DISSERTATION

Efficient Solutions to Modeling Time Series of NMR Spectroscopic Data

Dissertation

zur

Erlangung des akademischen Grades

doctor rerum naturalium (Dr. rer. nat.)

der Mathematisch-Naturwissenschaftlichen Fakultät

der Universität Rostock

vorgelegt von

Jan Hellwig, geb. am 13.04.1999 in Wismar.

Rostock, 16.07.2025.

https://doi.org/10.18453/rosdok_id00005643

Erstgutachter:	Prof. Klaus Neymeyr, Universität Rostock
Zweitgutachter:	Prof. Krzysztof Kazimierczuk, Uniwersytet Warszawski
Eingereicht am:	16.07.2025
Verteidigt am:	17.10.2025

Contents

List of Figures

List of Tables

Mathematical Notation

T	Set of points in time $t \in T$
X	Set of frequency channels $x \in X$
D	Data matrix containing row-wise the series of spectra $D(t,:)$
M / $M_{\mathcal{P}}$	Matrix containing row-wise the series of hard-modeled spectra $M(t,:)$
$\|\cdot\|_F / \|\cdot\|_2$	Frobenius matrix norm, or sum of squares norm
$d_W(\cdot,\cdot)$	1-dimensional Wasserstein 1-metric
m	Number of peaks present in D
$\mathcal{P}$	Set of all parameters of the model $M_{\mathcal{P}}$
$p(t)$	Parameter tuple defined as function over time
$c(t)$	Center or peak position parameter
$v(t)/w(t)$	Half-width parameters
$h(t)$	Height parameter
$\lambda(t)$	Convex combination parameter for pseudo-Voigt functions
$d(t)$	Distance parameter for multiplets
$\alpha(t)$	Linear combination coefficient
Lor	Set of all Lorentzians $L_{c,w}$
Gau	Set of all Gaussians $G_{c,w}$
$pVoi$	Set of all pseudo-Voigt functions $pV_{c,w,\lambda}$
Voi	Set of all Voivgt functions $V_{c,v,w}$
$k_{\mathcal{F}}(m,n)$	Max. # points, s.t. ambiguous solutions with m and n functions $\in \mathcal{F}$ exist
$\hat{f}$	Continuous Fourier transform of $f \in L^1(\mathbb{R})$
$\mathrm{supp}(f)$	Set of points $\{x \in \mathbb{R} \mid f(x) > 0\}$
F^{-1}	Quantile function of F
λ_2	Two-dimensional Lebesgue-measure
T_{sel}	Subset of spectra serving as spline knots
t_{init}	Element of T_{sel}, where optimization begins
bw	Bandwidth, defines several window widths
len_w	Number of spectra from T_{sel} simultaneously being optimized
deg	Degree of polynomial of the Savitzky-Golay filter
δ	Negative threshold for second derivatives used for peak detection
ε	Positive threshold used for peak detection
σ	Estimated standard deviation of noise
tol_*	Several thresholds used for multiplet detection and optimization variants
b	Batch size
η	Learning rate
$\mathrm{sig}(x)$	Standard sigmoid function
ρ_*	Correlation coefficients

Chapter 1

Analyzing Time Series of NMR Spectra

Nuclear magnetic resonance (NMR) spectroscopy is a powerful analytical tool for examining chemical compounds. Compared to other methods for analyzing chemical species, NMR is a noninvasive, highly sensitive, and very precise measurement technique. Furthermore, NMR can be used to extract quantitative, qualitative, and structural information about chemicals. It is therefore widely used in many fields, including food processing [35], pharmaceutical research [60], biological chemistry [106], medicine [34], and environmental chemistry [118], to name a few. For the determination of the three-dimensional structures of biological macromolecules using NMR spectroscopy, a Nobel Prize was even awarded in 2002 [133].

1.1 A Mathematical Perspective

This thesis does not aim to provide a comprehensive overview of NMR spectroscopy. For a detailed introduction to the topic, we refer to [80]. Nevertheless, from a mathematical point of view there are some interesting points worth discussing. NMR spectral data is obtained by first measuring the oscillations of the magnetic fields of nuclei, such as ^{1}H, ^{13}C, ^{19}F, and ^{31}P, in the presence of a strongly oscillating magnetic field. These oscillations are then detected by inducing a voltage in a *detection coil* and decay approximately exponentially. Hence, this type of data is called *free induction decay* (FID). Ideal FID data consists of a sum of sine and cosine functions of different frequencies and amplitudes, multiplied by a linear exponential decay. Next, the Fourier transformation is applied to the FID data. Assuming ideal FID data, the resulting Fourier transform creates a *spectrum* consisting of *Lorentz* functions, see Eq. (2.2) and Lem. 3.6.

In the presence of noise, two cases are generally considered, see also [131]. First, homoscedastic noise, which can be represented by adding Gaussian white noise to the FID data. After the Fourier transform, this added noise remains Gaussian, see [107] and [93, Sec. 2.10]. Second, linear heteroscedastic noise, which scales linearly depending on the amplitude of the FID data. This type of noise can be represented by multiplying Gaussian white noise to the FID data, [131, Eqs. (3)-(4)]. After the Fourier transform, heteroscedastic noise leads to deviations from the Lorentz profile and causes *Voigt* functions to appear, see Eq. (2.4) and Cor. 3.9. Therefore, an NMR spectrum can be mathematically defined as a sum of different Lorentz and Voigt functions with added Gaussian noise.

From a mathematical point of view, there are two oddities regarding NMR spectral data. First, the x-axis of an NMR spectrum that denotes the frequencies conventionally runs from right to left, i.e., higher frequencies are to the left and vice versa[1]. Second, the frequency

[1]Before 1970, two scales, τ and δ, were in use. Ultimately, the International Union of Pure and Applied Chemistry (IUPAC) recommended using only the inverted δ scale, see [89, Preamble] and [47].

axis is dimensionless. Since the frequencies of nuclei in different chemical compounds vary only by tiny amounts, the x-axis denotes the relative frequency difference from a standard compound, which in the case of ^{1}H and ^{13}C is tetramethylsilane (TMS), expressed in parts per million (ppm).

1.2 Advantages of Analyzing NMR Time Series

One property of nuclear magnetic resonance is the fact that the frequencies of the oscillating nuclei vary when external or internal conditions change. Spectra depend not only on the magnetic field strength, but on the number of nuclei, the structure of the molecule, the interactions between different nuclei inside and outside the molecule, the pressure, the pH value, etc. This enables the first application of NMR spectral analysis: the detection of chemical species and the extraction of information about their structure. For example, in ^{1}H NMR, the protons belonging to different functional groups of an organic compound behave differently, see [17] for a detailed introduction. Typically, structural or qualitative information is extracted by observing the peak positions, the appearance of special groups of peaks, called *multiplets*, the distance between multiplet peaks, and peak widths.

Conversely, quantitative information can also be obtained. This information is closely related to the values of the peak integrals, which depend on peak widths and heights.

Quantitative information e.g., concentrations of a mixture, is relevant for monitoring chemical reactions or other of processes. In this case, information about the reaction kinetics can be obtained by observing changes in concentration profiles over time. This creates the need for modeling NMR time series. Examples of using connected series of NMR spectra[2] include general reaction monitoring, e.g., using flow NMR [40, 43, 82], monitoring metabolomic reactions [6], observing changes in peak positions over time when the pH value changes [121], determining the effectiveness of catalysts at different temperatures [49], and many more.

Whenever concentrations and peak integrals are of interest, numerical integration is the intuitive and most commonly used computational solution. This relatively simple approach of obtaining peak integrals works well when the peaks are sufficiently separated. However, when the peaks are in close proximity, change position drastically, cross over, or change order, numerical integration of single peaks is often difficult and inaccurate. Additionally, the integration bounds must be adjusted for peaks that change position within the time series. In these cases, it is beneficial to first model each individual peak, or to *deconvolute* the spectrum. In some cases, defining the integration bounds and detecting the peaks is done manually. These manual steps are very time-consuming and limit the effectiveness of using NMR as an analytical tool.

Therefore, an automated approach is called for. Since the goals of analyzing NMR spectra are broad, this thesis aims to provide a general solution. Our proposed solution is to model the entire time series using peak model functions. This not only eliminates the need for manual peak picking and numerical integration by providing peak positions and integral values for every spectrum, it also enables a low-dimensional representation and an easy recreation of the entire time series using the peak parameters. Additionally, the models and the peak parameters facilitate further analysis as all relevant information from the data can be stored more easily accessible in the set of peak parameters.

Unlike other analytical techniques, such as liquid chromatography, gas chromatography, and optical spectroscopy, NMR spectra do not satisfy the LAMBERT-BEER law. This law ultimately enables the use of nonnegative matrix factorization (NMF) methods, which

[2] In this thesis, we colloquially call all such data *time series*, even though the second dimension can represent something different.

retrieve pure component spectra and concentration profiles by factorizing the data matrix. There, solutions can be found by using a simple principal component analysis (PCA) [72] or more sophisticated multivariate curve resolution (MCR) methods [58]. However, these NMF methods suffer from an intrinsic ambiguity. Therefore, a *set of all feasible solutions* can also be determined [42]. As it turns out, this type of ambiguity does not exist for NMR spectra, assuming they are sufficiently selective. For spectra with overlapping peaks, the additional information provided by time series can increase selectivity and mitigate the remaining ambiguity.

1.3 Overview of This Work

The main aim of this thesis is to present solutions to the modeling problem of NMR time series, which we define as an optimization problem in Ch. 2. Furthermore, we present common peak model functions and illustrate important properties. Lastly, we introduce the Wasserstein metric, a useful tool when dealing with spectral data.

Chapter 3 addresses the question of ambiguity versus uniqueness of solutions to the modeling problem. We find that, under certain weak assumptions and in the continuous setting, modeling NMR spectra has a unique solution. For discrete NMR spectra, we provide upper and lower boundaries on the number of points necessary to ensure a unique decomposition into peak model functions. However, the presence of noise can cause NMR spectra to have multiple feasible models with significantly different sets of parameters.

In Chapter 4, we present practical approaches to using numerical optimization algorithms, and compare possible solutions. Our main solution is an approach which reduces the optimization dimension while simultaneously increasing the convergence stability for the optimizer by optimizing only a few peak parameters and interpolating the rest using cubic spline interpolation. Furthermore, we discuss several variants of this optimization routine and their respective advantages and drawbacks.

Because the peak parameters, such as height, position, and half-width, correspond to measureable quantities, we can use a second type of approach involving the observation of peak structures within the spectra to obtain the peak parameters. In a sense, the most general version of this approach is to not define a fixed heuristic method for parameter prediction, as is described in Secs. 4.2.2 and 4.2.3. Rather, it is to allow an arbitrarily complex heuristic method, which itself needs to be optimized to achieve satisfactory quality. This can be achieved through machine learning. A neural network can be viewed as a heuristic prediction algorithm, where an NMR spectrum is input and parameters are output. Chapter 5 is entirely devoted to neural networks, their possible architectures, how to conduct training, and their applications in the context of NMR spectroscopy.

Ultimately, both types of approaches have their characteristic strengths and weaknesses. Typically, the second type produces results more quickly, but its models are not always of sufficient quality. In contrast, the first approach has the potential to achieve high-quality models, albeit at the expense of increased runtime. Therefore, we propose to combine both approaches and present a nonlinear optimization algorithm, which is initialized by parameter predictions from neural networks. Usually, this results in both a decrease in runtime and an increase in model quality. The results of extensive testing of both approaches and all optimization variants are presented in Chapter 6. We use three constructed and three experimental data sets for testing, and for measuring the runtime and the model quality.

Parts of this thesis have been published before, but have been overhauled and refurbished in this work. The mathematical discussion in Ch. 3 has been published in [52]. The spline-based approach in Ch. 4 has been published in [86] and the improvements and amendments using machine learning in Ch. 5 have been published in [53].

The implementation of the methods proposed in this thesis can be found on GitHub using the following link.

`https://github.com/CITlabRostock/nmr_timeseries_model`

The code is written mostly in `Matlab` and partially in `Python`.

Chapter 2

Mathematical Foundations

Hard modeling NMR spectral data is not a recent phenomenon. In 1952, Lorentz functions were used to fit NMR data [101]. Even earlier, in 1949 it was noted that for certain spectral shapes, the Lorentz function did not yield satisfying results [108]. Still, due to the underlying physical equations, it was assumed that an NMR spectrum consists of a linear combination of different types of model functions, for example, Lorentzians, Gaussians and Voigt functions. While the rise of more potent computational devices allowed for more complicated model functions to be used, computing the Voigt functions still remained expensive. Therefore, an approximation was introduced in 1968 [129]. This approximation uses a convex combination of Gauss and Lorentz functions, is easier to compute then Voigt functions, and has an additional degree of freedom compared to Lorentzians. From that point on, these so-called *pseudo-Voigt* functions have been thoroughly studied [4, 19, 66, 128], and used to model certain types of spectral data, such as NMR [81], X-ray photoelectron spectroscopy (XPS) [56, 116] and other types of spectroscopy [14, 79, 97, 108].

In this chapter we lay the necessary mathematical foundations for modeling time series of NMR spectra. First, we define NMR peaks, NMR spectra and entire time series of NMR spectral data mathematically. Second, we define the modeling problem as an optimization problem. Third, we discuss useful properties of common peak model functions. Finally, we introduce another metric for determining the model quality.

2.1 NMR Time Series

In the previous Chapter 1, we established that an NMR spectrum can be modeled by differently parametrized peak model functions. Therefore, from a pure mathematical point of view, we define NMR spectra as a sum of m parametrized peak model functions. As we discuss in Ch. 3, this relatively simple definition helps to address the questions of ambiguity and uniqueness, while still originating from physical reasoning.

To define NMR spectra, we first need to define what a peak is. In this general case we describe a modeled peak by a continuous *peak model function* $f_p : \mathbb{R} \to \mathbb{R}$, parametrized by a tuple $p \in \mathbb{R}^k$.

Definition 2.1. *Let $k \in \mathbb{N}$ and $p \in P \subseteq \mathbb{R}^k$. Then, a* peak *is defined by a continuous function $f_p : \mathbb{R} \to \mathbb{R}$, where for all $p \in P$, we have $f_p \in L^1(\mathbb{R})$.*

Since in NMR spectroscopy, the peak integrals correspond to the number of nuclei, we also assume that f is integrable. We can now define an NMR spectrum as a mere sum of different peak model functions.

Definition 2.2. *Let $k, m \in \mathbb{N}$, $p_1, \ldots, p_m \in P \subseteq \mathbb{R}^k$ and $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$. Then, an* NMR spectrum *s is defined as a weighted sum of parametrized peak model functions f_{p_i}.*

$$s : \mathbb{R} \to \mathbb{R}, \quad \textit{with} \quad s = \sum_{i=1}^{m} \alpha_i \cdot f_{p_i}. \tag{2.1}$$

In the setting of time series, we need to be able to vary the parameters for different points in time. Therefore, all parameter tuples $p_i(t)$ and linear combination coefficients $\alpha_i(t)$ can be viewed as a function of $t \in T$.

Next, we look at the data to which we want to fit the model. An NMR time series is obtained by performing single NMR scans in sequence for a set of points in time $t \in T$. Since a single spectrum can be written as a vector representing the amplitudes in the frequency domain X, a time series can be expressed using a data matrix $D \in \mathbb{R}^{|T|\times|X|}$. Each row $D(t,:)$ of D represents the spectrum measured at time $t \in T$ and each column $D(:,x)$ of D represents the amplitudes within the frequency channel $x \in X$. The data sets we apply our methods to are introduced in App. B.

Finally, let $d(\cdot,\cdot)$ be a metric, whose inputs are matrices of the same dimensions. Then, we can define the hard modeling optimization problem as follows.

Definition 2.3. *Let $\mathcal{P} = \{(\alpha_i(\cdot), p_i(\cdot)) : T \to \mathbb{R} \times P) \mid i \in [m]\}$ represent the set of parameters, let $M_{\mathcal{P}} \in \mathbb{R}^{|T|\times|X|}$ be defined by*

$$M_{\mathcal{P}}(t,x) = \sum_{i=1}^{m} \alpha_i(t) \cdot f_{p_i(t)}(x),$$

and let $D \in \mathbb{R}^{|T|\times|X|}$ be the data matrix. Then, the hard modeling problem *is to find $m \in \mathbb{N}$ and $\mathcal{P}$, such that $d(D, M_{\mathcal{P}}) \to \min$. Furthermore, $M_{\mathcal{P}}$ is called a* model *of the time series D.*

2.2 Peak Model Functions

In the literature, three different types of model functions are often used to describe NMR peaks. Typically, Lorentzians are used, when perfect acquisition of NMR spectral data without noise or other interferences can be assumed.

Definition 2.4. *Let $c \in \mathbb{R}$ and $w \in \mathbb{R}^+$. Then, a* Lorentzian $L_{c,w}(x) : \mathbb{R} \to \mathbb{R}$ *is defined by*

$$L_{c,w}(x) = \frac{w^2}{w^2 + (x-c)^2}. \tag{2.2}$$

The parameter c is called center *or* peak position *and $w > 0$ is called* half-width*. Furthermore, the set of all possible Lorentzians is denoted by $Lor := \{L_{c,w} \mid c \in \mathbb{R}, w \in \mathbb{R}^+\}$.*

For the other two commonly used model functions, we must first define a Gaussian.

Definition 2.5. *Let $c \in \mathbb{R}$ and $w \in \mathbb{R}^+$. Then, a* Gaussian $G_{c,w}(x) : \mathbb{R} \to \mathbb{R}$ *is defined by*

$$G_{c,w}(x) = \exp\left(-\ln(2) \cdot \frac{(x-c)^2}{w^2}\right). \tag{2.3}$$

Again, c represents the center *and w the* half-width*. Furthermore, the set of all possible Gaussians is denoted by $Gau := \{G_{c,w} \mid c \in \mathbb{R}, w \in \mathbb{R}^+\}$.*

If Gaussian noise is present in the FID data, then the peaks in the Fourier transformed spectral data cannot fully be represented by Lorentzians, but rather by a convolution of Gaussians and Lorentzians, called *Voigt* functions, see also Sec. 1.1.

Definition 2.6. *Let $c \in \mathbb{R}$ and $v, w \in \mathbb{R}^+$. Then, a* Voigt *function $V_{c,v,w}(x) : \mathbb{R} \to \mathbb{R}$ is defined by*

$$V_{c,v,w}(x) = (G_{0,v} * L_{c,w})(x) = \int_{\mathbb{R}} G_{0,v}(y) \cdot L_{c,w}(x-y)\mathrm{d}y. \tag{2.4}$$

Here, the parameter c represents the center. *However, the* half-width *cannot be explicitly described in terms of v, w, but can only be approximated [5]. Furthermore, the set of all possible Voigt functions is denoted by* $Voi := \{V_{c,v,w} \mid c \in \mathbb{R}, v, w \in \mathbb{R}^+\}$.

Because of the complicated definition by a convolution integral, it is difficult to evaluate Voigt functions at any point analytically, even though there are results regarding an easier analytic description [87, 110]. Despite the fact, that for quite some time lookup tables exist for the Voigt function [12, 122], it is common to approximate the Voigt function [4]. In this thesis, we opt for the commonly used convex combination of a Gaussian and a Lorentzian, in order to facilitate understanding and computation. These approximations are appropriately called *pseudo-Voigt* or *Gauss-Lorentz* functions.

Definition 2.7. *Let $c \in \mathbb{R}$, $w \in \mathbb{R}^+$ and $\lambda \in [0,1]$. Then, we define a* pseudo-Voigt *function $pV_{c,w,\lambda}(x) : \mathbb{R} \to \mathbb{R}$ by*

$$pV_{c,w,\lambda}(x) = \lambda \cdot G_{c,w}(x) + (1-\lambda) \cdot L_{c,w}(x), \tag{2.5}$$

where c and w represent center *and* half-width*, respectively and the* convex combination coefficient *λ represents the deviation from the Lorentzian profile, thereby providing a measure on the amount of interference. Furthermore, the set of all possible pseudo-Voigt functions is denoted by $pVoi := \{pVoi_{c,w,\lambda} \mid c \in \mathbb{R}, w \in \mathbb{R}^+, \lambda \in [0,1]\}$.*

The Gaussians are intentionally defined unconventionally. Note that choosing

$$v = \sqrt{\frac{\ln(2)}{2}} \cdot w$$

results in a Gaussian

$$G_{c,v}(x) = \exp\left(-\frac{(x-c)^2}{2v^2}\right)$$

in the more commonly used form. Fig. 2.1 presents differently parametrized model functions. For example, Gaussians have lighter tails than Lorentzians.

Several functions of similar appearance can be used to model peaks of NMR spectra. Usually, this is done by introducing additional parameters, for example to account for skew [116]. Often the purpose is to generalize the set of possible peak model functions in the hope that more accurate fits to experimental data become possible. In this thesis, we only consider the four functions mentioned above.

Lemma 2.8 (Useful Equations)**.** *There are several equations facilitating the use of these functions.*

1. *For all $c \in \mathbb{R}, w \in \mathbb{R}^+ \lambda \in [0,1]$, we have*

$$L_{c,w}(c) = G_{c,w}(c) = pV_{c,w,\lambda}(c) = 1 \tag{2.6}$$

$$L_{c,w}(c \pm w) = G_{c,w}(c \pm w) = pV_{c,w,\lambda}(c \pm w) = \frac{1}{2}. \tag{2.7}$$

2. *For all* $c \in \mathbb{R}, v, w \in \mathbb{R}^+, \lambda \in [0,1]$ *and* $x \in \mathbb{R}$, *we have*

$$L_{c,w}(x) = L_{0,1}\left(\frac{x-c}{w}\right) \tag{2.8}$$

$$G_{c,w}(x) = G_{0,1}\left(\frac{x-c}{w}\right) \tag{2.9}$$

$$PV_{c,w,\lambda}(x) = pV_{0,1,\lambda}\left(\frac{x-c}{w}\right) \tag{2.10}$$

$$V_{c,v,w}(x) = v \cdot V_{0,1,w/v}\left(\frac{x-c}{v}\right) = w \cdot V_{0,v/w,1}\left(\frac{x-c}{w}\right). \tag{2.11}$$

3. *All* $f \in (Lor \cup Gau \cup pVoi \cup Voi)$ *satisfy* $f(x) > 0$ *for all* $x \in \mathbb{R}$.

4. $(Lor \cup Gau \cup pVoi \cup Voi) \subset L^1(\mathbb{R})$

5. $(Lor \cup Gau \cup pVoi \cup Voi) \subset C(\mathbb{R})$

6. *All* $f \in (Lor \cup Gau \cup pVoi \cup Voi)$ *with* $c = 0$ *are even functions, i.e.,* $f(x) = f(-x)$ *for all* $x \in \mathbb{R}$.

7. *Let* $f \in (Lor \cup Gau \cup pVoi \cup Voi)$. *Then,* f *is strictly monotonically decreasing for all* $x \in [c, \infty)$. *In other words,* f *is unimodal.*[3]

Proof. 1. Simple calculations lead to the desired results.

2. Let c, w as defined above. We then have

$$\begin{aligned} L_{0,1}\left(\frac{x-c}{w}\right) &= \frac{1}{1 + ((x-c)/w)^2} \\ &= \frac{w^2}{w^2 + (x-c)^2} = L_{c,w}(x). \\ G_{0,1}\left(\frac{x-c}{w}\right) &= \exp\left(-\ln(2) \cdot \left(\frac{(x-c)}{w}\right)^2\right) \\ &= G_{c,w}(x). \end{aligned}$$

Eq. (2.10) follows directly for all $\lambda \in [0,1]$ from Eqs. (2.8) and (2.9). To prove the first equation of (2.11), let c, v, w be defined as above. Using the substitution $y = vt$, we have

$$\begin{aligned} v \cdot V_{0,1,w/v}\left(\frac{x-c}{v}\right) &= v \int_{\mathbb{R}} G_{0,1}(t) \cdot L_{0,w/v}\left(\frac{x-c}{v} - t\right) \mathrm{d}t \\ &= \int_{\mathbb{R}} G_{0,1}\left(\frac{y}{v}\right) \cdot L_{0,w/v}\left(\frac{x-c-y}{v}\right) \mathrm{d}y \\ &\overset{(2.8),(2.9)}{=} \int_{\mathbb{R}} G_{0,v}(y) \cdot L_{c,w}(x-y) \mathrm{d}y. \end{aligned}$$

The second equation can be proved analogously.

3. This fact follows from the definition of Lorentzian and Gaussian functions. Pseudo-Voigt functions are a positive convex combination of both, therefore they are also

[3] The fact that the Voigt function is a convolution of two unimodal probability densities does not imply unimodality for the Voigt function itself. In fact, a convolution of two unimodal functions can even have infinitely many modes [112]. Therefore, we need to prove the unimodality *by hand* here.

strictly positive. For Voigt functions we can see that for all $x, y \in \mathbb{R}$ the product $G_{0,v}(y) \cdot L_{c,w}(x-y) > 0$, hence $V_{c,v,w}(x)$ is also positive.[4]

4. We only need to show, that each of the functions is integrable once. Note that the strict positivity of all functions allows us to omit the absolute value.

   Let $c \in \mathbb{R}, w \in \mathbb{R}^+$. First,

$$\begin{aligned}\int_{\mathbb{R}} L_{c,w}(x)\mathrm{d}x &= \int_{\mathbb{R}} L_{0,1}\left(\frac{x-c}{w}\right)\mathrm{d}x \\ &= w \cdot \int_{\mathbb{R}} \frac{1}{1+y^2}\mathrm{d}y = w \cdot \arctan(y)\Big|_{-\infty}^{\infty} = w \cdot \pi < \infty \end{aligned} \tag{2.12}$$

   using the substitution $y := \frac{x-c}{w}$ and Eq. (2.8).

   Second,

$$\begin{aligned}\int_{\mathbb{R}} G_{c,w}(x)\mathrm{d}x &= \int_{\mathbb{R}} \exp\left(-\ln(2)\frac{(x-c)^2}{w^2}\right)\mathrm{d}x \\ &= \frac{w}{\sqrt{\ln(2)}}\int_{\mathbb{R}} \exp(-y^2)\mathrm{d}y = w \cdot \sqrt{\frac{\pi}{\ln(2)}} < \infty \end{aligned} \tag{2.13}$$

   using the substitution $y := \frac{\sqrt{\ln(2)}}{w}(x-c)$, Eq. (2.9) and the well-known Gaussian integral [33, Ch. 4.4].

   Let $\lambda \in [0,1]$. Thirdly,

$$\begin{aligned}\int_{\mathbb{R}} pV_{c,w,\lambda}(x)\mathrm{d}x &= \lambda \int_{\mathbb{R}} G_{c,w}(x)\mathrm{d}x + (1-\lambda)\int_{\mathbb{R}} L_{c,w}(x)\mathrm{d}x \\ &= w \cdot \left(\lambda \cdot \sqrt{\frac{\pi}{\ln(2)}} + (1-\lambda)\cdot\pi\right) < \infty. \end{aligned} \tag{2.14}$$

   Additionally, let $v \in \mathbb{R}^+$.

   Lastly,

$$\begin{aligned}\int_{\mathbb{R}} V_{c,v,w}(x)\mathrm{d}x &= \int_{\mathbb{R}}\int_{\mathbb{R}} G_{0,v}(y)L_{c,w}(x-y)\mathrm{d}y\mathrm{d}x. \\ &= \int_{\mathbb{R}} G_{0,v}(y)\left(\int_{\mathbb{R}} L_{c,w}(x-y)\mathrm{d}x\right)\mathrm{d}y \\ &= \int_{\mathbb{R}} G_{0,v}(y)\cdot w\pi\,\mathrm{d}y = vw \cdot \pi\sqrt{\frac{\pi}{\ln(2)}} < \infty \end{aligned} \tag{2.15}$$

   using Fubini's theorem [33, Thm. 4.4.5.].

5. From their definitions, it is easy to see that $L_{c,w}$, $G_{c,w}$ and $pV_{c,w,\lambda}$ are continuous functions, since $w > 0$. Furthermore, convolutions of continuous functions are continuous again [33, Thm. 9.1.6.]. Therefore, $V_{c,v,w}$ is also continuous for all possible choices of parameters.

6. Again, it is easy to see that for all $x \in \mathbb{R}$, the functions $L_{0,w}(-x) = L_{0,w}(x)$, and $G_{0,w}(-x) = G_{0,w}(x)$, and $pV_{0,w,\lambda}(-x) = pV_{0,w,\lambda(x)}$ are symmetric. For Voigt functions,

[4] Strictly speaking, we also use the fact that $G_{0,v}(y) \cdot L_{c,w}(x-y)$ is continuous (5.).

we have

$$\begin{aligned} V_{0,v,w}(-x) &= \int_{\mathbb{R}} G_{0,v}(y) L_{0,w}(-x-y)\mathrm{d}y \\ &\overset{6.}{=} \int_{\mathbb{R}} G_{0,v}(-y) L_{0,w}(x+y)\mathrm{d}y \\ &= -1 \cdot \int_{\infty}^{-\infty} G_{0,v}(t) L_{0,w}(x-t)\mathrm{d}t \\ &= V_{0,v,w}(x) \end{aligned}$$

using the symmetry of Lorentz and Gauss functions and the substitution $t = -y$.

7. Let $c < x < y$, and $0 < d_1 \coloneqq x - c < d_2 \coloneqq y - c$. Then,

$$\begin{aligned} \frac{L_{c,w}(x)}{L_{c,w}(y)} &= \frac{w^2 + d_2^2}{w^2 + d_1^2} > 1 \\ \frac{G_{c,w}(x)}{G_{c,w}(y)} &= \exp \underbrace{\left(-\ln(2) \left(\frac{d_1^2 - d_2^2}{w^2} \right) \right)}_{>0} > 1. \end{aligned}$$

The monotonicity of the pseudo-Voigt functions again follows from the definition and from the monotonicity of the Gauss and Lorentz functions themselves.

To prove the monotonicity of Voigt functions, we first assume w.l.o.g. that $c = 0$. Let $0 < a < b$. We then need to show that $V_{0,v,w}(a) - V_{0,v,w}(b) > 0$. We define $l(y) \coloneqq G_{0,v}(y) \cdot L_{0,w}(a-y)$ and $r(y) \coloneqq G_{0,v}(y) \cdot L_{0,w}(b-y)$, and obtain

$$V_{0,v,w}(a) - V_{0,v,w}(b) = \int_{\mathbb{R}} l(y) - r(y)\mathrm{d}y. \tag{2.16}$$

We can also see that

$$\begin{aligned} & l(y) - r(y) > 0 && \Leftrightarrow \\ & L_{0,w}(a-y) > L_{0,w}(b-y) && \Leftrightarrow \\ & \quad (b-y)^2 > (a-y)^2 \end{aligned} \tag{2.17}$$

using the monotonicity of the Lorentzian. Therefore, we have

$$l(y) - r(y) \begin{cases} > 0, \text{for } y < \frac{b+a}{2} \\ < 0, \text{for } y > \frac{b+a}{2} \end{cases} .$$

This fact and the transformation $y = -y' + (b+a)$ allow us to rewrite Eq. (2.16) as follows

$$\begin{aligned} \int_{\mathbb{R}} l(y) - r(y)\mathrm{d}y &= \int_{-\infty}^{\frac{b+a}{2}} l(y) - r(y)\mathrm{d}y + \int_{\frac{b+a}{2}}^{\infty} l(y) - r(y)\mathrm{d}y \\ &= \int_{\frac{b+a}{2}}^{\infty} l(-y'+b+a) - r(-y'+b+a)\mathrm{d}y' + \int_{\frac{b+a}{2}}^{\infty} l(y) - r(y)\mathrm{d}y \\ &= \int_{\frac{b+a}{2}}^{\infty} l(-y+b+a) - r(-y+b+a) + l(y) - r(y)\mathrm{d}y. \end{aligned}$$

Note, that $L_{0,w}(a+y-b-a) = L_{0,w}(b-y), L_{0,w}(b+y-b-a) = L_{0,w}(a-y)$ and therefore

$$l(-y+b+a) - r(-y+b+a) + l(y) - r(y) = \\ [G_{0,v}(-y+b+a) - G_{0,v}(y)] \cdot [L_{0,w}(b-y) - L_{0,w}(a-y)].$$

If both of these factors are positive for $y > \frac{b+a}{2}$, then the proof is complete. For the first factor, we have

$$\begin{aligned} \frac{b+a}{2} < y \quad &\Rightarrow \\ -2(b+a)y + (b+a)^2 < 0 \quad &\Rightarrow \\ 0 < (-y+b+a)^2 < y^2 \quad &\Rightarrow \\ \exp\left(-\ln(2)\frac{(-y+b+a)^2}{v^2}\right) > \exp\left(-\ln(2)\frac{y^2}{v^2}\right). \end{aligned}$$

Using Ineq. (2.17) ff., we can see that the second factor is positive as well. Therefore, both factors are positive for all $y > \frac{b+a}{2}$. Then, the integral over all $y \geq \frac{b+a}{2}$ must also be strictly positive. This is equivalent to the initial inequality.

□

Using the second fact from Lem. 2.8, we can see that an additional center parameter does not give an additional degree of freedom for the functions in *Voi*. Let $c, d \in \mathbb{R}, v, w \in \mathbb{R}^+$. Then

$$\begin{aligned} (G_{d,v} * L_{c,w})(x) &= \int_{\mathbb{R}} G_{c,v}(y) \cdot L_{d,w}(x-y)\mathrm{d}y \\ &= \int_{\mathbb{R}} G_{0,v}(y-d) \cdot L_{c,w}(x-y)\mathrm{d}y \\ &= \int_{\mathbb{R}} G_{0,v}(y') \cdot L_{c,w}(x-(y'+d))\mathrm{d}y' \\ &= \int_{\mathbb{R}} G_{0,v}(y') \cdot L_{c-d,w}(x-y')\mathrm{d}y' \\ &= V_{(c-d),v,w}(x). \end{aligned}$$

Remark 2.9. The peak functions are intentionally parametrized in a way that allows for the parameters to carry meaning about the spectral properties. The model functions are normed to one, thereby allowing the introduction of another parameter h, which is multiplied to the funtion and denotes the peak height[5]. The parameter c represents the position where the function reaches its maximum. The height parameter h represents this maximum value. The parameter w represents $\frac{1}{2}$ of the width of the function at half height. In the case of pseudo-Voigt functions, the convex combination coefficient λ cannot easily be obtained by looking at the spectra. Still, λ approaching 1 leads to lighter tails, while λ approaching 0 leads to heavier tails of the model function. Fig. 2.1 illustrates the meaning of the four parameters of pseudo-Voigt functions. When it comes to Voigt functions, only the parameters c possesses the same meaning compared to pseudo-Voigt functions. Both v and w contribute to the width of the function via their ratio $\frac{v}{w}$ and to its height, as per Eq. (2.11).

Additionally, the properties of these peak functions allows for derivative-based parameter detection.

[5] In Ch. 3, the height parameter is omitted since linear independence is observed. The coefficients of the linear combinations are denoted by α_i to avoid confusion with the optimizable height parameter h.

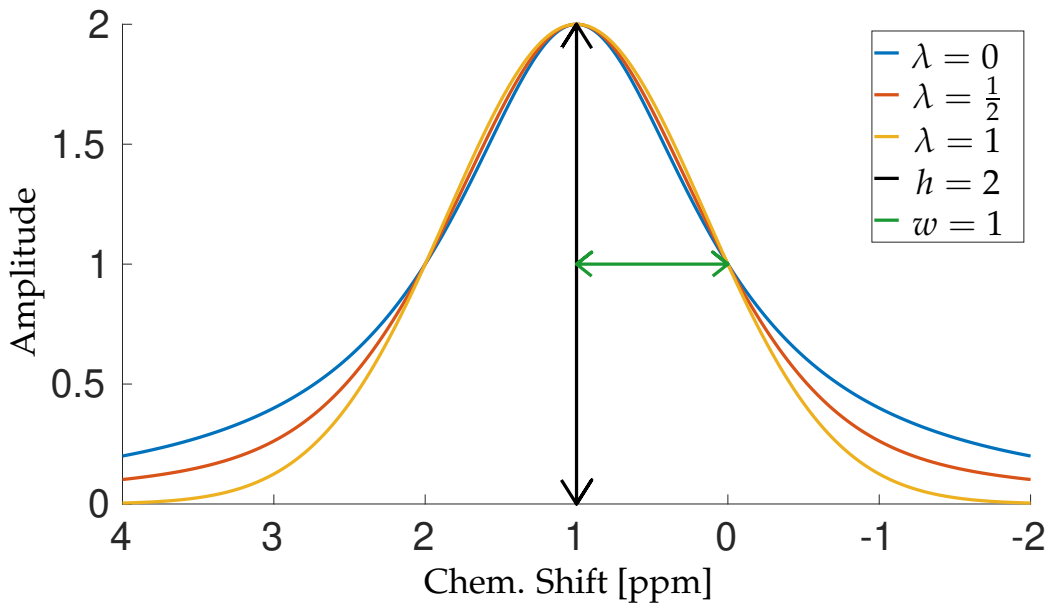

FIGURE 2.1: Three pseudo-Voigt functions with different values for λ. Center and height (black line) and half-width (green line) are identical.

Lemma 2.10 (Derivatives and Their Implications). *Let $c \in \mathbb{R}, w \in \mathbb{R}_+, \lambda \in [0,1]$. Then, all functions $L''_{c,w}, G''_{c,w}, pV''_{c,w,\lambda}$ possess local minima at $x = c$ and these local minima are sharper than the original peaks, i.e., their half-width is $< w$.*

Proof. Since $f_{c,w}(x) = f_{0,1}\left(\frac{x-c}{w}\right)$ for $f \in (Lor \cup Gau \cup pVoi)$, we may consider the derivatives $f^{(k)}_{0,1}(0)$ w.l.o.g.

For Lorentzians, we have

$$\begin{aligned}
L'_{0,1}(x) &= \frac{-2x}{(1+x^2)^2} \\
L''_{0,1}(x) &= \frac{8x^2 - 2(1+x^2)}{(1+x^2)^3} \\
L'''_{0,1}(x) &= \frac{24x(1-x^2)}{(1+x^2)^4} \\
L^{(iv)}_{0,1}(x) &= \frac{24(1-3x^2)(1+x^2) - 192x^2(1-x^2)}{(1+x^2)^5}.
\end{aligned}$$

In particular, we have $L'''_{0,1}(0) = 0$ and $L^{(iv)}_{0,1}(0) = 24 > 0$, thus proving the local minimum at $L''_{0,1}(0)$.

Furthermore, $L''_{0,1}(x)$ has two additional roots, $\pm\frac{1}{\sqrt{3}}$. This implies that the half-width of the *negative peak* is strictly less than $\frac{1}{\sqrt{3}}$, which is smaller than the original half-width of 1.

For Gaussians, let $\alpha = 2\ln(2)$. Then, we have

$$\begin{aligned} G'_{0,1}(x) &= G_{0,1}(x)\cdot(-\alpha x) \\ G''_{0,1}(x) &= G_{0,1}(x)\cdot(\alpha^2x^2-\alpha) \\ G'''_{0,1}(x) &= G_{0,1}(x)\cdot(-\alpha^3x^3+3\alpha^2x) \\ G^{(iv)}_{0,1}(x) &= G_{0,1}(x)\cdot(\alpha^4x^4-6\alpha^3x^2+3\alpha^2). \end{aligned}$$

Again, we see that $G'''_{0,1}(0) = 0$ and $G^{(iv)}_{0,1} = 3\alpha^2 > 0$, thus proving the local minimum at $G''_{0,1}(0)$. Furthermore, $G''_{0,1}$ has two additional roots, $\pm\frac{1}{\sqrt{\alpha}}$. This implies that the half-width of the *negative peak* is strictly less than $\frac{1}{\sqrt{\alpha}}$, which is less than 1, the original half-width.

Using the linearity of the derivative, we know that $pV''_{0,1,\lambda}$ has a local minimum at $x = 0$. To estimate the half-width, we look at the roots of

$$pV''_{0,1,\lambda}(x) = \lambda G''_{0,1}(x) + (1-\lambda)L''_{0,1}(x).$$

We know that $pV''_{0,1,\lambda}\left(\frac{1}{\sqrt{3}}\right) \leq 0$ and that $pV''_{0,1,\lambda}\left(\frac{1}{\sqrt{\alpha}}\right) \geq 0$. The intermediate value theorem [13, Thm. 5.3.7] implies that there exists a root in the interval $\left[\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{\alpha}}\right]$. Therefore, the half-width of this negative peak is also strictly less than 1 and the peak is sharper. □

The fact that the second, and more generally $2k$-th, derivatives produce sharper peaks can be used to create more robust peak detection methods. In Sec. 4.2.2, we describe one such method. There, Fig. 4.8 illustrates the advantage of using numerical second derivatives compared to peak detection using only the data vector.

2.3 Objective Functions $d(\cdot,\cdot)$

Regarding the optimization problem described in Def. 2.3, we mainly implemented using the Frobenius norm $d(D, M_P) = \|D - M_P\|_F^2$, or the sum of squares error, as an objective function. Using the sum of squares has several advantages. This metric is well-known and minimization of the sum of squares is well-understood. For a good introduction on nonlinear least squares, we refer to [64]. Furthermore, since the algorithms of this thesis are implemented in `Matlab`, we can make use of the optimization algorithms specifically designed to minimize the Frobenius norm. Mainly, we use `lsqnonlin` for optimization purposes, which exploits the structure of the sum of squares function to approximate some of the more complex computations [55].

The optimization problem then simplifies to finding $m \in \mathbb{N}$ and $\mathcal{P}$ such that

$$d(D, M_P) = \|D - M_P\|_F^2 = \sum_{x\in X}\sum_{t\in T}(D(x,t) - M_P(x,t))^2 \tag{2.18}$$

is minimal.

Another objective function that can be used is the *Wasserstein distance*. Originally, this metric was defined as a metric between two probability measures [30, 127]. We only need the special case of absolutely continuous measures on $\mathbb{R}$, in which case a simpler definition can be found in [123].

The Wasserstein metric can be viewed as the mean distance each grain of sand has to be moved, if both probability distributions are assumed to represent piles of sand. Therefore, it is used in the context of optimal transport. For a detailed overview on the Wasserstein metric in that context, we refer to [124].

Definition 2.11. *Let $f, g : \mathbb{R} \to \mathbb{R}$ be two probability density functions and let $F, G : \mathbb{R} \to [0,1]$ be their corresponding cumulative distribution functions. Then, the (one-dimensional)* Wasserstein metric, *or* Wasserstein distance, *$d_W(f,g)$ can be written as*

$$d_W(f,g) = \int_{\mathbb{R}} |F(x) - G(x)| \mathrm{d}x. \tag{2.19}$$

Now, an NMR spectrum s consisting of a finite number of peaks modeled by Eqs. (2.2) - (2.5) are also integrable functions. The function $s \cdot \|s\|_1^{-1}$ is therefore a probability density. This allows us to apply the Wasserstein metric to normed NMR spectra as well.

For spectra s, t where $X = x_1 < \ldots < x_N$ is finite, we can compute the Wasserstein distance in linear time.

Definition 2.12. *Let $s, t \in \mathbb{R}^N$, with $s(i), t(i) \geq 0$ for $i = 1, \ldots, N$. Let c_s, c_t be the normed cumulative sums of s and t, where for some vector $v \in \mathbb{R}^N$ the cumulative sum c_v of v is defined by $c_v(n) = \sum_{i=1}^{n} v(i) \cdot \|v\|_1^{-1}$. Then, the Wasserstein metric in the discrete case can be written as*

$$d_W(s,t) = \sum_{j=1}^{N-1} |c_s(j) - c_t(j)| \cdot (x_{j+1} - x_j).$$

The Wasserstein metric has some interesting properties regarding the optimization of NMR spectra. Its properties allow us to separate the optimization into linear and nonlinear parameters. This is a direct result of Lem. 2.14. First, we quickly define quantile functions.

Definition 2.13. *Let $F : R \to [0,1]$ be a cumulative distribution function. Then, its* quantile function *$F^{-1} : [0,1] \to \mathbb{R}$ is defined by*

$$F^{-1}(z) = \inf\{x \in \mathbb{R} \mid z \leq F(x)\}.$$

Lemma 2.14 (Vallender, [123])**.** *Let f, g, F, G be defined as in Def. 2.11 and let $F^{-1}(z), G^{-1}(z)$ denote the quantile functions of F and G respectively. Then, the Wasserstein distance can alternatively be represented by*

$$d_W(f,g) = \int_0^1 |F^{-1}(z) - G^{-1}(z)| \mathrm{d}z. \tag{2.20}$$

Proof. In [123, p. 785], this fact follows from "*obvious geometric considerations*". To facilitate the understanding of this metric, we recapitulate these considerations here.

Let λ_2 denote the two-dimensional Lebesgue measure. Then, the Integral from Eq. (2.20) is equal to the area in between the graphs of the quantile functions. This allows us to write

$$\int_0^1 |F^{-1}(z) - G^{-1}(z)| \mathrm{d}z. = \lambda_2(A_q),$$

where

$$A_q \coloneqq \left\{(x,z) \in \mathbb{R} \times [0,1] \mid \min\left\{F^{-1}(z), G^{-1}(z)\right\} \leq x \leq \max\left\{F^{-1}(z), G^{-1}(z)\right\}\right\}.$$

Analogously, we may write

$$\int_{\mathbb{R}} |F(x) - G(x)| \mathrm{d}x = \lambda_2(A_p),$$

where

$$A_p := \{(x,z) \in \mathbb{R} \times [0,1] \mid \min\{F(x), G(x)\} \leq z \leq \max\{F(z), G(z)\}\}.$$

Furthermore, we define the two λ_2- null sets

$$N_q := \left\{(x,z), \mid x = \max\left\{F^{-1}(z), G^{-1}(z)\right\}\right\},$$
$$N_p := \{(x,z), \mid z = \min\{F(x), G(x)\}\}.$$

Now, for any point $(x,z) \in A_q \setminus N_q$, we can assume that $F^{-1}(z) \leq x < G^{-1}(z)$ w.l.o.g. Using the defining inequalities of the quantile functions $F(x) \geq z \Leftrightarrow F^{-1}(z) \leq x$, we can infer that $F(x) \geq z > G(x)$. Therefore, $A_q \setminus N_q \subseteq A_p$.

Conversely, for any point $(x,z) \in A_p \setminus N_p$, we again assume $F(x) < z \leq G(x)$ w.l.o.g. Again, using the definition of quantile functions, we obtain $G^{-1}(z) \leq x < F^{-1}(z)$ and $A_p \setminus N_p \subseteq A_q$.

Finally, we have

$$\lambda_2(A_q) = \lambda_2(A_q \setminus N_q) \leq \lambda_2(A_p) = d_W(f,g)$$
$$d_W(f,g) = \lambda_2(A_p) = \lambda_2(A_p \setminus N_p) \leq \lambda_2(A_q),$$

completing the proof. □

The Wasserstein distance is sensitive to changes in nonlinear parameters such as center and half-width, when applied to NMR spectra.

Remark 2.15. For any NMR spectrum s, and some parameters $c \in \mathbb{R}, h \in \mathbb{R}_{\geq 0}, w \in \mathbb{R}_{>0}$, changes in position $s(x-c)$ or half-width $s(x/w)$ change the Wasserstein metric (almost) linearly, while changes in height $h \cdot s(x)$ are not detected. More specifically, we have

$$d_W(s(x), s(x-c)) = |c| \tag{2.21}$$
$$d_W(s(x), s(x/w) = |1-w| \cdot \mathbb{E}(|S|) \tag{2.22}$$
$$d_W(s(x), h \cdot s(x)) = 0, \tag{2.23}$$

where S represents a random variable with the probability density function $s \cdot \|s(x)\|_1^{-1}$.

Proof. Since the Wasserstein metric applies only to normed spectra, we can see that for Eq. (2.23), after norming, we have

$$d_W(s(x) \cdot \|s\|_1^{-1}, h \cdot s(x) \cdot \|h \cdot s\|_1^{-1}) = d_W(s(x) \cdot \|s\|_1^{-1}, s(x) \cdot \|s\|_1^{-1}) = 0.$$

We further assume that $\|s\|_1 = 1$ w.l.o.g. Let $F(x) = \int_{-\infty}^{x} s(y)\mathrm{d}y$ and $G_1(x) = \int_{-\infty}^{x} s(y-c)\mathrm{d}y$ be the cumulative distribution functions of the two spectra from Eq. (2.21). We can easily see that $G_1(x) = F(x-c)$. This implies that $G_1^{-1}(z) = F^{-1}(z) + c$. Using Lem. 2.14, we obtain

$$d_W(s(x), s(x-c)) = \int_0^1 |F^{-1}(z) - G_1^{-1}(z)|\mathrm{d}z = \int_0^1 |c|\mathrm{d}z = |c|.$$

To prove Eq. (2.22), first note that $\frac{1}{w}s\left(\frac{x}{w}\right)$ again has norm one. Let $G_2(x) = \int_{-\infty}^{x} \frac{1}{w}s\left(\frac{y}{w}\right)\mathrm{d}y$ be the corresponding cumulative distribution function. Using the substitution $u = y/w$ we again are able to derive $F(x/w) = G_2(x)$ and $G_2^{-1}(z) = w \cdot F^{-1}(z)$. Using Lem. 2.14, we

obtain

$$d_W(s(x), s(x/w)) = \int_0^1 |F^{-1}(z) - G_2^{-1}(z)|\mathrm{d}z = |1-w| \int_0^1 |F^{-1}(z)|\mathrm{d}z.$$

Finally, we need to show that for real-valued random variables X with density functions f and quantile functions F^{-1}, the equality $\int_0^1 |F^{-1}(z)|\mathrm{d}z = \mathbb{E}(|X|)$ holds. Now, we have

$$\begin{aligned}\mathbb{E}(|X|) &= \int_{\mathbb{R}} |x| \cdot f(x)\mathrm{d}x \\ &= \int_{\mathrm{supp}(f)} |x| \cdot f(x)\mathrm{d}x \\ &= \int_0^1 |F^{-1}(z)|\mathrm{d}z.\end{aligned}$$

where $\mathrm{supp}(f) = \{x \in \mathbb{R} \mid f(x) > 0\}$. For the last equation we substituted $z = F(x)$. Note that the image $F(\mathrm{supp}(f)) = [0,1]$ and that $x = F^{-1}(z)$ for all $x \in \mathrm{supp}(f)$. □

To use the Wasserstein metric as an objective function for parameter optimization requires us to split the optimization into two parts. This has the advantage of reducing the number of parameters during optimization, since only the centers c_i and the half-widths w_i need to be optimized. Finding the linear heights h_i and the convex combination parameters λ_i is trivial, as optimal linear parameters (w.r.t. the Frobenius norm) can be computed directly using linear least squares. Details on how exactly we implement the Wasserstein objective function can be found in Ch. 4. In general, we compute the Wasserstein metric on each spectrum and thus minimize

$$d(D, M_p) = \sum_{t \in T} d_W(D(:,t), M_P(:,t)). \tag{2.24}$$

Chapter 3

Uniqueness and Ambiguity of Solutions

The question whether the hard models of NMR spectra are unique or ambiguous has not yet been discussed prior to [52]. However, in similar fields of spectroscopy, ambiguous solutions play a crucial role. Whenever nonnegative matrix factorization (NMF) methods can be applied, the so-called *rotational* ambiguity arises [2, 115]. Since in NMR spectroscopy, the LAMBERT-BEER law does not apply, one might assume that ambiguity is even worse for NMR data. However, the ambiguity of the NMF methods arises because there, the spectral shapes are arbitrary. As we discuss in Sec. 3.1, the physical model and the properties of the model functions limit the set of possible NMR spectra. Under certain assumptions regarding selectivity, this leads to unique spectral decompositions. In the literature, it is usually assumed that modeling NMR data has no unique solution. Indeed, there are some types of ambiguity that are preserved even from a rigorous mathematical point of view. For example, multiplets with sub-peak distances equal to the distance from each other can form shapes identical to another multiplet, see [80, Ch. 8]. Another type of chemical ambiguity is caused by the fact that some protons behave identically due to symmetry in the chemical structure, see [80, Ch. 4] and [9, 92]. Aside from these types of ambiguity, it is usually assumed that if peaks are in close proximity, there is no unique model. Usually, this type of ambiguity is not clearly stated, but can for example be found in [21, p. 14f.]. This type of ambiguity is typically mitigated through different types of NMR experiments, see [74], in hopes of increasing the selectivity of the new data.

Similar methods to those used in this chapter can be found regarding the *HRT* conjecture, see [28] and Sec. 3.1.6, in functional analysis [25] or in wavelet theory [27].

In this chapter, we aim to answer the following questions:

1. Is the solution to the modeling problem under ideal conditions unique?
2. Is the solution unique, if the spectrum is measured on a discrete set of points X?
3. If not, can certain requirements be met to ensure uniqueness?
4. If noise is present, are all solutions relatively close in parameter space?

