within $(-88; 0]$ and $\mathcal{S}_{4,j}$ within $(-1000; 0]$, respectively. Here, $\mathcal{S}_4$ is only taken into account for computation, since most of the values are smaller than 2^{-88} and thus can not be represented by float values. Hence, a computations of a relative error is not reasonable. To avoid influences falsifying the computation time like the computation of random numbers, only the execution of the main loop will be measured. Additionally, the main loop contains ten calls of each tested function to reduce the computation overhead of loop unrolling by the compiler. We use the exact exp function provided by C/C++ standard math library as the reference. The functions fastexp, fasterexp and EXP as described in Section 2.2 are also analyzed for comparison. For accuracy, the relative error $\mathcal{E}$ is being computed using Equation (1.5). The mean relative error as well as the maximal relative error are taken into account.

Results Table 2.1 and 2.2 present the results of the benchmark. On a Desktop PC, the developed exp_approx function computes about 5.3 times faster than the exact exp function while the relative error is only up to 10^{-5}. Even on the Raspberry Pi, exp_approx is $3 - 4$ times faster. We elucidated empirically, that approximating with a fourth order polynomial provides the best trade-off between the accuracy and computation time. During the evaluation, we profiled the function calls and found that exp_approx requires about 25% of the complete execution time when executing the online peak model estimation described in Section 4. For measurements with normal resolution (e.g. dataset 69, consider Appendix A.2), exp_approx was called several million times and for the highest resolution even about a billion times. When using the ordinary builtin exp function, the calls occupy about 60% of the processing time while the results only differ slightly. The high accuracy is especially necessary for probabilities in statistical distributions. Since the probabilities converge towards zero with increasing negative values, the absolute error also shrinks. Although exp_approx is not the fastest method, it provides the best

Table 2.1: Comparison of exp function provided from standard library in terms of computation time compared to our exp_approx function, fastexp and fasterexp provided by fastapprox-library and EXP provided by Schraudolph (1999). The mean error, mean standard deviation and maximum error is evaluated. The values indicate the factor by which the tested function is faster than the exp function.

Speedup with exp as reference					
	set	exp_approx	fastexp	fasterexp	EXP
Desktop PC	$\mathcal{S}_1$	5.3278	4.5346	12.239	13.1526
	$\mathcal{S}_2$	5.3335	4.5394	12.292	13.1607
	$\mathcal{S}_3$	4.7287	4.6112	10.761	10.4196
	$\mathcal{S}_4$	13.787	9.2259	0.6109	18.88
Raspberry Pi	$\mathcal{S}_1$	3.9802	2.5476	6.7054	13.0584
	$\mathcal{S}_2$	3.9649	2.5414	6.7803	13.214
	$\mathcal{S}_3$	3.6902	2.5477	6.2900	10.175
	$\mathcal{S}_4$	13.222	8.6952	0.7304	0.6164

trade-off between speed and accuracy. The functions EXP and fasterexp are even twice as fast as exp_approx but provide relative errors of about 1%.

Table 2.2: Comparison of exp function provided from standard library against the exp_approx function and fastexp and fasterexp provided by fastapprox library in terms of accuracy. Both measures, the mean relative error and the maximal relative error are benchmarked. Set $\mathcal{S}_4$ is not taken into account, since most contained values are too low for computation.

set	exp_approx		fastexp	
	mean error	max error	mean error	max error
$\mathcal{S}_1$	$7.1 \cdot 10^{-7} \pm 1.1 \cdot 10^{-6}$	$8.94 \cdot 10^{-6}$	$1.0 \cdot 10^{-5} \pm 1.7 \cdot 10^{-5}$	$6.16 \cdot 10^{-5}$
$\mathcal{S}_2$	$8.1 \cdot 10^{-7} \pm 1.3 \cdot 10^{-6}$	$1.06 \cdot 10^{-5}$	$1.0 \cdot 10^{-5} \pm 1.7 \cdot 10^{-5}$	$6.27 \cdot 10^{-5}$
$\mathcal{S}_3$	$1.1 \cdot 10^{-6} \pm 1.4 \cdot 10^{-6}$	$1.26 \cdot 10^{-5}$	0.0019 ± 0.026	0.48
set	fastexp		EXP	
	mean error	max error	mean error	max error
$\mathcal{S}_1$	0.0071 ± 0.0094	0.038	0.01 ± 0.01	0.05
$\mathcal{S}_2$	0.00804 ± 0.0097	0.03894	0.01 ± 0.018	0.05
$\mathcal{S}_3$	0.0093 ± 0.022	0.45	0.012 ± 0.038	0.93

3 Offline Peak Model Estimation in PEAX

The basis of MCC/IMS analysis is *peak extraction*, by which we mean a representation of all high-intensity regions (peaks) in the measurement. An adequate representation is using a few descriptive parameters per peak instead of the full measurement data. Usually, peaks are described by a $\mathcal{D}$-dimensional point indicating the highest signal intensity of the peak as well as the signal intensity itself. In the following, we describe automated peak extraction methods from literature extracting peaks from IMSCs, introduce a parameterized model describing the complete two-dimensional shape of a peak and present a method for automated peak extraction by determination of their descriptive parameters.

3.1 Background

The representation of peaks as a list of descriptive parameters has several advantages, especially in a resource-constrained context. Almost every post-processing step handles peak lists with less complexity than the signal intensity matrix. This includes a comparison and connection of peaks within a set of measurements as well as the determination of the analytes by comparing them with a reference peak list. It is less expensive to store a peak list in terms of storage space as well as a possible transmission of the data in terms of energy consumption. Furthermore, the transformation of the parameters for an easier pairwise peak comparison becomes feasible. Since the use of distance measures can be improved, error tolerant comparisons become more robust.

In general, automated peak extraction methods can be distinguished between two categories: On the one hand, *offline* methods have the whole data matrix of an IMSC available at any time during the entire extraction process. On the other hand, methods are referred to as *online* in which a single (or a small constant amount of) IM spectrum / spectra is processed right after the capturing and then directly discarded. In this section we concentrate on offline methods whereas Section 4 will focus on online methods.

3.2 Related Work

In this section we present peak extraction methods from literature. The following methods *automated extraction in VisualNow*, *automated extraction in IPHEx* as well as *local maxima* are offline methods. All three extraction methods have the description of a single peak in common, which consists of three parameters $\theta = \{\rho^*, \tau^*, h^*\}$ where $h^* = S_{r^*,t^*}$ is the highest signal of the peak and $\rho^* := R_{r^*}, \tau^* := T_{t^*}$. As next, we describe two functions from literature describing the theoretical surface of a peak and briefly consider their properties regarding the efficiency for parameter estimation.

Automated Extraction in VisualNow

VisualNow is a commercial program (distributed by the B&S Analytik, Dortmund, Germany) for the analysis of IMSCs basing on the developments of Bödeker et al. (2008). It provides an automated peak extraction method that was introduced by Bader et al. (2005). Having an IMSC signal matrix, the first step is performed by a k-means algorithm. Every matrix cell is labeled as "peak" or "non-peak". In the second step, the matrix is first processed row-wise. Neighboring cells labeled "peak" are considered as one row unit. This step is associated to a run-length encoding. Having determined all runs in each row, the matrix is now processed column-wise. Adjacent runs of two neighboring rows are merged together. In a last step all centroids of the emerged regions are computed.

Automated Extraction in IPHEx

Another approach for automated peak extraction was introduced by Bunkowski (2011). For this purpose, the watershed method is utilized. Given the IMSC signal matrix, all intensities are first sorted in descending order. The following conditions are checked for all intensities:

- it has no labeled surrounding intensities, it is being labeled as a new region,
- all its surrounding intensities have the same region label, it gets the same label,
- at least two its surrounding intensities have different region labels, it is labeled as a watershed.

Every region is treated as a peak and the position of the highest intensity is assigned to the peak.

Local Maxima

D'Addario et al. (2014) introduced a complete pipeline (called PEAX) for automated peak extraction containing the four steps preprocessing, candidate detection, peak picking and estimation. The pipeline will be described in more detail in Section 3.4, since the peak model estimation which is introduced in this section is a part of this pipeline. One module within the candidate detection step is called *local maxima*, which is an easy and fast method for peak finding. Going through the data matrix of an IMSC, every cell as well as its eight surrounding neighbor cells are taking into consideration. If the values of all neighbors are lower than the value of the current cell and this value exceeds a tunable threshold, it is considered a peak candidate. In the second step, peak candidates which are too close to each other are merged. Here, a weighted cluster editing problem is being solved. A more detailed summary is given in Section 5.2. Cliques of peak candidates detected by cluster editing are merged into final peaks. When merging two or more candidates, both the position and the signal value of the highest candidate is assigned to the final peak. The estimation step is omitted in this process.

Theoretical surface of a Peak

Bödeker and Baumbach (2009) introduced a function with nine parameters to describe the theoretical surface of a peak using the log-normal distribution. Having $\theta = (\rho^*, \tau^*, h^*)$ defined as the mode position of a peak (retention time, IRM) and its signal intensity, the function $\mathbf{P}$ describing the surface is defined as follows:

$$
\begin{aligned}
\tilde{\mu}_{b,\sigma}(x) &:= \log(x-b) + \sigma, \\
a(h, \mu, \sigma) &:= \frac{h\sigma\sqrt{2\pi}}{\exp^{0.5\sigma^{-2}-\mu}}, \\
z_{x^*,b,\beta}(x) &:= \beta \cdot (x - x^*) + x - b, \\
\mathcal{N}_{\mathrm{L}}(x \mid \mu, \sigma) &:= \exp^{-\frac{(\log(x)-\mu)^2}{2\sigma^2}} x \cdot \sigma \cdot \sqrt{2\pi}, \\
\mathbf{P}_{\theta}(\rho, \tau) &:= \frac{1}{h^*} \cdot a(h^*, \tilde{\mu}_{b_{\mathrm{R}},\sigma_{\mathrm{R}}}(\rho), \sigma_{\mathrm{T}}) \cdot \mathcal{N}_{\mathrm{L}}(z_{\rho^*,b_{\mathrm{R}},\beta_{\mathrm{R}}}(\rho) \mid \tilde{\mu}_{b_{\mathrm{R}},\sigma_{\mathrm{R}}}(\rho), \sigma_{\mathrm{R}}) \\
&\quad \cdot a(h^*, \tilde{\mu}_{b_{\mathrm{T}},\sigma_{\mathrm{T}}}(\tau), \sigma_{\mathrm{T}}) \cdot \mathcal{N}_{\mathrm{L}}(z_{\tau^*,b_{\mathrm{T}},\beta_{\mathrm{T}}}(\tau) \mid \tilde{\mu}_{b_{\mathrm{T}},\sigma_{\mathrm{T}}}(\tau), \sigma_{\mathrm{T}})
\end{aligned}
$$

for every $\rho \in R, \tau \in T$ where $\theta = \{\rho^*, \tau^*, h^*, \sigma_{\mathrm{R}}, b_{\mathrm{R}}, \beta_{\mathrm{R}}, \sigma_{\mathrm{T}}, b_{\mathrm{T}}, \beta_{\mathrm{T}}\}$. This peak description is the product of two one-dimensional statistical distributions. However, the rather complex usage of parameters allows no intuitive understanding of their functions. Providing maximum likelihood estimators for every parameter is rather difficult. The number of parameters also makes the model less suitable since more parameters mean more consumption of computation power.

Function for describing the Peak Surface

Another function for describing the peak surface was introduced by Vogtland and Baumbach (2009) using eight parameters. Considering a peak in both dimensions retention time and IRM, the peaks follow a similar pattern, i.e. they all have a more pronounced tailing. Hence, the approach is to split the function at the modes position. The fronting is described by a Gaussian distribution and the tailing by a Breit-Wigner function. Since the tailing becomes extreme strong in retention time, a logarithmic Breit-Wigner function is suggested. Here, $\theta = \{\rho^*, \tau^*, h^*\}$ is again the mode position of a peak (retention time, IRM) and its signal intensity, thus the peak surface function $\mathbf{P}$ is defined as follows:

$$f_\mathrm{T}(x \,|\, x_0, h, w_\mathrm{g}, w_\mathrm{bw}) := \begin{cases} h & \text{if } x = x_0 \\ h \cdot 2^{\frac{(x-x_0)^2}{w_\mathrm{g}^2}} & \text{else if } x < x_0 \\ h \cdot \frac{w_\mathrm{bw}^2}{w_\mathrm{bw}^2 + (x-x_0)^2} & \text{else} \end{cases},$$

$$f_\mathrm{lbw}(y \,|\, y_0, h, a, w, \zeta) := \begin{cases} \frac{h \log\left(\frac{a+w}{a}\right)^\zeta}{\log\left(\frac{a+w}{a}\right)^\zeta + \log\left(\frac{a+w-y_0}{a}\right)^\zeta} & \text{if } y - y_0 + > 0 \\ 0 & \text{else} \end{cases},$$

$$\mathbf{P}_\theta(\rho, \tau) := f_\mathrm{T}(\tau \,|\, \tau^*, f_\mathrm{lbw}(\rho \,|\, \rho^*, h^*, a, w_\mathrm{r}, \zeta), w_\mathrm{g}, w_\mathrm{bw})$$

for every $\rho \in R, \tau \in T$ where $\theta = \{\rho^*, \tau^*, h^*, w_\mathrm{g}, w_\mathrm{bw}, w_\mathrm{R}, a, \zeta\}$, $w_\mathrm{g}, w_\mathrm{bw}$ are the width for both Gaussian as well as Breit-Wigner function in IRM, w_R the width of logarithmic Breit-Wigner function in retention time, a the negative distance from ρ^* in retention time and ζ a tunable exponent. This model has one parameter less than the previous model and also describes the two-dimensional peak as a product of two one-dimensional distributions. Besides the fact that it is unreasonable to describe a physical process with two truncated statistical distributions for one dimension, approaches like the EM algorithm become unsuitable for parameter estimation.

3.3 Peak Model

For our purpose of analyzing MCC/IMS measurements, a peak is characterized by the following assumptions.

Assumption. A $\mathcal{D}$-dimensional peak $\mathfrak{P}$ is a product of $\mathcal{D}$ log-concave functions with two inflection points in each dimension and a parameter set θ. The peak width at half height $\omega_{1/2,i}$ can be calculated with respect to the mode for each dimension $i = 1, \ldots, \mathcal{D}$. At its mode $(m_1, \ldots, m_\mathcal{D})$, $\mathfrak{P}_\theta$ exceeds the average background noise level by a certain factor multiplied by the standard deviation of the noise, i.e., $I := \mu_\mathrm{N} + \texttt{noise\_margin} \cdot \sigma_\mathrm{N}$.

Assumption. For MCC/IMS measurements, the peak $\mathbf{P}$ is a special case of $\mathfrak{P}$ with $\mathcal{D} = 2$ dimensions.

To describe peak $\mathbf{P}$ in both dimensions retention time and IRM, we use the shifted Inverse Gaussian distribution. Other skewed statistical distributions, i.e. Beta, Chi^2, Erlang, F, Gamma or the Weibull distribution, were examined but discarded since all these distributions have the disadvantage of a potential curve formation with a single inflection point when parameters are set inappropriately. The Inverse Gaussian is defined by its density

$$g(x \mid \mu, \lambda, o) := \frac{1[x > o]}{\sqrt{2\pi}} \cdot \sqrt{\frac{\lambda}{(x-o)^3}} \cdot \exp\left(-\frac{\lambda\big((x-o)-\mu\big)^2}{2\mu^2(x-o)} \right). \tag{3.1}$$

Its parameters are the shift (or offset) o, the relative mean $\mu > 0$ (to the right of o) and the shape parameter $\lambda > 0$. A peak is then given as the product of two shifted Inverse Gaussians, scaled by a volume factor v, i.e., by seven parameters $\theta = \{v, \mu_\mathrm{R}, \lambda_\mathrm{R}, o_\mathrm{R}, \mu_\mathrm{T}, \lambda_\mathrm{T}, o_\mathrm{T}\}$; so the density function of a peak is defined as

$$\mathbf{P}_\theta(\rho, \tau) := v \cdot g(\rho \mid \mu_\mathrm{R}, \lambda_\mathrm{R}, o_\mathrm{R}) \cdot g(\tau \mid \mu_\mathrm{T}, \lambda_\mathrm{T}, o_\mathrm{T})$$

for all $\rho \in R, \tau \in T$ as we introduced in Kopczynski et al. (2012).

Since the parameters μ, λ, o of a shifted Inverse Gaussian may be different despite the similar shape of the resulting distributions, it is more intuitive to describe the shifted Inverse Gaussian in terms of three different *descriptors*: the (absolute) mean μ', the standard deviation σ and the mode m. There is a bijection between (μ, λ, o) and the descriptors (μ', σ, m). Given (μ, λ, o), we have

$$\begin{aligned}
\mu' &= \mu + o,\\
\sigma &= \sqrt{\mu^3/\lambda},\\
m &= \mu \cdot \big(\sqrt{1 + (9\mu^2)/(4\lambda^2)} - (3\mu)/(2\lambda)\big) + o,
\end{aligned}$$

and, given (μ', σ, m), we use auxiliary expressions p_1 and p_2 to find

$$\begin{aligned}
p_1 &:= \big(-m(2\mu' + m) + 3 \cdot (\mu'^2 - \sigma^2)\big)/\big(2(m - \mu')\big), && (3.2)\\
p_2 &:= \big(m(3\sigma^2 + \mu' \cdot m) - \mu'^3\big)/\big(2(m - \mu')\big), && (3.3)\\
o &= -p_1/2 - \sqrt{p_1^2/4 - p_2},\\
\mu &= \mu' - o,\\
\lambda &= \mu^3/\sigma^2.
\end{aligned}$$

The advantages of this description in contrast to both peak descriptions presented in Section 3.2 are i) intuitive parameters, ii) inherent normalization

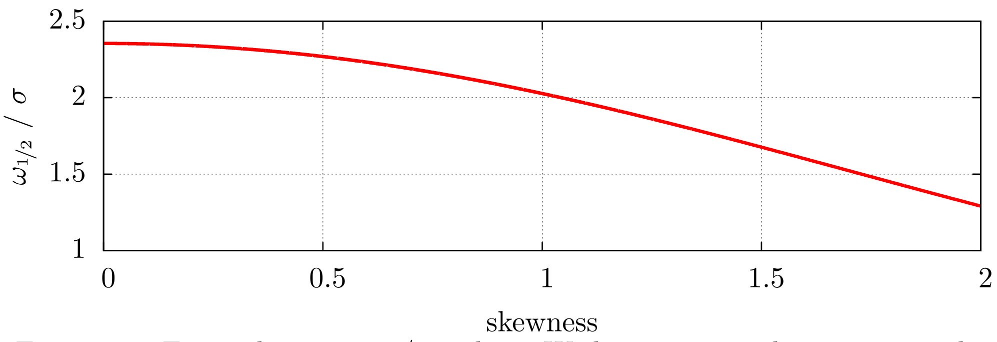


Figure 3.1: Factor between $\omega^{1/2}$ and σ. With increasing skewness it is decreasing. The highest factor ~ 2.3548 is achieved for a symmetric inverse Gaussian.

of the model (important for statistical methods) and iii) fewest number of parameters.

We also employ the following empirically observable properties of peaks in real IMSCs which concern the peak widths on both the IRM axis and the retention time axis. The full width at half maximum (FWHM) can be described as the length $\omega_{1/2}$ of the interval around the mode where the peak height is at least half of its maximum height. For a (symmetric) Gaussian distribution, there is a linear relation between the standard deviation σ and $\omega_{1/2}$:

$$\omega_{1/2} = \phi \cdot \sigma \qquad \text{with } \phi = 2\sqrt{2\ln 2} \approx 2.3548\,. \tag{3.4}$$

This relation also holds well for not too skewed Inverse Gaussian distributions (see Figure 3.1) and is a good approximation to estimate its descriptor σ from an empirically observed $\omega_{1/2}$.

Given the mode d^* of a peak in drift time (in ms), we can estimate its descriptors (m, σ, μ') in IRM units as follows. Recall that the IRM mode (in $\mathrm{V\,s\,cm^{-2}}$) is simply $m = f_{\text{ims}} \cdot d^*$, where f_{ims} is the conversion constant between drift time and IRM (see Section 1.3). Spangler and Collins (1975) empirically derived that $\omega_{1/2} = \sqrt{(11.09\,\mathbf{D}\,d^*)/\mathcal{V}_{\text{d}}^2 + d_{\text{grid}}^2}$, where $\mathbf{D}$ is the diffusion coefficient, $\mathcal{V}_{\text{d}}$ the drift velocity. Using the relation stated by Einstein (1905), $\mathbf{D}$ can be computed as $\mathbf{D} = K\mathcal{K}_{\text{B}}\mathcal{T}/\mathcal{Q}$, where K is the ion mobility which can be derived from d^* using Equation (1.1), $\mathcal{K}_{\text{B}}$ the Boltzmann constant, $\mathcal{T}$ the absolute temperature and $\mathcal{Q}$ the electric charge. We then use Equation (3.4) to estimate $\sigma \approx \omega_{1/2}/\phi$. Finally, the mean value is empirically found to be $\mu' \approx f_{\text{ims}} \cdot \left(d^* + \sqrt{(4.246 \cdot 10^{-5})^2 + (d^*)^2/585048.1633}\right)$.

On the retention time axis, the peak width $\omega_{1/2}$ grows approximately linearly with retention time, i.e., there are the constants `r_width_offset` > 0 and `r_width_factor` > 0 such that the width of a peak with mode at retention time ρ is approximately

$$\xi(\rho) := \rho \cdot \texttt{r\_width\_factor} + \texttt{r\_width\_offset}\,. \tag{3.5}$$

Both parameters are adjustable and depend on the temperature and carrier gas flow within the MCC. The standard deviation in retention time can finally be calculated as $\Delta\rho := \xi(\rho)/\phi$. Since two or more peaks in an IMSC can slightly influence each other in shape and intensity, these equations can be used to compute initial parameters and maximize them according to the observed data.

3.4 Algorithm

Peak model estimation (PME) is a particular choice of certain modules within the PEAX pipeline as stated by D'Addario et al. (2014). The peak extraction process is divided into four steps. Each step has a defined task but can be performed by different methods which are referred to as modules. Several modules are already implemented within every step. We now discuss the four distinct steps.

Preprocessing transforms a raw IMSC into another processed IMSC, i.e., no data reduction or peak extraction takes place. Raw IMSCs are noisy and include the additional RIP. To remove both noise and the RIP, we describe three modules: *baseline correction*, *de-noising* and *smoothing*; every module's input and output is an IMSC. Baseline Correction handles the RIP (and the baseline in general), removes it, and uncovers underlying peaks. De-Noising estimates the probability of a data point resulting from noise in order to remove the noise. Smoothing applies a smoothing filter. The order of execution is commutable, but none of these modules can be omitted.

Peak Candidate Detection finds a list of potential peaks within the preprocessed IMSC. One of the implemented modules within this step is *Cross Finding*, belonging to the peak model estimation process. The input of the module is a processed IMSC, and the output is a list of candidate peaks, further refined in the next step.

Peak Picking examines the proposed candidates and generates the final list of extracted peaks. Three implemented modules are available. All three methods calculate a representative peak for a set of peaks whose positions are too close to be considered distinct. Peak model estimation uses the module *EM Clustering* which discovers peak clusters utilizing the EM algorithm.

Peak Modeling is an optional final step to estimate additional peak parameters, describing shape and position more precisely. Peak model estimation utilizes this optional step whereas the EM algorithm is used to estimate parameters of the given peak shape from Section 3.3 for every peak.

Although some modules in the PEAX pipeline are commutable (especially all modules within the preprocessing step), PME has a fixed defined execution order, namely: baseline correction, de-noising, smoothing, cross finding, EM clustering, peak model estimation. Since PEAX is collaboratively developed, only the own developed modules are being presented in the following.

3.4.1 Preprocessing

Spectra usually have several inherent features in common beside their desired signals, e.g. background noise. In particular for IMSC, additional features like the reactant ion peak (RIP) and the tailing function as described in Section 1.3 influence the spectra and hence the peak extraction methods. To suppress the disturbing features, the three preprocessing methods *baseline correction*, *de-noising* and *smoothing* are applied for cleaning IMSCs as described in the following.

Baseline Correction

In the following, we describe the properties of a baseline within IMSCs and how to correct the data.

Background Intuitively and informally, a baseline spectrum $B = (B_t)_{t \leq |T|}$ is defined such that B_t is a typical or insignificantly high value at IRM τ when considering the whole measurement. Formally, for each IRM τ we consider the distribution of all intensities of chromatogram $S_{\cdot,t}$. Applying the baseline correction, the features like RIP, tailing function or other signals occurring within all IM spectra at the same IRM position are eliminated, since no information can be extracted which is relevant for the desired signals. Bader et al. (2008) presented a method which assumes a log-normal model as baseline and estimates its parameters using the Quasi-Newton before subtracting from spectrum. We developed a new method since Bader's method was designed to suppress the tailing function and does not erase the complete RIP.

Mixture Model A typical chromatogram denoted as $S_{\cdot,t}$ contains mostly noise, one or a few peak(s) and no RIP. Hence, considering a histogram H_t of $S_{\cdot,t}$ (with the bin size assigned to the smallest signal intensity value), the most prominent peak within H_t indicates that level. In the RIP chromatogram, the most prominent peak corresponds to the RIP level (Figure 3.2). In both cases and in the intermediate ones as well, we model the most prominent peak of the histogram by a Gaussian distribution and the remainder by the uniform distribution between the lowest and highest observed intensity.

Thus, we describe the chromatogram distribution by a heterogeneous two-component mixture model (Gaussian plus uniform) and estimate its parameters ($\mu_\mathrm{G}, \sigma^2_\mathrm{G}$ for the Gaussian, ω_G for the Gaussian mixture coefficient) by the EM algorithm.

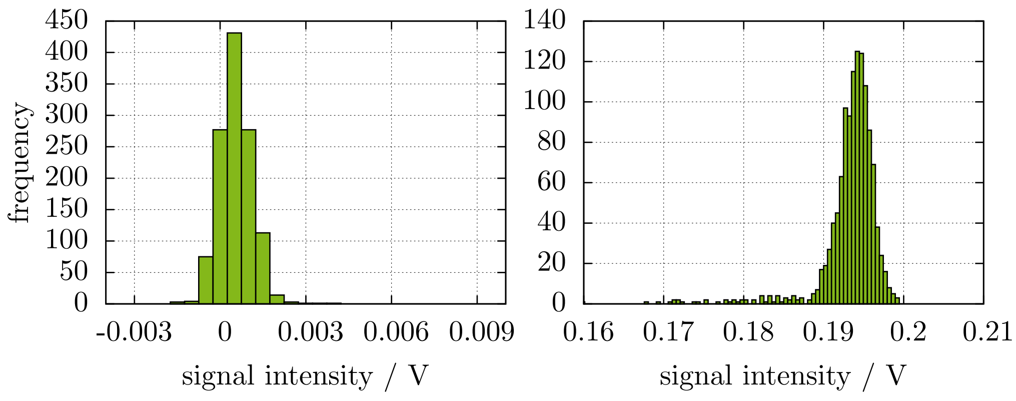


Figure 3.2: Histograms (y-axis: frequency) of signal intensities (x-axis) of two chromatograms, a typical one (left) and a RIP chromatogram (right). The prominent intensity peak is modeled by a Gaussian distribution.

Initial Parameters To start the EM iteration, we set μ_G to the location of the maximum of H_t and $\sigma_\mathrm{G} := \sigma_\mathrm{N}$, while $\omega_\mathrm{G} = \big|\big\{t \in |T| \,\big|\, |H_t - \mu_\mathrm{G}| < 3\sigma_\mathrm{G}\big\}\big| / |T|$ which is relative number of values H_t within the interval $[\mu_\mathrm{G} - \sigma_\mathrm{G}; \mu_\mathrm{G} + \sigma_\mathrm{G}]$.

Final Step After convergence, having estimated $\mu_\mathrm{G}, \sigma_\mathrm{G}$, we say that all intensities up to $\mu + 2\sigma$ belong to the baseline and adjust the chromatogram as follows: $S'_{r,t} := S_{r,t} - (\mu_\mathrm{G} + 2\sigma_\mathrm{G})$ for all $r = 1, \ldots, R$. After repeating this step for every t with individually estimated $\mu_{\mathrm{G},t}, \sigma_{\mathrm{G},t}$, the baseline $B_t = \mu_{\mathrm{G},t} + 2\sigma_{\mathrm{G},t}$ has been removed.

De-Noising

The next section considers an inherent noise of IMSC measurements and how to suppress it.

Background Noise is understood as low signal intensity values occurring around the mean value with a low variance. The goal of de-noising is to subtract a substantial amount of noise from the IMSC S by estimating whether the intensity $S_{r,t}$ at index (r, t) belongs to a peak region or can solely be explained by noise. In previous work on de-noising, Bader et al. (2008) use a wavelet transform but apply it only for each spectrum individually. Our novel approach is similar to the Baseline Correction module in the sense that the EM algorithm is used, but the model is more complex and the subtraction works differently.

The method is not directly applied to S, but to a matrix A created by a moving average filter defined as

$$\begin{aligned}
\kappa_{\mathrm{T}} &:= \frac{\Delta\tau \cdot |T|}{\max(T)}, \\
\kappa_{\mathrm{R}} &:= \frac{\Delta\rho \cdot |R|}{\max(R)}, \\
A_{r,t} &:= \frac{1}{(2\kappa_{\mathrm{R}}+1)(2\kappa_{\mathrm{T}}+1)} \cdot \sum_{r'=r-\kappa_{\mathrm{R}}}^{r+\kappa_{\mathrm{R}}} \sum_{t'=t-\kappa_{\mathrm{T}}}^{t+\kappa_{\mathrm{T}}} S_{r',t'},
\end{aligned}$$

where $\kappa_{\mathrm{T}}, \kappa_{\mathrm{R}}$ are the smoothing radii according to the standard deviations of the peaks. Here, $\Delta\tau$ = `t_width_offset` is also an adjustable parameter and $\Delta\rho$ is computed as described in Section 3.3.

Mixture Model Considering the distribution of all A-values, we identify three components: the noise component (the one of interest and to be removed) is modeled as a Gaussian distribution, the signal component (to be kept) is modeled as an Inverse Gaussian distribution and a background component (explaining every intensity not specifically explained by the other components) is modeled as a uniform distribution over all measured intensities. This yields a three-component heterogeneous mixture model (Gaussian plus Inverse Gaussian plus uniform), whose parameters are again estimated by the EM algorithm.

Final Step After convergence and a final E-step, we obtain the expected membership values $W_{r,t}$ (which are a weighted normalized values of every probability computed by the probability models) of each data point at index (r,t) in the noise component. We adjust the original IMSC such that only the non-noise fraction remains, i.e., $S'_{r,t} := (S_{r,t} - \mu_{\mathrm{G}}) \cdot (1 - W_{r,t})$ where μ_{G} is the mean parameter for the Gaussian model for all $r := 1, \ldots, |R|$ and $t := 1, \ldots, |T|$.

Smoothing

Finally, the last method preprocessing a raw data matrix is introduced in the following section.

