5-6-2019

A robust and automated deconvolution algorithm of peaks in spectroscopic data

William Johan Burke IV
Rowan University

A ROBUST AND AUTOMATED DECONVOLUTION ALGORITHM OF PEAKS IN SPECTROSCOPIC DATA

by

William Johan Burke IV

A Thesis

Submitted to the
Department of Computer Science
College of Science and Mathematics
In partial fulfillment of the requirement
For the degree of
Master of Science in Computer Science
at
Rowan University
April 26, 2019

Thesis Chair: Serhiy Y. Hnatyshyn, Ph.D.

Dedications

I would like to dedicate this thesis to my parents, whose love, support, and encouragement made it possible for me to pursue my education at Rowan University.

Acknowledgments

I would like to thank my thesis adviser Dr. Serhiy Y. Hnatyshyn for his continued support and assistance. His expertise in the subject matter of metabolomics has helped me greatly in the preparation of my thesis. I appreciate all the time he took to answer my questions about the concepts of liquid chromatography-mass spectrometry and thank him for his guidance and feedback on issues relating to this research and for providing spectroscopic data for testing.

The support of Dr. Vasil Hnatyshin has meant a lot to me. It was upon his recommendation that I started undergraduate research at Rowan University in collaboration with Bristol-Myers Squibb, and he has been there for me to answer questions and offer his guidance since then through my graduate studies. I am grateful for his time, assistance, and encouragement he provided throughout my research.

I thank Dr. Umashanger Thayasivan for his ongoing support, for the time he took helping me understand mathematical concepts related to this research, and for providing assistance in performing statistical analysis of test results.

I thank Dr. Petia Shipkova and Dr. Michael Reily from the Bristol-Meyers Squibb Company Pharmaceutical Research Institute for providing their experimental high-resolution LC-MS data acquired from rat plasma samples, which were used for algorithm testing and validation described in this thesis in chapter 5.

Abstract

William Johan Burke IV
A ROBUST AND AUTOMATED DECONVOLUTION ALGORITHM OF PEAKS IN SPECTROSCOPIC DATA
2018-2019
Serhiy Y. Hnatyshyn, Ph.D.
Master of Science in Computer Science

The huge amount of spectroscopic data in use in metabolomic experiments requires an algorithm that can process the data in an autonomous fashion while providing quality of analysis comparable to manual methods. Scientists need an algorithm that effectively deconvolutes spectroscopic peaks automatically and is resilient to the presence of noise in the data. The algorithm must also provide a simple measure of quality of the deconvolution. The deconvolution algorithm presented in this thesis consists of preprocessing steps, noise removal, peak detection, and function fitting. Both a Fourier Transform and Continuous Wavelet Transform (CWT) method of noise removal were investigated. The performance of the automated algorithm was compared with the manual approach. The tests were conducted using data partitioned into categories based on the amount of noise and peak types. The CWT is shown to be an adequate method for estimating the locations of peaks in chromatographic data. An implementation was provided in Microsoft Visual C# with .NET 5.0.

Table of Contents

Table of Contents (Continued)

Table of Contents (Continued)

List of Figures

List of Figures (Continued)

List of Figures (Continued)

List of Tables

Chapter 1

Introduction

In the 21st century, the prevailing view of health care focuses on personalized medicine where the information regarding an individual's metabolic phenotype[1] is extracted from the analysis of small molecules in body fluids such as plasma and urine [1]. Endogenous[2] metabolites are analyzed using spectroscopic[3] experiments that contribute to drug discovery efforts and gaining new understanding of the relationships between individual genetic variations and environmental triggers of disease [1].

Various spectroscopic techniques such as nuclear magnetic resonance (NMR), liquid chromatography-mass spectrometry (LC-MS), gas chromatography-mass spectrometry (GC-MS), capillary electrophoresis-mass spectrometry (CE-MS), and infrared spectroscopy allow scientists to monitor changes of multiple parameters in endogenous small-molecule metabolites that an organism may experience while in a perturbed state, for example, after administering drugs. The primary goal of metabolomics studies is to detect and measure a living system's metabolic responses to external perturbations. This goal is accomplished by processing spectroscopic data to measure and identify peaks that correspond to the signals of endogenous metabolites [2].

Many types of analytical laboratories face the challenge of reliably processing data. Many software packages have been developed for computer-assisted analysis of spectroscopic data. However, a reliable and fully automated procedure with minimal

[1] Observable presence of organic molecules resulting from chemical reactions of enzymes.
[2] Produced by the host organism.
[3] Spectroscopy is the use of the interaction between matter and electromagnetic radiation to study the composition of the matter.

human supervision has yet to be defined. In particular, the procedure for unsupervised automated processing should include identification, localization, and quantification of spectroscopic peaks while handling noise, missing signals, and abnormalities in collected data. An ideal procedure for automated data processing should be heavily self-optimizing to alleviate the need for human input. Ultimately, the peak analysis algorithm should be able to provide its user with a simple quality metric that specifies the confidence in the accuracy of produced results [3].

Recent improvements in data recording systems (high-speed analog-to-digital converters) have led to such large amounts of data that manual peak analysis has become virtually impossible [4]. The enormity of collected spectroscopic data poses new requirements on computer algorithms for peak analysis, forcing the focus of research to shift from computer-assisted peak analysis to creating completely automated and autonomous processing systems. There are many peak processing algorithms for various types of spectroscopic data. While most published algorithms perform well for certain specific experimental settings, few of them are applicable to a general case. There is a clear need for an algorithm that provides robust peak deconvolution with completely automated output without the need for manual verification of results.

One of the most commonly used analytical tools for metabolomic analysis is LC-MS. It is typically used in conjunction with other analytical techniques such as NMR spectroscopy, GC-MS, CE-MS, and so on. Liquid chromatography-mass spectrometry provides a precise and exhaustive measurement of the sample in terms of molecular weight and structure as well as the quantity and identity of present metabolites. High-resolution accurate mass measurements coupled with ultrahigh pressure liquid

chromatography (UHPLC) has become the preferred platform for nontargeted LC-MS metabolomics due to superior chromatographic and mass spectral resolution as well as speed of analysis [2].

The combination of the LC and MS techniques for the simultaneous separation and detection of metabolite analytes results in complex data sets. This complexity necessitates significant preprocessing before the statistical analysis of multiple samples becomes possible. A peak detection–based preprocessing routine requires a robust method that results in reproducible characterizing peaks [5].

Multiple analytic tools are available for preprocessing data, but several difficulties hinder the integration of off-the-shelf analytics tools into workflows. One of the difficulties is the inability to verify the algorithm and examine the intermediate results because off-the-shelf tools can only be accessed as a nonmodular black box. Another issue is limited access to the underlying context information (e.g., peak shape or neighboring peaks) of intermediate results. Finally, the varying data formats used during different steps of the process increase the difficulty of rearranging the pipeline components to suit new experiments or technologies [6].

Zhang et al. [7] note that the first step in biomarker extraction from the mass spectrometry data is peak detection. This step significantly influences the following steps' results. Proper method design for peak detection greatly depends on the data's properties. The different types of data consist of different characteristics (e.g., width at half height, asymmetry factor, etc.). As a result, each different data type requires different MS instruments and the proper peak detection methods. A unique noise pattern often affects the data. Removing noise significantly improves the peaks' signal-to-noise ratio, making

the data easier to process. Zhang et al. [7] propose one method of noise removal, which is an "adaptive short time discrete Fourier transform combined with wavelet transform to remove the chemical noise and the random noise."

This thesis studies the known approaches to peak detection and deconvolution, and their suitability for automation of processing high volumes of data. The combination of selected best practices from literature results in an algorithm that fully supports automated processing of spectroscopic data. The algorithm includes a noise-filtering module, a 5-point peak detection module, function fitting, and optimization.

Chapter 2 describes the analysis of a typical high-resolution LC-MS data set in order to evaluate experimental data abnormalities and to develop a noise-handling approach. Peak shapes for different spectroscopic techniques were modeled by Gaussian, Lorentzian, and Voigt functions. The chapter also describes the problems encountered during peak detection and the use of the Maximum Entropy Principle to address these problems. Finally, chapter 2 details the process for function deconvolution, including approaches for eliminating noise, peak detection, and function fitting.

Chapter 3 reports on the application of the Fourier transform decomposition algorithm for the analysis of spectroscopic data. The algorithm includes noise filtering by means of a discrete Fourier transform (DFT) low-pass filter and parameter estimation for each detected peak using least squares curve fitting. The algorithm can be modified to estimate the parameters of exponentially modified Gaussian, Lorentzian, and pseudo-Voigt functions. The chapter includes the mathematical background for the use of the DFT in signal processing as well as test results using modeled data.

Chapter 4 discusses the wavelet transform approach for analyzing spectroscopic data. The continuous wavelet transform (CWT) procedure eliminates both high- and low-frequency noise from the data. Peak detection is performed on CWT transformed data. The chapter includes the mathematical background of the CWT as well as a summary of test results for evaluating the approach.

Chapter 5 explores the variability of experimental data and the selection of convolution cases for algorithm testing. It describes the results validation and testing methodology. Two groups of LC-MS data were tested for reproducibility of peak detection deconvolution per distinct type of chromatographic shapes defined in chapter 4. For each type of chromatographic curves, manual (computer-assisted) peak deconvolution was applied to calculate peak parameters. Then an automated deconvolution algorithm processed the data. Differences between manually picked peaks and peaks found by an automated deconvolution algorithm were subject to statistical analysis. Performance of the algorithm was analyzed in the context of both modeled and experimental data. Limitations of the algorithm were also defined.

Chapter 6 provides conclusions for each part of the study. The results of the study show that the CWT is an effective means for estimating the locations of peaks in chromatographic data. The thesis also shows that a mixture of symmetric Gaussian functions provides an adequate model for chromatographic data.

Chapter 2

Analysis of Spectroscopic Data Sets[4]

Assuming a spectrum is represented by a given mixture of Gaussian functions that may overlap, the goal of this study was to design a deconvolution algorithm for processing various liquid chromatography-mass spectrometry (LC-MS) data using the maximum entropy principle. Typical spectroscopic data were analyzed to assess usual abnormalities in order to develop an approach to deal with noise. The Gaussian, Lorentzian, and Voigt functions were used to model different peak shapes.

Background: Peak Detection and Deconvolution – Overview

Peak detection is essential to obtaining information from mass spectral data. Manual peak detection is very time-consuming, and it becomes increasingly unattainable to manually pick peaks as mass spectrometry data sets become ever larger [8]. Furthermore, this approach runs into the problem that a human may not be able to identify peaks by looking at the data. Peaks with height differences by one order of magnitude or more may be made invisible in the process of rendering the data points, which is called the zoom problem. Additionally, manual peak detection suffers from the issue that substances may be hidden in noise. Figures 1 and 2 show examples with about 20 peaks each discernible by eye.

[4] Chapter 2 is based on D. Gaffney and W. J. Burke's research in 2015-2016.