Parts of this chapter have been published in [52]. However, this thesis contains refurbished theorems, more elegant proofs, and discusses the question of uniqueness in greater detail. Lastly, to avoid confusion between the summation index i and the complex constant $\mathbf{i}$, the latter is denoted in **bold font**.

3.1 The Continuous Problem

To get a hold of the first question, we must first define what *ideal conditions* actually are. In the case of this chapter, we assume that an NMR spectrum consisting of m different peaks is

given by $s : \mathbb{R} \to \mathbb{R}$, where $s = \sum_{i=1}^{m} \alpha_i f_{p_i}$. We also assume that all model functions f are of the same type, e.g., all Lorentzian or pseudo-Voigt functions and that there is no noise. This definition already implies the existence of a solution to the modeling problem, namely the sum of m model functions $\alpha_i \cdot f_{p_i}$.

An ambiguous solution to this problem would imply that there is a possibly different number of model functions n with different parameters β_j, q_j, such that the sum is identical to s and we have

$$s = \sum_{i=1}^{m} \alpha_i f_{p_i} = \sum_{j=1}^{n} \beta_j f_{q_j}. \tag{3.1}$$

There are two trivial ways to achieve such ambiguous solutions. First, we can add zeros to the sums. Since f_{p_i} are usually non-zero for all choices of parameters, we can prevent this by requiring non-zero heights $\alpha_i \neq 0, \beta_j \neq 0$.

The second way is to split up individual summands. Let $t_1, \ldots, t_k \in \mathbb{R} \setminus \{0\}$ such that $\sum_{l=1}^{k} t_l = 1$. Then, we can choose any $i_0 \leq m$ and substitute

$$\alpha_{i_0} f_{p_{i_0}} = \alpha_{i_0} \cdot \left(\sum_{l=1}^{k} t_l \right) f_{p_{i_0}}, \tag{3.2}$$

creating a trivial ambiguity. Conversely, if for some index set $I = \{i_1, \ldots, i_k\}$ with $k > 1$, the corresponding parameter set $\{p_i \mid i \in I\}$ has only one element, we may use Eq. (3.2) to reduce the number of summands in the definition of s and obtain

$$s = \sum_{i=1}^{m} \alpha_i f_{p_i} = \Lambda f_{p_I} + \sum_{\substack{i=1 \\ i \notin I}}^{m} \alpha_i f_{p_i}, \tag{3.3}$$

where p_I is the single parameter tuple present for indices $i \in I$ and $\Lambda = \sum_{i \in I} \alpha_i$. Now, if we reduce the number of summands for every parameter tuple that appears multiple times, we obtain a spectrum s, where the parameter tuples are pairwise distinct.

Definition 3.1. *Let $\mathcal{F} \in \{Lor, Gau, Voi, pVoi\}$ and, for all $i = 1, \ldots, m$ let $f_{p_i} \in \mathcal{F}$. Let s be an NMR spectrum as per Def. 2.1. Then, a set of parameters $\{(\alpha_i, p_i) \mid i = 1, \ldots, m\}$ is called a* decomposition *of s, if*

$$s = \sum_{i=1}^{m} \alpha_i f_{p_i}$$

holds.

If, for $i \neq j$, we have $p_i \neq p_j$, and if, for all $i = 1, \ldots, m$, we have $\alpha_i \neq 0$, then the decomposition is called minimal.

Remark 3.2. Every spectrum has at least one minimal decomposition. If s is defined as in Eq. (2.1), then omitting all parameters where $\alpha_i = 0$, and simplifying identical parameter tuples p_i as described in Eq. (3.3), leads to a minimal decomposition of s.

Now, ambiguous solutions to the modeling problem can be defined as different minimal decompositions of the same spectrum s, i.e., as different sets of parameters

$$\{(\alpha_i, p_i) \mid i = 1, \ldots, m\} \neq \{(\beta_j, q_j) \mid j = 1, \ldots, n\}.$$

However, we are not only interested in whether a specific spectrum has ambiguous decompositions. Rather, we are interested in whether all spectra composed of functions of type

$\mathcal{F}$ are uniquely decomposable. First, note that, by Rem. 3.2, if two non-minimal, different decompositions of s exist, then there are two cases. Either, both decompositions can be reduced to the same minimal decomposition, or two different minimal decompositions exist. Therefore, examining only minimal decompositions is sufficient.

Definition 3.3. *An NMR spectrum s with a minimal decomposition $\{(\alpha_i, p_i) \mid i = 1, \ldots, m\}$ has* ambiguous decompositions*, or is* ambiguous*, if there exists another minimal decomposition $\{(\beta_j, q_j) \mid j = 1, \ldots, n\}$, where additionally, for all $i \leq n$ and $j \leq m$, the parameter tuples $p_i \neq q_j$ are different. Otherwise, we call the single decomposition of s* unique*.*

Without loss of generality, we require the tuples of both decompositions to be different. If there are equal parameter tuples $p_{i_0} = q_{j_0}$, we can simplify Eq. (3.1) to

$$\begin{aligned} s = \sum_{i=1}^{m} \alpha_i f_{p_i} &= \sum_{j=1}^{n} \beta_j f_{q_j} \\ \sum_{\substack{i=1 \\ i \neq i_0}}^{m} \alpha_i f_{p_i} &= \sum_{\substack{j=1 \\ j \neq j_0}}^{n} \beta_j f_{q_j} + (\beta_{j_0} - \alpha_{j_0}) f_{q_{j_0}}, \end{aligned}$$

thereby eliminating one summand on the left and possibly one summand on the right.

Lemma 3.4. *Let $\mathcal{F} \in \{Lor, Gau, Voi, pVoi\}$. Then, all spectra s composed of finitely many functions $f_{p_i} \in \mathcal{F}$ have unique decompositions if and only if, for all $m \in \mathbb{N}$ and pairwise distinct parameters $p_1, \ldots, p_m$, the set $\{f_{p_1}, \ldots, f_{p_m}\} \subseteq \mathcal{F}$ is linearly independent.*

Proof. "⇒": Assume that for some $m \in \mathbb{N}$ there exist $f_{p_1}, \ldots, f_{p_m}$ with distinct p_i and nonzero $\alpha_1, \ldots, \alpha_m$ such that

$$\alpha_1 f_{p_1} = \sum_{i=2}^{m} \alpha_i f_{p_i}. \tag{3.4}$$

Note that $m > 1$, since $0 \notin \mathcal{F}$. We can view Eq. (3.4) as two different minimal decompositions of the spectrum $s = \alpha_1 f_{p_1}$, thereby showing the existence of an ambiguously decomposable spectrum s.

"⇐": Now, assume that a spectrum s with two minimal ambiguous decompositions

$$s = \sum_{i=1}^{m} \alpha_i f_{p_i} = \sum_{j=1}^{n} \beta_j f_{q_j}$$

exists. This implies

$$0 = \sum_{k=1}^{m+n} \beta_j f_{q_j},$$

with $\beta_{n+i} = -\alpha_i$ nd $q_{n+i} = p_i$ for all $i = 1, \ldots, m$. Given the assumption that all parameters are distinct, the set $\{f_{q_1}, \ldots, f_{q_{m+n}}\}$ is linearly dependent.

□

Using this Lemma, it becomes apparent that for a given function class $\mathcal{F}$, either all spectra are ambiguous or none are.

Before we examine all four types of model functions, we first need to introduce the Fourier transform.

3.1.1 The Fourier Transform

Definition 3.5. *Let $f \in L^1(\mathbb{R})$. Then, the continuous Fourier transform $\hat{f} : \mathbb{R} \to \mathbb{C}$ is defined by*

$$\hat{f}(\omega) = (\mathcal{F}f)(\omega) = \int_{\mathbb{R}} f(x) e^{-\mathbf{i}x\omega} \mathrm{d}x \tag{3.5}$$

This definition and subsequent lemmas can be found in [98, Ch. 2]. Sometimes, the Fourier transform is defined with an additional factor $\frac{1}{\sqrt{2\pi}}$ attached. Since we are only interested in linear independence, omitting this factor simplifies the subsequent equations.

Why do we study Fourier transforms in this context? According to [98, Thm. 2.5], the Fourier transform is a linear operator. If some functions $f_1, \ldots, f_m$ form a nontrivial linear combination $0 = \sum_{i=1}^{m} \alpha_i f_i$, then their Fourier transformed counterparts $\widehat{f_1}, \ldots, \widehat{f_m}$ also fulfill $\hat{0} = 0 = \sum_{i=1}^{m} \alpha_i \hat{f}_i$. This is equivalent to the fact that linearly independent functions in Fourier space also imply linearly independent functions in the preimage of the Fourier transform. Another argument is that our data sets are the result of Fourier transforming the measured FID data. Applying the Fourier transform again yields functions with useful properties, such as exponential decay rates, facilitating the proof of linear independence. The following lemmas provide formulas for Fourier transforming the four model functions.

Lemma 3.6. *For a Lorentzian $L_{c,w}$ with parameters $c, w \in \mathbb{R}$ and $w > 0$, we have*

$$\widehat{L_{c,w}}(\omega) = C(w) \cdot e^{-\mathbf{i}c\omega} \cdot e^{-w|\omega|}, \tag{3.6}$$

where $C(w)$ is a constant depending only on w.

The Fourier transforms of commonly used functions in signal theory, e.g., Lorentzians and Gaussians, are well-known. For the sake of completeness, the formulas and proofs are presented here as well.

Proof. We obtain

$$\begin{aligned}
\widehat{L_{c,w}}(\omega) &= \int_{\mathbb{R}} \frac{w^2}{w^2 + (x-c)^2} \cdot \mathrm{e}^{-\mathbf{i}x\omega} \mathrm{d}x \\
&= w \cdot \int_{\mathbb{R}} \frac{1}{1+y^2} \mathrm{e}^{-\mathbf{i}(yw+c)\omega} \mathrm{d}y \\
&= w \cdot \mathrm{e}^{-\mathbf{i}c\omega} \cdot \int_{\mathbb{R}} \frac{1}{1+y^2} \mathrm{e}^{-\mathbf{i}(yw)\omega} \mathrm{d}y,
\end{aligned}$$

where $y = \frac{x-c}{w}$. The remaining integral term is well-known and proven in App. A. Using this Lem. A.1, we see that

$$\widehat{L_{c,w}}(\omega) = w\mathrm{e}^{-\mathbf{i}c\omega} \cdot \pi \mathrm{e}^{-w|\omega|}.$$

Defining $C(w) := \pi w$ completes the proof. □

Lemma 3.7. *For a Gaussian $G_{c,w}$ with parameters $c, w \in \mathbb{R}$ and $w > 0$, we have*

$$\widehat{G_{c,w}}(\omega) = C(w') \cdot e^{-\mathbf{i}c\omega} \cdot e^{-w'^2\omega^2}, \tag{3.7}$$

where $w' = \frac{w}{2\sqrt{\ln(2)}}$ and $C(w')$ is depends only on w'.

Note, that when we are interested in statements for all Gaussians, the distinction between w and w' does not matter.

Proof. Again, we obtain

$$\begin{aligned}\widehat{G_{c,w}}(\omega) &= \int_{\mathbb{R}} \exp\left(-\ln(2)\cdot\frac{(x-c)^2}{w^2} - \mathbf{i}x\omega\right)\mathrm{d}x \\ &= \frac{w}{\sqrt{\ln(2)}}\mathrm{e}^{-\mathbf{i}c\omega}\int_{\mathbb{R}} \exp\left(-y^2 - \mathbf{i}y\cdot 2w'\omega\right)\mathrm{d}y,\end{aligned}$$

where w' is defined above and $y = \frac{\sqrt{\ln(2)}}{w}(x-c)$. Completing the square yields

$$\widehat{G_{c,w}}(\omega) = \frac{w}{\sqrt{\ln(2)}}\mathrm{e}^{-\mathbf{i}c\omega}\cdot\mathrm{e}^{-(w'\omega)^2}\int_{\mathbb{R}} \exp\left(-(y+\mathbf{i}w'\omega)^2\right)\mathrm{d}y.$$

The remaining integral can be seen as one value of a holomorphic function $g : \mathbb{C} \to \mathbb{C}$, where

$$g(\alpha) = \int_{\mathbb{R}} \exp\left(-(y+\alpha)^2\right)\mathrm{d}y.$$

For all $\alpha \in \mathbb{R}$, we have $g(\alpha) = \sqrt{\pi}$, using the well-known integral of the density function of a normal distribution [33, Ch. 4.4]. According to the identity theorem for holomorphic functions [3, Thm. 3.2.6], the function g must equal the constant $\sqrt{\pi}$ on the entire domain. Therefore, the integral in question also equals $\sqrt{\pi}$.

Finally, we have

$$\widehat{G_{c,w}}(\omega) = 2w'\mathrm{e}^{-\mathbf{i}c\omega}\cdot\mathrm{e}^{-(w'\omega)^2}\sqrt{\pi}.$$

Now, we define $C(w') \coloneqq 2w'\sqrt{\pi}$ and obtain Eq. (3.7). □

The linearity of the Fourier transform allows us to easily obtain the formula for pseudo-Voigt functions.

Corollary 3.8. Let $c, w \in \mathbb{R}, \lambda \in [0,1]$ and $w > 0$. Then,

$$\widehat{pV_{c,w}}(\omega) = \mathrm{e}^{-\mathbf{i}c\omega}\left((1-\lambda)C_L(w)\mathrm{e}^{-w|\omega|} + \lambda C_G(w')\mathrm{e}^{-(w'\omega)^2}\right),$$

where $C_L(w)$ and $C_G(w)$ are the constants from the Lorentzian and Gaussian formulas, respectively.

As it turns out, we can prove a more general statement that does not require using this formula. The definition of pseudo-Voigt functions as a linear combination of Lorentzians and Gaussians also allows for an additional type of trivial ambiguity, which we discuss in Sec. 3.1.4.

Lastly, we use the convolution property of the Fourier transform [98, Thm. 2.15] to easily obtain the formula for Voigt functions.

Corollary 3.9. Let $c, v, w \in \mathbb{R}$, and $v, w > 0$. Then,

$$\widehat{V_{c,v,w}}(\omega) = C(v', w)\cdot\mathrm{e}^{-\mathbf{i}c\omega}\cdot\mathrm{e}^{-w|\omega|}\cdot\mathrm{e}^{-(v'\omega)^2},$$

where $v' = \frac{v}{2\sqrt{\ln 2}}$ and $C(v', w)$ is a constant dependent only on the parameters of $V_{c,v,w}$.

Finally, we take a look at each type of model function to determine if their corresponding spectra are uniquely decomposable.

3.1.2 Lorentz Functions

Theorem 3.10. *Let $m \in \mathbb{N}$ and let $\{(c_i, w_i) \mid i = 1, \ldots, m,\ c_i, w_i \in \mathbb{R},\ w_i > 0\}$ be a set of pairwise distinct parameter tuples. Then, the set of Lorentzians $\{L_{c_i,w_i} \mid i = 1, \ldots, m\}$ is linearly independent. In other words: Any finite number of different Lorentzians are linearly independent.*

We can prove Thm. 3.10 in two different ways. The first proof is direct and only requires basic algebra. It uses the fact that Lorentzians are rational functions, so equations containing Lorentzians can easily be transformed into polynomial equations.

The second method of proof is much more similar to those used in the following subsections. It exploits the linearity of the Fourier transform and the fact that Fourier transformed functions exhibit different asymptotic behavior.

Direct proof. Assume that $m \in \mathbb{N}$ different Lorentzians are given. We want to show that

$$0 = \sum_{i=1}^{m} \alpha_i \frac{w_i^2}{w_i^2 + (x - c_i)^2} \tag{3.8}$$

implies $\alpha_i = 0$ for all $i = 1, \ldots, m$. Multiplying Eq. (3.8) by all denominators yields

$$0 = \sum_{i=1}^{m} \alpha_i w_i^2 \left(\prod_{\substack{j=1 \\ j \neq i}}^{m} w_j^2 + (x - c_j)^2 \right). \tag{3.9}$$

This allows us to define polynomials

$$g_i(x) \coloneqq \prod_{\substack{j=1 \\ j \neq i}}^{m} w_j^2 + (x - c_j)^2$$

of degree $2m - 2$. Each of these non-zero polynomials has exactly $2m - 2$ roots. The set of roots for the i-th polynomial is equal to $R_i \coloneqq \{c_k \pm w_k \cdot \mathbf{i} \mid 1 \leq k \leq m, k \neq i\}$. Every root has multiplicity one, due to the distinct tuples $(c_i, w_i) \neq (c_j, w_j)$. Lastly, note that for all $i = 1, \ldots, m$ we have $z_i = c_i + w_i \mathbf{i} \notin R_i$ and therefore $g_i(z_i) \neq 0$.

Now, from Eq. (3.9) we can derive the following system of linear equations. For every k we may substitute x with z_k in Eq. (3.9) and obtain

$$0 = \sum_{i=1}^{m} \alpha_i w_i^2 g_i(z_k) = \alpha_k w_i^2 g_k(z_k). \tag{3.10}$$

Since both w_i^2 and $g_k(z_k)$ are non-zero, Eqs. (3.10) imply $a_k = 0$, thereby completing the first, direct proof. □

The second proof is very similar to the proofs of Thms. 3.12, 3.16, and 3.14. We provide a second proof, since it illustrates the methodology used in all of these proofs in greater detail. Here, we use the properties of the Fourier transformed model functions. However, before beginning the second proof, we require an auxiliary lemma.

Lemma 3.11. *Let $c_1, \ldots, c_m$ be pairwise distinct real numbers and $\alpha_1, \ldots, \alpha_m \in \mathbb{C}$ such that*

$$\sum_{j=1}^{m} \alpha_j e^{-\mathbf{i} c_j \omega} \overset{\omega \to \infty}{\to} 0. \tag{3.11}$$

Then, $\alpha_j = 0$ must hold for all $j = 1, \ldots, m$.

Proof. This proof is conducted in two steps. First, we prove that this convergence implies that there are $\beta_j \in \mathbb{C}$ such that

$$\sum_{j=1}^{m} \beta_j \mathrm{e}^{-\mathbf{i}c_j x} = 0. \tag{3.12}$$

Second, we show that the trigonometric functions $\mathrm{e}^{-\mathbf{i}c_j x}$ are linearly independent, if the parameters c_j are pairwise distinct.

The first step uses an argument similar to a finite version of CANTOR's diagonal argument. We construct a sequence of points $(\omega_k)_{k\in\mathbb{N}}$, such that all of the m summands in Eq. (3.11) converge simultaneously.

First, we need to address a technicality. We assume that $|c_1| = \max\{|c_j| \mid j = 1,\dots,m\}$. Usually, this implies $c_1 \neq 0$. However, $c_1 = 0$ is possible, if $m = 1$. We consider two cases. First, let $c_1 = 0$. Then, Eq. (3.11) implies

$$\begin{aligned} \alpha_1 \mathrm{e}^{-\mathbf{i}0} &= 0 \\ \alpha_1 &= 0, \end{aligned}$$

which is what we wanted to prove.

Now, let $c_1 \neq 0$. To simplify the notation we define $f_j(\omega) \coloneqq \mathrm{e}^{-\mathbf{i}c_j\omega}$. The first summand $\alpha_1 f_1(\omega)$ is $\frac{2\pi}{c_1}$-periodic, therefore $\alpha_1 f_1(\omega_k^{(1)}) = \alpha_1$ for the sequence $w_k \coloneqq k \cdot \frac{2\pi}{c_1}$.

If we now examine the second summand, we see that set $Im_2 \coloneqq \left\{\alpha_2 f_2\left(\omega_k^{(1)}\right) \mid k \in \mathbb{N}\right\}$ is bounded by α_2. Using the theorem by BOLZANO and WEIERSTRASS [13, Thm. 3.4.8], there exists a subsequence $\left(\omega_k^{(2)}\right) \subseteq \left(\omega_k^{(1)}\right)$, for which

$$\begin{aligned} \alpha_1 f_1\left(\omega_k^{(2)}\right) &\overset{k\to\infty}{\to} \beta_1 (= \alpha_1) \\ \alpha_2 f_2\left(\omega_k^{(2)}\right) &\overset{k\to\infty}{\to} \beta_2 \end{aligned}$$

holds for some $\beta_2 \in Im_2$. Iteratively using BOLZANO-WEIERSTRASS lets us define the subsequences $\left(\omega_k^{(j)}\right)$ for $j = 1,\dots,m$ which fulfill the following convergences

$$\begin{aligned} \alpha_1 f_1\left(\omega_k^{(m)}\right) &\overset{k\to\infty}{\to} \beta_1, \\ \alpha_2 f_2\left(\omega_k^{(m)}\right) &\overset{k\to\infty}{\to} \beta_2, \\ \vdots \qquad &\qquad \vdots \\ \alpha_m f_m\left(\omega_k^{(m)}\right) &\overset{k\to\infty}{\to} \beta_m. \end{aligned}$$

Note that for all $j = 1,\dots,m$ we have $\beta_j \in Im_j = \left\{\alpha_j f_j\left(\omega_k^{(j-1)}\right) \mid k \in \mathbb{N}\right\}$, and therefore $|\beta_j| = |\alpha_j|$.

It turns out that $\left(\omega_k^{(m)}\right)$ is not the only sequence on which all of the summands converge simultaneously. Let $x \in \mathbb{R}$. We define $\omega_k^{(m,x)} \coloneqq \omega_k^{(m)} + x$ and obtain

$$\alpha_j f_j\left(\omega_k^{(m,x)}\right) = \alpha_j f_j\left(\omega_k^{(m)}\right) \cdot \mathrm{e}^{-\mathbf{i}c_j x} \overset{k\to\infty}{\to} \beta_j \cdot \mathrm{e}^{-\mathbf{i}c_j x}. \tag{3.13}$$

This equality holds for all $j = 1, \dots, m$ and all $x \in \mathbb{R}$. Using Eq. (3.13), we finally have

$$\sum_{j=1}^{m} \alpha_j f_j \left(\omega_k^{(m,x)} \right) \overset{k\to\infty}{\to} \sum_{j=1}^{m} \beta_j \mathrm{e}^{-\mathbf{i}c_j x}.$$

Since for all $x \in \mathbb{R}$ the sequence $\left(\omega_k^{(m,x)}\right)_{k\in\mathbb{N}}$ diverges to ∞, we know that for all $x \in \mathbb{R}$ Eq. (3.12) must hold.

The second step involves proving the linear independence of the functions $\mathrm{e}^{-\mathbf{i}c_j x}$.

We use a Vandermonde matrix. Let $x_1 := \frac{\pi}{2\cdot|c_1|}$. Since $|c_1| > |c_j|$ for $j \neq 1$ we have $x_1 \cdot c_j \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$. This implies

$$\mathrm{e}^{-\mathbf{i}x_1 c_i} \neq \mathrm{e}^{-\mathbf{i}x_1 c_j} \qquad \text{for } i \neq j. \tag{3.14}$$

We then define $x_k = k \cdot x_1$ for $k \in \{0, 1, 2, \dots, m-1\}$ and evaluate Eq. (3.12) at all points x_k.

The resulting linear system of equations can be represented as follows:

$$\underbrace{\begin{pmatrix} \mathrm{e}^{-\mathbf{i}x_0 c_1} & \mathrm{e}^{-\mathbf{i}x_0 c_2} & \dots & \mathrm{e}^{-\mathbf{i}x_0 c_m} \\ \mathrm{e}^{-\mathbf{i}x_1 c_1} & \mathrm{e}^{-\mathbf{i}x_1 c_2} & \dots & \mathrm{e}^{-\mathbf{i}x_1 c_m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{e}^{-\mathbf{i}x_{m-1} c_1} & \mathrm{e}^{-\mathbf{i}x_{m-1} c_2} & \dots & \mathrm{e}^{-\mathbf{i}x_{m-1} c_m} \end{pmatrix}}_{:=V} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ \left(\mathrm{e}^{-\mathbf{i}x_1 c_1}\right)^1 & \left(\mathrm{e}^{-\mathbf{i}x_1 c_2}\right)^1 & \dots & \left(\mathrm{e}^{-\mathbf{i}x_1 c_m}\right)^1 \\ \vdots & \vdots & \ddots & \vdots \\ \left(\mathrm{e}^{-\mathbf{i}x_1 c_1}\right)^{m-1} & \left(\mathrm{e}^{-\mathbf{i}x_1 c_2}\right)^{m-1} & \dots & \left(\mathrm{e}^{-\mathbf{i}x_1 c_m}\right)^{m-1} \end{pmatrix} = V.$$

We can see that V is indeed a Vandermonde matrix. Using the well-known formula for its determinant, see [10, Ch. 16, Prob. M.3], we obtain

$$\det(V) = \prod_{j<k} \left(\mathrm{e}^{-\mathbf{i}x_1 c_k} - \mathrm{e}^{-\mathbf{i}x_1 c_j} \right) \neq 0,$$

using Eq. (3.14). Therefore, the unique solution to this system of equations is the trivial $(\beta_1, \beta_2, \dots, \beta_m)^T = 0$. If we then recall that $|\alpha_j| = |\beta_j|$, the proof is complete. □

Proof of Thm. 3.10 in the time domain. We consider the Fourier transformed Lorentz functions $\frac{1}{C(w_j)} \cdot \widehat{L_{c_j,w_j}}(x) = \mathrm{e}^{-\mathbf{i}c_j\omega}\mathrm{e}^{-w_j|\omega|}$. Let $\alpha_j \in \mathbb{C} \setminus \{0\}$ such that

$$\sum_{j=1}^{n} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega}\mathrm{e}^{-w_j|\omega|} = 0. \tag{3.15}$$

The idea behind this proof is that these functions all converge at different rates. The convergence rate of the sum is defined by the function converging the slowest. We can eliminate one parameter as follows: Let $w = \min\{w_j \mid j = 1, \dots, m\}$, $J = \{j \mid w_j = w\}$ and

we multiply Eq. (3.15) by $e^{w|\omega|}$, essentially dividing by the smallest rate of decay.

$$\sum_{j\in J}\alpha_j e^{-\mathbf{i}c_j\omega} + \sum_{j\notin J}\alpha_j e^{-(w_j-w)|\omega|}\cdot e^{-\mathbf{i}c_j\omega} = 0$$
$$\sum_{j\in J}\alpha_j e^{-\mathbf{i}c_j\omega} + 0 \overset{\omega\to\infty}{\to} 0.$$

As ω approaches infinity, all functions with larger convergence rates vanish. Note that for $i \neq j \in J$ we have $c_i \neq c_j$, since $w_i = w_j = w$. Now, Lem. 3.11 implies that $\alpha_j = 0$ for $j \in J$, which contradicts our assumption and completes the proof. □

3.1.3 Gauss Functions

Theorem 3.12. *Let $m \in \mathbb{N}$ and let $\{(c_i, w_i) \mid i = 1,\dots,m,\ c_i, w_i \in \mathbb{R},\ w_i > 0\}$ be a set of pairwise distinct parameter tuples. Then, the set of Gaussians $\{G_{c_i,w_i} \mid i = 1,\dots,m\}$ is linearly independent. In other words: Any finite number of different Gaussians are linearly independent.*

It is possible to prove the linear independence of different Gaussians without the help of the Fourier transform[6]. However, after transforming them, we achieve a useful separation of the parameters c_j and w_j that can be exploited when examining the asymptotic behavior of the summands.

Proof. We only need to show that a finite set of Fourier transformed Gaussians is linearly independent. Now, let $\{(c_j, w_j) \in \mathbb{R}\times\mathbb{R}_{>0} \mid j = 1,\dots,m\}$ be the set of parameters of the functions $\frac{1}{C(v_j)}\cdot\widehat{G_{c_j,w_j}}(x) = e^{-\mathbf{i}c_j\omega}e^{-v_j^2\omega^2}$, where $v_j = \frac{w_j}{2\sqrt{\ln(2)}}$, and assume w.l.o.g. that $\alpha_j \in \mathbb{C}\setminus\{0\}$ exist, such that

$$\sum_{j=1}^{m}\alpha_j e^{-\mathbf{i}c_j\omega}e^{-v_j^2\omega^2} = 0. \tag{3.16}$$

Similar to the previous proof, we need to divide by the smallest rate of decay. We define $v = \min\{v_j \mid j = 1,\dots,m\}$ and $J = \{j \mid v_j = v\}$. Multiplying Eq. (3.16) by $e^{v^2\omega^2}$ leads to

$$\sum_{j\in J}\alpha_j e^{-\mathbf{i}c_j\omega} + \sum_{j\notin J}\alpha_j e^{-\mathbf{i}c_j\omega}\cdot e^{-(v_j^2-v^2)\omega^2} = 0$$
$$\sum_{j\in J}\alpha_j e^{-\mathbf{i}c_j\omega} \overset{\omega\to\infty}{\to} 0,$$

using the fact that the second sum converges to zero. Again, Lem. 3.11 implies $\alpha_j = 0$ for all $j \in J$. This contradicts the initial assumption. □

3.1.4 Pseudo-Voigt Functions

In the case of pseudo-Voigt functions, there is another type of ambiguity that must be addressed first. Since these functions consist of a convex linear combination of a Lorentzian and a Gaussian, changing only the parameter λ while fixing c and w does not lead to linearly independent functions.

Lemma 3.13. *Let $m \in \mathbb{N}, m > 1$, $\lambda_1,\dots,\lambda_m \in [0,1]$ be pairwise distinct, and let $(c,w) \in \mathbb{R}\times\mathbb{R}_{>0}$. Then, the spectrum defined by the sum of the corresponding pseudo-Voigt functions has ambiguous decompositions.*

[6] One example can be found online: `https://math.stackexchange.com/questions/1081350`

Proof. We define $\Lambda = \frac{1}{m}\sum_{j=1}^{m} \lambda_i \in [0,1]$. Then,

$$\begin{aligned}\sum_{j=1}^{m} pV_{c,w,\lambda_i}(x) &= \sum_{j=1}^{m} \lambda_j G_{c,w}(x) + \sum_{j=1}^{m}(1-\lambda_j)L_{c,w}(x)\\ &= m\cdot\Lambda G_{c,w}(x) + m\cdot(1-\Lambda)L_{c,w}(x)\\ &= m\cdot pV_{c,w,\Lambda}(x).\end{aligned}$$

□

In Cor. 3.15, we show that this type of linear dependence between pseudo-Voigt functions is the only possible type. Therefore, if the tuples (c_j, w_j) are distinct, the corresponding pseudo-Voigt functions are linearly independent. We prove a slightly different version of this statement in the following theorem.

Theorem 3.14. *Let $j = 1,\dots,m, k = 1,\dots,n$, and $(c_j, v_j), (d_k, w_k) \in \mathbb{R}\times\mathbb{R}_{>0}$, where for $a \neq b$ we have $(c_a, v_a) \neq (c_b, v_b)$ and $(d_a, w_a) \neq (d_b, w_b)$. However, we allow equalities in between tuples from different sets, i.e., $(c_j, v_j) = (d_k, w_k)$ for some indices j, k.*

Then, the set

$$\left\{G_{c_j,v_j} \mid j = 1,\dots,m\right\} \cup \left\{L_{d_k,w_k} \mid k = 1,\dots,n\right\}$$

of Gauss and Lorentz functions is linearly independent. In other words: Any finite number of different Gaussians and Lorentzians are linearly independent.

A special case of this theorem are pseudo-Voigt functions, where $n = m$ and for all $j = 1,\dots,m$ we have $(c_j, v_j) = (d_j, w_j)$, since we only require $(c_i, v_i) \neq (c_j, v_j)$ for $i \neq j$.

Proof. The case of $n = 0$ is proven by Thm. 3.12. Therefore, let $n \geq 1$. Using the well-established scheme, we assume that $\alpha_j, \beta_k \in \mathbb{C}\setminus\{0\}$ exist, such that

$$\sum_{j=1}^{m} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega}\mathrm{e}^{-v_j^2\omega^2} + \sum_{k=1}^{n} \beta_k \mathrm{e}^{-\mathbf{i}d_k\omega}\mathrm{e}^{-w_k|\omega|} = 0. \tag{3.17}$$

Since $n \geq 1$, we know that the slowest converging summand is a Fourier transformed Lorentzian. We can then define $w = \min\{w_k \mid k = 1,\dots,n\}$ and $K = \{k \mid w_k = w\}$ is not empty. We then multiply Eq. (3.17) by $\mathrm{e}^{w|\omega|}$, the inverse of the slowest converging function, and obtain

$$\begin{aligned}\sum_{j=1}^{n} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega}\mathrm{e}^{-v_j^2\omega^2+w\omega} + \sum_{k\in K} \beta_k \mathrm{e}^{-\mathbf{i}d_k\omega} + \sum_{k\notin K} \beta_k \mathrm{e}^{-\mathbf{i}d_k\omega}\mathrm{e}^{-(w_k-w)\omega} &= 0\\ \sum_{k\in K} \beta_k \mathrm{e}^{-\mathbf{i}d_k\omega} \overset{\omega\to\infty}{\to} 0.\end{aligned}$$

The first and the third sums both converge to zero and by Lem. 3.11, we know that for $k \in K$ all $\beta_k = 0$, contradicting the assumption. □

The subsequent corollary immediately follows from this theorem.

Corollary 3.15. Let s be a minimal NMR spectrum consisting of pseudo-Voigt functions with ambiguous decompositions

$$s = \sum_{i=1}^{m} \alpha_i \cdot pV_{c_i,v_i,\lambda_i} = \sum_{j=1}^{n} \beta_j \cdot pV_{d_j,w_j,\mu_j}. \tag{3.18}$$

Then, for every $i = 1, \ldots, m$ there is some $j = 1, \ldots, n$ such that $(c_i, v_i) = (d_j, w_j)$, or $\sum_{k \in I} \alpha_k = \sum_{k \in I} \alpha_k \lambda_k = 0$, where $I = \{k \mid (c_k, v_k) = (c_i, v_i)\}$.

In other words, if an ambiguous decomposition exists, then it has ambiguities of the type presented in Lem. 3.13, or the spectrum contains superfluous zero summands allowed by the structure of the pseudo-Voigt functions.

An example of a non-trivial zero only requires three summands. Let $(c, w) \in \mathbb{R} \times \mathbb{R}_{>0}$ and $\alpha_1 = 1, \alpha_2 = -\frac{3}{2}, \alpha_3 = \frac{1}{2}, \lambda_1 = \frac{1}{2}, \lambda_2 = \frac{1}{3}, \lambda_3 = 0$. Then

$$\begin{aligned}\sum_{i=1}^{3} \alpha_i \cdot pV_{c,w,\lambda_i} &= \left(1 \cdot \frac{1}{2} - \frac{3}{2} \cdot \frac{1}{3} + \frac{1}{2} \cdot 0\right) G_{c,w} + \left(1 \cdot \frac{1}{2} - \frac{3}{2} \cdot \frac{2}{3} + \frac{1}{2} \cdot 1\right) L_{c,w} \\ &= 0 G_{c,w} + 0 L_{c,w} = 0,\end{aligned}$$

even though the parameter tuples (c, w, λ_i) are distinct.

Proof. We conduct this proof by contradiction. Assume that there is an index $\ell \in \{1, \ldots, m\}$ such that $(c_\ell, v_\ell) \neq (d_j, w_j)$ for all $j = 1, \ldots, n$ and that

$$\sum_{k \in I} \alpha_k \neq 0 \qquad \text{or}$$
$$\sum_{k \in I} \alpha_k \lambda_k \neq 0,$$

where $I = \{k \mid (c_k, v_k) = (c_\ell, v_\ell)\}$. We rewrite Eq. (3.18) to separate all Gaussians and Lorentzians with parameters (c_ℓ, v_ℓ).

$$\begin{aligned}\sum_{j=1}^{n} \beta_j \cdot pV_{d_j,w_j,\mu_j} - \sum_{i \notin I} \alpha_i \cdot pV_{c_i,v_i,\lambda_i} &= \sum_{k \in I} \alpha_k \cdot pV_{c_\ell,v_\ell,\lambda_k} \\ &= \left(\sum_{k \in I} \alpha_k \lambda_k\right) G_{c_\ell,v_\ell} + \left(\sum_{k \in I} \alpha_k - \sum_{k \in I} \alpha_k \lambda_k\right) L_{c_\ell,v_\ell}.\end{aligned}$$

By our assumption, one of the two coefficients on the right-hand side is non-zero. Additionally, all of the Gaussians and Lorentzians appearing on the left have parameters different from (c_ℓ, v_ℓ). This implies a non-trivial linear combination of the Gaussian or the Lorentzian on the right hand side, which does not exist by Thm. 3.14. □

3.1.5 Voigt Functions

Lastly, we study the linear independence of Voigt functions.

Theorem 3.16. *Let $m \in \mathbb{N}$ and let $\{(c_i, v_i, w_i) \mid i = 1, \ldots, m,\ c_i, v_i, w_i \in \mathbb{R},\ v_i, w_i > 0\}$ be a set of pairwise distinct parameter tuples. Then, the set of Voigt functions $\{V_{c_i,v_i,w_i} \mid i = 1, \ldots, m\}$ is linearly independent. In other words: Any finite number of different Voigt functions are linearly independent.*

Proof. Once again, we show that a finite set of Fourier transformed Voigt functions is linearly independent. Let $\{(c_j, v'_j, w_j \mid j = 1, \ldots, m\}$ be the set of parameters of the functions

$$\frac{1}{C(v'_j, w_j)} \widehat{V_{c_j,v'_j,w_j}} = \mathrm{e}^{-\mathbf{i} c_j \omega} \cdot \mathrm{e}^{-w_j|\omega|} \cdot \mathrm{e}^{-(v_j \omega)^2},$$

where $v_j = \frac{v'_j}{2\sqrt{\ln 2}}$.

Assume that there are $\alpha_j \in \mathbb{C} \neq 0$ such that

$$\sum_{j=1}^{m} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega} \cdot \mathrm{e}^{-w_j|\omega|} \cdot \mathrm{e}^{-(v_j\omega)^2} = 0. \tag{3.19}$$

Again, we want to eliminate the function with the smallest rate of convergence. Therefore, we define

1. $v = \min\{v_j \mid j = 1, \ldots, n\}$ and $w = \min\{w_j \mid v_j = v, j = 1, \ldots, n\}$
2. $I_1 = \{j \mid v_j = v, w_j = w\} \neq \emptyset$ and $I_2 = \{j \mid v_j = v, w_j \neq w\}$
3. $I_3 = \{1, \ldots, n\} \setminus (I_1 \cup I_2)$.

The functions f_j, with $j \in I_1$, have the smallest rate of convergence. We multiply Eq. (3.19) by $\mathrm{e}^{w|\omega|+(v\omega)^2}$ and obtain

$$\sum_{j\in I_1} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega} + \sum_{j\in I_2} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega} \cdot \mathrm{e}^{-(w_j-w)|\omega|} + \sum_{j\in I_3} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega} \cdot \mathrm{e}^{-(w_j-w)|\omega|} \cdot \mathrm{e}^{-(v_j^2-v^2)\omega^2} = 0.$$

Again, as ω approaches infinity, we observe that both the second and third sums converge to zero, since I_2 and I_3 only contain indices of functions with strictly larger rates of convergence. Consequently, the following convergence arises

$$\sum_{j\in I_1} \alpha_j \mathrm{e}^{-\mathbf{i}c_j\omega} \overset{\omega\to\infty}{\to} 0.$$

Now, we use Lem. 3.11 to confirm that all $\alpha_j = 0$ for $j \in I_1$, which contradicts our assumption. □

3.1.6 Outlook: Different Types of Model Functions

These proofs were conducted only for the four most commonly used types of model functions. There are certainly more families of model functions suitable for modeling NMR spectra, for example skewed pseudo-Voigt functions [116] might also result in unique decompositions. In theory, it may be possible to prove the linear independence of *frequency-width* families, i.e., families containing one model function $f_{0,1}(x) \in L^1(\mathbb{R}) \cap \mathcal{C}(\mathbb{R}) \setminus \{0\}$ with all other model functions being obtained via the parameter transformation $f_{c,w}(x) = f_{0,1}\left(\frac{x-c}{w}\right)$. A similar statement can be found when studying families of *time-frequency translated* functions. The HRT conjecture, named after HEIL, RAMANATHAN and TOPIWALA, studies the set of functions

$$\{\exp(2\pi\mathbf{i} \cdot bt) \cdot f(t + a) \mid a, b \in \mathbb{R}\},$$

where $f \in L^2(\mathbb{R}) \setminus \{0\}$. The conjecture claims that for all such f any finite subset of this family of functions is linearly independent [51]. Several advancements have been made regarding the HRT conjecture [15, 46, 68, 78]. Unfortunately, an HRT-like conjecture does not hold in our case, where the set of possible functions is

$$\left\{ f_{c,w}(t) = f\left(\frac{t-c}{w}\right) \mid c \in \mathbb{R}, w \in \mathbb{R}_+ \right\}.$$

For example, consider the triangular function, also known as the hat function

$$f(t) = (1 - |t|) \cdot \mathbb{1}_{[-1,1]}(t) \in L^2(\mathbb{R}).$$

Here, the equation

$$2f_{0,2}(t) = f_{-1,1}(t) + 2f_{0,1}(t) + f_{1,1(t)}$$

presents a counterexample to a possible conjecture similar to the HRT conjecture. Fig. 3.1 illustrates this equation.

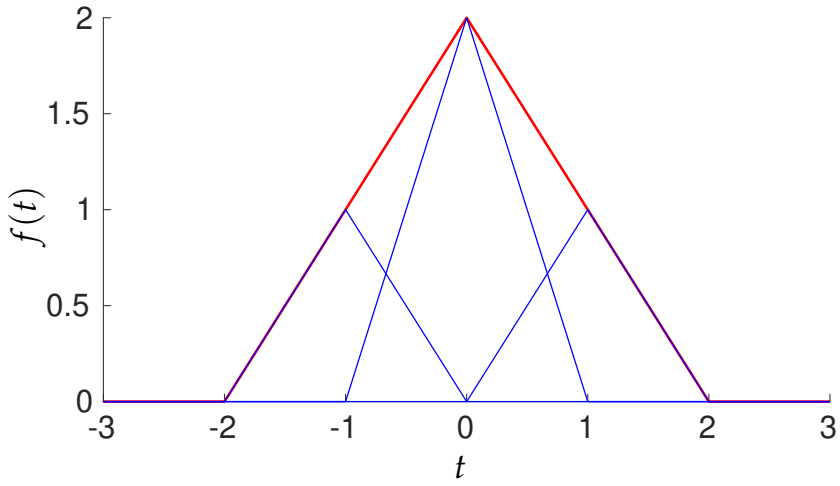

FIGURE 3.1: Counterexample to the HRT-like conjecture.

The fact that hard modeling NMR spectra has – at least in theory and under perfect conditions – only one solution, if no two functions with equal centers and half-widths are allowed, seems to be counterintuitive. In fact, problems often arise when dealing with experimental data algorithmically, whenever multiple solutions with significantly different parameters fulfill the modeling problem. For the next section, we move one step closer to experimental data by assuming that the spectrum s is not continuous, but is measured at a finite set of points $x \in X$.

3.2 The Discrete Problem

In the discrete case, one type of ambiguous solution can be summarized by the term *overfitting*. If too many model functions are chosen to model a given spectrum, which is only evaluated at a discrete set of usually equidistant points $X = \{x_1, \ldots, x_N\}$, then intuitively, it should be possible to ambiguously solve the equations defined by $\sum_{j=1}^{m} \alpha_j \cdot f_j(x_i) = s(x_i)$ for some sufficiently large m. However, the linear independence of the functions f_j does not imply linear independence of all vectors $v_j = (f_j(x_1), \ldots, f(x_N))^T$. The only statement that can be made, is that there exists a set $X \subseteq \mathbb{R}$ such that these vectors v_j are linearly independent. Using more sophisticated arguments, we can show that a large number of model functions can indeed interpolate any spectrum $s(x_i)$. Additionally, it is interesting to determine when exactly a modeling problem with a unique solution becomes ambiguous, and thus the second and third questions from the beginning can be rewritten as:

2. How many model functions are necessary to allow the existence of ambiguous solutions?

3. How many points does the spectrum need to contain for the solution to be unique given a fixed number of model functions?

Attempting to answer these questions necessitates providing a rigorous definition of the discrete modeling problem first.

Definition 3.17. *Let $X = \{x_1 < x_2 < \ldots < x_k\} \subseteq \mathbb{R}$ and let s be an NMR spectrum composed of functions f of type $\mathcal{F}$. The discrete modeling problem is to find functions $f_j \in \mathcal{F}$ and coefficients $\alpha_j \in \mathbb{R} \setminus \{0\}$ such that for all $x_i \in X$ we have*

$$s(x_i) = \sum_{j=1}^{m} \alpha_j \cdot f_j(x_i). \tag{3.20}$$

This definition gives rise to another question. Is overfitting possible with our model functions? In other words: Must s be an NMR spectrum composed of, e.g., pseudo-Voigt functions, or can any set of arbitrary values be fitted?

Lemma 3.18. *Let $X \subseteq \mathbb{R}$ be a set of k points and let $s \in \mathbb{R}^k$. Let $\mathcal{F} \in \{Lor, Gau, pVoi, Voi\}$ be a family of functions. Then, the discrete modeling problem with arbitrary values has at least one solution, or in other words: There exist k distinct functions $f_1, \ldots, f_k \in \mathcal{F}$ and $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$ such that*

$$s_i = \sum_{j=1}^{k} \alpha_j \cdot f_j(x_i). \tag{3.21}$$

Proof. The idea of this proof is to construct model functions that behave similarly to the KRONECKER delta $\delta_{i,j}$ and show that the matrix resulting from Eq. (3.21) is invertible.

Assume that we can choose model functions $f_1, \ldots, f_k$ with the following properties. First, we require that $f_i(x_i) = 1$ for all $i = 1, \ldots, k$. Second, we require that $0 \leq f_j(x_i) \leq \frac{1}{k}$ for $i \neq j$. Later on, we show that it is indeed possible to choose suitable $f \in \mathcal{F}$.

The matrix representing the system of equations has the following form

$$\begin{aligned} A &= \begin{pmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_k(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_k(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_k) & f_2(x_k) & \cdots & f_k(x_k) \end{pmatrix} \\ &= I_k - E, \end{aligned}$$

where $I_k \in \mathbb{R}^{k \times k}$ is the identity matrix. According to the properties of the NEUMANN series [104, Cor. VI. 5], $A = I_k - E$ is invertible, if for some operator norm, $\|E\| < 1$. Choosing the 1-operator norm, we have

$$\max_{j=1,\ldots,k} \sum_{\substack{i=1 \\ i \neq j}}^{k} |-f_j(x_i)| \leq \frac{k-1}{k} < 1.$$

Therefore, A is invertible (with $A^{-1} = \sum_{n=0}^{\infty} E^n$) and Eq. (3.21) has a unique solution in α_i. Note that the uniqueness only holds for the parameters of the functions f_j specifically chosen for this proof. If we allow the parameters to be variable as well, possibilities for ambiguous solutions would arise. Note that we allow $\alpha_i = 0$, in which case the points are still interpolated by fewer model functions. If, for all $i = 1, \ldots, k$ we have $\alpha_i = 0$, then we define that zero model functions solve the interpolation problem.

Finally, we demonstrate that our model functions can satisfy the two required properties. For Lorentzians, Gaussians and pseudo-Voigt functions, we can choose $c_i = x_i$ to fulfill the first property. Additionally, for sufficiently small values of $w_i > 0$, we can guarantee that

$f_i(x_j) < \varepsilon$ for all $\varepsilon > 0$. This can be easily verified by observing Eqs. (2.8),(2.9) and (2.10) in the limit $w \searrow 0$ and noting that all of these functions vanish, when $x \to \infty$.

In the case of Voigt functions, we observe the following: A sequence of Gaussians with half-widths $v_n \to 0$ satisfies the properties of an *approximate identity*, i.e., the convolution $G_{c,v_n} * f(x) \overset{n\to\infty}{\to} f(x)$ w.r.t. the L^1-norm [50, Thm. 1.71] for any $f \in L^1(\mathbb{R})$. If we choose $f = L_{c,w}$, then the fact that both $\lim_{n\to\infty} V_{c,v_n,w}$ and $L_{c,w}$ are continuous already implies pointwise convergence. Thus, for every $\varepsilon > 0$ there exists some $\delta > 0$ such that for all $v < \delta$, we have $|V_{c,v,w}(x) - L_{c,w}(x)| < \varepsilon$ for all $x \in \{x_1, \ldots, x_k\}$. Therefore, we can define the v_i in such a way that $|V_{c_i,v_i,w_i}(x_j) - L_{c_i,w_i}(x_j)| < \frac{1}{2k^2}$ for all $j = 1, \ldots, k$. Then,

$$|V_{c_i,v_i,w_i}(x_i) - 1| = |V_{c_i,v_i,w_i}(x_i) - L_{c_i,w_i}(x_i) + L_{c_i,w_i}(x_i) - 1| < \frac{1}{2k^2}$$
$$|V_{c_i,v_i,w_i}(x_j)| = |V_{c_i,v_i,w_i}(x_j) - L_{c_i,w_i}(x_j) + L_{c_i,w_i}(x_j)| < \frac{1}{2k^2} + \frac{1}{k}.$$

The absolute sums of the columns of the resulting matrix E are at most

$$\sum_{i=1}^{n} |E(i,j)| < \frac{1}{2k^2} + \sum_{\substack{i=1 \\ i\neq j}} \left(\frac{1}{2k^2} + \frac{1}{k}\right) \tag{3.22}$$
$$= \frac{1}{2k} + \frac{k-1}{k} = 1 - \frac{1}{2k} < 1. \tag{3.23}$$

Thus, the initial matrix A is invertible and a solution also exists for Voigt functions. □

As it turns out, selecting too many model functions creates ambiguous solutions to all interpolation problems.

From now on, we are only interested in fitting actual spectral data, meaning values $s(x_i)$ obtained by evaluating NMR spectra s at some points $x_i \in X$. Therefore, we extend the definition of ambiguity to the discrete problem.

Definition 3.19. *Let $\{x_1 < \ldots < x_k\} = X \subseteq \mathbb{R}$. An NMR spectrum s with a minimal decomposition $\{(\alpha_i, p_i) \mid i = 1, \ldots, m\}$ is* (discretely) ambiguous *on X, if another minimal decomposition $\{(\beta_j, q_j) \mid j = 1, \ldots, n\}$ exists, such that $p_i \neq q_j$ for all $i \neq j$, and where*

$$s(x_\ell) = \sum_{i=1}^{m} \alpha_i f_{p_i}(x_\ell) = \sum_{j=1}^{n} \beta_j f_{q_j}(x_\ell) \qquad \text{for } \ell = 1, \ldots, k. \tag{3.24}$$

This definition has some drawbacks. According to Lem. 3.18, all spectra are discretely ambiguous, if the number of model functions is sufficiently large. However, we also know that, for some sets of infinitely many points x, e.g., $X = \mathbb{R}$ or, using continuity, $X = \mathbb{Q}$, the decomposition is unique for all possible $n \in \mathbb{N}$. This implies that there is a number of points, dependent on the numbers m and n, where unique and ambiguous solutions change. We study this behavior from a cautious perspective, trying to guarantee a unique solution. Therefore, it is possible that a set X_1 of k points exists, where a spectrum has two decompositions into m and n functions, whereas another set X_2 of k points guarantees a unique decomposition.

Example 3.20. We define $s = 1 \cdot L_{-1,1} + 1 \cdot L_{1,1}$ and evaluate this spectrum on the two sets of points $X_1 = \{-2, 0, 2\}$ and $X_2 = \{-1, 0, 1\}$. For X_1 and $n = 1$, there is a spectrum $t = 1 \cdot L_{0,\sqrt{6}}$ consisting of a single Lorentzian that interpolates s. However, there is no spectrum composed of a single Lorentzian that interpolates s on X_2. To prove that, assume

there are α, c, w such that

$$\alpha \frac{w^2}{w^2 + (c+1)^2} = s(-1) = \frac{6}{5}$$
$$\alpha \frac{w^2}{w^2 + (c)^2} = s(0) = 1$$
$$\alpha \frac{w^2}{w^2 + (c-1)^2} = s(1) = \frac{6}{5}.$$

The first and third equations imply $c = 0$. Then, the second equation gives $\alpha = 1$. Finally, the first equation can be transformed into $w^2 = -6$, which has no real solutions. Fig. 3.2 illustrates this example.