Background In general, background noise is a mixture of a complete frequency bandwidth whereas desired peaks contain only low frequencies. It manifests itself as a slight jittering of the signal intensities. To reduce the amount of background noise within the spectrum, the two consecutive methods *lowpass filter* and *Savitzky-Golay Filter* are utilized in the smoothing step. Liu et al. (2012), for example, use a wavelet transform to smooth two-dimensional nuclear magnetic response spectra.

Lowpass Filter The first method is a lowpass filter. The IMSC is transformed from the time/signal domain into the frequency/signal domain by a two-dimensional fast Fourier transform (2DFFT), detailed description given in (Cormen et al., 2009, Chapter 30). All frequencies exceeding a tunable frequency threshold (parameter `fftcutoff`) are removed, i.e., set to zero intensity. Only the adjusted percentage of the first frequencies within this domain are preserved. The inverse transformation of the filtered matrix is executed using the inverse two-dimensional fast Fourier transform (I2DFFT).

Savitzky-Golay Filter The IMSC is smoothed by a Savitzky and Golay (1964) filter (SGF) on local windows using a $(2\kappa_{\mathrm{T}} + 1) \times (2\kappa_{\mathrm{R}} + 1)$ kernel around each data point. In contrast to a smoothing filter with Gaussian kernel, the Savitzky-Golay filter has the advantage of a preserved peak height and width. To handle the boundaries of the measurement, we expand the data matrix with a margin containing only zero values. This procedure is uncritical since the data at the boundary of the measurement does not contain important data. The SGF computes a weighted average across the window.

Cropping to Zero

As a step between the preprocessing and the following steps, all signal intensities < 0 are set to zero. All peaks should have intensities above zero. This step is important, since some modules like the EM algorithm cannot operate with negative values.

3.4.2 Candidate Detection

The basic idea of Cross Finding is to find maxima based on the ideas by Fong et al. (2011), searching for roots within the first derivatives in both retention time and IRM dimension.

We construct two auxiliary matrices V^{R} and V^{T}, both with the same dimensions $|R| \times |T|$. In V^{T}, discrete derivatives of spectra are stored (partial derivatives with respect to the IRM), $V^{\mathrm{T}}_{r,t} := S_{r,t+1} - S_{r,t}$; analogously derivatives of chromatograms are stored in V^{R}. We describe how V^{T} is analyzed.

In each derived spectrum (for fixed retention time index r), we mark downward zero crossings; these are indices t with $V^{\mathrm{T}}_{r,t-1} \geq 0$ and $V^{\mathrm{T}}_{r,t} < 0$. The resulting indices t are called *active positions* for retention time index r.

During the scan through the spectra, two data structures are maintained. The first one is an *active set* containing lists of active positions connected

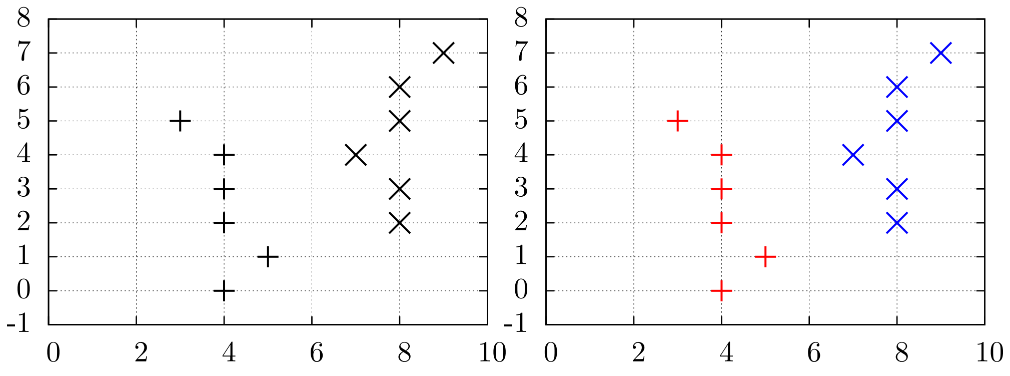


Figure 3.3: Cross Finding as described by D'Addario et al. (2014): Active positions (marking potential peak maxima) are initially unaligned (left) and then connected by alignment across spectra (right; shown as red + and blue x). The same procedure is repeated for all chromatograms giving horizontal bands instead of vertical bands. Intersecting the results from both dimensions results in peak candidates.

across several spectra. The second one is a *finalized set*, to which lists from the active sets are moved after having been processed. Initially both sets are empty.

Our aim is to connect active positions between consecutive retention times, i.e., we want to find active positions for spectrum $r+1$ corresponding to active positions in spectrum r (see Figure 3.3(left)). To find the correspondences, we compute a global alignment between the two sorted integer lists $\mathcal{A} \subset \{1, \ldots, |T|\}$ and $\mathcal{A}^+ \subset \{1, \ldots, |T|\}$ containing the active positions. Here, we utilize the Needleman and Wunsch (1970) algorithm. Depending on the dimension, the following distance functions are applied:

$$\mathbf{dist}(i', j')^{\mathrm{R}} := \frac{|R_{\mathcal{A}[i']} - R_{\mathcal{A}^+[j']}|}{\Delta\rho},$$
$$\mathbf{dist}(i', j')^{\mathrm{T}} := \frac{|T_{\mathcal{A}[i']} - T_{\mathcal{A}^+[j']}|}{\Delta\tau}$$

for retention time and IRM dimension, respectively. To prevent that two positions with a high distance are aligned, we introduce a gap $\mathbf{g} = 1$.

Applying a global alignment, three scenarios are possible in regard to the aligned position pairs:

(1) If $\mathcal{A}^+[j']$ is not aligned to any $\mathcal{A}[i']$, it is a "new" active position and a new list containing only $\mathcal{A}^+[j']$ is inserted into the active set.

(2) If $\mathcal{A}^+[j']$ is aligned to some $\mathcal{A}[i']$, the corresponding list containing $\mathcal{A}[i']$ is already in the active set and extended by $\mathcal{A}^+[j']$.

(3) Each $\mathcal{A}[i']$ which is not aligned to any $\mathcal{A}^+[j']$ finalizes its corresponding active list and the list is moved into the finalized set.

After processing all spectra and finalizing each remaining list, we obtain several position lists pointing out consecutive maxima throughout each spectrum; see Figure 3.3(right).

The same procedure is analogously performed with matrix V^{R}. We report the intersection of positions found from both matrices (which can be visualized as crosses; hence the name "Cross Finding"). If more than one position overlap is found between two lists, the position with the higher signal intensity is reported. Each reported point whose signal intensity exceeds the threshold $\sigma_{\mathrm{N}} \cdot$ `noise_margin` is a candidate for a peak location with `noise_margin` as an adjustable parameter.

3.4.3 Peak Picking

The mean retention time and IRM of a peak is random within a small interval. Given the physical separation properties of an MCC/IMS device, different peaks with too high overlapping appearance intervals must be considered as one peak. The objective of this module is to recognize peak candidates from the previous step which comply with this property. This module also utilizes the EM algorithm. Initially, each peak candidate represents a component but during the course of the algorithm, components can be merged. The remaining components will represent the picked peaks. This method is referred to as EM Clustering and a detailed explanation is given in Section 5.

Each component is a two-dimensional Gaussian distribution with independent dimensions, i.e., diagonal covariance matrix. Initially, the mean of every component is the (ρ, τ) location of the corresponding peak candidate. The standard deviation on the R- and T-axis is set to $\Delta\rho$ and $\Delta\tau$, respectively. In the E-step, the hidden membership coefficients of each peak are estimated for each component. When a peak candidate is close to another one, the probability that the first model also (partially) describes the second candidate is comparatively high. In the maximization phase, the parameters of each component are re-estimated based on candidate membership using maximum likelihood estimators. In the case of two close candidates, the mean of both components moves towards their middle. When the distance between the means of two components drops below a given threshold, the components are merged: The component of the candidate with the lower signal intensity is removed, and its weight is added to the remaining model. The E- and M-steps are repeated until convergence.

When updating the variance by maximum likelihood estimation, we must be aware that the variance of a component described by only one peak tends to zero, leading to a singularity in the Gaussian density function. Therefore, we restrict the estimated standard deviation to values exceeding the threshold $\tau_{\min} := \Delta\tau \cdot 10^{-5}\,\mathrm{V\,s\,cm^{-2}}$.

3.4.4 Peak Modeling

Peak modeling is an optional step which estimates a parametric model of a peak shape. Within the PME pipeline it is enabled, providing a module of the same title introduced by Kopczynski et al. (2012). It outputs the peak list with the additional parameters appended.

Mixture Model

A whole IMSC is interpreted as a sample from a mixture model of different shifted Inverse Gaussians and a uniform background noise model. Then, each component (peak) can be described by seven parameters (three for both shifted Inverse Gaussians in both T- and R-dimension, plus one mixture coefficient). The challenge is to estimate the parameters correctly, especially when peaks overlap. Again, the EM algorithm is utilized for this purpose.

For efficiency, each component model is only evaluated in a surrounding box enclosing the peak. Starting from the picked peak location, the borders of the box are expanded in all four main directions until the signal intensity drops below σ_{N} in each direction. Additionally, the box is expanded by κ_{T} in both main IRM directions and by κ_{R} in both main retention time directions, respectively. When two boxes intersect, both boxes are merged into their convex hull. After this process, we have a set of boxes containing at least one peak.

Initial Parameters

Now we can apply EM to each box independently with the advantage of processing smaller boxes in contrast to the whole signal matrix. The weights are at the beginning $\omega := 1/c$ having c components. Starting parameters for each component are estimated from the locations of picked peaks and additional assumptions: The descriptors are chosen such that their modes correspond to the known (ρ^*, τ^*) values, the mean is set slightly higher ($\mu_{\mathrm{T}} = \tau^* + \tau_{\min}$), and the standard deviation is set with $\Delta\tau$ in IRM dimension and $\Delta\rho$ in retention time, respectively. The initial model parameters can be recomputed using the descriptors as explained in Section 3.3. Hence, we obtain for model j a parameter vector $\theta_j = \{\omega, \mu_{\mathrm{R}}, \lambda_{\mathrm{R}}, o_{\mathrm{R}}, \mu_{\mathrm{T}}, \lambda_{\mathrm{T}}, o_{\mathrm{T}}\}$.

M-Step

Let c be the number of peaks provided by the picking step. Additionally, we add a background component with index $j = 0$. Using the membership variables $W_{(r,t),j}$ for all $j := 0, \dots, c$, the data point intensity $S_{r,t}$ decomposes into independent components $S_{r,t}^{(j)}$, each of which represents a single peak (or the background), via

$$S_{r,t}^{(j)} := W_{(r,t),j} \cdot S_{r,t}.$$

We first estimate the new model weights ω_c^* by maximum likelihood; a standard calculation shows that

$$\omega_j^* = \frac{1}{\tilde{S}} \sum_{r \in R, t \in T} S_{(r,t)}^{(j)},$$

where $\tilde{S} = \sum_{r,t} S_{r,t}$ is the total signal intensity for all $r := 1, \dots, |R|$ and $t := 1, \dots, |T|$. There are no further parameters for the background component and the process is complete.

For the peak component, we estimate each parameter of θ_j. Maximum likelihood estimators for the one-dimensional shifted Inverse Gaussian are known as stated by Cheng and Amin (1981) and Koutrouvelis et al. (2005). Since our two-dimensional model factors into two one-dimensional Inverse Gaussians, these estimators can be used by marginalizing over the respective other dimension. We provide the resulting estimators $(\mu_j^*, \lambda_j^*, o_j^*)$ for the retention time axis. Let $\tilde{S}_{(r,\cdot)} := \sum_{t \in T} S_{(r,t)}^{(j)}$ be the marginalized signal.

The relative mean is naturally estimated as

$$\mu_j^* = \frac{\sum\limits_{r \leq |R|} R_r \cdot \tilde{S}_{(r,\cdot)}^{(j)}}{\omega_j^* \, \tilde{S}} - o_j^*.$$

Thus, the offset estimate o_j^* has to be known to estimate μ_j^*. We solve this problem by first using the previous value o_j^0, then estimating o_j^* (see below) using μ_j^*, and finally updating μ_j^* by using o_j^*. This can be repeated until convergence, but we found that in one iteration is sufficient. The shape parameter λ_j^* is ML-estimated by

$$\lambda_j^* = \frac{\omega_j^* \, \tilde{S}}{\sum\limits_{r \leq |R|} \tilde{S}_{(r,\cdot)}^{(j)} \cdot \Big(1/(R_r - o_j^*) - 1/\mu_j^*\Big)}.$$

We circumvent the difficulty of requiring o_j^* as above. The ML-estimator for the offset o_j^* is determined by

$$0 = \sum_{r \leq |R|} \tilde{S}^{(j)}_{(r,\cdot)} \cdot \left(\frac{3}{\lambda_j^*(R_r - o_j^*)} + \frac{1}{(\mu_j^*)^2} - \frac{1}{(R_r - o_j^*)^2} \right).$$

There is no closed formula for o_j^*, but its value can be efficiently determined using Newton's method. The EM algorithm interrupts the process, once all convergence criteria are fulfilled.

Final Step

As the model parameters have a rather technical interpretation, they are translated back into the descriptors, i.e. mode, mean and standard deviation of the distribution, which are conveniently compared and interpreted.

3.5 Evaluation

In the following, we evaluate several aspects introduced within this section. First, the quality of the introduced peak model is evaluated. Subsequently, an evaluation of the complete algorithm is presented as documented in Section 3.4.

3.5.1 Quality of the Model

We evaluate the quality of the obtained peak models by computing the log-likelihood of the observed normalized signal $S'_{r,t} := S_{r,t}/\tilde{S}$ where $\tilde{S} = \sum_{r \leq |R|, t \leq |T|} S_{r,t}$ under three different distributions: (1) the empirical distribution $S'_{r,t}$ itself, the most accurate description of the data requiring $|R|\,|T|$ parameters; (2) the estimated mixture model $\mathbf{P}_\theta(\rho, \tau)$ requiring seven parameters θ per peak; (3) the uniform distribution on $|R| \times |T|$, irrespective of the data and requiring no parameters. Thus, let

$$\begin{aligned} L_{\mathrm{D}} &:= \sum\nolimits_{r,t} S'_{r,t} \log(S'_{r,t}), \\ L_{\mathrm{M}} &:= \sum\nolimits_{r,t} S'_{r,t} \log(\mathbf{P}_\theta(R_r, T_t)), \\ L_{\mathrm{U}} &:= \sum\nolimits_{r,t} S'_{r,t} \log(1/(|R|\,|T|)) \;=\; -\log(|R|\,|T|). \end{aligned}$$

Of course $L_{\mathrm{D}} \geq L_{\mathrm{M}} \geq L_{\mathrm{U}}$; we claim that for IMS measurements with high peak density, $L_{\mathrm{D}} \approx L_{\mathrm{M}} \gg L_{\mathrm{U}}$, even though our model requires only seven parameters per peak instead of $|R|\,|T|$ data points. Hence, we compute the relative size of L_{M} in the interval $[L_{\mathrm{U}}, L_{\mathrm{D}}]$, that is $Q := (L_{\mathrm{M}} - L_{\mathrm{U}})/(L_{\mathrm{D}} - L_{\mathrm{U}}) \in$

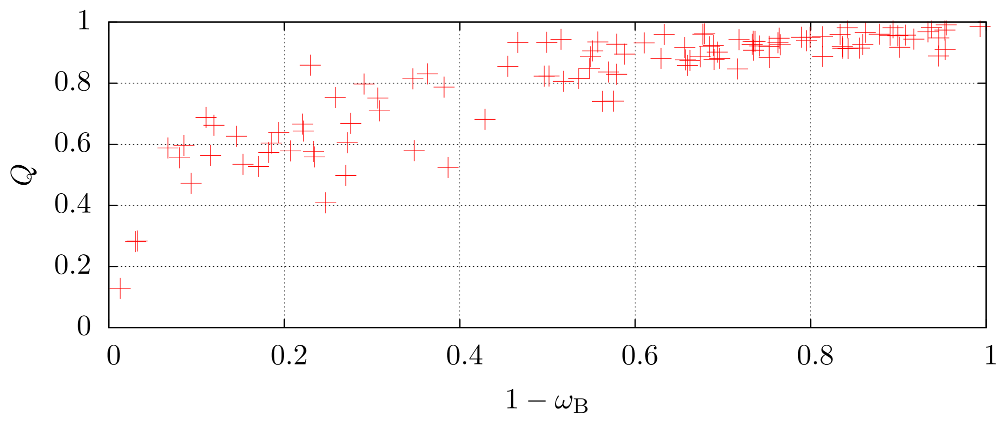


Figure 3.4: Q in relation to the signal weight $1-\omega_B$ for 110 sections containing peaks as illustrated in Kopczynski et al. (2012).

$[0,1]$; values towards 1 mean more accurate models. Sparse measurements with few peaks (and much background noise) are (globally) already well described by the uniform model; thus, we plot Q in relation to the weight $1 - \omega_B$ of the signal components in our model (Fig. 3.4), where ω_B is the weight of the uniform distribution, i.e., background model. For the evaluation, we manually extracted 110 sections containing peaks from Dataset 69 (described in Appendix A.2). For data sets that predominantly consist of peaks ($1 - \omega_B > 0.4$), almost all Q-values exceed 0.8.

3.5.2 Quality of the Pipeline

As introduced for PEAX, a sequence of consecutive processing steps is referred to as a pipeline. We compare the described pipeline with all remaining pipelines provided by PEAX. For this purpose, we use Dataset 69 again. Since there is a lack of single measurement annotations, a domain expert additionally pinpointed the peaks for every measurement within the dataset. By combining the provided modules, 108 individual pipelines can be set up. We name the pipelines by concatenating the shortcuts of the used modules in order. For example, the pipeline using (in this order) the modules Smoothing, De-Noising, Baseline Correction, Cross Finding, EM Clustering and No Modeling is named `s-dn-bc-lm-emc-e`. There are not 144 pipelines because of the redundancy between pipelines using the empty module as fourth step (for more details consider D'Addario et al. (2014)). Our PME pipeline has a particular name: `bc-dn-s-cf-emc-pme`.

To evaluate each pipeline, we compare the final obtained peak list with one which was manually annotated by an expert MCC/IMS development engineer. For the comparison we considered only peaks with a retention time exceeding 5 s and an IRM above 0.48 V s cm^{-2}, as is standard practice. Agreement of an automatically obtained peak list with that obtained by a domain expert is generally considered favorable. However, one should be

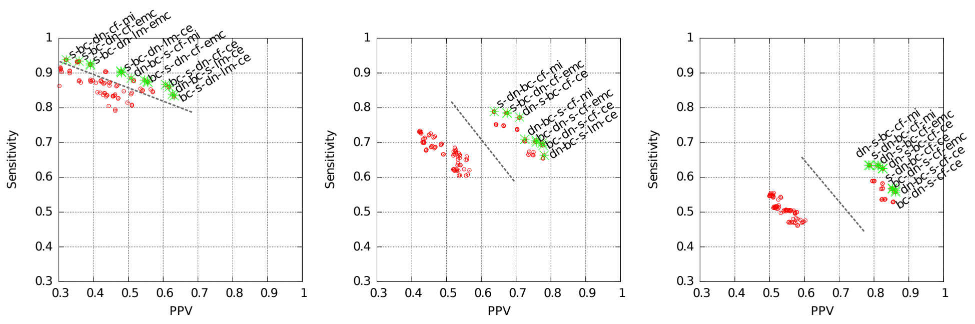


Figure 3.5: Comparing the results of all pipelines with the manually picked peaks according to sensitivity and precision (or PPV). The green crosses indicate the Pareto front. Left, middle and right figure correspond to signal intensity thresholds `noise_margin` $= 5, 10, 15$, respectively. The dashed lines separate two clusters of pipelines.

aware that manually annotated lists may also be incomplete or contain extraneous peaks. Nevertheless, assuming that the manually annotated peaks are correct, we compute the following quantities. Peaks detected by both methods, manual and automatic, within one measurement are true positives (TP). Accordingly, manually annotated peaks that are not detected by the pipeline are false negatives (FN) and automatically detected peaks not found in the manual annotation are false positives (FP). We compute the sensitivity SENS $:=$ TP$/($TP $+$ FN$)$ and the precision (or positive predictive value) PPV $:=$ TP$/($TP $+$ FP$)$. To determine these quantities for a particular pipeline, we average over all measurements of the dataset. It remains to define what it really means that "the same" peak has been detected by both methods, since the location parameters (ρ, τ) may differ slightly. All peak picking modules can be used for this decision, and we chose "Merging by Signal Intensity" (ms) as described by D'Addario et al. (2014). Imagine a box around every manually annotated peak (ρ, τ) of widths $2\Delta\rho$ and $2\Delta\tau$, respectively. Then we successively count each box containing at least one automatically detected peak and delete it. In case of two or more peaks within the box, we count the closest one.

Ranking the Pipelines

Figure 3.5 shows a plot of SENS against PPV for each pipeline for different parameter values of the signal intensity threshold $I := \sigma_{\mathrm{N}} \cdot$ `noise_margin`. The Pareto front is visualized in each plot. We find that, for the candidate detection step, almost all Pareto-optimal pipelines use Cross Finding (cf) whereas EM clustering (EMC) shows the best trade-off between sensitivity and PPV. For every signal intensity threshold I, the pipelines split into two groups. The first group has both relatively low sensitivity and low positive predictive value, while the second one has high values for both measures.

By modeling, the peak coordinates move slightly, yielding larger average differences to the manual annotation based on grid coordinates. However, the volume of a peak may contain important information (not evaluated here) which we cannot solely infer from the position and intensity at those coordinates. Concerning the threshold `noise_margin`, selecting a value of 10 yields superior results. We note that individual measurement properties (high or low noise, characteristic VOCs, etc.) were not taken into consideration for choosing the module parameters.

4 Online Peak Model Extraction

State-of-the-art software, like IPHEx by Bunkowski (2011), Visual Now presented by Bödeker et al. (2008) or PEAX introduced by D'Addario et al. (2014) only extract peaks when the whole measurement is available. A measurement may take up to 10 minutes because of the pre-separation of the analytes in the MCC. Our own PEAX software in fact defines modular pipelines for fully automatic peak extraction and competes favorably with a human domain expert doing the same work manually when presented with a whole MCC/IMS measurement. However, storing the whole measurement is neither desirable nor possible when memory or CPU power are restricted. To process a single (noise-reduced) IM spectrum $\mathcal{S} = (\mathcal{S}_t)$, it is deconvoluted into separate components described by statistical distribution functions. This is called *online peak extraction*. Then, storing the whole matrix becomes obsolete, a desirable property especially for resource-constrained embedded devices. In the following we first summarize online methods from literature. Afterwards, we explain in detail the three capital steps of our proposed method, namely extracting peak parameters from a single IM spectrum containing preprocessing, connecting one-dimensional peak models towards consecutive IM spectra and estimating parameters for a two-dimensional model for all peak chains as illustrated in Figure 4.1. Accordingly, we describe the online peak model estimation (OPME) method we previously introduced in Kopczynski and Rahmann (2014) and explain the enhancements under resource constraints.

4.1 Related Work

As described in Section 3.2, only a few offline automated peak extraction software products exist in literature. Here we describe two online methods: *peak detection by slope analysis* (PDSA) and *Savitzky-Golay Laplace-operator filtering Thresholding Regions* (SGLTR) are methods, in which the single IM spectrum (or a small constant consecutive subset) is processed immediately after capturing and then directly discarded again.

Peak Detection by Slope Analysis (PDSA)

One approach for detecting peaks is described by Egorov et al. (2013) and is referred to as *peak detection by slope analysis* (PDSA). It is an online algorithm, thus it discards every raw IM spectrum data directly after the analysis.

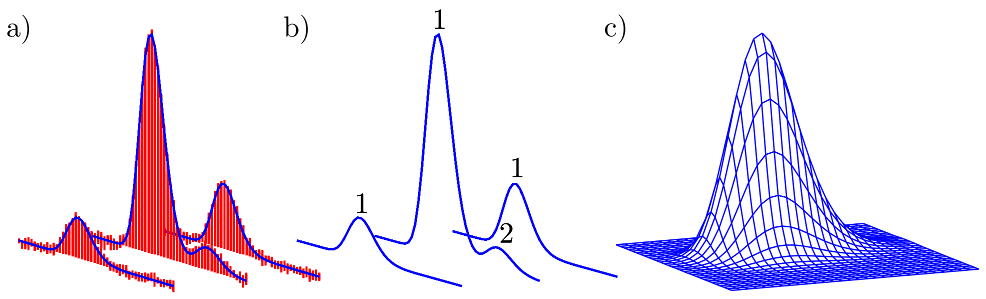


Figure 4.1: Demonstration of the three major steps in the online process: extracting peak parameters from a single IM spectrum and preprocessing (a); connecting one-dimensional peak models towards consecutive IM spectra (b); estimating parameters for two-dimensional model for all peak chains (c).

A moving window with a pre-defined width is sliding through the spectrum, searching for a specific pattern. For convenience, we call this pattern a run. When the sum of all values within the window exceeds a threshold computed on an area containing only noise, the run begins. In every iteration, the window is shifted one position ahead. It continues as long as the sum of a window from the previous iteration is smaller than the current. During the process, the difference between two sums (i.e. the slope) is being tracked. A change of the algebraic sign of the slope is allowed only once. When the sums are decreasing during the sliding until they drop below the threshold, the run ends and is accepted. Runs with more than one change of sign are discarded. For every run, the mode of the detected peak is stored. Runs are merged in retention time, when certain distance criteria stated by Hauschild et al. (2013) are fulfilled.

Savitzky-Golay Laplace-operator filtering Thresholding Regions (SGLTR)

The second detection method described by Egorov et al. (2013) utilizes the Laplace operator. For this, a set of consecutive IM spectra is processed at once. The Laplace operator is the sum of both second partial derivatives in retention time and IRM dimension. This value is a measure for the bending of the matrix section. A filtering kernel is moving through the set of spectra with tunable width and height, approximating the Laplace operator. To approximate the second derivative for the kernel, a Savitzky-Golay filter is utilized. Peaks can be inferred when the operator provides high values.

4.2 Extracting Peak Parameters from a Single Spectrum

Here we describe the process of extracting peaks from a single IM spectrum. After a spectrum is captured, it is still noisy and contaminated with the RIP and its tailing function. Thus, a preprocessing has to be performed to suppress the disturbing influences. Afterwards, we repeat three sub-steps to extract all peaks from a spectrum (from left to right):

1. scanning for a potential peak, starting where the previous iteration stopped;
2. determining peak parameters using an Inverse Gaussian distribution;
3. subtracting the peak from the spectrum and continuing with the remainder.

4.2.1 Preprocessing

The systematic error of each IM spectrum has to be suppressed to allow for a decent peak extraction. Again, both the baseline correction as well as a noise reduction have to be performed. In contrast to the two-dimensional preprocessing, it is not feasible to utilize the preprocessing methods described in Section 3.4.1, since these methods considered whole IMSC to lower the baseline. Thus, considering a single IM spectrum, three preprocessing steps have to be performed: *determining and suppressing the tailing function*, *denoising* and *baseline correction*. Accordingly, the suppression of the RIP and the tailing function was performed within the baseline correction step in the offline mode. Aggravating the situation, both shape and scale change from spectrum to spectrum without any recognizable relation. The scale of the RIP, for example, decreases significantly whereas the scale of the tailing function remains almost constant in several consecutive IM spectra. Hence, capturing a RIP-only spectrum and subtracting it from the current spectrum is not reasonable. For the consecutive preprocessing we claim the following assumptions during one measurement:

Assumption. Two consecutive IM spectra $\mathcal{S}, \mathcal{S}'$ containing no peaks produced by analytes are almost equal, hence $|\mathcal{S}'_t - \mathcal{S}_t| < 3\sigma_\mathrm{N}$ for all $t = 1, \ldots, |T|$. The RIP is both shifted within an IM spectrum in comparison to the RIP-only spectrum and scaled. The shape remains unchanged. Position and shape of the tailing function usually differ in contrast to RIP-only spectrum S_R. Since the analytes need a certain time to pass the drift tube, a particular number of the first T' signals at the beginning of every IM spectrum contains only noise. Thus, most signals within a spectrum have no amplitude, i.e. belong to the noise. The noise is normally distributed with mean μ_N and standard deviation σ_N.

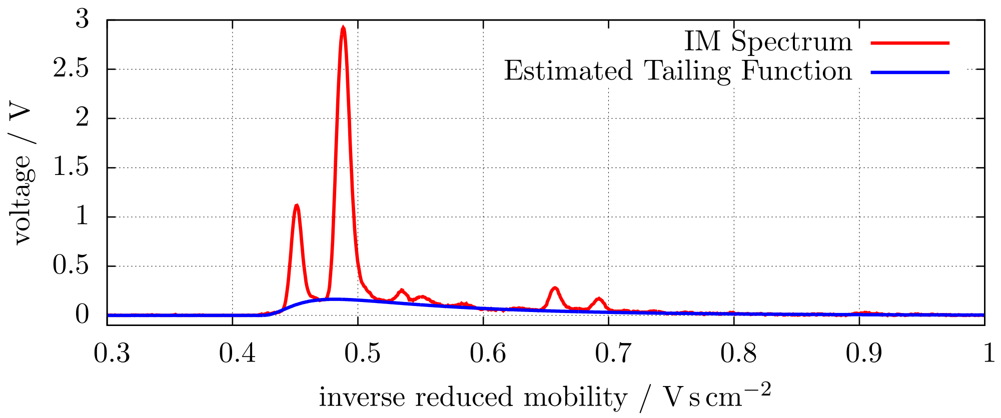


Figure 4.2: An exemplary single IM spectrum and its estimated tailing function.

Following this assumption, S_{R} has to be processed before any following spectra can be processed. This includes:

1. determining the parameters $\mu_{\mathrm{N}}, \sigma_{\mathrm{N}}$ of the noise level by considering the first T' signals of S_{R},
2. determining and subtracting the tailing function from S_{R} as described in the paragraph 'Determining the Tailing Function',
3. de-noising and setting the ground level to zero as explained in the paragraph 'De-Noising'.

The preprocessed RIP-only spectrum S'_{R} now only contains the RIP (as the name suggests) as well as peaks which appear in every following IM spectrum, e.g. the pre-RIP.

Determining the Tailing Function

Background The tailing function appears in every IM spectrum (see Figure 4.2 for an example). Its shape and scale change slightly from spectrum to spectrum but the shape always follows a right skewed peak form; hence it has to be determined anew in each spectrum and subtracted in order to extract peaks from the remaining signal in the further step. Empirically, we observe that the tailing function $f(\tau)$ can also be described by a scaled shifted Inverse Gaussian, $f(\tau) = v \cdot g(\tau \mid \mu, \lambda, o)$ with g given by Equation (3.1). The goal is to determine the parameters $\theta = (v, \mu, \lambda, o)$ such that $f_\theta(\tau)$ under-fits the given data $\mathcal{S} = S_{(r,\cdot)}$, as shown in Figure 4.2.