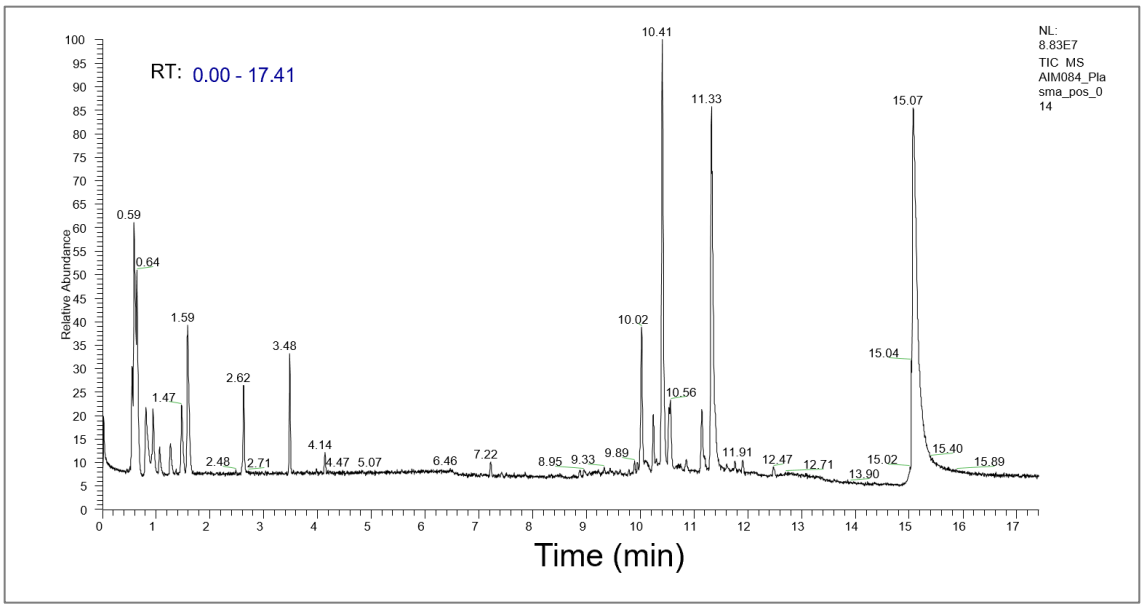

Figure 1. Chromatogram of the total ion current for a rat plasma sample, collected using the Q-Exactive™ mass spectrometer (Thermo Scientific, Bremen, Germany) interfaced with the Thermo Scientific Open Accela 1250 UHPLC system (Thermo Scientific, San Jose, CA).

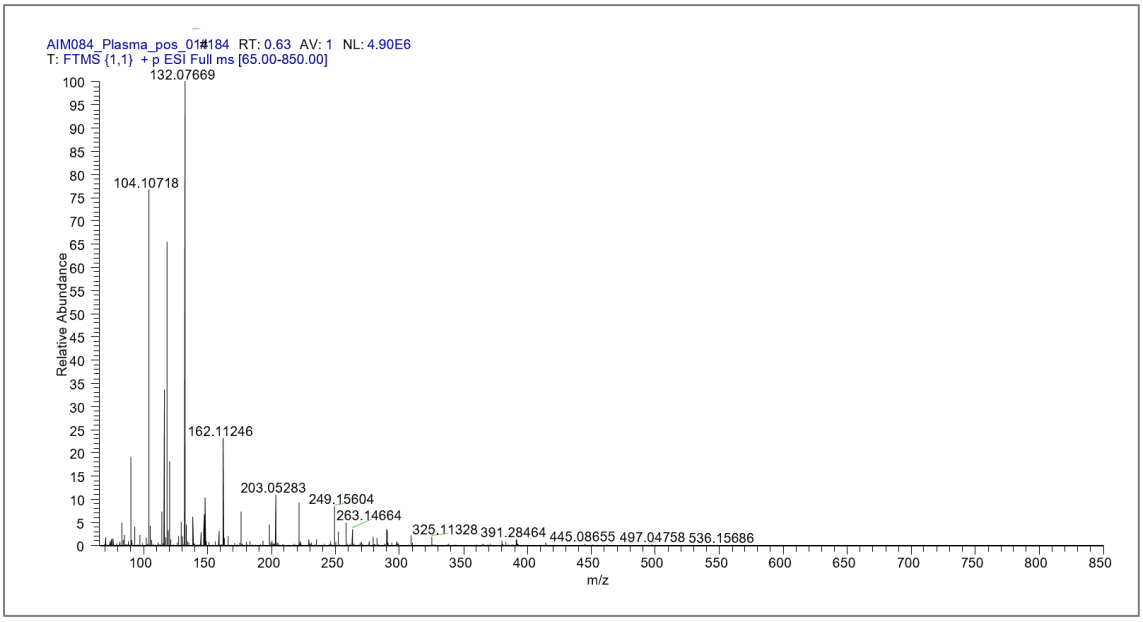

Figure 2. Underlying mass spectrum (electro-spray ionization, positive mode) for a peak at 4.14 minutes. The main ion peak in the spectrum (*m/z* = 185.0968) is consistent with a monoisotopic mass of spiked compounds in chemical d5-hippuric acid, which was introduced into the experiment as the internal standard.

Another common approach is to use a 5-point peak detection algorithm, which selects five consecutive points in the data set and marks the selection as a peak if the set's middle point has the highest value of the five points. Formally, the algorithm can be described as follows. Assuming the data set $D = \{(x_i, y_i), i \in [0, n-1]\}$, where $\forall i > 0, \{y_{i-1} < y_i\}$, (i.e., the data points are sorted by *y* values), then $\max(x_1, x_2, \ldots, x_n) = x_i \rightarrow x_i$ is a peak.

Real data were input into an implementation of the aforementioned algorithm, yielding inconsistent results. The approach found nearly all of the given mixture's peaks, regardless of their magnitude relative to the other data. Therefore, the algorithm identified both actual peaks and insignificant spikes as peaks. The insignificant spikes occur due to the nature of raw data, which do not provide an image that is smooth enough for analysis. The lack of a smooth image is a result of the raw data neither being smooth nor interpolated. Another problem with the 5-point approach is that it will not detect the contribution of a smaller curve that is dominated by a larger curve. In this case, the convoluted curve covers the original component of the smaller curve. The process will not find a peak or part of a curve in the region because there is no remaining portion of the smaller curve left to find. Since the 5-point peak detection algorithm cannot correctly identify peaks, a different peak detection method is needed.

LC-MS Peak Detection Problems

Different types of noise in chromatograms result in peak detection problems in LC-MS. One such type of noise is baseline noise, which is noise that filtering and smoothing fail to remove [9]. Baseline noise makes measurement of peak areas difficult, thereby reducing confidence in analysis results [10]. Other types of noise include short-

term noise, long-term noise, and drift, which are defined, respectively, as "random variations in detector signal whose frequency is greater than 1 cycle minute^{-1}," "[v]ariations in detector signal whose frequency lies between 6 and 60 cycles hour^{-1}," and "change in baseline position" [9]. Another problem for peak detection of LC-MS data is that it is difficult to identify peaks that are close to baseline or are overlapping [11]. For instance, when measuring peak areas, the peak start and endpoints must be identified, which can be difficult to accomplish if the peak overlaps with other peaks [12].

These problems can make it difficult to estimate the composition of the observed feature. A solution, therefore, is to identify and separate overlapping chromatographic peaks. This separation, known as deconvolution, is a difficult problem in and of itself. Two proposed methods for this problem are tangent skimming and the perpendicular drop method [11]. The former method involves measuring the area between the curve of the data and a baseline drawn across the peak's bottom. This is useful in the case of a single peak being superimposed over a straight or broadly curved baseline. The latter method, on the other hand, involves drawing two vertical lines from the bounds of the peak down to the *x*-axis and measuring the total area of the figure created by the lines, the curve, and the *x*-axis [12]. However, both deconvolution methods are only approximate and are best used when there is only slight overlap in the peaks [11]. Another approach to deconvolution is to employ the maximum entropy, or maximum likelihood, principle.

Maximum Entropy (Likelihood) Algorithm

Deconvolution involves separating real observations from a point spread function (PSF) in a digital image [13]. The maximum entropy method, as introduced by Agmon et al. [14], is a means of accomplishing deconvolution [15].

Maximum entropy (likelihood) principle. Maximum entropy algorithms are often derived from an application to a generic estimation problem, which is estimating an unknown, deterministic vector parameter in the linear model $\mathbf{y} = \mathbf{G}x + \mathbf{w}$, where $\mathbf{G}$ is a linear transformation, and $\mathbf{w}$ is a Gaussian noise vector. This equation in a simple form describes a large variety of signal processing and statistics problems. In their letter, Wiesel et al. [16] consider the problem of finding the maximum likelihood (ML) estimator in the linear model given a model matrix $\mathbf{G}$ composed of independent and identically distributed Gaussian elements. In the linear model, $\mathbf{G}$ is an $N \times K$ matrix that has a known mean $\mathbf{H}$ and independent elements that have variance $\sigma_h^2 > 0$. The Gaussian noise vector has a mean of 0 and independent elements that have variance $\sigma_w^2 > 0$ [16].

An estimator of $\mathbf{x}$, $\hat{\mathbf{x}}(\mathbf{y}, \mathbf{H}, \sigma_h^2, \sigma_w^2)$, is a function of the observation vector and the given statistics that results in values close to $\mathbf{x}$. In ML estimation, the estimator is chosen as the parameter vector maximizing the likelihood of the observations. This is expressed mathematically in Equation (1):

$$\max_x \log p(\mathbf{y}; \mathbf{x}), \tag{1}$$

where $p(\boldsymbol{y}; \boldsymbol{x})$ is the probability density function of $\mathbf{y}$ parameterized by $\mathbf{x}$. Because $\mathbf{y}$ is a Gaussian vector having mean $\mathbf{Hx}$ and covariance $(\sigma_h^2 \|\mathbf{x}\|^2 + \sigma_w^2)\mathbf{I}$, the ML estimator is the solution to Equation (2):

$$\min_x \left\{ \frac{\|\mathbf{y} - \mathbf{Hx}\|^2}{\sigma_h^2 \|\mathbf{x}\|^2 + \sigma_w^2} + N \log(\sigma_h^2 \|\mathbf{x}\|^2 + \sigma_w^2) \right\}. \tag{2}$$

Wiesel et al. [16] solve this difficult optimization problem by reformulating Equation (2) into Equation (3):

$$\min_{t\geq 0}\left\{\frac{f(t)}{\sigma_h^2 t+\sigma_w^2}+N\log(\sigma_h^2 t+\sigma_w^2)\right\}, \tag{3}$$

where $f(t)=\min_{\mathbf{x}:\|\mathbf{x}\|^2=t}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2$ with optimal argument $\mathbf{x}(t)$. The ML estimator in the linear model is simply $\mathbf{x}(t^*)$, where t^* is the solution to Equation (3). The solution, shown in Equation (4), is found by using a simple line search:[5]

$$\mathbf{x}(t)=(\mathbf{H}^T\mathbf{H}+\alpha\mathbf{I})^{\dagger}\mathbf{H}^T\mathbf{y}, \tag{4}$$

where $\alpha\geq-\lambda_{\min}(\mathbf{H}^T\mathbf{H})$ is the unique root of Equation (5):

$$\|\mathbf{x}(t)\|^2=t. \tag{5}$$

Upon finding an α satisfying Equation (5), $f(t)$ is found by evaluating $\|\mathbf{y}-\mathbf{H}\mathbf{x}(t)\|^2$ with the appropriate $\mathbf{x}(t)$ [16].

For most biologically relevant samples, quantitative analysis becomes extremely complicated due to the high degree of spectral overlap [17]. Chylla et al. [17] developed an algorithm called fast maximum likelihood construction (FMLR) that performs spectral deconvolution of 1D–2D NMR spectra for the purpose of accurate signal quantification. They apply maximum likelihood to NMR spectra, but a similar concept is useful for chromatographic spectra.

When there is some, but not enough, information to characterize a probability distribution, the maximum entropy principle can be used. This principle states that the correct distribution is the one that contains the maximal amount of unpredictability while still conforming to the known characteristics of the distribution [18]. Applying this principle here, the problem to be solved is: *Given a set of data, what is the most probable*

[5] The operation † refers to the generalized inverse.

parent mass spectrum? The solution to this problem involves creating probability distributions based on what is known (calculated Gaussian distributions), inferring the missing information based on the known characteristics, and inputting the known information into the maximum entropy method.

Mathematical framework. A proficient understanding of the mathematical background for the maximum entropy method is required to successfully implement the method for deconvolution. Deconvolution becomes a difficult problem because of the presence of noise in images [19].