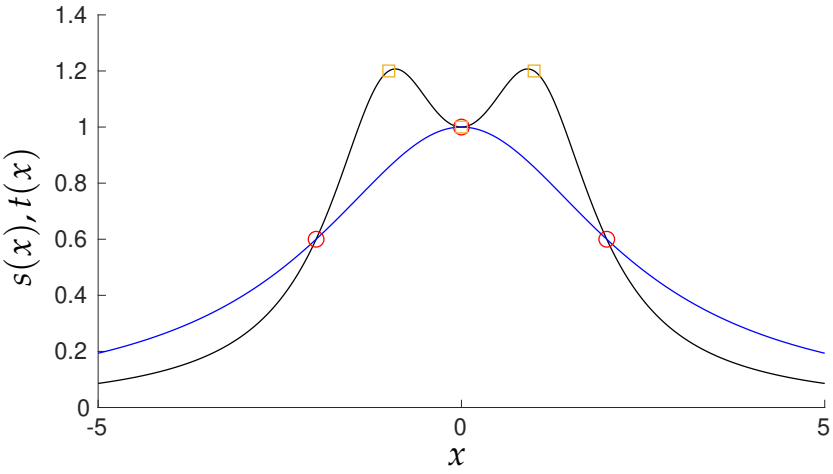

FIGURE 3.2: Two sets of three points, allowing one Lorentzian to fit all (red) and requiring two to be fitted (yellow).

To guarantee that no spectrum consisting of n functions can interpolate the initial spectrum s on any set of k points, we need to find the maximum number of intersections, that two spectra composed of m and n functions can have. If we then choose k to be larger than this number, our problem is guaranteed to have a unique solution.

Definition 3.21. *Let $\mathcal{F} \in \{Lor, Gau, pVoi, Voi\}$, $m, n \geq 1$. Then, $k_{\mathcal{F}}(m, n)$ is the maximum number of points $X = \{x_1, \ldots, x_k\}$ such that there exists an NMR spectrum s, defined by the minimal decomposition $\{(\alpha_i, p_i) \mid i = 1, \ldots, m\}$, which is discretely ambiguous on X with a different minimal decomposition of $\leq n$ model functions $f_j \in \mathcal{F}$.*

In other words, if $k_{\mathcal{F}}(m, n) + 1$ points are given, and s is defined by n model functions, then no second decomposition consisting of m or fewer functions exists.

There are two main ways to obtain values for $k_{\mathcal{F}}(m, n)$. The first is to find a lower bound. If we can construct different spectra s and t that intersect at k points, then $k \leq k_{\mathcal{F}}(m, n)$. Lem. 3.18 implies that any spectrum can be interpolated by n specifically designed model functions on n points. Therefore, if we choose any set of n points $X \subseteq \mathbb{R} \setminus \{0\}$, then $L_{0,1}$ can be interpolated by n different model functions and we have $k_{\mathcal{F}}(1, n) \geq n$. To obtain an upper bound, we must consider the algebraic structure of possible ambiguous decompositions and need to estimate the number of possible zeros when subtracting both sides of Eq. (3.24).

3.2.1 Lower Bounds for $k_{\mathcal{F}}(m,n)$

We previously established that $k_{\mathcal{F}}(m,n) \geq k_{\mathcal{F}}(1,n) \geq n$. However, an inductive proof allows us to improve this bound for a more general class of functions.[7]

Theorem 3.22. *Let $f_{0,1} \in L^1(\mathbb{R})$ and define $\mathcal{F} := \{f_{c,w}(x) = f_{0,1}\left(\frac{x-c}{w}\right) \mid c, w \in \mathbb{R},\ w > 0\}$. If the functions $f_{c,w} \in \mathcal{F}$ fulfill the following set of conditions*

1. *$f_{0,1}$ is continuous,*
2. *$f_{0,1}$ is monotonically increasing for $x < 0$, and monotonically decreasing for $x > 0$,*
3. *$f_{0,1}$ is normed, i.e., $f_{0,1}(0) = 1$ and $f_{0,1}(1) = \frac{1}{2}$,*

then $k_{\mathcal{F}}(m,n) \geq 2m + 2n - 2$.

Proof. We prove this theorem by mathematical induction. First, note that

A: $\lim_{x \to \pm\infty} f_{0,1}(x) = 0$,

B: $\lim_{w \searrow 0} f_{c,w}(x) = 0$, for all $x \neq 0$.

C: $f_{c,w}(x) \geq 0$ for all $x \in \mathbb{R}$.

We quickly prove these three statements.

A: Since $f_{0,1} \in L^1(\mathbb{R})$ and $f_{0,1}(x)$ is monotonically decreasing for large $x > 0$, we know that for all $\varepsilon > 0$ eventually, $|f_{0,1}(x)| < \varepsilon$ holds. The same argument holds for large negative $x < 0$. This proves both convergences.

B: Statement *A* implies that $f_{c,w}(x) = f_{0,1}\left(\frac{x-c}{w}\right) \overset{w \searrow 0}{\to} 0$, if $x \neq 0$.

C: Statement *A* also implies that $f_{c,w}(x) \overset{x \to \pm\infty}{\to} 0$. Since the convergence is monotonic and $f_{c,w}(c) = 1$, statement *C* is proven.

Using an induction on the numbers of summands m and n, we prove the slightly more general statement that two sums f_m and f_n composed of m and n model functions of type $\mathcal{F}$ possess at least $2m + 2n - 1$ points $x_0, \ldots, x_{2m+2n-2}$ at which the sign of $f_m - f_n$ alternates, i.e., we prove that for all i we have $f_m(x_i) > f_n(x_i) \Rightarrow f_n(x_{i+1}) > f_m(x_{i+1})$, and vice versa. Since both sums are continuous functions, BOLZANO'S theorem [111] then implies the existence of at least $2m + 2n - 2$ intersections.

Base case: $m = n = 1$.

First we show that there are three points x_0, x_1, x_2 in which

$$f_m = f_{0,1}(x)$$
$$f_n = \frac{3}{2} \cdot f_{0,\delta}(x)$$

alternate. The intermediate value theorem [13, Thm. 5.3.7] implies that there exists a point $x^* < 0$, where $f_{0,1}(x^*) = \frac{1}{2}$. We choose a sufficiently small $\delta > 0$ using statement *B* such that $f_{0,\delta}(1) < \frac{1}{4}$ and $f_{0,\delta}(x^*) < \frac{1}{4}$.

[7] In the previous publication [52], this proof was constructive rather than inductive.

We define $x_0 = x^*, x_1 = 0, x_2 = 1$. Then,

$$f_{0,1}(x^*) = \frac{1}{2} > \frac{3}{8} > \frac{3}{2} f_{0,\delta}(x^*)$$
$$f_{0,1}(0) = 1 < \frac{3}{2} = \frac{3}{2} f_{0,\delta}(0)$$
$$f_{0,1}(1) = \frac{1}{2} > \frac{3}{8} > \frac{3}{2} f_{0,\delta}(1).$$

Fig. 3.3 illustrates this construction in the case of $\mathcal{F} = Lor$.

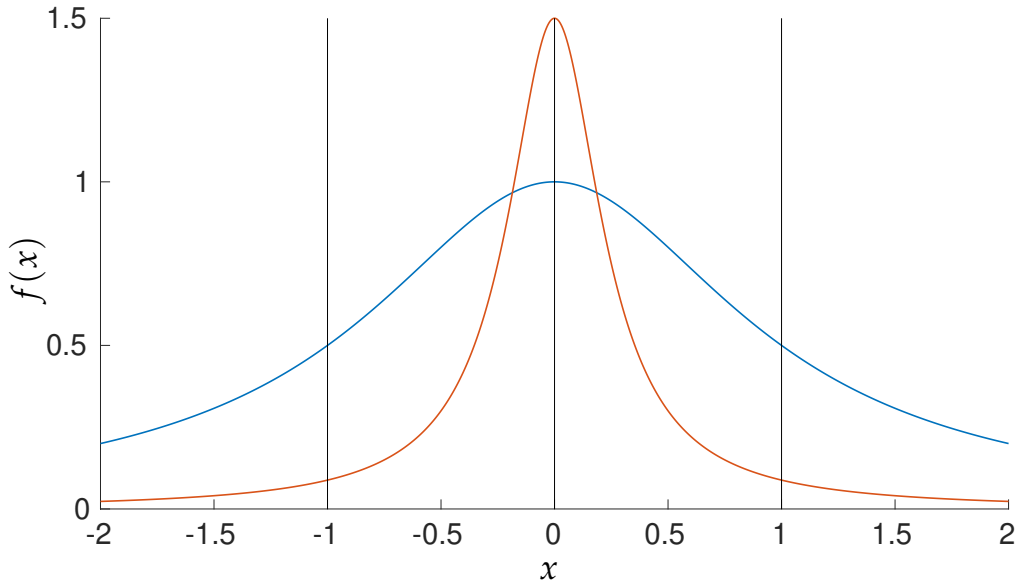

FIGURE 3.3: Construction for two Lorentzians, $L_{0,1}$ (blue) and $L_{0,\delta}$ (red), where $\delta = \frac{1}{4}$, $x^* = -1$. The points x_0, x_1, x_2 are denoted by black lines.

Induction step: $(m, n) \to (m+1, n)$.

Assume there are two sums f_m and f_n, that alternate at points $x_0 < \ldots < x_{2m+2n-2}$. Since $2m + 2n - 1 \geq 2$ there is a point x_i where $f_n(x_i) > f_m(x_i)$, and where $f_m(x_{i\pm1}) > f_n(x_{i\pm1})$, if $x_{i\pm1}$ exists. We append the sum f_m with one summand, which we define as $\alpha \cdot f_{x_i,\delta}$. This function is centered at x_i and has a linear combination coefficient of $\alpha = 2 \cdot (f_n(x_i) - f_m(x_i))$. Again, we need to choose δ to be sufficiently small. Note that there are two points x_L, x_R such that $x_{i-1} < x_L < x_i < x_R < x_{i+1}$, where $f_n(x_L) - f_m(x_L) > \frac{\alpha}{4}$ and $f_n(x_R) - f_m(x_R) > \frac{\alpha}{4}$. If one of the two points $x_{i\pm1}$ does not exist, then we omit the leftmost or rightmost inequality, respectively. Let

$$d^* := \min_{j=0,\ldots,2m+2n-2} |f_m(x_j) - f_n(x_j)| > 0$$
$$d := \min\left\{d^*, \frac{\alpha}{4}\right\}.$$

Finally, we choose δ using statement B such that $f_{x_i,\delta}(x_L) < \frac{d}{\alpha}$ and $f_{x_i,\delta}(x_R) < \frac{d}{\alpha}$.

We append the set of points with x_L and x_R such that it now contains $2m + 2n - 1$ elements

$$X = \{x_0 < \ldots < x_{i-1} < x_L < x_i < x_R < x_{i+1} < \ldots < x_{2m+2n-2}\}.$$

For the previously defined points of this set, all inequalities regarding f_m and f_n remain the same, except at x_i. Finally, for the appended sum f'_m, we have

$$f'_m(x_j) = \begin{cases} f_m(x_j) + \alpha \cdot f_{x_i,\delta}(x_j) \overset{C}{\geq} f_m(x_j) > f_n(x_j), & i \neq j, f_m(x_j) > f_n(x_j) \\ f_m(x_j) + \alpha \cdot f_{x_i,\delta}(x_j) \overset{2.}{<} f_m(x_j) + d^* < f_n(x_j), & i \neq j, f_m(x_j) < f_n(x_j) \\ f_m(x_i) + \alpha \cdot f_{x_i,\delta}(x_i) = 2f_n(x_i) - f_m(x_i) > f_n(x_i), & i = j \end{cases}$$

$$f'_m(x_L) = f_m(x_L) + \alpha \cdot f_{x_i,\delta}(x_L) < f_m(x_L) + d < f_m(x_L) + \frac{\alpha}{4} < f_n(x_L)$$

$$f'_m(x_R) = f_m(x_R) + \alpha \cdot f_{x_i,\delta}(x_R) < f_m(x_R) + d < f_m(x_R) + \frac{\alpha}{4} < f_n(x_R).$$

Therefore, f'_m and f_n alternate on this set.

Induction step: $(m, n) \to (m, n+1)$

This step can be proven analogously by exchanging f_m and f_n in the previous step. □

Using this powerful theorem, we now only need to show that the families of functions $\mathcal{F} \in \{Lor, Gau, pVoi\}$ fulfill all of the conditions of Thm. 3.22. For pseudo-Voigt functions we can define $f_{0,1} = pV_{0,1,0}$, representing Lorentzians. Both Lorentzians and Gaussians trivially fulfill conditions 1.-3. Therefore, Thm. 3.22 yields

$$k_L(m, n) \geq 2m + 2n - 2$$
$$k_G(m, n) \geq 2m + 2n - 2$$
$$k_{pV}(m, n) \geq 2m + 2n - 2.$$

In the case of Voigt functions with sufficiently small parameters $v(\varepsilon)$, we know that for any finite set of points $x \in X$ and for all $\varepsilon > 0$, we have $|V_{c,v(\varepsilon),w}(x) - L_{c,w}(x)| < \varepsilon$, similar to the proof of Lem. 3.18. Therefore, the construction of new peaks used in the proof of Thm. 3.22 can also be used for Voigt functions with sufficiently small $v > 0$ and we have

$$k_V(m, n) \geq 2m + 2n - 2.$$

Although this inductive proof is sufficient, it is possible to construct sums of m and n model functions such that the number of intersections is at least $2m + 2n - 2$. This construction, however, is inelegant and requires additional conditions for the set of model functions. It can be found in App. A.2.

3.2.2 Upper Bounds for $k_{\mathcal{F}}(m, n)$

While the lower bounds are useful in determining the number of points where ambiguous decompositions can be constructed, ensuring uniqueness requires obtaining upper bounds for $k_{\mathcal{F}}(m, n)$. We examine the behaviour of our four model functions in the following sections.

3.2.3 Lorentz Functions

We are interested in the maximum number of intersections of two sums of Lorentzians. Therefore, assume that there are $m, n \in \mathbb{N}$, parameters $\alpha_i, (c_i, v_i)$, and $\beta_j, (d_j, w_j)$ defining the corresponding Lorentz functions. The equation

$$\sum_{i=1}^{m} \alpha_i \frac{v_i^2}{v_i^2 + (x - c_i)^2} = \sum_{j=1}^{m} \beta_j \frac{w_j^2}{w_j^2 + (x - d_j)^2}$$

can be transformed into a polynomial equation by multiplying by each denominator. The degrees of the resulting polynomials are exactly $(2m + 2n - 2)$. A polynomial equation of degree $(2m + 2n - 2)$ can have at most $(2m + 2n - 2)$ solutions. Therefore, we have

$$k_L(m, n) \leq 2m + 2n - 2.$$

3.2.4 Gauss Functions

The situation is more complex for Gaussians. We cannot easily transform the equation

$$\sum_{i=1}^{m} \alpha_i \exp\left(-\ln(2)\frac{(x - c_i)^2}{v_i^2}\right) = \sum_{j=1}^{n} \beta_j \exp\left(-\ln(2)\frac{(x - d_j)^2}{w_j^2}\right) \tag{3.25}$$

into a form in which a maximum number of solutions can be estimated. To accomplish this, we eliminate each of the $m + n$ summands one by one in an iterative procedure.

Lemma 3.23. *Let $m, n \in \mathbb{N}$. Then,*

$$k_G(m, n) \leq 2^{m+n} - 2.$$

Remark 3.24. While writing this thesis, we realized that this method of proving the upper bounds, particularly the iterative use of ROLLE'S theorem, is already known in the literature, e.g., in [61, Prop. (Laguerre)], and even earlier, in [130, p. 421], where it is used to estimate the number of solutions to equations containing *exponential polynomials*, see [125, Eq. (1)] for a general definition. Even earlier, a similar problem was posed to students by PÓLYA and SZEGŐ, see [99, Prob. 77]. Unfortunately, equations containing Gaussian sums, such as in Eq. (3.25), are not covered by the theory of exponential polynomials.

Proof. The proof of this lemma is conducted by iteratively eliminating each summand until we reach an equation, where the number of solutions is easily identifiable.

We can rearrange Eq. (3.25) to obtain

$$\sum_{i=1}^{m+n} p_{d_0,i}^{(0)}(x) \cdot \exp(q_i^{(0)}(x)) = 0. \tag{3.26}$$

In Eq. (3.26), the upper index (k) counts the number of iterations performed by our procedure. The term $p_{d_k,i}(x)$ denotes polynomials of degree $\leq d_k$, and $q_i(x)$ denotes polynomials of degree ≤ 2 for $i = 1, \ldots, m + n$. In each step of the procedure, we divide the sum by one of the exponential functions. Since $\exp(q_i(x)) \cdot \exp(-q_j(x)) = \exp(q^*(x))$, we end up with another exponential of a polynomial of at most degree 2. We thereby remove the exponential term of one summand using this method. This summand then only consists of a degree d polynomial. In the second part of the iteration step, we take $d + 1$ derivatives of the sum. Thus, this summand vanishes, while retaining the structure of all the other summands. Note

that

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x} p_{d,i}(x)\exp(q_i(x)) &= (p'_{d,i}(x) + p_{d,i}(x)\cdot q'_i(x))\exp(q_i(x)) \\ &= p^*_{d+1,i}(x)\exp(q_i(x)).\end{aligned}$$

Thus, a single derivative increases the degree of the polynomial factors by one.

For the k-th iteration step we have eliminated $k-1$ summands so far, and the procedure is as follows.

$$\begin{aligned}\sum_{i=k}^{m+n} p^{(k-1)}_{d_{k-1},i}(x)\cdot\exp(q_i^{(k-1)}(x)) &= 0 \qquad \Big| \cdot\exp(-q_k^{(k-1)}(x)) \\ p^{(k-1)}_{d_{k-1},i}(x) + \sum_{i=k+1}^{m+n} p^{(k-1)}_{d_{k-1},i}(x)\cdot\exp(q_i^{(k)}(x)) &= 0 \qquad \Big| \left(\frac{\mathrm{d}}{\mathrm{d}x}\right)^{d_{k-1}+1} \\ \sum_{i=k+1}^{m+n} p^{(k)}_{2\cdot d_{k-1}+1,i}(x)\cdot\exp(q_i^{(k)}(x)) &= 0.\end{aligned}$$

Note that, at each step, the degrees behave as follows: $d_k = 2\cdot d_{k-1}+1$. Since $d_0 = 0$, a simple induction implies that $d_k = 2^{k-1}-1$. Therefore 2^{k-1} derivatives are performed in step k.

After $m+n-1$ steps, we obtain

$$p^{(m+n-1)}_{d_{m+n-1},m+n}(x)\cdot\exp(q_{m+n}^{m+n-1}(x)) = 0. \tag{3.27}$$

The function on the left can have at most $d_{m+n-1} = 2^{m+n-1}-1$ roots. Note that the polynomial cannot equal the zero polynomial, since all derivatives increased its degree by exactly one and $p^{(0)}_{0,m+n} \neq 0$ because $\alpha_i \neq 0 \neq \beta_j$.

For the second part, we use ROLLE's theorem [13, Thm. 6.2.3], which states that between two roots of some differentiable function f, there exists at least one root of its derivative f'. Conversely, if the number of roots of f' can be estimated to be $\leq n$, then f cannot have more than $n+1$ roots.

Therefore, the original function on the left of Eq. (3.26) can have at most $2^{m+n-1}-1$ roots, plus the total number of derivatives performed throughout the iteration. Therefore, the original function has at most

$$\begin{aligned}\sum_{k=1}^{m+n-1}(d_k+1) + 2^{m+n-1} - 1 &= \sum_{i=0}^{m+n-1} 2^i - 1 \\ &= 2^{m+n} - 2\end{aligned}$$

roots, and the proof is complete. □

3.2.5 Pseudo-Voigt Functions

Algebraic equations containing pseudo-Voigt functions are even more complex than those containing only Gaussians. Since there are more degrees of freedom to a single pseudo-Voigt functions, we expect a larger number of possible roots. Still, the same iterative procedure as in the proof of Lem. 3.23 can be used.

Lemma 3.25. *Let $m, n \in \mathbb{N}$. Then,*

$$k_{pV}(m,n) \leq (2^{m+n+2}-2)(m+n) - 2.$$

Proof. After some initial differences, we use the same iterative procedure as in Lem. 3.23. We are again interested in the number of solutions to the equation

$$\sum_{i=1}^{m+n} \alpha_i pV_{c_i,w_i,\lambda_i}(x) = 0 \tag{3.28}$$
$$\sum_{i=1}^{m+n} \alpha_i \left(\lambda_i \exp\left(-\ln(2) \frac{(x-c_i)^2}{w_i^2} \right) + (1-\lambda) \frac{w_i^2}{w_i^2 + (x-c_i)^2} \right) = 0.$$

We then multiply both sides by the denominators $\prod_{i=1}^{m+n}(w_i^2 + (x-c_i)^2)$, and rewrite the result with similar nomenclature

$$\sum_{i=1}^{m+n} p_i^*(x) \exp\left(q_i^{(0)}(x)\right) + r(x) = 0.$$

The polynomial resulting from all Lorentzians $r(x)$ has degree $2m + 2m - 2$. The polynomials p_i^* have degree $2m + 2n$. Now, we derive both sides $2m + 2n - 1$ times to eliminate $r(x)$. Finally, we obtain the equation

$$\sum_{i=1}^{m+n} p_{d_0,i}^{(0)}(x) \exp\left(q_i^{(0)}(x)\right) = 0.$$

Performing $m + n - 1$ steps of the iteration from Lem. 3.23 leads us to

$$p_{d_{m+n-1},m+n}^{(m+n-1)}(x) \exp\left(q_{m+n}^{(m+n-1)}(x)\right) = 0,$$

which has at most d_{m+n-1} many roots.

Next, using ROLLE's theorem, we know that the maximum number of roots of Eq. (3.28) can be bounded by the number of derivatives used in the procedure, plus the number of derivatives used during initialization, plus the number of solutions of the last equation, or by

$$k_{pV}(m,n) \leq \sum_{k=1}^{m+n-1} d_{k-1} + 1 + (2m + 2n - 1) + d_{m+n-1}.$$

The recursion $d_k = 2d_{k-1} + 1$ with $d_0 = 4m + 4n - 1$ leads to the explicit representation $d_k = (m+n) \cdot 2^{k+2} - 1$. Finally, we have

$$\begin{aligned} k_{pV}(m,n) &\leq \sum_{k=1}^{m+n-1} \left[(m+n)2^{k+1} \right] + (m+n)2^{m+n+1} - 1 + 2(m+n) - 1 \\ &= 2(m+n) \left[\sum_{k=0}^{m+n} 2^k \right] - 2 = (m+n)[2^{m+n+2} - 2] - 2. \end{aligned}$$

□

3.2.6 Voigt Functions

It remains an open question, whether $k_V(m,n) < \infty$. We have not yet found a way to manipulate Voigt functions algebraically that allows us to find an upper bound.

$m+n$	2	3	4	5	6	7	8	9	10
$n=1$	2	4	6	8	10^*	12^*	14^*	14^*	14^*
$n=2$	/	/	6	8	10^*	12^*	12^*	12	16
$n=3$	/	/	/	/	10	12^*	12^*	14	14^*
$n=4$	/	/	/	/	/	/	12^*	14^*	14^*
$n=5$	/	/	/	/	/	/	/	/	14

TABLE 3.1: Maximum number of intersections of randomly defined Gaussian sums with m and n summands.

3.2.7 Tightness of the Boundaries

Now that lower and upper bounds have been established, the question arises as to whether these bounds are tight. Obviously,

$$k_L(m,n) = 2m + 2n - 2$$

is tight for all tuples $(m,n) \in \mathbb{N}^2$, since the lower and upper bounds coincide.

For m and n Gaussians, we know that at least $2m + 2n - 2$ intersections are possible. In the base case of $m = n = 1$, the maximum number of intersections can be algebraically determined as 2. Also both lower and upper bounds are equal, since $2 + 2 - 2 = 2^2 - 2$.

By choosing random parameters, we can at least numerically hint at whether the bound is tight. For every combination of m, n in Tab. 3.1, we tested for 10^7 and 10^8 random Gaussian sums[8]. Numbers k^* with an asterisk indicate that $k-1$ intersections were found, and that one intersection is outside of $[-1,1]$. Parameters are defined by a uniform distribution on certain intervals, e.g., $c_i \sim \mathcal{U}_{[-1,1]}$.

As shown in Tab. 3.1, the number of intersections never exceeds $2m + 2n - 2$ for our tests. For larger values of $n + m$, the proven lower bound of the maximum number of intersections was not even achieved. For a construction with at least $2m + 2n - 2$ intersections, see App. A.2. This indicates that the lower bound is tight for all m, n.

Conjecture 1. *For $m, n \in \mathbb{N}$, we have*

$$k_G(m,n) = 2m + 2n - 2.$$

However, proving this statement would require handling sums of exponential functions algebraically, which is not an easy task. Even in the case of $m \in \mathbb{N}, n = 1$, it is difficult to prove that m Gaussians intersect another Gaussian no more than $2m$ times. The simpler case of $n = 1$ could be proven by showing that m Gaussian functions have at most m maxima, since the equation

$$\alpha_0 G_{c_0,w_0} = \sum_{i=1}^{m} \alpha_i G_{c_i,w_i}$$

can be transformed into

$$1 = \sum_{i=1}^{m} \frac{\alpha_i}{\alpha_0} G_{c_i',w_i'}, \tag{3.29}$$

[8]From $n + m = 7$ onwards, 10^8 random sums were used.

if $w_0 > w_i$. Nevertheless, proving that a (positive) linear combination of Gaussians has at most m maxima is not trivial. In fact, there is some literature surrounding these kinds of problems. First, a book by KHOVANSKIJ in which the author describes an upper bound for the number of solutions of systems of equations containing polynomials and linear exponential functions [62], however, it does not touch upon quadratic exponentials. Second, a work in which the number of modes of a sum of d-dimensional Gaussians is estimated [8]. The latter work also proposes a conjecture that, for m one-dimensional Gaussians, estimates the maximum number of modes as m. If this conjecture is correct, it implies that the maximum number of solutions to Eq. (3.29) is $\leq 2m$, which would confirm Conj. 1 in the case of $n = 1$. The proven upper bound, as described in [8, Cor. 5.1], yields in our case, that $k_G(m, 1) \leq 2^{1/2 \cdot m^2 + 5/2 \cdot m + 2}$, which exceeds our upper bound. In the case of $m = 2, n = 1$, however, [103, Thm. 1] improves our upper bound of $2^{2+1} - 2 = 6$, to two modes, or four intersections, which coincides with our lower bound.

Finally, it came to our attention that in one dimension, this problem is claimed to be proven in [23, Thm. 2]. There, using an induction on the number of Gaussians, it is claimed that a convolution of any integrable function, or distribution, with a Gaussian kernel does not create additional maxima. However, since a proof of this convolution property is only described loosely, it is unclear to us, whether it is true, especially, since a similar conjecture was mistakenly proven for the two-dimensional case, and was later disproven. Possible proofs of the convolution property can be found in [11, 65, 109, 134]. However, if it is true, proving Conj. 1 only requires recapitulating the proofs from [23] for possibly negative sums of Gaussians, which is done in App. A.

When attempting to falsify Conj. 1, we can consider an optimization problem in which some function f describing the number of intersections of two sums of Gaussians is to be maximized. For this purpose, we use the function `fminsearch` [55], as is implemented in `Matlab`, which allows us to optimize f, without using derivatives, since the derivatives of f are zero almost everywhere. To prevent premature convergence, we use a random method to obtain a suitable initial condition, where some intersections are already found and further appended `fminsearch` for our purposes. Still, this was not enough to improve the number of intersections found in Tab. 3.1.

In the case of pseudo-Voigt functions, the lower bound $2m + 2n - 2$ is not tight. The smallest example occurs for $m = n = 1$, with at least six solutions for

$$pV_{0,1,1/2}(x) = \frac{13}{25} pV_{0,2,3/4}(x)$$

see [52, App. C.4] for the proof.

However, it is also unreasonable to assume that the upper bound of pseudo-Voigt functions is tight, since it would imply the existence of two pseudo-Voigt functions that intersect exactly $(2^{1+1+2} - 2)(1 + 1) - 2 = 26$ times.

Since these upper bounds grow quickly, their practical use is limited. Suppose an NMR spectrum s contains 20 peaks. To prevent overfitting, no more than 25 model functions are used. If the underlying functions are perfect Lorentzians, the spectrum only needs to contain at least $k_L(25, 20) + 1 = 89$ points to guarantee uniqueness. For perfect Gaussians on the other hand, a minimum number of $k_G(25, 20) + 1 \leq 2^{45} - 1 \approx 35.2 \cdot 10^{12}$ points is required. In the case of pseudo-Voigt functions the number of required points is even larger at $\approx 6.3332 \cdot 10^{15}$. Conversely, if a spectrum is measured at 100,000 points, then $m + n$, the sum of the number of model functions and peaks in the spectrum may not exceed 16 Gaussians, or 12 pseudo-Voigt functions to ensure unique decompositions.

3.3 Uniqueness and Noisy Data

So far, we have only considered the modeling problem to be successful if, for all points $x_0, \ldots, x_N$, the obtained model recreates the spectrum exactly at a given set of points X. Usually, a more lenient approach is considered. However, hopes for unique decompositions quickly fade, when introducing an accuracy threshold $\varepsilon > 0$. Not only is it possible to choose another model with parameters that differ only slightly from the first model, but even for greatly different model parameters, we can recreate NMR spectra, if for all points x_i we only require that the model $m(x_i)$ fits the spectrum $s(x_i)$ up to this threshold $\varepsilon > 0$.

Example 3.26. Let $\varepsilon = 0.005$. Consider the two models created by four pseudo-Voigt functions, which are displayed in Fig. 3.4. The two models have completely different sets of parameters. However, their maximum deviation is $\approx 0.0026 < \varepsilon$. Therefore, both sums solve the modeling problem, if the given spectrum were equal to the first sum of pseudo-Voigt functions. If noise levels were estimated to be up to 0.5% of the maximum amplitude, identifying the correct model would be impossible by looking at the spectrum alone.

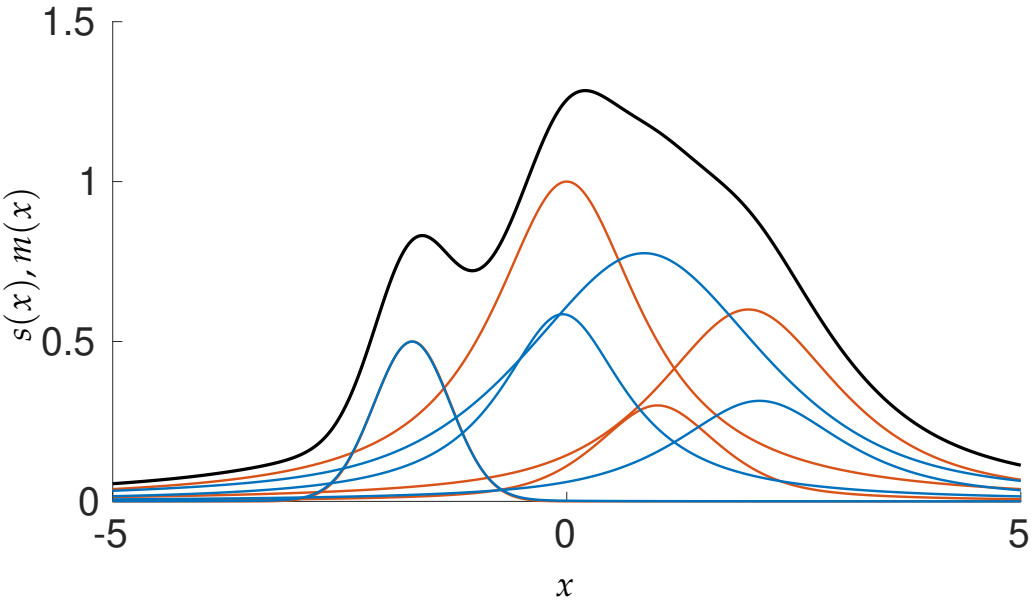

FIGURE 3.4: Two sets of pseudo-Voigt functions (red, blue) with different parameters but with similar sums (black).

3.4 Conclusion

Models of NMR spectral data are useful for gaining a deeper understanding of the underlying chemical compounds, their structures, and their concentrations. Knowledge about the ambiguity or uniqueness of these models is necessary, for example, to verify whether the obtained structural information is indeed the only possible conclusion, or to acknowledge different feasible solutions for fitting concentration profiles and reaction kinetics.

In this chapter, our initial aim is to answer the four questions on page 17. Regarding the first question, we are able to prove that all of the commonly used model functions are indeed linearly independent. This implies unique solutions to the modeling problem over a continuous frequency axis, which contradicts the general understanding of hard modeling

in the literature. There, the practical approach of fitting models as closely as possible is considered, which introduces intrinsic ambiguity. Therefore, the fact that the problem has unique solutions without noise may be surprising. However, the existence and uniqueness of decompositions does not imply a simple and direct method of obtaining them.

To address the second and third questions, for a discrete spectrum, the solutions are only unique, if the number of model functions is sufficiently small, or if the number of points is sufficiently large. In this context, we introduce the number $k_{\mathcal{F}}(m, n)$, which denotes the maximum number of points for which a decomposition of a spectrum into m and n different sets of model functions exists. We prove a lower bound for $k_{\mathcal{F}}(m, n)$, for general types of model functions in Thm. 3.22, and upper bounds for Lorentzians, Gaussians, and pseudo-Voigt functions in Thms. 3.23 and 3.25. The lower bound is proven to be tight for Lorentzians, and we propose a conjecture that this also holds for Gaussians. This conjecture is supported by different results in the literature and by our own numerical experiments. Regarding the last question, the noisy case always implies ambiguous solutions. In many cases, there even exist solutions that are far apart in parameter space, see Ex. 3.26. Since our methods of proof involve observing the asymptotic behavior of spectra, and are not constructive, they cannot be applied to modeling experimental NMR spectra. To obtain models for NMR spectra numerically and computationally, we need to establish different methods.

Chapter 4

Obtaining Parameters by Nonlinear Optimization

Typically, detemining the parameters for a suitable model of NMR spectra involves nonlinear optimization. We first examine possible approaches to this optimization problem.

The simplest method of modeling NMR spectra is to model each peak using peak model functions. This type of peak fitting has been used for more than half a century [101, 129] and is still in use today [14, 79]. An overview of the literature on this approach can also be found at the beginning of Ch. 2.

An alternative modeling method is the *indirect hard modeling* (IHM). First described by ALSMEYER, KRIESTEN, et al. [7, 69, 70], this approach applies chemical information and uses pure component spectra to obtain models of the mixture spectra. Unlike to the approach of this thesis, which they call *direct hard modeling*, the peaks corresponding to pure component spectra are not treated separately, but rather as a whole. Additionally, it is assumed that the parameters of the mixture spectra only vary within certain ranges from the parameters of the pure component spectra. These assumptions allow for a well-posed initialization of the optimizer. Optimization occurs in multiple subsystems throughout the spectrum.

A method similar to IHM using Bayesian statistics was proposed by MATVIYCHUK, VON HARBOU, et al. [71, 83, 84]. Again, mixture spectra are modeled using prior knowledge of the pure component spectra. In this case, the Bayes estimator is used for the concentration profiles, assuming that the mixture spectrum is a linear combination of the pure component spectra. Using the Bayes method also works for time series of NMR spectra, since the optimized results from previous spectra improve the accuracy of the Bayes estimator for subsequent spectra.

A third approach, called *Magnetstein* by DOMŻAŁ, KAZIMIERCZUK, et al. [31], aims to optimize the concentration profiles of chemical compounds using mixture spectra. This method assumes that the mixture spectrum is close to a linear combination of the pure component spectra. The linear combintion closest to the mixture spectrum w.r.t. the Wasserstein distance is then chosen to determine the concentrations. This method also allows for reaction monitoring [32].

In another approach, KUPČE, KAZMIERCZUK, et al. [26, 73], propose using the *Radon transform* to detect linear changes in peak positions throughout NMR time series, thereby increasing the selectivity of the data and facilitating the optimization of single-peak models.

Lastly, an approach using neural networks for quantitative analysis of NMR time series was proposed by KERN, MAIWALD, et al. [59]. For more details on applications of machine learning in the context of NMR spectroscopy, we refer to Sec. 5.1.3.

Unlike other existing approaches, our two main methods of modeling individual peaks do not require any prior knowledge, such as the number of chemical compounds or a database of pure component spectra. Our methods are also not limited to analyzing single spectra, but they excel when dealing with time series of connected spectra, since the second dimension provides additional information on each individual spectrum. The main idea of

both approaches is to not optimize all parameters on every spectrum. Instead, we propose a spline-based optimization routine that only optimizes parameters on selected spectra. The remaining parameters can be obtained by interpolating the optimized values with cubic spline functions. Not only does this reduce the dimension of the underlying optimization problem, it also forces the parameters to behave smoothly over time. As we established in Ch. 3, ambiguous solutions can be found in noisy data, especially when dealing with peak overlaps. The forced smoothness mitigates this ambiguity, since information provided by the second dimension, such as movements of peak positions, helps to identify the true solution. Furthermore, this stabilizes the optimization routine.

In Sec. 4.1 we first discuss the advantages of the spline-based approach and other improvements to the optimization. In Sec. 4.2, we describe the spline-based nonlinear optimization algorithm that is used in our first and second approaches to the modeling problem in great detail. We consider several variants to this optimization problem to efficiently obtain hard models of NMR spectral time series. Lastly, we discuss the possibility of using a different objective function with the spline-based optimizer.

The spline-based optimization method was first published in [86].

4.1 Efficient Optimization Strategies

For obtaining the model parameters, applying a nonlinear optimization algorithm alone does not capture the entirety of the problem. Most optimizers require a fixed number of parameters, which in our case is dependent on the number of model functions m. Therefore, approaching the modeling problem using nonlinear optimization requires splitting the problem into two parts. First, the number of peaks must be obtained. Then, the number of parameters is known, and the corresponding parameters can be optimized. The number of peaks m plays a crucial role in the success of the optimization, since too few peaks result in an incomplete model, while too many peaks run the risk of *overfitting*, i.e., fitting peaks with multiple functions, or fitting noise and other interferences.

Note that there are algorithms capable of handling the mixed discrete and continuous nature of optimizing m and all p_i simultaneously, see for example [75]. However, if we assume that m is known during optimization, then the error function is usually continuous, and differentiable almost everywhere. This enables the use of derivative-based optimization algorithms.

We choose the Frobenius norm $\|\cdot\|_F$ as our main objective function in Def. 2.3. Then, as discussed in Sec. 2.3, the goal of the optimization is to minimize

$$\|D - M_{\mathcal{P}}\|_F^2 = \sum_{t=1}^{|T|} \sum_{x=1}^{|X|} \left(D(t,x) - M_{\mathcal{P}}(t,x)\right)^2 \tag{4.1}$$

over all possible sets of parameters $\mathcal{P}$. In this section, we discuss how to efficiently and accurately optimize peak parameters. Crucially, this involves avoiding high-dimensional optimization routines, while simultaneously preserving the ability to obtain meaningful models.

4.1.1 The Spine-Based Approach

Optimizing all parameters at once is very inefficient. For example, assume that ten peaks are modeled using pseudo-Voigt functions in 100 spectra. The optimization routine would need to optimize 40,000 parameters simultaneously. Not only is this computationally expensive, but it is also unstable, as gradient-based optimizers tend to optimize the areas where the

objective function is largest, and neglect areas with smaller contributions. This can lead to premature termination or convergence to one of many local minima.

Conversely, it is also possible to optimize spectra sequentially. While this approach may seem promising for reducing the runtime, it can negatively impact the optimization stability and model quality, especially when peaks overlap or cross. This effect can be observed when optimizing the center and height parameters of data set D_0, as shown in Fig. 4.1. During the sequential optimization method, non-smooth peak tracing leads to jumps in peak heights. The ground truth parameters used to generate D_a are denoted by the colored markers.

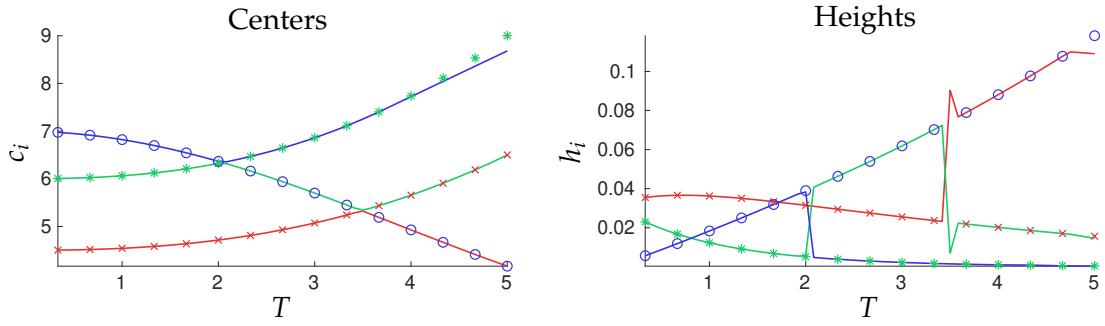

FIGURE 4.1: Sequential optimization of centers (left) and heights (right).

Our proposed approach involves the simultaneous optimization of a few selected spectra within a moving window through time. To further decrease the number of parameters per optimization, we assume that the optimal parameters, sometimes called *ground truth*, behave like smooth functions through time, assuming the spectra are measured in sufficiently quick succession. Using the smoothness, we are able to interpolate the parameter functions with cubic splines, which are defined in [120, Ch. 2.4].

Let $T_{sel} = \{t_{i_1}, \dots, t_{i_s}\} \subseteq T$ be a subset of spectra, e.g., every tenth spectrum. Then, we optimize in a window of size[9] $len_w = \ell$, ranging from spectrum t_{i_j} to spectrum $t_{i_{j+\ell}}$. During this optimization, the parameters of all spectra inside the window, i.e., parameters of $\{t_{i_j}, t_{i_{j+1}}, \dots, t_{i_{j+\ell}}\}$, are optimized. Usually, this limits the number of parameters to 200. In our example, a moving window consisting of four spectra from T_{sel} results in a total number of 160 parameters being optimized simultaneously. However, the results of the previous optimizations are used to initzalize the next, so only 40 completely unoptimized parameters are added in each step. Then, the window moves along to the next set $\{t_{i_{j+1}}, \dots, t_{i_{j+\ell+1}}\}$, until all parameters of the selected spectra are optimized. The parameters of all other spectra are obtained via interpolation.

Advantages of the Spline-Based Method:

Compared to the simultaneous optimization of all parameters, this approach is extremely efficient, as only a few parameters are required to model the entire time series. This is not only a dimensionality reduction of the optimization problems, but also of the data itself, since the high-dimensional data matrix can be reconstructed using only a few parameters. Furthermore, when compared to the sequential approach, the moving window optimization ensures that the additional information provided by the second dimension is retained. The continuous and differentiable spline functions force the parameter functions to behave smoothly, which is especially helpful when peaks overlap or cross. Fig. 4.2 considers the same optimization problem as Fig. 4.1, but uses the spline approach to ensure smooth parameters and meaningful models.

[9] All hyperparameters appearing in this thesis can be found on page xi.

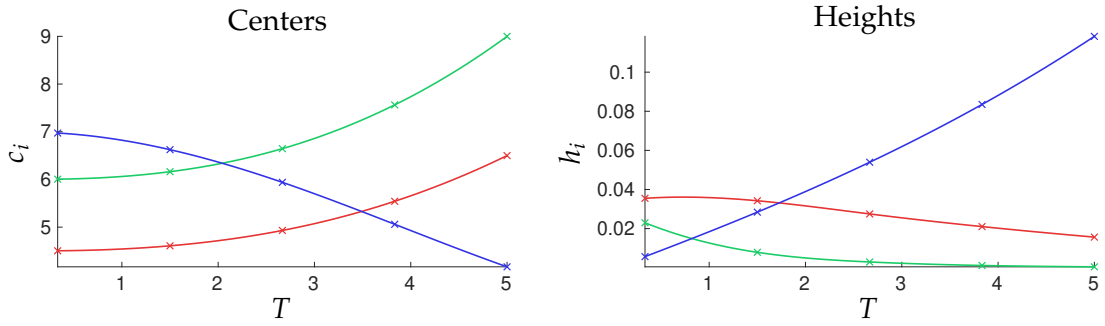

FIGURE 4.2: Spline-based optimization of centers (left) and heights (right).

4.1.2 Multiplets vs. Singlets

To further reduce the number of optimization parameters, we introduce multiplets. They are defined as coupled peaks with equidistant centers, identical half-widths. In the case of pseudo-Voigt functions, they possess an additional, identical convex combination parameter λ. Their heights are proportional to a row of binomial coefficients, see [80, Sec. 8.5].

Definition 4.1. *Let $d \in \mathbb{R}_+$ and $k \in \mathbb{N}$. Further, let $c \in \mathbb{R}, w \in \mathbb{R}_{>0}$, and $\lambda \in [0,1]$. Then, a* k-tuplet *consisting of model functions $f_{c,w(,\lambda)}$ is defined by*

$$m_{k,c,d,w(,\lambda)}(x) = \sum_{j=1}^{k} \binom{k}{j} f_{c+(j-1)d,w(,\lambda)}(x).$$

The parameter d represents the distance *between its* sub-peaks, *and k is called* multiplicity.

A spectrum consisting of m multiplets can be written as

$$s(x) = \sum_{i=1}^{m} \alpha_i m_{k_i,c_i,d_i,w_i(,\lambda_i)}(x),$$

where $\alpha_i \in \mathbb{R}$ correspond to the peak heights of the smallest sub-peaks. In the context of optimization, the heights are denoted by h_i and are incorporated into the parameter tuples p_i.

Note that a spectrum consisting of multiplets requires four (or five) parameters to be optimized for each multiplet: $(h, c, d, w, (\lambda))$. The multiplicity k only needs to be defined once and does not change over time. Therefore, fewer parameters are required to model a spectrum when using multiplets. Usually, this reduces the number of parameters by $40\% - 60\%$, depending on the data set. For instance, modeling all nine peaks of D_2 using separate pseudo-Voigt functions requires $36 = 9 \cdot 4$ parameters per spectrum. When using multiplets, this number can be reduced to $18 = 2 \cdot 5 + 2 \cdot 4$. In the case of D_3, the 23 peaks can be divided into two septuplets, three doublets and three singlets, reducing the $92 = 23 \cdot 4$ parameters to $37 = 5 \cdot 5 + 3 \cdot 4$ when using multiplets.

Since there are fewer possible choices for the smaller number of parameters, optimization using multiplets also decreases the likelihood of converging to a local minimum, which increases the optimization stability. Fig. 4.3 illustrates this fact for the optimization of the center parameter of a doublet versus the optimization of two center parameters of two singlets. When fitting the center parameter of the doublet, two local minima exist in addition to the global minimum, (blue line), whereas for the singlets, the optimum is not unique and there are four local minima and eight valleys with small derivatives. Note that if the optimizer could reliably start close to the global optimum, it would not only increase stability but also decrease the runtime.

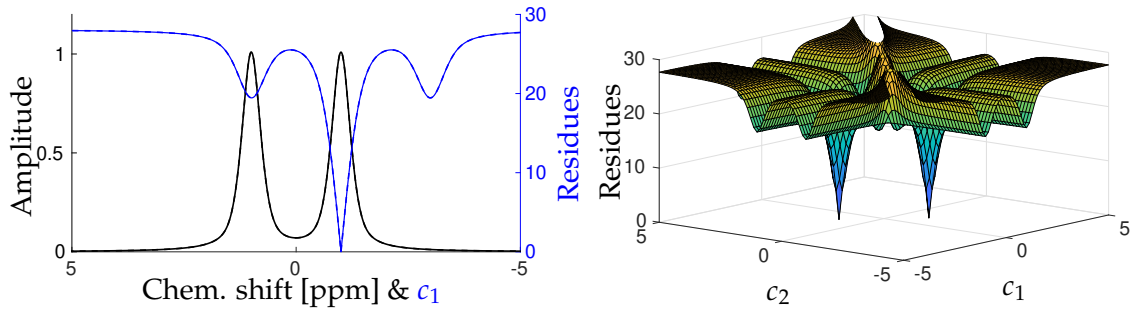

FIGURE 4.3: Residues for optimizing a doublet versus two single peaks.

However, the improved stability and runtime come at the expense of model quality, since a modeled multiplet must precisely match the height pattern. If, for any reason, a k-tuplet in the data does not perfectly fit this pattern, it can most accurately be modeled by a sum of k individual peaks. In the case of D_2, this is displayed in Fig. 6.3.

4.1.3 Localizing the Optimization Procedures

Despite improvements in runtime and complexity, the computational expenses remain high. There are two main reasons. First, the dimensions of the data matrices D are typically large. Naturally, the spectra are measured as accurately as is feasible. It would be counterintuitive to then decrease the number of data points or spectra afterwards. Nevertheless, this remains a possibility and still, a balance must be achieved between accuracy and runtime.

Second, the dimensions of the underlying optimization problems are large. In other words, we need to reduce the number of simultaneously optimized parameters. To combat large optimization routines, we suggest to split large problems into several problems of lower dimensions. Again, we need to strike a balance between slow and expensive, high-dimensional optimization problems, which allow a wider set of possible solutions and quick and cheap low-dimensional probems with a narrower set of possible solutions.

The general idea behind the optimization methods that aim to localize the optimization procedures is as follows: If the peak centers are already correctly defined, then modeling all the other parameters becomes much more convenient, see also Fig. 4.3. Therefore, we propose four variants of optimization, in which the peak centers are modeled first, before either the optimization problems are split or the data is partitioned. We discuss the optimization variants in Sec. 4.2.4 in greater detail.

4.1.4 Using the Wasserstein Distance as an Objective Function

An alternative to minimizing the Frobenius norm is to minimize the Wasserstein metric, which is defined in Def. 2.11. Recalling Rem. 2.15, we know that the Wasserstein metric is sensitive to changes in nonlinear parameters, but insensitive to changes in linear parameters. We can use this fact to our advantage and split the optimization of nonlinear and linear parameters. We mainly propose a two-step objective function. First, given a set of nonlinear parameters, we use linear least squares to compute the optimal linear parameters with respect to the sum of squares. Then, the Wasserstein distance for the set of parameters is computed. The nonlinear optimization thereby only occurs on the nonlinear parameters.

Using this metric offers several theoretical advantages. First, optimization is more stable since generally, there are fewer local minima. This fact is illustrated in Fig. 4.4, when center and half-width parameters of a triplet are optimized. The Frobenius norm yields

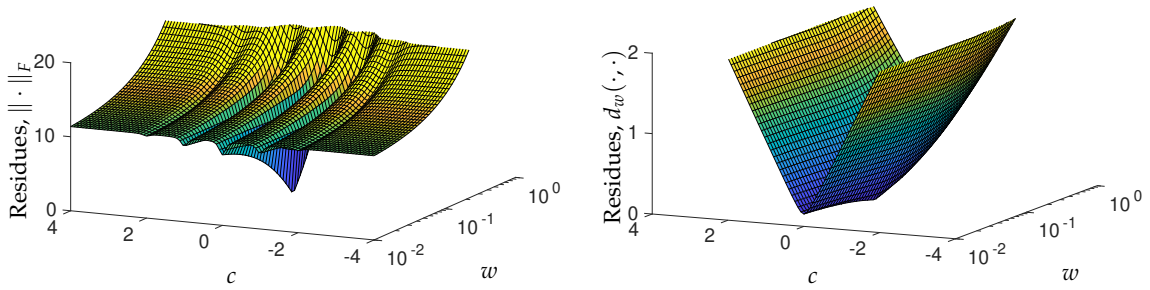

FIGURE 4.4: Residues for the Frobenius and Wasserstein objective functions for optimizing the center and half-width parameters of a triplet.

larger derivatives, but five local minima. Conversely, the Wasserstein-metric yields smaller derivatives, but a unique minimum.

Second, since the optimization of the linear parameters is done separately the dimension is reduced by 40% in the pseudo-Voigt case. To illustrate the separation, consider an NMR spectrum s consisting of m different pseudo-Voigt functions with $4m$ parameters $p_i = (c_i, w_i, h_i, \lambda_i), i = 1, \ldots, m$. For fixed values of the nonlinear parameters c_i, w_i the functions $L_{c_i,w_i}(x), G_{c_i,w_i}(x)$ are independent of h_i and λ_i. Finally, let $X = \{x_1, \ldots, x_N\}$. Then, fitting the spectrum to a given spectrum $D(:,t)$ requires approximately satisfying the following equations for all $x_j \in X$

$$s(x_j) = \sum_{i=1}^{m} \underbrace{h_i \cdot (1-\lambda_i)}_{:=a_i} L_{c_i,w_i}(x_j) + \underbrace{h_i \cdot \lambda_i}_{:=b_i} G_{c_i,w_i}(x_j) = d_j = D(x_j, t). \tag{4.2}$$

In cases where the data D is noisy, this linear system of equations cannot be solved. Instead, we can find values for a_i, b_i such that the sum of squared errors of all equations is minimal. This solution is unique and can be determined rather quickly and directly, e.g., using linear least squares [120, Thm. 4.8.1.3.].