Related Work Bader et al. (2008) presented an approach using a log-normal distribution $\mathcal{L}(t)$ to model the tailing function. This model also has four parameters but one is fixed since they claim that the modes of both the RIP and the tailing function have the same IRM value. A penalty term was

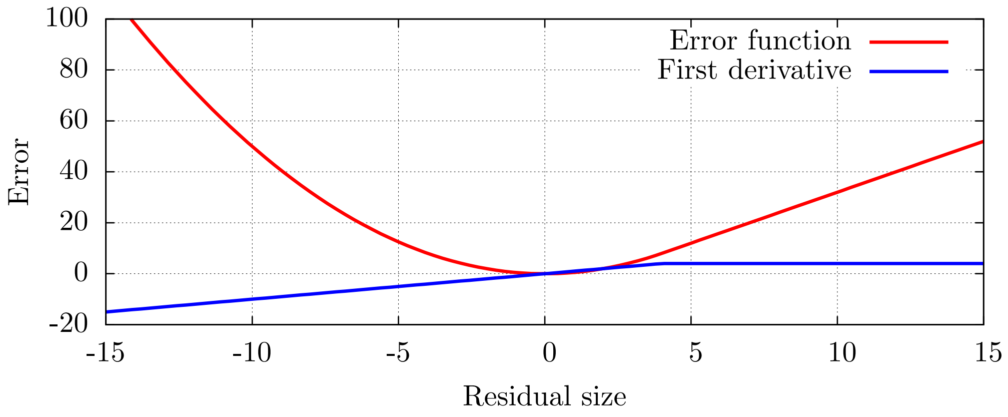


Figure 4.3: Error function $\mathbf{E}_t$ (red line) with $\gamma = 4$ and its first derivative (blue line).

developed considering the residual error of $\left(S^{\text{med}} - \mathcal{L}(t)\right)$ where S^{med} is a median spectrum (every data point is the median of row $S_{\cdot,t}$). This term is minimized by a quasi-Newton method. Since the median spectrum needs the whole IMSC to be computed, this approach is not applicable in an online context.

Parameter Estimation Let $\mathcal{R}_\theta(t) := \mathcal{S}_t - f_\theta(T_t)$ be the residual function for a given choice θ of parameters. As we want to penalize $\mathcal{R}(t) < 0$ but not $\mathcal{R}(t) > 0$, we use the following non-symmetric loss function which depends on a threshold parameter $\gamma > 0$:

$$\mathbf{E}_t(\theta;\gamma) := \begin{cases} \mathcal{R}_\theta(t)^2/2 & \text{if } r_\theta(t) < \gamma\,, \\ \gamma \cdot r_\theta(t) - \gamma^2/2 & \text{if } r_\theta(t) \geq \gamma\,. \end{cases}$$

That is, the loss at time t is the squared residual when it has a negative or low positive value less than the given threshold $\gamma > 0$, but becomes a linear function of the residual for larger positive residuals (consider Figure 4.3 for an example). The goal is to find the parameter set θ to minimize the total loss $L(\theta) := \sum_t \mathbf{E}_t(\theta;\gamma)$ for the given spectrum $\mathcal{S}$ and given $\gamma > 0$. We use gradient descent to solve this nonlinear optimization problem, to which we refer as *non-linear loss minimization* (NLLM). By modifying the computation of the parameter shift vector of the NLLS, we obtain the computation instructions in matrix notation

$$\begin{aligned} \Delta y &= y - f(x, \theta^k), \\ \Delta\theta &= (J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\mathbf{E}'(\Delta y, \gamma), \\ \theta^{k+1} &= \theta^k + \Delta\theta \end{aligned}$$

where $\mathbf{E}'$ is the derivative of $\mathbf{E}$. Since the derivative of $\mathbf{E}(x,\gamma)$ is continuous with respect to x, we can compute it as follows: $\mathbf{E}'(x,\gamma) = \max(x,\gamma)$. Here, the computation of the Jacobian matrix J remains unchanged. Deriving $f(\tau)$ with respect to all four parameters, we obtain the following instructions for J:

$$\begin{aligned}
J_{v,t} &= g(T_t \,|\, \mu, \lambda, o),\\
J_{\mu,t} &= v\frac{\lambda(x-o-\mu)}{\mu^3} \cdot g(T_t \,|\, \mu, \lambda, o),\\
J_{\lambda,t} &= 0.5v\left(\frac{1}{\lambda} - \frac{(T_t-o-\mu)^2}{\mu^2(T_t-o)}\right) \cdot g(T_t \,|\, \mu, \lambda, o),\\
J_{o,t} &= 0.5v\left(\lambda\left(\frac{T_t-o}{\mu^2} - \frac{1}{T_t-o}\right)+3\right) \cdot \frac{g(T_t \,|\, \mu, \lambda, o)}{x-o}
\end{aligned}$$

for every $t = 1, \ldots, |T|$.

Initial Parameters The initial parameter values (v, μ, λ, o) are determined as follows: For the scaling factor, we initially set $v = (1/2)\sum_{t\leq|T|} S_t$. For the other parameters, we first estimate the descriptors (μ', σ, m) as described below and then use the correspondence to the parameters listed in Section 3. The initial σ is set to the standard deviation of the whole RIP-only spectrum. We determine the initial m as the mode of the RIP. One property of the Inverse Gaussian distributions is that the mean μ' is always within the interval $I = [m, m + \mathbf{b} \cdot \sigma]$. Here, $\mathbf{b} = 0.71743893$ is the maximal factor for the relation $\mu' = \mathbf{b} \cdot \sigma + m$, yielding valid parameters for the inverse Gaussian. To compute $\mathbf{b}$, Equation (3.2) and (3.3) must be equated as: $p_1^2/4 = p_2$ with setting $\mu' = \mathbf{b} \cdot \sigma + m$ and solving for $\mathbf{b}$. To obtain an appropriate value for μ', an auxiliary offset variable o' is set to the largest IRM left of m where the signal is lower than σ_{R}, hence

$$o' = \max\{\tau = T_t \,|\, \tau < m \;\wedge\; |\mathcal{S}_t - \mu_{\mathrm{N}}| < \sigma_{\mathrm{N}}\}.$$

Accordingly, μ' is increased in increments within I. The candidate descriptors (μ', σ, m) are converted into corresponding parameters (μ, λ, o) until $o \geq o'$. These initial parameters are used to estimate the function parameters as described in the following paragraph.

Executing the Estimation To estimate the parameters for the tailing function, we

1. determine reasonable initial values for the parameters $\theta = (v, \mu, \lambda, o)$ as described in the previous paragraph,
2. solve NLLM with $\gamma = \sigma_{\mathrm{N}}^2$ to estimate the scaling factor v, leaving the other parameters fixed,
3. solve NLLM with $\gamma = \sigma_{\mathrm{N}}^2$ to estimate all four parameters,
4. solve NLLM with $\gamma = \sigma_{\mathrm{N}}^2/100$ to re-estimate the scaling factor v .

Final Step Having estimated $\theta = (v, \mu, \lambda, o)$, the tailing is subtracted from the spectrum, let

$$\mathcal{S}_t^* := \mathcal{S}_t - v \cdot g(T_t \,|\, \mu, \lambda, o)$$

for every $t = 1, \ldots, |T|$ at retention time index r. The obtained spectrum S^* is now devoid of the tailing function and we can proceed to the following preprocessing steps.

Baseline correction

As the consecutive part of the preprocessing for the single spectra, a baseline correction is performed. Since signals at a certain IRM can occur constantly among the whole retention time, they disturb the peak detection, since they could be recognized as a false positive peak. Here, we treat the RIP-only spectrum as a reference baseline which also has to be prepared for a suitable subtraction. The following processing steps are performed to align the preprocessed RIP-only spectrum $\mathcal{S}_\mathrm{R}$ to every remaining spectrum $\mathcal{S}$:

1. shifting $\mathcal{S}_\mathrm{R}$ so that the RIP modes of both spectra $\mathcal{S}_\mathrm{R}$ and $\mathcal{S}$ are aligned,
2. scaling $\mathcal{S}_\mathrm{R}$ so that the RIPs of both spectra $\mathcal{S}_\mathrm{R}$ and $\mathcal{S}$ have the same maximum height.

The aligned RIP-only spectrum $\mathcal{S}_\mathrm{R}^+$ is now suppressed from the actual spectrum and the completely preprocessed spectrum $\mathcal{S}^+$ is

$$\mathcal{S}_t^+ := \max\left\{\mathcal{S}_t - \mathcal{S}_{\mathrm{R},t}^+, 0\right\}$$

for all $t = 1, \ldots, |T|$.

De-Noising

As the last step in the preprocessing chain, the de-noising is described.

Background A major challenge during the peak detection in an IM spectrum is to find peaks which only slightly exceed the background noise level in a spectrum $\mathcal{S} = (\mathcal{S}_t)$. To determine whether the intensity $\mathcal{S}_t$ at coordinate t belongs to a peak region or can be solely explained by background noise, we propose a method based on the EM algorithm. It runs in $\mathcal{O}(\iota|T|)$ time where ι is the number of EM iterations.

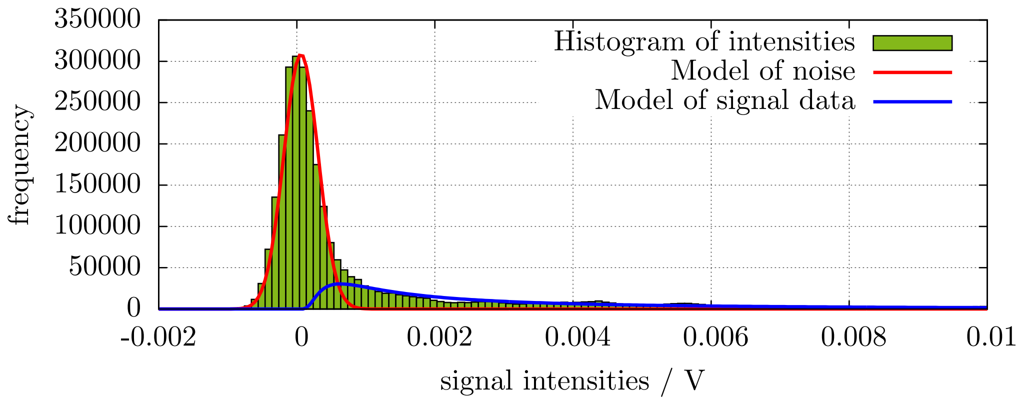


Figure 4.4: Histogram of signal intensities of an exemplary IM spectrum (green bars), an estimated distribution of the noise component (red line) and of the signal component (blue line). The parameters for both components were estimated with the EM algorithm.

Mixture Model Based on observations of IM spectra signal intensities, we assume that

- the noise intensity follows a Gaussian distribution over low intensity values with mean μ_{G} and standard deviation σ_{G},

$$p_{\mathrm{G}}(s \mid \mu_{\mathrm{G}}, \sigma_{\mathrm{G}}) = \frac{1}{\sqrt{2\pi}\,\sigma_{\mathrm{G}}} \cdot \exp\big(-(s-\mu_{\mathrm{G}})^2/(2\,\sigma_{\mathrm{G}}^2)\big),$$

- the true signal intensity follows an Inverse Gaussian distribution with mean μ_{S} and shape parameter λ_{S}, i.e.,

$$p_{\mathrm{S}}(s \mid \mu_{\mathrm{S}}, \lambda_{\mathrm{S}}) = \sqrt{\lambda_{\mathrm{S}}/(2\pi s^3)} \cdot \exp\big(-\lambda_{\mathrm{S}}(s-\mu_{\mathrm{S}})^2/(2\mu_{\mathrm{S}}^2 s)\big),$$

- there is an unspecific background component which is not well modeled by either of the two previous distributions; we model it by the uniform distribution over all intensities in IMSC S,

$$p_{\mathrm{B}}(s) = 1/(\max(S) - \min(S)),$$

  and we expect the weight ω_{B} of this component to be close to zero in standard IM spectra. High weights indicate an anomaly during the measurement.

We interpret the observed spectrum $\mathcal{S}$ as a sample of a mixture of these three components $\mathbf{c}$ with unknown mixture coefficients. To illustrate this approach, Figure 4.4 shows the empirical intensity distribution (histogram) of an exemplary IM spectrum with the estimated components. The uniform distribution is not shown, because the estimated coefficient is close to zero as expected. Finally, there are six independent parameters to estimate: μ_{G},

σ_G, μ_S, λ_S and weights $\omega_\mathrm{G}, \omega_\mathrm{S}, \omega_\mathrm{B}$ (for noise, signal and background, where $\omega_\mathrm{B} = 1 - \omega_\mathrm{G} - \omega_\mathrm{S}$).

Initial Parameter Values Background noise intensities are assumed to follow a Gaussian distribution with low intensity values. We can determine its approximate mean μ_G and standard deviation σ_G by considering the first and last 10% of data points in each spectrum since these areas typically contain no signals except noise.

The initial weight of the noise component is set to include most points covered by this Gaussian distribution, i.e.,

$$\omega_\mathrm{G} := \big|\{t \le |T| \,|\, \mathcal{S}_t \le \mu_\mathrm{G} + 3\,\sigma_\mathrm{G}\}\big| \;/\; |T|.$$

We assume that almost all of the remaining weight belongs to the signal component, thus $\omega_\mathrm{S} = (1 - \omega_\mathrm{G}) \cdot 0.999$, and $\omega_\mathrm{B} = (1 - \omega_\mathrm{G}) \cdot 0.001$. To obtain initial parameters for the signal model, $I' := \{t \le |T| \,|\, \mathcal{S}_t > \mu_\mathrm{G} + 3\,\sigma_\mathrm{G}\}$ is the complement of intensities initially assigned to the noise component. We set

$$\begin{aligned} \mu_\mathrm{S} &= \big(\sum_{t \in I'} (\mathcal{S}_t - \mu_\mathrm{G})\big)/|I'|, \\ \lambda_\mathrm{S} &= \big(\sum_{t \in I'} (1/(\mathcal{S}_t - \mu_\mathrm{G}) - 1/\mu_\mathrm{S})\big)^{-1} \end{aligned} \tag{4.1}$$

which are the maximum likelihood estimators for Inverse Gaussian parameters.

E-Step The hidden parameters $W_{t,\mathbf{c}}$ are computed using Equation (1.3), where the three component distributions $f_\mathbf{c}$ are the three component densities $p_\mathrm{G}, p_\mathrm{S}, p_\mathrm{B}$ with their parameters and the data x is a mean-smoothed version of the original spectrum $\mathcal{S}$:

$$x_t := \frac{1}{2\alpha + 1} \cdot \sum_{t'=t-\alpha}^{t+\alpha} S_{t'},$$

where the smoothing window width is $\alpha := f_\mathrm{ims} \cdot d_\mathrm{grid} \cdot |T| / \max(T)/\phi$. Here, d_grid is the grid opening time of the spectrometer and $\max(T)$ is the maximum IRM in T. Since the borders of a spectrum do not contain important information, we deal with boundary effects by computing x_t in those cases as averages of the existing spectrum entries only.

M-Step In the maximization step (M-step), we estimate the maximum likelihood parameters for the non-uniform components using the original intensities of S again for all $t = 1, \dots, |T|$:

$$\begin{aligned}
\mu_{\mathrm{G}} &= \frac{\sum_t W_{t,\mathrm{G}} \cdot \mathcal{S}_t}{\sum_t W_{t,\mathrm{G}}},\\
\mu_{\mathrm{S}} &= \frac{\sum_t W_{t,\mathrm{S}} \cdot (\mathcal{S}_t - \mu_{\mathrm{G}})}{\sum_t W_{t,\mathrm{S}}},\\
\sigma_{\mathrm{G}}^2 &= \frac{\sum_t W_{t,\mathrm{G}} \cdot (\mathcal{S}_t - \mu_{\mathrm{G}})^2}{\sum_t W_{t,\mathrm{G}}},\\
\lambda_{\mathrm{S}} &= \frac{\sum_t W_{t,\mathrm{S}}}{\sum_t W_{t,\mathrm{S}} \cdot (1/(\mathcal{S}_t - \mu_{\mathrm{G}}) - 1/\mu_{\mathrm{S}})}.
\end{aligned}$$

Final Step After convergence, we correct the ground level of the spectrum and remove noise: We first subtract μ_{G} from the signal value and then reduce the remaining value by the estimated noise weight. The corrected spectrum $\mathcal{S}^+$ is

$$\mathcal{S}_t^+ := (1 - W_{t,\mathrm{G}})(\mathcal{S}_t - \mu_{\mathrm{G}})$$

for all $t = 1, \dots, |T|$. The preprocessing is now finished for the spectrum, hence the peak detection is performed as a consecutive step. All preprocessing steps, i.e. suppressing the tailing function, baseline correction, de-noising and finished spectrum are illustrated in Figure 4.5.

4.2.2 Peak Detection

The algorithm scans for peaks, starting at the left end of the preprocessed spectrum $\mathcal{S}$ by sliding a window of a given width across $\mathcal{S}$ and fitting a quadratic polynomial model to the data points within the window. The width of the window (in index units) is related to the grid opening time d_{grid} of the spectrometer and given as $f_{\mathrm{ims}} \cdot d_{\mathrm{grid}} \cdot |T| / \max(T)$ data points. Let $f(\tau; \theta) = \theta_2\, \tau^2 + \theta_1\, \tau + \theta_0$ be the fitted quadratic polynomial inside the window. We call a window a *peak window* if the following conditions are fulfilled:

- the extreme IRM $\tau^* = \theta_1/(2\theta_2)$ lies within the IRMs of the window;
- the extreme IRM τ^* indicates a maximum (i.e., $\theta_2 < 0$);
- the maximum is sufficiently higher than the noise level (which is zero after preprocessing): $f(\tau^*; \theta) \geq \sigma_{\mathrm{N}}$.

The first condition can be more severely restricted to achieve more reliable results by shrinking the interval towards the center of the window.

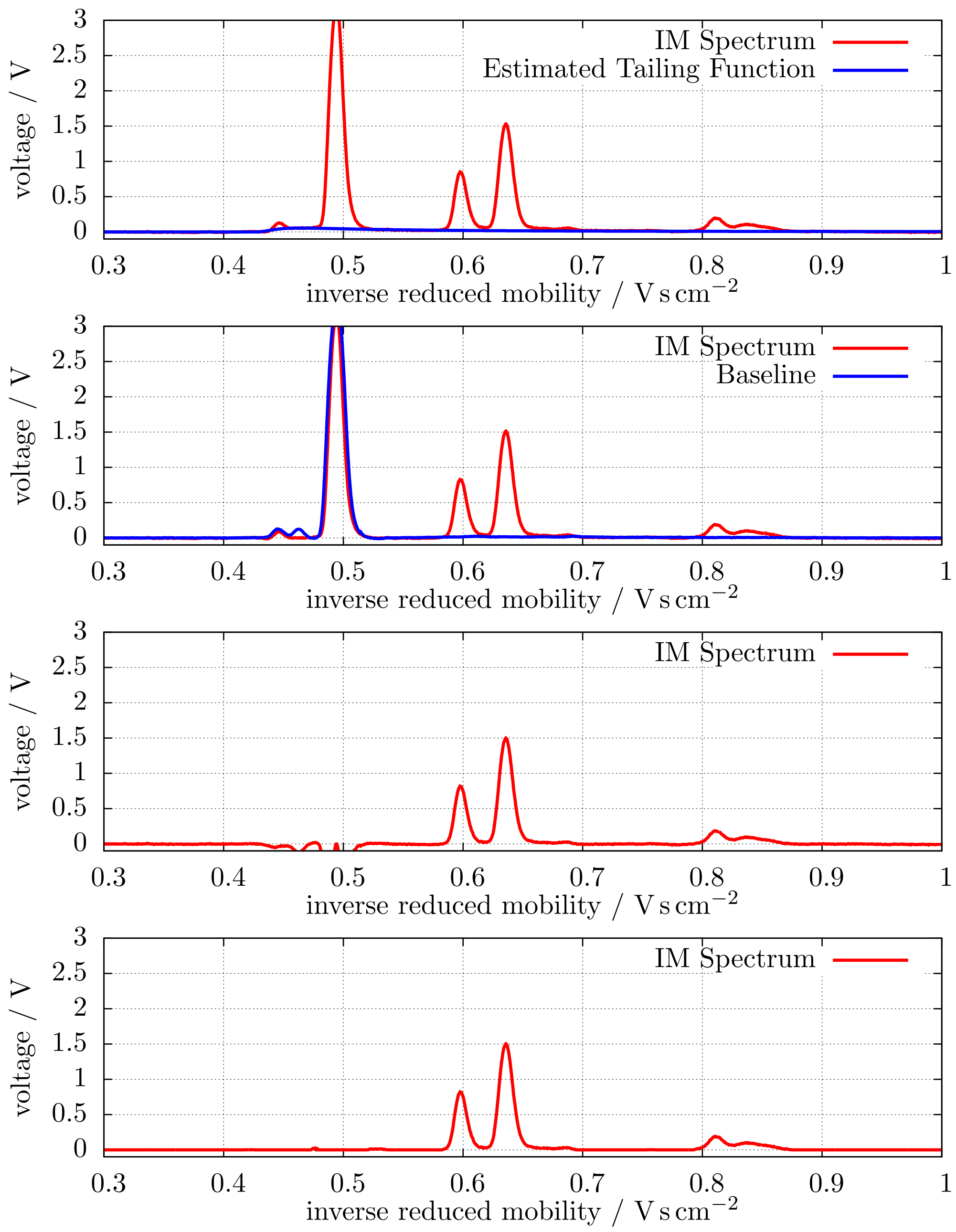


Figure 4.5: Exemplary process from a raw IM spectrum into a processed spectrum, showing all intermediate steps: (first) raw spectrum and the estimated tailing function; (second) suppressed spectrum and the aligned baseline; (third) baseline corrected spectrum; (fourth) de-noised / preprocessed spectrum.

If no peak is found, the moving window is shifted forward by one index. If a peak is detected, the peak parameters are computed. To avoid the detection of the same peak again when shifting forward by one index, the window is shifted by half its length and the next scan begins.

Determining Peak Parameters

As described in Section 3.3, we can estimate all peak descriptors (μ', σ, m) having the mode $m = \tau^*$ of a peak. We convert them into the parameters $\theta = (\mu, \lambda, o)$ of the Inverse Gaussian parameterization. The scaling factor v for the peak is $v = f(\tau^*; \theta)/g(\tau^*; \mu, \lambda, o)$.

The model function is subtracted from the spectrum and the next iteration is started with a window shifted by α index units (consider Section 4.2.1). For each spectrum, the output of this step is a *spectrum peak list*, which is a set of parameters for a mixture of weighted Inverse Gaussian models describing the peaks. As the next processing step, the behavior of peaks between spectra must be analyzed.

4.3 Aligning Consecutive Spectrum Peak Lists

4.3.1 Background

Having a set of peak parameters for each spectrum, the question arises how to merge the sets $P = (P_i)$ and $P^+ = (P_j^+)$ of two consecutive spectra. For each peak P_i (and for P_j^+, respectively), we have stored the Inverse Gaussian parameters μ_i, λ_i, o_i, the peak descriptors μ_i', σ_i, m_i (mean, standard deviation, mode) and the scaling factor v_i. The idea is to compute a global alignment between P and P^+, similar to the Needleman and Wunsch (1970) method. We need to specify how to score the aligning of P_i to P_j^+ and how to score leaving a peak unaligned (i.e., a gap).

4.3.2 Scoring Peak Alignments

The score ζ_{ij} for aligning P_i to P_j^+ is chosen by evaluating P_i's density function at the new mode m_j^+ and comparing it to the expected value which is an approximate standard deviation away from the mode at $m_i + \Delta\tau$, resulting in the log-odds score

$$\zeta_{i,j} = \ln \left(\frac{g(m_j^+;\, \mu_i, \lambda_i, o_i)}{g(m_i + \Delta\tau;\, \mu_i, \lambda_i, o_i)} \right).$$

Alternatively, leaving a peak unmatched results in a gap score of zero. Applying the Needleman-Wunsch global alignment, we can compute the optimal score of aligning the first i peaks in the latter spectrum with the first j peaks in the current spectrum by dynamic programming. We initialize a matrix Z, setting all $Z_{i,0}$ and $Z_{0,j}$ to zero and then compute, for $i \geq 1$ and $j \geq 1$,

$$Z_{i,j} = \max \begin{cases} Z_{i-1,j-1} + \zeta_{i,j}, \\ Z_{i-1,j}, \\ Z_{i,j-1}. \end{cases}$$

4.3.3 Obtaining Peak Chains

Once the matrix Z has been computed, the alignment is obtained by backtracing and recording the optimal case in each cell, as usual. There are three cases to consider.

- If P_j^+ is not aligned with a peak in P, a new peak starts potentially at this retention time. Thus, the peak P_j^+ is put into a new peak chain.
- If P_j^+ is aligned with a peak P_i, the chain containing P_i is extended with P_j^+.
- All peaks P_i that are not aligned to any peak in P^+ indicate the end of a peak chain at the current retention time.

A peak chain $\mathbb{P} = (P_1, \ldots, P_n)$ is now a set of one-dimensional Inverse Gaussian models. All completed peak chains are forwarded to the next step, two-dimensional peak model estimation.

4.4 Estimation of 2-D Peak Models

4.4.1 Background

The goal of this step is to estimate a two-dimensional peak model (product of two one-dimensional Inverse Gaussians) from the chain, as described in Section 3.3. Alternatively, rejection of the chain may be necessary if it does not fit such a model well. Potential problems are that peak chains i) may contain noisy 1-D peaks truncated at their borders, ii) consist only of noise or iii) in fact consist of several consecutive 2-D peaks at the same IRM and successive retention times.

4.4.2 Related Work

For the determination of the number of peaks within a peak chain in the second dimension of a two-dimensional spectrum, Vivó-Truyols (2012) introduced a Bayesian approach. Given a vector of heights $h = \{h_1, \ldots, h_n\}$ of 1-D peaks provided by a peak chain, a configuration $\mathbb{P}_i$ is determined as a

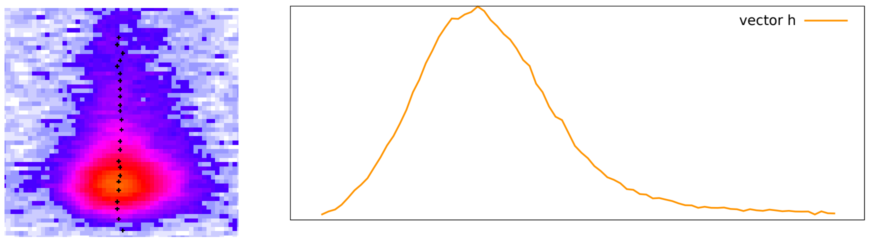


Figure 4.6: Exemplary assembly of height vector h (right) from the modes of the single 1-D models of an exemplary peak (left).

particular connection of the entries of h with each other. Every connected component within the chain marks a discrete peak. Obviously, connecting only consecutive entries is reasonable. For every configuration a conditional probability

$$p(\mathbb{P}_i \,|\, h) := p(h \,|\, \mathbb{P}_i)\frac{p(\mathbb{P}_i)}{p(h)}$$

is estimated where $p(h \,|\, \mathbb{P}_i)$ is the likelihood of h, given an assumed configuration $\mathbb{P}_i$. Furthermore, $p(\mathbb{P}_i)$ is the prior probability of having a certain configuration and $p(h)$ the probability of obtaining the current data h. This approach only holds when the configuration contains only one connected component. When assuming two peaks, the following equation has to be solved:

$$p(h \,|\, \mathbb{P}_i) = p(h_\mathrm{A} \,|\, \mathbb{P}_i, h_\mathrm{B})p(h_\mathrm{B} \,|\, \mathbb{P}_i),$$
$$p(h \,|\, \mathbb{P}_i) = p(h_\mathrm{A} \,|\, \mathbb{P}_i)p(h_\mathrm{B} \,|\, \mathbb{P}_i).$$

To avoid a combinatorial explosion of configurations, impossible cases in terms of analytics (e.g. h_1 is connected with h_3 skipping h_2) are discarded. Additionally, several conditions are stated, e.g. entries can only be merged if they are sufficiently close to each other.

4.4.3 Estimating the Parameters

As discussed in Section 3.3, the half-height width $\omega_{1/2}$ in retention time of a peak centered at retention time ρ can be described by an affine function $\xi(\rho)$, Equation (3.5), and $\omega_{1/2}$ can be converted to the corresponding number of data points (width of the window). For each individual peak $i = 1, \ldots, n$ in a peak chain, we know the parameters $(\hat{v}_i, \hat{\mu}_{i,\mathrm{T}}, \hat{\lambda}_{i,\mathrm{T}}, \hat{o}_{i,\mathrm{T}})$ and the corresponding descriptors $(\hat{\mu}'_{i,\mathrm{T}}, \hat{\sigma}_{i,\mathrm{T}}, \hat{m}_{i,\mathrm{T}})$, as well as the associated retention time ρ_i. The peak heights can be obtained computing $h_i = \hat{v}_i \cdot g(\hat{m}_{i,\mathrm{T}}; \hat{\mu}_{i,\mathrm{T}}, \hat{\lambda}_{i,\mathrm{T}}, \hat{o}_{i,\mathrm{T}})$, an exemplary assembly of the heights is demonstrated in Figure 4.6. We proceed by fitting quadratic polynomials $b(\rho; \theta) = \theta_2\rho^2 + \theta_1\rho + \theta_0$ in sliding windows of the appropriate width $\xi(\rho_i)$ to h such that $h_i \approx b(\rho_i; \theta)$ (similar to the scanning approach described in Section 4.2.2). To omit outliers within vector h during the fitting, we use the Huber loss function $\mathcal{L}$ and derivative $\mathcal{L}'$

$$\begin{aligned}\mathcal{L}(x,\gamma) &:= \begin{cases}\frac{1}{2}x^2 & \text{if } |x| < \gamma \\ \gamma \cdot |x| - \frac{1}{2}\gamma^2 & \text{else}\end{cases}, \\ \mathcal{L}'(x,\gamma) &:= \max\big(\min(x,\gamma), -\gamma\big)\end{aligned}$$

for the non-linear loss minimization. Hence, as the new computation instructions we obtain:

$$\begin{aligned}\Delta y &= y - f(x,\theta^k), \\ \Delta\theta &= (J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\mathcal{L}'(\Delta y,\gamma), \\ \theta^{k+1} &= \theta^k + \Delta\theta\end{aligned}$$

where $\gamma = \sigma_{\mathrm{N}}^2$ equals the variance of the background noise. Having found a window which fits a peak, we estimate initial descriptors for an Inverse Gaussian model in retention time as follows:

$$\begin{aligned}v'_{\mathrm{R}} &= -\theta_1^2/(4\theta_2) + \theta_0, \\ \sigma_{\mathrm{R}} &= \sqrt{v'_{\mathrm{R}}/(2|\theta_2|)}, \\ m_{\mathrm{R}} &= -\theta_1/(2\theta_2), \\ \mu'_{\mathrm{R}} &= m_{\mathrm{R}} + \xi(m_{\mathrm{R}})\,/\,(4\,\phi).\end{aligned}$$

The descriptors are then converted into model parameters.

After processing each window, we have obtained a list of k Inverse Gaussian distributions. We expect these distributions to be a mixture of k overlapping peaks in a single peak chain. To obtain an optimal deconvolution, we first normalize the volume factors v'_{R} of the k components to obtain $v_{\mathrm{R},j}$ such that $\sum_{j=1}^{k} v_{\mathrm{R},j} = 1$ and then apply the EM algorithm. As a byproduct, we obtain an $(n \times k)$ matrix $W = (W_{i,j})$ which determines the membership probability for each of the n data points (ρ_i, h_i) to each of the k models.