Convolution. Convolution involves the formation of a new signal from two input signals. With linear expressions, convolution is used as follows: an input signal, $x[n]$, enters a linear system with an impulse response, $h[n]$, resulting in an output signal, $y[n]$. This can be expressed in equation form as $x[n] * h[n] = y[n]$. That is, the output signal equals the input signal convolved with the impulse response. The star represents the convolution operation. If $x[n]$ is an N point signal with points numbered 0 to $N-1$, and $h[n]$ is an M point signal with points numbered 0 to $M-1$, the convolution of the two, $y[n] = x[n] * h[n]$, is an $N + M - 1$ point signal with points numbered from 0 to $N + M - 2$, given by Equation (6) [20]:

$$y[i] = \sum_{j=0}^{M-1} h[j]x[i-j]. \tag{6}$$

The index, i, identifies the sample in the output signal being calculated [20].

Deconvolution. Given an image, where the "real image" O is observed through an optical system, and an intensity distribution I corresponding to O, then the relation

between the data and the image in the same coordinate frame is a convolution if the imaging system is linear and shift-invariant, defined by Equation (7):

$$\begin{aligned} I(x,y) &= \int_{x_1=-\infty}^{+\infty}\int_{y_1=-\infty}^{+\infty} P(x-x_1, y-y_1)O(x_1,y_1)dx_1dy_1 + N(x,y) \\ &= (P * O)(x,y) + N(x,y), \end{aligned} \tag{7}$$

where P is the point spread function (PSF) of the imaging system and N is the additive noise. The goal in deconvolution is to determine $O(x, y)$ based on the known values of I and P. This problem is difficult to solve and requires addressing the following two main difficulties: (1) the cutoff frequency of the PSF and (2) additive noise. In practice, there is no unique and stable solution to the equation above [19].

Maximum entropy method. The research presented in this thesis began with developing a prototype algorithm to process artificial data consisting of custom x and y values that form a mixture of Gaussian functions with "known" initial parameters. These data were used to develop and test the expectation-maximization (EM) portion of the algorithm.

***The expectation-maximization (EM) algorithm*.** The EM algorithm is an iterative approach for finding maximum likelihood parameter estimates. It consists of repetition of the alternating steps of expectation and maximization. Given initial parameter estimates, the EM algorithm optimizes the parameters to obtain the best approximation of the data [21]. Expectation is performed with respect to the unknown underlying variables, using the parameters' current estimate and the observations as constraints. Then, the maximization step calculates a new estimate for the parameters. The two steps alternate and repeat until convergence [22]. The expectation and maximization steps are defined

by Equations (8) and (9), respectively:

$$Q_i\left(z^{(i)}\right) := p\left(z^{(i)}|x^{(i)};\theta\right) \tag{8}$$

$$\theta := \arg\max_{\theta} \sum_{i} \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \log \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}, \tag{9}$$

where Q is a chosen probability distribution, z is a set of latent random variables, x is the set of training data, and θ is the set of distribution parameters [23].

Testing implementation of the EM algorithm. When tested on modeled data, the implementation of the EM algorithm quickly converged to correct parameter values, providing an accurate estimation of individual function components in the given mixture. However, when run on real data, the algorithm did not converge and could not accurately compute individual functions. This showed the dependence of the algorithm on proper initial parameter estimates, as the first implementation used arbitrary initial parameter values. To alleviate this problem, a k-means clustering algorithm was implemented to more accurately identify point membership across the individual functions, in the hope that a more accurate knowledge of point membership would lead to better peak estimates and initial parameter values.

K-means algorithm for initial parameter estimation. The k-means algorithm is a clustering algorithm that separates *n* data points into *k* clusters such that each data point is assigned the cluster whose mean it is closest to. The algorithm works by taking an initial guess for what the optimal clusters would be and then assigns each point to the cluster with the smallest Euclidean distance between it and the cluster's centroid, where the centroid is the mean position of all of the points in the cluster. Next, the algorithm computes the mean point of all the points belonging to each cluster and identifies the

calculated point as a new centroid of the cluster. Data points' membership is recalculated with the new centroids. The process of cluster assignment and mean computation repeats until cluster centroids do not change [24].

K-means algorithm summary. K-means is formally described by the following algorithm [24]:

1. Select K points as initial centroids.
2. **Repeat**
3. Form K clusters by assigning each point to its closest centroid.
4. Recompute the centroid of each cluster.
5. **until** centroids do not change.

Steps 3 and 4 are defined by Equations (10) and (11), respectively [25]:

$$c^{(i)} := \arg\min_j \left|\left|x^{(i)} - \mu_j\right|\right|^2 \tag{10}$$

$$\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}} \tag{11}$$

***Testing prototype implementation*.** When testing the prototype implementation, it became clear that the algorithm worked well for custom data but still failed to converge for real data input. Further investigation revealed that the assumption that the initial mixture is described by a Gaussian function of evenly distributed data points is incorrect. This is because data points produced by a mass spectrometer are not guaranteed to be evenly distributed. Thus, the prototype could not correctly identify the number of peaks and initial values for individual function parameters. Input data must therefore be preprocessed before running the EM algorithm. This is why the EM algorithm worked

well for evenly distributed custom data but not for real data. The prototype implementation did not directly solve the problem, but it provided valuable insight into the nature of the problem.

Function Deconvolution

The process of deconvolving a mixture of Gaussian functions into its comprising individual curves involves data preprocessing, peak detection and function fitting, the expectation maximization (EM) algorithm, and the maximum entropy principle. Before peaks can be identified, raw data need to be preprocessed. Preprocessing steps include interpolation, smoothing, and spline calculation. After the data have been preprocessed, peaks are found by calculating the curve's derivative and finding the points where the derivative changes sign from positive to negative. The found peaks are each fit into a Gaussian function. Taking into account initial parameter estimates of the Gaussian functions, the EM algorithm optimizes these parameters to achieve the best approximation of the data. Finally, the maximum entropy principle is used to optimize the data for the entire chromatogram given the finished Gaussian solutions for the picked peaks [21]. Figure 3 shows a convolution of Gaussian curves. Upon first glance, it is obvious that the mixture contains three convoluted curves, but it is unclear what the exact individual curves of the mixture are. The objective is thus to break down the convoluted mixture of Gaussian functions into individual curves.

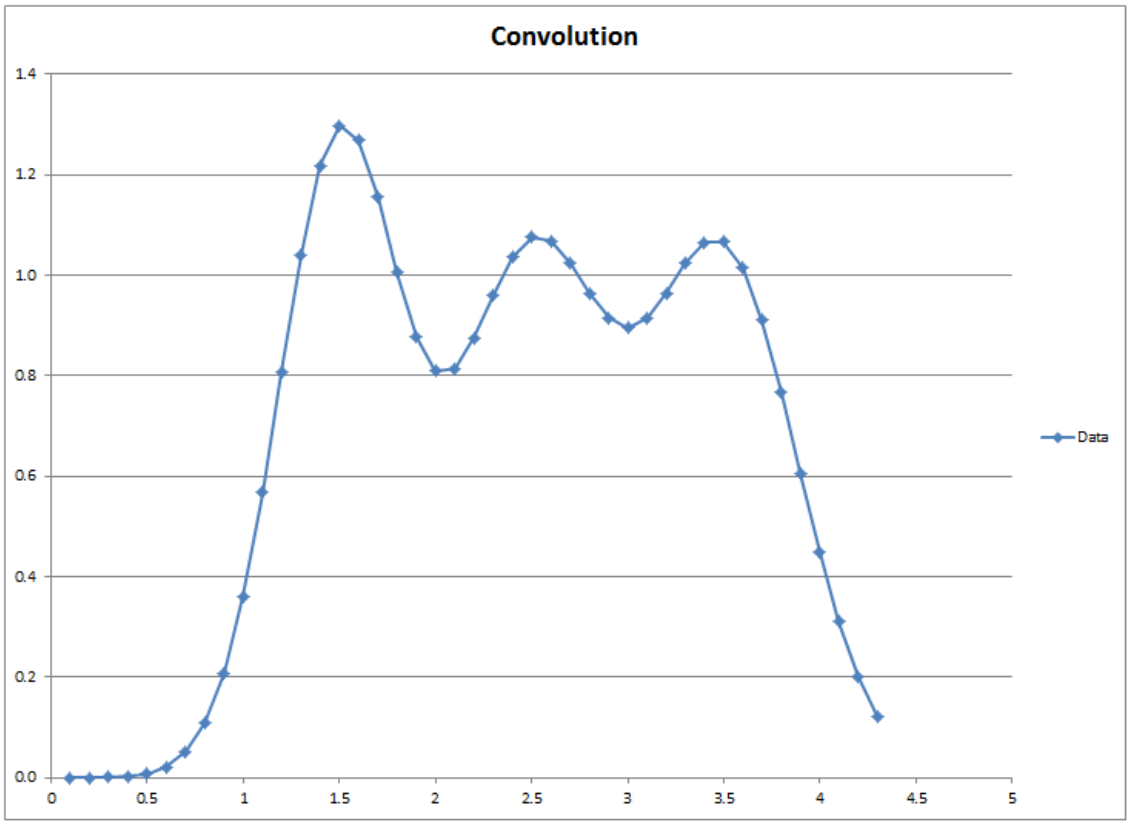

Figure 3. Example of a Gaussian function mixture.

An example deconvolution resulting from the curve in Figure 3 is presented in Figure 4, which shows the three curves F1, F2, and F3 that form a Gaussian mixture titled *Data*.

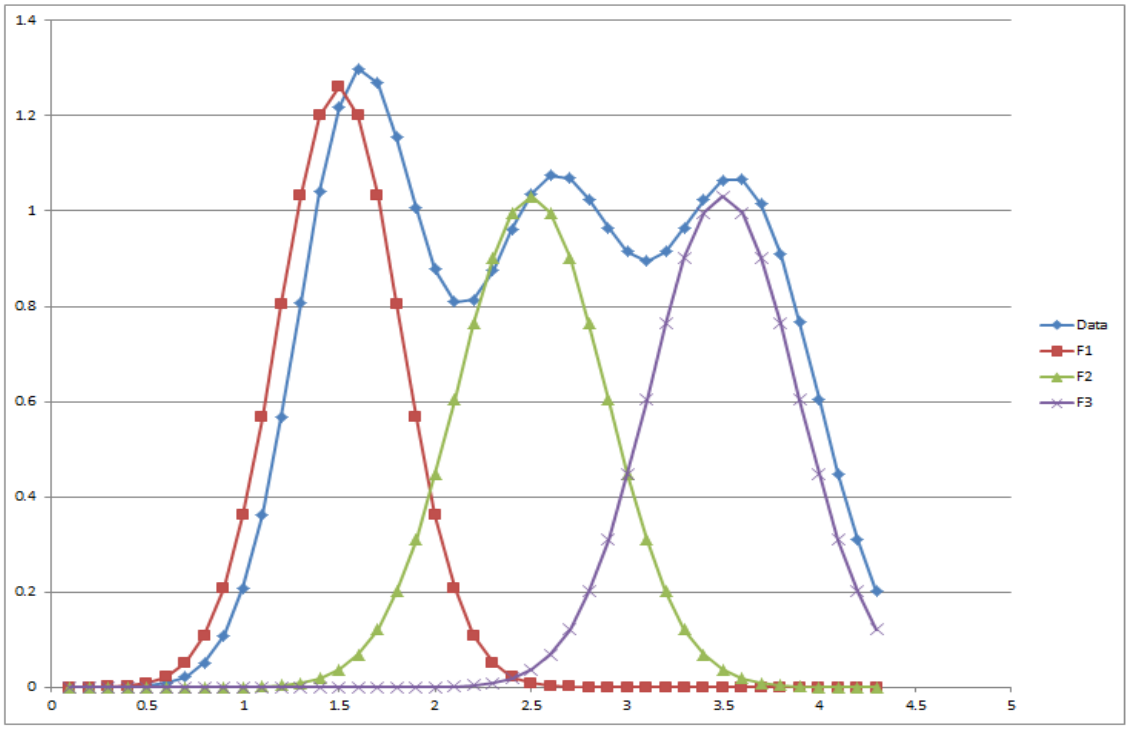

Figure 4. Example of a Gaussian function mixture deconvolution.