From the values for a_i, b_i, we can then reconstruct the values of the heights $h_i = a_i + b_i$ and of $\lambda_i = \frac{b_i}{a_i+b_i}$.

Nevertheless, there are still practical disadvantages when using the Wasserstein-Metric, which are discussed in Sec. 6.6.2.

4.2 Implementing the Spline-Based Optimization Algorithm

In this section, we describe the first approach to modeling NMR time series, the spline-based algorithm. To illustrate the procedures, we use model data. Fig. 4.5 illustrates the optimization procedure for D_a within a moving window on the left as well as the resulting parameter functions on the right.

The algorithm has many hyperparameters, i.e., custom parameters that are fixed for the entire algorithm and that fulfill a meta-purpose. For example the bandwidth $bw \in \mathbb{N}$ defines various window sizes required during the algorithm. We define the most important hyperparameters in this section. An overview on all hyperparameters is provided on page xi.

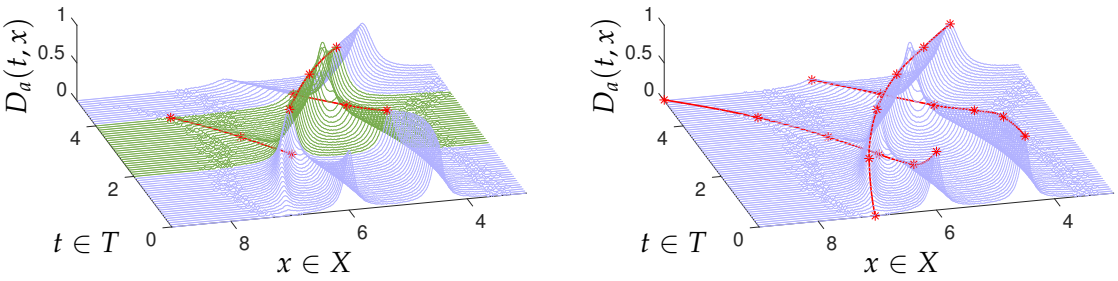

FIGURE 4.5: Windowed optimization for D_a (green rectangle). Parameters on $t \in T_{sel}$ (red stars) are optimized, all others are interpolated (red lines).

4.2.1 Initialization

Several steps need to be completed before optimization can begin. We omit the necessary steps to load the given data set in a suitable format for this thesis and assume that the data is given in matrix form with $|T|$ rows and $|X|$ columns, as well as two vectors representing the time axis T and the frequency axis X. We also assume that all spectra are phase- and baseline-corrected, which allows for a hard modeling approach in the first place.

The first step during initialization is to choose the set of spectra $T_{sel} \subseteq T$, that serves as the set of spline knots and on which the parameters are optimized. The naive approach is an equidistant choice on T. A manual selection is implemented, that can be performed by choosing spectra in a graphical user interface (GUI). Additionally, one of the spectra of T_{sel} is chosen as the starting spectrum t_{init}[10], i.e., the spectrum from which the optimization begins. The spectrum t_{init} should be chosen in such a way that peaks are sufficiently separated and that all peaks of the data set are present, and detectable. The advantage of manually selecting T_{sel} is that there may be intervals in time, where the parameters vary more and other intervals where they vary less. In areas of greater variation, it is advantageous to define more spline knots, whereas in areas of lesser variation, fewer spline knots can be chosen to decrease the runtime of the algorithm. Fig. 4.6 illustrates this for the moving peak of D_3. The equidistant choice is not sufficient to interpolate the actual peak movement. However, the same number of manually selected spline knots allows for an improved peak tracing by the cubic spline function.

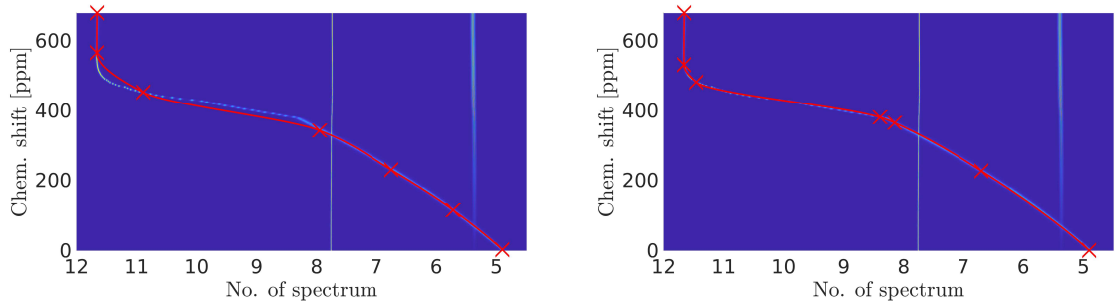

FIGURE 4.6: Choosing seven spectra of D_3 as spline knots (red crosses), equidistantly (left), manually (right), and the resulting peak tracing (red line).

[10]This is a short notation for the actual spectrum $D(t_{init}, :)$

4.2.2 Peak Detection

The second step of the initialization process is peak detection. Recall that we required t_{init} to contain all relevant peaks and that they are sufficiently separated. We then determine the peak positions on the first spectrum. First, the data set is smoothed using a Savitzky-Golay filter [113]. This method uses Gaussian least squares to fit a polynomial of degree deg to the data points within a moving window of size $2 \cdot bw + 1$. The polynomial evaluated at the midpoints of this windows defines the components of the smoothed data vector. One significant advantage of the Savitzky-Golay filter is that the polynomials provide easily computable derivatives, presenting a numerically stable approximation of the derivatives of the data vector. Fig. 4.7 presents the smoothing using a Savitzky-Golay filter with $deg = 3$ and $bw = 4$.

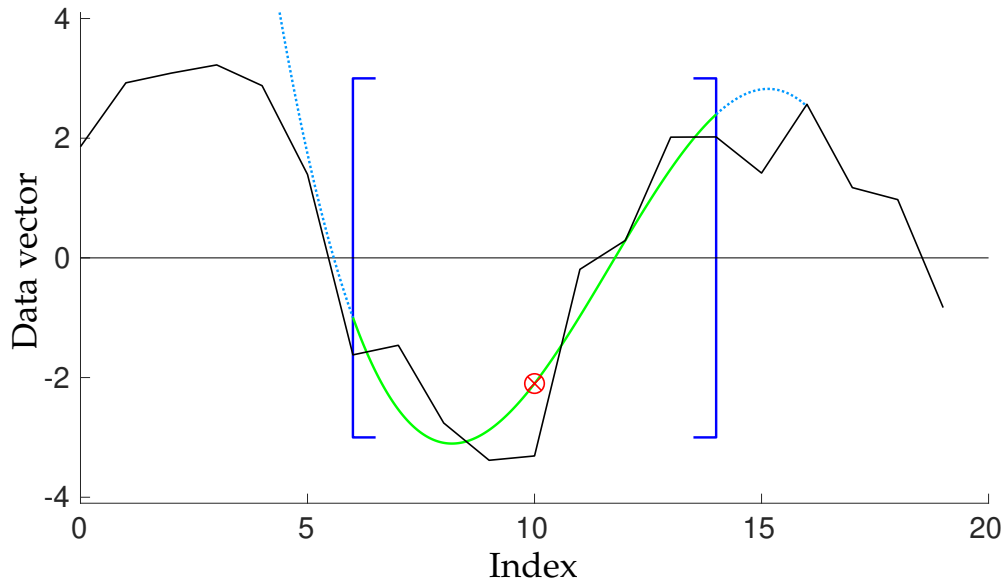

FIGURE 4.7: Savitzky-Golay filter using a cubic polynomial (green, light blue) inside a moving window (dark blue). The smoothed point is displayed in red.

Due to the nature of our four peak model functions, their centers are also minima of their second derivatives. However, the second derivative presents sharper negative peaks. See also Lem. 2.10. Second derivatives can also be used to detect peaks in close proximity, also known as *shoulder peaks*. Using fourth, or $2k$-th derivatives further amplifies this effect, i.e., negative peaks in the fourth derivative are even sharper. However, smoothing the data using the Savitzky-Golay method means fitting a low-dimensional polynomial to the data to avoid overfitting the noise. Usually, a cubic polynomial is used. Its fourth derivatives are zero everywhere and thus cannot be used. Still, higher-degree polynomials can be chosen to obtain higher-order derivatives, albeit at the cost of a worse smoothing effect. Fig. 4.8 shows the advantage of the second derivatives in situations where multiple peaks overlap. Using second derivatives facilitates the detection of the shoulder peak on the left.

The process of peak detection is illustrated in Fig. 4.9. First, local minima are detected within a moving window of size $2 \cdot bw + 1$. If local minima lie too close together, only the minimum with the largest absolute value inside of the window is considered a peak. If these local minima are too large, i.e., not far enough below a certain threshold $\delta < 0$, then they are also neglected. Due to noise, not all of these minima correspond to actual peaks.

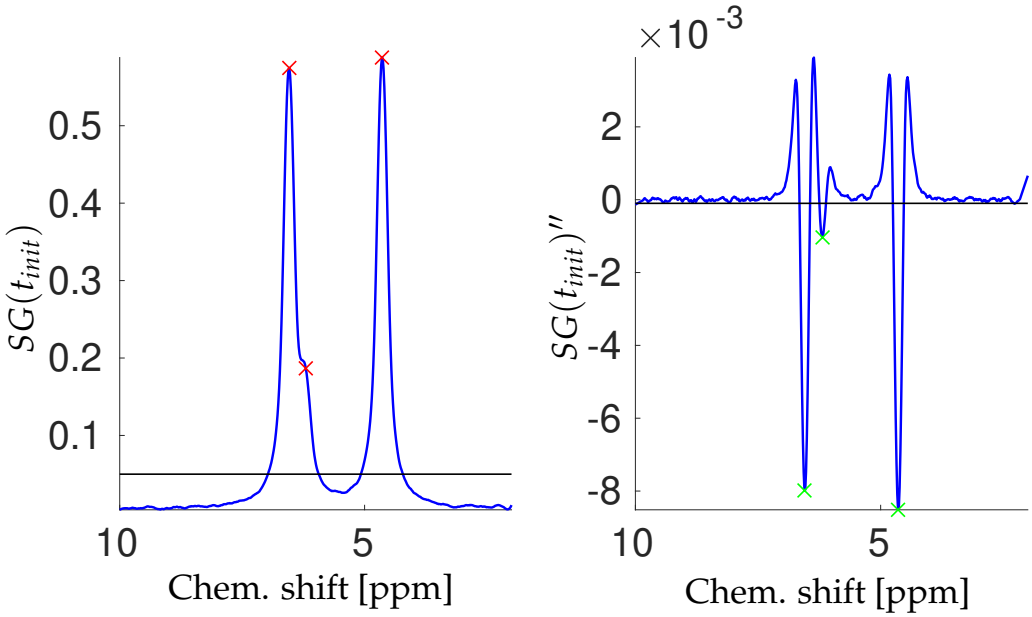

FIGURE 4.8: Smoothed data (left) and smoothed second derivatives (right). Peak positions are derived from minima on the right (red, green crosses).

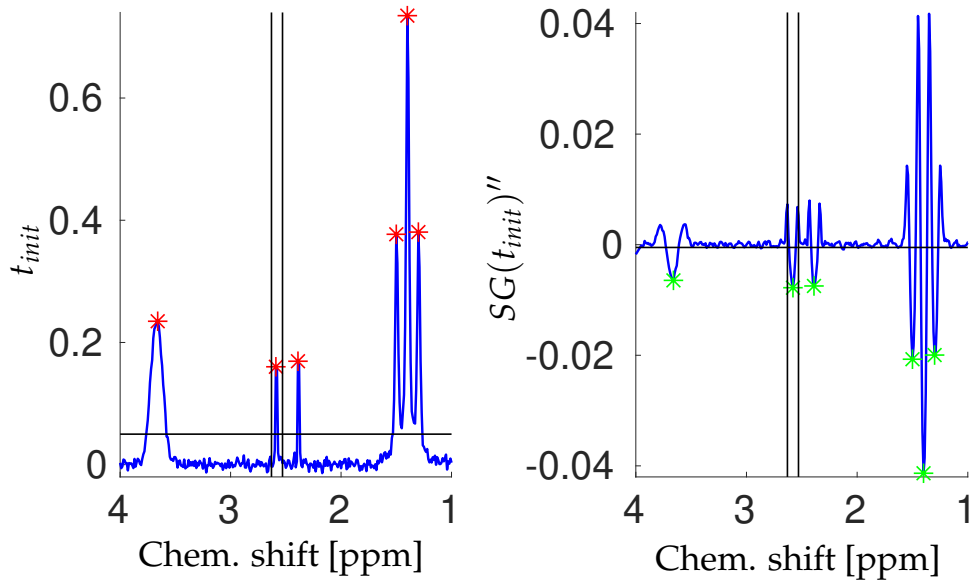

FIGURE 4.9: Peak detection procedure. Moving window and thresholds δ, ε are displayed by black lines. Crosses mark the detected peaks.

Second, we use the potential peak centers and determine if a peak exists at these positions in the original data vector. As it turns out, these points often lie slightly apart from the actual peak. We suspect that this is due to the discretization and the discrete approximation of the derivatives. To address this issue, we look for the closest peak, i.e., the closest local maximum, in the data vector. Potential duplicates are then eliminated. Additionally, peaks below a certain threshold $\varepsilon > 0$ are neglected.

4.2.3 Multiplet Detection

After detecting the peaks in the spectrum, the third step of the initialization is to decide where multiplets can be found in the data set. In other words, we need a robust clustering method to group the peaks into multiplets. Fig. 4.10 illustrates the procedure for the spectrum from Fig. 4.9. The main criterion in this method is the distance between peaks. Typically, peaks of multiplets lie relatively close together. Also, the peak heights follow a characteristic height pattern. We also assume that each peak belongs to only one multiplet.

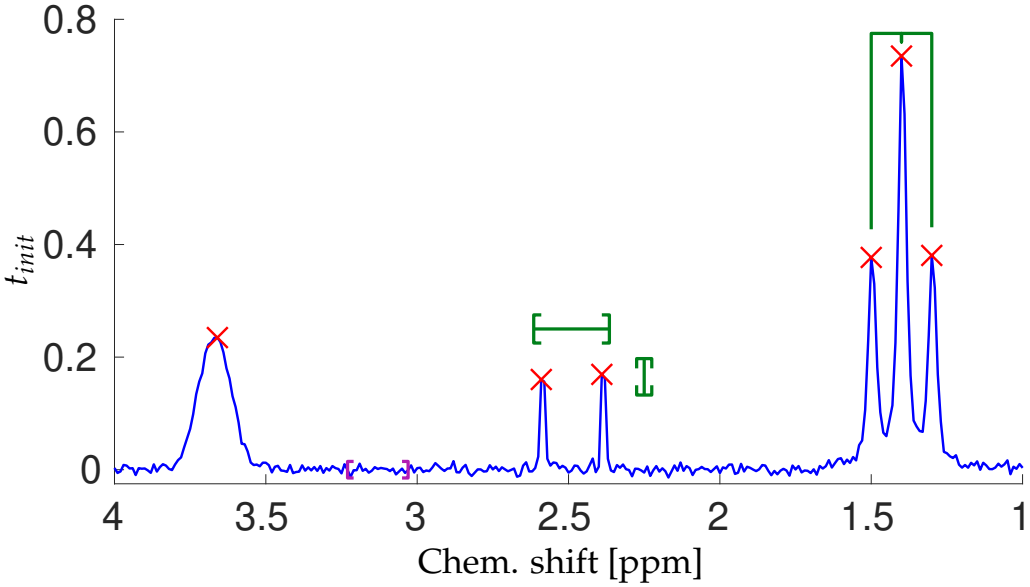

FIGURE 4.10: Multiplet detection for the spectrum from Fig. 4.9. Thresholds are displayed by green lines, noise is measured in violet window.

Similar to the heuristic peak detection method, we use another heuristic method to detect multiplets. First, we check for multiplets with a multiplicity of $k \geq 3$. Such multiplets have equidistant peaks. Therefore, we check for groups of three or more peaks whose distances are very similar, i.e., identical up to a predefined index threshold $tol_{ind} \in \mathbb{N}$, and smaller than $tol_{hor} > 0$. Before sorting the peaks of this group into a multiplet, we must test if the peak heights follow the characteristic pattern. To do so, we verify that the data vector evaluated at the peak centers is unimodal[11], while allowing for some noise. To measure the noise level, we estimate the variance of the data vector in an area, where presumably no peaks are expected using the square root of the unbiased sample variance, denoted by σ. In Fig. 4.10, this area is displayed in purple. Groups of peaks fulfilling the ascending, then descending pattern then qualify as multiplets of multiplicity $k \geq 3$.

[11]Only testing the similarity to a row of binomial coefficients is not a robust criterion for detecting slightly off-shape multiplets.

Second, we look for doublets ($k = 2$). If two of the remaining peaks are situated within tol_{hor}, we evaluate the data vector again at these points. If the values only differ by less than $3 \cdot \sigma + tol_{vert}$, the peaks qualify as a doublet. In Fig. 4.10, both tolerance values are displayed by the horizontal and vertical green intervals, respectively.

Third, all of the remaining peaks are then defined as singlets. However, to reduce the number of required parameters, we set their distance to $d = -1$ and avoid optimizing this parameter.

In the case of Fig. 4.10, the first three peaks from the right have distances of 10 and 10. In this case, if the peak heights of the triplet peaks are denoted by h_1, h_2, h_3, then the following inequalities must hold:

$$h_1 + 3 \cdot \sigma < h_2 \qquad h_2 > h_3 + 3 \cdot \sigma.$$

Indeed, h_1, h_2, h_3 fulfill these inequalities even without the addition of noise and a triplet is defined. For referece, the observed standard deviation of the noise is $\sigma \approx 0.0049$.

The other peak indices are all at different distances from each other, so no other k-tuplet with $k \geq 3$ can be found. Next, the two middle peaks are close together with a distance of $0.20 < tol_{hor} = 0.25$ and their corresponding heights are also similar, since

$$0.009 < tol_{vert} + 3 \cdot \sigma = 0.05 + 3 \cdot 0.0049.$$

Therefore, a doublet is detected. The remaining peak can only be a singlet.

4.2.4 Parameter Optimization

After all peaks have been detected and grouped into multiplets, the set of parameters is also defined. Then, the optimization procedure can begin. Initially, we optimize the parameters of the first selected spectrum t_{init}.

Optimization of t_{init}

Since all peak and multiplet positions are known, we can optimize each multiplet individually. To do so, we identify local optimization windows on the frequency axis.

For multiplets with multiplicity $k > 1$, we choose the corresponding window to encompass all subpeaks. If possible, both sides of the window are extended by the maximum of distance and bandwidth $\max\{d_i, bw\}$ from the centers of the outer subpeaks. This may encompass other singlets or multiplets. However, due to the characteristic shape of the multiplet, fitting a given k-tuplet into more complicated spectral shapes is comparatively easy.

If the spectrum contains only one peak, defining a local optimization window is unnecessary.

For the remaining singlets, we cannot use their distance parameters. Furthermore, fitting single peaks into complicated spectral shapes may result in poor models where several model peaks are fitted onto one spectral peak. Therefore, the window must encompass the peak, while still producing a meaningful result. Additionally, modeling one peak multiple times when windows overlap should be avoided. There are two cases. First, if one of the remaining peaks does not lie inside another multiplet's window, we define its window to extend to the midpoint between its and the next and previous peak's centers, but no further than $4 \cdot bw$. Second, if the peak lies inside another multiplet's window, we define its window to extend only bw in both directions.

Once all windows are defined, we optimize each set of parameters belonging to a peak or multiplet individually and locally within its respective window. After these initial optimizations, we finally optimize all parameters belonging to t_{init} simultaneously to obtain

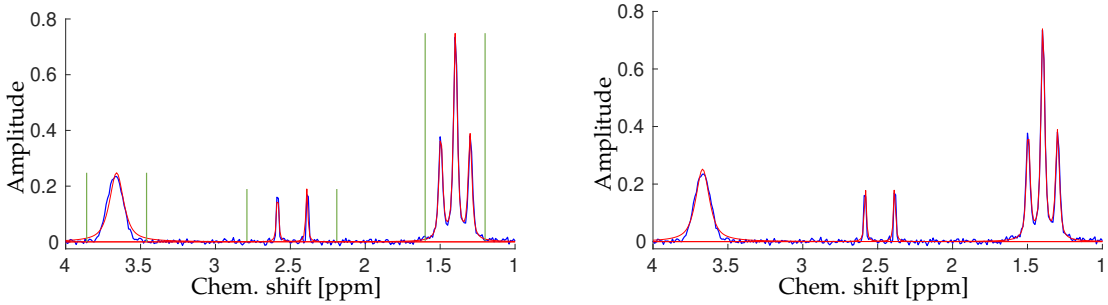

FIGURE 4.11: Initial optimization results for the spectrum from Fig. 4.9. Local optimization intervals are displayed by green vertical lines.

the initial model. To facilitate the global optimization, the local, windowed models serve as an improved initialization to the optimization routine for the entire spectrum. Otherwise, we run the risk of converging to a local minimum. Fig. 4.11 presents the local optimization procedure for the spectrum from Figs. 4.9 and 4.10 on the left and the final results on the right.

Optimization Variants

There are several ways to improve the runtime of the entire optimization procedure. To accomplish this, we must split the optimization into smaller subroutines while maintaining a meaningful chemical model. Initially, we define five different variants. The second approach, which is defined in Ch. 5, introduces two additional variants.

Variant 0: [Main Spline-Based Optimization]

In this main method, we optimize all parameters belonging to the selected spectra within a moving window through time, as initially described in this Sec. 4.1.1. Therefore, we minimize the objective function over all spectra inside this window. Parameters in between the selected spectra are interpolated using cubic spline functions, with one spline corresponding to one parameter. Fig. 4.5 illustrates this procedure for D_a and one step of its moving window optimization.

After one optimization subroutine finishes, the moving window advances to incorporate the next selected spectrum. The parameters for the newly added spectrum are initialized by linearly extrapolating the previously optimized parameters. Although it is possible to use the same cubic splines to extrapolate the parameters of the newly selected spectrum, a linear function is chosen to follow a simpler, more general trend. Additionally, linear extrapolation is numerically more stable and increases the likelihood of meaningful initializations, even though this extrapolation method is still rather unreliable for moving peaks, especially with changing movement velocities.

Starting with t_{init}, the windowed optimization progresses through T_{sel} in ascending order, then in descending order, again starting with t_{init}. If desired, the selected spectra with the smallest and largest indices in T_{sel} can be optimized multiple times, since every other spectrum is optimized several times as well.

Variant 1: [Optimizing Centers First]

Since the main optimization method suffers from large runtimes and poor reliability, we

split up the optimization process into smaller parts. All four variants [1-4] employ an alternative, quicker, but less accurate peak detection method. Given a spectrum and some number of peaks m, the method yields the m most likely locations of peaks in this spectrum.

In this heuristic peak detection method, we approximate the first derivatives using the Savitzky-Golay method. Then, we look for sign changes in the resulting derivative vector. Finally, we output the indices of the m most significant sign changes. Significance is measured by the distance between the most negative and most positive values before and after the sign changes again. Additionally, we neglect sign changes with significance below a certain threshold tol_{vpd}. If desired, we also neglect peaks that are less than tol_{hpd} apart. Although this method is less reliable than the peak detection described in Sec. 4.2.2, it is significantly faster while providing rough estimates for peak positions on the entire data set. Fig. 4.12 presents this method for a single spectrum of D_a and shows the results for the entire data set D_a. Note the scattered points on the right, which implies that this method should not be used to define exact peak positions. Another advantage is that areas where no peaks are detected, most likely do not contain peaks. So, even if the number of false positives is large, false negatives rarely occur.

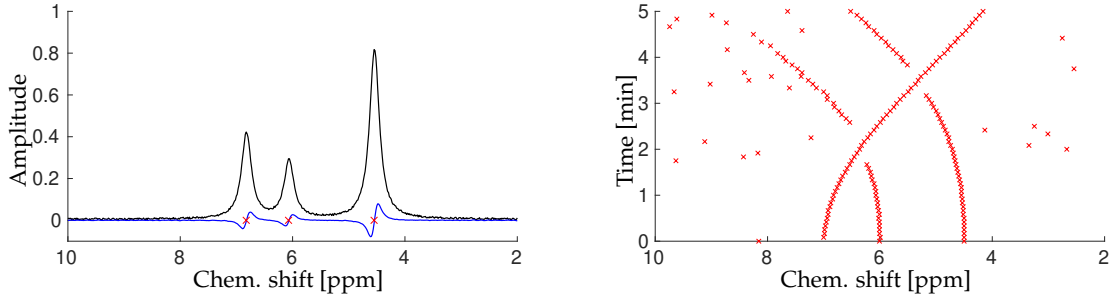

FIGURE 4.12: Heuristic peak detection on $t = 13$ (left) and on D_a (right). The first derivative is displayd in blue, and red crosses represent potential peaks.

After the heuristic peak detection yields the potential peak positions, we optimize the set of multiplet centers on the selected spectra T_{sel} as follows:

Let $s_i(t)$ be the cubic spline function that interpolates the value of the i-th peak center[12] and let $pos(t) \in \mathbb{R}^m$ be the vector of proposed peak positions at $t \in T$. Furthermore, let $T_w = \{t_l, t_{l+1}, \ldots, t_u\} \subseteq T$ be the set of all spectra within the moving window. Then, the optimizer aims to minimize

$$\sum_{t \in T_w} \sum_{i=1}^{m} \left(\min_{j=1,\ldots,k} |s_i(t) - pos_j(t)| \right)^2 \rightarrow \min! \tag{4.3}$$

In other words, the sum of minimum distances of between model peaks to any of the proposed peaks is minimized.

Finally, we fix the optimized values for the center parameters. Then, all the other parameters are optimized using the same procedure as in variant 0. If the optimization of the center parameters is successful, the second optimization is not only more stable, but also runs in less time.

Variant 2: [Optimization of Subsystems]

This method aims to find subsystems that can be modeled independently. In theory, this

[12] In the case of multiplets, we compute the sub-peak centers and optimize both d_i and c_i.

results in smaller data matrices with fewer peaks and fewer parameters that need to be optimized.

The first step is identical to the previous method. We heuristically detect peaks in the entire data set, and then optimize only the centers. Due to the rare occurrence of false negatives, we can reliably identify intervals on the frequency axis, where for all spectra, no peak is present. To ensure that these areas are indeed without peaks, we may can also take the optimized peak centers into account. If no detected, optimized, or interpolated peak centers are in an area larger than tol_{subsys}, we assume that all peaks are situated exclusively on one side of this area. We usually define $tol_{subsys} = 0.2$ ppm.

For each of these areas, we split the data matrix along the corresponding column at their midpoints. Then, we optimize all parameters, including centers, in each of the newly defined subsystems. Since the multiplets are contained within a subsystem, this approach does not compromise generality. Fig. 6.5 presents the results of subsystem detection for the data sets D_1, D_2 and D_3.

Remark 4.2. A similar subroutine is described in Sec. 5.4.2. It splits single spectra into smaller subspectra, which are then input to neural networks, see also Fig. 5.17. To avoid confusion, the smaller data sets are called *subsystems*, whereas the parts of single spectra are called *subspectra*.

Variant 3: [Individual Multiplet Optimization]

The third method optimizes each multiplet individually over time. First, we require heuristic peak positions so that the centers can be optimized similar to variant 1. Then, for each multiplet, we define a window $[a_i(t), b_i(t)]$ encompassing the i-th multiplet for all $t \in T$. These windows are defined to be narrow, with their edges no more than $4 \cdot bw$ apart from the outer subpeaks of the multiplet. Finally, we optimize the parameters for each multiplet individually, in a moving window T_w over time, locally inside $[a_i(t), b_i(t)]$ for all $t \in T_w$.

This approach typically suffers from peak crossings, as all of the crossing multiplets are modeled individually. Therefore, the sum of the models usually overestimates the heights at those crossings.

Variant 4: [Sequential Optimization]

The fourth method is a sequential optimization of the selected spectra, with additional knowledge of the order of the peak positions for each spectrum.

Again, the first step is the same as in variants 1, 2, and 3, where the peak positions are optimized initially. Importantly, we take note of the peak positions and the different orders in the spectra, which change due to peak crossings. We then model all of the selected spectra similar to modeling t_{init}. First, peaks and multiplets are detected using the more sophisticated detection methods described in Secs. 4.2.2, 4.2.3. Second, each detected multiplet is optimized locally on this spectrum. Lastly, all parameters of this spectrum are optimized simultaneously. We then use the order obtained from optimizing the centers to assign the parameters from the single spectra to the correct peaks.

Note that all of these variants reduce the runtime, sometimes drastically. While the second method takes a conservative approach to dimension reduction, especially the third and fourth methods can be considered as *quick-and-dirty*[13] methods. In certain situations, for example to provide an overview of the data set, these methods are still of use.

[13] `https://dictionary.cambridge.org/dictionary/english/quick-and-dirty`

Minimizing the Wasserstein Distance

One naive approach to implementing the Wasserstein distance as an objective function is to simply neglect the linear parameters during optimization and compute them by solving Eqs. (4.2) after optimizing the nonlinear parameters. This works well for one peak. However, for multiple peaks, the linear parameters cannot simply be fixed, because norming the spectra for the Wasserstein metric eliminates only one degree of freedom. Implementing this approach can be achieved through a linear constraint on the heights. For example, we can require that the sum of all heights be constant. The correct constant can be found using linear least squares after the optimization. This method, however, only decreases the number of parameters by one, which is insignificant when the total number of parameters for m peaks is $4m$.

To fully exploit the Wasserstein metric and the split between linear and nonlinear parameters we suggest to compute the objective function from Eq. (2.24) by the two-step process described in Sec. 4.1.4. Although each individual computation of the objective function is more expensive, the optimization dimension is halved.

Since the optimization algorithm that was used for the Frobenius norm, `lsqnonlin`, can only be used to minimize the sum of squares, a different optimization algorithm must be used for the Wasserstein metric. In this case, we choose `fmincon`, which is implemented in MATLAB [55]. This complicates the comparison of the two objective functions, as we cannot assume that the same hyperparameters work well for both optimization routines. Furthermore, `fmincon` is a more general function, whereas `lsqnonlin` is specifically designed to solve nonlinear least squares problems. The results of using the Wasserstein objective function are displayed in Sec. 6.6

4.3 Conclusion

The most common method of modeling NMR spectra involves using an optimization procedure of some sort. Our proposed spline-based approach strikes a balance between a simultaneous optimization of all parameters on all spectra – which is computationally very expensive – and a sequential optimization of individual spectra. Unlike sequential optimization, our proposed approach uses the information on the behavior of the parameter functions over time ($p_i(t)$) to obtain information for the currently optimized spectra. For one, this eliminates the different possibilities of continuing the parameter functions, which is especially challenging for sequential optimization methods when peaks overlap or cross, see Figs. 4.1 and 4.2. Furthermore, using cubic spline functions forces the parameter functions to behave smoothly over time, and leads to chemically meaningful models. Additionally, this approach reduces the intrinsic ambiguity of the noisy case, see Sec. 3.3, by utilizing the parameter behavior of predecessing and subsequent spectra, that may be more selective.

Our methods can be applied to several fields of chemistry, but are particularly useful for monitoring chemical reactions, especially those with changes in temperature or pH. Since it is commonly assumed that the nonlinear peak parameters do not vary drastically, the moving peaks from D_1, D_2 and D_3, see App. B, can only be modeled accurately, if the peak tracing is conducted manually. Our more general approach resolves this problem by automatically tracking the peak movements and the behavior of other parameters. In the context of reaction monitoring, the moving window optimization also allows for online monitoring, as newly acquired spectra can easily be appended to the data set.

Interpolation using cubic spline functions allows to accurately model parameter changes while retaining a sufficiently simple model, because each spline function is uniquely determined by the values $\{p_i(t) \mid t \in T_{sel}\}$. Therefore, the dimensionality reduction is further

amplified, as only a select few spectra contain all essential information about the entire data set.

Despite being an efficient method, the nonlinear optimization routines are computationally expensive. Thus, we introduce several improvements to the runtime, such as a set of variants designed to reduce the number of simultaneously optimized parameters, by partitioning the data set into subsystems, dividing the set of parameters, or a more general type of model that captures the multiplet patterns that appear because of J-coupling.

The most significant reduction in runtime can be achieved by improving the initialization of the optimization routines. Therefore, if a heuristic can be defined that predicts the peak parameters with sufficient accuracy for any given spectrum, the optimizer can be initialized more precisely. This would lead not only to a decrease in runtime, but also to an increase in model quality, as the likelihood of converging to local minima is reduced. Defining such a prediction method is the topic of Chapter 5.

Chapter 5

Parameter Prediction Using Neural Networks

The fact that the peak parameters, such as height, position and half-width, correspond to measureable quantities, see Rem. 2.9, enables us to use a second type of approach, which involves obtaining the peak parameters by observing the structure within the spectra.

There are possible heuristic methods which can be used to obtain the peak parameters without optimization, see Secs. 4.2.2, 4.2.3. However, it is easy to define counterexamples to any fixed heurstic method, where the parameter prediction fails, e.g., due to complex peak structures and overlaps. In a more general sense, a neural network can be viewed as an arbitrarily precise heuristic method of parameter prediction.

While the use of machine learning in the context of NMR is a comparatively recent phenomenon, similar tasks to those found in NMR spectroscopy also appear in other fields. For example, parameter prediction from NMR spectra can be viewed as a problem of *feature recognition*, which is a well-known problem in deep learning.

Effective application of machine learning to NMR spectroscopy seems to have begun in the early 90'es [137], although some uses of simple machine learning have been proposed earlier [67, 105]. Its many applications include the detection of peaks and multiplets, also known as *peak picking* [22, 36, 90], identification and classification of chemical compounds [37, 38] deconvolution of 1D [117] and 2D [77] NMR spectra, quantitative analysis [59], prediction of chemical shift values [119], and many more [48, 100]. Neural networks are also used in commercial applications [20, 94]. For further applications, we refer to the review paper [24]. For other types of spectroscopy, where pseudo-Voigt model functions are used, such as X-ray crystallography, similar applications can be found [79, 85].

This chapter describes how we use neural networks (NNs) to aid in the optimization described in the previous chapter. First, we define simple neural networks for our purposes and explain how they can be used in NMR spectroscopy. Then, we provide a complete overview of all the architectures and experiments on which we trained neural networks on. Next, we define the architecture for the NNs that are used in our algorithm and explain the training process in greater detail. Finally, we describe the implementation of the NNs in the spline-based optimization algorithm and present some results predicting parameters of single spectra. The application of convolutional neural networks to NMR time series has been published in [53].

5.1 Feed-Forward Neural Networks

This thesis does not aim to provide a comprehensive overview of machine learning. For an introduction, best practices and a general overview, we refer to [44, 57, 102]. One of the simplest forms of neural networks is the *feed-forward network* (FFN). In this thesis, we only

use very specific types of FFNs and therefore omit a more general and thus more complex definitions. For these, we refer to [44, Ch. 6].

5.1.1 Architecture

Definition 5.1. *A* feed-forward neural network *is a directed graph $FFN = (V, E)$ where V is the set of* vertices*, $E \subseteq V \times V$ is the set of* edges*, combined with a set of* activation functions *$f_v : \mathbb{R} \to \mathbb{R}$ associated with each vertex $v \in V$ and with a* weight function *$w : E \to \mathbb{R}$. The vertices $v \in V$ are also called* nodes *or* neurons*. Usually, the set V is completely partitioned into several disjoint subsets $V = \dot{\bigcup}_{i=1}^{l} L_i$ called* layers*. Typically, the first layer $L_1 = L_{in}$ is called the* input layer *and the last layer $L_l = L_{out}$ is called the* output layer*. In feed-forward networks, all edges $vw \in E$ where $v \in L_a, w \in L_b$ also satisfy $a < b$. This induces a partial order and implies a progression of data being* fed forward *throughout the layers. Furthermore, the graph and the set of activation functions are collectively referred to as the* architecture *of the neural network.*

An example network consisting of eleven nodes grouped into four layers $L_1 = \{in_1, in_2, in_3\}, L_2 = \{1, 2, 3, 4\}, L_3 = \{5, 6\}, L_4 = \{out_1, out_2\}$ is displayed in Fig. 5.1.

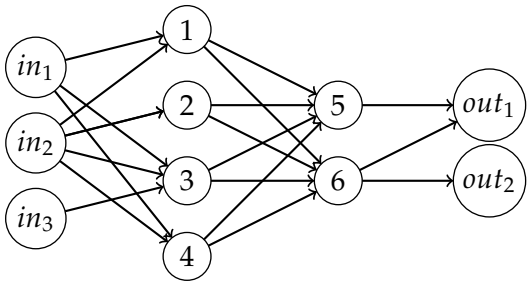

FIGURE 5.1: Example feed-forward network (FFN) with four layers.

This definition does not yet capture the full potential of neural networks, as it lacks the computational aspects. NNs can therefore be seen as functions $FFN : \mathbb{R}^{|L_{in}|} \to \mathbb{R}^{|L_{out}|}$.

Definition 5.2. *Let FFN be a feed-forward network and let $x \in \mathbb{R}^{|L_{in}|}$ be an* input vector*. Then, the* output vector *$FFN(x)$ is computed as follows: We assign a real numbers to each node $v \in V$ by defining a* computation function *$c : V \to \mathbb{R}$, where*

$$c(v) = \begin{cases} x_j, \text{ if } v \text{ is the } j\text{-th node of } L_1 \\ f_v\left(b_v + \sum_{u \in N^-(v)} c(u) \cdot w(uv)\right), \text{ else.} \end{cases} \tag{5.1}$$

In the previous definition, $N^-(v)$ denotes the set of all vertices u, where an edge $uv \in E$ exists and b_v denotes a bias term*. Using this iterative computation, we then define $FFN(x)$ as the vector of values $c(v)$ of all $v \in L_{out}$.*

Starting from the input Layer L_{in}, all computations are forwarded through the network to produce values in the output layer L_{out}. In the case of the network from Fig. 5.1, we take a closer look at the calculations for node 3 in Fig. 5.2. Its output $c(3)$ can be computed using the bias b_3, the weight function w and the results of previous nodes.

$$\begin{aligned} c(3) &= f_3\left(b_3 + c(in_1) \cdot w_1 + c(in_2) \cdot w_2 + c(in_3) \cdot w_3\right) \\ &= f_3\left(b_3 + x_1 \cdot w(\{in_1, 3\}) + x_2 \cdot w(\{in_2, 3\}) + x_3 \cdot w\{in_3, 3\})\right). \end{aligned}$$

Note that since the in_i are input nodes, their outputs equal the values of the corresponding components of the input vector x_i.

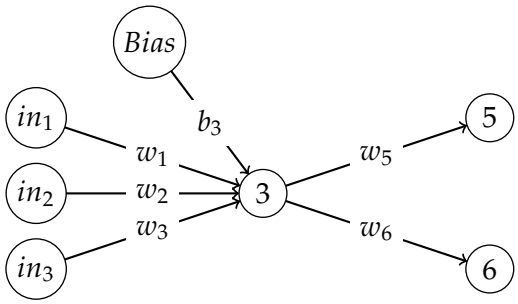

FIGURE 5.2: Computation of the output of node 3.

Although several layers can be used to for feed-forward networks, we only require three different types.

1. Dense layers.
   A dense layer connects to all nodes of the previous layer L_{prev}. In other words, for all nodes v in a dense layer, $N^-(v) = \{u \mid u \in L_{prev}\}$. The activation function for nodes in dense layers is usually some kind of sigmoid, such as the *standard sigmoid*,

   $$sig(x) = \frac{e^x}{1 + e^x}.$$

   Other common activation functions include the *rectified linear unit* function,

   $$ReLU(x) = \max(0, x),$$

   the identity function

   $$Id(x) = x,$$

   sometimes called *linear activation*, or the *Heaviside step function*

   $$H(x) = \mathbb{1}_{x \geq 0}(x).$$

2. Convolution layers.
   A convolution layer connects to all nodes in a certain index window of the previous layer. If $L_{prev} = u_1, \ldots, u_N$, then all nodes v in a convolutional layer are fed by $N^-(v) = \{u_i, u_{i+1}, \ldots, u_{i+k-1}\}$ for some index i and a *kernel size* of k. Additionally, the k weights of the edges connecting to a single vertex are identical for all v of that convolutional layer. Therefore, its computation resembles that of a convolution kernel of size k moving through the previous layer. Similar activation functions can be used for convolutional layers, though the identity function is often chosen, especially if the layer is directly followed by a max-pooling layer. This is because the nonlinear activation function can be applied after reducing the dimension during pooling. An example convolution layer with a kernel size of 2 and two kernels F_1, F_2 can be found in Fig. 5.3.

3. Pooling layers.
   Similar to a convolutional layer, a pooling layer is only locally connected to the previous layer. However, it is designed to reduce the dimension, so its computations may not always fulfill the definition from (5.1). Conventionally, pooling layers directly follow convolution layers, and, in the case of max-pooling, compute the maximum value of k consecutive nodes of the previous layer. Other aggregate functions can also be

applied, such as the average, or a mixture of the average and the maximum. If no activation function was applied to the convolution layer beforehand, it is usually applied after pooling. Otherwise, the activation after pooling is often omitted. An example pooling layer which aggregates the outputs of three previous nodes each is displayed in Fig. 5.3.

4. Recurrent layers.
   The last commonly used type of layer violates the partial order in the network graph by introducing cycles. Its output values are given to previous nodes to be used during future input predictions. These layers may also use their previous outputs to compute the next set of outputs, hence the name. For the task at hand, we opted not to use recurrent layers and refer to [102, Ch. 15] for further details.

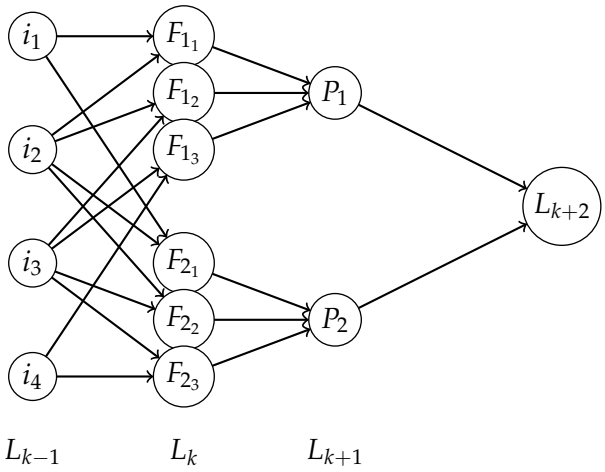

FIGURE 5.3: Example of a convolutional (L_k) and pooling layer (L_{k+1}).

While this definition of feed-forward networks is simple, it still allows us to create several *data channels*, meaning a layer can be the predecessor to multiple subsequent layers, and several layers can be connected to a single successor layer.

5.1.2 Training

The architecture of neural networks is typically defined in such a way that there are several layers containing tens of thousands of nodes in total. For industrial applications the number of parameters may reach 10^9 for ImageNet [76] and 10^{11} for GPT-3 [18]. For our purposes, the number of parameters does not exceed 10^6. Still, manually defining each bias term b_v and each weight $w(uv)$ is not an option. Instead, the values for these network parameters must be found automatically. This is done by a large-scale nonlinear optimization algorithm. This process is called *training* and requires a large amount of different inputs, called *training data*.

Usually, the training is conducted as follows. First, a *batch*, i.e., a subset of the training data, is given to the neural network to compute the outputs. Then, an objective function is computed to measure the error of the outputs. Finally, the network parameters $b_v, w(uv)$ are altered according to some basic optimization procedure. For example, the adjustments may be made in the direction of the steepest gradient. The gradients of each individual weight and bias term are computed from back to front in a process called *backpropagation*, which uses the previous gradients and the chain rule on the activation functions. For a detailed explanation, we refer to [44, Ch. 6.5]. Since the number of optimizable parameters

is very large, the training process requires significant amounts of computational power and time. Typically, the large training time is mitigated by choosing a very basic optimization procedure, such as steepest gradient descent. Training large-scale neural networks can take several weeks. In our case no training time exceeded two days, and the networks we eventually used were trained in less than one day. The training process stops either after a certain amount of data has been processed, or when the objective or another loss function is below a certain threshold.

Since the optimization process works by computing errors on the training data set, this data set should caputre the general properties of the data that will eventually be processed by the NN.

5.1.3 Potential Applications of NNs to NMR Spectral Analysis

Typically, neural networks can be used to great effect, when there are calculations requiring extensive and expensive numerical computations. For example in quantum chemistry, neural networks can provide improved heuristic methods and quick approximations for large-scale quantum-chemical computations. For instance, `DEERNet` predicts electron distance distributions [132], which, without machine learning, would involve computing expensive and complex integral terms.

Other applications include procedures, in which a human user is able to quickly determine the results, when an automatied approach fails to detect subtle differences in the given data, or when no sufficiently good heuristic can be found. One such application is peak detection. Peak detection can be performed either using a classification approach, e.g., in [37, 94, 117], where each pixel is classified into at least two classes, peak or no peak, or using a regression approach, where peak positions, i.e., real numbers, are predicted [90]. Peak detection is also possible for two-dimensional spectra [22, 77]. Additionally, spectra may be automatically divided into subsystems or clusters of peaks [36]. Another possible application of NNs is the quantitative analysis of a mixture spectrum, given information on the pure component spectra [31, 38, 59]. For more applications of neural networks in NMR spectroscopy, we refer to the overview article [24].

We initially considered using neural networks as we implemented the optimization variants. There, it became clear that, once the center parameters were known, optimizing all the other parameters would be easier, more stable, and quicker.

Therefore, our initial apporach using machine learning is a classification approach as well. However, it quickly became apparent that this approach does not yield satisfactory results. Thus, we instead opted for a regression approach of predicting peak centers. Since only minor adjustments need to be made to the regression approach, predicting all peak parameters simultaneously is only a logical extension of the regression approach.

5.2 Suitable Architectures

First, we describe a pure classification approach. For the training data, we use a single spectrum from data set D_3, where the peak positions are annotated by hand. All data sets used are presented in detail in App. B. This spectrum also contains several *artifacts*, that are often detected by heuristic detection methods, see also Fig. 6.2. However, modeling these artifacts is not necessary. Therefore, manually defined peak positions without artifacts serve as the ground truth. The training data sample is very small because this test merely serves as a proof of concept and an experiment to familiarize ourselves with the different methods that need to be implemented. Therefore, the architecture of the given network is also simple. It consists of the input layer of size 51 followed by a dense layer of 128 nodes, activated by

the *ReLU* function. The next and final layer is the output layer of size 1, activated by the standard sigmoid $sig(x) = \frac{e^x}{1+e^x}$. This output layer is densely connected to the previous 128 nodes. However, each edge is deleted with a 50% chance. This process is called *dropout*.

The simple task of the network is as follows. Given a connected window of this input spectrum, i.e., the values of $[s(x_{i-w}), s(x_{i-w+1}), \ldots, s(x_{i+w})]$, where i is randomly chosen, uniformly distributed over all non-edge indices, the network predicts its confidence $\in [0,1]$ of a peak being present at position x_i. A confidence of > 0.5 corresponds to a peak prediction. For the objective function we choose the *binary cross entropy*, which in our case can be written as

$$BCE(y_{pred}, y_{gt}) = -(y_{gt} \cdot \ln(y_{pred}) + (1 - y_{gt}) \cdot \ln(1 - y_{pred})),$$

where $y_{gt} \in \{0,1\}$ and $y_{pred} \in [0,1]$ represent ground truth and predicted confidence, respectively.

The vector y_{gt} contains only zeros except at the annotated indices, where its components equal one. The implementation of all machine learning was done in `python`, initially using the `tensorflow` package [1], which is well-established in the literature and provides a robust framework for implementing our neural network experiments.

Training is typically conducted in *epochs*. One epoch involves inputting several batches to the neural network and adjusting the parameters several times. After each epoch, the progress is measured on *validation data*, i.e., data that is separate from, but similar to the training data set. In this case, since only one spectrum was manually annotated to begin with, both validation and training data is identical. We trained for 100 epochs, each involving computations on 2,500 batches of size 1, using the optimizer `adam` [63] which is already implemented in `tensorflow`. It very quickly became apparent that some adjustments were necessary.

There are 23 peaks in this data-set and 48,005 indices that can be chosen. This means that in $\frac{47982}{48005} \approx 99.9\%$ of cases, no peak is present. Therefore, during training, the network adapts a trivial strategy of always predicting *no peak*. This method indeed has an accuracy of $\approx 99.7\% - 100\%$ per epoch. However, the neural network is unusable for any practical purposes.

5.2.1 Pixel Classification

To combat training trivial classifiers, we force the training data to contain a disproportionately high amount of peaks. To achieve this, we implement a parameter $p \in [0,1]$ corresponding to the probability that a window centered around a peak is chosen during training.[14]

For different values of p we obtain different results after training for 100 epochs of 2,500 batches each. We not only measure the *loss*, the binary cross entropy, but also the number of true and false positives *tp, fp*, the number of true and false negatives *tn, fn*, and the accuracy

$$acc = \frac{tp + tn}{2500}.$$

Table 5.1 and Fig. 5.4 show that for $p \in [0.25, 0.75]$, detection of both classes is trained even during these short training sessions. Note that accuracy alone is not sufficient for observing prediction quality. Therefore, we also computed the balanced accuracy, defined as

[14] This is not quite correct, as the implementation requires us to perform two pseudo-random experiments. First, a uniformly distributed random number r in $[0,1]$ is chosen, that decides if a peak should be selected. Second, a random index i that is either chosen as one of the peak indices, if $r \leq p$, or as one of all indices, if $r > p$. Therefore, the actual probability to obtain a peak is $p + (1-p) \cdot \frac{23}{48005}$.

p	0	0.05	0.1	0.25	0.5	0.75	0.9	0.95	1
$loss$	0.0117	0.0566	0.0845	0.0809	0.1494	0.1386	0.1872	0.1277	$7 \cdot 10^{-12}$
acc	0.9988	0.9816	0.9732	0.9784	0.9640	0.9628	0.9528	0.9600	1
tp	0	85	222	593	1224	1850	2209	2354	2500
fp	0	22	32	31	46	51	79	76	0
tn	2497	2369	2211	1853	1186	557	173	46	0
fn	3	24	35	23	44	42	39	24	0

TABLE 5.1: Selected values of p and corresponding training results.

$acc_b = \frac{1}{2}(\frac{tp}{tp+fn} + \frac{tn}{tn+fp})$, which can display weaknesses in finding positives and negatives alike. Considering there only were 48,055 possible inputs, these results only demonstrate that a neural network can learn to detect peaks in an experimental spectrum, or at least learn to recognize the input data set.

We considered this proof of concept to be successful, yet with unsatisfying results. If we were to train NNs on similar approaches using more complicated and different data, the results would likely be worse. Therefore, the next set of experiments is conducted using a different approach.

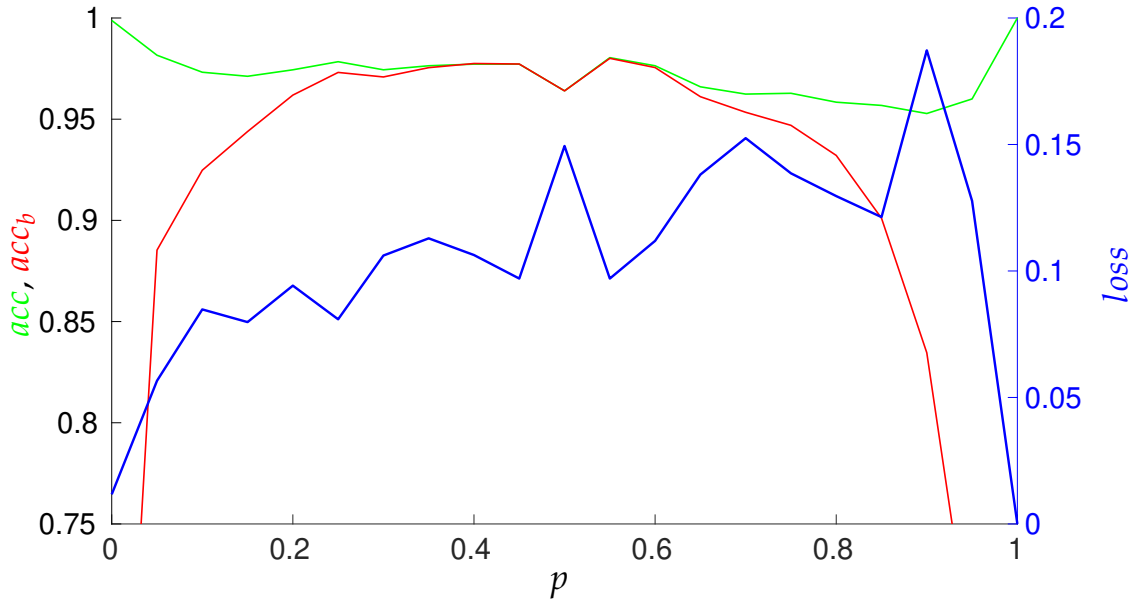

FIGURE 5.4: Accuracy measurements acc in green, acc_b in red and $loss$ in blue for different values of p.