To obtain the Inverse Gaussian distribution parameters in the IRM dimension for each of the k models, we first compute model descriptors using a membership-weighted average over the individual model descriptors: For $j \in \{1, \ldots, k\}$, let

$$\begin{aligned}
\overline{W}_j &:= \sum_{i \leq n} W_{i,j}, \\
\mu'_{j,\mathrm{T}} &:= \frac{1}{\overline{W}_j} \sum_{i \leq n} W_{i,j} \cdot \hat{\mu}'_{i,\mathrm{T}}, \\
\sigma_{j,\mathrm{T}} &:= \frac{1}{\overline{W}_j} \sum_{i \leq n} W_{i,j} \cdot \hat{\sigma}_{i,\mathrm{T}}, \\
m_{j,\mathrm{T}} &:= \frac{1}{\overline{W}_j} \sum_{i \leq n} W_{i,j} \cdot \hat{m}_{i,\mathrm{T}}.
\end{aligned}$$

We then convert these descriptors back into model parameters (consider Section 3.3). The final peak volume is computed as $v^*_j = v'_{j,\mathrm{R}} \cdot \sum_{i \leq n} v_{i,\mathrm{T}}$.

For every model $j \in \{1, \ldots, k\}$, we check the following conditions:

- the width at half height in the retention time dimension has approximately the expected size (cf. Eqs. (3.4), (3.5)): $\xi(m_{j,\mathrm{R}})/2 \leq \sigma_{j,\mathrm{R}} \cdot \phi < 2 \cdot \xi(m_{j,\mathrm{R}})$,
- the maximum peak height exceeds the noise level: $v^*_j \cdot g(m_{j,\mathrm{T}}; \mu_{j,\mathrm{T}}, \lambda_{j,\mathrm{T}}, o_{j,\mathrm{T}}) \cdot g(m_{j,\mathrm{R}}; \mu_{j,\mathrm{R}}, \lambda_{j,\mathrm{R}}, o_{j,\mathrm{R}}) \geq \texttt{noise\_margin} \cdot \sigma_{\mathrm{N}}$,
- the Inverse Gaussian peak model g in retention time correlates well with its quadratic approximation b in a window around the mode in terms of the Pearson product-moment correlation coefficient

$$\mathbf{Pearson}(X, Y) = \frac{\sum_{i \leq n, j \leq m} (X_i - \overline{X})(Y_j - \overline{Y})}{\sqrt{\sum_{i \leq n} (X_i - \overline{X})^2} \sqrt{\sum_{j \leq m} (Y_j - \overline{Y})^2}}$$

  with $\overline{X}, \overline{Y}$ as the sample mean. More precisely, consider the window $M = [m_{j,\mathrm{R}} - \Delta\rho,\ m_{j,\mathrm{R}} + \Delta\rho]$, the model vector $G = g(\rho; \mu_{j,\mathrm{R}}, \lambda_{j,\mathrm{R}}, o_{j,\mathrm{R}})$ for $\rho \in M$ and the quadratic approximation vector $B = b_j(\rho; \theta)$ for $\rho \in M$, and test whether the Pearson correlation satisfies $\mathbf{Pearson}(G, B) \geq \texttt{min\_correlation}$ where $0 \leq \texttt{min\_correlation} \leq 1$ is a tunable parameter.

If all conditions are fulfilled, we have identified a 2-D peak model with its corresponding parameters $(v^*_j, \mu_{j,\mathrm{T}}, \lambda_{j,\mathrm{T}}, o_{j,\mathrm{T}}, \mu_{j,\mathrm{R}}, \lambda_{j,\mathrm{R}}, o_{j,\mathrm{R}})$. Otherwise the model is discarded.

4.5 Enhancement under Resource Constraints

In the following, several approaches are introduced which aim at the important notion of maintaining speed under resource constraints.

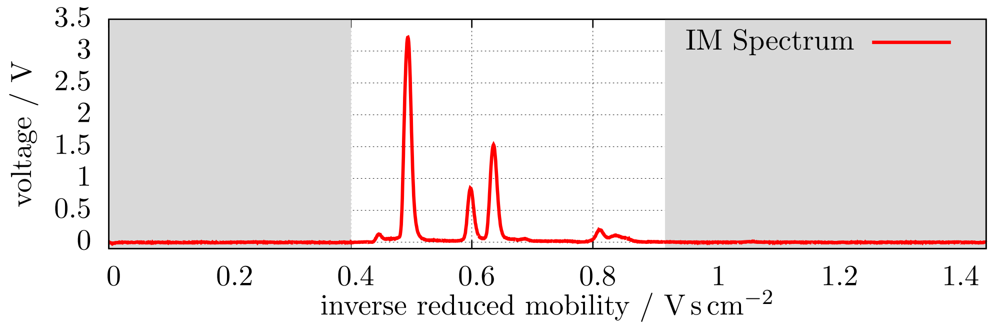


Figure 4.7: Example of an IM spectrum truncation. Only the middle part of the spectrum is being considered for the analysis, the remaining parts (gray) are being discarded.

Truncation of the Spectrum

A simple but effective method to reduce the number of computations is the truncation the spectrum. Having determined the parameters $\mu_\mathrm{N}, \sigma_\mathrm{N}$ for the background noise model, those parts of an IM spectrum $\mathcal{S} = (\mathcal{S}_t)$ can be truncated which consist solely of noise. This is achieved by computing the beginning (t_L) and ending (t_R) of the part of the spectrum which contains true signal information as follows:

$$t_\mathrm{L} := \min\left(t \,\middle|\, \mathcal{S}_t - \mu_\mathrm{N} < 2 \cdot \sigma_\mathrm{N}\right),$$
$$t_\mathrm{R} := \max\left(t \,\middle|\, \mathcal{S}_t - \mu_\mathrm{N} > 2 \cdot \sigma_\mathrm{N}\right).$$

Since the peaks have only positive values, the difference between $\mathcal{S}_t$ and μ_N is compared to σ_N and not to the absolute difference. Depending on the physical properties and adjustment of the MCC/IMS device, up to 30% of a spectrum may contain noise only and thus can be truncated. As an example, consider Figure 4.7.

Speedup of NLLS and NLLM

One bottleneck of the whole process is to suppress the tailing function by determining its model parameters. This is achieved by utilizing the non-linear loss minimization as described is Section 4.2.1. An advantage of choosing the shifted Inverse Gaussian distribution is the moderate requirement of only four parameters. With a fixed quantity of parameters, some optimizations like loop unrolling are possible. Instead of using a `for`-loop with a fixed number of iterations, all commands within this loop are stated explicitly. The advantage of loop unrolling is that the internal processor pipeline does not have to be interrupted at the end of every loop to decide whether to jump back to the beginning of the loop or to continue. Considering the computation step ($J^\mathrm{T} J$) within the NLLM, a normal matrix multiplication can be implemented with a triple-nested loop. Matrix $J^{\mathrm{T}} \in \mathbb{R}^{q \times n}$ and $J \in \mathbb{R}^{n \times q}$ both have a fixed $q = 4$.

```
inline float* matrix_mul_JTJ(const float* J, const int n){
    // matrix J = [J_0,0; J_0,1; J_0,2; J_0,3; . . .
    // J_n-1,0; J_n-1,1; J_n-1,2; J_n-1,3]
    float* const J_star = new float[16];
    // determine end of array
    const FTYPE* const matrix_end = J + (n * 4);
    // initializing the upper right triangle of the matrix
    J_star[0] = J_star[1] = J_star[2] = J_star[3] = 0;
    J_star[5] = J_star[6] = J_star[7] = 0;
    J_star[10] = J_star[11] = 0;
    J_star[15] = 0;
    // perform the J^T * J computation
    for(; J < matrix_end; J += 4){
        J_star[0] += J[0] * J[0];

        J_star[1] += J[0] * J[1];
        J_star[5] += J[1] * J[1];

        J_star[2] += J[0] * J[2];
        J_star[6] += J[1] * J[2];
        J_star[10] += J[2] * J[2];

        J_star[3] += J[0] * J[3];
        J_star[7] += J[1] * J[3];
        J_star[11] += J[2] * J[3];
        J_star[15] += J[3] * J[3];
    }
    // adding the symmetric values
    J_star[4] = J_star[1];
    J_star[8] = J_star[2];
    J_star[9] = J_star[6];
    J_star[12] = J_star[3];
    J_star[13] = J_star[7];
    J_star[14] = J_star[11];
    return J_star;
}
```

Listing 4.1: Efficient C++ code for matrix multiplication $J^* = (J^{\mathrm{T}}J)$.

The resulting matrix $J^* = (J^{\mathrm{T}}J) \in \mathbb{R}^{q \times q}$ will be a 4×4 matrix for which a single loop over $1, \ldots, n$ remains. A second feature of matrix J^* is its symmetry. Including the diagonal vector, only half of the values have to be computed. In this case six values can be omitted. Finally, it is more efficient to compute the matrix multiplication within one step instead of computing the transposed matrix J^{T} in a separate step. The efficient C++ code for matrix multiplication is shown in Listing 4.1. Due to its linear scanning of the array containing the linearized matrix J, this code is more cache efficient. Since J^* has 16 values, its inverse matrix $(J^*)^{-1}$ can also be efficiently computed with 108 arithmetic instructions.

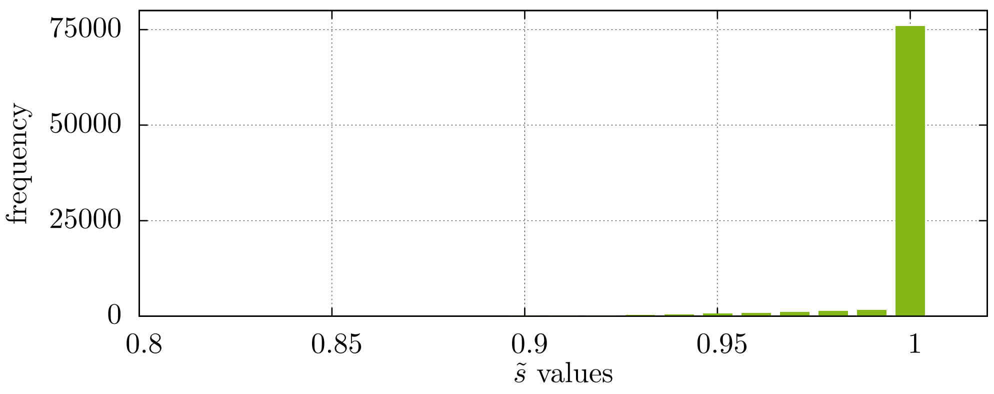


Figure 4.8: Histogram of weights $\tilde{s}$ between all 82 000 spectra and their related reduced spectra. Almost 92% of all reductions achieve an optimal weight.

Pre-computation

We can exploit that IRM array T has equidistant entries. Thus, when initializing OPME, several vectors or matrices can be precomputed once and used repeatedly as needed. For example, during the scanning (described in Section 4.2.2) a second order polynomial is fitted to a section of the complete spectrum within a moving window. The window size is invariant during this process. Since the Jacobian matrix J for a polynomial regression does not change, the intermediate matrix $(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}$ has to be computed only once. As a consequence, up to millions of computations can become obsolete, especially for high resolution measurements with 12 500 data points per IM spectrum and typically 6000 IM spectra. This idea cannot be implemented in retention time, since the moving window is linearly increasing. On the other hand, the number of polynomial regression steps is significantly decreased in R-dimension.

Remaining Optimizations

Some small but efficient optimizations are listed below:

- Since many methods use matrices (i.e. EM algorithm, NLLS and NLLM, aligning), the matrices are stored as linearized vectors. This offers the advantage that many methods iterate linearly through the matrices which is cache efficient and lowers the number of cache misses.

- If one divisor is used for several divisions, the inverse is multiplied instead. Within our tested hardware, a multiplication requires less computation time than a division.

Table 4.1: Average processing time in ms of de-noising, baseline correction and spectrum reduction on two platforms with different clock rates, averaging methods (single spectra, averages of 2 and 5 spectra) and convergence thresholds ε.

ε	Platform	average 1 / ms	average 2 / ms	average 5 / ms
	Desktop PC	4.36	2.09	0.88
0.1%	Rasp. Pi (700 MHz)	119.48	55.02	21.82
	Rasp. Pi (900 MHz)	97.19	43.62	17.42
	Desktop PC	4.26	2.01	0.66
1.0%	Rasp. Pi (700 MHz)	116.69	52.63	16.99
	Rasp. Pi (900 MHz)	94.03	41.46	13.48

- The scanning process according to Section 4.2.2 scans a spectrum once from left to right. It is not reasonable to update the complete spectrum when suppressing a detected peak. Hence, only the suffix of the spectrum starting at the moving window's position is updated.

4.6 Evaluation

Several aspects of the online peak extraction are evaluated including (1) the quality of the reduction of an IM spectrum, (2) the execution time as well as (3) the quality of automated peak extraction in comparison with manual offline annotation.

Parameters

For evaluation measurements, the BioScout was adjusted according to the suggested parameter values as listed in Section 1.3. We chose the following parameters:

- `r_width_offset` = 2.5 s (width offset for peaks in retention time),
- `r_width_factor` = 0.06 (width slope for peaks in retention time),
- `t_width_offset` = $0.003\,\mathrm{V\,s\,cm^{-2}}$ (standard deviation for peaks in IRM),
- `convergence_threshold` (convergence threshold; value varies within evaluation),
- `noise_margin` = 4 (factor multiplied with standard deviation of background noise for minimal peak height),
- `min_correlation` = 0.95 (minimal Pearson product-moment correlation coefficient).

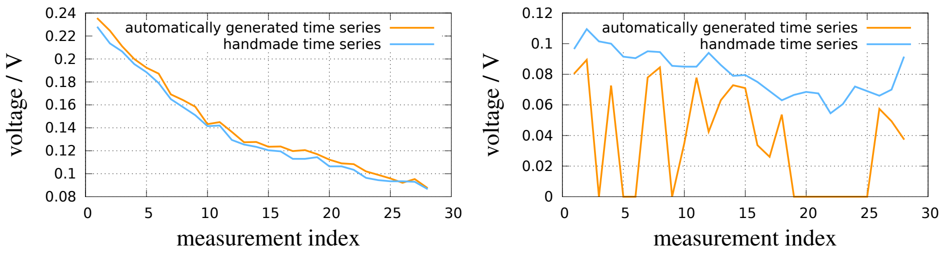


Figure 4.9: Time series of discovered intensities of two peaks. **Left:** A peak with agreement between manual and automated online annotation. **Right:** A peak where the online method fails to extract the peak in several measurements.

A determination of the best parameter values is presented in Section 9.

4.6.1 Quality of Single Spectrum Reduction

In a first experiment, we tested the quality of the spectrum reduction method using an approach by Munteanu and Wornowizki (2015). Its core idea is to determine the agreement between an observed set of data points, interpreted as an empirical distribution function F (the data) and a model distribution G (the mixture distribution obtained from the peak list parameters). The approach states that $F = \tilde{s} \cdot G + (1 - \tilde{s}) \cdot H$ with $\tilde{s} \in [0, 1]$, where H is a non-parametric distribution whose inclusion ensures the fit of the model G to the data F. If the weight $\tilde{s}$ is close to 1.0, then F is a plausible sample from G.

We compare the original spectra and reduced spectra (peaks from peak lists) from Dataset 69 (consider Appendix A.2). Each measurement contains 1 200 spectra. For all measurements, we computed the reduced spectrum model for each spectrum and determined $\tilde{s}$. Over 92% of all 82 000 models achieved $\tilde{s} = 1$ and over 99% reached $\tilde{s} \geq 0.9$. No $\tilde{s}$ dropped below 85%. In summary, spectrum reduction provides an accurate parametric representation of most spectra. Figure 4.8 illustrates that consideration of all 82 000 reductions leads to almost 92% achieving an optimal correlation of 100% of cases, in which the algorithm suggests to keep the distribution untouched.

4.6.2 Execution Time

Recall that each spectrum contains 12 500 data points. It is current practice to analyze not the full spectra, but aggregated ones, in which five consecutive values are averaged. Here we consider the full spectra, slightly aggregated ones (average over two values, 6 250 data points) and standard aggregated ones (average over five values, 2 500 data points). We measured the average execution time of de-noising, baseline correction and consecutive spectrum

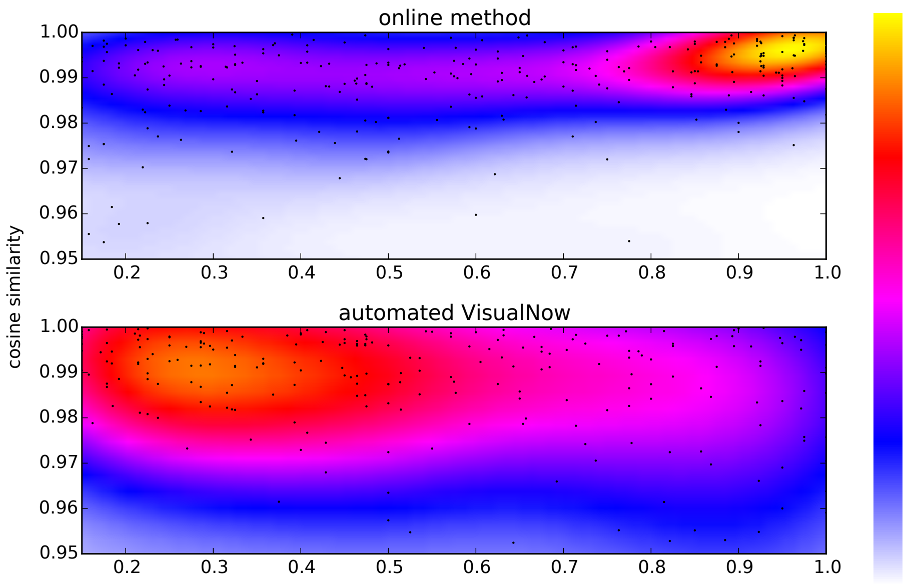


Figure 4.10: Kernel density estimation (kde) plots of recall and cosine similarity of peak intensity, comparing automatically picked peaks from our online algorithm and VisualNow against expert annotation. Each dot corresponds to the time series of one peak. Optimal results would be a recall of 1.0 and a cosine similarity of 1.0 for each time series.

reduction. The results are presented in Table 4.1. At the highest resolution (Average 1), the Raspberry Pi with 900 MHz barely keeps the time bound of 100 ms (provided by the measuring device) between consecutive spectra. At lower resolutions, the Raspberry Pi satisfies the time restrictions running about half the time boundary. The desktop PC effortlessly copes with the analysis on any setting. We found that in the steps using the EM algorithm, on average 25–30 EM iterations were necessary for a precision of $\varepsilon := 0.001$ (i.e., 0.1%) (see convergence in Section 1.5.3). Relaxing the threshold from 0.001 to 0.01 halved the number of required iterations without a noticeable difference in the resulting estimated parameters.

4.6.3 Comparison of Automated Online Peak Extraction with Manual Offline Annotation

The third experiment compares the quality of extracted peaks from a time series of measurements of two automated methods with an expert manual annotation. The automated methods are our online analysis process described here and automated peak detection using the commercial VisualNow (consider Section 3.2) software. For this experiment, we used the Rats dataset as described in Appendix A.2. To track peaks over time, we used the EM clustering method described in Section 5. As an example, Figure 4.9 shows time series of intensities of two peaks detected on the one hand by computer-assisted manual annotation and on the other hand using our online algorithm. The example illustrates the existence of cases with imperfect sensitivity of the online algorithm. This is mainly true for peaks whose intensity only slightly exceeds the background noise.

To obtain an overview over all time series, we computed the cosine similarity with bounds $[-1; +1]$ between the time series of peak intensities discovered by manual annotation and both automated methods. We also computed the recall for both automated methods for each time series, i.e. the relative fraction of measurements where the peak was found by the algorithm among those where it was found by manual annotation. Figure 4.10 shows an overall good agreement between both automated methods (our online method and automated VisualNow peak extraction) and the expert manual annotation. The cosine similarity of the inferred time series is in better agreement than the recall. When comparing the automated methods against each other, we outperform VisualNow in terms of sensitivity and computation time: About 31% of the points extracted by the online method exceed 90% recall and 98% cosine similarity, whereas only 5% of the time series extracted by VisualNow achieve these values. The peak detection of one measurement takes about 2 seconds on average (when the whole measurement is available at once) with the online method and about 20 seconds with VisualNow on the desktop computer as described in Section 4.6.2. VisualNow only provides the position and signal intensity of the peak's maximum, whereas our method additionally provides shape parameters. Problems of our online method stem from low-intensity peaks only slightly above the detection threshold, resulting in fragmentary or rejected peak chains.

5 Adaptive EM Clustering

We now consider a series of IMS measurements, for each of which we have extracted peaks available in the form of parameter vectors or descriptors. The question arises how to decide which descriptors in different measurements represent the same peak (and hence potentially the same VOC).

5.1 Background

Assuming that not a single IMSC is to be analyzed, but a time series or a data set of several entities, providing the same condition, e.g. patients suffering from the same disease, it is important to determine if any two peaks of different measurements have the same origin, e.g. are produced by the same analyte. In the peak clustering step, a set of n peak lists of a given data set is provided. The objective is to obtain an $n \times k$ matrix $\mathbf{D} = (\mathbf{D}_{i,j})$ with k consensus peaks where every cell stores the signal intensity of a certain peak contained in measurement i and consensus peak j. For convenience, matrix $\mathbf{D}$ is referred to as a data table.

5.2 Related Work

Several different clustering methods are known from literature. It is a necessary feature to dynamically determine the number of clusters and not set it as a fixed parameter. We briefly describe the manual peak picking and clustering (state-of-the-art method), k-means and the enhanced version k-means++ as the only clustering algorithm with a fixed cluster number, and DBSCAN, grid squares, cluster editing and hierarchical clustering as methods being independent of a predefined cluster number.

Manual Peak Detection and Clustering

Since the manual peak detection and clustering in VisualNow is one step, it is explained here once. Having a set of IMSC measurements, quadratic regions can be drawn over a visualization of a considered data matrix within the set. Every region corresponds to a cluster and therefore the regions must not overlap. For all measurements, the highest value within a region is being taken into consideration for a cluster.

K-Means

Given a set of n observation $\{x_1, \ldots, x_n\}$ with $\mathcal{D}$ dimensions for each, k-means partitions these observations into k clusters $\{P_1, \ldots, P_k\}$ with centers $\{m_1, \ldots, m_k\}$. The value k is fixed and will not be altered during the processing. The original k-means approach was first introduced by Lloyd (1982) for quantizations in pulse-code modulations, where the objective function

$$\min \sum_{j=1}^{k} \sum_{x \in P_j} ||x - m_j||^2$$

is solved with a heuristic approach. The algorithm comprises three steps:

1. First, choose k observations randomly as cluster centers.
2. Update the partitions, let $P_j := \{x \mid ||x - m_j||^2 < ||x - m_{j'}||^2 \text{ for all } j' = \{1, \ldots, k\} \setminus \{j\}\}$ for all $j \leq k$.
3. Update the centers, let $m_j := |P_j|^{-1} \sum_{x \in P_j} x$ for all $j \leq k$.

Repeat steps 2-3 until no observation is reassigned to another partition.

K-Means++

The enhanced version of k-means is k-means++ with an improved first step as described by Arthur and Vassilvitskii (2007). Choosing the first or random k observations as cluster centers can lead to disadvantageous initial positions of the centers. The k-means++ approach first picks one random observation as a cluster center. All remaining observations are weighted. Let $\delta_i := \min\{||x_i - m_j||^2 \mid j \leq k'\}$ be the minimal distance of observation x_i to all already picked k' cluster center. Hence, let $w_i := \delta_i / \sum_{i'=1}^{n} \delta_{i'}$ be their weight. The next observation is randomly picked as cluster center with a probability according to its weight. The weighting and picking steps are repeated until k cluster centers have been picked.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Introduced by Ester et al. (1996), DBSCAN is the most frequently used clustering algorithm in the application field of data mining. Starting with a random point p, the algorithm considers all neighbors of p dropping below a tunable distance ϵ. When the number of neighbors exceeds a second tunable threshold `minPts`, p is put into a new cluster or otherwise labeled as noise. Continuing with the neighbors, their neighbors are determined and put into the same cluster when `minPts` is exceeded. As long as observations can be reached according to the ϵ distance, the observations are put into the same cluster. When no remaining unvisited point can be reached, the cluster gets closed and the algorithm starts again with an unvisited point. All observations that do not fulfill the `minPts` criterion are labeled as noise. DBSCAN has the property to cluster points without any knowledge about a possible underlying distribution of the points building a cluster. Thus, no features like centroids are provided in the original approach.

Grid Squares

A naive method is the so-called *grid squares*, the only method with linear time complexity according to the number of detected peaks over a set of measurements. These grid squares (or hypercubes having dimension $\mathcal{D} > 2$) are spanned over a unified point set. One grid square equals one cluster and basically has a fixed width and height. In a more complex scenario, the height is linearly increasing depending on the retention time stated by Hauschild et al. (2013). For every point, its corresponding grid square can be computed in constant time. Empty grids or grids with too few contained peaks are discarded from consideration. For every sufficient grid, all contained points are considered to be a cluster.

Cluster Editing

An instance of a weighted cluster editing problem is solved to find clusters within a point set, as examined by Rahmann et al. (2007) and Böcker et al. (2011). Having a symmetric weight function, every pair of points gets a weight assigned which corresponds to an weighted edge in a graph with the peaks as vertices. The objective is to find a set of disjointed cliques by eliminating a subset of edges with minimal costs. All points building a clique are treated as a cluster.

Hierarchical Clustering

Often used in data mining or statistics, *hierarchical clustering* is a method to construct a dendrogram where the leaves contain the points and the inner nodes mark a conjunction of all descendant leaves. Two main techniques to construct a dendrogram are agglomerative or a divisive construction. Taking the agglomerative method, two nodes (having no ancestor yet) with the smallest distance are connected by becoming the successors of a newly inserted node. This process is repeated as long as at least two nodes do not have any ancestor or the distance between any pair of nodes exceeds a given threshold. Common distance metrics are Euclidean distance, Manhattan distance or Mahalanobis distance. The challenge is to determine a distance $\delta(x, y)$ between an inner node containing several points and another node. Several linkage criteria comparing two sets of points X, Y are

- complete-linkage clustering: $\max\{\delta(x,y) \mid x \in X, y \in Y\}$,
- single-linkage clustering: $\min\{\delta(x,y) \mid x \in X, y \in Y\}$,
- average-linkage clustering: $\frac{1}{|X|\,|Y|} \sum\limits_{x \in X} \sum\limits_{y \in Y} \delta(x,y)$.

Purkhart et al. (2012) utilized the hierarchical clustering to distinguish potential biomarkers from environmental volatile compounds in exhaled air.

5.3 Algorithm

In the following, we explain the method of dynamically adjusting EM clustering as already introduced in Kopczynski and Rahmann (2015) and introduce some consecutive developments to enhance the tool for a resource-constrained context.

Let X be the union of peak locations in all measurements, let $n := |X|$, and let $X_{i,\mathrm{R}}$ be the retention time of peak i and $X_{i,\mathrm{T}}$ its IRM. We introduce a clustering approach using the EM algorithm with two-dimensional Gaussian mixtures that differs from the standard approach in its ability to dynamically adjust the number of clusters in the process.

5.3.1 Mixture Model

We assume that the measured retention times and IRMs belonging to peaks from the same compound are independently normally distributed in both dimensions around the (unknown) true retention time and IRM. Let $\theta_j := (\mu_{j,\mathrm{R}}, \sigma_{j,\mathrm{R}}, \mu_{j,\mathrm{T}}, \sigma_{j,\mathrm{T}})$ be the parameters for component j, and let $f_j(x' \mid \theta_j)$ be a two-dimensional Gaussian product distribution for a peak location $x = (x_\mathrm{R}, x_\mathrm{T})$ with $x \in X$ containing these parameters. The mixture distribution is $f(x) = \sum_{j=1}^{c} \omega_j \, f_j(x \mid \theta_j)$ with a yet undetermined number c of clusters. Note that there is no "background" model component.

5.3.2 Initial Parameter Values

In the beginning we initialize the algorithm with as many clusters as peaks, i.e., we set $c := n$. This assignment makes a background model obsolete, because all peaks are assigned to at least one cluster. As start parameters for $\mu_{j,\mathrm{R}}, \mu_{j,\mathrm{T}}$, all clusters get the original retention time and IRM of peak location X_j, respectively, for $j = 1, \dots, n$. We set $\sigma_{j,\mathrm{T}} := \Delta\tau > 0$ and $\sigma_{j,\mathrm{R}} := \Delta\rho$ where $\Delta\tau$ and $\Delta\rho$ are widths as introduced in Section 1.4. The weights are equally distributed, thus $\omega_j := 1/c$.

5.3.3 Dynamic Adjustment of the Number of Clusters

After computing weights in the E-step, but before starting the M-step, we dynamically adjust the number of clusters by merging clusters whose centroids are close to each other. Every pair $j < k$ of clusters is compared in a nested for-loop. Let $f_{j,k} = f((\mu_{k,\mathrm{R}}, \mu_{k,\mathrm{T}}) \,|\, \theta_j)$ be the probability density that the model of cluster j describes the centroid position of cluster k. Furthermore, let $\tilde{f}_j((\mu_{j,\mathrm{R}} - \sigma_{j,\mathrm{R}}, \mu_{j,\mathrm{T}} - \sigma_{j,\mathrm{T}}) \,|\, \theta_j)$ the probability density that the model of cluster j describes its elliptical standard deviation ring. When $f_{j,k}/\tilde{f}_j \geq 1$ is satisfied, the centroid of cluster k ranges within the standard deviation of cluster j. W.l.o.g. if $\mu_{j,\mathrm{R}} < \mu_{k,\mathrm{R}}$, the condition $f_{k,j}/\tilde{f}_k \geq 1$ will be considered. Hence, clusters j and k are merged by summing the EM weights: $\omega^+ := \omega_j + \omega_k$ and $W_{i,+} := W_{i,j} + W_{i,k}$ for all $i = 1, \ldots, n$. Real case scenarios offer the empirical finding, that there is a correlation between the peak intensity and the distance of the peak to its cluster center. Therefore, the summed weights are assigned to the location of the cluster with its higher original peak intensity. Thus, the intensity over all peaks within a cluster is stored and if a merging occurs, the lower intensity cluster is deleted and the higher intensity cluster remains at its current position and gets the weights w and membership weights W assigned from the deleted cluster. The re-computation of the parameters is executed immediately after merging in the maximization step. The order of comparisons may matter in rare cases for the decision which peaks are merged first, but since new means and variances are computed, possible mergings omitted in the current iteration will be performed in the next iteration.