Addressing Noise Issues

Preprocessing of MS data involves transforming a large amount of raw spectral data into much smaller, statistically manageable peaks. Since each spectrum contains tens of thousands of data points, mass spectrometry is inherently noisy. Therefore, a variety of algorithms, each with different principles, implementations, and performance, have been created to address the problem of noise [26].

Data preprocessing. Data preprocessing involves several steps before the data's peaks can be found. These steps include interpolation, smoothing, and spline calculation [21].

***Lagrange interpolation*.** First, the time interval between points must be made uniform via interpolation. To make *x* values evenly spread, new points must be inserted,

ensuring that each fits the trend of the data. The implementation constructs Lagrange polynomials that model the behavior of a curve passing through $n + 1$ data points (x_0, y_0), (x_1, y_1), … , (x_n, y_n) [27]. The nth degree Lagrange polynomial is defined by Equation (12) [28]:

$$P(x) = \sum_{j=1}^{n} P_j(x), \tag{12}$$

where

$$P_j(x) = y_j \prod_{k=1, k \neq j}^{n} \frac{x - x_k}{x_j - x_k}.$$

Figure 5 displays an application of complete interpolation to a chromatogram in an interval between 3 and 4 minutes. The green triangle points represent the original data, which have inconsistent spread. The data set after interpolation, represented by red points, has a uniform spread. The points of the output have a uniform distance between x values.

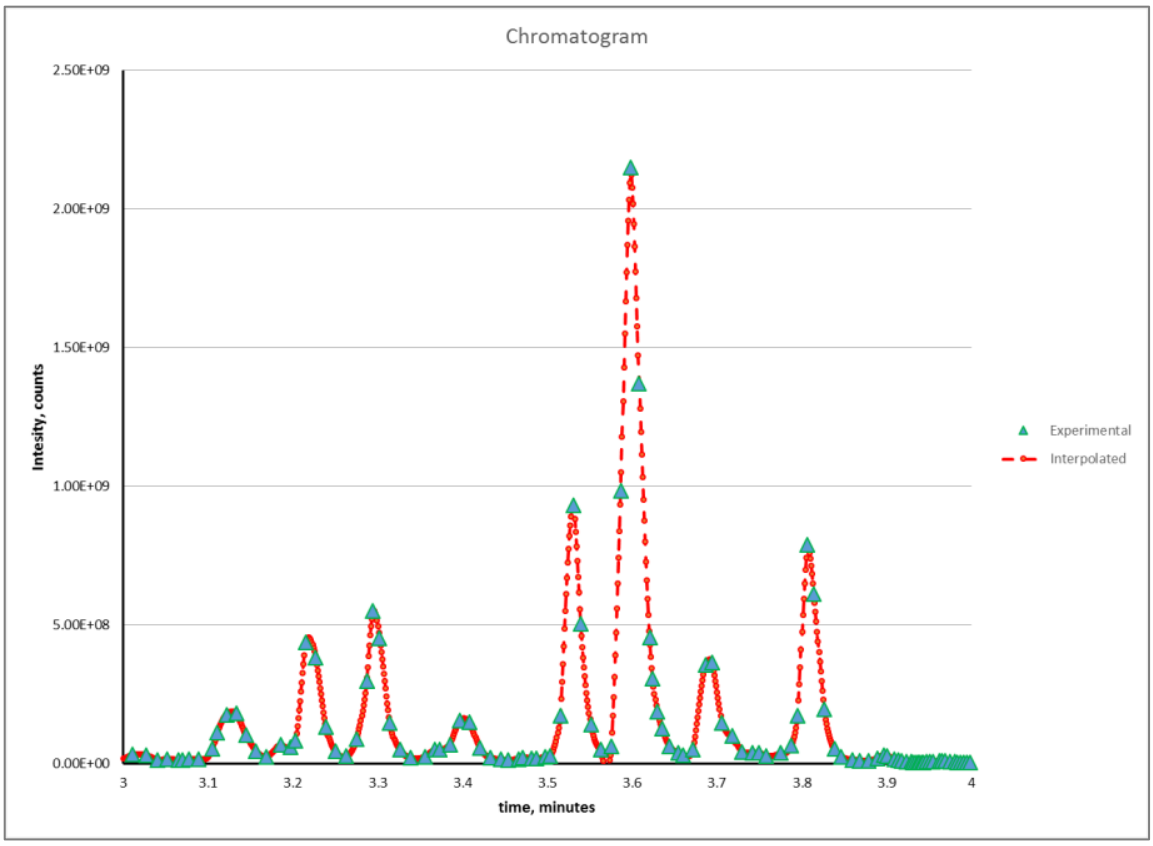

Figure 5. An example of applying the Lagrange formula to interpolation of a chromatogram.

However, the resulting data still have some noise. The data are thus treated with a 5-step polynomial smoothing function, the Savitzky-Golay filter, in order to increase the signal-to-noise ratio of the interpolation algorithm's output.

***Savitzky-Golay filter*.** The Savitzky-Golay filter increases the signal-to-noise ratio of input data without distorting the signal very much. It is often applied to digital data sets for the purpose of smoothing. The filter achieves smoothing by fitting successive subsets of adjacent data points with a low-degree polynomial via the method of linear least squares. The data are a set of $n(x_j, y_j)$ points $j \in 1, \ldots, n$, where xj is an independent variable and y_j is an observed value [29]. The points are treated with a set of m

convolution coefficients according to Equation (13):

$$Y_j = \sum_{i=-\frac{m-1}{2}}^{\frac{m-1}{2}} C_i y_{j+i}, j \in \left[\frac{m-1}{2}, n - \frac{m-1}{2}\right], \quad (13)$$

where $m = 5$, $i \in [-2, 2]$. With the 5-point smoothing formula, the jth smoothed data point Yj is given by Equation (14):

$$y_j = \frac{1}{35}\left(-3y_{j-2} + 12y_{j-1} + 17y_j + 12y_{j+1} - 3y_{j+2}\right), \quad (14)$$

where

$$C_{-2} = -\frac{3}{35}, C_1 = \frac{12}{35}, \text{etc.}$$

Figure 6 shows an example of 5-point polynomial smoothing. It shows an example of smoothing in a time interval of 3.5 to 3.7 minutes. The green points are the output of the interpolation algorithm, while the red points are the points after being treated by the smoothing algorithm.

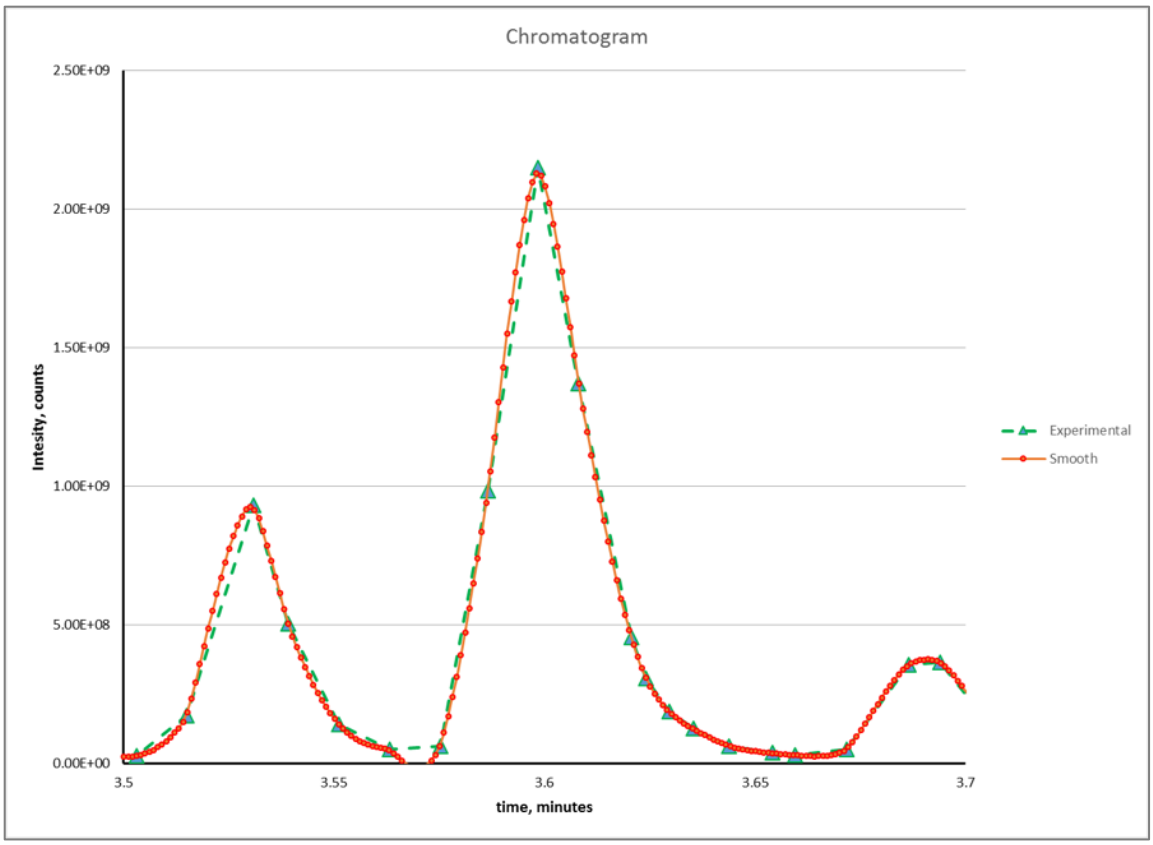

Figure 6. Experimental, post-interpolation data plotted in tandem with smoothed data.

Spline. The result of the Savitzky-Golay smoothing algorithm is a uniformly spread, low-noise set of discrete points. The remainder of the algorithm requires a continuous function as input, so the *y* value for a given *x* value is calculated using two linearly independent cubic polynomial terms. These terms avoid spoiling the agreement with the functional values y_j and y_{j+1}.

The spline function first calculates three sets of coefficients based on the smoothed input data, and then the spline at any *x* value can be calculated based on the coefficients, according to Equation 15 [30]:

$$y = Ay_i + By_{i+1} + Cy_j'' + Dy_{j+1}'', \tag{15}$$

where

$$A = \frac{x_{j+1} - x}{x_{j+1} - x_j},$$

$$B = 1 - A,$$

$$C = \frac{1}{6}(A^3 - A)\left(x_{j+1} - x_j\right),^2 \text{ and}$$

$$D = \frac{1}{6}(B^3 - B)\left(x_{j+1} - x_j\right).^2$$

Figure 7 shows the result of the spline algorithm. The green points are the smoothed and interpolated experimental data, while the red points are the points of the continuous spline function.