5.2.2 Parameter Regression

Contrary to the assumptions of the classification method, it suffices to determine a peak position up to a few indices to achieve a good model. If afterwards, the resulting indices are converted to floating point numbers anyways, the idea to not requiring the exact indices is not too far-fetched.

The first experiments of this regression approach are only focused on the peak positions, as they are the most crucial type of parameter for successful modeling. All the other parameters would then be obtained by our nonlinear optimization routine.

Since this task differs from the previous set of experiments, we must change the training process. For the training data, we created a connected set of spectra D_b. The data set contains three peaks that intersect thrice. More details on the construction of D_b can be found inApp. B. An overview on this data set is presented in Fig. B.2.

The advantage of constructing D_b is that all parameters are known and can serve as the ground truth values. We split the time series into three subsets of spectra. The first set contains 75% of the spectra and is used as the *training set*. The second set, containing 15% of the spectra is used for *validation* after each epoch. The final 10% are used as the *test set* after the training is complete. To avoid having only 75 spectra in the training set, we interpolate the spectra of the original data-set to 10,000 spectra. Each of the three sets contains randomly selected spectra. We train the NNs for 100 epochs of size 7,500 and input the entire spectrum each time.

For the objective functions we choose the mean absolute error (MAE) and the mean squared error (MSE) and minimize them using the `adam` optimizer. Since both metrics behaved similarly, we ultimately decided to use only MAE as an objective function. For each pair of predicted and ground truth samples c_{pred}, c_{gt}, we have

$$MAE(c_{pred}, c_{gt}) = \frac{1}{3} \cdot \|p_{pred} - p_{gt}\|_1$$
$$MSE(c_{pred}, c_{gt}) = \frac{1}{3} \cdot \|p_{pred} - p_{gt}\|_2^2.$$

Here, our architecture is slightly more complex cpmpared to the initial classification approach. The input layer consists of 1,000 nodes that are densely connected to a layer of 128 neurons, which then connects to another dense layer of the same size. Both of these layers are activated by the standard sigmoid. The two layers connect to the output layer of size 3, which has no activation function, as the image of the sigmoid is $[0, 1]$ and our data set contains peaks in the range of $[0, 10]$. Additionally, to prevent the model from merely learning the data set, each edge between the two dense layers of size 128 is dropped with a probability 0.5.

Fig. 5.5 shows the development of the validation loss after each epoch. At first glance the loss looks very promising. A mean average error in the range of $0.01 - 0.015$ implies that, on average, each predicted peak position is $1 - 1.5$ indices off the ground truth. In fact, the error on the testing data set is equal to 0.0154.

There are several problems with this approach. First, even though the network is designed to have redundant information transfer thanks to the droupout, it still learns the entire data set quickly. Note that after 20 epochs the training error is already below 0.1, which can be considered suspiciously good. Second, the validation, training and test data sets are very similar to each other, due to the interpolation from only 100 spectra, which explains the great results on the test data as well. On the other hand, it would have been difficult to create large amounts of training data consisting of connected spectra. However, the biggest giveaway is the order of the predicted peak parameters. The peak crossings are not reflected in the ground truth vector, i.e., the peak position values of the ground truth vector are not always in ascending order. However, by looking at a spectrum alone, there are $3! = 6$ possible orders for the peak centers. Still, the losses imply that, on average, the order is guessed correctly, even though there should not be enough information in the spectrum to reconstruct it.

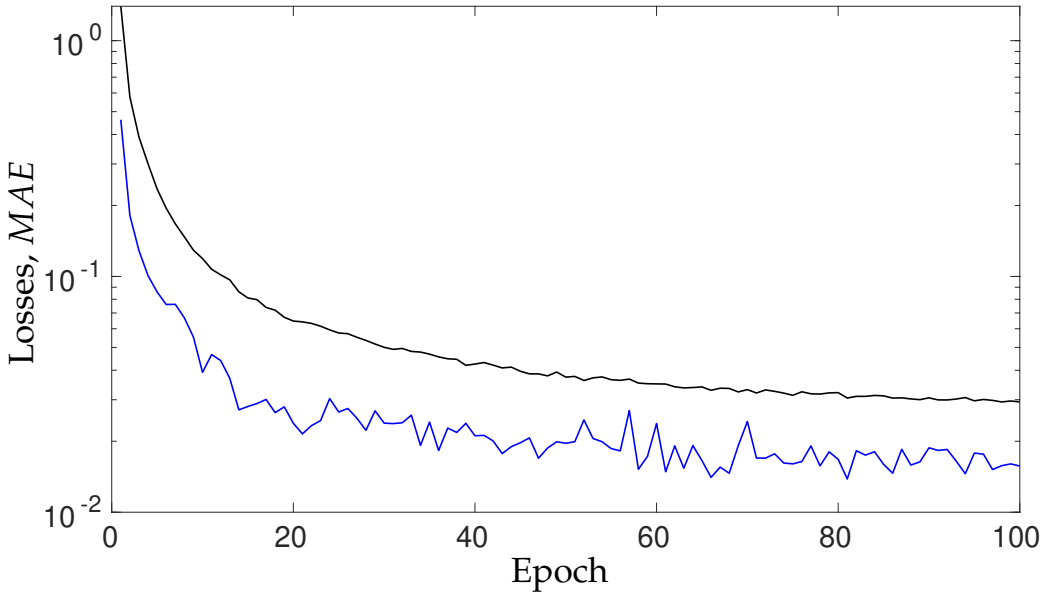

FIGURE 5.5: Training progress during the first parameter regression. Training (black) and validation (blue) losses are displayed.

5.2.3 Predicting All Peak Parameters

Once again, we came to the conclusion that our training approach needs to change. After many fruitful discussions with Professor Labahn's group, particularly with Dr. Tobias Strauß, the idea of predicting all parameters at once arose. Since this task is more complex that previous ones, we first tried to determine its feasibility.

Another proof of concept:

To detect all parameters, we first changed the architecture. The first layer after the input layer contains 100 nodes, whereas the second layer consists of only 50. However, the output layer contains 12 nodes, or four parameters for each of the three peaks. Additionally, we also activate the output layer. Therefore, we must normalize each of the parameters to $[0, 1]$ before beginning the training. This has the added benefit of making each parameter equally important in minimizing the MAE.

The results, as shown in Fig. 5.6 are very similar to those of the previous experiments. Training apears to occur quickly, and reliably. Additionally, after only 20 epochs, all parameters are already predicted with an absolute error of $< 1\%$. After 100 epochs, the loss on the test data is ≈ 0.001.

We again consider this a successful proof of concept. However, the networks merely memorize the data sets. Therefore, they are still not applicable to more general data sets, let alone experimental data. Crucially, the training data lacked sufficient variance between different samples.

More Applicable NNs:

For the next set of experiments, we ensure that the training data set is not only sufficiently large, but also sufficiently general. Therefore, rather than using a fixed data set from which spectra are selected, we generate the training data ourselves. Since we know the general form of an NMR spectrum, given by Eq. (2.1), we can cover (almost) the entire space of

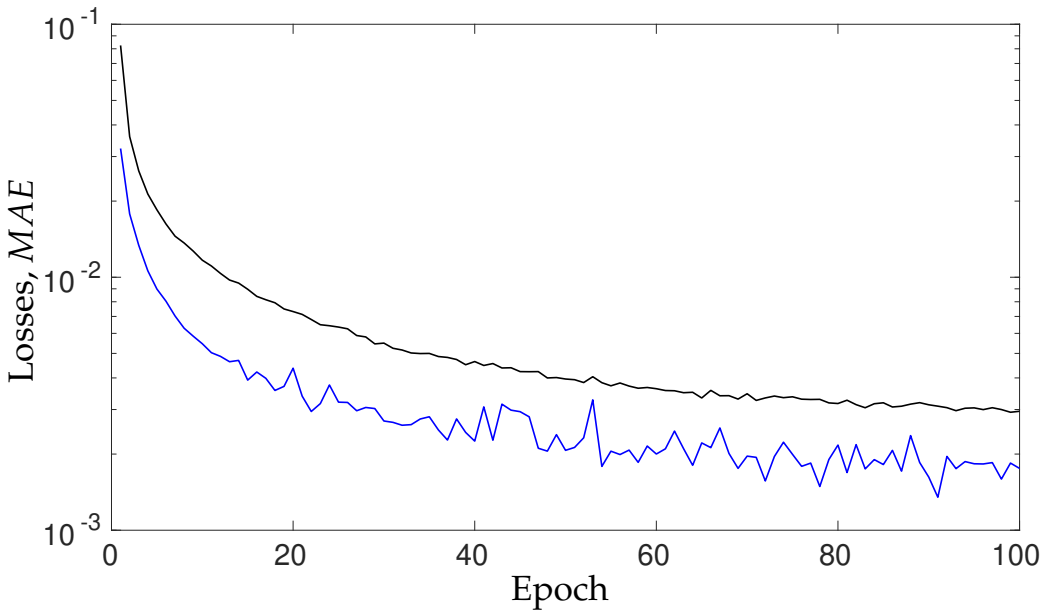

FIGURE 5.6: Training progress during the second parameter regression. Training (black) and validation (blue) losses are displayed.

possible NMR spectra composed of a fixed number of peaks. When compared to phase- and baseline-corrected experimental spectra, which can be achieved using the SINC algorithm [114], the spectra defined by Eq. (2.1) merely lack added noise. Thus, we define our training spectra as follows.

$$s(x) = N(x) + \sum_{i=1}^{m} h_i \cdot GL_{c_i,w_i,\lambda_i}(x). \tag{5.2}$$

Here, $N(x)$ describes a normally distributed random vector of finite dimension. This approach can generalized further to accomodate for incorrect phasing or a baseline using different and more general model functions. However, generating spectra using Eq. (5.2) has proven sufficiently general in practice. Note that because we generate the training data, we also obtain the ground truth parameter vectors.

For the following sets of experiments, we randomly choose all peak parameters with uniform distributions with in the intervals

$$c_i \in [0, 10], w_i \in [0.0001, 0.5], h_i \in [0.1, 1], \lambda_i \in [0.0001, 0.2].$$

This is the only time, where we compromise on generality[15]. During trainig, all parameters are linearly transformed into the interval $[0, 1]$ due to the image of the sigmoid. Additionally, we now sort the values of the ground truth by the center values in ascending order.

Spectra consisting of a fixed number of m peaks are generated during training on 1000 points in the interval $[0, 10]$ and the normal distribution of all components $N(x_i)$ has mean 0 and variance 0.005. Training is conducted for 250 epochs. Each epoch consists of $\approx \frac{500}{b}$ many batches of size b.

For the first set of experiments, we vary the batch size b to determine the optimal value. We use a network with $1,000$ input nodes and two subsequent dense layers of 24 and 16

[15]Still, any spectrum consisting of finitely many peaks can be linearly transformed into the interval $[0, 10]$. Similarly, it can be normed such that the heights never exceed 1.

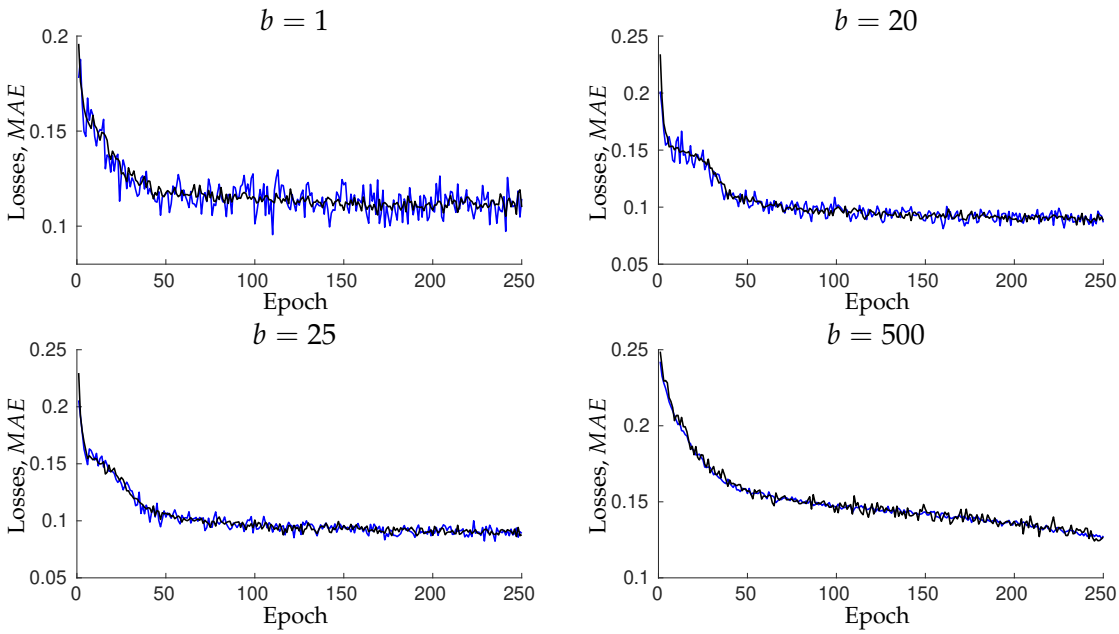

FIGURE 5.7: Training (black) and validation (blue) losses for different batch sizes b.

nodes for all experiments. Since in this case, $m = 1$, the output layer has a size of four. Fig. 5.7 present the MAE for four batch sizes. During each of the experiments, approximately 125,000 specta were generated.

The training results for different b are very similar due to the comparable size of the training sets. However, for large $b > 100$, the number of adjustments to the weights decreases, and the training slows down. Additionally, adjusting the weights after each input results in a quick, but unstable training process. Therefore, the validation loss for small $b < 10$ behaves more erratic, which also slows down the training. We conclude that batch sizes $b \in [10, 25]$ are the most promising.

Another important hyperparameter is the *learning rate*[16] η. It determines the portion of the gradients added to the weights during backpropagation. The default value for the `adam` optimizer is $\eta = 0.001$. Again, Fig. 5.8 displys the loss for standardized training using different learning rates η. It turns out that the default value is not only useable, but often optimal. When η is too small, learning occurs very slowly. On the other hand, when η is too large, the weights change too drastically and this sometimes prevents any learning at all.

These experiments demonstrate that relatively simple NNs can learn to predict peak parameters to a certain extent. In fact, most parameters are already predicted correctly, whereas parameter prediction is difficult, especially for the spectra with small h and w. Additionally, the parameter λ often has no significant impact on the appearance of the data and cannot reliably be predicted. Furthermore, the peak positions are inaccurate, since 24 nodes must take into account information from 1,000 input nodes. The dense layers do not take advantage of the fact that the same shape of a peak may appear at any position.

This type of feature recognition is the strong suit of convolution layers and CNNs. Convolution layers resemble a moving window through the previous layer and eventually allows the NNs to fit a mask of weights suited for peak detection onto the input spectrum.

[16] In the literature, the learning rate is also denoted by α.

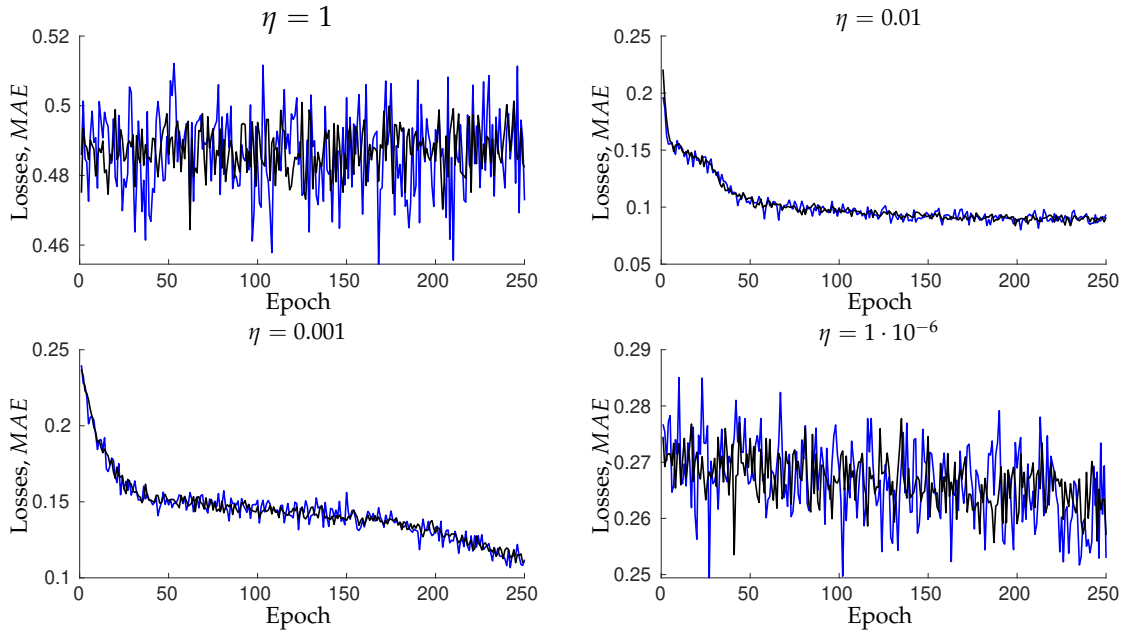

FIGURE 5.8: Training (black) and validation (blue) losses for different learning rates η.

5.2.4 Convolutional Neural Networks

The typical architecture of a convolutional neural network (CNN) is as follows: First, there are multiple convolution and pooling layers. The results of the pooling layers are typically fed into multiple dense layers.

Since CNNs perform significantly better, we increase the number of peaks per spectrum. Additionally, we also train CNNs to predict the four parameters of Voigt functions. Since detecting sharp and small peaks proved difficult during previous experiments, we also implement an option to overly distribute sharp peaks in the training data. Due to the variety of the architectures used, we only describe some experiments in detail. For most experiments, we briefly describe the achieved losses. In this subsection, all losses are measured using the MAE.

We begin our experiments with a CNN with the following architecture

$$P + (C + C + P) + (C + C + P) + D + D + D + D = P + 2(2C + P) + 4D.$$

Here, P, C, D represent max-pooling, convolution, and dense layers, respectively. Input and output layers are omitted, since they are always of size 1,000 and $4m$, respectively. In this case, the detailed parameters for the layers are as follows:

- Pooling layers:
  Three max-Pooling layers pooling 20 consecutive nodes of the previous layers
- Convolution layers:
  Four layers with descending kernel sizes of 50, 25, 17, and 13. All convolution layers are activated by $ReLU(x)$.
- Dense layers:
  Four layers, densely connected with 60 nodes each. All dense layers are activated by $sig(x)$

- Input and output layers:
  Input size 1,000, output layer is dense with $4m$ nodes, activated by $Id(x)$.

CNNs with this architecture achieved promising results for spectra with one or two peaks. For similarly generated test spectra containing one peak, the average deviation of the predicted center parameters is about 0.0015, or about 1.5 pixel in terms of resolution. The average deviation of half-widths is ≈ 0.014. For heights it is ≈ 0.008. However, for the convex combination parameter, the deviation is ≈ 0.2447. In other words, the trained predictions are almost as bad as guessing[17]. This leads to a total MAE of ≈ 0.067.

Modeling Spectra Using Voigt Functions:

Training CNNs with this type of architecture for more than one peak is difficult, and the best results can be achieved using Voigt functions. For two peaks, the MAE is ≈ 0.1174. However, the comparable parameters c and h had lesser average deviations of ≈ 0.021 and 0.055 respectively.

Nevertheless, this indicates that, on average, each peak is approximately 21 pixels from the ground truth. Therefore, in many cases, the deviation from the correct center values is larger than the half-width. We cannot guarantee that these predictions are an improvement on extrapolation during our spline approach.

Note that, because of the similarities between the two functions, a network trained on Voigt functions can reliably detect pseudo-Voigt peaks and vice versa. In this case, the average deviation for c amounts to ≈ 0.034 and for h is ≈ 0.15. This fact confirms our suspicion that some general information about the shape of peaks and their relationship to the corresponding parameters can be learned by NNs.

More Efficient and Large-Scale Training:

The results of the previous experiments were not satisfactory, but they are promising enough to continue. The next step is to train the models on a larger scale, i.e., for longer periods of time. Previously, the CNNs were trained over the course of several hours, whereas for the next few experiments, the training is conducted for several days. To improve the performance, we are taking a step back from the generalized approach. Where previously, the training data was created during training, now, the traning data is generated beforehand. We create separate data sets containing traning and validation data. Test data can be generated when necessary after the traninig completes.

Additionally, despite the emphasis on sharp peaks, it should be noted that sharp, thin peaks are detected less reliably than larger or broader peaks. This may be due to the kernel sizes of the convolutional layers, as in `tensorflow`, only one size per layer can be defined. However, to forward information about broader and sharper peaks simultaneously, it can be advantageous to provide the ability to detect sharp peaks in narrow windows and broad peaks in wide windows. Therefore, we implement a custom subtype of the convolution layer, where the input is split into several data channels, each convoluted with a kernel of custom size. Fig. 5.9 shows two consecutive custom convolution layers with different kernel sizes.

The following architectures revolve around the modified convolution (mC) layer. We conduct experiments with about 20 different architectures and model functions with similar results. All of the architectures have in common that the first few layers consist of some number of convolution and pooling layers, with various numbers and sizes of convolution kernels. After these layers, the last 2-6 layers consist of densely connected layers.

[17] If 0.5 is always guessed, the expected distance from a uniformly distributed random variable $\in [0, 1]$ is 0.25.

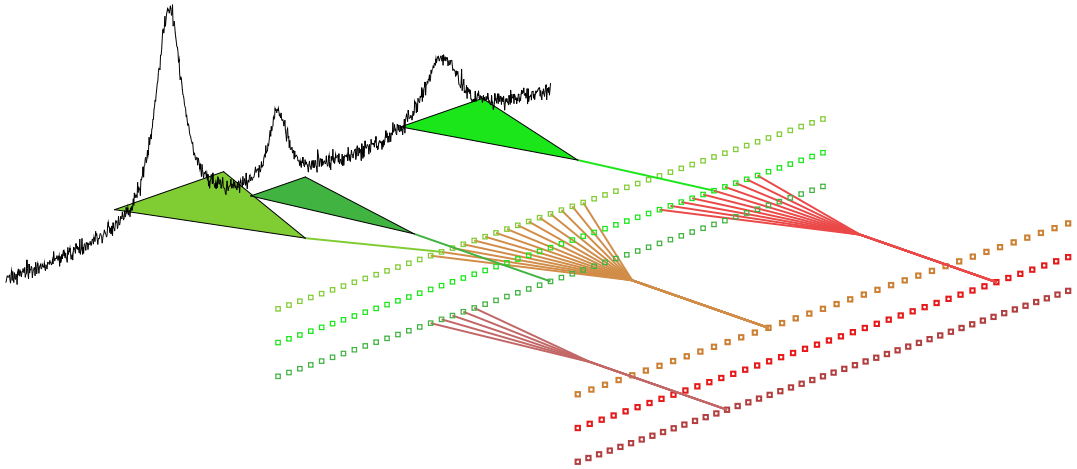

FIGURE 5.9: Input spectrum (black) and two convolution layers (green, red) with three different kernel sizes each.

For example, Fig. 5.10 displays the losses for two networks with the architecture $3(mC) + P + 6D$, trained on 25,000 spectra consisting of two pseudo-Voigt peaks on the left and on 100,000 spectra consisting of three Voigt peaks on the right. Since the training data set is generated before the training begins, the networks can experience overfitting, indicated by a growing difference between training and validation loss on the left of Fig 5.10. Increasing the number of available training spectra mitigates the problem, but does not completely solve it.

To further combat overfitting , we implement *early stopping*, which stops the training prematurely, if no significant progress is made. It then restores the weights corresponding to the lowest validation loss achieved.

The average test losses for the parameters c, h, w and λ amount to $\approx 0.012, 0.054, 0.038$, and 0.263, respectively with a total MAE of ≈ 0.092 for the two-peak pseudo-Voigt network. The three-peak Voigt network achieves average deviations of ≈ 0.017 for centers, ≈ 0.0570 for heights and $\approx$ 0.0724, 0.0791 for the two half-width parameters with a total MAE of ≈ 0.0563. There are two observations of note. First, it is notoriously difficult for pseudo-Voigt networks to predict λ. On the other hand, the two half-width parameters of the Voigt functions have a more direct influence on the peak shapes, enabling the networks to learn to predict these parameters. Second, the pseudo-Voigt models tend to have slightly smaller euclidean norms when reconstructing the data.

Applying CNNs to Experimental Spectra:

Finally, we apply the CNNs to expermental spectra. We use different types of data. First, we use a set containing Highfield and Benchtop data of twelve mixture spectra containing different carbonhydrates[18]. Second, we tested the networks against spectra from data set

[18]The data was provided by Patrick Sterner at RPTU Kaiserslautern-Landau.

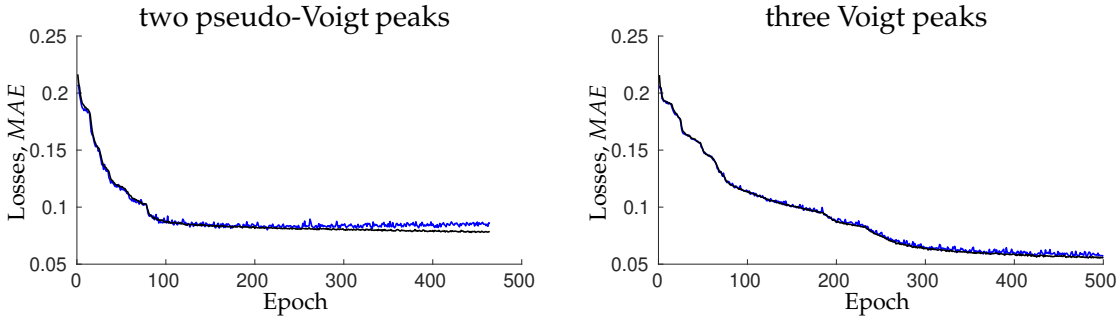

FIGURE 5.10: Training (black) and validation (blue) losses for networks trained on two pseudo-Voigt peaks (left) and three Voigt peaks (right).

D_2, see also B.

To illustrate the precision, Fig. 5.11 shows the results of applying the second network from Fig. 5.10 to one of the highfield mixture spectra.

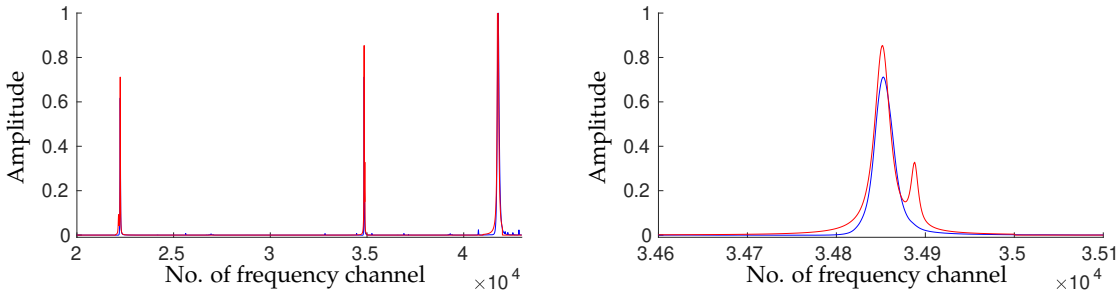

FIGURE 5.11: Experimental mixture spectrum (blue) and predicted models (red). *MSE* for the model is $\approx 0.0474‰$

Similarly, Fig. 5.12 shows the results of predicting peaks from one of the benchtop spectra. On the left hand side, we use a network with architecture $2(2mC + P) + 3D$, which is trained on two pseudo-Voigt peaks. On the right hand side, we apply the second network from Fig. 5.10.

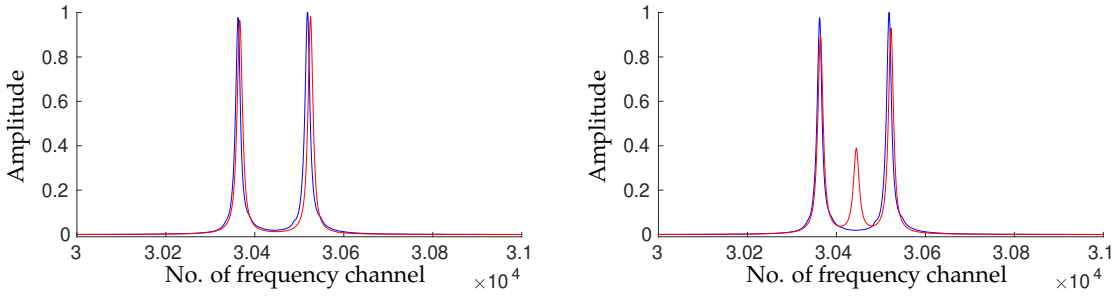

FIGURE 5.12: Experimental mixture spectrum (blue) and predicted models (red). *MSE* for the models are $\approx 0.0411‰$ (left) and $\approx 0.0402‰$ (right).

Both figures demonstrate successful knowledge transfer, as CNNs trained only on generated data manage to learn a basic understanding of peak positions, half-widths, and heights. However, upon closer inspection, e.g., on the right side of Fig. 5.11, it becomes apparent, that

the predicted models are not precise enough. For other spectra from this data set, the peaks of Fig. 5.12 are modeled more precisely.

Applying the trained networks requires some additional work, since the spectra usually do not contain exactly 1,000 data points. Therefore, the spectrum must either be split into disjoint subspectra of size 1,000, or be interpolated if the number of data points is lower than 1,000. Furthermore, an m-peak network can only predict parameters for exactly m peaks, as can be seen in Fig. 5.12. Still, two of the three peaks still carry meaningful information. The two figures also illustrate that both the Voigt and pseudo-Voigt networks manage to properly model the given experimental peaks. This is no surprise, as the pseudo-Voigt functions approximates any Voigt function quite well, since the maximum absolute error between Voigt and pseudo-Voigt is $\leq 0.77\%$ relative to the peak maximum [128].

Since we trained multiple networks with different architectures, different model functions and different numbers of peaks, we can input any spectrum to a several CNNs and rank the predictions according to some metric. Here, we choose the sum of squares, but other metrics, such as the Wasserstein distance, can be used, see Sec. 6.6.1. Applying several spectra does not significantly increase the runtime, since a single prediction is typically computed in < 1 ms.

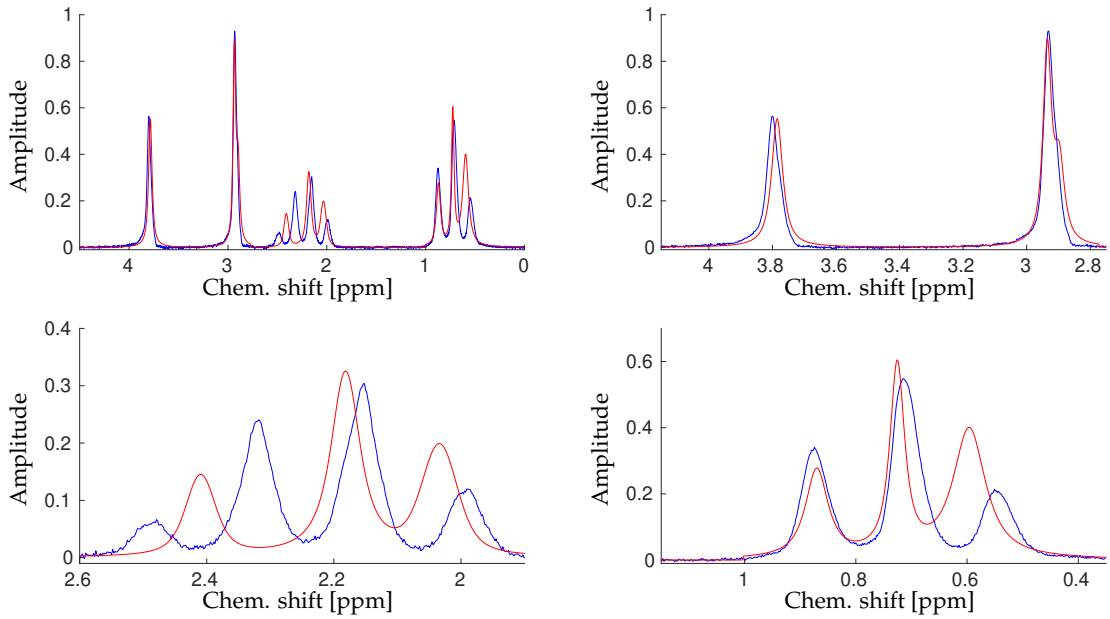

FIGURE 5.13: Spectrum $t = 143$ of D_2 (blue) and predicted models (orange). The average euclidean distance is $\approx 0.509‰$.

The models displayed Fig. 5.13 solidify the point that CNNs can indeed learn a general understanding. However, the predictions are often imprecise. Note that quality is dependent on the complexity of the data. For example, the top right shows a simpler section of this spectrum, which is modeled with sufficient precision. On the other hand, the bottom plots show more complex sections with less accurate models.

5.3 The Final Set of CNNs

Since all previous experiments encountered similar issues, we conclude that a different approach is once again required. One remaining, largely unaltered, hyperparameter is the

training time. Experiments with significantly longer tranining times also require significantly more computations. This is why, for the following set of experiments, the training is conducted on a more powerful server, which has several *graphical processing units*, or GPUs, installed.

Additionally, we change the `python` framework. `Tensorflow` is great when it comes to large-scale projects and high performance. However, the level of detail it presents is often overwhelming and requires a lot of fine-tuning. Another popular code library is `PyTorch` [95]. This framework is less complex, and allows for quick and easy alterations to each training subroutine and to the associated hyperparaters. Furthermore, we find its methods easier to understand and to customize. For example, the `autograd` method provides an automatic way of obtaining the gradients necessary during backpropagation. This allows us to easily define custom loss functions.

Ultimately, the goal is to reduce the sum of squares between model and data. For the final set of experiments, we use a function that combines the loss in parameter and function space. Let $p = (p_1, \dots, p_{4m})^T \in [0,1]^{4m}$ be the set of normalized input parameters, and let $q = (q_1, \dots, q_{4m})^T$ denote the predicted values. Furthermore, let $s(x) \in \mathbb{R}^{1000}_{\geq 0}$ denote the spectrum generated from the parameters of x. Then, our custom loss function is defined by

$$\frac{\|p-q\|_2}{4m} + \beta \cdot \frac{\|s(p)-s(q)\|_2}{1000}, \tag{5.3}$$

where we choose $\beta = 0.1$.

Using this function limits our choice of model functions to Lorentzians, Gaussians, and pseudo-Voigt functions, as the partial derivatives of the loss functions with respect to each weight need to be considered. In the case of Voigt functions, computing $s(q)$ requires using the subroutine `scipy.special.voigt_profile`, which uses the FADEEVA function. The FADEEVA function itself uses a Taylor series to approximate the Gaussian error function $erf(z)$ [135]. However, these operations are not easily differentiable, and are therefore not supported by `autograd`. All computations associated with pseudo-Voigt functions are simple algebraic operations.

5.3.1 Architecture and Training

The architecture of this set of networks is as follows: The input layer is exactly 1,000 nodes wide. The data is then processed by a series of convolution layers with two different kernels. The two kernels create two data channels after the convolution. For the first convolution layer, both kernels have size 32. This layer is activated by the standard sigmoid $\mathrm{sig}(x)$. The outputs are then pooled by choosing the maximum of every two neighboring nodes, thereby halving the number of data points. This process of convolution, activation, and pooling is repeated using two convolution kernels of size 16. Since the size of the layer was reduced by a factor of four, both data channels contain 250 points.

This data is fed to a series of dense layers, each of which is activated by the standard simoid. These layers contain 320, 160, 80 and finally, 60 nodes. Finally, we *flatten* both data channels into a single channel of size 120. The output layer then computes $4m$ outputs. Only here does the number of peaks affect the architecture of these CNNs. Note that the output layer is not activated. For example, if we used a sigmoid for activation, extreme values near 0 or 1 would be less likely, as the sigmoid tends to move values closer to $\frac{1}{2}$. However, the training parameters are uniformly distributed. A schematic overview on the architecture is given in Fig. 5.14 in the case of $m = 3$.

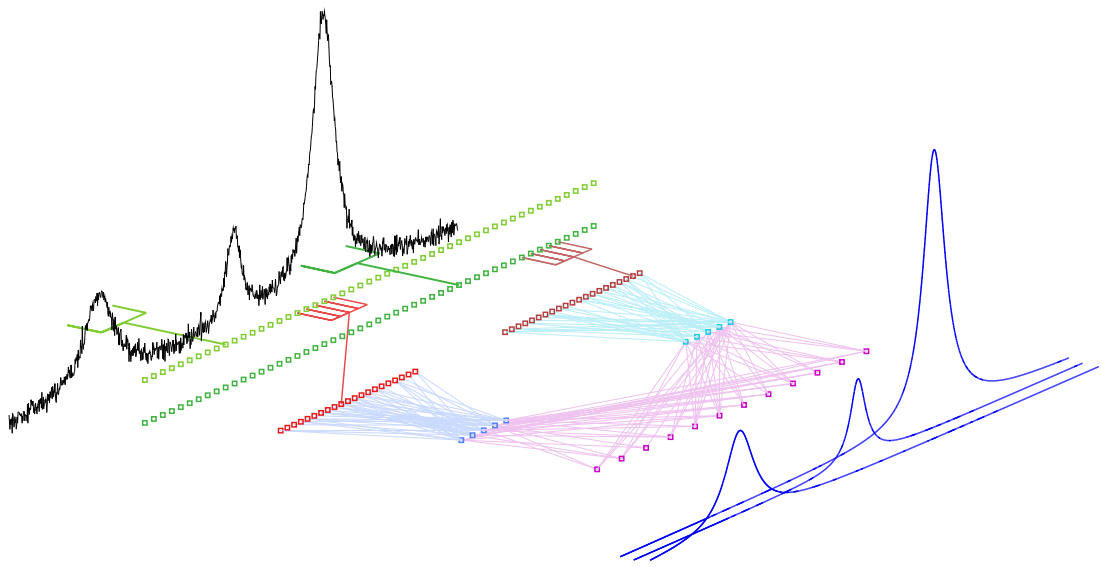

FIGURE 5.14: Schematic architecture of the CNN for $m = 3$ peaks. The input layer (black) is fed to, convolution layers (green) , max-pooling layers (red), dense layers (blue) and an output layer (violet), defining the peaks.

In total, five networks are trained to predict the parameters of $m = 1, \ldots, 5$ pseudo-Voigt functions. For each training data set, 100,000 spectra were generated using uniformly distributed parameters $c_i \in [0, 1], h_i \in [0.1, 1], w_i \in [0.00005, 0.5], \lambda_i \in [0.00002, 0.2]$. The spectra then are generated as the sum of m pseudo-Voigt functions with these parameters and added Gaussian noise with a standard deviation of $\sigma = 0.005$. Ultimately, the probability of sharp peaks or large lambdas, both weakpoints of previous networks, is not disproportionally increased. The validation data set contains 10,000 spectra that are generated in similar fashion.

To aid the training, the parameters are linearly transformed into $[0, 1]$. The training runs for 250 epochs and the training data is delivered in batches of size 25, for the optimization, the `adam` optimizer [63] is used with learning rate $\eta = 0.001$. The loss function is described in Eq. (5.3), where $\beta = 0.1$. All computations were performed on the GPUs on servers of the Institute of Mathematics in Rostock. This server has four *Nvidia GeForce RTX 2080 Ti* cards installed and we use `CUDA` [91] for efficient access to the GPUs.

5.3.2 Prediction Quality

To compare the five networks with each other, Tab. 5.2 - Tab. 5.5 display various metrics for parameter prediction, spectra reconstruction, and usablilty. Note that, with one noteable exception marked in red, the k peak network achieves the best results for k-peak spectra, marked in blue. Also note that the model quality rapidly decreases for spectra consisting of five peaks – in every metric. Therefore, training further networks for larger numbers of peaks was not pursued. While the final results may appear worse than those of previous experiments, the loss function from Eq. (5.3) on which the CNNs are trained incorporates

the reconstruction loss w.r.t. $\|\cdot\|_2$, and therefore, the MAE can be worse. The data was gathered by constructing 1,000 spectra, each consisting of k peaks, with added Gaussian noise with a standard deviation of $\sigma = 0.02$.

In addition to Tab. 5.2 - 5.5, Fig. 5.15 presents an example spectrum consisting of three peaks, as well as the reconstructed model spectra in comparison. In this case, the three-peak CNN has the best fit w.r.t. $\|\cdot\|_2$. Its lack of fit is ≈ 0.5711, while the reconstruction errors for the 1, 2, 4 and 5-peak networks amount to $\approx 4.2681, 3.5158, 1.2323$ and 1.4508, respectively. Additionally, the three-peak CNN also fits best w.r.t. d_w, where its error equals ≈ 0.0513, while the other errors amount to $\approx 0.5883, 0.2785, 0.0758$ and 0.1763 for the 1, 2, 4 and 5-peak networks, respectively.

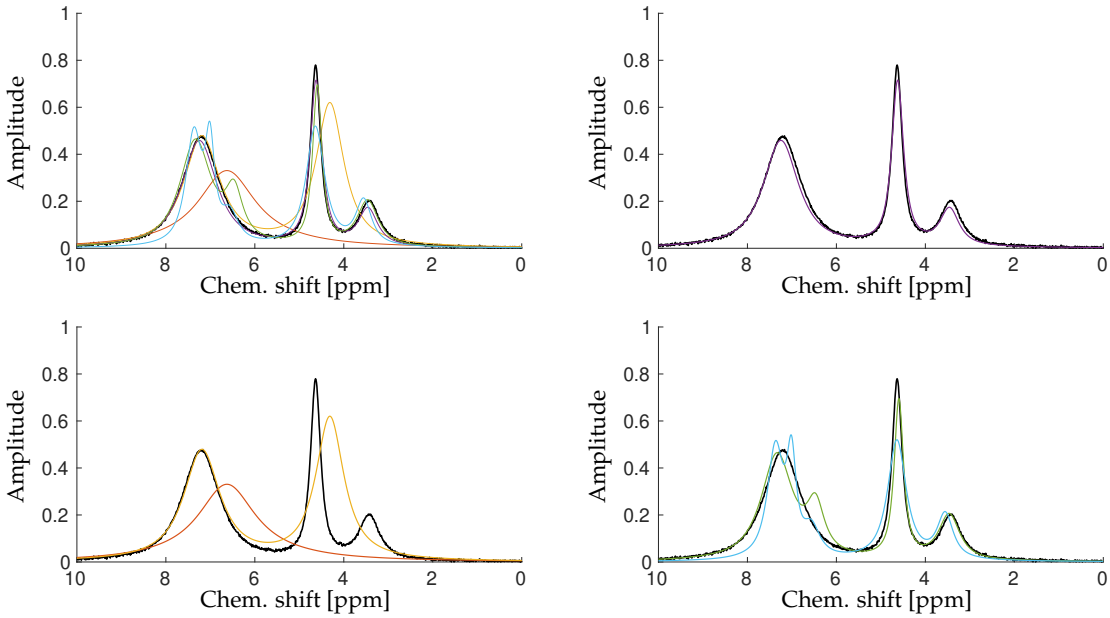

FIGURE 5.15: Comparing models from CNN predictions. From top left to bottom right, the data spectrum (black) is compared to all predictions, to the predictions with three peaks , one and two peaks, and four and five peaks.

According to Tab. 5.5, in many cases, the k peak network does not produce the best fitting predictions when k peaks are present. There are two types of mistakes: Underfitting and overfitting.

In our case, as displayed in the bottom left plot, the networks trained on fewer peaks are unable to properly reconstruct the spectrum. This fact is usually reflected in the reconstruction errors. Still, the two-peak CNN (yellow line) manages to properly model the leftmost peak. If a network trained on fewer peaks achieves a small lack of fit, it is likely due to peak overlaps. There, a less complex model suffices for a good fit. One challenge when implementing these CNNs into the optimization algorithm is to prevent that a previously modeled peak, has no parameters assigned in this spectrum due to underfitting and the lack of predicted parameters.

Conversely, networks trained on a greater number of peaks tend to assign meaningless peak parameters. In Fig. 5.15, this can be observed in the bottom right plot, where the four- and five-peak CNNs (green and blue lines) assign two and three peaks to the leftmost peak of the spectrum. This behavior is desirable, since it prevents overfitting and therefore a falsely improved reconstruction error. When one peak is fitted using m model functions, we experience the effect, that the peak heights of each individual model function are underestimated by an average factor of m, since the sum should equal the original single peak. So

	m-peak CNN				
MAE	$m=1$	$m=2$	$m=3$	$m=4$	$m=5$
c	0.0244	0.0213	0.0342	0.0395	0.0839
w	0.0515	0.0684	0.1125	0.1195	0.2029
h	0.0504	0.0807	0.1061	0.1079	0.2089
λ	0.1802	0.2474	0.2476	0.2536	0.2492
total	0.0766	0.1045	0.1251	0.1301	0.1862

TABLE 5.2: Comparison of mean average errors for each parameter type.

$\|s-s(p)\|_2$	m-peak CNN				
n-peak inputs	$m=1$	$m=2$	$m=3$	$m=4$	$m=5$
$n=1$	0.2803	0.5390	0.7945	0.8638	1.1587
$n=2$	2.6542	0.6103	0.7999	0.8576	1.2058
$n=3$	4.2244	2.3017	0.9740	0.9611	1.3707
$n=4$	5.6687	3.9773	1.8235	1.1789	1.6093
$n=5$	6.7025	5.2856	2.8935	1.6795	1.9191

TABLE 5.3: Comparing 2-norm reconstruction errors for n-peak spectra.

$d_W(s,s(p))$	m-peak CNN				
n-peak inputs	$m=1$	$m=2$	$m=3$	$m=4$	$m=5$
$n=1$	0.2687	0.3107	0.5417	0.5117	1.3306
$n=2$	0.9991	0.1271	0.1661	0.2202	0.4666
$n=3$	1.2406	0.3511	0.1357	0.1499	0.2730
$n=4$	1.5508	0.4927	0.1943	0.1327	0.2051
$n=5$	1.6569	0.5797	0.2668	0.1439	0.1831

TABLE 5.4: Comparing Wasserstein reconstruction errors for n-peak spectra.

# of best predictions	m-peak CNN				
n-peak inputs	$m=1$	$m=2$	$m=3$	$m=4$	$m=5$
$n=1$	890	79	15	16	0
$n=2$	142	582	176	94	6
$n=3$	14	194	399	359	34
$n=4$	2	36	263	597	102
$n=5$	1	7	187	582	223

TABLE 5.5: Comparing the frequency of best reconstruction out of 1000 n-peak spectra.

assigning only one of the parameter sets to a peak leads to an underestimation of the height parameter. Additionally, two peaks in close proximity and of similar height sometimes add up to a shape similar to a single peak. In these cases, the center parameter is slightly displaced. However, the four- and five-peak CNNs still manage to properly model the two rightmost peaks. Therefore, assigning these parameters when initializing the optimizer is permissible and usually not too bad. For the implementation, we only need to address the situation where a single peak of a previous spectrum is assigned multiple sets of parameters.

5.3.3 Stability Analysis

Starting from the spectrum in Fig. 5.15, one might ask how small changes in the input spectrum affect the output parameters. In theory, every CNN is a continuous function. However, there can be situations, where the derivatives of this function are quite steep, i.e., where small changes in the input spectrum, such as noise or other data impurities, greatly affect the outcome.

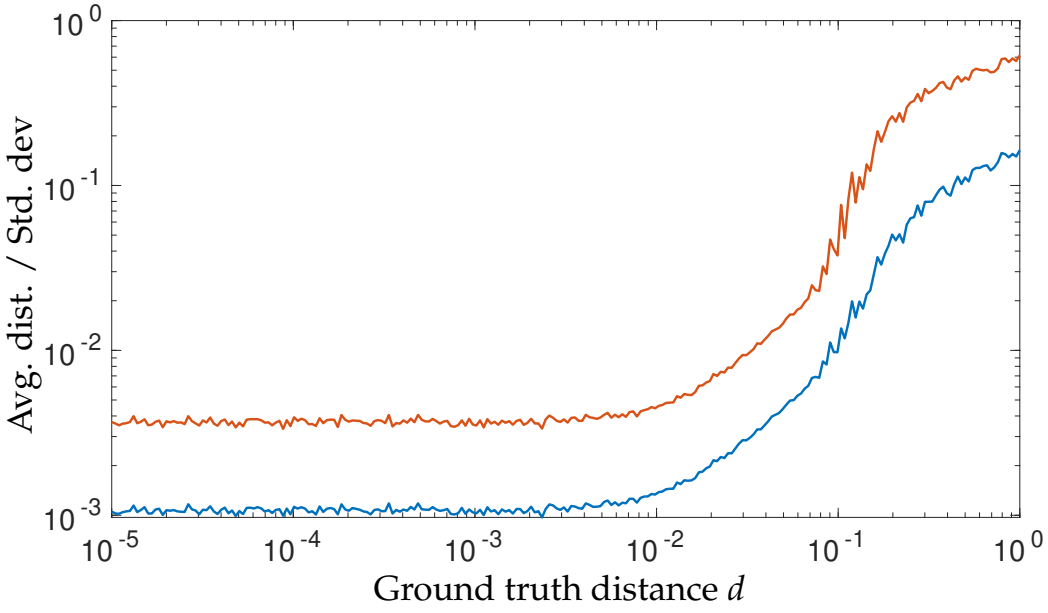

FIGURE 5.16: Average distance in parameter space (blue) and empirical standard deviation (red) of 100 CNN predictions, if the ground truth parameters have distance d from the spectrum in Fig. 5.15.

Fig. 5.16 displays how variations in the input affect the predicted parameters. Starting from the spectrum in Fig. 5.15, we adjust the ground truth parameters by a random vector of length d in parameter space. Then, we observe the average distance of the predicted parameters from the original outputs over 100 random iterations at each point and compute the empirical standard deviation of the distances. For each iteration, we added a different, Gaussian noise vectors. This explains the relatively large deviations for small parameter changes $d \in [10^{-5}, 10^{-3}]$, leading to slightly different predictions. Note that, within this range, the deviation of the distances is very small. This means that for two identical spectra with different, but identically distributed noise vectors, we expect that the parameters deviate by about 10^{-3}. For larger deviations of the parameters $d \in [10^{-3}, 10^{-2}]$, we see a gradual increase in distance. Interestingly, this increase does not exactly mirror the actual distance d. This may be due to the fact that small changes in c_i and w_i might be indistinguishable after the pooling layers reduce the dimension by a factor of 4, resulting in very similar outputs.

Finally, when $d \in [10^{-2}, 10^{0}]$, the average distance increases slightly. However, the individual distance values are increasingly scattered for larger d. This can be explained by the fact that spectra, whose parameters have a distance of 1 in parameters space also appear quite differently in function space. Additionally, a parameter vector that differs from the initial vector by 1 can contain nonsensical values, e.g., $w < 0, h < 0$. Nevertheless, for relatively small parameter changes $d \leq 10^{-1}$, we can guarantee that the neural network predictions do not behave erratically, but remain confined around their original predictions.

5.4 Implementation into the Spline-Based Algorithm

Implementing the final set of CNNs requires addressing two problems. First, the fixed input size only allows spectra with exactly 1,000 points to be input. Second, each CNN always predicts parameters for m peaks, even if the actual number of peaks present does not equal m.

5.4.1 Practical Problems

Harmonizing the Input Sizes:

The problem of a fixed input dimension is well-known in the literature. Typically, this issue is resolved by reshaping the input data to meet the required size. Another approach is to consider the input spectrum as a sequence. Recurrent neural networks are especially well-suited to deal with sequences of arbitrary length. One such example is the recognition of handwriting, [45, 88]. Alternatively, *deep-set networks* [136] use sets of arbitrary size as inputs. For example, this is used to extract chemical information by giving a set of spectral features to a deep-set network [126]. We choose the first, and arguably simplest method of reshaping the input data.

Most NMR spectra contain more than 1,000 data points. This implies the need for an automatic method of splitting any given spectrum into several subwindows, which we call *subspectra*. This method must run relatively quickly, and yield disjoint subspectra to prevent modeling the same area twice. For the same reason, the method needs to prevent splitting in areas, where peaks are present, as this can also lead to duplicate modeling. Lastly, we want to avoid subspectra containing too few or too many points to prevent a loss of accuracy caused by binning or interplation. Ultimately, the algorithm avoids subspectra with fewer than 200 and more than 5,000 data points.