5.3.4 Maximum Likelihood Estimators

The maximum likelihood estimators for the mean and variance of a two-dimensional Gaussian are the standard ones, taking into account the membership weights,

$$\mu_{j,\mathbf{d}} = \frac{\sum_{i=1}^{n} W_{i,j} \cdot X_{i,\mathbf{d}}}{\sum_{i=1}^{n} W_{i,j}}, \qquad \mathbf{d} \in \{\mathrm{T}, \mathrm{R}\}, \tag{5.1}$$

$$\sigma_{j,\mathbf{d}}^2 = \frac{\sum_{i=1}^{n} W_{i,j} \cdot (X_{i,\mathbf{d}} - \mu_{j,\mathbf{d}})^2}{\sum_{i=1}^{n} W_{i,j}}, \qquad \mathbf{d} \in \{\mathrm{T}, \mathrm{R}\}, \tag{5.2}$$

for all components $j = 1, \ldots, c$.

One problem using this approach emerges from the fact that each cluster initially only contains one peak, leading to an estimated variance of zero in many cases. To prevent this, minimum values are enforced such that $\sigma_{j,\mathrm{T}} \geq \Delta\tau$ and $\sigma_{j,\mathrm{R}} \geq \Delta\rho$ for all $j = 1, \ldots, c$.

5.3.5 Final Step

The EM loop terminates when no merging occurs and the convergence criteria for all parameters are fulfilled. The resulting membership weights determine the number of clusters as well as the membership coefficient of peak location X_i to cluster j. If a hard clustering is desired, the merging step has to be traced.

5.4 Enhancement under Resource Constraints

One major challenge and the decisive bottleneck of this method is to estimate the hidden membership variables in the E step and the merging phase, since in theory, both methods need $\mathcal{O}(cn)$ runtime in every iteration, where n is the number of points and c the number of clusters. Actually, in the first iteration both algorithms need $\mathcal{O}(n^2)$, since the algorithm starts with $c := n$. In the worst case, all points are superposed, but on average the points build separable clusters with a certain distance to each other. Considering both steps, a probability density between two clusters (merging) or cluster and point (E step) is computed, respectively. Hence, it is reasonable to use a clever preselection of clusters (which is referred to as range searching) when considering a certain cluster during the merging or data point in expectation step.

5.4.1 Related Work

We claim that an appropriate data structure for the preselection has the following features:

- using an index or having sublinear point access time,
- being capable for updates, i.e. mergings or point shifts,
- using little additional memory.

Several approaches have been examined:

Fast Gauss Transform

The evaluation of sums of multivariate Gaussian kernels is referred to as discrete Gauss transform G and has the computational complexity of $\mathcal{O}(mn)$ with n source points $x = (x_i)$ and m target points $y = (y_j)$ with $x_i, y_j \in \mathbb{R}^{\mathcal{D}}$.

Given an additional vector of coefficients $\mathbf{q} = (\mathbf{q}_j)$ depending on y_j, G is defined as follows:

$$G(x_i) := \sum_{j}^{n} \mathbf{q}_j \mathrm{e}^{-||x_i - y_j||^2/h} \qquad \text{for all } i = 1, \ldots, n.$$

Greengard and Strain (1991) introduced an approach called *fast Gauss transform* with a time complexity of $\mathcal{O}(n + m)$. Since the standard deviation h is a fixed value over all dimensions within all source and target points, their idea is to split the space into $\mathcal{D}$-dimensional hypercubes of side length $r'\sqrt{2h}$ with $r' < 1$. Additional computation is saved by using the box centers and computing coefficients that are equal for every source point within a box. Thus these coefficients have to be computed once only. Although the asymptotic computation time is low, this approach was not adopted, since the computation time can only be reached when the points are equally distributed and the variance h^2 is the same for every dimension.

Delaunay Triangulation

Another approach is to connect any point within a point set with its nearest neighbors with the restriction that the connections are not allowed to cross each other. One solution is a triangulation as described by Delaunay (1934). This method spans a grid of triangles with the point set as the vertices for every triangle by maximizing the minimum angle of all angles within all triangles. More informally: the choice of slim triangles shall be omitted. Several methods are implemented, utilizing common techniques, i.e. incremental, divide and conquer or sweep line. These methods have a runtime of $\mathcal{O}(n \log(n))$ in common. An incremental method is to span a triangle covering all points, inserting every point p by determining in which triangle $\overline{abc}$ it is in, splitting into triangles $\overline{abp}, \overline{apc}, \overline{pbc}$ and performing recursively edge flips, when the circumcircle condition (i.e. no interior points) of the new triangles is violated. This approach provides the best pre-separation of points but the maintaining of this graph is costly after a merging of two points or a point shift.

Range Tree

A *range tree* is a multi-dimensional, balanced binary search tree and was introduced by Bentley (1979). For the first dimension, a balanced binary search tree is constructed in which the leaves store one point each and the inner nodes store the values of its leftmost and rightmost descendant leaf. Furthermore, each inner node is a root to a further binary search tree, sorted according to the next dimension and still containing all its descendant leaves. Over all, the total construction time as well as the storage have a complexity of $\mathcal{O}(n \log^{\mathcal{D}-1}(n))$ and a query needs $\mathcal{O}(\log^{\mathcal{D}}(n) + k)$ where k is the number of hits. Due to the rather complex structure of the multi-dimensional tree, maintaining the tree is also costly when updates are allowed. Although Chazelle (1990) reduced the storage complexity to $\mathcal{O}\left(n\left(\frac{\log(n)}{\log(\log(n))}\right)^{\mathcal{D}-1}\right)$ and query time to $\mathcal{O}(\log^{\mathcal{D}-1}(n)+k)$ using fractional cascading amongst others, the maintaining still remains challenging after updating.

5.4.2 Utilized Methods and Approaches

In the following section, several methods are described which helps to achieve a better computation time without loosing too much precision.

Dynamical Adjustment

Since only the dynamical adjustment step can reduce the number of clusters, this step is performed at the beginning of every EM iteration and not during it. The advantage is, that the consecutive E step does not have to consider all beginning clusters. Especially during the first iteration, many clusters can exist due to a high number of peaks.

Range Searching

Before the clustering starts, all n clusters are sorted with respect to their IRM value. Choosing the IRM dimension as the indexed measure is advantageous as its variance for peaks remains constant (consider Section 1.3). Let $\mathcal{C}$ be the sorted cluster list and $\mu_{\mathcal{C},j,\mathrm{T}}$ the centroids IRM value of the cluster $j = 1, \ldots, c$ within $\mathcal{C}$. When performing the dynamical adjustment, the tolerance interval $[j_\mathrm{L}, j_\mathrm{R}]$ determines the position of the minimum and maximum IRM of remaining clusters that have to be considered for cluster j, let

$$\begin{aligned} j_\mathrm{L} &:= \min\left(j' \,\middle|\, \mu_{\mathcal{C},j',\mathrm{T}} - \mu_{\mathcal{C},j,\mathrm{T}} \geq -\beta \cdot \sigma_j\right), \\ j_\mathrm{R} &:= \max\left(j' \,\middle|\, \mu_{\mathcal{C},j',\mathrm{T}} - \mu_{\mathcal{C},j,\mathrm{T}} \leq \beta \cdot \sigma_j\right) \end{aligned}$$

where β is a tolerance factor and can be set to $\beta = 4$ in practice (note that that $3\,\sigma$ covers almost the complete half of a Gaussian curve). Since the clusters are sorted, the update of the interval can be performed by simply increasing both values $j_\mathrm{L}, j_\mathrm{R}$ as long as their conditions are violated. This kind of range searching corresponds to a moving window and can be performed in $\mathcal{O}(c)$.

Within the E step, the position of both peaks and clusters is compared to each other. To find all sufficient clusters for a peak, two binary searches are performed to find $j_\mathrm{L}, j_\mathrm{R}$. The membership variables $W_{i,j}$ for peak i are computed considering only the clusters $j_\mathrm{L} \leq j \leq j_\mathrm{R}$. All remaining variables $W_{i,j'}$ with $j < j_\mathrm{L}, j_\mathrm{R} < j$ are set to zero. The memory consumption is also decreased, since cluster j stores only a list of the tuples $(i, W_{i,j})$ with $i \subseteq \{1, \ldots, n\}$ indexing the data points for the consecutive M step.

Probability Density Function (pdf) of a Multivariate Normal Distribution

The pdf of a multivariate normal distribution also contains an exponential computation. The approximate exp function (introduced in Section 2) can be enhanced to reduce the number of atomic computations and the computation time for a bivariate Gaussian bell as in our case.

Computation of the Variance

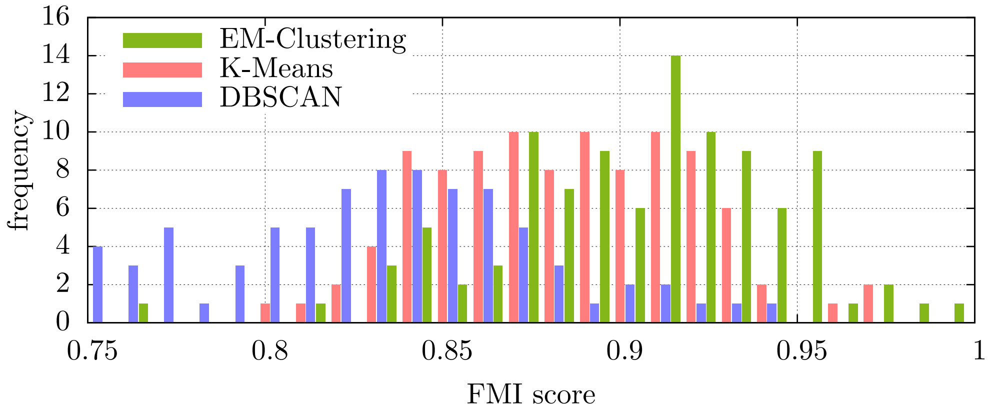
It is inevitable to estimate the weighted mean of every cluster using the maximum likelihood estimators. For the variance, however, we use the already predefined standard deviations according to the peak width observations. Thus, having computed $\mu^*_{j,\mathbf{d}}$ for $\mathbf{d} \in \{\mathrm{R}, \mathrm{T}\}$ and $j = 1, \ldots, C$, the standard definitions are assigned as follows:

$$\begin{aligned} \sigma^*_{j,\mathrm{T}} &:= \Delta\tau, \\ \sigma^*_{j,\mathrm{R}} &:= \Delta\rho. \end{aligned}$$

We empirically observed, that the restriction of cluster variances provides significantly improved results, since close clusters are not merged too quickly and definite singleton clusters remain singletons.

Maintaining the sorted Cluster List

After the M step, the means of the cluster have eventually shifted and thus the order of the cluster list could be violated. A solution is the utilization of a bubble sorting approach. When the parameters of cluster j within $\mathcal{C}$ are maximized, the cluster is compared with its predecessor $j-1$ and swapped as long as $\mu_{\mathcal{C},j-1,\mathrm{T}} > \mu_{\mathcal{C},j,\mathrm{T}}$. A comparison with its successor is not necessary, since the successor will be compared to its predecessor after the update. Of course an upper bound of this method results in $\mathcal{O}(N^2)$ comparisons. Thus we describe the tendency of the number of swaps with increasing EM iterations: When all cluster centroids overlap, $N-1$ mergings are performed within the first adjustment step; hence only one cluster remains and no swap can occur. Now we (need to) consider the general case with suggested distribution of the data points. A merging is only possible when two cluster centroids are too close to each other. When a merging happens, the new cluster centroid is located in the center of the new cluster and in between of both former centroids, because both former clusters were convex. Thus, the distances of



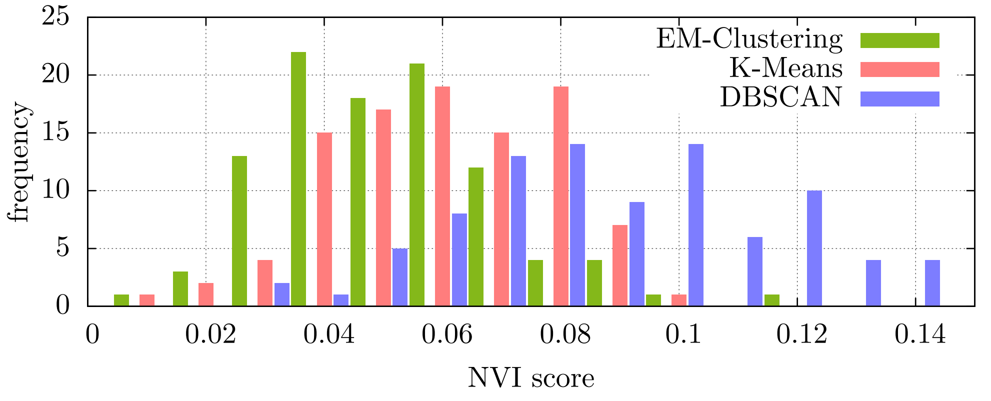


Figure 5.1: Histograms of Fowlkes-Mallows index (FMI; higher is better) and normalized variation of information (NVI; lower is better) comparing 100 simulated measurements containing partitioned peak locations with their clusters produced by the different methods.

the new centroids increase in general, decreasing the probability of consecutive mergings. At the same time, the cluster content can only increase, hence the variance of an eventual shifting of the centroid decreases. This is how the centroid positions are determined by definition. Here we can conclude that the expected number of swaps decreases with the increasing iteration of the algorithm. The more the data points are well-parted, the faster the algorithm works since more mergings are possible within the first iterations. The algorithm stops when no merging occurs anymore.

5.5 Evaluation

To evaluate peak clustering methods, we simulate peak locations according to locations in real MCC/IMS datasets, together with the true partition $\mathcal{P}$ of peaks. Most of the detected peaks appear in a small, dense area and rather early in the measurement. The remaining peaks are distributed widely, which is referred to as the sparse area (we let the areas overlap such that the dense area is contained within the sparse area). The areas have approximately the following boundaries (in units of ($\mathrm{V\,s\,cm^{-2}}$, s) from lower left to upper right point, cf. Figure 1.3:

$$\begin{aligned}\text{measurement: } & (0,0),(1.45,600)\\ \text{dense area: } & (0.5,4),(0.7,60)\\ \text{sparse area: } & (0.5,4),(1.2,450)\end{aligned}$$

Peak clusters are ellipsoidal and dense. From Bödeker et al. (2008) we know the minimum required distance between two peaks to allow identification as two separate compounds. We simulate 30 peak cluster centroids in the dense area and 20 in the sparse area, all randomly picked and uniformly distributed in the respective area. We then randomly pick both the number of peaks per cluster and the distribution of peaks within a cluster. Since we do not know the actual distribution model, we decided to simulate with three models: normal (n), exponential (e) and uniform (u) distribution with the following densities:

$$\begin{aligned}f_\mathrm{n}(r,t\,|\,\mu_\mathrm{t},\sigma_\mathrm{t},\mu_\mathrm{r},\sigma_\mathrm{r}) &= \mathcal{N}(t\,|\,\mu_\mathrm{t},\sigma_\mathrm{t})\cdot\mathcal{N}(r\,|\,\mu_\mathrm{r},\sigma_\mathrm{r}),\\ f_\mathrm{e}(r,t\,|\,\mu_\mathrm{t},\lambda_\mathrm{t},\mu_\mathrm{r},\lambda_\mathrm{r}) &= \lambda_\mathrm{t}\lambda_\mathrm{r}\exp\big(-(\lambda_\mathrm{t}|t-\mu_\mathrm{t}|+\lambda_r|r-\mu_\mathrm{r}|)\big)/4,\\ f_\mathrm{u}(r,t\,|\,\mu_\mathrm{t},\nu_\mathrm{t},\mu_\mathrm{r},\nu_\mathrm{r}) &= \begin{cases}(\pi\nu_\mathrm{t}\nu_\mathrm{r})^{-1} & \text{if } \frac{|t-\mu_\mathrm{t}|^2}{\nu_\mathrm{t}^2}+\frac{|r-\mu_\mathrm{r}|^2}{\nu_\mathrm{r}^2}\le 1\\ 0 & \text{otherwise}\end{cases},\end{aligned}$$

where $\mathcal{N}$ is the pdf of the Gaussian distribution. Here $(\mu_\mathrm{t},\mu_\mathrm{r})$ is the coordinate of the centroid with IRM in $\mathrm{V\,s\,cm^{-2}}$ and retention time in s. For the normal distribution, we used $\sigma_\mathrm{t}=0.002$ and $\sigma_\mathrm{r}=\mu_\mathrm{r}\cdot 0.002+0.2$. For the exponential distribution, we used $\lambda_\mathrm{t}=(1.45\cdot 2500)^{-1}$ (reduced mobility width for in single cell within M) and $\lambda_\mathrm{r}=1/(\mu_\mathrm{r}\cdot 0.002+0.2)$. For the uniform distribution, we used an ellipsoid with radii $\nu_\mathrm{t}=0.006$ and $\nu_\mathrm{r}=\mu_\mathrm{r}\cdot 0.02+1$. Figure 5.2 presents an exemplary distribution of simulated clusters.

We compared our adaptive EM clustering with two common clustering methods: k-means and DBSCAN. Since k-means needs a fixed number of clusters k and appropriate start values for the centroids, we used k-means++ by Arthur and Vassilvitskii (2007) for estimating good starting values and give it an advantage by assigning the true number of partitions. DBSCAN has the advantages that it does not need to know the number of clusters in advance and that it can find non-linear cluster boundaries, but it does not easily yield parametric cluster descriptors.

To measure the quality of an obtained clustering $\mathcal{C}$, we utilize the Fowlkes-Mallows index (FMI, Fowlkes and Mallows (1983)) and the normalized variation of information (NVI) score introduced by Reichart and Rappoport (2009).

The FMI requires consideration of all pairs of data points. If two data points belong to the same true partition of $\mathcal{P}$, they are called *connected*. Accord-

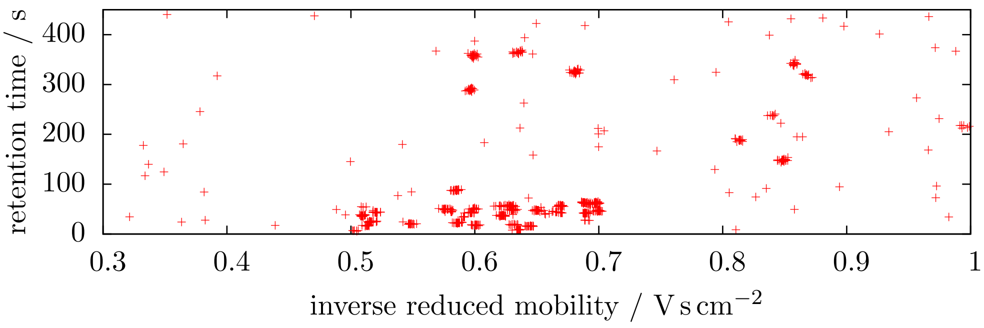
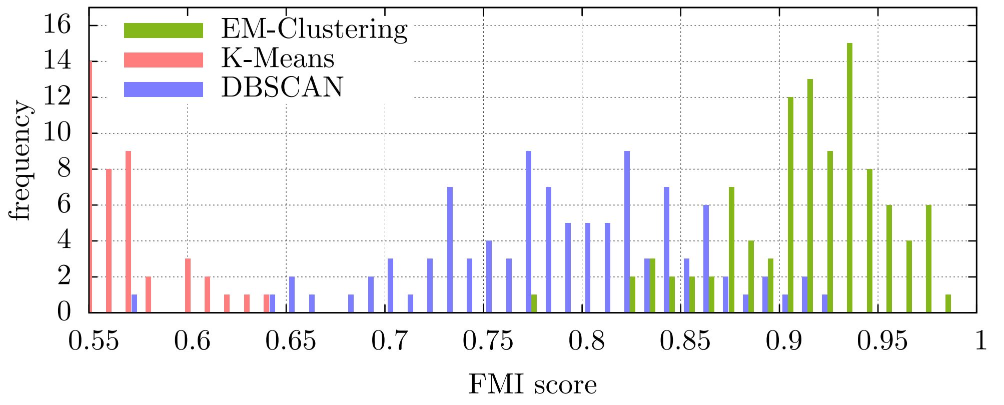


Figure 5.2: Exemplary distribution of simulated clusters randomly using one of the three proposed distributions for each cluster. Additional noise is added.

ingly, a pair of data points is called *clustered* if they are clustered together by the clustering method. Pairs of data points both connected and clustered are called true positives (TP). False positives (FP, not connected but clustered) and false negatives (FN, connected but not clustered) are computed analogously. The FMI is the geometric mean of precision and recall: $\mathrm{FMI}(\mathcal{P},\mathcal{C}) := \sqrt{TP/(TP+FP)\cdot TP/(TP+FN)}$,
where $\mathcal{P}$ is the true partition and $\mathcal{C}$ is the clustering. We have $\mathrm{FMI}(\mathcal{P},\mathcal{C}) \in [0,1]$, and $\mathrm{FMI}(\mathcal{P},\mathcal{C}) = 1$ indicating perfect agreement. The FMI is difficult to interpret when the number of clusters in $\mathcal{C}$ and $\mathcal{P}$ differs significantly.

Therefore we use a second measure which considers cluster sizes only, the normalized variation of information (NVI). To compute the NVI, an auxiliary $(|\mathcal{P}|\times|\mathcal{C}|)$-dimensional matrix $A = (a_{i,j})$ is computed, where $a_{i,j}$ is the number of data points within partition i assigned to cluster j. The NVI score is defined via entropies; let n be the number of data points and

$$
\begin{aligned}
H(\mathcal{P}) &:= -\sum_{i\leq|\mathcal{P}|} \frac{\sum_{j\leq|\mathcal{C}|} a_{i,j}}{n} \log \frac{\sum_{j\leq|\mathcal{C}|} a_{i,j}}{n},\\
H(\mathcal{C}) &:= -\sum_{j\leq|\mathcal{C}|} \frac{\sum_{i\leq|\mathcal{P}|} a_{i,j}}{n} \log \frac{\sum_{i\leq|\mathcal{P}|} a_{i,j}}{n},\\
H(\mathcal{P}|\mathcal{C}) &:= -\sum_{j\leq|\mathcal{C}|}\sum_{i\leq|\mathcal{P}|} \frac{a_{i,j}}{n} \log \frac{a_{i,j}}{\sum_{i'\leq|\mathcal{P}|} a_{i',j}},\\
H(\mathcal{C}|\mathcal{P}) &:= -\sum_{j\leq|\mathcal{C}|}\sum_{i\leq|\mathcal{P}|} \frac{a_{i,j}}{n} \log \frac{a_{i,j}}{\sum_{j'\leq|\mathcal{C}|} a_{i,j'}},\\
\mathit{NVI}(\mathcal{P},\mathcal{C}) &:= \begin{cases} \frac{H(\mathcal{P}|\mathcal{C})+H(\mathcal{C}|\mathcal{P})}{H(\mathcal{P})} & \text{if } H(\mathcal{P}) \neq 0,\\ H(\mathcal{C}) & \text{otherwise.}\end{cases}
\end{aligned}
$$



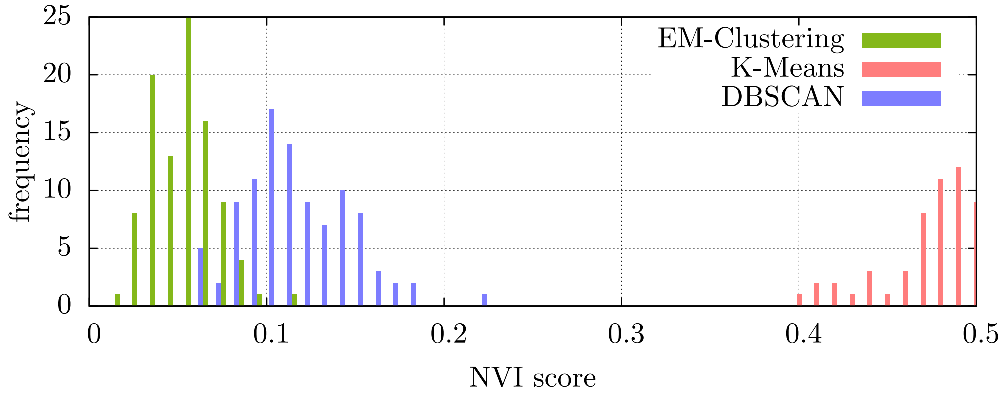


Figure 5.3: Histograms similar to Figure 5.1, but in a more realistic noisy scenario (see text). An FMI of 1 and NVI of 0 would be optimal.

Here $NVI(\mathcal{P}, \mathcal{C}) = 0$ indicates perfect agreement between the cluster size distributions. Together, an FMI of 1 and an NVI score of 0 indicate a perfect clustering.

For the first test, we evaluated 100 sets of data points distributed as described above. The cluster model (normal, exponential or uniform) was drawn randomly. The results show that even with the advantage of k-means having been given the true k, our adaptive EM clustering performs best on average in terms of FMI and NVI score (Figure 5.1). For the second test, we additionally inserted 200 uniformly distributed (noise) peaks into the measurement area. All these peaks are singletons and have no matching peaks. The results (Figure 5.3) show that adaptive EM clustering still performs best on average, whereas k-means fails.

6 Comparison with a Reference

6.1 Background

Having a set of (consensus) peak lists, it is desirable to identify the actual analytes within these lists. Thus, when a reference (i.e. a reference peak list) is available, a comparison has to be performed. However, some issues must be taken into consideration which argue against a simple search of the minimal Euclidean distance of a peak position against the reference positions.

On the one hand, both the dimensions of retention time and drift time provide different magnitudes of their values, so a normalization is essential. On the other hand, the hardware parts like MCC can scuff and subsequently generate biased data. The IM spectrometer device could also not be adjusted perfectly to the defined voltage, leading to a shift in drift time. These influences must be taken into account for a comparison. It results in a challenge to align two peak lists (produced and reference set) having a constant shift in one dimension and a linear shift in the other dimension. The objective is to find parameters of the constant and linear shift that minimize the distance between both peak lists.

6.2 Related Work

The problem of aligning a set of points to another is referred to as point set registration. A coherent point drift algorithm was introduced by Myronenko and Song (2010). They handle two methods with different constraints. The rigid point set registration only allows a transform from one point set to another by shifting, scaling and rotating the point set. A nonrigid point set registration typically allows nonlinear transform, reflection, rotation in all dimensions, shear mapping and other affine transformations. In both cases one point set is referred to as the source point set $x^{n \times \mathcal{D}}$ and the second as the target point set $y^{m \times \mathcal{D}}$, respectively. A Gaussian mixture model is used in which every point of the source point set is the centroid of a $\mathcal{D}$-dimensional Gaussian model. Having a rotation matrix $\mathbf{R}^{\mathcal{D} \times \mathcal{D}}$, a shift-vector $\mathbf{t}^{\mathcal{D} \times 1}$ and a scaling parameter s, let $\mathcal{G}$ be the transformation function and Q the objective that defined as follows:

$$\begin{aligned}
\mathcal{G}(y_j \,|\, \mathbf{R}, \mathbf{t}, s) &:= s\mathbf{R}y_j + \mathbf{t}, \\
P^{\text{old}}(j \,|\, x_i) &:= \frac{\exp^{-\frac{1}{2}\left\|\frac{x_i - \mathcal{G}(y_j \,|\, \mathbf{R},\mathbf{t},s)}{\sigma}\right\|^2}}{\sum_{k\le m} \exp^{-\frac{1}{2}\left\|\frac{x_i - \mathcal{G}(y_k \,|\, \mathbf{R},\mathbf{t},s)}{\sigma}\right\|^2} + c}, \\
N_{\text{P}} &:= \sum_{i\le n, j\le m} P^{\text{old}}(j \,|\, x_i), \\
Q &:= \frac{1}{2\sigma} \sum_{i\le n, j\le m} P^{\text{old}}(j \,|\, x_i) ||x_i - \mathcal{G}(y_j \,|\, \mathbf{R}, \mathbf{t}, s)|| + \frac{N_{\text{P}}\mathcal{D}}{2} \log \sigma^2,
\end{aligned}$$

where σ^2 is the variance. Here, P^{old} denotes the posterior probabilities of the Gaussian mixture model components where $c = (2\pi\sigma^2)^{\mathcal{D}/2}\omega/(1-\omega)m/n$, a uniform background distribution to explain outliers with the background model weight ω. The objective function Q is minimized by utilizing the maximum likelihood estimators for $\sigma, \mathbf{R}, \mathbf{t}$ and s. Since this method assumes the same variance over all dimensions and utilizes the fast Gauss transform, it is not applicable for our purposes but it can be enhanced.

6.3 Algorithm

In the following, an approach based on the work of Myronenko and Song (2010) is described which again utilizes the EM algorithm for parameter estimation.

6.3.1 Mixture Model

We are given a reference peak list $\mathcal{R} = (\mathcal{R}_i)$ with $\mathcal{R}_i = \{\mu_{\mathcal{R},\text{T}}, \mu_{\mathcal{R},\text{R}}\}, i \le n$ and a target peak list $\mathcal{T} = (\mathcal{T}_j)$ with $\mathcal{T}_j = \{\mu_{\mathcal{T},\text{T}}, \mu_{\mathcal{T},\text{R}}\}, j \le m$ where $\mu_{\text{T}}, \mu_{\text{R}}$ are the centroid positions of the consensus peaks. We intend to estimate the shift parameter s_{T} in IRM and scaling parameter s_{R} in retention time. Again, a Gaussian mixture model is utilized in which the centroids of $\mathcal{R}$ indicate the model means and the variances can be computed using the peak definitions

$$\begin{aligned}
\sigma^*_{\text{T}} &:= \Delta\tau, \\
\sigma^*_{\text{R}} &:= \xi(\mu^*_{\mathbf{d},\text{R}})/\phi \quad \text{for all } \mathbf{d} \in \{\mathcal{R}, \mathcal{T}\}.
\end{aligned}$$

When the target peak list is a single peak list either provided by the offline or the online peak model estimation method from Section 3 or 4, the variances do not have to be computed since they were estimated and are more precise. Let

$$p(\mathcal{R}_i, \mathcal{T}_j) := \frac{\exp\left(-0.5\left(\left(\frac{\mu_{\mathcal{R},\mathrm{R},i}-\mu_{\mathcal{T},\mathrm{R},j}\cdot s_{\mathrm{R}}}{\sigma_{\mathcal{R},\mathrm{T},i}}\right)^2 + \left(\frac{\mu_{\mathcal{R},\mathrm{T},i}-(\mu_{\mathcal{T},\mathrm{T},j}+s_{\mathrm{T}})}{\sigma_{\mathcal{R},\mathrm{T},i}}\right)^2\right)\right)}{2\pi\sigma_{\mathcal{R},\mathrm{T},i}\sigma_{\mathcal{R},\mathrm{R},i}}$$

be the Gaussian probability distribution for the described consensus peak $\mathcal{T}_j$ with model $\mathcal{R}_i$.

6.3.2 Initial parameters

At the beginning we assume the best case, i.e. that we have no shift and no scale and thus set the initial parameters $s_{\mathrm{R}} = 1$ and $s_{\mathrm{T}} = 0$.