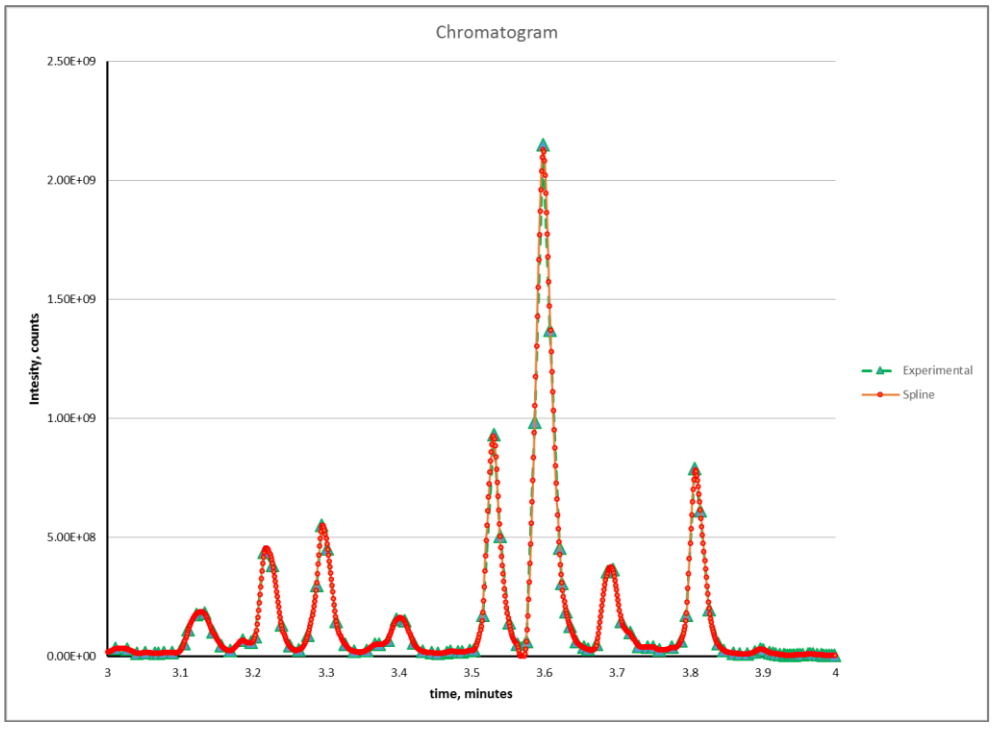

Figure 7. Smoothed, interpolated, experimental data overlaid with continuous data adjusted using the spline function.

Peak detection. After data preprocessing, peaks are found by calculating the curve's derivative. Each peak, after being found, can be fit into a Gaussian function.

***Differentiation*.** Peaks occur at the function's local maxima, which can be found based on Equation (16):

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{f(x-2h) - 8f(x-h) + 8f(x+h) - f(x+2h)}{1200h}, \tag{16}$$

where

$$h = \frac{x_1 - x_0}{10}.$$

Figure 8 shows the spline function (in blue) plotted against its first derivative (in red).

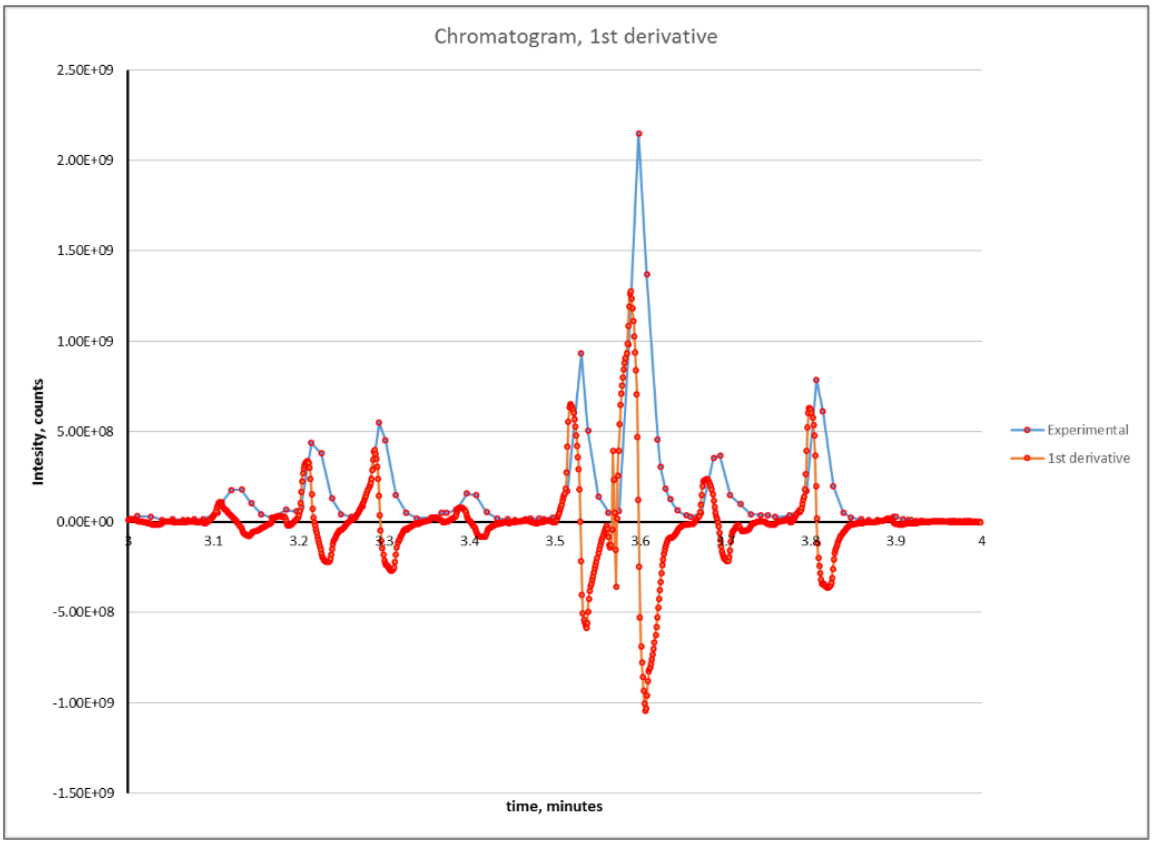

Figure 8. First derivative (slope) overlaid against experimental data approximated by the spline function.

***Peak picking*.** Peak picking involves extracting frequencies of peaks, either from the entire spectrum or from selected regions. The frequencies are then typically displayed on the plot. This process does not consider all identified peaks [31].

Given the spline and the derivative, peaks of the chromatogram can be found using the properties of the derivative function. At each point the following test is conducted: the *y* value at the point is examined to see if it is greater than that of the point before it as well as that of the point after it, and the derivative value at the point is checked to see if it changes from positive to negative. If the test is satisfied, then a potential peak has been found. Following the computation of the set of potential peaks, noninfluential peaks are eliminated by only keeping peaks whose apex values are greater than 1% of the overall maximum *y* value. All remaining peaks are considered influential. For each influential peak, the peak's start and endpoints are determined by finding the local minima closest to the peak's apex. Table 1 shows peak parameters detected in an experimental chromatogram between 3 and 4 minutes.

Table 1

Peak Parameters Calculated in Peak-Picking Procedure

Peak Start	Intensity at Peak Start	Peak Apex	Intensity at Peak Apex	Peak End	Intensity at Peak End
3092	1.42E+07	3127	1.87E+08	3166	2.33E+07
3167	2.32E+07	3184	6.75E+07	3193	5.56E+07
3194	5.58E+07	3217	4.55E+08	3257	2.47E+07
3258	2.49E+07	3294	5.51E+08	3344	1.70E+07

Table 1 (continued)

Peak Start	Intensity at Peak Start	Peak Apex	Intensity at Peak Apex	Peak End	Intensity at Peak End
3345	1.70E+07	3368	4.98E+07	3370	4.96E+07
3371	4.96E+07	3400	1.63E+08	3452	1.23E+07
3485	1.71E+07	3496	2.43E+07	3499	2.37E+07
3500	2.40E+07	3529	9.26E+08	3566	1.00E+03
3572	1.03E+07	3597	2.13E+09	3663	2.57E+07
3664	2.59E+07	3690	3.75E+08	3735	3.76E+07
3736	3.76E+07	3746	3.95E+07	3760	2.60E+07
3761	2.60E+07	3774	3.67E+07	3776	3.60E+07
3777	3.59E+07	3806	7.83E+08	3873	9.49E+06
3874	9.46E+06	3896	3.01E+07	3937	4.07E+06

The data are visualized in Figure 9.

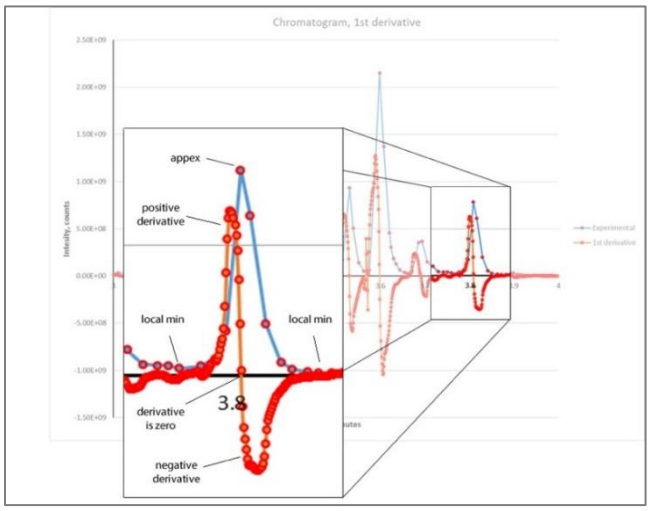

Figure 9. Close-up visualization of peak picking.

Function fitting. After peaks have been found, each is fit into a Gaussian function. Fitting involves using the data set to calculate parameters that represent a Gaussian distribution that closely models the data set.

***Anatomy of a Gaussian distribution*.** The Gaussian distribution is a continuous function that approximates the exact binomial distribution of events. The Gaussian distribution is also commonly called the "normal distribution" [32]. Its probability density function is defined by Equation (17):

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{17}$$

where μ is the mean, and σ^2 is the variance [33].

***Fitting Gaussian functions for peak identification*.** The mathematical procedure to fit experimental data with a Gaussian-like exponential function is described by Jean Jacquelin [34]. For a given data set of (x_0, y_0), (x_1, y_1), … , (x_n, y_n), a direct fit with the function

$$y = ce^{\frac{(x-a)^2}{b}} \tag{18}$$

requires transformation into a new coordinate system as defined by Equations (19) and (20):

$$s_1 = 0, s_i = s_{i-1} + \frac{1}{2}(y_i - y_{i-1})(x_i - x_{i-1}), \tag{19}$$

$$t_1 = 0, t_i = t_{i-1} + \frac{1}{2}(x_i y_i + x_{i-1} y_{i-1}). \tag{20}$$

Coefficients *a*, *b*, and c are obtained by solving the following system of equations:

$$\begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} (s_i)^2 & \sum_{i=1}^{n} s_i t_i \\ \sum_{i=1}^{n} s_i t_i & \sum_{i=1}^{n} (t_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} (y_i - y_1) s_i \\ \sum_{i=1}^{n} (y_i - y_1) t_i \end{bmatrix}. \quad (21)$$

The values of the coefficients are defined by Equation (22):

$$a = -\frac{2}{B}, b = -\frac{A}{B}, c = \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} e^{-\frac{(x_i - a)^2}{b}}}. \quad (22)$$

The coefficients are used to generate the Gaussian peak approximations. An example of fitting experimental data from an extracted ion chromatogram peak is shown in Figure 10. The coefficients of the Gaussian function were determined as *a* = 9.0957, *b* = 18.2562, and *c* = 2.0436.

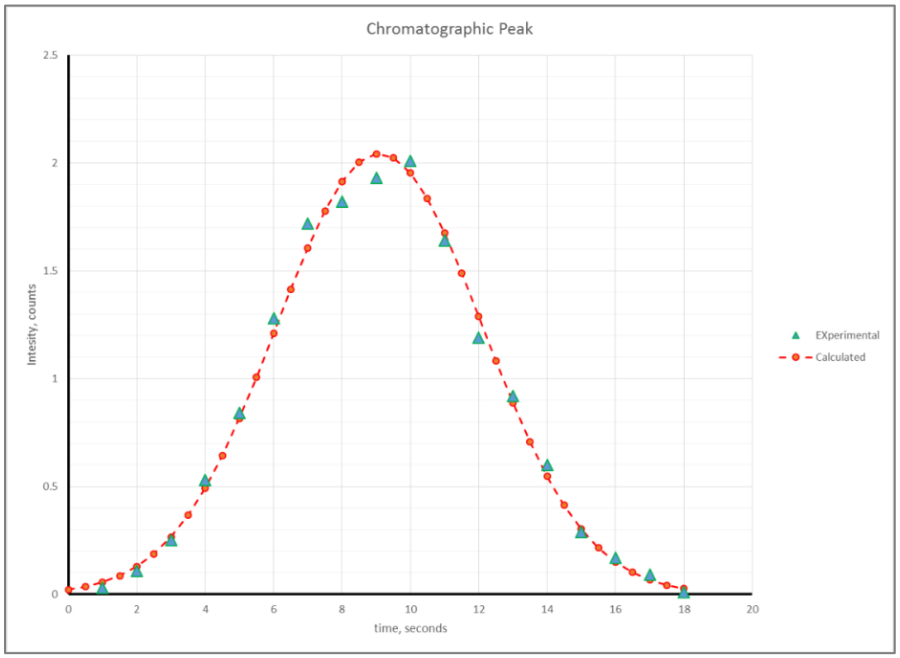

Figure 10. An example of fitting experimental data from an extracted ion chromatogram peak.