Using the Predicted Parameters:

Since predictions are not very costly ($< 1\,\text{ms}$), we can easily use all available networks. However, the predicted parameters result in two additional problems. First, we need to determine which prediction to use. Ideally, this involves decision criteria which select the predictions of the CNN with the correct number of peaks[19]. Also, the prediction ultimately should lead to a quick nonlinear optimization. Second, there can be changes in the order of the peak centers due to peak crossings. The peak centers are predicted in ascending order, since the training data is structured in this manner. Therefore, we need an automatic assignment method for assigning the predicted peak parameters to the correct peaks to guarantee correct peak tracing.

[19] This is not an easy task, see Tab. 5.5

5.4.2 Suggested Solutions

We incorporate the CNNs into our routine in four steps. First, we split the spectrum into subspectra containing an appropriate amount of data points and peaks. Next, we predict parameters using all five convolutional neural networks while also taking into account extrapolation of previous parameters. Then, we apply a decision criterion which compares the sum of squares of all models. This criterion also heuristically predicts the correct number of peaks and finally decides which initialization is best. Lastly, we sort the peak parameters into the data structure we use for optimization. When multiplets are included in the optimization procedure, we then need to combine the predictions for sub-peaks of the same multiplet.

Subspectra Splitting:

To avoid multiple parameter predictions wherever possible, we aim to avoid splitting the spectra in areas where peaks reside. Using the results from the quick heuristic peak detection algorithm, see Sec. 4.2.4 [Variant 1], we identify areas in the spectrum larger than some threshold, $tol_{subsp} = 0.25\,\text{ppm}$, where no peaks are detected. Recall that this method is especially good at determining areas with no peaks. Initially, the spectrum is split at the midpoints between two of these peaks. Usually, this creates intervals which are too large or too small. Therefore, we then find subspectra of 5,000 or more data points. For any of these large subspectra, we look at the detected peaks within this subspectrum, and find the largest gap between each of the peaks and the subspectrum bounaries. Finally, we split this subspectrum in the middle of this gap. We repeat this process until no subspectrum with more than 5,000 points remains.

In some cases, the splitting process creates subspectra containing fewer than 200 data points. Therefore, we append these small subspectra to neighboring subspectra. To avoid creating subspectra with too many peaks, we aim to guess the number of peaks present. Note that the heuristic method described in Sec. 4.2.4 is not suitable for this task. Moreover, we are only interested in the peaks which are being optimized. Therefore, we linearly extrapolate the peak centers – and the subpeaks of any multiplet – of previous spectra to the current spectrum and then count the extrapolated peak centers. This would be an inaccurate heuristic, if not for the fact that we do not require accurate positions. Peak positions only need to be extrapolated into the correct subspectrum.

After determining the number of peaks per subspectrum, we check for subspectra containing ≤ 200 data points. If any such spectrum exists, and the predicted number of peaks in it and in one of its neighboring subspectra combined is less than five, both subspectra are combined. This only needs to be checked once per subspectrum, though. If a subspectrum with ≤ 200 data points is not combined with any of its neighbors the first time, it will also not be combined in any additional attempts. Note that this method has the possibility of creating subspectra of 5,000 or more points. However, in our experience, this is a rare occasion. After splitting, we harmonize the size of all subspectra to 1,000 points using cubic spline interpolation on 1,000 equidistant points inside the ppm range of each subspectrum. Fig. 5.17 displays the splitting process for spectrum $t = 140$ of D_2. Here, the spectrum composed of 2,827 data points (black) is split into four disjoint subspectra containing 805, 719, 341 and 962 data points from right to left. If the red crosses are $> tol_{subsp} = 0.25\,\text{ppm}$ apart, a split is defined at the midpoint between two consecutive peaks. In this case, the parameter extrapolation from previous spectra agreed with the number of heuristically found peaks per subspectrum, namely 3, 4, 1, and 1. For the sake of visibility, the first derivatives in Fig. 5.17 are multiplied by five.

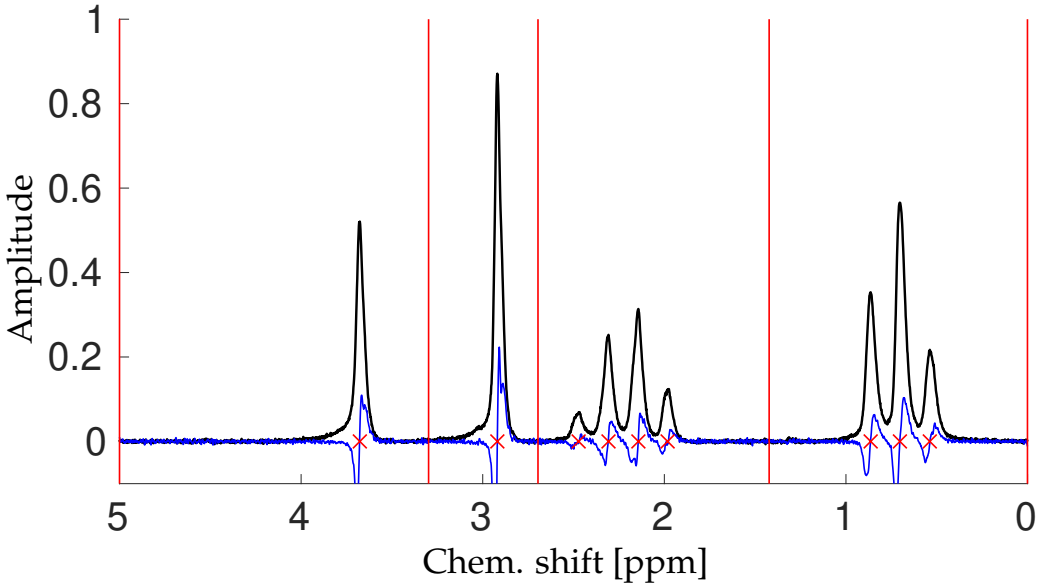

FIGURE 5.17: Subspectra of $t = 140$ from D_2. The original spectrum (black), the smoothed first derivatives (blue, exaggerated), the heuristically detected peaks (crosses) and the final splits (red lines) are shown.

Choosing Optimal Initializations:

Using the previously defined subspectra, we input each one into all available CNNs and obtain five possible sets of parameters per subspectrum. This process is shown in Fig. 5.15. These predictions are also compared to the null case, where no peak is found and to the previously used extrapolation method. To select the best of the seven initializations, we compare the lack of fit w.r.t. $\|\cdot\|_F$ of the corresponding models to the given data within the range of the subspectrum. Using the predicted number of peaks in this range, we can opt to give an advantage to the CNN with the corresponding number of peaks. In Sec. 6.6.1, we discuss the possible application of a scoring function using the Wasserstein distance.

Parameter Assignment:

Assume that, for a given subspectrum, the best prediction contains parameters for k peaks, whereas the predicted number of peaks is m. Let $c_1, \dots, c_m$ be the predicted center values and $d_1, \dots, d_k$ be the extrapolated center values, including the center values of multiplet subpeaks. Then, we define the matrix of center distances

$$M_d = \begin{pmatrix} |c_1 - d_1| & \cdots & |c_1 - d_k| \\ \vdots & \ddots & \vdots \\ |c_m - d_1| & \cdots & |c_m - d_k| \end{pmatrix} \in \mathbb{R}^{m \times k}.$$

We then assign the parameters to the peaks corresponding to the minimal remaining center distance. For example, if $M_d(i', j') = \min_{i,j} M_d(i, j)$, then the multiplet belonging to $d_{j'}$ is initialized using the parameters corresponding to $c_{i'}$. Then, the i'-th row and j'th columns are removed from M_d and this process is repeated until M_d has no more entries. This method works for all cases $m < k, m = k$, and $m > k$. There is one addition that is specific to multiplets. Several sub-peaks of a single multiplet can be close to predicted peak centers. Since

a multiplet only has one set of parameters for all its subpeaks, we count the number of initializations per multiplet. We then add subsequent predictions to the previously initialized values and finally divide by the count, thereby averaging over all predictions. Still, there are two exeptions to this rule. First, the peak heights need to be harmonize by dividing and multiplying with the corresponding binomial coefficients. Second, the distance parameter of the multiplet is initialized as the average distance between its predicted centers.

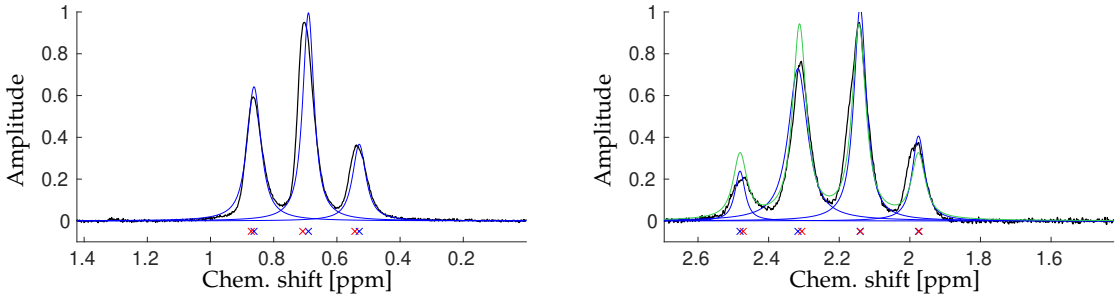

FIGURE 5.18: Parameter prediction in the first two subspectra of Fig. 5.17. The data (black), the CNN predictions (blue), predicted and extrapolated centers (blue and red crosses), and the averaged multiplet (green) are shown.

To better explain the procedure, Fig. 5.18 shows the predictions in the first two subspectra of spectrum, $t = 140$ of D_2. In the left plot, the three-peak network has the best-scored prediction. In this case, the triplet is treated as three individual peaks, and the predicted parameters are assigned based on the smallest distance between red and blue crosses. Thus, the leftmost peak is assigned first, followed by the rightmost peak and lastly, the middle peak. The right plot also shows this procedure. Here, the prediction of the four-peak network scored best. In this case, the quartuplet is internally treated as a multiplet. Thus, the red crosses represent the results of extrapolating its center and distance value. All sets of parameters are then assigned to the same multiplet, and the final initialization uses the average of all values[20].

Optimization Variants using CNNs:

For the variants defined in Sec. 4.2.4, we added two machine learning counterparts. THe first one improves variant 0, where only for the initializations, the aforementioned implementation is used. We call this pendant of variant 0, *variant -1*. Second, a variation of variant 2, where at first, subsystems are found using the same procedure. Then, for each of the subsystems, we use the initialization method from above. We call this variation of variant 2, *variant -2*.

5.5 Conclusion

In this chapter, we introduce a second type of approach for determining the model parameters, namely using neural networks. This approach is enabled by the specific definition of the peak parameters that carry information about peak shapes. Conversely, this allows for parameter prediction based on the spectrum alone. In the field of machine learning, this *feature recognition* problem is well-known and often solved using colvolutional neural networks (CNNs).

[20] With the exception of height and distance parameters.

Here, we compare different approaches of parameter prediction. First, in Sec. 5.2.1, we introduce a classification problem where each frequency channel is classified based on the presence or absence of a peak. Next, in Sec. 5.2.2, we propose architectures designed to tackle the more complex regression problem, and describe the experiments we conducted in great detail. In summary, the regression approach with its more sophisticated architectures yields significantly better results than the classification approach. However, this machine learning approach is not suited to deal with time series of NMR data, a topic that certainly calls for further investigation.

Since the CNNs are not suited for handling time series of spectra, they need to be integrated into another algorithm that incorporates the additional information provided by the second dimension. One option is to use the optimization routine from Ch. 4. Another option is to use some sort of sequential prediction of parameters. However, as we establish in Sec. 4.1.1, a sequential modeling approach is inferior to an approach that handles several subsequent spectra simultaneously, especially when peaks cross or overlap. Furthermore, the modeling quality of the CNNs is far from perfect, even though in most cases, the predictions are close in parameter space. Conversely, the spline-based optimization routine is computationally expensive, and sometimes converges in local rather than in global minima.

The advantages of both methods can be seen as complementary. On the one hand, we have a fast, but inaccurate prediction method that is unable to handle time series on its own. On the other hand, we have a slower, but more accurate optimization method, that is able to incorporate the information from the second dimension, though it sometimes converges unreliably. Therefore, in Sec. 5.4, we propose combining both methods by first predicting the peak parameters of the next spectrum to be optimized and using the predictions from the CNNs as well-informed initializations for the optimization routines. This hybrid approach is quicker than the spline-based optimization alone, it allows for the inaccuracies of the CNNs to be corrected, and it incorporates the additional information from the time series. Lastly, the initializations are closer to the global optimum, which increases the optimization stability.

The following Chapter 6 discusses these improvements and the detailed results of all methods on several constructed and experimental data sets. There, we also compare all methods in terms of runtime, modeling quality and optimization stability.

The implementation of our two main approaches can be accessed on GitHub, using the following link.

`https://github.com/CITlabRostock/nmr_timeseries_model`

Also, all CNNs that are ultimately used can be downloaded there.

Chapter 6

Applying Our Methods

After establishing the functionality of the presented methods in Chapters 4 and 5, it remains to demonstrate their practical use, the quality of their achieved models, and general algorithm usability through quanititative measures tested on both constructed and experimental data sets. A detailed presentation of all data sets can be found in App. B. First, we display the resulting models for the initial steps of the algorithm, i.e., peak and multiplet detection. Third, we demonstrate how to improve the parameter initialization and compare extrapolation and machine learning methods. Next, we provide a detailed comparison of all sub-variants and at the final models. Fifth, we examine how the resulting models behave when the initial assumptions are not met. Therefore, we consider cases where the data contains overlapping peaks, peak artifacts, or a *baseline*. Finally, we determine the relationship between the Wasserstein metric of the initialized model spectra, the runtime, the optimization stability, and how optimization using the Wasserstein distance compares to the more common sum of squares.

6.1 Comparing the Optimization Preliminaries

For the first part of this chapter, we present how the algorithms performed for all of the data sets described in App. B before the optimization phase and examine the resulting models during each part of the algorithm.

6.1.1 Peak and Multiplet Detection

After selecting an appropriate starting spectrum t_{init}, we measure the quality of peak detection similar to the classification in Sec. 5.2.1. We observe how many peaks in the spectrum were successfully detected, how many of the artifacts were falsly detected and how many multiplets were assigned correctly. The results of the peak and mutliplet detection are shown in Tab. 6.1. Note that each data set has its specific set of hyperparameters, such as ε, δ, etc., since no single set of hyperparameters can reliably predict peaks in all available data sets. Fig. 6.1 shows the procedure of peak picking for the data set D_2. Here, both the triplet and the quartuplet deviate from the specific height pattern. In general, the peak detection is very reliable, and there is some leeway when choosing the hyperparameters.

The complexity of D_3, e.g., its many peaks and artifacts in close proximity, can still be addressed using our method. However, since the moving peak is relatively wide and small, the sharper artifacts of the larger peaks possess more pronounced minima in the second derivatives, for example in $x \in [1.4, 1, 9]$. Therefore, we must either neglect both the artifacts and the moving peak – which is not feasible since the moving peak is chemically very interesting – or detect the moving peak as well as some additional artifacts. Fig. 6.2 shows the results of the peak detection in this interval. Note the detection of the relatively small artifacts.

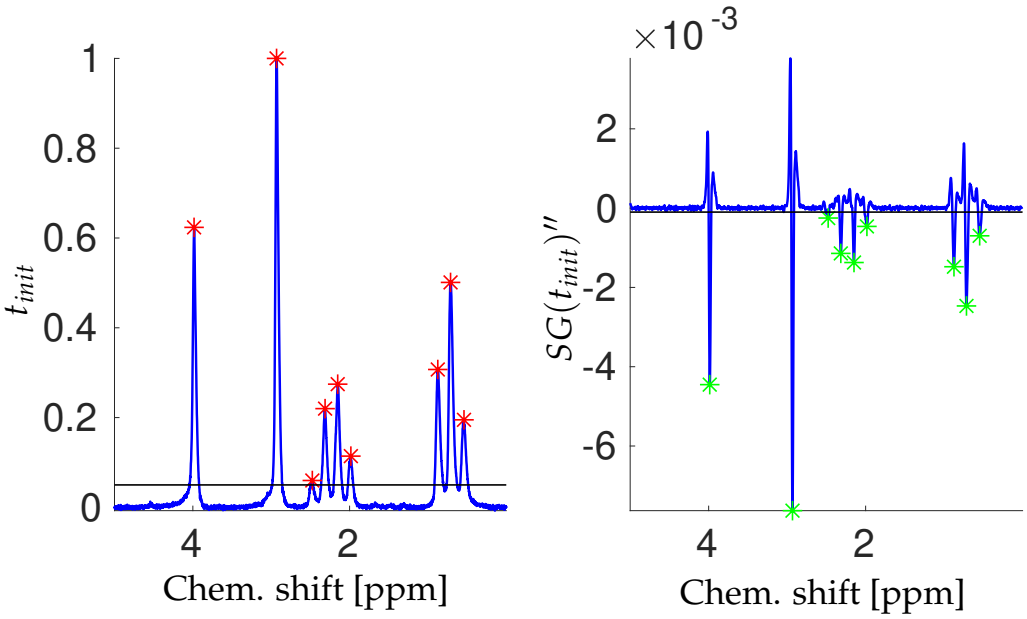

FIGURE 6.1: Peak detection for spectrum $t_{init} = 151$ of D_2.

In general, the results of peak and multiplet detection presented in Tab. 6.1 are satisfying. All peaks are detected, thus allowing the model recreate each one. In the case of D_3, the results can be improved by considering D_3 as two distinct modeling problems. Manually splitting the data set beforehand allows us to separate the relatively small peaks that should be modeled and the relatively large artifacts using different sets of hyperparameters, mainly ε, δ. Increasing tol_{vert} would also lead to 8/8 correct multiplet definitions. For the sake of comparability, we avoid the manual splitting of D_3.

Data Set	Peaks	Art.	Mult.	ε	δ	bw	tol_{ind}	tol_{vert}	tol_{hor}
D_a	3/3	none	3/3	0.01	-0.0001	11	5	0.05	0.05
D_b	3/3	none	3/3	0.05	-0.0001	21	5	0.05	0.05
D_c	25/25	none	7/7	0.025	-0.001	3	3	0.05	0.25
D_1	5/5	0/8	5/5	0.01	-0.001	11	5	0.05	0.25
D_2	9/9	none	4/4[21]	0.05	-0.0001	11	5	0.05	0.25
D_3	23/23	13/44	7/8[22]	0.00025	-0.000025	11	3	0.05	0.02

TABLE 6.1: Peak and multiplet detection results with hyperparameters.

6.1.2 Initial Models of the First Selected Spectrum

After successfully detecting peaks and multipelts, we compare the models for the first spectrum t_{init}. We use two metrics and measure thrice. First, we measure the local deviations of models from the data averaged over all multiplets, denoted by *average local*, after the windowed optimization. Second, we determine the quality on the entire spectrum composed by these local models, denoted by *compound local*. Finally, we measure the model quality on the entire spectrum after optimizing all parameters simultaneously, denoted by *global*. Tab. 6.2 compares the Wasserstein metric normed to $[0, 1]$ as well as the average euclidean norm of

[21] It is also possible to consider all nine peaks as singlets here.
[22] One duplet is improperly detected as two singlets.

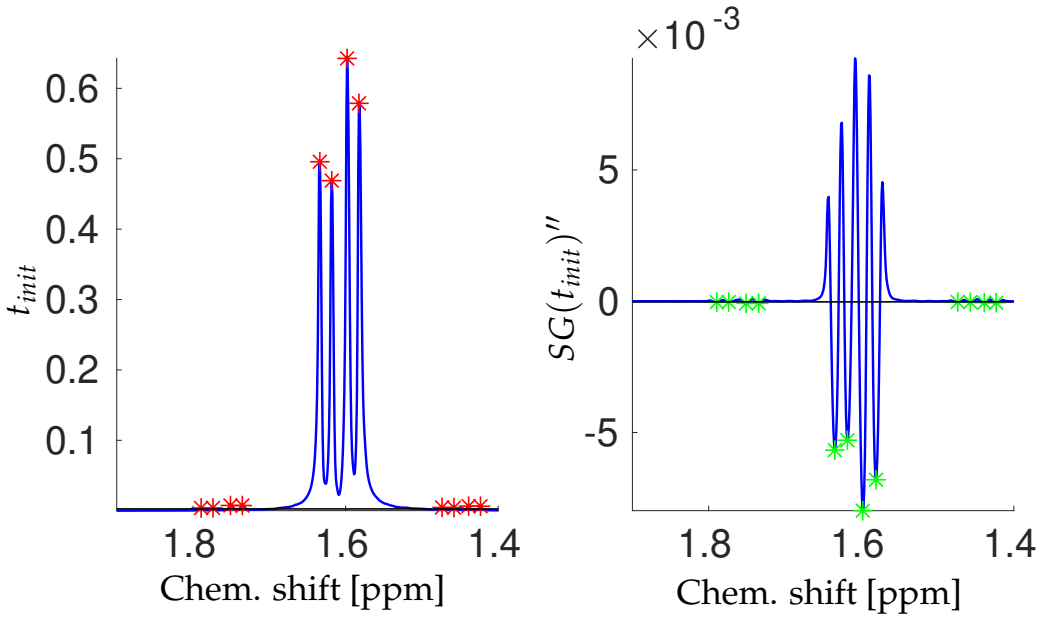

FIGURE 6.2: Peak detection in a subwindow of spectrum $t_{init} = 90$ of D_3.

all data sets. In general, both norms tend to decrease after the second optimization phase. In the case of Data sets D_a, D_b, D_2, this leads to a error reduction of $\approx 30 - 60\%$. Note that the errors vary between data sets due to the different noise levels and peak densities.

Data Set	Average Local		Compound Local		Global	
	d_W, [%]	$\|\cdot\|_2$, [‰]	d_W, [%]	$\|\cdot\|_2$, [‰]	d_W, [%]	$\|\cdot\|_2$, [‰]
D_a	1.2341	1.4950	1.1739	0.5497	1.0168	0.2052
D_b	2.1281	2.2932	1.0762	0.8654	0.4096	0.3727
D_c	0.6001	2.0802	1.3433	0.3297	0.3829	0.2714
D_1	0.9627	2.0118	0.2287	0.0403	0.2742	0.0352
D_2 w. Mult.	2.7830	3.5083	0.9252	0.3639	0.9447	0.2677
D_2 w. o. Mult.	2.3357	2.4955	1.7149	0.3675	0.8920	0.2131
D_3	3.5438	1.6317	0.2766	0.0490	0.2476	0.0246

TABLE 6.2: Comparing the models of t_{init} of all data sets.

We consider two cases for modeling D_2. The first one uses multiplets while the second one intentionally avoids them. Modeling D_2 without multiplets has the obvious drawback of increasing the number of parameters during optimization, which significantly increases the runtime. However, in terms of modeling quality, there is a clear advantage, see Fig. 6.3. Most noteably, the quartuplet and the triplet are more accurately modeled when multiplets are avoided.

Because we assume that the initial spectra are sufficiently selective, the modeling works well in general. However, in the case of D_3, where the peak and multiplet detection produces uncertainties, Fig. 6.4 examines the initial model more closely. It displays the entire starting spectrum and one subsection, where artifacts are falsely detected as peaks, and where a doublet is modeled as two singlets. Despite these minor issues, mostly senseable and meaningful models were assigned to the peak artifacts. The incorrect doublet, similar to D_2, only moderately increases the runtime.

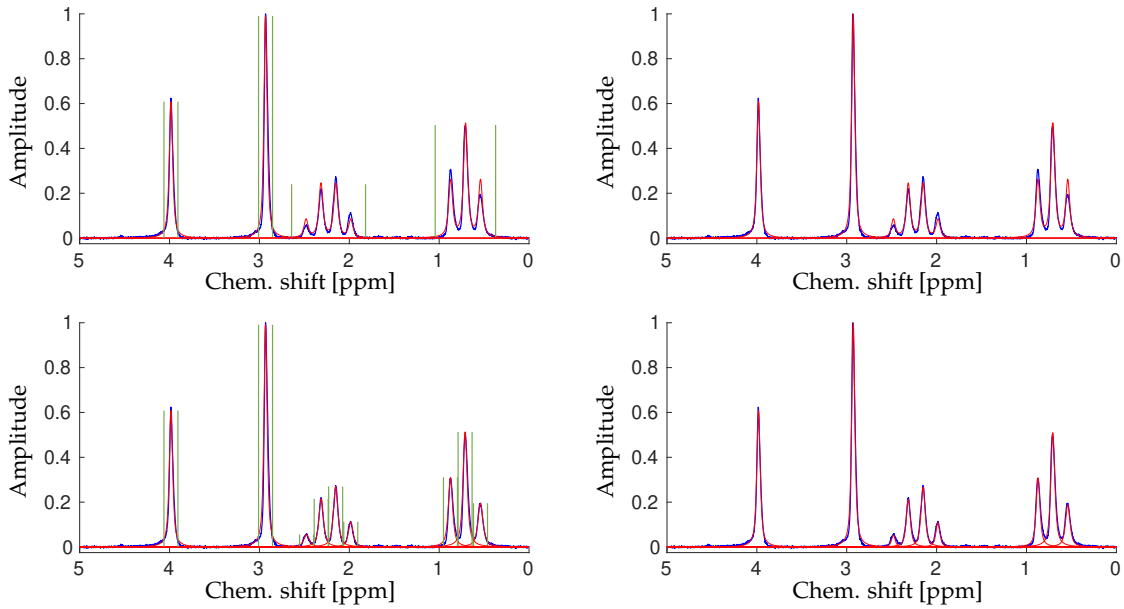

FIGURE 6.3: Modeling $t_{init} = 151$ of D_2 with (top) and without multiplets (bottom). Results after local (left) and global (right) optimization are shown.

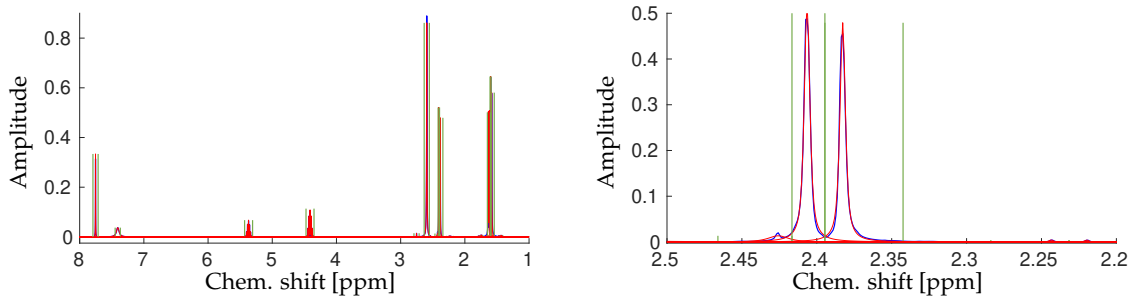

FIGURE 6.4: Modeling $t_{init} = 90$ of D_3 with a closer look at modeled artifacts and the incorrectly detected doublet on the right.

6.1.3 Heuristic Peak and Subsystem Detection

Next, we compare the results of the subroutine that splits the data set into subsystems. This subroutine requires an quick and effective heuristic peak detection, which is described in Sec. 4.2.4. Here, the most important aspect of this heuristic detection method is not to identify where the different peaks are, but rather to find areas of the frequency axis, in which no peaks are located for all spectra. Sometimes, no such area can be found, e.g,. for all of the constructed data sets. However, for the experimental data sets, senseable subsystems can be found.

Before applying this method, we define its goals. Visually, data set D_1 can be split into three parts, one containing the moving peak and two containing the stationary groups on both sides. Data set D_2 can be split into two subsystems, one containing the triplet and the other one containg all other peaks. Lastly, D_3 can be split into four independent parts. The first two contain the peak groups at $\approx 2\,\mathrm{ppm}$, the third subsystem contains the first septuplet, and the fourth contains all other peaks.

Fig. 6.5 displays the heuristic peak detection and the computed subsystems for all three experimental data sets. For D_1 and D_3, the heuristic peak detection clearly identifies areas, where no peaks lie. Even for the peak crossings of D_3, the heuristic method proves to be robust. Surprisingly, the artifacts which were superfluously detected in D_3 do not cause an influx of heuristically detected peaks. This may be due to the fact, that all artifacts in D_3 lie relatively closely to their corresponding peaks. Thus, D_1 and D_3 are successfully split into three and four disjoint subsystems, respectively. However, the method is less robust for D_2, where during peak crossings, only one of the two crossing peaks can be heuristically detected. This causes the algorithm to incorrectly identify sign changes in the first derivative caused by noise as peaks. Fortunately, these cases are rare enough that still, the data set can be meaningfully split. One implication of this issue is that ultimately, too many subsystems are detected. For D_2, we obtain five subsystems instead of the expected two. In practice, this does not cause any further problems, as subsystems without any peaks are not being optimized and thus, only marginally increase the runtime.

6.2 Initializing the Optimization Subroutines

Introducing machine learning to the algorithm primarily improves the initialization of the optimization subroutines. Predicting the peak parameters using the CNNs before optimization results in quicker and more stable convergence. Table 6.3 compares the initial predictions of machine learning (ML) to the extrapolation method (extrap.) used previously and illustrates the total runtime as well as the number of optimization steps following these initializations. In general, the machine learning approach improves both model quality and runtime. Note that for data sets D_b and D_2, the extrapolation method fails to trace all peaks, whereas the machine learning method succeeds. Also note that sometimes in this table, the runtimes appear to be too large. This is because the between data sets, the hyperparameters are chosen similarly for comparability reasons. Data set D_c is especially difficult for the machine learning variants since often, more than five peaks appear in any of the subspectra. Furthermore, the high peak density and the small number of 1,001 data points are problematic. Additionally, the changes in parameters are not drastic enough to cause the linear extrapolation to be inaccurate.

Fig. 6.6 compares the two initialization approaches for D_a and D_2. Note that parameter extrapolation results in initial models which are significantly far apart from the data for the moving peaks. These errors need to be dealt with by the nonlinear optimization which increases the runtime and decreases the stability.

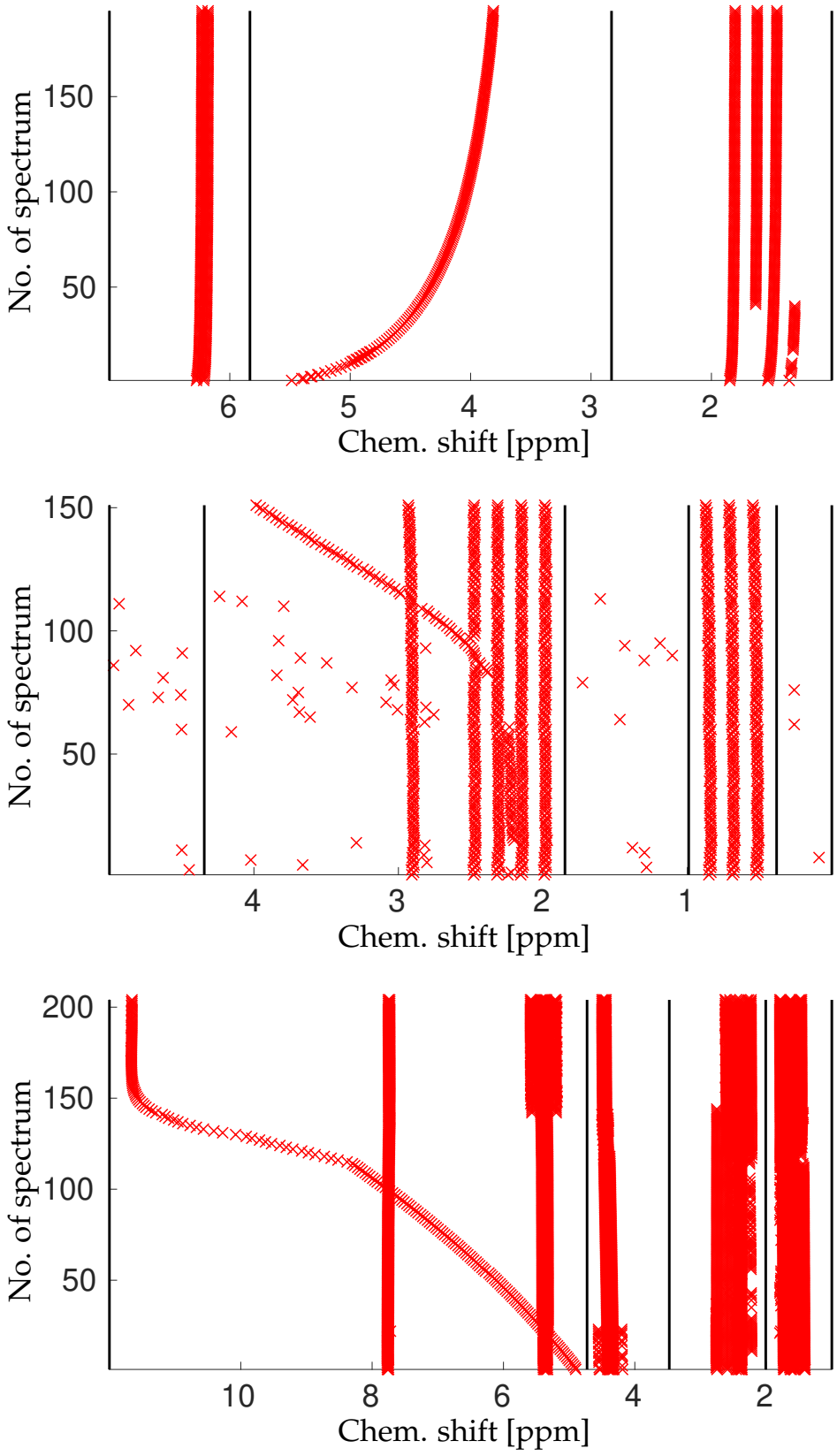

FIGURE 6.5: Heuristic peak detection and splitting of D_1, D_2 and D_3. Red crosses denote the detected peaks, black lines denote the subsystems.

Data Set	Initialization Errors				Avg. Computational Expenses			
	d_W, [%]		abs. $\|\cdot\|_2$, [‰]		Runtime [s]		Opt. Steps [#]	
	extrap.	ML	extrap.	ML	extrap.	ML	extrap.	ML
D_a	2.4051	1.7555	2.3857	0.8352	39.0	29.5	120.71	111.7
D_b	2.6658	1.8425	2.6658	0.6559	41.6	31.1	45.2	28.8
D_c	0.7841	1.5889	1.5757	2.6996	226.8	415.3	32.2	36.8
D_1	3.1554	0.8450	0.5510	0.1936	273.5	181.2	45.3	30.9
D_2 w. Mult.	2.3559	1.0850	1.2011	0.8194	837.5	319.8	101.1	39.3
D_2 w. o. Mult.	2.2490	1.1036	1.3653	0.9811	566.7	352.6	130.9	74.9
D_3	3.3043	3.2886	0.7713	0.5457	1280.8	640.8	174.2	96.8

TABLE 6.3: Comparing different initialization methods for all data sets. All values, except for the runtime, are averaged over $t \in T_{sel}$.

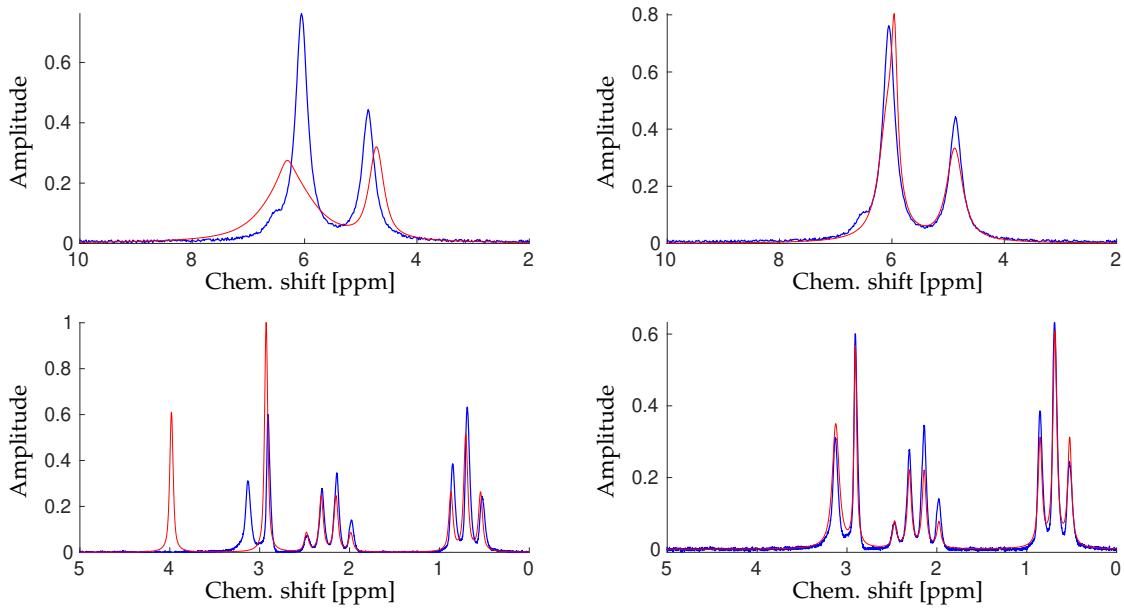

FIGURE 6.6: Extrapolation (left) and machine learning (right) initialization methods (red). Spectra $t = 31$ of D_a (top) and $t = 138$ of D_2 (bottom) are used.

6.3 Comparing the Algorithm Variants

In Sec. 4.2.4 and Sec. 5.4.2, we introduced several optimization variants after initially modeling t_{init}. Here, we compare all variants in terms of runtime and accuracy on the data sets D_a and D_2. We measure the computational expenses in terms of total runtime, and the number of optimization steps. Conversely, the model quality is measured using the relative Frobenius error and the normed Wasserstein distance. Table 6.4 displays all metrics for each of the seven variants and both data sets. Overall, we see that the negative variants that use machine learning perform better in terms of runtime and similar in terms of model quality. In the case of Tab. 6.4, the convergence criteria for the optimizer are chosen very strictly. The improved initializations of the machine learning methods allow us to choose more lenient criteria, thereby resulting in even faster runtimes. However, doing so would result in significant quality loss when using the other variants. Furthermore, in both D_a and D_2, the variants 1, 3 and 4 do not correctly trace all of the peaks. Due to the fact that the incorrectly traced peaks are small in comparison; the incorrect tracing is only partially reflected by the Wasserstein metric and not at all by the Frobenius norm. The *quick-and-dirty* variants

1, 3 and 4 perform especially poorly for data set D_a. Still, for the experimental data set, the results are surprisingly good, given the low runtime. However, since the most important moving peak is not traced correctly, their usefulness is limited.

Variant	Model Quality				Computational Expenses			
	d_W, [%]		rel. $\|\cdot\|_2$, [%]		Runtime [s]		Opt. Steps [#]	
	D_a	D_2	D_a	D_2	D_a	D_2	D_a	D_2
-2	1.03	1.56	4.55	19.58	29.2	118.6	103.0	32.5
-1	0.92	1.68	4.58	20.68	29.5	319.8	111.7	39.3
0	0.96	1.87	4.66	20.68	39.0	837.5	120.7	101.1
1	1.23	2.57	13.84	24.73	20.7	82.8	66.3	106.4
2	0.97	1.56	4.54	19.65	92.6	316.5	183.9	66.3
3	2.60	2.57	124.2	26.53	16.1	59.5	31.6	40.8
4	1.67	2.55	48.7	24.38	19.1	45.3	136.4	106.8

TABLE 6.4: Comparing quality and runtime for all variants. The number of optimization steps is averaged over the total number of optimizations.

6.4 Modeling Entire Data Sets

Finally, we present the results of the models for entire data sets. We compare the following metrics from the variant with the best results. First, the Wasserstein distance between the model and the original spectrum, normalized to $[0, 1]$ and averaged over all spectra. Second, the Frobenius norm of the error matrix relative to the original data. Third, the total runtime. Fourth, we count the number of steps that the nonlinear optimization function `lsqnonlin` conducted. Therefore, we measure the model quality with the former two measures and computational expenses with the latter two. All of these metrics for all data sets are displayed in Tab. 6.5. For a visual comparison of the modeling results, we refer to App. B.3. For all data sets except two, the best performance is achieved by the machine learning variants -1 and -2. For D_1, the best variant is variant 4, since the data set contains moving peaks, but no overlaps or peak crossings. Therefore, only for this data set, variants 3 and 4 excel. In the case of D_c, where many peaks but few data points are present, the machine learning predictions struggle to outperform the extrapolation. This is further amplified by the fact that the multiplets behave relatively tamely, i.e., the peak movements are slow, thereby mitigating the distadvantages of the extrapolation method.

Data Set	Model Quality		Computational Expenses		Variant
	d_W, [%]	rel. $\|\cdot\|_2$, [%]	Runtime [s]	Opt. steps [#]	
D_a	0.92	2.58	29.5	111.7	-1
D_b	1.35	17.20	35.1	27	-1
D_c	0.35	5.98	226.8	32.2	0
D_1	0.36	13.65	15.2	26.2	4
D_2 w. Mult.	1.56	19.58	118.6	32.5	-2
D_2 w. o. Mult.	1.51	16.64	357.0	34.1	-2
D_3	0.57	31.13	622.5	24.2	-2

TABLE 6.5: Comparing quality and runtime for all data sets. The number of optimization steps is averaged over the total number of optimizations.

Once the model for the entire time series is obtained, a further analysis of the data is simplified. For instance, peak integral values correspond to the concentration profiles of

chemical compounds. Without a proper model, obtaining these values involves numerical integration which is particularly challenging for overlapping peaks. However, after deconvoluting the spectra, the integrals for each spectrum can easily be computed using Eq. (2.14). Fig. 6.7 illustrates this procedure for D_2 and compares integrals obtained by using variants 0 and -1. Note that the variant using machine learning achieves chemically meaningful integrals over time, while the extrapolation variant does not. This can be explained by the fact that the blue and red peaks overlap within the range of $t \in [60, 90]$. Since their centers are very close together, an ambiguous model as described in Eq. (3.2) becomes possible. Variant 0 selects the meaningless option due to poor initialization.

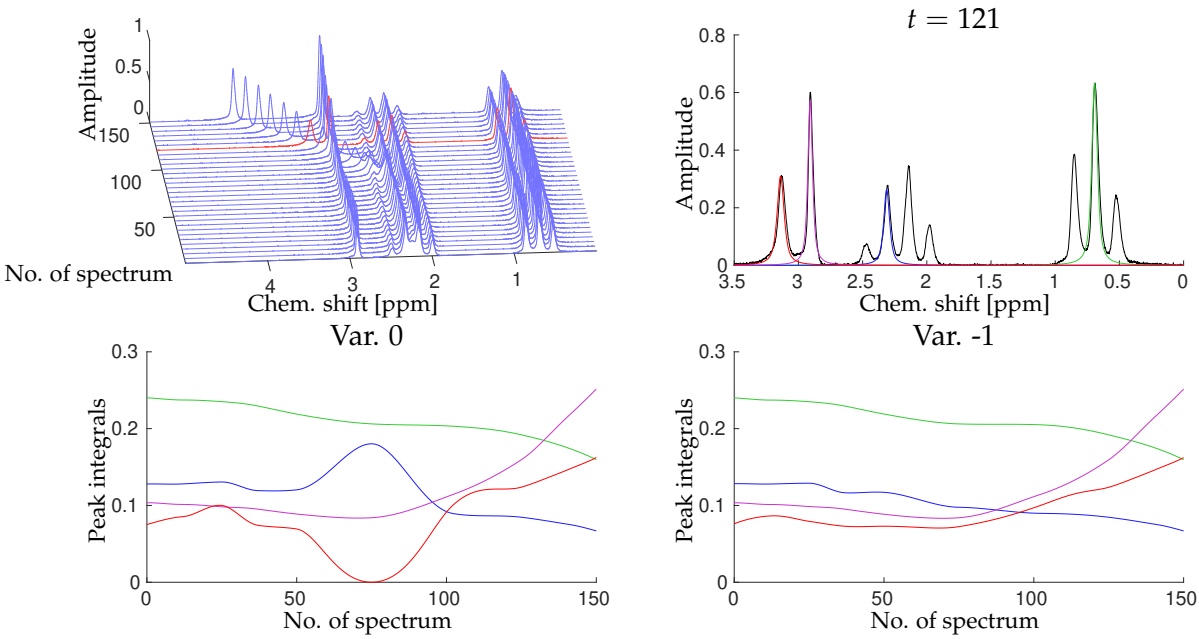

FIGURE 6.7: Computation of peak integrals of D_2 and comparison between variants 0 (bottom left) and -1 (bottom right).

6.5 Data Impurity

Initially, we use several assumptions about our data.

1. The signal-to-noise ratio (SNR) is sufficiently high.
2. The NMR spectra are properly phased and have no baseline.
3. In the time series, there is a spectrum in which all peaks are sufficiently separated.
4. Where artifacts are present, they do not interfere with the optimization routine.

The algorithm is designed with these assumptions in mind. Here, we demonstrate the behavior of our methods when one or more of these requirements is not met. Therefore, we violate each of these assumptions to varying degrees. First, we add Gaussian white noise with signal-to-noise ratios in the range of $[10^{-2}, 10^{-1}]$. For each of the different SNRs, we measure runtime and accuracy w.r.t. the Wasserstein and relative Frobenius metrics. The results are displayed in the top left of Fig. 6.8. For the second assumption, we first add artificial phasing errors in the range of $[0, \frac{\pi}{6}]$ with no additional baseline to D_a. The results are displayed in the top right of Fig. 6.8. Next, we add baselines with amplitudes in the range

of $[0, 0.2]$ by adding randomly shifted sine functions with random frequencies to D_a. Lastly, we add both phasing and baseline errors. The bottom row of Fig. 6.8 displays the results of the latter two experiments. Note that a decrease in runtime most likely corresponds to a nonsensical optimization, since when no significant improvements can be achieved, the optimization stops prematurely.

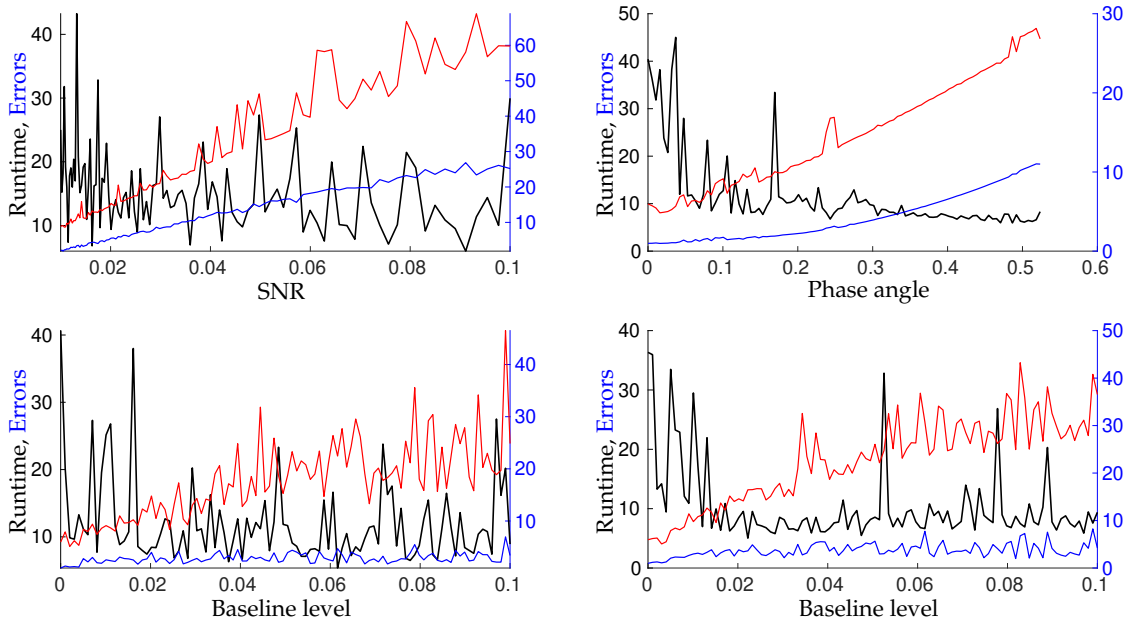

FIGURE 6.8: Relationship of runtime (black lines) and accuracy w.r.t. the Wasserstein (blue) and Frobenius (red) metrics with added impurities.

Third, if no spectrum is present, where all peaks can be detected, then the optimization cannot fully model the data set. Fourth, if artifacts are detected by the peak detection routine, then the model suffers from overfitting. This can also result in nonsensical models. Both of these assumptions can be generalized. We require a successful peak detection. Otherwise, two cases arise, when too few or too many peaks are detected. Typically, these cases have different implications.

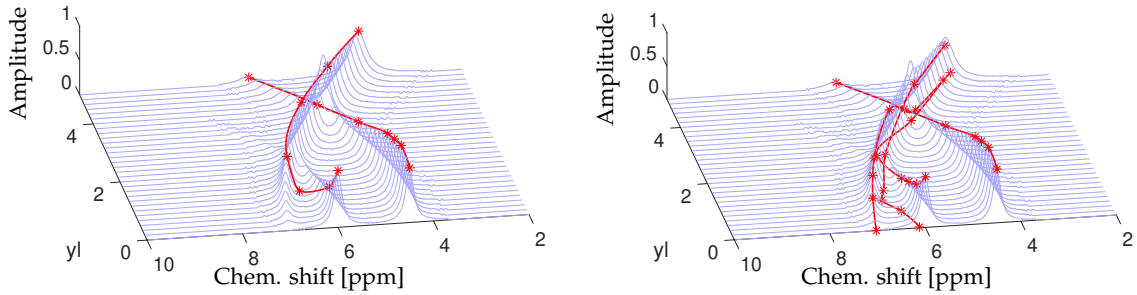

FIGURE 6.9: Optimization with fewer (left) or more (right) detected peaks.

Fig. 6.9 illustrates the two cases using D_a as an example. Here, we manually define the incorrect number of peaks as one too few and one too many. For the two-peak case, the optimization algorithm managed to model two of the three peaks quite robustly. The peak tracing only fails when the leftmost peak vanishes for $t \in [0, 1]$. Nevertheless, the entire procedure takes only 3.43 seconds, while achieving an average Wasserstein distance of 1.8%

and a comparatively large relative Frobenius error of 17.09%, which is related to missing a peak. Conversely, the four-peak case also does not trace the peaks correctly. Furthermore, this leads to an underestimation of peak heights for each of the model functions but to a correct sum overall. Due to the increased number of parameters, here the runtime equals 16.57 seconds, and the Wasserstein and Frobenius errors amount to 0.94% and 6.57%, respectively.

Similar issues can arise from detecting peak artifacts. Take D_3, where some artifacts are detected. Fig. 6.4 shows that the articfacts in close proximity are initialized too widely. While this type of modeling error can be propagated during further optimization steps, the larger optimization routines tend to hide unnecessary model functions by reducing their heights.

6.6 Utilizing the Wasserstein Distance

In Sec. 4.2.4, we established that the Wasserstein distance can be used as an objective function and discussed possible implementation strategies. There is another use case. When we compare different initial predictions, e.g., for the machine learning approaches, we use the Frobenius norm to measure the model quality. However, several observations led us to believe that the Wasserstein metric of the initial spectrum is a good predictor on the remaining runtime of the subsequent optimization. If so, this fact could be utilized by combining Frobenius and Wasserstein metrics to improve the quality measure, thereby further improving the initializations.

6.6.1 Predicting Optimization Times

Previously, Tab. 6.2 suggested that improving the Wasserstein distance for the initial parameters also reduces the runtime. While that table considers improving the initializations altogether, here we examine the relationship between the Wasserstein distance of initial spectra and the number of optimization steps, as well as the runtime for each subsequent optimization routine. Fig. 6.10 displays sets of points (x, y), where x represents the initial Wasserstein distance d_W, normalized to $[0, 1]$ and y represents the number of optimization steps, or the runtime in seconds. The top left plot shows all data points for D_a and D_2, while the other three represent the data points when optimizing D_a (top right), D_2 with (bottom left) and without multiplets (bottom right). We cannot observe a clear relationship by looking at the scatter plots alone.

To measure the relationship between the measured quantities, we compute Pearson's, Spearman's and Kendall's correlation coefficients, which we denote by ρ_P, ρ_S and ρ_K, respectively. For their definitions and further explanations, we refer to [41, Ch. 11] and [29, Ch. 9]. Tab. 6.6 displays all values for different subsets of points. Since the runtime and the number of optimization steps varies between different variants, we classify the data into similar variants. In general, we observe no significant correlation between the runtime and the initial Wasserstein distance. For some variants and data sets, such as D_2 without the use of multiplets, an increase in wasserstein distance leads to a noticeable increase in runtime for variants -1 and 0. However, for other data sets, even negative correlations can be measured. Conversely, for the number of optimization steps, all three coefficients show a mild positive correlation for almost all combinations of data sets and variants. Therefore, the Wasserstein distance can be considered an indicator of optimization stability through the number of optimization steps, rather than an indicator of runtime.