6.3.3 Estimating the parameters

Again, we take the maximum likelihood estimators in the maximization step, which are

$$\begin{aligned} s^*_{\mathrm{T}} &:= \frac{1}{n}\sum_{i\le n, j\le m} W_{i,j}(\mu_{\mathcal{R},\mathrm{T},i} - \mu_{\mathcal{T},\mathrm{T},j}), \\ s^*_{\mathrm{R}} &:= \frac{1}{n}\sum_{i\le n, j\le m} W_{i,j}(\mu_{\mathcal{R},\mathrm{R},i}/\mu_{\mathcal{T},\mathrm{R},j}). \end{aligned}$$

6.3.4 Final step

To determine which target peak belongs to which reference peak, we consider the membership variables indicating that information. Let

$$j^* := \arg\max_{j\le m}(W_{i,j})$$

be the target peak describing model i best for all $i \le n$ and thus being connected. Here, we also claim a maximal distance to ensure that unreasonable connections are not established. For this purpose, we utilize the standard deviation in both dimensions to compute the minimal probability for a model. Let $\mathbf{V}_{s^*_{\mathrm{T}},s^*_{\mathrm{R}}}(\{x,y\}) := \{x + s^*_{\mathrm{T}}, y \cdot s^*_{\mathrm{R}}\}$ be the transformation function to compute the transformed coordinates. Connections between $\mathcal{R}_i$ and $\mathcal{T}_j$ are established when

$$\log\left(\frac{p(\mathcal{R}_i, \mathbf{V}_{s^*_{\mathrm{T}},s^*_{\mathrm{R}}}(\mathcal{T}_j))}{p(\mathcal{R}_i, \{\mu_{\mathcal{R},\mathrm{T}} + \sigma^*_{\mathrm{T}}, \mu_{\mathcal{R},\mathrm{R}} + \sigma^*_{\mathrm{R}}\}_i)}\right) \ge 0$$

is satisfied or discarded otherwise.

6.4 Evaluation

The setting of the evaluation for the comparison approach is structured as follows: as the reference peak list, we use a consensus peak list provided by the Pseudomonas aeruginosa (described in Section A.2) utilizing the offline peak extraction described in Section 3 and clustered with EM clustering as described in Section 5. The dataset contains 67 measurement files, the consensus peak list has 27 peaks listed and mean/standard deviation of extracted peaks with the single peak lists is 13.47 ± 2.19. For one experiment, the shift parameter s_{T} in IRM and the scaling parameter s_{R} in retention time is determined randomly, a comparison peak list is created with peaks transformed according to both parameters and accordingly, these parameters again are inferred by considering only the reference and the comparison peak list. For a comparison peak list, i) a subset of the reference peak list is taken in which the size is a random Gaussian number with $\mu = 13.47, \sigma = 2.19$, ii) a set of false peaks is added with a random size between $[5, 10]$ with both IRM and retention time randomly chosen with τ between $[0.5, 1.2]\,\mathrm{V\,s\,cm^{-2}}$ and ρ between $[0, 350]\,\mathrm{ms}$, iii) all peak positions are transformed according to the shifting and scaling parameter and iv) all peak positions are jittered using random Gaussian numbers with $\mu = t, \sigma = \Delta\tau$ in IRM and $\mu = \rho, \sigma = \xi(\rho)/\phi$ in retention time. The device parameters are taken as described in Table 8.1. To see the influence of both shift and scaling parameter, we perform nine independent experiment batches with alternating parameters. Each experiment batch consists of 1000 independent experiments. The parameters are randomly chosen within ranges of a Cartesian product of $([-0.01, 0.01], [-0.025, 0.025], [-0.1, 0.1]) \times ([1, 1.1], [1, 1.25], [1, 2])$ for shift parameter and scaling parameter, respectively.

Figure 6.1 presents all nine experiment batches. All figures show a histogram of the difference between the determined and estimated parameter, the figures on the left the difference for the shift parameter in IRM and the figures on the right for the scaling parameters, respectively. The upper figures show the results of the scaling parameter randomly set between $[1, 1.1]$, the middle figures with $[1, 1.25]$ and the bottom figures with $[1, 2]$. The combination of 0.01 maximum shift and 1.1 maximum scale is the most realistic scenario. In the top figures it is visible that the estimation performs best in this scenario. On the harder scenarios with 0.025 maximum shift and 1.2 maximum scale, the method still copes well. Considering unrealistic scenarios with 0.1 shift corresponding to a shift of over 3 ms and a scale up to 2, the method fails. Although the correct parameters are estimated on average, the variance is very high.

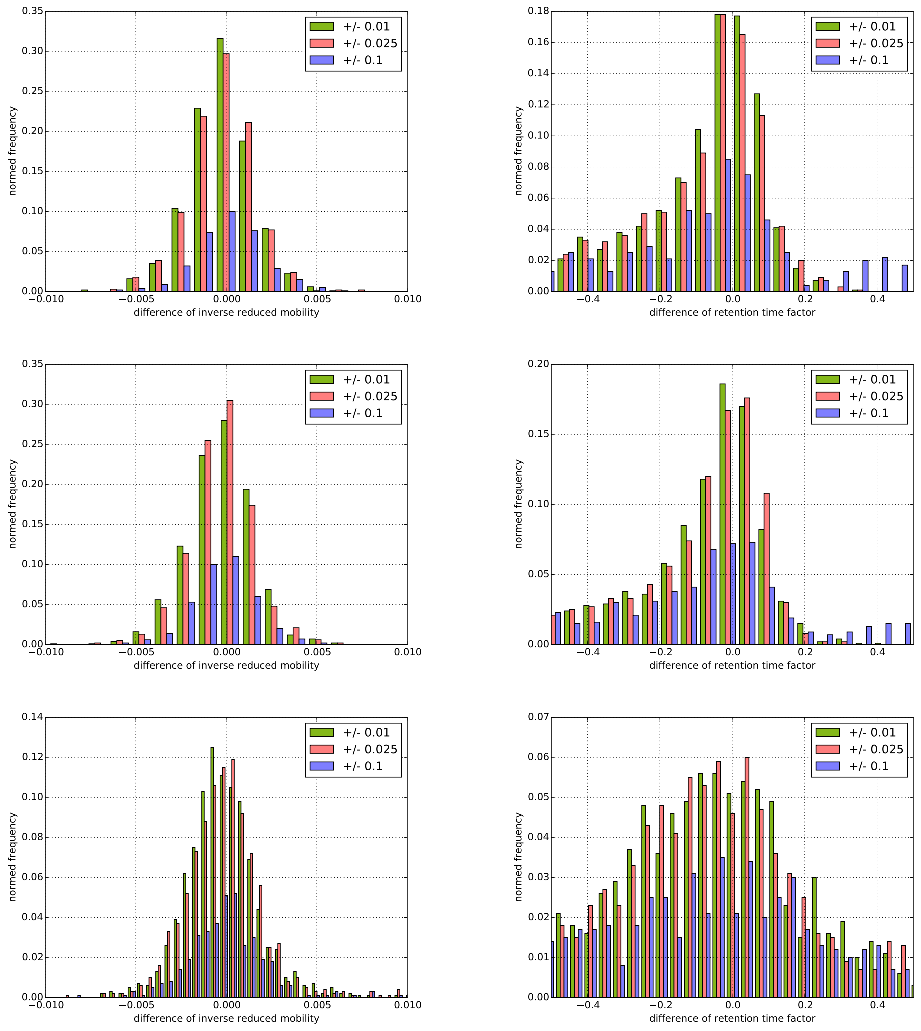


Figure 6.1: Histograms of the differences between determined and estimated shift (left) and scale (right) parameters of 1000 experiments each; (top) maximum scale: 1.1; (middle) maximum scale: 1.2; (bottom) maximum scale: 2. Each figure contains three histograms for all three maximum shifts each.

7 Evaluation of Determination Limit of Peak Extraction

The first global evaluation takes both online and offline methods into account and focuses on the determination limit of the peak extraction. We intend to determine the noise margin level `noise_margin` which amplifies the background noise deviation σ_{N}. In general we claim that peaks with minimal peak height exceeding `noise_margin` $\cdot \sigma_{\mathrm{N}}$ will be detected correctly with reliable results.

7.1 Setting

We are using the calibration curve data sets as described in Appendix A.1. Since these series have only one analyte injected and the concentration of this analyte decreases exponentially (thinning series), it appears reasonable to check at which signal intensity level the methods fail to find peaks. We use the following parameters for both methods as listed in Table 7.1. Afterwards the single peak lists are clustered using the EM clustering described in Section 5. The data sets have the following average standard deviations for the background noise model: $\sigma_{\mathrm{N},1} = 4.26 \times 10^{-3}$ V and $\sigma_{\mathrm{N},2} = 4.35 \times 10^{-3}$ V.

To evaluate the results, we compare the results with the determination limit introduced by the German Institute for Standardization. The DIN 32645 standard defines a limit of detection and limit of determination for chemical analyses. A limit of detection determines the lowest quantity of analytes that can be distinguished from blank samples. This limit is defined as the mean of the blank with a standard deviation scaled by a confidence factor. Typically, for a mean with zero value, the limit of detection is defined as $3 \cdot \sigma_{\mathrm{N}}$. The limit of determination is the lowest quantity of analytes that can quantified with a defined precision. In chemical analyses, the limit is typically defined as $9 \cdot \sigma_{\mathrm{N}}$.

Table 7.1: Parameters of both offline and online methods for the determination limit evaluation.

Parameter	*Value*	*Parameter*	*Value*
offline			
`r_width_offset`	2.5	`candidate_detection`	cf
`r_width_factor`	0.01	`picking`	emc
`t_width_offset`	0.003	`modeling`	pme
`preprocessing_first`	bc	`fftcutoff`	0.2
`preprocessing_second`	dn	`convergence_threshold`	0.001
`preprocessing_third`	s		
online			
`r_width_offset`	2.5	`convergence_threshold`	0.01
`r_width_factor`	0.01	`min_correlation`	0.9
`t_width_offset`	0.003		

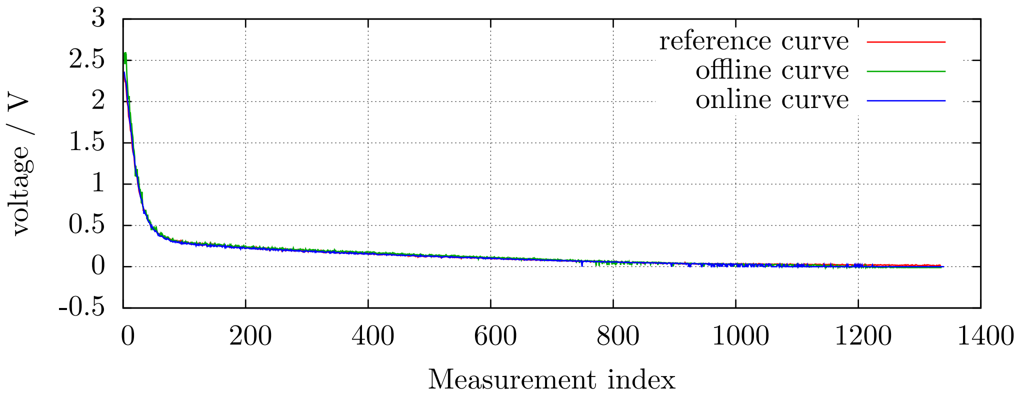
7.2 Results

Experiment 1

Figure 7.1 (top) illustrates the results of the complete calibration curve for data set 1. As a reference (red line), we first manually determine the peaks position (τ^*, ρ^*) over all measurements and search the highest signal intensity within the rectangle window from $(\tau^* - \Delta\tau, \rho^* - \Delta\rho)$ to $(\tau^* + \Delta\tau, \rho^* + \Delta\rho)$. The limit of determination is $\sigma_{\mathrm{N},1} \cdot 9 = 0.0383$ (brown dashed line). For the first 700 measurements, both methods keep well with both the identification and extraction as well as the estimation of the peak model to get the peak height. To get a better comparison, Figure 7.1 (bottom) presents a zoomed view at the end of the curve. Additionally, the determination limit is plotted. For both methods it is clearly visible that the determination limit is kept, thus `noise_margin` = 9 suffices. The offline method performs even slightly better for the recognition at low quantities whereas it estimates the model parameters slightly worse.

Experiment 2

Figure 7.2 shows the results for the second calibration curve, respectively. The reference line and the limit of determination are computed as in Experiment 1. Some points of the reference line are corrupted since a few IM spectra are faulty within the measurements. When these IM spectra are within the range of the corresponding peak to the analyte, outlier values are stored. Hence, the online method fails to detect a peak when its height exceeds of half the maximum deflection due to a deformation of the actual peak. The



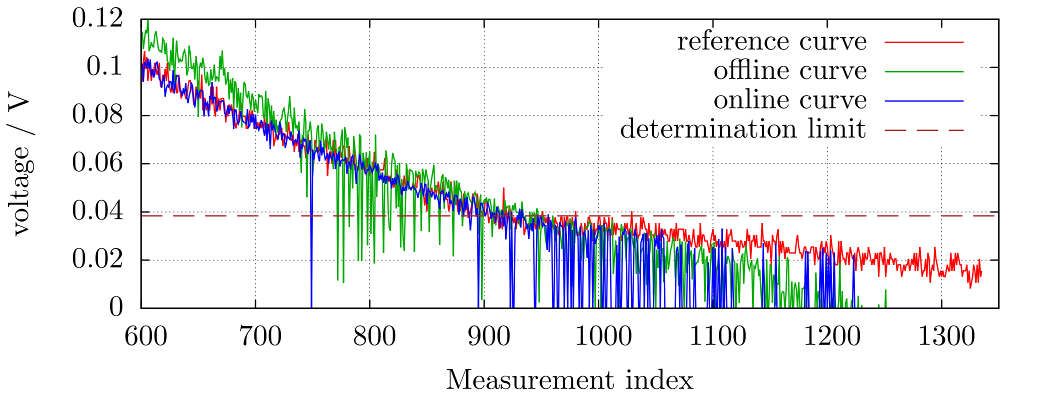
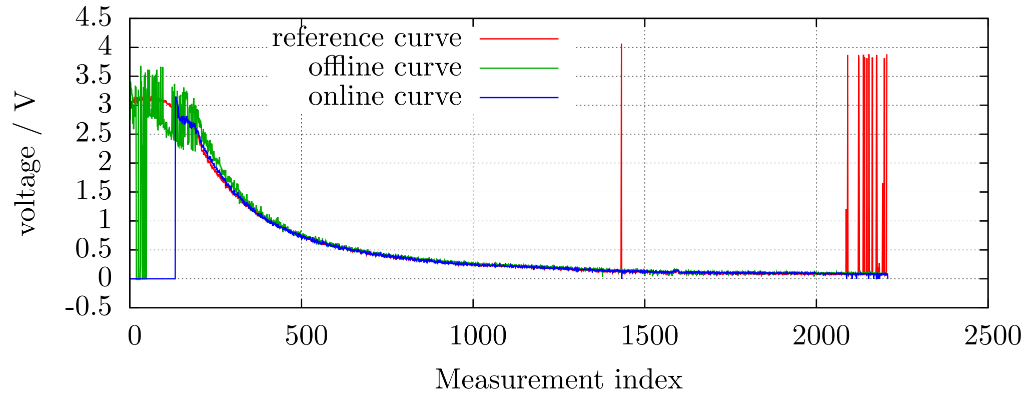


Figure 7.1: Top: complete calibration curve for data set 1: (red line) the calibration analyte; (green line) extracted peaks using PME; (blue line) extracted peak heights using OPME. Bottom: zoom in of the last part of the calibration curve for data set 1 (top): (red line) the calibration analyte; (green line) extracted peaks using PME; (blue line) extracted peak heights using OPME. The online method copes with the standard determination limit (dashed brown line).

offline method recognizes peaks with quantities of such height but also fails to estimate good model parameters. For quantities below half of the maximum deflection, both methods perform well. The methods estimate slightly higher peak intensities due to their estimation process which is not critical, since this is a linear error, it can be suppressed. Generally, the second experiment confirms the result considering the determination limit of the first experiment.



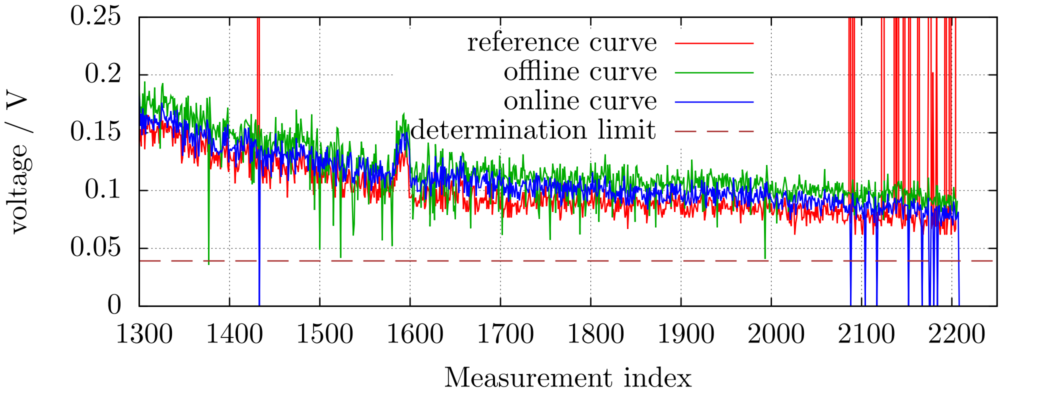


Figure 7.2: Top: complete calibration curve for data set 2: (red line) the calibration analyte; (green line) extracted peaks using PME; (blue line) extracted peak heights using OPME. Bottom: zoom in of the last part of the calibration curve for data set 2 (top): (red line) the calibration analyte; (green line) extracted peaks using PME; (blue line) extracted peak heights using OPME. The online method copes with the standard determination limit (dashed brown line).

8 Evaluation of Automated Peak Extraction compared with Manual Annotation

The next evaluation considers a comparison of manually annotated datasets in which the peaks were pinpointed by a domain expert using a visualization tool as VisualNow (written by Bödeker et al. (2008)) and the automated peak extraction using both the online and offline method.

8.1 Setting

For this experiment we are using three datasets, namely *Dataset 69*, *Pseudomonas aeruginosa* and *Dataset 508* as described in Appendix A.2. A manual data table (measurement $\times$ consensus peak) is provided for all datasets. The peak extraction methods were performed with the parameter adjustment listed in Table 8.1. The noise margin was set as determined in Section 7, namely `noise_margin` $= 9$.

Here, we treat the manually detected data as the basic truth. We count for each single measurement the peaks detected from both manual and automated method (true positives TP), as well as false positives FP, false negatives FN and true negatives TN, respectively. For this particular experiment it is possible to count TN, since both manually and automatically generated data tables provide signal intensities for every measurement / consensus peak combination. If a peak is not found in a particular measurement, the corresponding signal intensity within the data table is zero (or definitely below the determination limit). Thus, when a signal intensity of a certain measurement / peak combination satisfies the "not found" criterion within both manually and automatically generated data table, a true negative is counted. To connect two consensus peak lists provided by the data tables, we use a comparison similar to the method as described in Section 6.

Table 8.1: Parameters of both offline and online methods for the comparison evaluation.

Parameter	*Value*	*Parameter*	*Value*
offline			
`r_width_offset`	2.5	`candidate_detection`	cf
`r_width_factor`	0.06	`picking`	emc
`t_width_offset`	0.003	`modeling`	pme
`preprocessing_first`	bc	`fftcutoff`	0.2
`preprocessing_second`	dn	`convergence_threshold`	0.001
`preprocessing_third`	s	`noise_margin`	9
online			
`r_width_offset`	2.5	`convergence_threshold`	0.01
`r_width_factor`	0.06	`min_correlation`	0.9
`t_width_offset`	0.003	`noise_margin`	9

Additionally, we claim only a 1 to 1 connection or no connection between the peak lists. Note that in contrast to the evaluation in Section 3.5.2, we do not compare peak lists of single measurements. After the counting, we compute the specificity spec = TN/(TN + FP) and precision (or positive predictive value) spec = TP/(TP+FP). One feature of dataset Pseudomonas aeruginosa and Dataset 508 is the difference between number of consensus peaks within manually generated peak lists and both automatically generated peak lists. They are about 5 and 10 times higher since the data table was not filtered with respect to singleton clusters or clusters with too few peaks. Thus, we do not use the sensitivity and accuracy as quality measures since it would bias the results. The numbers for (manual, offline, online) consensus peak lists for the data sets are follows: Dataset 69 (60, 57, 56); Pseudomonas aeruginosa (224, 26, 19); Dataset 508 (156, 36, 37).

8.2 Results

The results for all the six experiments are plotted in Figure 8.1. Both methods achieve a specificity above 0.75 for almost all measurement comparisons. Surely there is space for improvement of the methods to increase the precision, but generally there is a good agreement between the manual and the automated extraction.

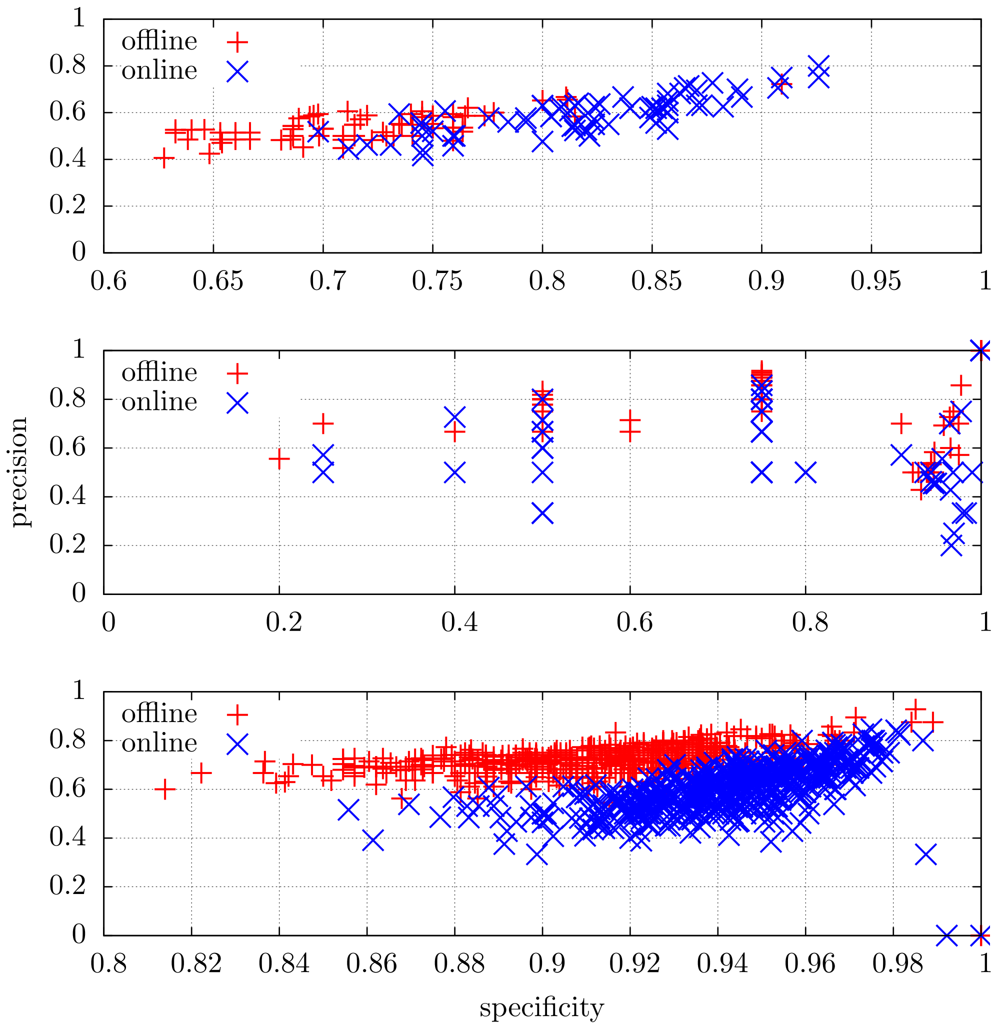


Figure 8.1: The specificity and precision for all tree experiments are plotted against each other for both offline (red points each) and online method (blue points each). Experiments: (top) Dataset 69; (middle) Pseudomonas aeruginosa; (bottom) Dataset 508.

9 Evaluation of Parameters

The adjustable parameters are significantly relevant for the quality of the peak extraction. In this section we try to determine the best combination of parameter values for both introduced methods, PME and OPME.

9.1 Setting

Since the parameters strongly depend on the adjustment of the spectrometer, we concentrate here on standard adjustments as listed in Table 1.1. Four of the adjustable parameters are used for both methods PME as well as OPME, namely `convergence_threshold`, `r_width_factor`, `r_width_offset` and `t_width_offset`. The parameter `fftcutoff` is solely used by PME and `min_correlation` by OPME, respectively. Since `noise_margin` was evaluated in Section 7, it is not taken into consideration here. Table 8.1 lists standard values for all parameters. For all mentioned parameters we take a set of values surrounding its standard value. Within every experiment only one parameter was increased, whereas the standard values were assigned to the remaining parameters. We use the Reference 6 mixture dataset described in Appendix A.1, since we know the exact position of the supposed peaks, we can check whether peaks are found and extracted or not. Additionally, we count the presence of all known peaks within every measurement to have reference count. As a quality measure, we use a normalized agreement $\mathbf{R} \in [-1; 1]$. A value $\mathbf{R} = 1$ corresponds to a perfect agreement (all true positives and no false positives found). Let $\mathbf{TP}_i$ be the number of true positives for measurement i and $\mathbf{FP}_i$ the number of false positives, respectively, then let

$$\mathbf{R} := \frac{1}{n} \cdot \sum_{i \leq n} \left(\frac{\mathbf{TP}_i}{\max(\mathbf{TP})} - \frac{\mathbf{FP}_i}{\max(\mathbf{FP})} \right)$$

where n is the number of measurements and $\max(\mathbf{FP})$ the highest false positive rate among all measurements.

9.2 Results

The Figures 9.1, 9.2, 9.3, 9.4, 9.5 and 9.6 present the results for all parameter experiments, each figure for one parameter. The following applies for

Table 9.1: Agreement **R** results for all parameter values of all PME and OPME parameters.

PME results									
`fftcutoff`		`convergence_threshold`		`r_width_factor`		`r_width_factor`		`t_width_offset`	
value	**R**	value	**R**	value	**R**	value	**R**	value	**R**
0.1	0.223	0.0001	0.339	0.01	0.241	0.0	0.165	0.001	-0.031
0.2	0.424	0.0005	0.351	0.02	0.257	0.5	0.22	0.002	0.439
0.3	0.429	0.001	0.355	0.03	0.231	1.0	0.22	0.003	0.248
0.4	0.434	0.005	0.397	0.04	0.261	1.5	0.23	0.004	0.271
0.5	0.415	0.01	0.407	0.05	0.252	2.0	0.241	0.005	0.266
0.6	0.401	0.05	0.373	0.06	0.3	2.5	0.246	0.006	0.306
		0.1	0.283	0.07	0.291	3.0	0.248	0.007	0.322
		0.5	0.177	0.08	0.334	3.5	0.265	0.008	0.346
				0.09	0.309	4.0	0.269	0.009	0.339
				0.1	0.321	4.5	0.278	0.01	0.366
				0.11	0.315	5.0	0.282		
				0.12	0.343				
OPME results									
`min_correlation`		`convergence_threshold`		`r_width_factor`		`r_width_factor`		`t_width_offset`	
0.7	0.215	0.0001	0.194	0.01	0.202	0.0	0.281	0.001	-0.006
0.75	0.229	0.0005	0.204	0.02	0.192	0.5	0.045	0.002	0.11
0.8	0.247	0.001	0.206	0.03	0.207	1.0	0.145	0.003	0.206
0.85	0.28	0.005	0.21	0.04	0.193	1.5	0.271	0.004	0.225
0.9	0.339	0.01	0.225	0.05	0.196	2.0	0.334	0.005	0.225
0.95	0.397	0.05	0.233	0.06	0.225	2.5	0.339	0.006	0.25
0.99	0.158	0.1	0.238	0.07	0.199	3.0	0.349	0.007	0.251
		0.5	0.149	0.08	0.168	3.5	0.341	0.008	0.252
				0.09	0.155	4.0	0.299	0.009	0.257
				0.1	0.153	4.5	0.285	0.01	0.259
				0.11	0.156	5.0	0.241		
				0.12	0.139				

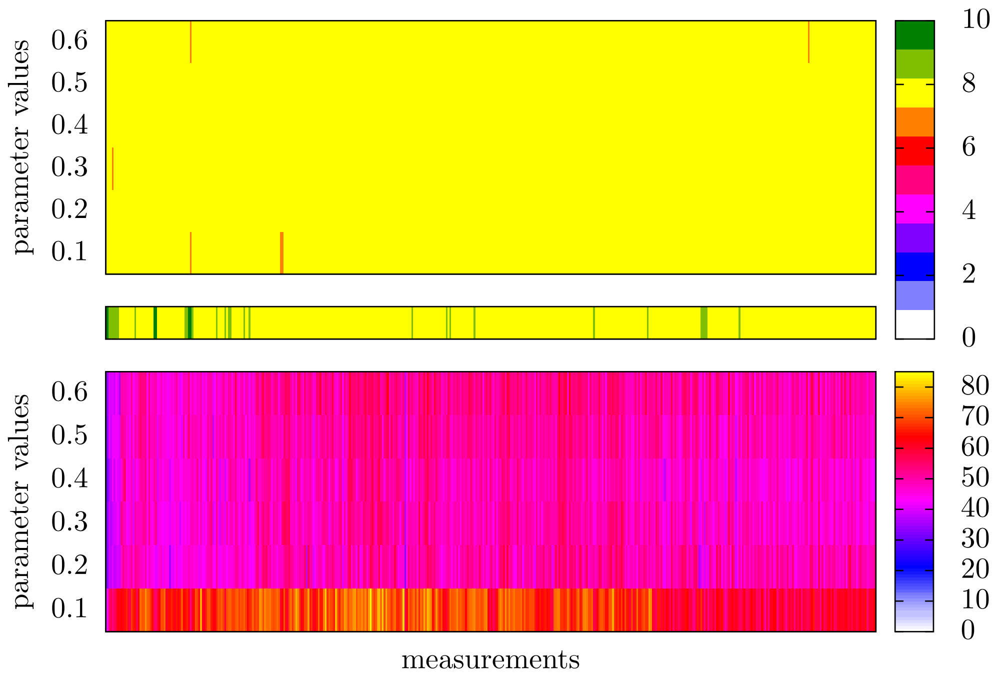


Figure 9.1: Experiment for `fftcutoff` for PME: (top) color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, the single row indicating the actual number of present peaks within the measurements; (bottom) color-coded matrix containing the **FP** rate. The more a row resembles to the counting row for the top matrix and the lower the numbers of the row in the bottom matrix, the better the parameter value.

every figure: every figure unit consists of two parts; the top part is a color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, an additional single row shows the number of truly present peaks for every measurement; the bottom part presents a color-coded matrix containing the **FP**. Since `fftcutoff` and `min_correlation` are only used by either PME or OPME, the corresponding Figures 9.1 and 9.2 contain only one figure unit. The remaining figures contain two figure units since both peak extraction methods utilize these parameters. Furthermore, Table 9.1 presents the **R** results for all parameter values of all PME and OPME parameters. All common parameters are listed within the same column.