Fitting Gaussian functions – future work. The function fitting procedure described above assumes Gaussian functions are symmetrical in nature. However, asymmetric Gaussian functions are more commonly found in experimental data. Asymmetry broadens the base of a peak and increases peak overlap, thereby resulting in more difficult measurement. The asymmetry of a curve can be described in terms of a Tailing Factor:

$$\text{Tailing Factor} = \frac{w_{0.05}}{2A} = \frac{B + A}{2A}. \tag{23}$$

The tailing factor is sometimes called the Asymmetry Ratio, and it compares the peak half widths on either side of the peak. The Asymmetry Ratio varies with peak height. Measurement of asymmetry is typically done near the peak base (at about 10% of peak height), where asymmetry is greatest [10].

Implementation

The procedures described in this chapter were implemented using the C# programming language in the Visual Studio 2013 Integrated Developer Environment. Input data were tabulated points extracted from an LC-MS experiment collected on the Thermo Fisher Orbitrap instrument in the form of a Microsoft Excel file. Data structures used included a simple dynamic list of input points and a Peak class that stores the calculated peak parameters. Data output consisted of detected peaks and parameters of the approximated Gaussian function reported in a Microsoft Excel file.

Resulting Software

Work on the project presented in this chapter resulted in a C# Windows Forms application that reads raw data and performs data preprocessing. After preprocessing, the

software detects peaks and fits experimental data into a Gaussian approximation and then optimizes Gaussian parameters with the EM algorithm. Finally, the software uses the maximum entropy principle to approximate and analyze the entire spectra.

Chapter 3

Deconvolution Algorithm for Spectroscopic Data with Noise

Deconvolution involves disassembling a spectrum into peaks and resolving overlapping signals. The original multistep algorithm to process data and determine configuration parameters was presented in chapter 2. This work resulted in software that reads raw data and performs data preprocessing, detects peaks and fits experimental data into a Gaussian approximation, optimizes Gaussian parameters with the EM algorithm, and uses maximum entropy to approximate and analyze entire spectra. However, stability testing revealed that the algorithm falters in the presence of noise. A major improvement to the algorithm has been made with the addition of low-pass filtering with Fourier transforms, which enable noise elimination. The new algorithm was implemented in C#, and software was tested using a variety of modeled and experimental data. Additionally, a graphical user interface (GUI) was implemented for the deconvolution software.

Discrete Fourier Transform Filtering

The goal of the algorithm is to remove residual error (i.e., noise) from the input data. In this implementation, it was assumed that unfiltered spectroscopic points are affected by high-frequency noise. The discrete Fourier transform (DFT) can be used to implement a low-pass filter to eliminate noise. The Fourier transform works by converting waveform data from the time domain into the frequency domain. This task is accomplished by breaking down the original time-based input into a series of sinusoidal terms, each having a unique magnitude, frequency, and phase. The process thus converts a difficult-to-describe waveform from the time domain to the frequency domain, creating

a more manageable series of sinusoidal functions that reproduce the original waveform exactly when added together [7].

Algorithm overview. The deconvolution algorithm with DFT filtering added functions that are very similar to those described in chapter 2. The main goal, however, remains the same: to decompose an input data set into a sum of functions that describe individual peaks and the residual error. Data points are read from an input file and preprocessed using interpolation to ensure they are evenly spaced. Discrete Fourier transform filtering occurs in the preprocessing phase following interpolation. It results in a smooth signal. After data preprocessing, peaks are detected, and deconvolution is carried out using function fitting. Finally, the results are displayed in a graphical user interface (GUI), as shown in Figure 11. The *x* axis shows the time, while the *y* axis shows the intensity.

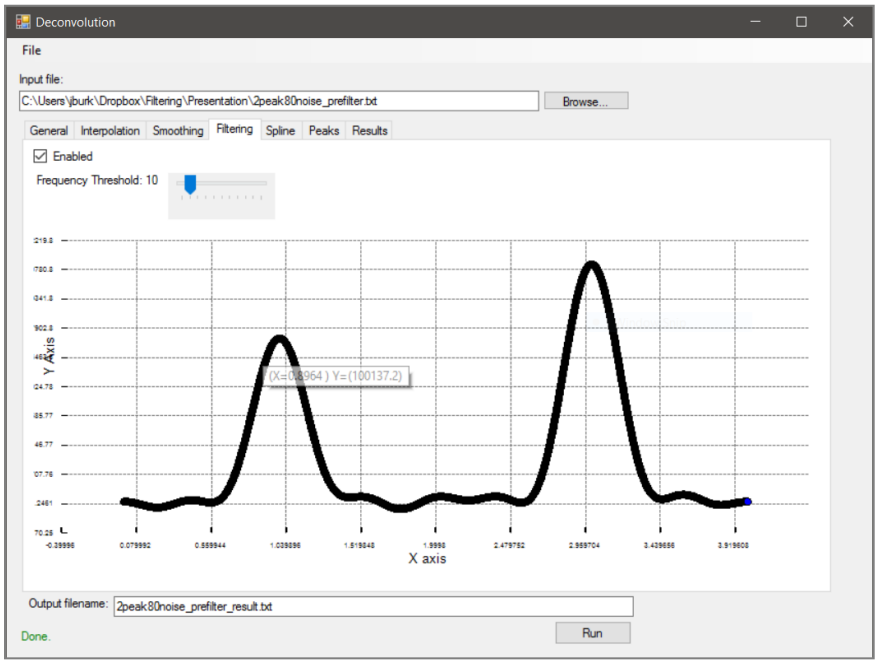

Figure 11. Result of DFT filtering displayed in GUI.

Due to the DFT's large computational requirements, a direct implementation of the DFT is not practical for real-time applications. However, a DFT can also still be implemented using algorithms known as Fast Fourier Transforms (FFTs) [35].

Noise removal – FFT limitations. The FFT is a computationally efficient method of calculating a Fourier transform. Its main advantage is speed, which results from decreasing the number of calculations needed to analyze a waveform [36]. However, restrictions may apply in most FFT algorithms [35]. The FFT is limited in application only to high-frequency noise, and it does not handle poorly resolved peaks well. Additionally, since the FFT generates a power spectrum based on a 2^{nth} power data point section of waveform (e.g., 512, 1024, 2048, etc.), the number of points in the power spectrum may be less than originally intended. A solution to this involves the user defining a precise range over which the Fourier transform will be calculated, circumventing the 2^{nth} power limitation. This method is called the discrete Fourier transform (DFT) and allows the evaluation of a waveform containing any number of points, providing more flexibility than the fixed-length FFT [36].

DFT definition. The DFT is defined by J. O. Smith [37], as shown in Equation (24):

$$X(\omega_k) \triangleq \sum_{n=0}^{N-1} x(t_n) e^{-j\omega_k t_n}, k = 0, 1, 2, \ldots, N-1, \tag{24}$$

where

$x(t_n)$ ≜ input signal *amplitude* (real or complex) at t_n (sec),

t_n ≜ nT = nth sampling instant (sec), n an integer ≥ 0,

T ≜ sampling interval (sec),

$X(\omega_k) \triangleq$ *spectrum* of *x* (complex valued), at frequency ω_k,

$\omega_k \triangleq k\Omega$ = *k*th frequency sample (radians per second),

$\Omega \triangleq \frac{2\pi}{NT}$ = radian-frequency sampling interval (rad/sec), and

N = number of time samples = no. frequency samples (integer).

How DFT filtering works. The Fourier transform takes real-valued data in the time domain and transforms them into complex-valued points in the frequency domain. Points whose frequencies are above a certain threshold, defined by the user, are eliminated. Using the DFT results in smooth, noise-free data. Figure 12 shows the experimental data before and after DFT filtering. The noise is up to 30% of the original signal's intensity. The graph with noise has much more variance, and, while it is still clear where the maxima of the peaks approximately are, it is impossible to tell where the peaks begin and end. The algorithm clearly makes the peaks much easier to analyze, thus allowing for easy numerical integration. As seen in Figure 13, DFT filtering can be useful even in an extreme case. Filtering makes it possible to determine the start and endpoints of the peaks after removing the excessive amount of noise. The threshold is the percentage of the highest frequency to cut off at.

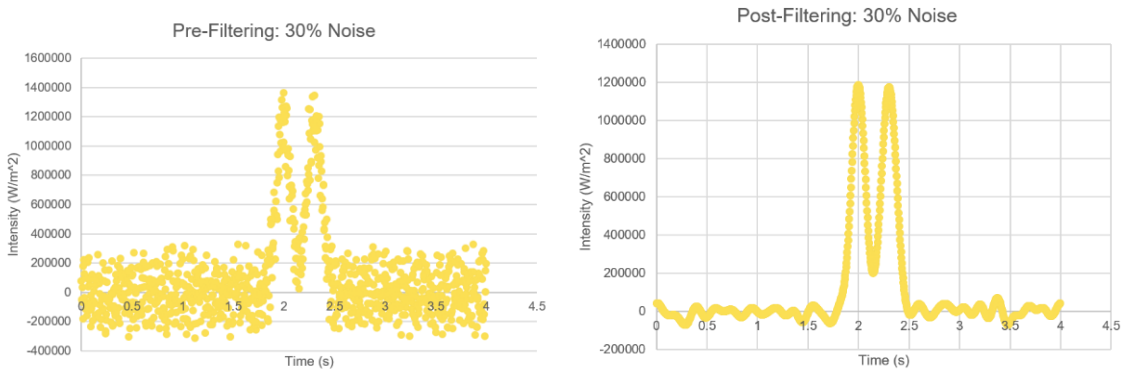

Figure 12. How DFT works, before filtering and after filtering (30% threshold).

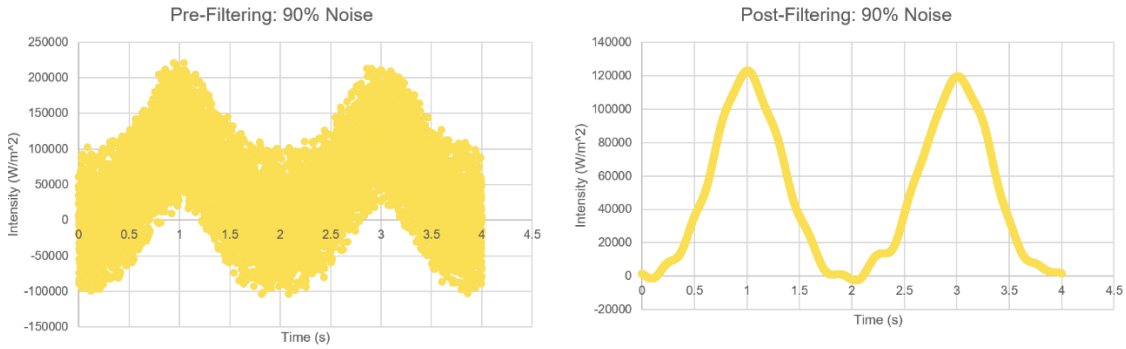

Figure 13. Filtering in extreme case (90% noise), before and after filtering (20% threshold).