6.6.2 Optimizing the Wasserstein Distance

When choosing the Wasserstein metric as an objective function, several issues have to be addressed. First, `lsqnonlin`, or other least squares optimizers, cannot be used. Therefore,

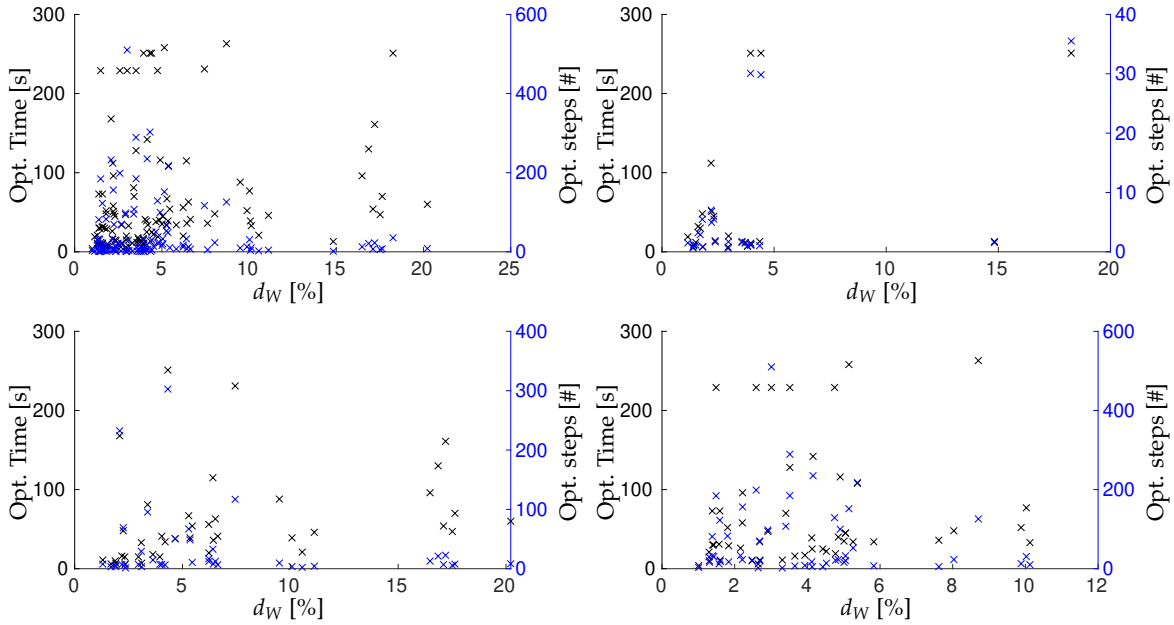

FIGURE 6.10: Relationship between Wasserstein distance and optimization expenses.

we select `fmincon` for optimization purposes. This impairs the optimization, as this routine is more general and therefore slower. Additionally, the comparison between Frobenius and Wasserstein objective functions is complicated due to different optimizers requiring similar, but not identical hyperparameters. While optimizing the Wasserstein distance is more stable, see Fig. 4.4, it seems to be less accurate in some cases. For example, the Frobenius norm aims to fit peaks exactly on the maxima, whereas the Wasserstein metric averages all errors in terms of peak position. This leads to strange optimal fittings. Fig. 6.11 illustrates this for the example of one doublet. If a single peak is fitted to this data, the Frobenius objective function forces one of the doublet's peaks to be modeled. The Wasserstein objective function averages the integral mass on the left and on the right side, and its unique optimum models none of the peaks.

In practice, the different, less fine-tuned optimization function and modeling inaccuracies manage to negate any advantage the reduced optimization dimension might result in. Tab. 6.7 compares both model quality and runtime when optimizing both objective functions for two of our data sets. Not only does it appear as if the Wasserstein objective function has no advantage, but using it is actively worse in both runtime and accuracy w.r.t. the Frobenius norm. As to the Wasserstein metric itself, the improved Wasserstein objective function manages to improve the Wasserstein errors. However, this does not come as a surprise when the same metric is minmized in the first place. Notably, computing the Wasserstein metric is surprisingly expensive, as computing it takes up to 60% of the total runtime, even though it can be computed in linear time. In conclusion, we recommend to avoid using the Wasserstein metric as an objective function.

6.7 Conclusion

In this chapter, we thoroughly test all of the methods presented in this thesis. First, we conclude that the peak and multiplet detection work well for all data sets and can generally be applied to all sufficiently selective data sets, see Tab. 6.1. Both peak and multiplet

Data Set	Variants	Runtimes [s]			Optimization Steps [#]		
		ρ_P	ρ_S	ρ_K	ρ_P	ρ_S	ρ_K
D_a	Combined	0.5020	0.2139	0.1449	0.4233	0.1723	0.1220
	-2, 2	0.6435	0.3810	0.2143	0.5523	0.4148	0.2646
	-1, 0	-0.1145	0.1399	0.1212	-0.1311	-0.0210	-0.0153
D_2 w. Mult.	Combined	-0.1913	0.0728	0.0308	0.2853	0.5940	0.4459
	-2, 2	0.0371	-0.0456	0.0058	0.4016	0.3731	0.2765
	-1, 0	-0.0493	0.0769	0.0303	0.2702	0.2028	0.1212
D_2 w.o. Mult.	Combined	-0.0988	-0.0567	-0.0188	0.1310	0.2980	0.2117
	-2, 2	0.0999	0.3684	0.2632	0.1937	0.6260	0.4656
	-1, 0	0.3436	0.4391	0.3158	0.2618	0.3820	0.2521
Combined	Combined	-0.1202	0.1046	0.0651	0.2172	0.4140	0.2889
	-2, 2	-0.0454	0.2105	0.1304	0.2057	0.4971	0.3541
	-1, 0	0.2048	0.3202	0.2283	0.2748	0.3005	0.1979

TABLE 6.6: Comapring correlation coefficients of data from Fig. 6.10.

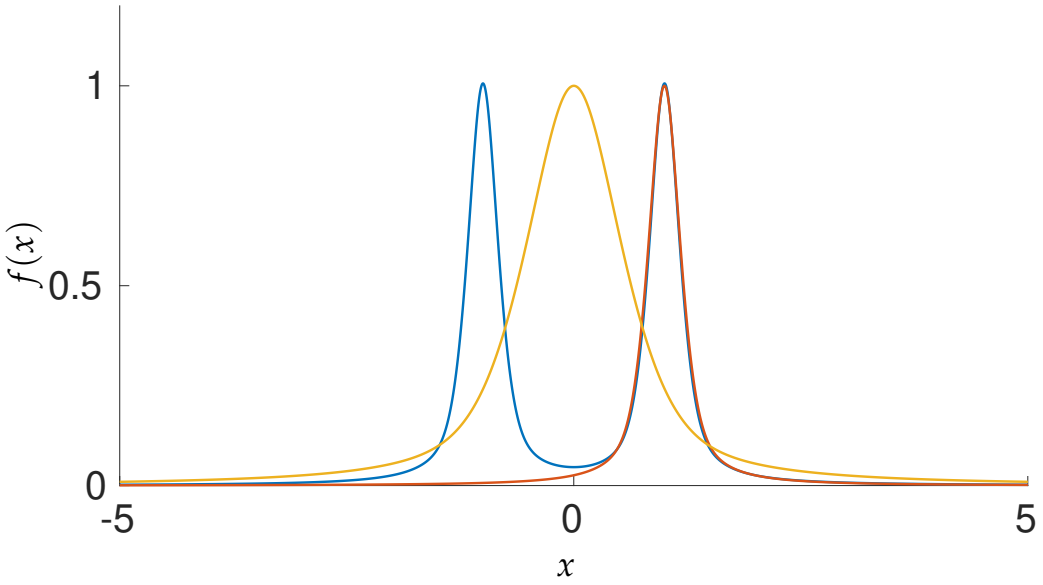

FIGURE 6.11: Doublet (blue), optimal Frobenius (red) and Wasserstein (yellow) peaks.

Data Set	Obj. Function	Runtime [s]	Opt. Steps [#]	d_W, [%]	rel. $\|\cdot\|_2$, [%]
D_a	$\|\cdot\|_2$	25.2	120.7	0.96	4.67
	d_W (all)	163.9	255.6	0.87	15.8
	d_W (nonlin.)	55.6	110.9	0.88	21.31
D_2 w. Mult.	$\|\cdot\|_2$	111.9	29.3	1.55	19.75
	d_W (all)	1502.2	251.8	2.16	39.72
	d_W (nonlin.)	672.1	159.6	1.31	23.52

TABLE 6.7: Quality and runtime for Wasserstein-based optimization.

detection are robust methods and can be computed in seconds, even in more complex situations where peaks overlap. The subsequent optimization ensures meaningful models for the initial spectrum t_{init} by combining local and global optimizations. The local parameters facilitate the spectrum-wide optimization by serving as well-defined initializations.

The hybrid method that combines the spline-based optimization with parameter prediction using CNNs leads to more stable optimization, while improving both the runtime and the model quality, see Tabs. 6.3, 6.4, and 6.5. These improvements are enabled by using the predicted parameters as well-posed initialization for the optimization. In most cases, the initial values are close to the assumed ground truth, thereby reducing the possibility of convergence within a local optimum, and thus ensuring a stable optimization. The optimization then only needs to conduct a few steps to correct the small deviations caused by the neural networks. Generally, variants -1 and -2, which use neural networks to initialize the optimization, are quicker while achieving models of similar quality. Furthermore, the increased stability allows us to choose less strict convergence criteria for the optimization, thereby decreasing the runtime further.

In Sec. 6.5, we demonstrate that the modeling process is stable, and produces meaningful models, if the initial assumptions are not entirely met. For instance, an increase in noise level results in a linear increase in model error, primarily due to the increase in noise. However, the underlying models remain similar to those with no added noise.

Finally, in Sec. 6.6, we observe that the Wasserstein distance of the initial models does not directly correlate with the optimization runtime, although it is a predictive measure of model quality. Rather, it may correlate with the number of steps the optimizer needs to perform, see Tab. 6.6. If this correlation can be further validated, then the Wasserstein metric can be used to select the best initialization in terms of optimization runtime or stability. For now, only the Frobenius norm is used. Additionally, we find that the Wasserstein distance is not a suitable objective function either, due to both theoretical reasons, as displayed in Fig. 6.11, and practical reasons, e.g., the different optimization algorithms and comparatively expensive computation of d_W.

Chapter 7

Summary and Outlook

Time series of NMR spectra can often be considered *big data* because analyzing several hundred spectra with tens of thousands of data points is necessary for understanding the underlying chemistry. Typically, big data analysis requires some kind of dimensionality reduction, as matrices with millions of elements can only be understood through low-dimensional representations. In other fields of spectroscopy such as optical spectroscopy, this is achieved using nonnegative matrix factorization methods. However, in NMR spectroscopy, the pure component spectra change over time, necessitating a different type of analysis. Usually, the spectra are recreated using a specific type of model, and this modeling problem is typically defined as an optimization problem.

In Chapter 3, we answer the question of when ambiguous solutions to the modeling problem are possible. Assuming that NMR spectra consist of linear combinations of a single type of model function, then the problem of finding the correct number of peaks and their parameters has only one solution in the continuous case, i.e., when the spectrum is given for all $x \in \mathbb{R}$. In the discrete case, we determine upper and lower bounds for the required number $k_{\mathcal{F}}(m,n)+1$ of data points[23], that ensure uniqueness, for fixed numbers of model functions (m,n). In the case of Lorentzians, we can determine the value for $k_L(m,n) = 2m+2n-2$. For Gaussians, we estimate k_G and propose the conjecture that the lower bound is tight, i.e., $k_G(m,n) = k_L(m,n)$. The proof of this conjecture remains an open problem. Similarly, establishing stricter bounds for $k_{PV}(m,n)$ and $k_V(m,n)$ is a topic for further investigation.

In this thesis, we suggest two main approaches for obtaining the model parameters. In Chapter 4, we propose a pure nonlinear optimization routine that treats each parameter as a continuous and smooth function over time. This allows us to optimize the parameters on only a few spectra T_{sel}, while interpolating the parameters in between T_{sel}. This approach has two advantages. For one, compared to optimizing all parameters, it reduces the optimization dimension and thus the complexity. Second, interpolation using cubic spline functions forces the parameter functions to behave smoothly and continuously as well, leading to chemically meaningful models. This approach is not only more stable than sequential spectra optimization, but it is also more efficient, as optimization of every parameter is not required. Further improvements to both runtime and stability can be achieved. First, the so-called J-coupling is implemented by modeling multiplets rather than single peaks. Depending on the data set, this reduces the optimization dimension by about $40\% - 60\%$. It also reduces the number of local minima, thereby increasing the chances of converging to the global optimum. Second, optimization variants are introduced, that split the optimization of all types of parameters into multiple parts, and that can find independent subsystems of peaks within the time series. Still, two main drawbacks remain for the first approach. First, despite the improvements, the expensive computation leads to long runtimes. Second, the

[23]The number $k_{\mathcal{F}}(m,n)$ represents the maximum number of points at which an ambiguous decomposition exists.

nonlinear optimizer is not guaranteed to converge. This sometimes results in chemically meaningless models.

In Chapter 5, we introduce a second approach. There, we utilize the fact that the model parameters carry intrinsic meaning to predict them by simply observing structure of the spectra. This allows for an approach based on machine learning. We trained convolutional neural networks to reliably detect peak parameters. This prediction method is very time-efficient, e.g., parameter detection on a single spectrum takes less than 1 ms. However, this neural network method has two main drawbacks. First, it does not incorporate information from the second dimension into its predictions. This implies that peak movements cannot be reliably traced, as the order of peaks is not consistent between spectra. Additionally, this leaves room for ambiguous solutions. Second, the average prediction quality is fixed after training, and is sometimes insufficient. In the future, recurrent neural networks (RNNs) can be used to handle entire time series as inputs, reducing the reliance on comparatively slow nonlinear optimization.

Since these two main approaches have complementary advantages, we propose combining them. In this combined method, we use the convolutional neural networks to first approximate the peak parameters, which are then used to initialize the optimization from the first approach. Because the initialization is sufficiently close to the ground truth, the optimizer only needs to fine-tune the predicted parameters. This forces convergence within the global minimum in a reasonable amount of time. Conversely, this method enables us to correct the predicted parameters and to assign them to the correct peaks unambiguously.

In the future, this method can be evolved further to model reaction kinetics. An automatic method of grouping the different multiplets into several chemical compounds would allow us to quickly compute the concentration profiles. These profiles are essential for creating kinetic models, which are essential for reaction monitoring.

Another open problem concerns data acquired by *benchtop* spectrometers with lower field strengths. Thisresults in wider peaks, increased peak overlap, and a reduced selectivity of spectra. In Fig. 5.12, we demonstrate that benchtop spectra can indeed be modeled by using CNNs. However, as the field strength decreases further, the modeling quality of our networks decreases. As benchtop spectrometers are typically smaller and more robust, they can be used to measure reactions *in situ*. One possible application of our methods is online reaction monitoring of benchtop data, where the modeled results can be used to control the reaction occurring in parallel.

All of the methods described are tested and compared in Chapter 6. Therein, we determine runtimes, stability, and model quality of all variants and approaches. Additionally, we examine the influence of noise, a baseline, and other data impurities on our methods. In summary, the proposed hybrid method improves upon the spline-based optimization algorithm in terms of runtime, stability, and model quality in several measures. Our proposed variants and the implementation of multiplet modeling further reduce the optimization dimension, enabling larger data sets to be modeled in a reasonable amount of time. Finally, the code for all of the presented methods can be downloaded from GitHub.

`https://github.com/CITlabRostock/nmr_timeseries_model`

Appendix A

Unused Lemmas

In the first appendix, we present the proof of those lemmas, whose proofs are either irrelevant, inelegant, or imprecise.

A.1 Fourier Transformation of a Lorentzian

The first lemma of this appendix is well-known in complex analysis and probability theory, as it gives an explicit formula for the complex integral that defines the characteristic function of a Cauchy distribution [96, App. D]. We study the integral in a slightly different form.

Lemma A.1. *Let $w, \omega \in \mathbb{R}$ and $w > 0$. Then,*

$$\int_{\mathbb{R}} \frac{1}{1+x^2} e^{-\mathbf{i}x(w\omega)} \mathrm{d}x = \pi \cdot e^{-w|\omega|}. \tag{A.1}$$

Proof. Let $R > 0$. First, we consider the two closed complex paths displayed in Fig. A.1

$$\Gamma_1(t) = \begin{cases} R \cdot \mathrm{e}^{\mathbf{i}t}, & t \in [0, \pi) \\ -R + (t - \pi), t \in [\pi, \pi + 2R] \end{cases}$$
$$\Gamma_2(t) = \begin{cases} R \cdot \mathrm{e}^{-\mathbf{i}t}, & t \in [0, \pi) \\ -R + (t - \pi), t \in [\pi, \pi + 2R] \end{cases}$$

and their corresponding path integrals

$$I_1 = \int_{\Gamma_1} \frac{\mathrm{e}^{-\mathbf{i}x(w\omega)}}{1+x^2}$$
$$I_2 = \int_{\Gamma_2} \frac{\mathrm{e}^{-\mathbf{i}x(w\omega)}}{1+x^2}.$$

Note that both paths encircle an isolated singulatiry located at $\pm\mathbf{i}$.

Let $\omega < 0$. Then, the integral

$$\int_{\gamma_1} \frac{\mathrm{e}^{-\mathbf{i}x(w\omega)}}{1+x^2}$$

on the upper arc γ_1 can be estimated. For x on Γ_1, we have $\mathrm{Im}(x) \geq 0$ and therefore

$$\begin{aligned} \left|\mathrm{e}^{-\mathbf{i}x(w\omega)}\right| &= \left|\mathrm{e}^{\mathrm{Im}(x)(w\omega)} \cdot \mathrm{e}^{-\mathbf{i}\mathrm{Re}(x)(w\omega)}\right| \\ &\leq 1, \end{aligned}$$

due to $w\omega < 0$.

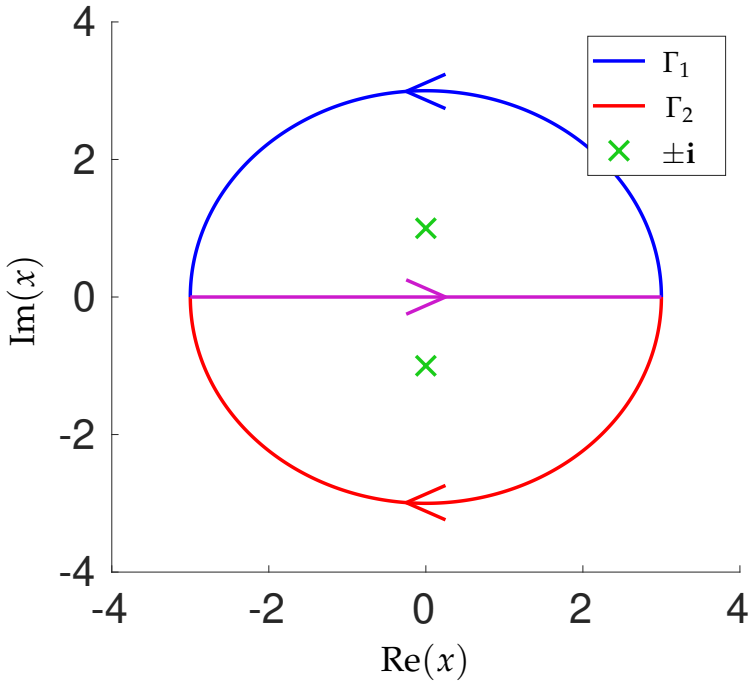

FIGURE A.1: Path integrals Γ_1, Γ_2 (upper and lower half circles) for obtaining the integral on the real axis (purple). The arcs of Γ_1, Γ_2 are denoted by γ_1, γ_2.

This fact and the estimation lemma [54, Thm. 5.24] allow us to find an upper bound for

$$\int_{\gamma_1} \frac{e^{-ix(w\omega)}}{1+x^2} \leq \left| \frac{\pi R}{R^2-1} \right|,$$

which converges to 0, as R approaches infinity.

A similar argument holds for the case of $\omega > 0$, the integral I_2 and its lower arc γ_2. Lastly, if $\omega = 0$, we use Eq. (2.12). Combining all cases results in a new equation for the integral from Eq. (A.1).

$$\int_{\mathbb{R}} \frac{e^{-ix(w\omega)}}{1+x^2} dx = \begin{cases} \lim_{R\to\infty} I_2, \omega > 0 \\ \pi, \qquad\qquad \omega = 0 \\ \lim_{R\to\infty} I_1, \omega < 0 \end{cases}. \tag{A.2}$$

Finally, we calculate I_1 and I_2. Note that

$$\frac{e^{-ix(w\omega)}}{1+x^2} = \frac{1}{2i}\left(\frac{e^{-ix(w\omega)}}{x-i} - \frac{e^{-ix(w\omega)}}{x+i} \right). \tag{A.3}$$

Using the residue theorem [3, Thm. 4.1.1] and the fact that the second summand is entirely holomorphic, we obtain

$$\begin{aligned} I_1 &= \frac{1}{2i} \int_{\Gamma_1} \frac{e^{-ix(w\omega)}}{x-i} \\ &= \pi \cdot e^{(w\omega)}, \end{aligned}$$

for the case of $\omega < 0$.

Similarly, the case $\omega > 0$ leads to

$$\begin{aligned} I_2 &= -\frac{1}{2\mathbf{i}} \int_{\Gamma_2} \frac{\mathrm{e}^{-\mathbf{i}x(w\omega)}}{x+\mathbf{i}} \\ &= -\pi \cdot (-1)\mathrm{e}^{-(w\omega)} = \pi \cdot \mathrm{e}^{-(w\omega)}, \end{aligned}$$

where the additional factor (-1) is caused by the opposite winding direction of Γ_2.

Combining all cases yields the desired formula. □

A.2 Constructing Spectra with Exactly $2m + 2n - 2$ Intersections

In addition to the proof of Thm. 3.22 in the main body, we can explicitly describe a construction that yields two spectra of m and n model functions intersecting at least $2m + 2n - 2$ times. We prove the existence of intersections by showing that the difference of both sums is first negative, then positive, and so on in at least $2m + 2n - 1$ points. Using the intermediate value theorem yields the desired number of intersections. In addition to the requirements 1. – 3. of Thm. 3.22, we also require that $f_{0,1}$ is even and a specific asymptotic property.

4. $f_{0,1}(x)$ is even, i.e., $f(x) = f(-x)$ for all $x \in \mathbb{R}$.
5. For all $c, w \in \mathbb{R}, w > 1$ there is some $\theta_w \in \left[0, \frac{1}{w}\right)$ such that

$$\frac{f_{c,1}(x)}{f_{0,w}(x)} \overset{x\to\infty}{\to} \theta_w. \tag{A.4}$$

Note that the conditions 1. – 5. imply several useful facts.

Lemma A.2 (Auxiliary lemma)**.** *Let $\mathcal{F}$ be defined as in Thm. 3.22. Then, the following statements hold.*

A: $f_{c,w}(x) \geq 0$ *for all* $x \in \mathbb{R}$.

B: $f_{c,w}(x) \geq \frac{1}{2}$ *for all* $x \in [c - w, c + w]$.

C: $f_{c,w}(x) \overset{x\to\infty}{\to} 0$.

D: $f_{c,w}(x) \overset{w\searrow 0}{\to} 0$ *for all* $x \neq c$.

E: For $0 < v \leq w$, *we have* $f_{c,v}(x) \leq f_{c,w}(x)$ *for all* $x \in \mathbb{R}$.

F: For every $m \in \mathbb{N}$ *there exists* $w' > 0$ *such that* $\frac{2}{5} \leq f_{0,w'}(x) \leq \frac{1}{2}$ *for all* $x \in [w', w' + m]$.

Proof. Most of the statements directly follow from the conditions 1. – 5.

A: Already proven in the main body, see Thm. 3.22.

B: This statement follows directly from conditions 1, 2, and 3.

C: Already proven in the main body.

D: Already proven in the main body.

E: We have $f_{c,v}(x) = f_{0,1}\left(\frac{x-c}{v}\right) \leq f_{0,1}\left(\frac{x-c}{w}\right) = f_{c,w}(x)$ using monotonicity and the fact that $\left|\frac{x-c}{v}\right| \geq \left|\frac{x-c}{w}\right|$.

F: By the intermediate value theorem, $f_{0,1}(w) = \frac{1}{2}$, and statement *C*, we know that there must be some $y > 1$ such that $f_{0,1}(y) = \frac{2}{5}$. We may define $w' = \frac{m}{y-1} > 0$ and obtain

$$\begin{aligned} f_{0,w'}(w' + m) &= f_{0,1}\left(\frac{w' + m}{w'}\right) = f_{0,1}\left(1 + \frac{m}{w'}\right) \\ &= f_{0,1}(y) = \frac{2}{5}. \end{aligned}$$

Using the monotonicity, all values in between must also lie in $\left[\frac{2}{5}, \frac{1}{2}\right]$.

□

Construcitve proof. Without loss of generality, let $m \geq n$. To provide an overview of the construction, we define the intersections in two main areas of interest. The first area lies sufficiently far to the left of 0. The second area lies sufficiently far to the right of 0, such that the functions centered in the first area do not significantly interfere with those centered in the second area. In the first area, the two sums, which we call f_m and f_n, intersect a total number of $4n - 4$ times, as we define $n - 1$ summands of both sums in this area. The remaining summand of f_n is centered in the second area, with $m - n$ functions from f_m centered there, creating an additional number of $2(m - n)$ intersections. The last peak of f_m is centered at 0 and is sufficiently broad. It serves to create the $(4n - 4)$ intersections in the first area, as well as two additional intersections, due to the asmymptotic behaviour of broad peaks. Figs. A.2 – A.6 presented in this proof give an overview of the construction in the case of $m = 10, n = 6$ Lorentzians.

Parameter Definitions:

For $i = 1, \ldots, m$ and $j = 1, \ldots, n$, we define $m + n$ functions f_{c_i,v_i}, f_{d_j,w_j} and prove that the sums

$$f_m(x) = \sum_{i=1}^{m} \alpha_i f_{c_i,v_i}(x) \qquad \text{and} \qquad f_n(x) = \sum_{j=1}^{n} \beta_j f_{d_j,w_j}(x)$$

are equal in at least $2m + 2n - 2$ points.

There are two peaks with large half-widths, one in each sum. Their half-widths must be sufficiently large to fulfill several criteria. Therefore, let w' be defined as in statement *F*. Note that all choices $w > w'$ also fulfill $\frac{2}{5} \leq f_{0,w} \leq \frac{1}{2}$ for $x \in [w, w + m]$. We define

$$c_1 = 0, \qquad v_1 = \max\{w', 3mn + 1\}, \qquad \alpha_1 = 1.$$

For the wide peak of f_n, we find a point $x^* > 0$ such that $f_{0,v_1}(x^*) < \frac{1}{10}$. In other words, the influence of f_{0,v_1} should be small for $x > x^*$. We then define

$$d_1 = x^* + \frac{1}{2}(m - n + 1), \qquad w_1 = \frac{1}{2}(m - n + 1), \qquad \beta_1 = \frac{1}{2}.$$

The function f_{d_1,w_1} is centered at the middle point of $[x^*, x^* + (m - n + 1)]$, and the inequalities $\frac{1}{4} \leq \beta_1 f_{d_1,w_1}(x) \leq \frac{1}{2}$ hold for all x from this interval. Fig. A.2 illustrates both wide peaks for $m = 10$, $n = 6$ Lorentzians.

Then, we define the $(n - 1)$ peaks in the first area of interest. Let $\delta > 0$ be sufficiently small such that $f_{c,\delta}\left(c \pm \frac{1}{2}\right) < \frac{1}{30 \cdot m}$. Furthermore, let $N \in \mathbb{N}$ be sufficiently large such that

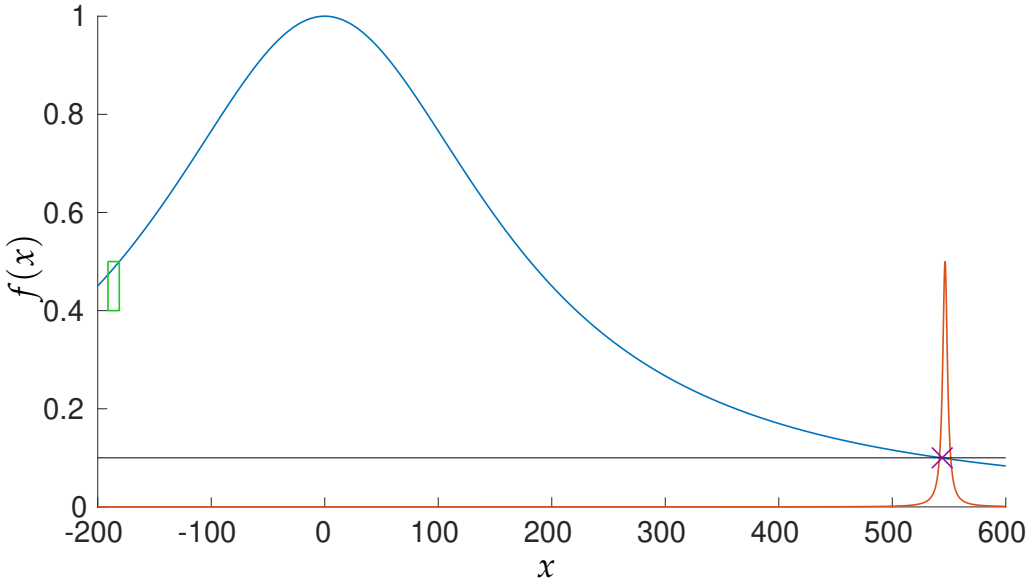

FIGURE A.2: Functions f_{0,v_1} (blue) and f_{d_1,w_1} (red). Here, $v_1 = 181$ and $d_1 = 546.5$, since $x^* = 544$ (purple cross). Note that $f_{c_1,v_1} < \frac{1}{10}$ for $x \geq x^*$ (black line) and $f_{c_1,v_1}([-v_1 - m, -v_1]) \subseteq [0.4, 0.5]$ (green box).

$f_{c,\delta/N}(c \pm \delta) < \frac{1}{10}$. Finally, for $i, j = 2, \ldots, n$ let

$$c_i = -v_1 - i + 1, \qquad v_i = \frac{\delta}{N}, \qquad \alpha_i = 3$$
$$d_j = -v_1 - j + 1, \qquad w_j = \delta, \qquad \beta_j = \frac{5}{2}.$$

The corresponding functions have most of their masses within the left area of interest, $[-v_1 - n, -v_1]$. Fig. A.3 illustrates the left area in the case of $m = 10$, $n = 6$. In this example, the area is $[-191, -181] = [-v_1 - m, -v_1]$. Additionally, we define $\delta = \frac{1}{36}$ and $N = 4$. Each of the $n - 1 = 5$ peak pairs causes four intersections.

Finally, we define the remaining $(m - n)$ peaks of f_m in the second area of interest. For $i = n + 1, \ldots, m$ let

$$c_i = x^* + i - n, \qquad v_i = \delta, \qquad \alpha_i = 1.$$

The corresponding functions and f_{d_1,w_1} have most of their masses within the right area of interest, $[x^*, x^* + (m - n + 1)]$. Fig. A.4 illustrates the right area of interest in the case of $n = 10, m = 6$. The fight area of interest is equal to $[544, 549] = [x^* + (m - n) + 1]$. Each peak causes two intersections.

We now define $2m + 2n - 1$ points at which $f_n < f_m$ and $f_m < f_n$ alternate.

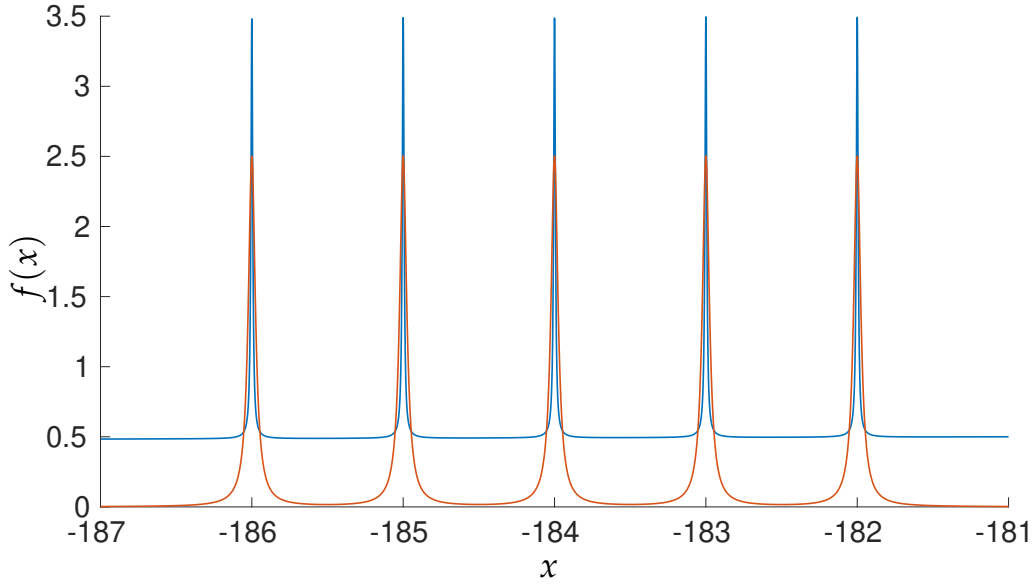

FIGURE A.3: The two sums f_m (blue) and f_n (red) displayed within the left area of interest.

First Area of Interest:

The first $4n-3$ points $x_0 < x_1 < \ldots < x_{4n-4}$ are defined as follows

$$x_k = \begin{cases} -v_1 - n + \frac{1}{2} + \frac{k}{4}, & k \equiv 0 \mod 4 \\ -v_1 - n + \frac{k-1}{4} + 1 - \delta, & k \equiv 1 \mod 4 \\ -v_1 - n + \frac{k-2}{4} + 1, & k \equiv 2 \mod 4 \\ -v_1 - n + \frac{k-3}{4} + 1 + \delta, & k \equiv 3 \mod 4 \end{cases}. \tag{A.5}$$

All of these points lie in the interval $[-v_1 - m, -v_1]$. Note that $f_{0,v_1}(x_k) \in \left[\frac{2}{5}, \frac{1}{2}\right]$ for all x_k defined in Eq. (A.5).

For $k \equiv 0 \mod 4$, we have $f_m(x_k) \geq f_{0,v_1}(x_k) \geq \frac{2}{5}$. For the summands of f_n, we have $f_{d_j,w_j}(x_k) < \frac{1}{30m}$ for $j = 2, \ldots, n$, since $|d_j - x_k| \geq \frac{1}{2}$. Using $v_1 > m > w_1$ we have

$$\begin{aligned} f_{d_1,w_1}(x_k) &\overset{2.}{\leq} f_{d_1,v_1}(0) = f_{0,v_1}(d_1) \\ &\overset{E}{\leq} f_{0,v_1}(d_1) \overset{2.}{\leq} f_{0,v_1}(x^*) < \frac{1}{10}. \end{aligned}$$

The last inequality holds by the definition of x^*. Finally, we estimate

$$\begin{aligned} f_n(x_k) &= \frac{1}{2} f_{d_1,w_1} + \sum_{j=2}^{n} \frac{5}{2} f_{d_j,w_j}(x_k) \\ &< \frac{1}{20} + \frac{5(n-1)}{2 \cdot 30m} < \frac{2}{15} < \frac{2}{5} \leq f_m(x_k). \end{aligned}$$

For $k \equiv 1 \mod 4$ and $k \equiv 3 \mod 4$, we have $x_k = d_\ell \pm \delta$ for some $\ell \in \{2, \ldots, n\}$. Consequently, we have $f_n(x_k) \geq \frac{5}{2} f_{d_\ell,\delta}(c_\ell \pm \delta) = \frac{5}{4}$. For the summands of f_m, we have $f_{0,v_1}(x_k) \leq \frac{1}{2}$ and $f_{c_j,v_j}(x_k) < \frac{1}{30m}$ for $j = 2, \ldots, m$ and $j \neq \ell$, since all of these functions have

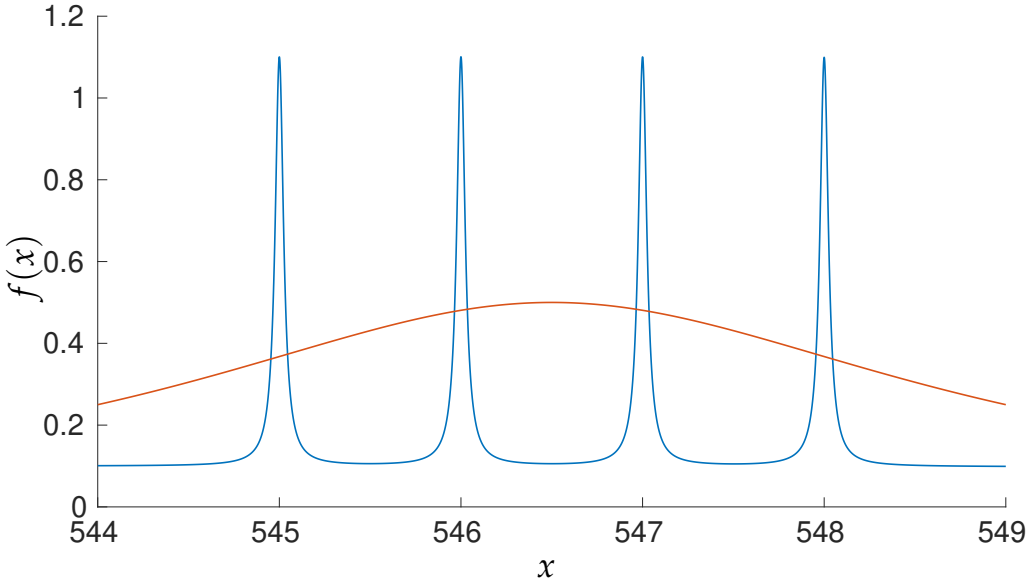

FIGURE A.4: The two sums f_m (blue) and f_n (red) displayed in the right area of interest.

half-widths smaller than or equal to δ and their centers are at least $\frac{1}{2}$ apart from x_k. Lastly,

$$f_{c_\ell,v_\ell}(d_\ell \pm \delta) = f_{d_\ell,\delta/N}(d_\ell \pm \delta) < \frac{1}{10}$$

by the definition of N.

Again, we can estimate

$$\begin{aligned} f_m(x_k) &= f_{c_1,v_1}(x_k) + 3 \cdot f_{c_\ell,v_\ell}(x_k) + \sum_{\substack{i=2 \\ i\neq\ell}}^{m} \alpha_j f_{c_i,v_i}(x_k) \\ &< \frac{1}{2} + \frac{3}{10} + \frac{3(m-2)}{30m} < \frac{9}{10} < \frac{5}{4} \leq f_n(x_k). \end{aligned}$$

Finally, consider $k \equiv 2 \mod 4$. Again, there is some index $\ell \in \{2, \ldots, n\}$ such that $x_k = c_\ell = d_\ell$. The most significant summand of f_m is $\alpha_\ell f_{c_\ell,v_\ell}(x_k) = 3$, implying $f_m(x_k) \geq 3$. On the other hand, we have $|x_k - d_j| \geq 1 > \frac{1}{2}$ for $j \neq \ell$. Therefore, we have $f_{d_1,w_1}(x_k) < \frac{1}{10}$ using similar arguments as before.

Finally, we obtain

$$\begin{aligned} f_n(x_k) &= \frac{1}{2} f_{d_1,w_1}(x_k) + \frac{5}{2} f_{d_\ell,w_\ell}(x_k) + \sum_{\substack{j=2 \\ j\neq\ell}}^{n} \frac{5}{2} f_{d_j,\delta}(x_k) \\ &< \frac{1}{20} + \frac{5}{2} + \frac{5(n-2)}{30m} < \frac{163}{60} < 3 \leq f_m(x_k). \end{aligned}$$

We conclude the first area of interest with points that satisfy the inequalities

$$f_m(x_0) > f_n(x_0), f_m(x_1) < f_n(x_1), \ldots, f_m(x_{4n-4}) > f_n(x_{4n-4}).$$

Fig. A.5 illustrates the relationship between f_n and f_m for one of the five peak pairs in the

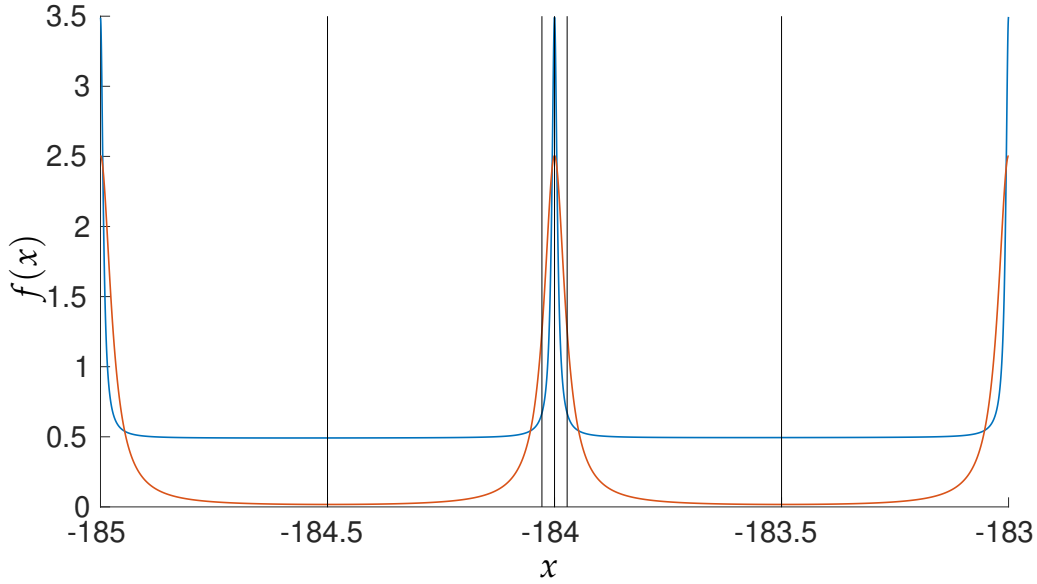

FIGURE A.5: Five points x_k (black lines), as defined by Eq. (A.5). For $x_k = -184 \pm \frac{1}{2}$ and $x_k = -184$, we have $f_n < f_m$. For $x_k = -184 \pm \delta$, we have $f_m < f_n$.

left area of our example.

Second Area of Interest:

We define $2m - 2n + 1$ points $y_0 < y_1 < \ldots < y_{2m-2n}$ in the interval $[x^*, x^* + m - n + 1]$ by

$$y_k = x^* + \frac{k+1}{2}. \tag{A.6}$$

Note that $y_k \in [d_1 - w_1, d_1 + w_1]$. We first consider the case of even k. Here, we have $|y_k - c_i| \geq \frac{1}{2}$ and we obtain

$$\begin{aligned} f_m(y_k) &= f_{c_1,v_1}(y_k) + \sum_{i=2}^{m} \alpha_i f_{c_i,v_i}(y_k) \\ &< \frac{1}{10} + \frac{3(m-1)}{30m} < \frac{1}{5} < \frac{1}{4} \leq f_{d_1,w_1}(y_k) \leq f_n(y_k). \end{aligned}$$

In the case of odd k, there is some $\ell \in \{n+1, \ldots, m\}$ such that $y_k = c_\ell$. Therefore, we have

$$\begin{aligned} f_n(y_k) &= f_{d_1,w_1}(y_k) + \sum_{j=2}^{n} \beta_j f_{d_j,w_j}(y_k) \\ &< \frac{1}{2} + \frac{5(n-1)}{2 \cdot 30m} < \frac{7}{12} < 1 = \alpha_\ell f_{c_\ell,v_\ell}(y_k) \leq f_m(y_k). \end{aligned}$$

Again, we conclude that

$$f_m(y_0) < f_n(y_0), f_m(y_1) > f_n(y_1), \ldots, f_m(y_{2m-2n}) < f_n(y_{2m-2n}).$$

Fig. A.6 illustrates the relationship between f_n and f_m for one of the four peaks in the right

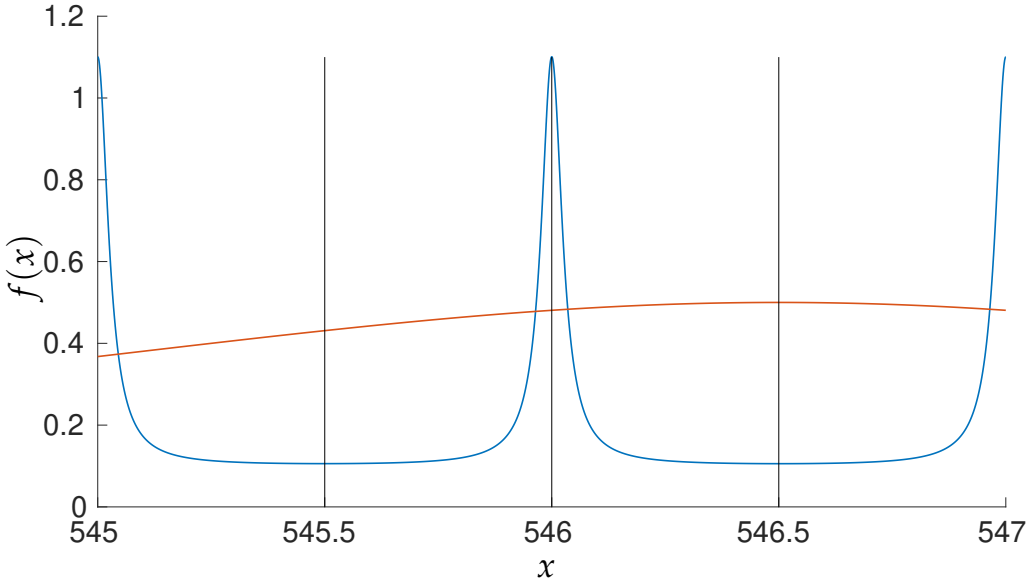

FIGURE A.6: Three points y_k (black lines), as defined by Eq. (A.6). For $y_k = 546 \pm \frac{1}{2}$, we have $f_m < f_n$. For $y_k = 546$, we have $f_m > f_n$.

area of our example.

Asymptotic Differences:

For the final point, we show the existence of some $x_a > y_{2m-2n}$, where $f_n(x_a) < f_m(x_a)$. We prove this by observing the different convergence rates of f_n and f_m.

$$\begin{aligned}\lim_{x\to\infty} \frac{f_n}{f_m}(x) &\leq \lim_{x\to\infty} \frac{f_n}{f_{c_1,v_1}}(x) = \lim_{x\to\infty} \sum_{j=1}^{n} \frac{\beta_j f_{d_j,w_j}(x)}{f_{0,v_1}(x)} \\ &= \lim_{x\to\infty} \sum_{j=1}^{n} \frac{\beta_j f_{d_j,1}(x \cdot w_j)}{f_{0,v_1/w_j}(x \cdot w_j)} \overset{5.}{<} \sum_{j=1}^{n} \beta_j \cdot \theta_{v_1/w_j} \\ &< \sum_{j=1}^{n} \frac{\beta_j w_j}{v_1} < \frac{3mn}{v_1} < 1,\end{aligned}$$

using $w_i < m$ for all $i = 1, \ldots, n$. Therefore, such a point x_a exists, and can be chosen sufficiently far from 0. We additionally require $x_a > x^* + m - n + \frac{1}{2}$.

Applying the Intermediate Value Theorem:

Now, we have a set of $(2m + 2n - 1)$ points

$$x_0 < x_1 < \ldots < x_{4n-4} < 0 < y_0 < y_1 < \ldots < y_{2m-2n} < x_a,$$

where the relations

$$f_n(x_0) < f_m(x_0), f_m(x_1) < f_n(x_1), \ldots, f_n(x_a) < f_m(x_a)$$

alternate. Both sums are continuous functions and by using the intermediate value theorem, we are able to infer the existence of $2m + 2n - 2$ points where f_n and f_m intersect. □

A.3 Possible Proof of Conj. 1

We want to estimate the maximum number of intersections between n and m Gaussians G_{c_i,v_i}, G_{d_j,w_j} for $i = 1,\dots,m, j = 1,\dots,n$. Therefore, we find solutions to

$$\sum_{i=1}^{m} \alpha_i G_{c_i,v_i}(x) = \sum_{j=1}^{n} \beta_j G_{d_j,w_j}(x)$$

If we define $k = m + n$ and $(-\alpha_{m+j}, c_{m+j}, v_{m+j}) = (\beta_j, d_j, w_j)$, then, we can rewrite the equation as

$$0 = \sum_{i=1}^{k} \alpha_i G_{c_i,v_i}(x) \tag{A.7}$$

where w.l.o.g. $v_1 \leq \ldots \leq v_k$.

If the maximum number of maxima of the right sum is estimated to be $\leq k$, then a total of $2k-1$ extrema can be found[24]. Since the continuous sum converges to 0 if $x \to \pm\infty$, we know that all roots must lie between the extrema. By Lem. 3.23, we know that the number of zeros is finite. Therefore, roots always imply a sign change, and the total number of roots can be estimated to be $\leq 2k-2 = 2m+2n-2$.

To prove the conjecture, we need to consider the following lemma, which is described in [23, Thm. 2] in the more general case of $\alpha_i \neq 0$.

Lemma A.3 (Carreira-Perpiñán, Williams)**.**

A weighted sum of k Gaussians, as defined in Eq. (A.7)*, has at most k maxima.*

Here, we recreate the short *proof*[25] from [23] in the more general setting of $\alpha_i \neq 0$. There, another auxiliary lemma is needed to prove Lem. A.3. This auximilary lemma was formulated by several different groups [11, 65, 109, 134], even though, it seems it was first formulated by the group of Andrew Witkin[26] in the context of image handling and computer animation, and claims that any function with k maxima, when convoluted with any Gaussian, has $\leq k$ Maxima after convolution. Unfortunately, we cannot verify this claim ourselves, since it is worded imprecisely. Here, we quote the formulation from [23].

Lemma A.4 (Witkin)**.**

"The Gaussian [convolution] kernel never creates new maxima [in one dimension]."

Other formulations include "[Gaussian] diffusion only destroys structure but cannot generate it", see [65], "the Gaussian is unique in guaranteeing that extrema may be added but not deleted in moving from coarse to fine scale"[27], see [11].

The problem with these definitions is that they are imprecise regarding the type of object that is convoluted by the Gaussian kernels. We further assume that any sum of Gaussians and delta distributions falls into this category.

Finally, we can *prove* Lem. A.3, similar to [23].

This *proof* uses an induction on the number of summands k.

[24] We assume that the number of minima is smaller. If not, multiply by -1.

[25] Due to the lack of rigorous definitions, we denote imprecise formulations in *italics*.

[26] Witkin became famous for his work on the movie *Toy Story 3* during his time at the studio *Pixar*.

[27] Compared to our setting, they consider movement in the opposite direction.

$\underline{k = 1}$:

Trivially, a single Gaussian has a unique maximum at $x = c_1$.

$\underline{1, 2, \ldots, k \to k+1}$:

We consider the sum

$$\sum_{i=1}^{k+1} \alpha_i G_{c_i, v_i}(x),$$

where $v = v_1 \leq \ldots \leq v_{k+1} = v$ and define $J = \{i = 1, \ldots, k+1 \mid v_i = v\}$ with $j = |J|$. Then, *Gaussian deblurring* is applied, which we can only imagine implies a convolution with the multiplicative inverse of the Gaussian $G_{c_1, v}$. Since v_1 is the smallest half-width, this creates different Gaussians for $i \notin J$, and delta distributions for all $i \in J$ at different points c_i. Crucially, the sum of Gaussians consists of only $k + 1 - j > k + 1$ summands and thus has $k + 1 - j$ maxima at most. Furthermore, it is assumed that delta distributions do not interfere with the structure of the Gaussian sum and only add j maxima. Therefore, the *deblurred* sum has at most $k + 1$ maxima. Next, the change is reverted, and the *deblurred* sum with $\leq k + 1$ maxima is convoluted with $G_{c_1, v}$ again. Due to Lem. A.4, the retrieved original sum of $k + 1$ Gaussians has no more than $k + 1$ maxima.

To recapitulate: This series of arguments is imprecise, at the very least. If all of the remaining uncertainties can be rigorously stated and proven, then Conjecture 1 can be considered a fact.