First of all, it is noticeable that most of the measurements contain eight peaks and only few contain nine or even ten peaks. The reason why the methods do detect any measurement with more than eight peaks is that some measurements contain dimers and thus deformed monomers. In those cases the dimer was detected but due to its deformation the monomers were not found. We recall that the methods were not designed to recognize deformed monomers or align monomers and dimers. In many measurements, one particular peak

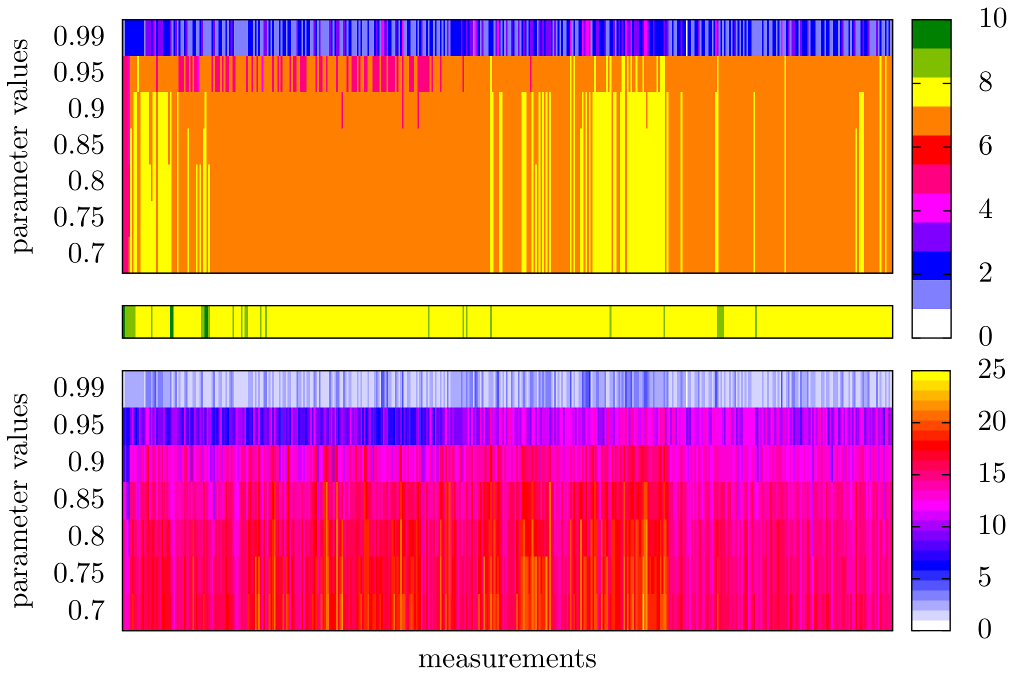


Figure 9.2: Experiment for `min_correlation`: for OPME: (top) color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, the single row indicating the actual number of present peaks within the measurements; (bottom) color-coded matrix containing the **FP** rate.

often occurs with a maximal signal intensity slightly exceeding the determination limit, resulting in OPME usually only finding seven peaks. We notice that OPME displays an overall lower **FP** rate. In general, PME has a high false positive rate, which can be explained by (1) the measurements may contain minimal concentrations of not characterized compounds or (2) compounds with high concentrations produce trails resulting in more peaks than only the noticed monomers and dimers.

Parameter `fftcutoff`

The first parameter is only used by PME for preprocessing. A visualization of the results for the `fftcutoff` experiment is presented in Figure 9.1. Although it seems that the value has no effect on the results, we recommend to preserve the first 20% of the frequency spectrum, thus `fftcutoff` $= 0.2$. The **R** results in Table 9.1 confirm our choice. For higher values, the effect of noise reduction is significantly lowered and for lower values, too many artifacts are provided and being misleadingly detected as peaks.

Parameter `min_correlation`

The only parameter solely used by the online method is `min_correlation`. The results for this parameter are illustrated in Figure 9.2. Since it is desirable that this parameter is being set as high as possible to get reliable results and few false positive extraction, our initial choice of `min_correlation` $= 0.9$ is reasonable. Considering Table 9.1, even 0.95 should be a good choice. Actually, we also tested the value 0.999 but at that high confidence value no peak was found.

Parameter `convergence_threshold`

Figure 9.3 presents the results for the performance parameter `convergence_threshold`. At first sight, this parameter has only negligible influence on the quality of extraction. This can be explained by the fact that this parameter is used for every estimation of the model parameters. When appropriate initial parameters are provided, the methods only slightly maximize the parameters. Since the number of iterations for the numerical optimization methods increase the computation time and thus significantly lower the performance, a relative error of 1% suffices, hence `convergence_threshold` $= 0.01$. Agreement $\mathbf{R}$ affirms this suggestion.

Parameter `r_width_factor`

The results for the second common parameter `r_width_factor` are visualized in Figure 9.4. It is visible that PME is less prone with respect to this parameter. For the online method, the value `r_width_factor` ≤ 0.07 seems legit and therefore confirms our pre-choice. Our assumed adjustment coincides with this result from Table 9.1, especially when considering the results for OMPE.

Parameter `r_width_offset`

The results of the `r_width_offset` experiment are illustrated in Figure 9.5. Again the results for PME are more stable when the parameter is alternated, in contrast to OPME. This can be explained by the rather simple peak candidate detection that does not take the peak model under consideration at the beginning. Thus, exceeding 0.5, the results show an almost perfect peak extraction. For both methods we recommend a value $2.0 \leq$ `r_width_factor` ≤ 3.5 which can also be reinforced by the $\mathbf{R}$ results, especially when considering OPME.

Parameter `t_width_offset`

Figure 9.6 presents the results for parameter `t_width_offset`. Again, PME is more tolerant towards a higher value range for this parameter due to its simplicity. For both counting matrices it is clearly visible that $0.003 \leq$ `t_width_offset` ≤ 0.004 becomes reasonable. This result supports our calculation for the offset in IRM dimension. Higher values make physically no sense, although the $\mathbf{R}$ values are increasing for increasing the parameter value for PME.

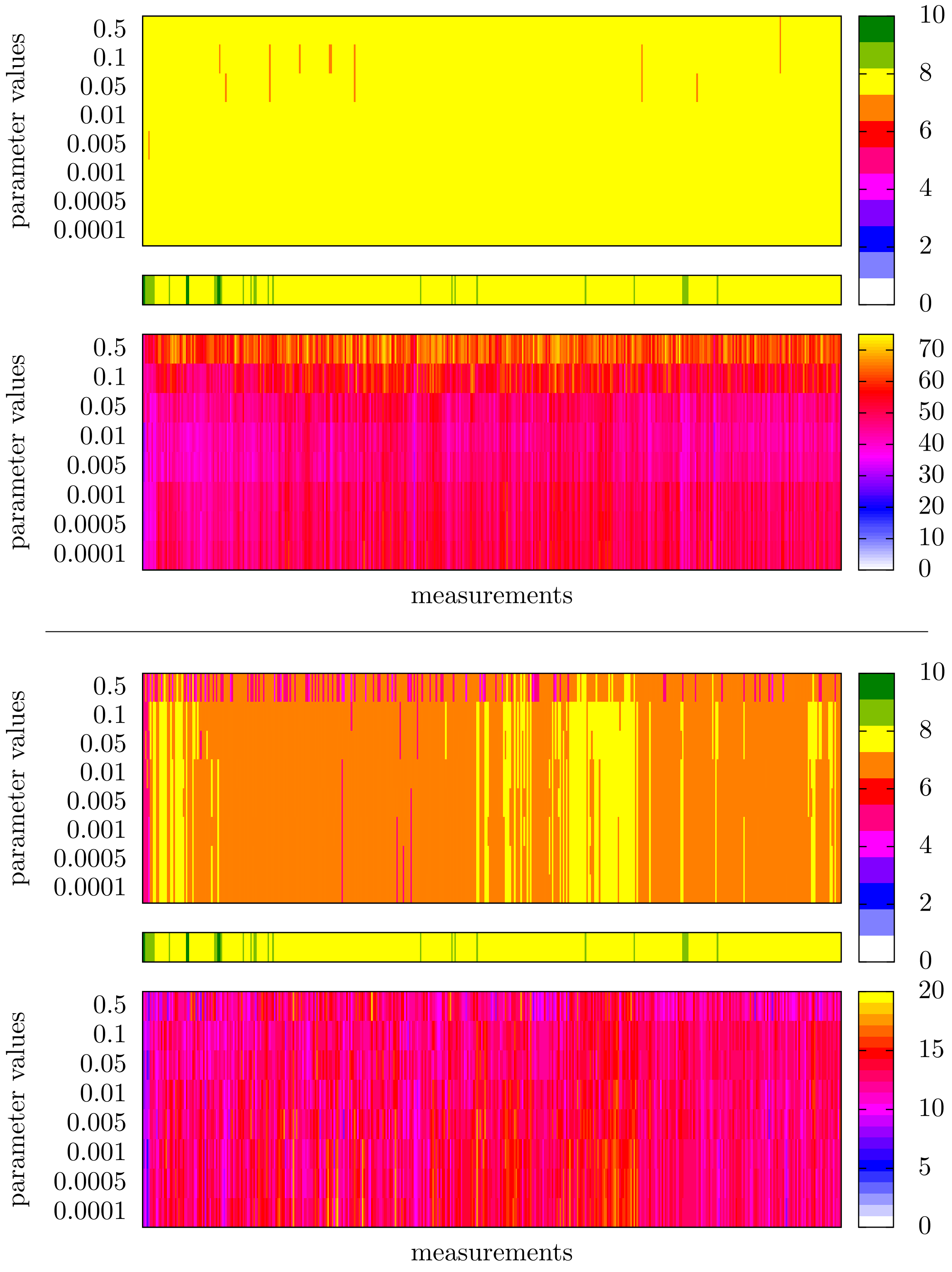


Figure 9.3: Experiment for `convergence_threshold` containing two figure units (upper) for PME and (lower) for OPME. Every unit consists of: (top) color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, the single row indicating the actual number of present peaks within the measurements; (bottom) color-coded matrix containing the **FP** rate.

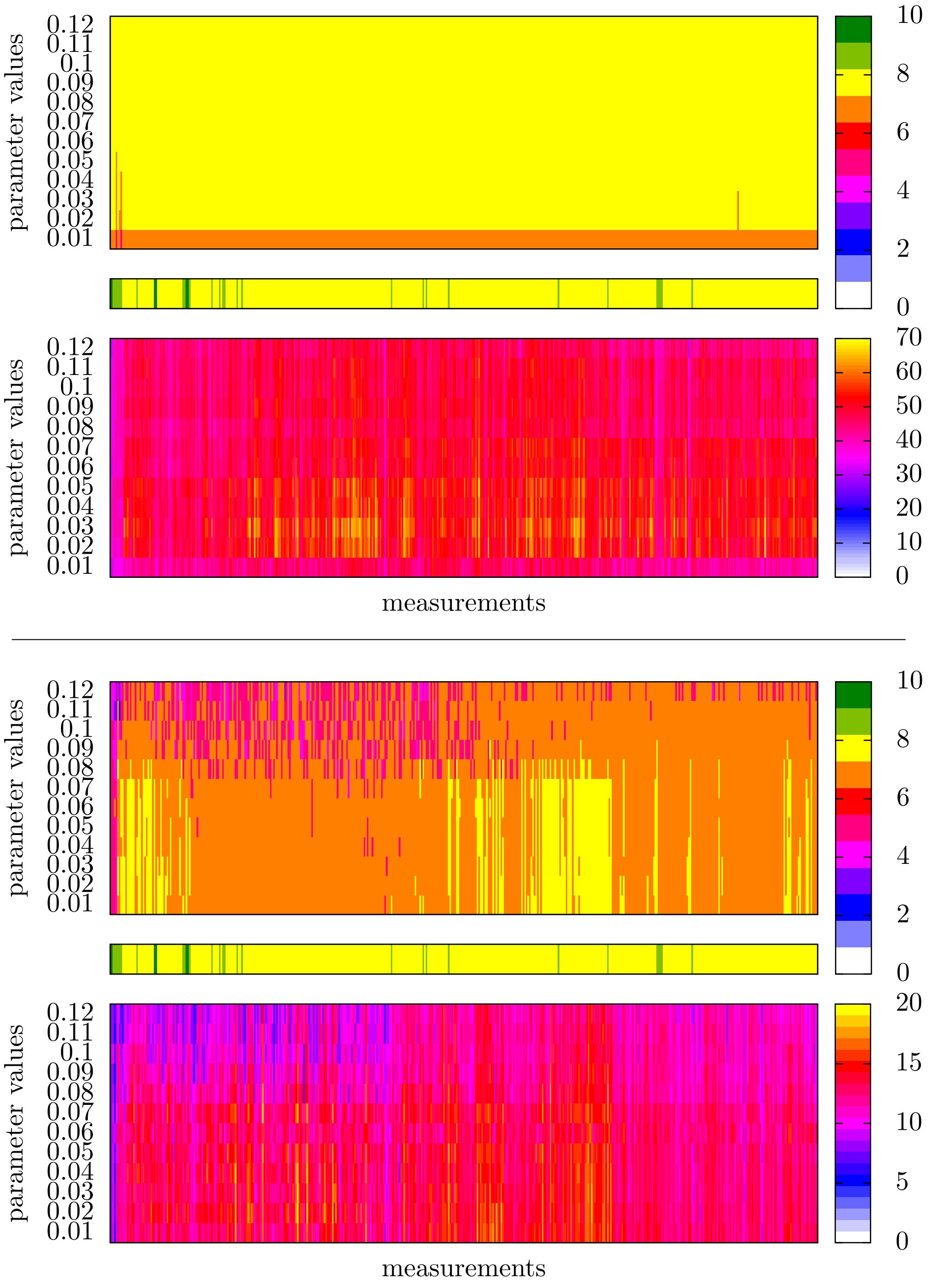


Figure 9.4: Experiment for `r_width_factor` containing two figure units (upper) for PME and (lower) for OPME. Every unit consists of: (top) color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, the single row indicating the actual number of present peaks within the measurements; (bottom) color-coded matrix containing the **FP** rate.

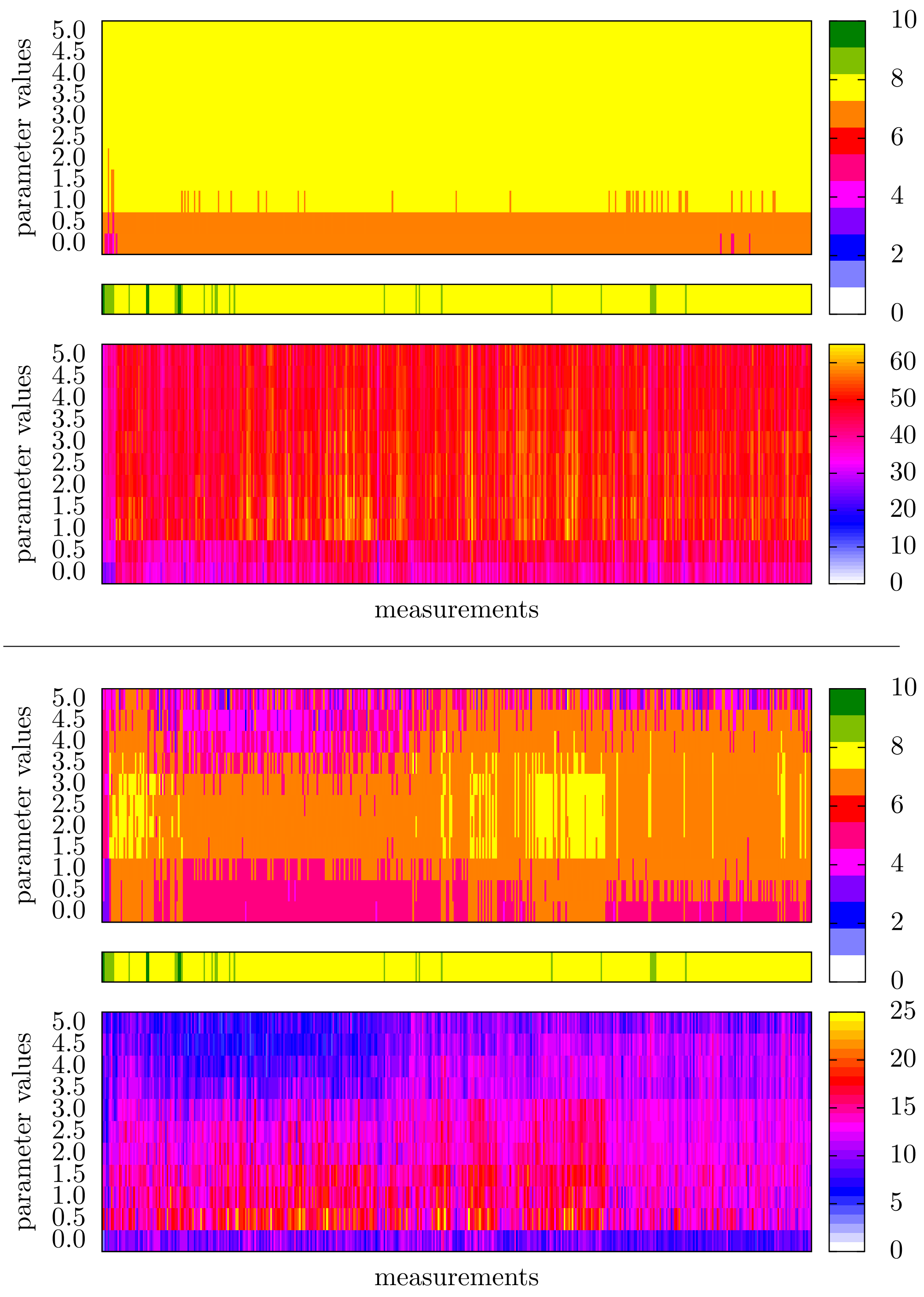


Figure 9.5: Experiment for `r_width_offset` containing two figure units (upper) for PME and (lower) for OPME. Every unit consists of: (top) color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, the single row indicating the actual number of present peaks within the measurements; (bottom) color-coded matrix containing the **FP** rate.

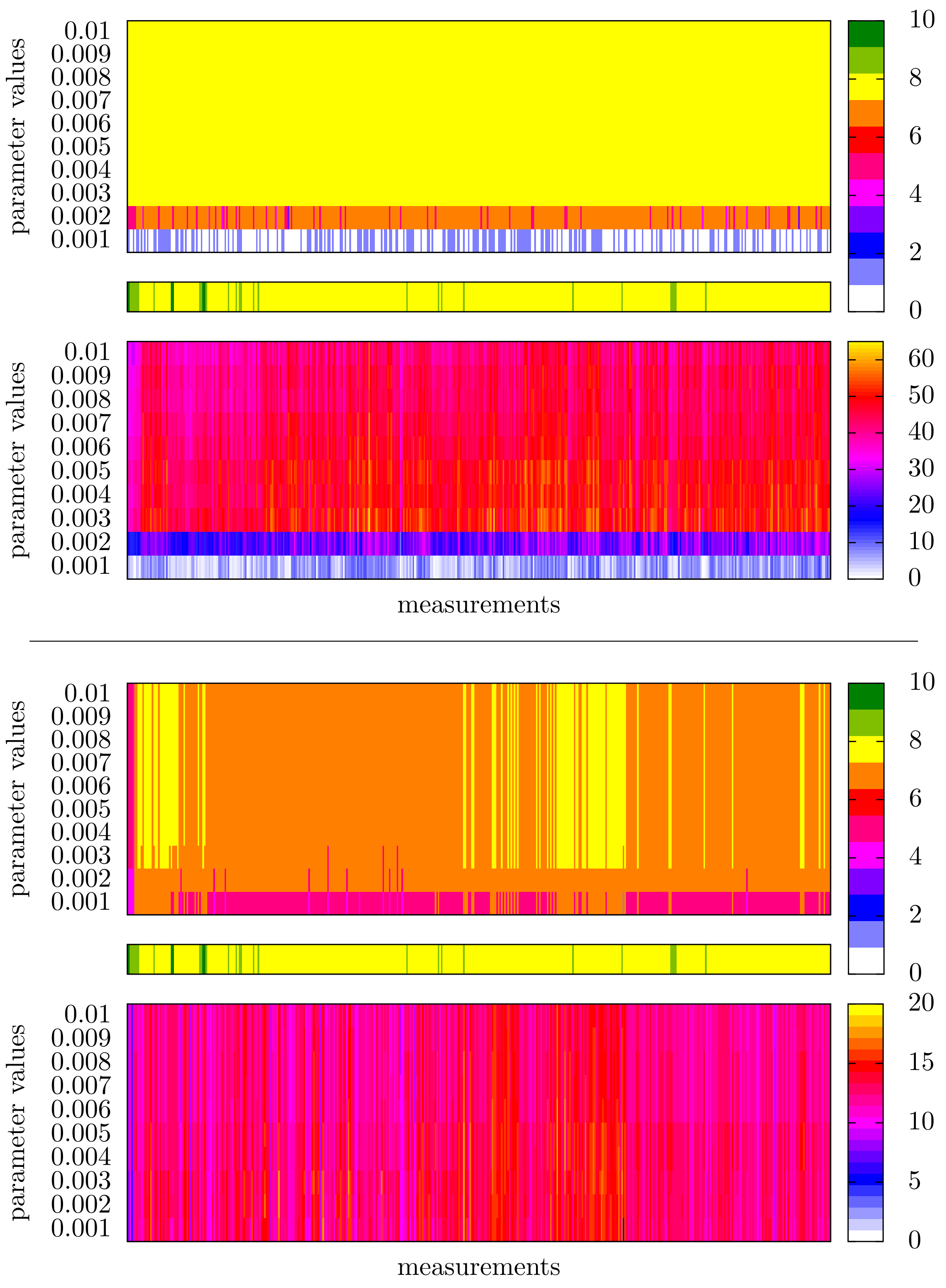


Figure 9.6: Experiment for `t_width_offset` containing two figure units (upper) for PME and (lower) for OPME. Every unit consists of: (top) color-coded matrix containing the counts (**TP**) for every pair of measurement / parameter values, the single row indicating the actual number of present peaks within the measurements; (bottom) color-coded matrix containing the **FP** rate.

10 Evaluation of Classification Quality with Two Classes of Data

In the last experiment we evaluate whether both peak extraction methods with the consecutive clustering method can compete with a hand-made extraction. As a quality measure, we use a two-class dataset and measure the accuracy when applying several classification methods with these data.

10.1 Related Work

Several two-class classification methods are known from literature. Here we briefly introduce these methods utilized for our experiment. For convenience, we call a consensus peak a variable within the classification context.

Support Vector Machine (SVM)

The objective of support vector machines is to detect boundaries separating the observations of two classes. Hereby, the points have to be as far away as possible from the boundary. For details consider the literature (Theodoridis and Koutroumbas, 2009, Chapter 3.7).

Linear SVM

The linear SVM determines a hyperplane separating the observations. Data points which violate the boundary by lying on the wrong side of the hyperplane (according to their class) are penalized with a cost parameter.

Radial Basis Function (RBF) SVM

Since a linear decision boundary might often be unsuitable, other boundaries can be achieved by transforming the data into a higher dimension and then searching for a linear boundary in this space. To optimize this boundary, the algorithm does not have to actually compute the transformations. A so-called kernel function calculates the necessary dot products between two transformed observations without computing the actual values first. One kernel function is the radial basis function with $K(x_i, x_j) = \exp\{-\gamma |x_i - x_j|^2\}$

for two points x_i and x_j. Besides the cost parameter, the second tuning parameter is $\gamma > 0$.

K-Nearest-Neighbor (kNN)

A simple idea to classify a new observation is to assign it to the class to which most of its closest neighbors belong to. The kNN algorithm is referred to as a lazy learning method, as in contrast to an eager learning method the classification training is applied during the request. The k closest points (in terms of Euclidean distance considering all variables) decide per majority vote. The number of considered neighbors k is treated as tuning parameter, taking the integers from one to ten in the inner cross-validation. A detailed description of kNN can be found in literature (Duda et al., 2000, Chapter 4.4).

Classification Tree (CT)

In a two-class classification problem, a classification tree splits the training set into two groups using a simple cut-point on a single variable as briefly described in (Theodoridis and Koutroumbas, 2009, Chapter 4.20). The procedure progresses iteratively for the two resulting sets, which means they are also split regarding one variable until a stopping criterion is reached. A set of observations in the tree is called node. To decide which variable and which cut-point should be used at each step, the Gini index is used to measure the decrease of impurity achieved by a certain split. A node is considered pure if it contains only observations of the same class. If the node impurity does not improve by a factor of 0.01, the node is not split any more. Since large trees with many splits and a considerable depth are prone to overfitting, the size of a terminal node is not split any further can be restricted. The parameter specifies the minimum number of observations in a terminal node.

Generalized Boosted Models (GBM)

Boosting is a strategy to combine many weak learners into a strong one as introduced by Friedman (2001). A set of simple trees with a limited quantity of nodes (interaction depth is a tuning parameter) is iteratively generated. The next tree is chosen such that it minimizes a loss criterion (here the deviance, the negative log-likelihood of the Bernoulli model) based on the current model. To determine the next tree, just 50% of the training data is used.

Random Forest (RF)

A Random Forest is a collection of many trees, each of which is based only on a random selection of $\sqrt{k}$ of the available variables and a bootstrap sample of size n. Each tree is fully grown, meaning that there are no restrictions in regard to minimum node size. To classify an observation, it is passed trough all trees and the class is assigned by majority vote.

10.2 Setting

We are given data tables which result from peak clustering and contain n observations and a differing number of k consensus peaks. For this experiment the dataset Pseudomonas aeruginosa (as described in Appendix A.2) was utilized. We apply different classification algorithms to each data table as described by Horsch et al. (2015). See Hastie et al. (2009) for an overview and the references after the descriptions of the methods for explicit implementations. We use 10-fold cross-validation (CV), i.e., we split our observations in 10 equally-sized groups, train our classification algorithm, sequentially on 90% of the data (training set) and predict the left out 10th partition of the data (test set) with the resulting model. In case of necessary parameter tuning for the classification algorithm a nested 10-fold CV is performed. Since the results of the CV depend on the 10 random splits, we repeat each procedure of classification 50 times.

A classification process of a dataset containing IMSC measurements is subdivided into three steps, namely a combined preprocessing and peak detection, peak clustering and the training of a classification model itself. For peak detection we use both our offline (PME) and online (OPME) methods and the automatic detection in VisualNow, Local Maxima (described in Section 3.1) as well as PDSA and SGLTR (see Section 4.1) for the sake of comparison. Of course the manual peak detection using VisualNow (which is state of the art) is also involved. The parameters for our methods are adjusted as listed in Table 8.1. To cluster the peak lists provided by every extraction method for all measurements, we utilize our augmented EM clustering as well as cluster editing, grid squares and DBSCAN (consider Section 5.2 for details). Finally, the data tables obtained from the previous clustering are classified using all classification methods described in Section 10.1. We apply a Cartesian product of all three steps. Since both manual peak extraction and clustering are one step within VisualNow, no manually extracted peak lists are provided and they are not combined with the remaining clustering methods. As a second exception, VisualNow is not able to cluster provided peak lists and therefore some combinations can not be deployed. In total, 156 combinations are applied as illustrated in Figure 10.1.

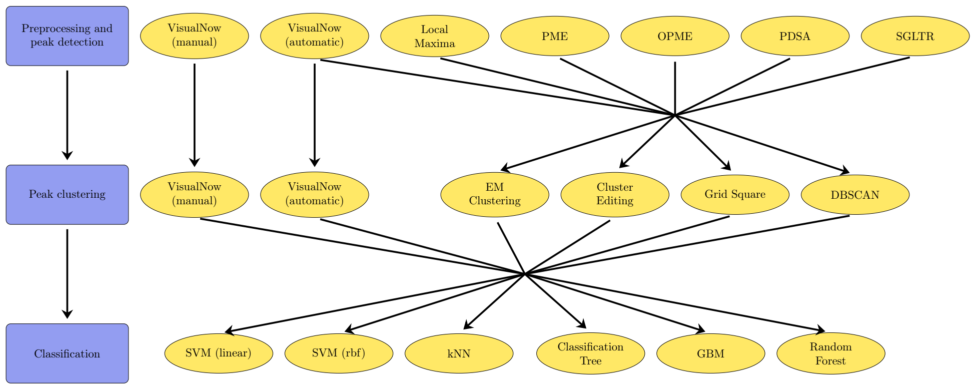


Figure 10.1: Possible combinations for the three steps in the analysis process from raw data to classification results as presented by Horsch et al. (2015). Not all peak detection and peak clustering methods can be combined. All results from peak detection are input for all classification methods.

10.3 Results

We now evaluate all meaningful combinations of peak detection, peak clustering, and classification algorithms on the dataset as initially described by Horsch et al. (2015). After peak detection and peak clustering, we have 26 data tables of peak intensities as input for the consecutive classification step. The number of consensus peaks and thus variables varies over the datasets. Whereas the VN variants find many consensus peaks – 224 for VN (manual) and 236 for VN (auto) combined with its own peak clustering method – the other methods identify much less, from 11 for OPME combined with Grid Squares up to 67 for SGLTR combined with EM clustering. This may indicate that the automated versions often miss or merge existing peaks and thus corresponding metabolites in the exhaled air. Still we will see that classification accuracy does not decline significantly.