Observations

There is a resolution requirement while running the deconvolution algorithm with DFT filtering. Peaks must be well enough resolved to be uniquely identifiable, otherwise the error in the calculated parameters increases. However, the algorithm is able to handle both complex and highly noisy data, as the number of peaks does not have an effect on the error, and the algorithm can still find peaks with up to 100% noise. In both cases, the only requirement is that peaks are sufficiently well resolved.

Deconvolution Algorithm Test Results

Tests were conducted to determine the impact of decreased peak spacing and of noise. Two sets of test data were examined: Test 1 describes the result of increasing the width of peaks, while Test 2 gives the result of moving peak centers closer together.

Impact of decreased peak spacing: Test 1. Figure 14 shows the impact of decreased peak spacing. Figure 14A has relatively large peak spacing, where the peaks are easily visually distinguished from each other, while Figure 14D has almost no spacing between the peaks, making them difficult to visually distinguish.

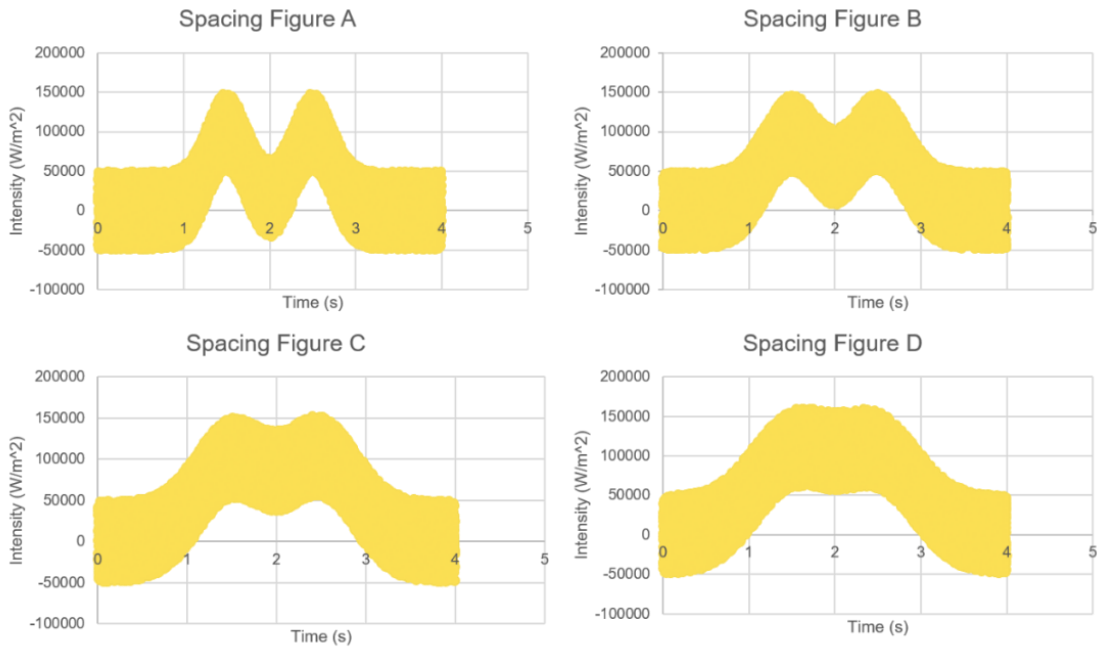

Figure 14. Test 1 results: Impact of decreased peak spacing.

Tables 2-3 show the results of Test 1. In the tables, the data for input file *2peak50noise_spacing_0.txt* correspond to the result of increasing width of peaks as shown in Figure 14A. The data for input file *2peak50noise_spacing_3.txt* correspond to

the result of increasing width of peaks as shown in Figure 14D. Errors in height, width, and position were calculated by dividing the absolute value of the difference between the calculated and actual parameters by the actual parameter. As the tables show, the overall error in the parameters height, width, and position increased as the spacing between the peaks decreased. For instance, the errors in position, width, and height started out in the ranges of 0% to 3%, 1.5% to 35%, and 0% to 5%, respectively, and ended in the ranges of 4% to 30%, 15% to 100%, and 5% to 45%, respectively. The error in width is especially pronounced in both cases, reaching percentages above 50% both times. Additionally, the errors seem to be smaller overall when lowering the filtering threshold**,** as indicated by comparing the starting and ending errors of the first test with those in the second test.

Table 2

Test 1 Results: Impact of Decreased Peak Spacing, Filtering With 20% Threshold

Filename	**Modeled Position**	**Calculated Position**	**Position Error (%)**	**Modeled Width**	**Calculated Width**	**Width Error (%)**	**Modeled Height**	**Calculated Height**	**Height Error (%)**
2peak50noise_spacing_0.txt	1.500	1.500	0.013	0.100	0.098	1.796	99706.337	100621.813	0.918
2peak50noise_spacing_0.txt	2.500	2.492	0.309	0.100	0.105	5.445	99643.775	99366.099	0.279
2peak50noise_spacing_1.txt	1.500	1.544	2.908	0.200	0.227	13.345	96658.805	101029.293	4.522
2peak50noise_ spacing_1.txt	2.500	2.434	2.650	0.200	0.270	34.834	98989.501	97507.638	1.497
2peak50noise_spacing_2.txt	1.500	1.521	1.402	0.300	0.308	2.577	97551.171	102809.133	5.390
2peak50noise_spacing_2.txt	2.500	1.866	25.374	0.300	0.018	94.156	99875.246	120085.092	20.235

Table 2 (continued)

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
2peak50noise_spacing_2.txt	2.500	2.347	6.111	0.300	0.473	57.685	99875.246	105386.074	5.518
2peak50noise_spacing_3.txt	1.500	1.665	10.995	0.400	0.563	40.846	99935.842	116400.868	16.476
2peak50noise_spacing_3.txt	2.500	2.138	14.472	0.400	0.017	95.769	99060.531	140451.345	41.783
2peak50noise_spacing_3.txt	2.500	2.397	4.131	0.400	0.470	17.522	99060.531	112331.844	13.397

Table 3

Test 1 Results: Impact of Decreased Peak Spacing, Filtering With 10% Threshold

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
2peak50noise_spacing_0.txt	1.500	1.501	0.074	0.100	0.102	1.904	99706.337	99580.635	0.126
2peak50noise_spacing_0.txt	2.500	2.498	0.076	0.100	0.098	1.558	99643.775	101480.000	1.843
2peak50noise_spacing_1.txt	1.500	1.546	3.070	0.200	0.231	15.612	96658.805	100723.327	4.205
2peak50noise_spacing_1.txt	2.500	2.435	2.591	0.200	0.257	28.476	98989.501	99021.184	0.032
2peak50noise_spacing_2.txt	1.500	1.606	7.079	0.300	0.389	29.610	97551.171	106255.038	8.922
2peak50noise_spacing_2.txt	2.500	2.352	5.907	0.300	0.460	53.446	99875.246	105741.414	5.873
2peak50noise_spacing_3.txt	1.500	1.684	12.268	0.400	0.584	46.087	99935.842	117534.076	17.610
2peak50noise_spacing_3.txt	2.500	2.576	3.043	0.400	0.260	35.021	99060.531	126613.507	27.814

Impact of decreased peak spacing: Test 2. Figure 15 shows the impact of decreased peak spacing, with limited resolution. Figure 15A shows extremely well resolved peaks, while Figures 15C and 15D show peaks with very poor resolution.

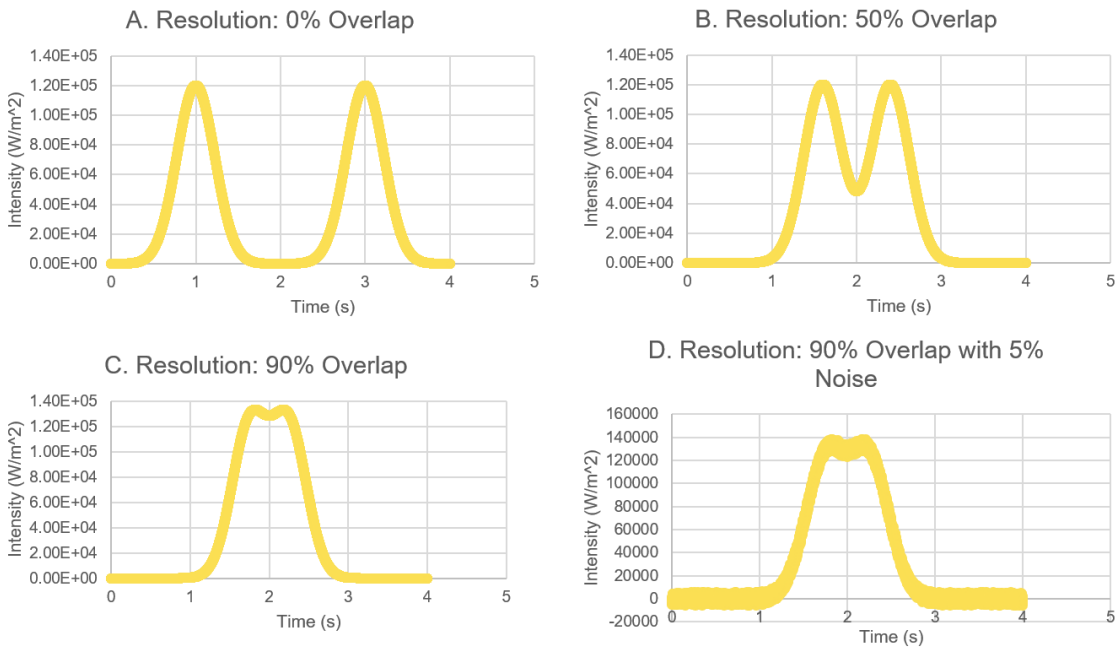

Figure 15. Test 2 results: Limitation of resolution.

Tables 4 and 5 show the results of running the algorithm on the peaks shown in Figure 15 (Test 2). Similar to Test 1, errors seem to have a negative correlation with peak spacing. Increasing the width caused an especially pronounced error, which reached up to 60%. A difference in Test 2, as compared to Test 1, is that the error values in height and width were very similar. Specifically, the errors were between the 10% threshold and the 20% threshold case (ranges of 0% to 30% and 0% and 35%, respectively).

Table 4

Test 2 Results: Limitation of Resolution, Filtering With 20% Threshold

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
spacing2_test0.txt	1.000	1.000	0.000	0.100	0.100	0.000	120000.000	120000.226	0.000
spacing2_test0.txt	3.000	3.000	0.000	0.100	0.100	0.000	120000.000	120000.674	0.001
spacing2_test1.txt	1.600	1.617	1.080	0.100	0.110	9.988	120000.000	120619.369	0.516
spacing2_test1.txt	2.400	2.406	0.255	0.100	0.093	6.851	120000.000	128143.800	6.786
spacing2_test2.txt	1.750	1.830	4.573	0.100	0.132	32.082	120000.000	140006.236	16.672
spacing2_test2.txt	2.250	2.278	1.258	0.100	0.073	27.118	120000.000	151459.486	26.216
spacing2_test3.txt	1.750	1.829	4.508	0.100	0.132	32.135	120000.000	139650.002	16.375
spacing2_test3.txt	2.250	2.273	1.040	0.100	0.072	28.482	120000.000	153813.320	28.178

Table 5

Test 2 Results: Limitation of Resolution, Filtering With 10% Threshold

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
spacing2_test0.txt	1.000	1.000	0.000	0.100	0.100	0.000	120000.000	120000.226	0.000
spacing2_test0.txt	3.000	3.000	0.000	0.100	0.100	0.000	120000.000	120000.674	0.001
spacing2_test1.txt	1.600	1.617	1.080	0.100	0.110	9.988	120000.000	120619.369	0.516
spacing2_test1.txt	2.400	2.406	0.255	0.100	0.093	6.851	120000.000	128143.800	6.786
spacing2_test2.txt	1.750	1.830	4.573	0.100	0.132	32.082	120000.000	140006.236	16.672

Table 5 (continued)

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
spacing2_test2.txt	2.250	2.278	1.258	0.100	0.073	27.118	120000.000	151459.486	26.216
spacing2_test3.txt	1.750	1.830	4.559	0.100	0.133	32.698	120000.000	139745.361	16.454
spacing2_test3.txt	2.250	2.133	5.222	0.100	0.158	58.411	120000.000	140067.457	16.723

Impact of noise. Tables 6 and 7 show the result of applying the algorithm to test the impact of increasing noise. The tables show that increasing noise does not seem to have much effect on error, with a very weak positive correlation in each case. Overall, the noise caused the errors, ranging from 0% to 1% for position, 0% to 12% for width, and 0% to 6% for height in each case.