Appendix B

Data Sets and Modeling Results

Here, we provide an overview of the constructed and experimental data sets. Additionally, we explain the measurement processes for the experimental data sets. These descriptions can also be found in [53, 86]. Lastly, we display the results of modeling each of the six data sets.

B.1 Constructed Data

These data sets were created as simpler counterparts to the more complex experimental data sets. Set D_a is used to demonstrate the behavior of the optimization variants and serves as a benchmark. The set D_b was created to observe and prove the functionality of the parameter detection for neural networks and is used to test the trained CNNs. Lastly, D_c was created to test multiplet detection and to observe differences in multiplet optimization.

Data set D_a:

This data set contains three Lorentzian peaks that cross twice. The center and half-width parameters are defined by continuous and smooth functions over time. A simple reaction kinetic $A + B \rightarrow C$ was used to determine the peak heights. Gaussian white noise with a signal-to-noise ratio (SNR) of approximately $100 : 1$ was added. Compared to the experimental data sets, the resolution in the frequency domain is low, since $D_a \in \mathbb{R}^{61\times500}$. Note that one peak vanishes at the beginning and another peak disappears at the end of the data set. Fig. B.1 provides an overview from two angles.

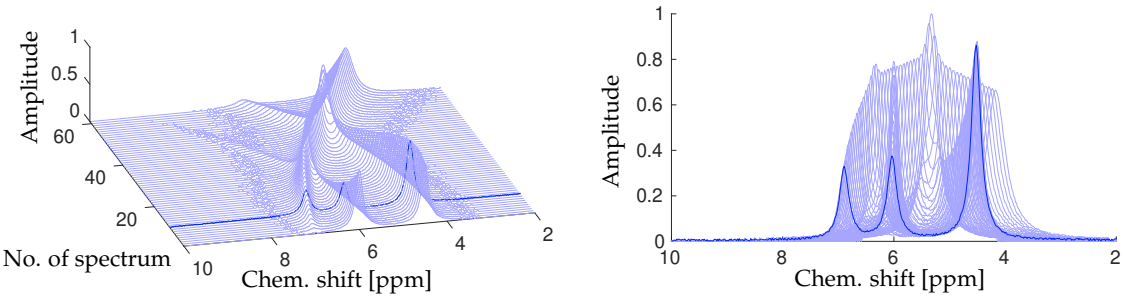

FIGURE B.1: Data set D_a displayed top-down and frontally.

Data set D_b:

This data set also contains three pseudo-Voigt peaks that cross three times. It was created specifically as training data for machine learning. Compared to D_a, the changes in parameters are more drastic. While the convex combination parameter of the peaks in D_a is constant ($\lambda_i = 0$), every parameter varies in D_b. However, this comes at the expense of chemical

meaning, as the peak integral values do not correspond to reaction kinetics. Both D_a and $D_b \in \mathbb{R}^{61\times500}$ have the same number of data points, but the noise level of D_b is comparatively high, with an SNR of roughly 40 : 1. Fig. B.2 displays D_b from two angles.

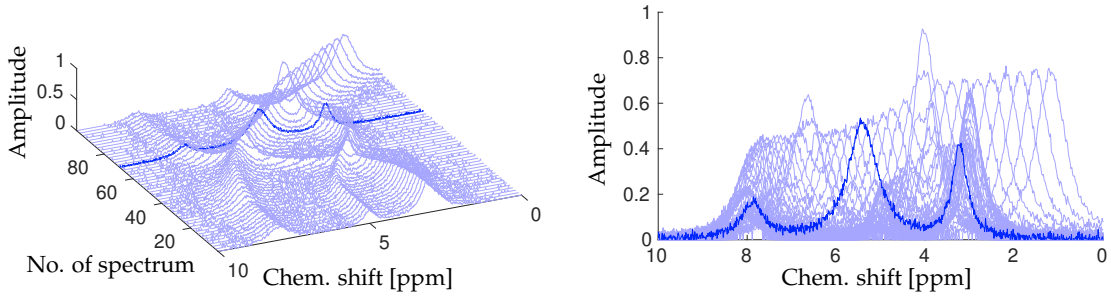

FIGURE B.2: Data set D_b displayed top-down and frontally.

Data set D_c:

The final constructed data set is composed of several pseudo-Voigt multiplets. It contains a septuplet, a sixtuplet, a quintuplet, three doublets, and one singlet, for a total of 25 peaks. When accounting for individual peak crossings, this data set contains a total of 25 peak crossings, while the multiplets switch positions five times. Data set D_c was designed to test multiplet detection and optimization behavior. Once again, all optimizable parameters vary and there is no underlying reaction kinetics. The SNR of D_c is higher than before at around 200 : 1, and it contains almost four times as many data points as D_a and D_b, with $D_c \in \mathbb{R}^{101\times1001}$. Fig. B.3 displays D_c from two angles.

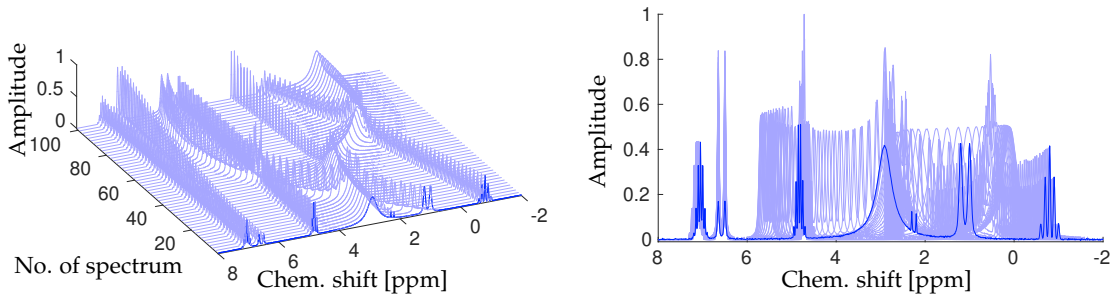

FIGURE B.3: Data set D_c displayed top-down and frontally.

B.2 Experimental Data

The remaining three data sets were measured during three separate chemical experiments using high-field NMR spectrometers. This implies that the peaks are comparatively sharp. Additionally, imperfect preprocessing of the data leads to deviations from the assumed peak shape. Furthermore, all three data sets are subject to only moderate amounts of noise.

Data set D_1:

The first of the experimental data sets was measured during the esterification of methanol (CH_3OH) and formic acid (HCOOH) to methyl formate ($HCOOCH_3$) and water. For more information on the experiment, we refer to [16]. During the reaction, 2 mass-% of sulfuric

acid was used as a catalyst. To acquire the spectra, a 400 MHz NMR spectrometer was used with single scans. One spectrum was measured every 15 s with an acquisition time of 1 s. The temperature during the reaction was 333 K, or about 60° C. A total of 199 spectra with $\approx$ 36,000 data points were recorded. The relevant portion contains fewer spectra and data points, thus $D_1 \in \mathbb{R}^{195\times 24577}$.

This data set contains five non-intersecting peaks. Four of the peaks can be considered *static peaks*, i.e., peaks whose position does not significantly change throughout the data set. Conversely, the last peak can be considered a *moving peak*. Additionally, at the beginning of the experiment, the moving peak exhibits distorted peak shapes. This data set contains several artifacts in each spectrum, two for each of the static peaks. The noise levels are small at $\approx 1.5 \cdot 10^{-4}$. Fig. B.4 gives an overview from two angles. Note that the artifacts are thinner

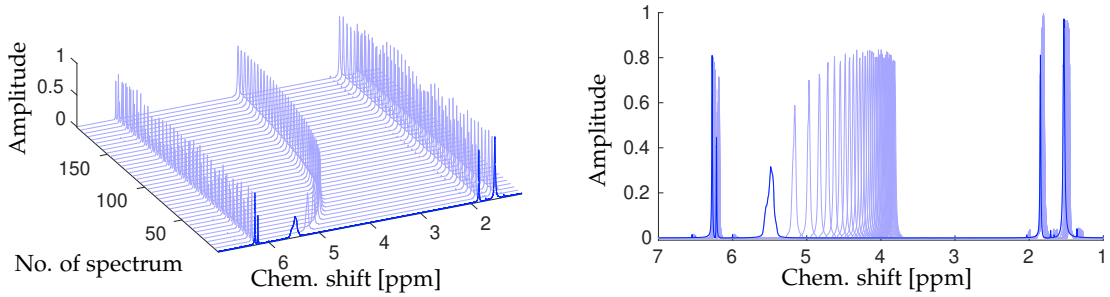

FIGURE B.4: Data set D_1 displayed top-down and frontally.

and smaller than the peaks and can barely be observed in the figure.

Data set D_2:

The second experimental data set was collected during an isobaric batch distillation of methanol (CH_3OH) and diethylamine ($C_4H_{11}N$). For further details on the experiment, we refer to [39]. The composition of the liquid phase was analyzed by a medium-field NMR spectrometer with a field strength of 1 T, corresponding to a proton Larmor frequency of 42.5 Mhz. NMR spectra were recorded at one-minute intervals with a single scan and an acquisition time of 3.2 s. In total, 151 spectra with $\approx$16,000 data points each were recorded. The relevant portion of the measurement covers about 6,001 data points, thus $D_2 \in \mathbb{R}^{151\times 6001}$.

This data set contains nine peaks in total, one of is a moving peak, while the remaining eight are static. Three of the static peaks form a triplet, and another four form a quartuplet. Initially, the moving peak is located in the same region as the quartuplet. During the course of the experiment, it crosses parts of the multiplet and another static peak. Since the multiplets deviate from the characteristic height pattern, we can choose to avoid modeling them as such. The two modeling options can be compared in Figs. B.11 and B.12. Fig. B.5 displays the data set from two angles. The moving peak is second from the left in the highlighted spectrum. While there are no artifacts in this data set, all peaks are slightly incorrectly *phased*, see also Sec. 6.5. Compared to D_1 and D_3, the noise levels of D_2 are relatively high at $\approx$ 0.5%.

Data set D_3:

The final and most complex data set was acquired during the reaction of acetic anhydride and 2-propanol in the presence of H_2SO_4 to isopropylacetate and acetic acid. As an inert component, benzene was added to the reacting mixture. The reaction was accelerated by sulphuric acid. The reactants were thoroughly mixed, loaded into a 5 mm NMR tube, and placed in a high-field NMR spectrometer equipped with a 9.4 T vertical superconducting magnet corresponding to a proton Larmor frequency of 400.25 MHz. The temperature of the

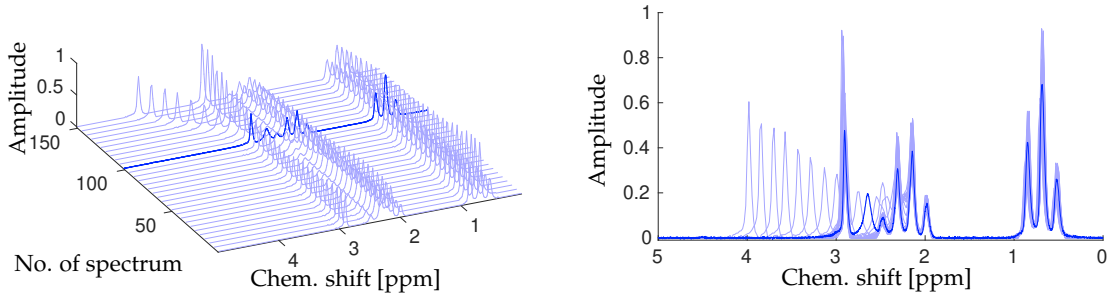

FIGURE B.5: Data set D_2 displayed top-down and frontally.

reacting mixture was approximately 25 °C. NMR spectra were recorded with a single scan, an acquisition time of 6 s and a pulse angle of 30°. A total of 680 spectra were recorded, each containing ≈65,000 data points. The relevant portion of the measurement covers 48,055 data points, thus $D_3 \in \mathbb{R}^{680\times 48055}$.

This data set consists of two septuplets, three doublets and three single peaks, adding up to 23 peaks in total. All peaks except for one single peak are static.[28]. The moving peak crosses the peak positions of one septuplet and one single peak. Additionally, one septuplet vanishes in the beginning and one disappears near the end of the experiment. Fig. B.6 shows the data set from two angles. Although no artifacts are visible, each static peak causes two artifacts to appear on either side. This data set is the most difficult to handle, not only because of the complex peak structures, but also because of the comparatively large ratio of largest to smallest peak, which sometimes exceeds 1000 : 1. Due to the high field strength of the magnetic field, the noise levels are relatively low, at $\approx 5 \cdot 10^{-5}$.

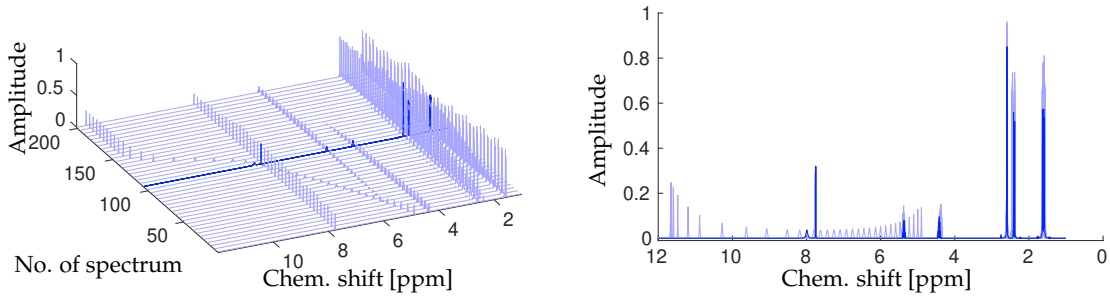

FIGURE B.6: Data set D_3 displayed top-down and frontally.

B.3 Modeling Results

Lastly, we present the results for all data sets. In the following Figs. B.7 – B.13 only the peak centers and heights are displayed for the sake of clarity. Each red star represents one parameter tuple of the spectra $t \in T_{sel}$, and the red lines represent interpolated or computed parameters. The modeling quality and runtime is displayed and discussed in the main body, see Sec. 6.4.

[28] All of the doublets merge later on, i.e., their distance decreases until only one peak remains.

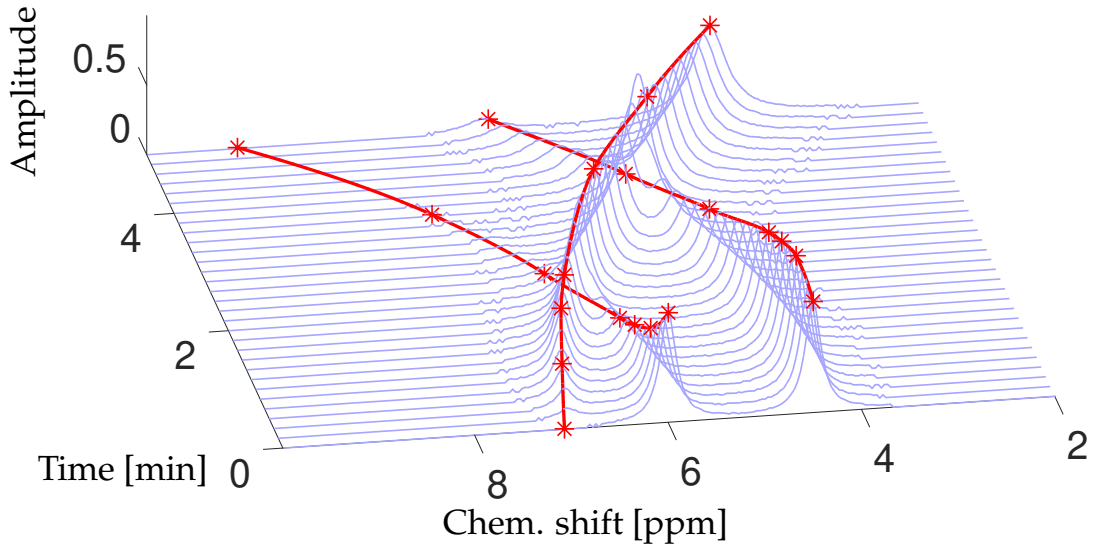

FIGURE B.7: Modeling results for D_a.

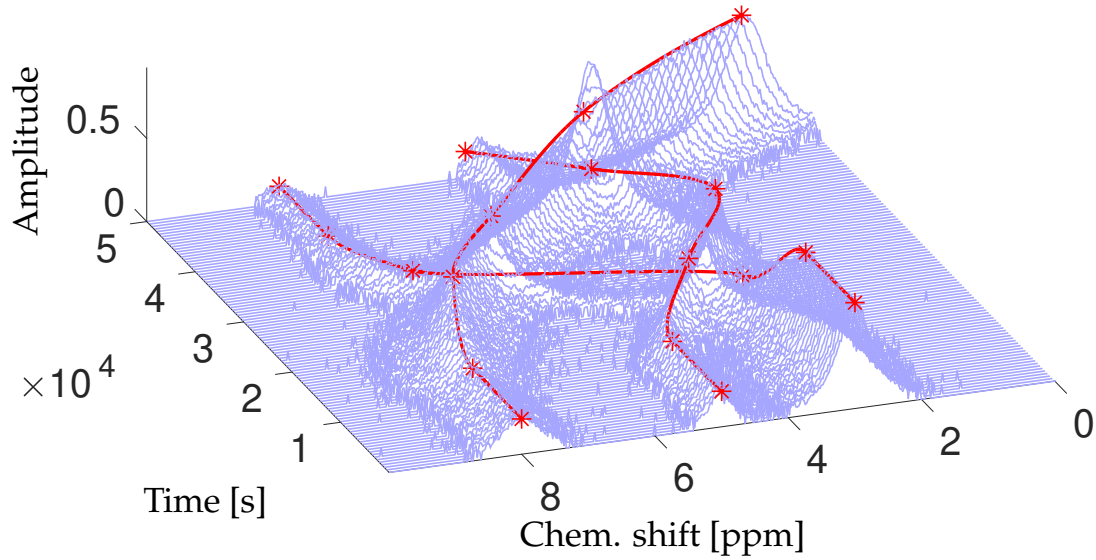

FIGURE B.8: Modeling results for D_b.

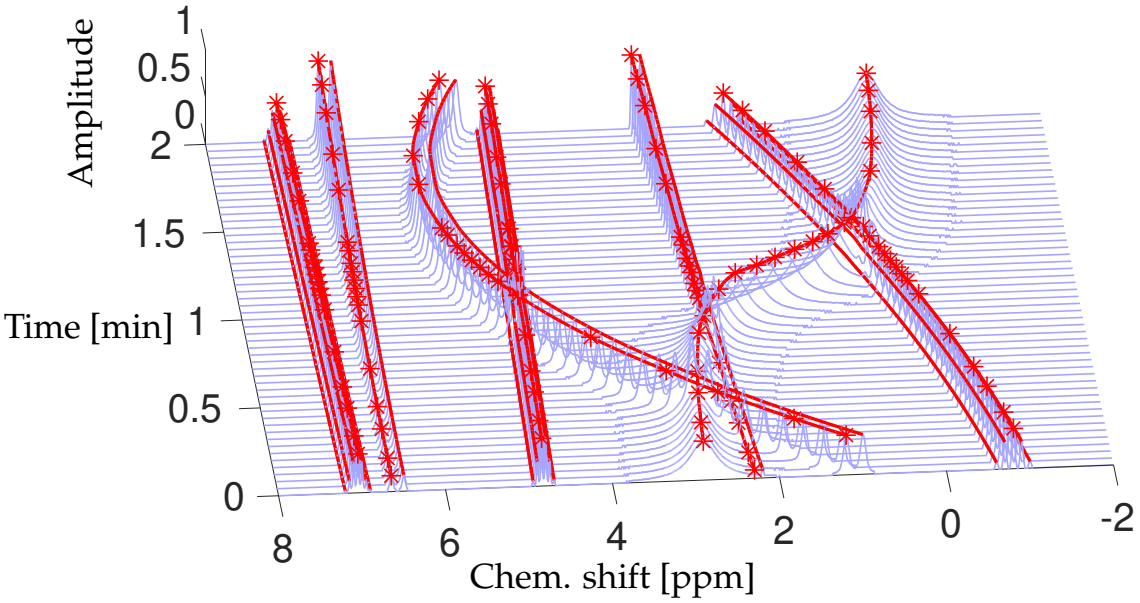

FIGURE B.9: Modeling results for D_c.

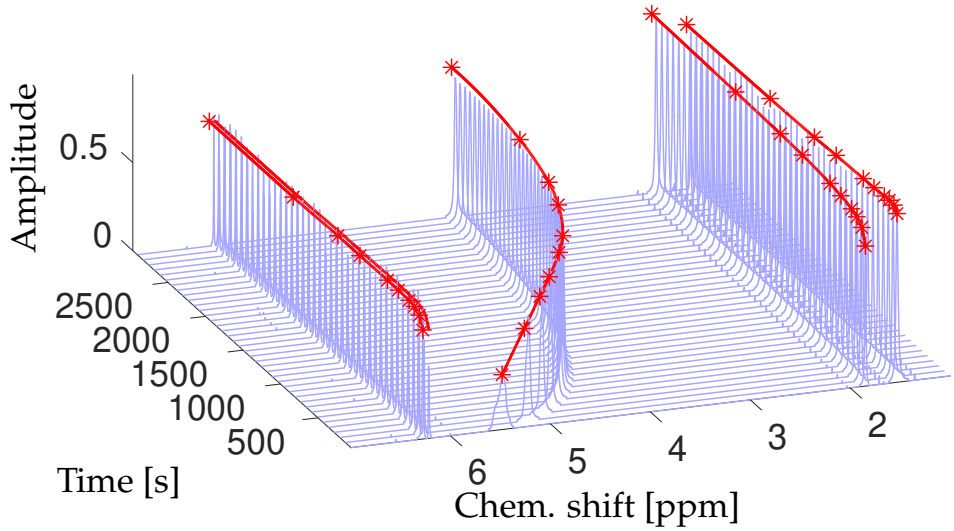

FIGURE B.10: Modeling results for D_1.

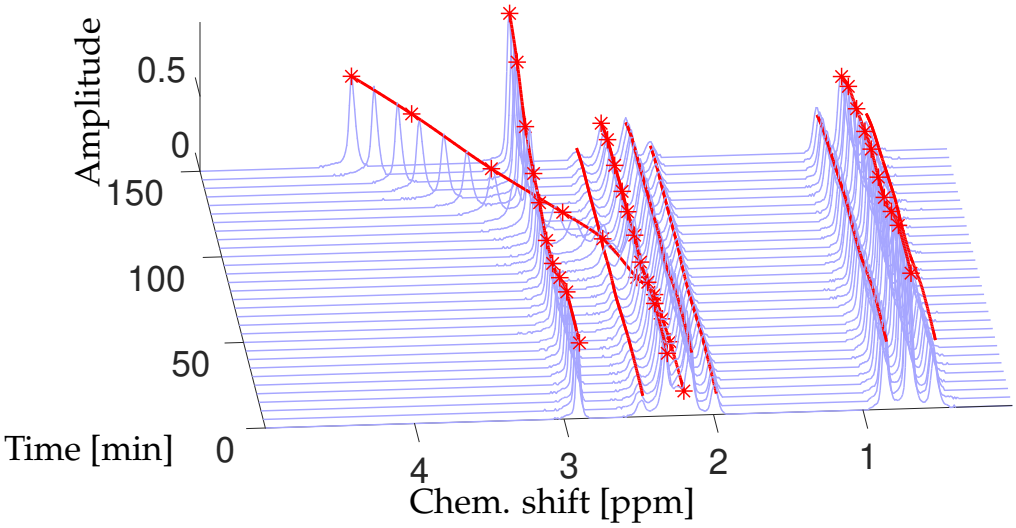

FIGURE B.11: Modeling results for D_2 when using multiplets.

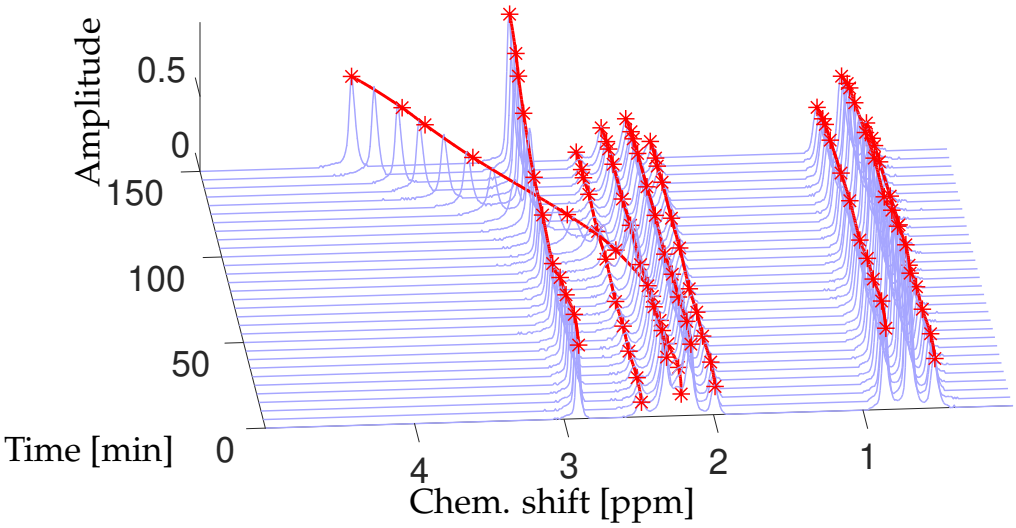

FIGURE B.12: Modeling results for D_2 without using multiplets.

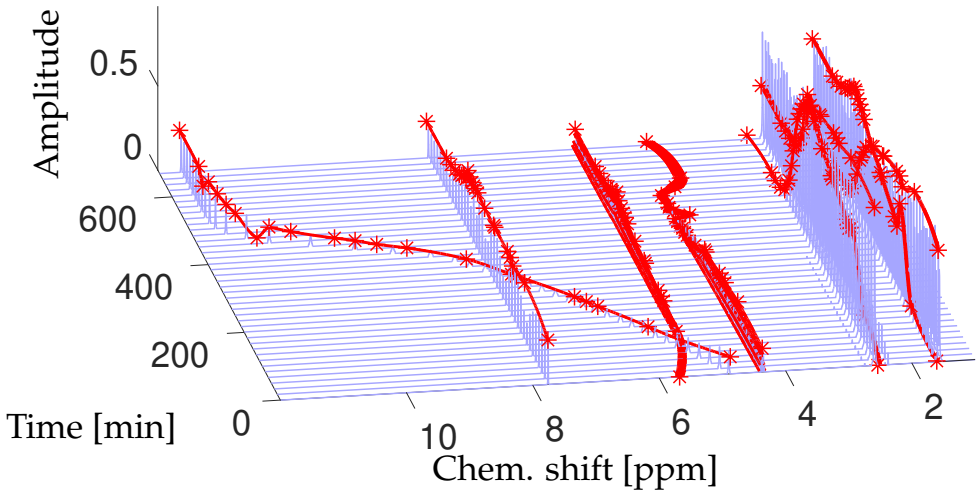

FIGURE B.13: Modeling results for D_3.

Bibliography

[1] M. Abadi et al. "TensorFlow: a system for Large-Scale machine learning". In: *12th USENIX symposium on operating systems design and implementation (OSDI 16)*. 2016, pp. 265–283.

[2] H. Abdollahi and R. Tauler. "Uniqueness and rotation ambiguities in multivariate curve resolution methods". In: *Chemom. Intell. Lab. Syst.* 108.2 (2011), pp. 100–111.

[3] M. J. Ablowitz and A. S. Fokas. *Complex variables, Cambridge texts in applied mathematics*. Cambridge University Press, Cambridge, 1997.

[4] S. M. Abrarov and B. M. Quine. "A rational approximation for efficient computation of the Voigt function in quantitative spectroscopy". In: *J. Math. Res.* 7.2 (2015).

[5] A. A. S. Al Omar. "Line width at half maximum of the Voigt profile in terms of Gaussian and Lorentzian widths: Normalization, asymptotic expansion, and chebyshev approximation". In: *Optik* 203 (2020), p. 163919.

[6] I. Alshamleh et al. "Real-time NMR spectroscopy for studying metabolism". In: *Angew. Chem.* 132.6 (2020), pp. 2324–2328.

[7] F. Alsmeyer, H.-J. Koß, and W. Marquardt. "Indirect spectral hard modeling for the analysis of reactive and interacting mixtures". In: *Appl. Spectrosc.* 58.8 (2004), pp. 975–985.

[8] C. Améndola, A. R. Engström, and C. Haase. "Maximum number of modes of Gaussian mixtures". In: *Inf. Inference* 9.3 (2020), pp. 587–600.

[9] C. Aroulanda et al. "Structural ambiguities revisited in two bridged ring systems exhibiting enantiotopic elements, using natural abundance deuterium NMR in chiral liquid crystals". In: *J. Phys. Chem. A* 107.50 (2003), pp. 10911–10918.

[10] M. Artin. *Algebra*. Birkhäuser, Basel, 1998.

[11] J. Babaud et al. "Uniqueness of the Gaussian kernel for scale-space filtering". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 1 (1986), pp. 26–33.

[12] V. Bakshi and R. J. Kearney. "New tables of the Voigt function". In: *J. Quant. Spectrosc. Radiat. Transf.* 42.2 (1989), pp. 111–115.

[13] R. G. Bartle and D. R. Sherbert. *Introduction to real analysis*. John Wiley & Sons, Inc., Hoboken, 2000.

[14] A. Bhakar, M. Taxak, and S. K. Rai. "Significance of diffraction peak shapes in determining crystallite size distribution: a peak shape analysis procedure for pseudo-Voigt profiles and its application". In: *J. Appl. Crystallogr.* 56.5 (2023), pp. 1466–1479.

[15] M. Bownik and D. Speegle. "Linear independence of time–frequency translates of functions with faster than exponential decay". In: *Bull. Lond. Math. Soc* 45.3 (2013), pp. 554–566.

[16] A. Brächer et al. "Application of a new micro-reactor 1H NMR probe head for quantitative analysis of fast esterification reactions". In: *Chem. Eng. J.* 306 (2016), pp. 413–421.

[17] E. Breitmaier. *Structure elucidation by NMR in organic chemistry: a practical guide*. John Wiley & Sons, Chichester, 2002.

[18] T. Brown et al. "Language models are few-shot learners". In: *Adv. Neural Inf. Process. Syst.* 33 (2020), pp. 1877–1901.

[19] S. D. Bruce et al. "An analytical derivation of a popular approximation of the Voigt function for quantification of NMR spectra". In: *J. Magn. Reson.* 142.1 (2000), pp. 57–63.

[20] S. Bruderer, F. Paruzzo, and C. Bolliger. "Deep learning-based phase and baseline correction of 1D 1H NMR Spectra". In: *Public Bruker White Paper* (2021).

[21] W. A. Bubb. "NMR spectroscopy in the study of carbohydrates: Characterizing the structural complexity". In: *Concepts Magn. Reson. A* 19.1 (2003), pp. 1–19.

[22] E. A. Carrara, F. Pagliari, and C. Nicolini. "Neural networks for the peak-picking of nuclear magnetic resonance spectra". In: *Neural Netw.* 6.7 (1993), pp. 1023–1032.

[23] M. A. Carreira-Perpiñán and C. K. I. Williams. "On the number of modes of a Gaussian mixture". In: *International Conference on Scale-Space Theories in Computer Vision*. Springer. 2003, pp. 625–640.

[24] D. Chen et al. "Review and Prospect: Deep Learning in Nuclear Magnetic Resonance Spectroscopy". In: *Chem. Eur. J.* 26.46 (2020), pp. 10391–10401.

[25] O. Christensen and K. L. Christensen. "Linear independence and series expansions in function spaces". In: *Am. Math. Mon.* 113.7 (2006), pp. 611–627.

[26] R. Dass, P. Kasprzak, and K. Kazimierczuk. "Quick, sensitive serial NMR experiments with Radon transform". In: *J. Magn. Reson.* 282 (2017), pp. 114–118.

[27] I. Daubechies. *Ten lectures on wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[28] B. Deng and C. E. Heil. "Density of Gabor Schauder bases". In: *Wavelet Applications in Signal and Image Processing VIII*. Vol. 4119. SPIE. 2000, pp. 153–164.

[29] J. V. Deshpande, U. Naik-Nimbalkar, and I. Dewan. *Nonparametric statistics: theory and methods*. World Scientific, Singapore, 2017.

[30] R. L. Dobrushin. "Prescribing a system of random variables by conditional distributions". In: *Theory Probab. Its Appl.* 15.3 (1970), pp. 458–486.

[31] B. Domżał et al. "Magnetstein: An Open-Source Tool for Quantitative NMR Mixture Analysis Robust to Low Resolution, Distorted Lineshapes, and Peak Shifts". In: *Anal. Chem.* 96.1 (2023), pp. 188–196.

[32] B. Domżał et al. "NMR reaction monitoring robust to spectral distortions". In: *ChemRxiv preprint* (2025). DOI: 10.26434/chemrxiv-2025-pjzl3. URL: https://chemrxiv.org/engage/chemrxiv/article-details/67a4937d81d2151a025ceef7.

[33] R. M. Dudley. *Real Analysis and Probability*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, Cambridge, 2002.

[34] A.-H. M. Emwas et al. "NMR-based metabolomics in human disease diagnosis: applications, limitations, and recommendations". In: *Metabolomics* 9 (2013), pp. 1048–1072.

[35] M. C. Ezeanaka, J. Nsor-Atindana, and M. Zhang. "Online low-field nuclear magnetic resonance (LF-NMR) and magnetic resonance imaging (MRI) for food quality optimization in food processing". In: *Food Bioproc. Tech.* 12 (2019), pp. 1435–1451.

[36] G. Fischetti et al. "A deep learning framework for multiplet splitting classification in 1H NMR". In: *J. Magn. Reson.* 373 (2025), p. 107851.

[37] G. Fischetti et al. "Automatic classification of signal regions in 1H Nuclear Magnetic Resonance spectra". In: *Front. Artif. Intell.* 5 (2023), p. 1116416.

[38] F. Fricke et al. "Artificial Intelligence for Mass Spectrometry and Nuclear Magnetic Resonance Spectroscopy Using a Novel Data Augmentation Method". In: *IEEE Trans. Emerg. Top. Comput.* 10.1 (2021), pp. 87–98.

[39] A. Friebel et al. "Online process monitoring of a batch distillation by medium field NMR spectroscopy". In: *Chem. Eng. Sci.* 219 (2020), p. 115561.

[40] A. Friebel et al. "Reaction monitoring by benchtop NMR spectroscopy using a novel stationary flow reactor setup". In: *Ind. Eng. Chem. Res.* 58.39 (2019), pp. 18125–18133.

[41] J. D. Gibbons and S. Chakraborti. *Nonparametric statistical inference: revised and expanded*. CRC Press, Boca Raton, 2014.

[42] A. Golshan et al. "A review of recent methods for the determination of ranges of feasible solutions resulting from soft modelling analyses of multivariate data". In: *Anal. Chim. Acta* 911 (2016), pp. 1–13.

[43] M. V. Gomez and A. de la Hoz. "NMR reaction monitoring in flow synthesis". In: *Beilstein J. Org. Chem.* 13.1 (2017), pp. 285–300.

[44] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, Cambridge, 2016.

[45] A. Graves and J. Schmidhuber. "Offline handwriting recognition with multidimensional recurrent neural networks". In: *Adv. Neural Inf. Process. Syst.* 21 (2008).

[46] K. Gröchenig. "Linear independence of time-frequency shifts?" In: *Monatsh. für Math.* 177.1 (2015), pp. 67–77.

[47] R. K. Harris et al. "NMR nomenclature. Nuclear spin properties and conventions for chemical shifts (IUPAC Recommendations 2001)". In: *Pure Appl. Chem.* 73.11 (2001), pp. 1795–1818.

[48] A. R. Hategan, A. Pirnau, and D. A. Magdas. "Applications of Machine Learning for Wine Recognition Based on 1H-NMR Spectroscopy". In: *Beverages* 11.2 (2025), p. 45.

[49] J. F. Haw et al. "Reactions of propene on zeolite HY catalyst studied by in situ variable temperature solid-state nuclear magnetic resonance spectroscopy". In: *J. Am. Chem. Soc.* 111.6 (1989), pp. 2052–2058.

[50] C. Heil. *Introduction to Harmonic Analysis*. `https://heil.math.gatech.edu/handouts/chap1.pdf`. Springer, Berlin, 2010.

[51] C. Heil, J. Ramanathan, and P. Topiwala. "Linear independence of time-frequency translates". In: *Proc. Am. Math. Soc.* 124.9 (1996), pp. 2787–2795.

[52] J. Hellwig and K. Neymeyr. "On the uniqueness of continuous and discrete hard models of NMR-spectra". In: *J. Math. Chem.* 63.1 (2025), pp. 267–296.

[53] J. Hellwig et al. "Using machine learning to improve the hard modeling of NMR time series". In: *J. Magn. Reson.* 370 (2025), p. 107813.

[54] J. M. Howie. *Complex analysis*. Springer Science & Business Media, Luxembourg, 2012.

[55] The MathWorks Inc. *Optimization Toolbox version: 9.4 (R2022b)*. Natick, Massachusetts, United States, 2022. URL: `https://www.mathworks.com`.

[56] V. Jain, M. C. Biesinger, and M. R. Linford. "The Gaussian-Lorentzian Sum, Product, and Convolution (Voigt) functions in the context of peak fitting X-ray photoelectron spectroscopy (XPS) narrow scans". In: *App. Surf. Sci.* 447 (2018), pp. 548–553.

[57] G. James et al. *An Introduction to Statistical Learning: with Applications in Python*. Springer Nature, Berlin, 2023.

[58] A. de Juan and R. Tauler. "Multivariate Curve Resolution: 50 years addressing the mixture analysis problem–A review". In: *Anal. Chim. Acta* 1145 (2021), pp. 59–78.

[59] S. Kern et al. "Artificial Neural Networks for Quantitative Online NMR Spectroscopy". In: *Anal. Bioanal. Chem.* 412 (2020), pp. 4447–4459.

[60] S. Kern et al. "Flexible automation with compact NMR spectroscopy for continuous production of pharmaceuticals". In: *Anal. Bioanal. Chem.* 411 (2019), pp. 3037–3046.

[61] A. Khovanskij. "Fewnomials and Pfaff manifolds". In: *Proceedings of the International Congress of Mathematicians*. Vol. 1. 1983, p. 2.

[62] A. G. Khovanskij. *Fewnomials*. Vol. 88. American Mathematical Society, Providence, 1991.

[63] D. P. Kingma and J. L. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[64] M. J. Kochenderfer and T. A. Wheeler. *Algorithms for optimization*. MIT Press, Cambridge MA, 2019.

[65] J. J. Koenderink. "The structure of images". In: *Biol. Cybern.* 50.5 (1984), pp. 363–370.

[66] H.-W. Koh. "Feature extraction in NMR data analysis". PhD thesis. TU Dortmund, Fachbereich Informatik, 2010.

[67] B. R. Kowalski and C. F. Bender. "k-Nearest Neighbor Classification Rule (pattern recognition) applied to nuclear magnetic resonance spectral interpretation". In: *Anal. Chem.* 44.8 (1972), pp. 1405–1411.

[68] M. Kreisel. "Linear independence of time-frequency shifts up to extreme dilations". In: *J. Fourier Anal. Appl.* 25.6 (2019), pp. 3214–3219.

[69] E. Kriesten et al. "Fully automated indirect hard modeling of mixture spectra". In: *Chemom. Intell. Lab. Syst.* 91.2 (2008), pp. 181–193.

[70] E. Kriesten et al. "Identification of unknown pure component spectra by indirect hard modeling". In: *Chemom. Intell. Lab. Syst.* 93.2 (2008), pp. 108–119.

[71] K. Krishnamurthy. "Complete reduction to amplitude frequency table (CRAFT)—a perspective". In: *Magn. Reson. Chem.* 59.8 (2021), pp. 757–791.

[72] K. Kumar. "Principal component analysis: Most favourite tool in chemometrics". In: *Resonance* 22.8 (2017), pp. 747–759.

[73] Ē. Kupče and R. Freeman. "Mapping molecular perturbations by a new form of two-dimensional spectroscopy". In: *J. Am. Chem. Soc.* 135.8 (2013), pp. 2871–2874.

[74] Ē. Kupče and R. Freeman. "Resolving ambiguities in two-dimensional NMR spectra: the 'TILT'experiment". In: *J. Magn. Reson.* 172.2 (2005), pp. 329–332.

[75] C. Landry, W. Welz, and M. Gerdts. "Combining discrete and continuous optimization to solve kinodynamic motion planning problems". In: *Optim. Eng.* 17 (2016), pp. 533–556.

[76] Q. V. Le. "Building high-level features using large scale unsupervised learning". In: *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE. 2013, pp. 8595–8598.

[77] D.-W. Li et al. "DEEP picker is a deep neural network for accurate deconvolution of complex two-dimensional NMR spectra". In: *Nat. Commun.* 12.1 (2021), p. 5229.

[78] P. Linnell. "Von Neumann algebras and linear independence of translates". In: *Proc. Am. Math. Soc.* 127.11 (1999), pp. 3269–3277.

[79] Z. Liu et al. "BraggNN: fast X-ray Bragg peak analysis using deep learning". In: *IUCrJ* 9.1 (2022), pp. 104–113.

[80] R. S. Macomber. *A complete introduction to modern NMR spectroscopy*. John Wiley & Sons, New York, 1997.

[81] W. F. Maddams. "The scope and limitations of curve fitting". In: *Appl. Spectrosc.* 34.3 (1980), pp. 245–267.

[82] M. Maiwald et al. "Quantitative high-resolution on-line NMR spectroscopy in reaction and process monitoring". In: *J. Magn. Reson.* 166.2 (2004), pp. 135–146.

[83] Y. Matviychuk, E. von Harbou, and D. J. Holland. "An experimental validation of a Bayesian model for quantification in NMR spectroscopy". In: *J. Magn. Reson.* 285 (2017), pp. 86–100.

[84] Y. Matviychuk, J. Yeo, and D. J. Holland. "A field-invariant method for quantitative analysis with benchtop NMR". In: *J. Magn. Reson.* 298 (2019), pp. 35–47.

[85] L. McDermott et al. "Neural Architecture Codesign for Fast Bragg Peak Analysis". In: *arXiv preprint* (2023). URL: `https://arxiv.org/pdf/2312.05978`.

[86] D. Meinhardt et al. "Model-based signal tracking in the quantitative analysis of time series of NMR spectra". In: *J. Magn. Reson.* 339 (2022), p. 107212.

[87] M. H. Mendenhall. "Fast computation of Voigt functions via Fourier transforms". In: *J. Quant. Spectrosc. Radiat. Transf.* 105.3 (2007), pp. 519–524.

[88] J. Michael et al. "Evaluating sequence-to-sequence models for handwritten text recognition". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2019, pp. 1286–1293.

[89] E. F. Mooney. *Annual reports on NMR spectroscopy*. Vol. 3. Academic Press, London, 1970.

[90] E. K. Nawrocka et al. "Radon peak-picker based on a neural network". In: *J. Magn. Reson. Open* 12 (2022), p. 100083.

[91] J. Nickolls et al. "Scalable parallel programming with cuda: Is cuda the parallel programming model that application developers have been waiting for?" In: *ACM Queue* 6.2 (2008), pp. 40–53.

[92] S. I. O'donoghue et al. "Unraveling the symmetry ambiguity in a hexamer: calculation of the R6 human insulin structure". In: *J. Biomol. NMR* 16 (2000), pp. 93–108.

[93] A. V. Oppenheimer, R. W. Schafer, and J. R. Buck. *Discrete-Time Signal Processing*. Prentice-Hall Inc., Upper Saddle River, 1998.

[94] F. Paruzzo et al. "Automatic signal region detection in 1H NMR spectra using deep learning". In: *Public Bruker White Paper* (2019).

[95] A. Paszke et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Adv. Neural Inf. Process. Syst.* 32 (2019).

[96] W. R. Pestman. *Mathematical statistics*. Walter de Gruyter, Berlin, 2009.

[97] J. Pitha and R. N. Jones. "A comparison of optimization methods for fitting curves to infrared band envelopes". In: *Can. J. Chem.* 44.24 (1966), pp. 3031–3050.

[98] G. Plonka et al. *Numerical fourier analysis*. Birkhäuser, Basel, 2018.

[99] G. Pólya and G. Szegő. *Problems and Theorems in Analysis II*. Vol. 2. Springer, Heidelberg, 1976.

[100] D. Raljević et al. "Machine learning approach for predicting crude oil stability based on NMR spectroscopy". In: *Fuel* 305 (2021), p. 121561.

[101] D. A. Ramsay. "Intensities and shapes of infrared absorption bands of substances in the liquid phase". In: *J. Am. Chem. Soc.* 74.1 (1952), pp. 72–80.

[102] S. Raschka, Y. H. Liu, and V. Mirjalili. *Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python*. Packt Publishing, Birmingham, 2022.

[103] S. Ray and D. Ren. "On the upper bound of the number of modes of a multivariate normal mixture". In: *J. Multivar. Anal.* 108 (2012), pp. 41–52.

[104] M. Reed and B. Simon. *I: Functional analysis*. Vol. 1. Academic press, London, 1981.

[105] C. A. Reilly and B. R. Kowalski. "Nuclear magnetic resonance spectral interpretation by pattern recognition". In: *J. Phys. Chem.* 75.10 (1971), pp. 1402–1411.

[106] N. V. Reo. "NMR-based metabolomics". In: *Drug Chem. Toxicol.* 25.4 (2002), pp. 375–382.

[107] M. A. Richards. "The discrete-time Fourier transform and discrete Fourier transform of windowed stationary white noise". In: *Technical Report, Georgia Institute of Technology* (2013).

[108] R.E. Richards and W. R. Burton. "Some applications of Intensity Measurements in the Infra-Red". In: *Trans. Faraday Soc.* 45 (1949), pp. 874–880.

[109] S. J. Roberts. "Parametric and non-parametric unsupervised cluster analysis". In: *Pattern Recognit.* 30.2 (1997), pp. 261–272.

[110] G. D. Roston and F. S. Obaid. "Exact analytical formula for Voigt spectral line profile". In: *J. Quant. Spectrosc. Radiat. Transf.* 94.2 (2005), pp. 255–263.

[111] S. B. Russ. "A translation of Bolzano's paper on the intermediate value theorem". In: *Hist. Math.* 7.2 (1980), pp. 156–185.

[112] K.-I. Sato. "Convolution of Unimodal Distributions Can Produce any Number of Modes". In: *Ann. Probab.* (1993), pp. 1543–1549.

[113] A. Savitzky and M. J. E. Golay. "Smoothing and differentiation of data by simplified least squares procedures." In: *Anal. Chem.* 36.8 (1964), pp. 1627–1639.

[114] M. Sawall et al. "Multi-objective optimization for an automated and simultaneous phase and baseline correction of NMR spectral data". In: *J. Magn. Reson.* 289 (2018), pp. 132–141.

[115] M. Sawall et al. "Reduction of the rotational ambiguity of curve resolution techniques under partial knowledge of the factors. Complementarity and coupling theorems". In: *J. Chemom.* 26.10 (2012), pp. 526–537.

[116] M. Schmid, H.-P. Steinrück, and J. M. Gottfried. "A new asymmetric Pseudo-Voigt function for more efficient fitting of XPS lines". In: *Surf. Interface Anal.* 46.8 (2014), pp. 505–511.

[117] N. Schmid et al. "Deconvolution of 1D NMR spectra: a deep learning-based approach". In: *J. Magn. Reson.* 347 (2023), p. 107357.

[118] A. J. Simpson, D. J. McNally, and M. J. Simpson. "NMR spectroscopy in environmental research: From molecular interactions to global processes". In: *Prog. Nucl. Magn. Reson. Spectrosc.* 58.3-4 (2011), pp. 97–175.

[119] J. Aires-de Sousa, M. C. Hemmer, and J. Gasteiger. "Prediction of 1H NMR chemical shifts using neural networks". In: *Anal. Chem.* 74.1 (2002), pp. 80–90.

[120] J. Stoer et al. *Introduction to numerical analysis*. Vol. 1993. Springer, Berlin, 1980.

[121] G. D. Tredwell et al. "Modelling the acid/base 1 H NMR chemical shift limits of metabolites in human urine". In: *Metabolomics* 12 (2016), pp. 1–10.

[122] J. Tudor Davies and J. M. Vaughan. "A New Tabulation of the Voigt Profile." In: *Astrophys. J.* 137 (1963), p. 1302.

[123] S. S. Vallender. "Calculation of the Wasserstein distance between probability distributions on the line". In: *Theory Probab. Its Appl.* 18.4 (1974), pp. 784–786.

[124] C. Villani. *Optimal transport: old and new*. Vol. 338. Springer, Heidelberg, 2008.

[125] M. Voorhoeve. "On the oscillation of exponential polynomials". In: *Math. Z.* 151.3 (1976), pp. 277–294.

[126] J. Wagner et al. "Deep Set Models for Elucidating Unknown Mixtures with NMR Spectroscopy". In: *Machine Learning for Chemistry and Chemical Engineering (ML4CCE)*. 2024.

[127] L. N. Wasserstein. "Markov processes over denumerable products of spaces, describing large systems of automata". In: *Probl. Peredachi Inf.* 5.3 (1969), pp. 64–72. URL: `https://www.mathnet.ru/eng/ppi1811`.

[128] G. K. Wertheim et al. "Determination of the Gaussian and Lorentzian content of experimental line shapes". In: *Rev. of Sci. Instrum.* 45.11 (1974), pp. 1369–1371.

[129] E. E. Whiting. "An empirical approximation to the Voigt profile". In: *J. Quant. Spectrosc. Radiat. Transf.* 8.6 (1968), pp. 1379–1384.

[130] C. E. Wilder. "Expansion problems of ordinary linear differential equations with auxiliary conditions at more than two points". In: *Trans. Amer. Math. Soc.* 18.4 (1917), pp. 415–442.

[131] A. M. Woodward, B. K. Alsberg, and D. B. Kell. "The effect of heteroscedastic noise on the chemometric modelling of frequency domain data". In: *Chemometrics Intell. Lab. Sys.* 40.1 (1998), pp. 101–107.

[132] S. G. Worswick et al. "Deep neural network processing of DEER data". In: *Sci. Adv.* 4.8 (2018).

[133] K. Wüthrich. "NMR studies of structure and function of biological macromolecules (Nobel Lecture)". In: *Angew. Chem., Int. Ed.* 42.29 (2003), pp. 3340–3363.

[134] A. L. Yuille and T. A. Poggio. "Scaling theorems for zero crossings". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 1 (1986), pp. 15–25.

[135] M. R. Zaghloul and A. N. Ali. "Algorithm 916: computing the Faddeyeva and Voigt functions". In: *ACM Trans. Math. Softw.* 38.2 (2012), pp. 1–22.

[136] M. Zaheer et al. "Deep sets". In: *Adv. Neural Inf. Process. Syst.* 30 (2017).

[137] J. Zupan and J. Gasteiger. "Neural networks: A new method for solving chemical problems or just a passing phase?" In: *Anal. Chim. Acta* 248.1 (1991), pp. 1–30.

Acknowledgements

This thesis count not have been completed without the support of many people.

The first and most important person to thank is undoubtedly my supervisor, Prof. Klaus Neymeyr. Without his vision, his ideas, and his talent for bringing structure into my own ideas, this work would hot have been possible. I am especially grateful for his honest criticism and for his patience, when it sometimes fell on deaf ears. Furthermore, it was a pleasure to propose new ideas to my colleagues and to always receive constructive criticism in a casual atmosphere.

I would also like to thank the group of Prof. Labahn, not only for providing access to the GPU servers of our institute, but also for the technical support by Dr. Tobias Strauß, and the fruitful discussions regarding which neural network architectures to use and which ones to avoid.

For ensuring that our designed methods have any practical application, I would like to express my gratitude to Prof. von Harbou's group from RPTU Kaiserslautern. For providing the experimental data sets, I thank Anne Friebel, Ellen Steimers, and Patrick Sterner.

I would also like to thank my colleagues from the Institute of Mathematics for their willingness to discuss mathematical issues, even during the sacred lunch breaks, especially to Prof. Schlage-Puchta, Prof. Kösters, and to Dr. Christoph Schwerdt, for hinting at the right direction of proof and for suggesting useful literature.

Lastly, I would like to express my deepest gratitude to Jan and Max, my family, and especially to Svenja. Thank you for accompanying me on my way through this thesis, for tolerating my presence, and absence, and for listening.

Eigenständigkeitserklärung

Ich versichere hiermit an Eides statt, dass ich die vorliegende Arbeit selbtständig angefertigt und ohne fremde Hilfe verfasst habe, keine außer den von mir angegebenen Hilfsmitteln und Quellen dazu verwendet habe und die den benutzten Werken inhaltlich und wörtlich entnommenen Stellen als solche kenntlich gemacht habe.

Rostock, den 16.07.2025

_____