We now consider all 156 analysis processes separately. Figure 10.2 shows the results in terms of accuracy. Each of the 156 boxplots represents one of the combinations of peak detection, peak clustering and classification. Each box is based on 50 points, the replications of the 10-fold CV. The six panels contain the results of the different classification processes. The horizontal line at an accuracy of 0.8 serves as orientation. In each panel of Figure 10.2, the boxes are ordered according the peak detection methods (annotation on the x-axis) and colored according peak clustering methods. Considering just RF and GBM, the best peak detection methods are combinations with LM, PME and SGLTR, depending on the peak clustering methods. Considering only these classification and peak detection methods, Grid Squares most frequently leads to the best accuracies. The combination of OPME and Grid Squares seems to be considerably inferior to all other combinations, as it displays a

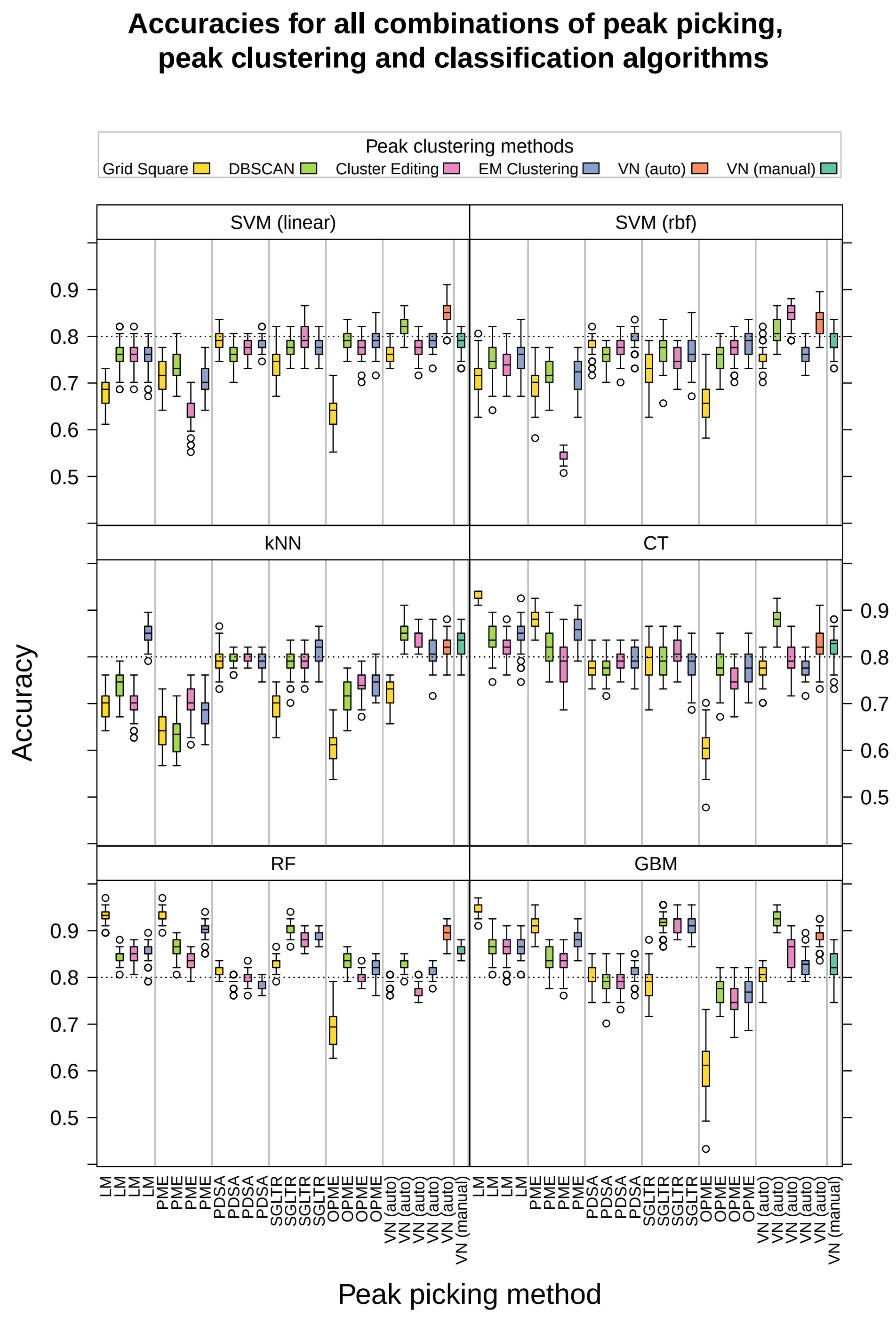


Figure 10.2: Accuracies for all combinations of peak detection, peak clustering and classification processes provided by Horsch et al. (2015).

Table 10.1: Median performance measures for all PME / EM Clustering and classification combinations, OPME / EM Clustering and classification combinations, VN (manual) and classification combinations as well as the best classification process.

Peak detection / clustering	*Classification*	*Accuracy*	*Sensitivity*	*Specificity*	*AUC*
PME / EM Clustering	SVM (linear)	0.701	0.533	0.865	0.776
PME / EM Clustering	SVM (rbf)	0.724	0.533	0.865	0.783
PME / EM Clustering	kNN	0.687	0.567	0.784	0.718
PME / EM Clustering	CT	0.858	0.800	0.919	0.895
PME / EM Clustering	RF	0.903	0.900	0.919	0.918
PME / EM Clustering	GBM	0.881	0.833	0.919	0.905
OPME / EM Clustering	SVM (linear)	0.791	0.633	0.919	0.859
OPME / EM Clustering	SVM (rbf)	0.791	0.633	0.892	0.844
OPME / EM Clustering	kNN	0.746	0.600	0.892	0.759
OPME / EM Clustering	CT	0.776	0.700	0.838	0.781
OPME / EM Clustering	RF	0.821	0.700	0.919	0.851
OPME / EM Clustering	GBM	0.769	0.600	0.892	0.780
VN (manual)	SVM (linear)	0.791	0.700	0.865	0.860
VN (manual)	SVM (rbf)	0.776	0.767	0.811	0.863
VN (manual)	kNN	0.836	0.750	0.892	0.857
VN (manual)	CT	0.828	0.800	0.851	0.859
VN (manual)	RF	0.851	0.733	0.973	0.923
VN (manual)	GBM	0.821	0.733	0.919	0.860
LM / Grid Square	GBM	0.940	0.967	0.946	0.960

lower median and higher variability of accuracy. That is not unexpected due to the simplicity of Grid Squares. The best results are achieved using random forest for the classification in which PME / EM Clustering clearly exceeds the VN (manual) detection in terms of accuracy, consider Table 10.1. OPME barely keeps up with the manual detection when providing a classification utilizing random forest again. In general, it is visible that the combination of peak extraction and clustering can compete with the state of the art manual peak extraction.

11 Conclusion and Discussion

In the final section we give a brief summary of all methods and tools introduced in this thesis. Afterwards we discuss some aspects of automated peak extraction and biological data in general and give an outlook for a possible continuation of this work.

11.1 Summary

Approximating the Exponential Function

The first method we introduced is a fast approximation of the exponential of a number. This method exploits the floating point representation within processors of real values to increase the performance while keeping the error low. We evaluated that our further algorithms utilize the exponential function up to billion times in a high-resolution measurement, which justifies a high-performance computation – especially in a resource-constrained application. Since all further algorithms use statistical distributions, we designed this approximation with respect to negative numbers. With the described methods and tricks we can decrease the computation time. Our performance tests show that the approximation is (depending on the processor) 4–6 times faster than exact functions. On the other hand, the error between the approximation and the corresponding exact result remains low; the highest relative error is about 0.001%. This result makes application in a statistical context very reasonable.

Offline Peak Model Estimation in PEAX

The peak model estimation for an offline application (PME) was initially designed to investigate the possibilities during the implementation of its subprocesses for enhancements into an online context. PME is allowed to access a whole IMSC for the peak extraction at any time. It is capable of estimating parameters for an introduced peak model describing the complete shape of a peak and providing a peak list with all parameters of the corresponding peaks. This peak model is helpful for any consecutive post-processing step. The adjustable parameters for PME are necessary, especially when adjustments at the spectrometer device alternate. We show that the proposed peak model is reasonable and describes the peaks well after estimation. Accordingly, we present that PME is suitable as a preprocessing step itself for a classification context. In combination with EM Clustering, PME reaches one of the best accuracy results (90.3%) within the classification evaluation.

Online Peak Model Estimation

The second introduced peak extraction method was dedicated to operate in a resource-constrained context. Thus, the online peak model estimation (OPME) was designed to analyze consecutively captured IM spectra only once, discard these spectra immediately and store only data in a highly compressed type, e.g. with statistical models. OPME also provides a peak list in which all model parameters of the according peaks are listed. We show that the time restriction of 100 ms to process a single IM spectrum is kept, even with high-resolution spectra on embedded systems (about 97 ms). This computation time can again be decreased by over-clocking the processor, relaxing the convergence threshold or averaging the IM spectra. Considering the accuracy of modeling single IM spectra, we show that over 90% of the IM spectra models reach a perfect agreement with the real spectra and about 99% of the spectra achieve an accuracy of over 90%. However, OMPE achieves classification evaluation accuracies which are slightly below the accuracies of manual peak extraction but still comparable.

Adaptive EM Clustering

We combined the advantage of providing an underlying distribution model for clusters with independence of a predetermination of the cluster number. We introduced the adaptive EM Clustering, which is capable of both considering an underlying model for clusters and dynamically adjusting the number of clusters. This method outperforms existing clustering methods which also dynamically determine the number of clusters. Another advantage is the provision of peak parameters by the peak extraction which can be exploited for the clustering. Since the application should also be located on a resource-constrained device, we presented a number of methods and techniques to increase the performance without noticeably losing any precision. Although the resulting tool is designed for MCC/IMS data, it can be easily reimplemented for other spectrometry data.

Comparison with a Reference

The final introduced method presented an approach for aligning two peak lists. Ideally, a reference peak list is provided with already determined corresponding compounds of the peaks. In contrast to the clustering approach, this method takes a constant and linear shifting of the peak sets under consideration. Due to the rather limited number of reference peaks or small peak lists provided by any of the peak extraction methods or the clustering, no need emerged to enhance the method for embedded systems, since it performs well on real case scenarios. Nevertheless, the idea of enhancing the EM Clustering can also be adopted for the comparison. The method was utilized for the evaluation of the adjustable program parameters for PME and OPME.

11.2 Discussion

Automated peak extraction will always be confronted with false positive and false negative detections. It remains a trade-off which of both cases provides more disadvantageous consequences. Of course, wrongly undetected peaks for a single measurement are detrimental. Considering a whole set of measurements, however, missing single peaks in single measurement becomes less critical. When clustering to consensus peaks over measurements and providing, e.g. time series, both outliers and missing values can be detected and corrected with appropriate methods. On the other hand, a faulty detection of peaks can also create complications, especially when considering single measurements again. But as in the previous case, the number of critical outliers biasing the results declines when more measurements are provided and clustered. Since false positives are usually uniformly distributed, they should be eliminated by a clustering. When a classification model is provided, the presence of false positives is also less critical – as long as true positives are found with a high confidence. As in many other biological fields, it is also challenging to evaluate the methods since we do not know the basic truth of our data. One way is to annotate the data by domain experts as we did, but many evaluations suffer from treating human annotation as gold standard: even in the best case they can only show that the evaluated methods are as good as the human annotation.

We presented the first methods capable of automated peak extraction in various application fields. Their unique feature is that these methods provide a complete description of the peak shape instead of the location of peaks mode only. All methods were implemented, and especially the methods dedicated for an online application (OMPE and EM Clustering) were optimized in terms of computation time. Surely, there is space for improvement. The PME suffers from a high false positive rate. This could probably be solved by increasing involvement of the peak model within the peak candidate detection step. On the other hand, OMPE displays a tendency towards a high false negative rate. It was a design decision to process an IMSC only, considering single IM spectra and not a set of spectra at once. A smoothing step over several IM spectra could probably increase the sensitivity, especially for detecting peaks slightly exceeding the determination limit. Features like the detection of monomer / dimer pairs were omitted and could be investigated in future work, just as properties of measurements captured in negative mode. At the moment, `r_width_offset` and `r_width_factor` are adjustable parameters, but can perhaps be inferred by device adjustments like MCC temperature of carrier gas flow.

To conclude, we believe that we accomplished our objectives and provided some powerful tools. Although OPME is a prototype and thus has to be implemented on a target system, the remaining tools are ready for application in practice.

A Datasets

Several datasets were utilized for certain aspects of the evaluation. All datasets were provided by B&S Analytik (Dortmund, Germany). Generally, these datasets can be separated into two groups: measurements captured under laboratory conditions for high reproducibility and measurements captured in hospitals for clinical studies with patients suffering from the same disease within a dataset. The diseases are known but irrelevant with respect to methods being evaluated within this thesis and cannot be disclosed due to confidentiality agreements within the clinical study approved by the state ethics committee. Since these datasets are confidential, they cannot be published along with this thesis. In the following, all utilized datasets are described.

A.1 Laboratory datasets

Calibration Curves

A calibration curve is a set of measurements to determine a correlation between the concentration of a certain analyte and the highest signal intensity (or volume) of its corresponding peak. These data sets have in common, that all measurement conditions are equal, i.e., equidistant measurement starting points (every minute one measurement is started), a 1×5 aggregation kernel (5 signal values within an IM spectrum are aggregated to their mean value), one measurement takes 30 s (hence it contains 300 IM spectra) and the curves are exponential thinning series. The thinning is performed until the concentration of the analyte drops below the determination limit and no signal of the corresponding peak is visible anymore. The adjustment of the BioScout device is different for these two data sets: the MCC temperature is 90 °C and the MCC carrier gas flow is $250\,\mathrm{ml\,min^{-1}}$. Two calibration curve data sets are available, the first contains 1336 measurements and the second contains 2209 measurements. The concentration of the analyte drops below the determination limit only within the first data set.

Reference 6 mixture

This dataset contains a set of 446 measurements captured with standard conditions (consider Table 1.1). Here, a defined set of six already known reference compounds was injected. All measurements are averaged with a 5×5 kernel. A peak list with peak parameters of the corresponding compounds was provided. Since the concentration of all compounds was rather high, almost all compounds produced monomers as well as dimers within the spectra.

A.2 Datasets from clinical Studies

Dataset 69

Captured for a clinical study, this dataset contains 69 measurements, of which 39 are from different patients suffering from the same disease and 30 from a control group not showing corresponding symptoms. This dataset was also used for classification by Hauschild et al. (2013) and Egorov et al. (2013). All measurements were performed within ten days and within the same hospital, captured for ten minutes and their IMSCs were averaged with a 5×5 kernel. A manual analysis of the dataset provided a consensus peak list with 60 peaks.

Pseudomonas aeruginosa

The second clinical two-class dataset contains infected patients and a healthy control group. Exhaled breath was measured from 30 patients whose airways are either infected or colonized by *Pseudomonas aeruginosa* and from 37 healthy non-smoker control volunteers. All patients were recruited from the Department of Pulmonology, Ruhrlandklinik, University Hospital of Essen, Germany, with no evidence of acute exacerbation for at least 4 weeks prior to enrollment. The diagnosis of Pseudomonas was established according to up-to-date guidelines. All healthy volunteers were employees of the hospital. The study was approved by the ethic committee of the University of Duisburg–Essen, with informed consent of all subjects. On an only slightly different dataset, Rabis et al. (2011) identified single peaks with differential intensities between the two groups. Here, 224 consensus peaks are extracted.

Dataset 508

Another dataset from a clinical study provides 508 measurements which can be subdivided into 348 measurements of diseased patients and a control group of 160 probands. This is the biggest dataset we consider. All measurements were captured for ten minutes and averaged with a 5×5 kernel. Again, a manual analysis by a domain expert provided a consensus peak list containing 156 peaks.

Rats dataset

In another clinical study by Wolf et al. (2014), 15 rats were monitored at 20 minute intervals for up to a day. Each rat resulted in 30–40 measurements (a time series) for a total of 515 measurements. The acquisition time for every measurement was about twelve minutes and a 5×5 kernel was used for averaging. A manually generated data table was provided for every rat. Over all, 100 consensus peaks were extracted.

B Contributions to co-authored Articles

This thesis contains several results from already peer-reviewed publications with other researchers. The following papers contain shortened versions of the corresponding section within this thesis:

Chapter 3

I stated the introduced peak model describing an approximation of the complete shape of a peak with statistical distributions using seven parameters. To estimate these parameters, I implemented a method using maximum likelihood estimators. For the PEAX pipeline, I implemented the following modules: (*Preprocessing*) baseline correction, de-noising, smoothing; (*Peak Candidate Detection*) cross finding; (*Peak Picking*) EM clustering; (*Peak Modeling*) peak model estimation.

D. Kopczynski, J. Baumbach, and S. Rahmann. Peak modeling for Ion mobility spectrometry measurements. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1801–1805, New York, NY, USA, Aug 2012. IEEE.

M. D'Addario, D. Kopczynski, J. I. Baumbach, and S. Rahmann. A modular computational framework for automated peak extraction from ion mobility spectra. *BMC Bioinformatics*, 15(1):25, 2014.

Chapter 4

I developed the online peak model estimation approach and implemented a recourse-constrained version to keep the time restrictions.

D. Kopczynski and S. Rahmann. An Online Peak Extraction Algorithm for Ion Mobility Spectrometry Data. In *WABI*, volume 8701 of *Lecture Notes in Computer Science*, pages 232–246, New York, 2014. Springer.

Chapter 5

I developed the adaptive EM Clustering and I implemented a resource-constrained version to cope well on low-power devices and performed the evaluation.

D. Kopczynski and S. Rahmann. An online peak extraction algorithm for ion mobility spectrometry data. *Algorithms for Molecular Biology*, 10(1):17, 2015.

Chapter 10

For both evaluation papers, I performed the main peak extractions and clusterings.

A. C. Hauschild, D. Kopczynski, M. D'Addario, J. I. Baumbach, S. Rahmann, and J. Baumbach. Peak Detection Method Evaluation for Ion Mobility Spectrometry by Using Machine Learning Approaches. *Metabolites*, 3(2):277–293, 2013.

S. Horsch, D. Kopczynski, J. I. Baumbach, J. Rahnenführer, and S. Rahmann. From raw ion mobility measurements to disease classification: a comparison of analysis processes. *PeerJ PrePrints*, 3:e1591, 2015.

Bibliography

E. Aguilera-Herrador, S. Cárdenas, V. Ruzsanyi, S. Sielemann, and M. Valcárcel. Evaluation of a new miniaturized ion mobility spectrometer and its coupling to fast gas chromatography multicapillary columns. *Journal of Chromatography A*, 1214(1–2):143–150, 2008.

D. Arthur and S. Vassilvitskii. K-means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035, Philadelphia, USA, 2007. Society for Industrial and Applied Mathematics.

S. Bader, W. Urfer, and J. I. Baumbach. Processing ion mobility spectrometry data to characterize group differences in a multiple class comparison. *International Journal for Ion Mobility Spectrometry*, 8:1–4, 2005.

S. Bader, W. Urfer, and J. I. Baumbach. Preprocessing of ion mobility spectra by lognormal detailing and wavelet transform. *International Journal for Ion Mobility Spectrometry*, 11(1-4):43–49, 2008.

J. I. Baumbach and G. A. Eiceman. Ion Mobility Spectrometry: Arriving On Site and Moving Beyond a Low Profile. *Appl. Spectrosc.*, 53(9):338A–355A, 1999.

J. L. Bentley. Decomposable searching problems. *Information Processing Letters*, 8(5):244–251, 1979.

V. Bessa, K. Darwiche, H. Teschler, U. Sommerwerck, T. Rabis, J. I. Baumbach, and L. Freitag. Detection of volatile organic compounds (VOCs) in exhaled breath of patients with chronic obstructive pulmonary disease (COPD) by ion mobility spectrometry. *International Journal for Ion Mobility Spectrometry*, 14(1):7–13, 2011.

A. Björck. *Numerical methods for least squares problems.* Siam, 1996.

S. Böcker, S. Briesemeister, and G. W. Klau. Exact algorithms for cluster editing: Evaluation and experiments. *Algorithmica*, 60(2):316–334, 2011.

B. Bödeker and J. I. Baumbach. Analytical description of IMS-signals. *International Journal for Ion Mobility Spectrometry*, 12(3):103–108, 2009.

B. Bödeker, W. Vautz, and J. I. Baumbach. Peak finding and referencing in MCC/IMS-data. *International Journal for Ion Mobility Spectrometry*, 11 (1):83–87, 2008.

B. Bödeker, W. Vautz, and J. I. Baumbach. Visualisation of MCC/IMS-data. *International Journal for Ion Mobility Spectrometry*, 11(1-4):77–81, 2008.

H. Buchinger, S. Kreuer, R. Hellbrück, A. Wolf, T. Fink, T. Volk, B. Bödeker, S. Maddula, and J. I. Baumbach. Minimal retarded Propofol signals in human breath using ion mobility spectrometry. *International Journal for Ion Mobility Spectrometry*, 16(3):185–190, 2013.

A. Bunkowski. *MCC-IMS data analysis using automated spectra processing and explorative visualisation methods.* PhD thesis, Bielefeld University, 2011.

A. Bunkowski, B. Bödeker, S. Bader, M. Westhoff, P. Litterst, and J. I. Baumbach. MCC/IMS signals in human breath related to sarcoidosis – results of a feasibility study using an automated peak finding procedure. *Journal of Breath Research*, 3(4):046001, 2009.

B. Chazelle. Lower Bounds for Orthogonal Range Searching: I. The Reporting Case. *Journal of the ACM (JACM)*, 37(2):200–212, 1990.

R. C. H. Cheng and N. Amin. Maximum likelihood estimation of parameters in the Inverse Gaussian distribution, with unknown origin. *Technometrics*, 23(3):257–264, 1981.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms.* MIT Press, 2009.

M. D'Addario, D. Kopczynski, J. I. Baumbach, and S. Rahmann. A modular computational framework for automated peak extraction from ion mobility spectra. *BMC Bioinformatics*, 15(1):25, 2014.

K. Darwiche, J. I. Baumbach, U. Sommerwerck, H. Teschler, and L. Freitag. Bronchoscopically Obtained Volatile Biomarkers in Lung Cancer. *Lung*, 189(6):445–452, 2011.

B. Delaunay. Sur la sphère vide. a la mémoire de georges voronoï. *Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et na*, 6:793–800, 1934.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

DIN 32645. Chemische Analytik - Nachweis-, Erfassungs- und Bestimmungsgrenze unter Wiederholbedingungen - Begriffe, Verfahren, Auswertung, November 2008.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2Nd Edition).* Wiley-Interscience, 2000.

A. Egorov, A. König, M. Köppen, H. Kühn, I. Kullack, E. Kuthe, S. Mitkovska, A. Niehage, R. Pawelko, M. Sträßer, and C. Striewe. Ressourcenbeschränkte Analyse von Ionenmobilitätsspektren mit dem Raspberry Pi. Technical report, Faculty of computer science, TU Dortmund, 2013.

A. Einstein. Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen. *Annalen der Physik*, 322(8):549–560, 1905.

M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery and Data Mining (KDD), Proceedings of first international conference*, volume 96, pages 226–231, 1996.

R. G. Ewing, D. A. Atkinson, G. Eiceman, and G. J. Ewing. A critical review of ion mobility spectrometry for the detection of explosives and explosive related compounds. *Talanta*, 54(3):515–529, 2001.

T. Fink, J. Baumbach, and S. Kreuer. Ion mobility spectrometry in breath research. *Journal of Breath Research*, 8(2):027104, 2014.

S. S. Fong, P. Rearden, C. Kanchagar, C. Sassetti, J. Trevejo, and R. G. Brereton. Automated Peak Detection and Matching Algorithm for Gas Chromatography-Differential Mobility Spectrometry. *Analytical Chemistry*, 83(5):1537–1546, 2011.

E. B. Fowlkes and C. L. Mallows. A Method for Comparing Two Hierarchical Clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

J. H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

R. Furtwängler, A.-C. Hauschild, J. Hübel, H. Rakicioglou, B. Bödeker, S. Maddula, A. Simon, and J. I. Baumbach. Signals of neutropenia in human breath? *International Journal for Ion Mobility Spectrometry*, 17 (1):19–23, 2014.

L. Greengard and J. Strain. The Fast Gauss Transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

C. Halbfeld, B. E. Ebert, and L. M. Blank. Multi-Capillary Column-Ion Mobility Spectrometry of Volatile Metabolites Emitted by Saccharomyces Cerevisiae. *Metabolites*, 4(3):751–774, 2014.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning, Second Edition.* Springer Series in Statistics, New York, USA, 2009.

A. C. Hauschild, D. Kopczynski, M. D'Addario, J. I. Baumbach, S. Rahmann, and J. Baumbach. Peak Detection Method Evaluation for Ion Mobility Spectrometry by Using Machine Learning Approaches. *Metabolites*, 3(2): 277–293, 2013.

S. Horsch, D. Kopczynski, J. I. Baumbach, J. Rahnenführer, and S. Rahmann. From raw ion mobility measurements to disease classification: a comparison of analysis processes. *PeerJ PrePrints*, 3:e1591, 2015.

IEEE Computer Society. IEEE Standard for Binary Floating-Point Arithmetic. *ANSI/IEEE Std 754-1985*, pages 1–14, 1985.

M. Jünger, B. Bödeker, and J. I. Baumbach. Peak assignment in multi-capillary column–ion mobility spectrometry using comparative studies with gas chromatography–mass spectrometry for VOC analysis. *Analytical and bioanalytical chemistry*, 396(1):471–482, 2010.

M. Jünger, W. Vautz, M. Kuhns, L. Hofmann, S. Ulbricht, J. I. Baumbach, M. Quintel, and T. Perl. Ion mobility spectrometry for microbial volatile organic compounds: a new identification tool for human pathogenic bacteria. *Applied Microbiology and Biotechnology*, 93(6):2603–2614, 2012.

T. Keller, A. Schneider, E. Tutsch-Bauer, J. Jaspers, R. Aderjan, and G. Skopp. Ion mobility spectrometry for the detection of drugs in cases of forensic and criminalistic relevance. *International Journal for Ion Mobility Spectrometry*, 2(1):22–34, 1999.

R. Koczulla, A. Hattesohl, S. Schmid, B. Bödeker, S. Maddula, and J. I. Baumbach. MCC/IMS as potential noninvasive technique in the diagnosis of patients with COPD with and without alpha 1-antitrypsin deficiency. *International Journal for Ion Mobility Spectrometry*, 14(4):177–185, 2011.

M. Kolehmainen, P. Rönkkö, and O. Raatikainen. Monitoring of yeast fermentation by ion mobility spectrometry measurement and data visualisation with Self-Organizing Maps. *Analytica Chimica Acta*, 484(1):93–100, 2003.

D. Kopczynski and S. Rahmann. An Online Peak Extraction Algorithm for Ion Mobility Spectrometry Data. In *Algorithms in Bioinformatics*, volume 8701 of *Lecture Notes in Computer Science*, pages 232–246. Springer Berlin Heidelberg, 2014.

D. Kopczynski and S. Rahmann. An online peak extraction algorithm for ion mobility spectrometry data. *Algorithms for Molecular Biology*, 10(1):17, 2015.

D. Kopczynski, J. Baumbach, and S. Rahmann. Peak modeling for Ion mobility spectrometry measurements. In *Proceedings of the 20th European Signal Processing Conference (EUSIPCO 2012)*, pages 1801–1805, New York, NY, USA, 2012. IEEE.

I. Koutrouvelis, G. Canavos, and S. Meintanis. Estimation in the three-parameter inverse Gaussian distribution. *Computational statistics & data analysis*, 49(4):1132–1147, 2005.

A. E. Kreuder, H. Buchinger, S. Kreuer, T. Volk, S. Maddula, and J. Baumbach. Characterization of propofol in human breath of patients undergoing anesthesia. *International Journal for Ion Mobility Spectrometry*, 14(4): 167–175, 2011.

N. Kunze, C. Weigel, W. Vautz, K. Schwerdtfeger, M. Jünger, M. Quintel, and T. Perl. Multi-capillary column-ion mobility spectrometry (MCC-IMS) as a new method for the quantification of occupational exposure to sevoflurane in anaesthesia workplaces: an observational feasibility study. *Journal of Occupational Medicine and Toxicology*, 10(1):12, 2015.

Z. Liu, A. Abbas, B.-Y. Jing, and X. Gao. WaVPeak: picking NMR peaks through wavelet-based smoothing and volume-based filtering. *Bioinformatics*, 28(7):914–920, 2012.

S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

S. Maddula, T. Rabis, U. Sommerwerck, O. Anhenn, K. Darwiche, L. Freitag, H. Teschler, and J. I. Baumbach. Correlation analysis on data sets to detect infectious agents in the airways by ion mobility spectrometry of exhaled breath. *International Journal for Ion Mobility Spectrometry*, 14(4):197–206, 2011.

F. Maurer, A.-C. Hauschild, K. Eisinger, J. Baumbach, A. Mayor, and J. I. Baumbach. MIMA–a software for analyte identification in MCC/IMS chromatograms by mapping accompanying GC/MS measurements. *International Journal for Ion Mobility Spectrometry*, 17(2):95–101, 2014.

V. Mazet, D. Brie, and J. Idier. Baseline spectrum estimation using half-quadratic minimization. In *Proceedings of the 12th European Signal Processing Conference (EUSIPCO 2004)*, pages 305–308, 2004.

A. Munteanu and M. Wornowizki. Correcting statistical models via empirical distribution functions. *Computational Statistics*, pages 1–31, 2015.

A. Myronenko and X. Song. Point Set Registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12): 2262–2275, 2010.

S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, New York, 2nd edition, 2006.

R. Purkhart, A. Hillmann, R. Graupne, and G. Becher. Detection of characteristic clusters in IMS-Spectrograms of exhaled air polluted with environmental contaminants. *International Journal for Ion Mobility Spectrometry*, 15(2):63–68, 2012.

T. Rabis, U. Sommerwerck, O. Anhenn, K. Darwiche, L. Freitag, H. Teschler, B. Bödeker, S. Maddula, and J. I. Baumbach. Detection of infectious agents in the airways by ion mobility spectrometry of exhaled breath. *International Journal for Ion Mobility Spectrometry*, 14(4):187–195, 2011.

S. Rahmann, T. Wittkop, J. Baumbach, M. Martin, A. Truss, and S. Böcker. Exact and Heuristic Algorithms for Weighted Cluster Editing. In *Computational Systems Bioinformatics Conference*, pages 391–401, 2007.

R. Reichart and A. Rappoport. The NVI clustering evaluation measure. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 165–173. Association for Computational Linguistics, 2009.

M. Salleras, A. Kalms, A. Krenkow, M. Kessler, J. Goebel, G. Müller, and S. Marco. Electrostatic shutter design for a miniaturized ion mobility spectrometer. *Sensors and Actuators B: Chemical*, 118(1–2):338–342, 2006.

A. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

N. N. Schraudolph. A Fast, Compact Approximation of the Exponential Function. *Neural Computation*, 11(4):853–862, 1999.

G. E. Spangler and C. I. Collins. Peak shape analysis and plate theory for plasma chromatography. *Analytical Chemistry*, 47(3):403–407, 1975.

M. Teepe, W. J. Kang, A. Neyer, J. I. Baumbach, and H. Schmidt. Miniaturized ion mobility spectrometer. *International Journal for Ion Mobility Spectrometry*, 4(2):173–176, 2001.

S. Theodoridis and K. Koutroumbas, editors. *Pattern Recognition.* Academic Press, Boston, fourth edition edition, 2009.

G. Vivó-Truyols. Bayesian Approach for Peak Detection in Two-Dimensional Chromatography. *Analytical chemistry*, 84(6):2622–2630, 2012.

D. Vogtland and J. I. Baumbach. Breit-Wigner-Function and IMS-signals. *International Journal for Ion Mobility Spectrometry*, 12(3):109–114, 2009.

M. Westhoff, P. Litterst, L. Freitag, W. Urfer, S. Bader, and J. Baumbach. Ion mobility spectrometry for the detection of volatile organic compounds in exhaled breath of lung cancer patients. *Thorax*, 64(9):744–748, 2009.

M. Westhoff, P. Litterst, S. Maddula, B. Bödeker, S. Rahmann, A. N. Davies, and J. I. Baumbach. Differentiation of chronic obstructive pulmonary disease (COPD) including lung cancer from healthy control group by breath analysis using ion mobility spectrometry. *International Journal for Ion Mobility Spectrometry*, 13(3-4):131–139, 2010.

A. Wolf, J. I. Baumbach, A. Kleber, F. Maurer, S. Maddula, P. Favrod, M. Jang, T. Fink, T. Volk, and S. Kreuer. Multi-capillary column-ion mobility spectrometer (MCC-IMS) breath analysis in ventilated rats: a model with the feasibility of long-term measurements. *Journal of Breath Research*, 8(1):016006, 2014.

S. Zimmermann and S. Barth. A Miniaturized Ion Mobility Spectrometer for Detection of Hazardous Compounds in Air. In *International Solid-State Sensors, Actuators and Microsystems Conference (TRANSDUCERS 2007)*, pages 1501–1504, 2007.