Table 6

Test Results: Impact of Noise, Filtering With 20% Threshold

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
2peak62.txt	1.500	1.501	0.091	0.100	0.097	3.208	102541.193	104395.329	1.808
2peak62.txt	2.500	2.501	0.021	0.100	0.102	1.688	99456.923	101666.985	2.222
2peak75.txt	1.500	1.504	0.244	0.100	0.099	0.541	101798.138	104789.288	2.938
2peak75.txt	2.500	2.502	0.092	0.100	0.097	3.233	97986.481	98239.387	0.258
2peak87.txt	1.500	1.495	0.307	0.100	0.100	0.261	103804.031	104898.812	1.055

Table 6 (continued)

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
2peak87.txt	2.500	2.498	0.087	0.100	0.088	11.699	101225.142	106735.985	5.444
2peak100.txt	1.500	1.491	0.569	0.100	0.104	3.544	102627.487	98137.866	4.375
2peak100.txt	2.500	2.486	0.554	0.100	0.104	3.822	100477.727	100414.466	0.063

Table 7

Test Results: Impact of Noise, Filtering With 10% Threshold

Filename	Modeled Position	Calculated Position	Position Error (%)	Modeled Width	Calculated Width	Width Error (%)	Modeled Height	Calculated Height	Height Error (%)
2peak62.txt	1.500	1.502	0.118	0.100	0.099	1.469	102541.193	103884.793	1.310
2peak62.txt	2.500	2.502	0.089	0.100	0.099	0.795	99456.923	102200.715	2.759
2peak75.txt	1.500	1.503	0.195	0.100	0.104	3.890	101798.138	103385.275	1.559
2peak75.txt	2.500	2.506	0.237	0.100	0.093	7.241	97986.481	99413.895	1.457
2peak87.txt	1.500	1.496	0.270	0.100	0.098	2.117	103804.031	105155.040	1.301
2peak87.txt	2.500	2.493	0.280	0.100	0.105	5.302	101225.142	100731.108	0.488
2peak100.txt	1.500	1.489	0.717	0.100	0.107	6.797	102627.487	96766.143	5.711
2peak100.txt	2.500	2.490	0.382	0.100	0.097	2.545	100477.727	103076.285	2.586

Conclusion

The algorithm described above is useful for most applications requiring deconvolution. However, it still has some limitations. Peaks that are not visually distinct will not be found. There may also be some false positives, although these can be alleviated by decreasing the noise threshold. There are also potential improvements to be made to the algorithm, namely, applying a more complex filter that produces fewer artifacts than the simple low-pass filter, and using curve fitting for types of functions other than Gaussian.

Chapter 4

Wavelet Deconvolution

In a previous study, Gaussian, Lorentzian, and Voigt functions were used to accurately model chromatographic peaks. Preprocessing steps included smoothing and interpolation. An adaptive polynomial formula was used to create equal spacing between points. The Fourier transform was originally used to remove high-frequency noise, but the application of the Fourier transform with low-pass filtering is limited only to high-frequency noise and is not effective in handling poorly resolved peaks. An alternative to the Fourier transform is the continuous wavelet transform (CWT), which can construct a time-frequency representation of a signal that offers very good time and frequency localization.

The Wavelet Transform

The problem with the Fourier transform is that it gives the spectral content of the signal but provides no indication of the time at which spectral components appear [38]. The Short Time Fourier Transform (STFT) was then developed as a solution. This method shows the times at which certain frequencies are active in the signal. While windowing a signal, a different function is used to select a subset of the signal, and then the Fourier transform is applied. The window then shifts to different portions of the signal, where more Fourier transforms are calculated until the entire analysis is complete. This technique provides time localization of a signal's frequencies [39]. However, the STFT has the disadvantage of having a time versus frequency resolution trade-off. Narrow windows provide good time resolution but bad frequency resolution, while wide windows provide good frequency resolution but bad time resolution [38]. Therefore, an

alternative method to the Fourier transform is necessary. One alternative is wavelet analysis, which removes the need for window widths entirely by computing a transform over *all* width scales [39]. The process of wavelet analysis involves shifting a wavelet with a certain scale across the signal. For the process to be useful, multiple wavelets, each with different scales, need to be employed. Using different scales for the wavelets allows information to be gained about both the signal's times and frequencies [39].

Peak Detection with the CWT

Du et al. [40] describe a method that uses the CWT to detect peaks in spectroscopic data. Their method involves identifying *ridge lines* in a matrix computed from the CWT and filtering these ridge lines according to a minimal signal-to-noise ratio and minimal ridge line length. Ridge lines are lines that link the local maxima for the CWT coefficients at each scale. The matrix contains coefficients reflecting the pattern matching between the input signal s and the wavelet function $\psi_{a,b}(t)$, where higher coefficients indicate a better match [40].

Algorithm overview. The wavelet transform creates a localized analysis of the input signal. A high-level overview of the CWT algorithm adapted from [40] can be expressed as follows:

1. Compute the $N \times M$ CWT matrix of the input, where N is the number of scales to use and M is the length of the input spectrum
2. Identify ridge lines
3. Filter ridge lines to identify peaks

CWT definition. The CWT is defined in [41] as shown by Equation (25):

$$C(a,b) = \int_{-\infty}^{\infty} s(t)\psi_{a,b}(t)\mathrm{d}t, \psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \quad (25)$$

where $s(t)$ is the signal, a is the scale, $\psi(t)$ is the mother wavelet function, and b is the translation. For ψ, the Marr wavelet was chosen [41]. The CWT is a linear transformation, and it is covariant under dilations [42]:

$$f(x) \to f(mx), \quad W_\psi s_a(b) \to m^{-1/2} W_\psi s_{ma}(mb). \quad (26)$$

Ridge line identification. After the CWT matrix is computed, the algorithm initializes ridge lines based on local maxima found in the $\boldsymbol{N}$th row of the CWT coefficient matrix, which corresponds to the row with the largest scale. Each ridge line is assigned a gap number with an initial value of 0. The gap number is a measure used to identify which ridge lines are still to be searched by the algorithm. The algorithm then iterates over ridge lines with gap numbers less than a given threshold, searching for the nearest maximum point at the next adjacent scale. If the maximum point is less than the sliding window size for the current scale level, the ridge line's gap number is set to 0, and otherwise it is increased by 1. After each iteration, ridge lines with a gap number higher than the threshold are saved and removed from the list of ridge lines to search. New ridge lines are initialized for maxima not linked to upper level points. The previous steps are repeated until row $\boldsymbol{n = 1}$ (the row with the smallest scale) is reached in the CWT matrix [40].

Ridge line filtering. Ridge line filtering occurs based on three factors [40]:

1. The scale of the ridge line at the maximum amplitude should be within a certain range.

2. The signal-to-noise (S/N) ratio should be larger than a given threshold.
3. Ridge lines should be longer than a given threshold.

The signal of a peak is defined as the maximum CWT coefficient for a ridge line within a given scale range. Noise for a peak is defined as the 95-percent quantile of the absolute CWT coefficient values ($a = 1$) within a window surrounding the peak. The SNR is thus defined as the ratio of the peak's estimated signal strength and the peak's local noise level [40]. After filtering is performed, the CWT provides the location of peaks and their heights. This information can be supplied into the next steps of the algorithm to find more detailed information about the peaks, including width, location of endpoints, and so on.

Testing

The CWT algorithm was implemented in C# with Microsoft .NET Framework 4.6.1. Testing of the algorithm involved testing the algorithm's accuracy on modeled data for which the Gaussian parameters were already known and measuring the runtime of the algorithm on several data sets with differing numbers of data points.

Testing on modeled data. The input data set is a series of points generated from a sum of two Gaussian functions that slightly overlap, as shown in Figure 16. The parameters used to generate the first peak are height = 110,000, width = 0.70, and center = 2.0. The parameters used to generate the second peak are height = 120,000, width = 0.70, and center = 2.3.

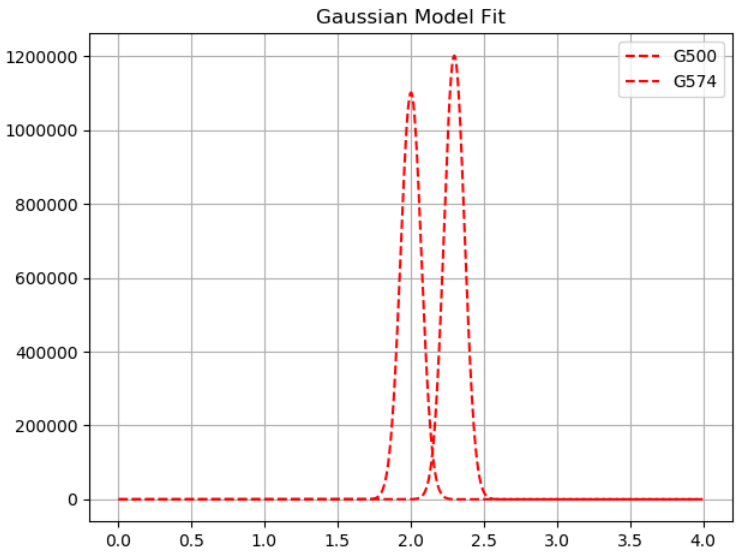

Figure 16. Modeled sum of Gaussian curves for CWT testing.

The CWT was used to calculate the positions of the peaks, which are shown with vertical dashed lines in Figure 17.

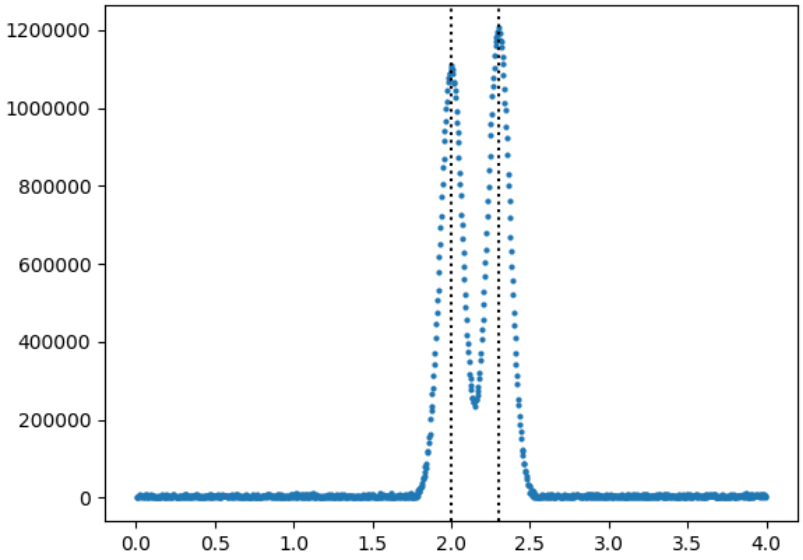

Figure 17. Peak position results from CWT.

The positions found by the CWT were used along with initial Gaussian parameter estimates to generate function fits with the package *lmfit* [43]. The results of the fit along with residuals are shown in Figure 18.

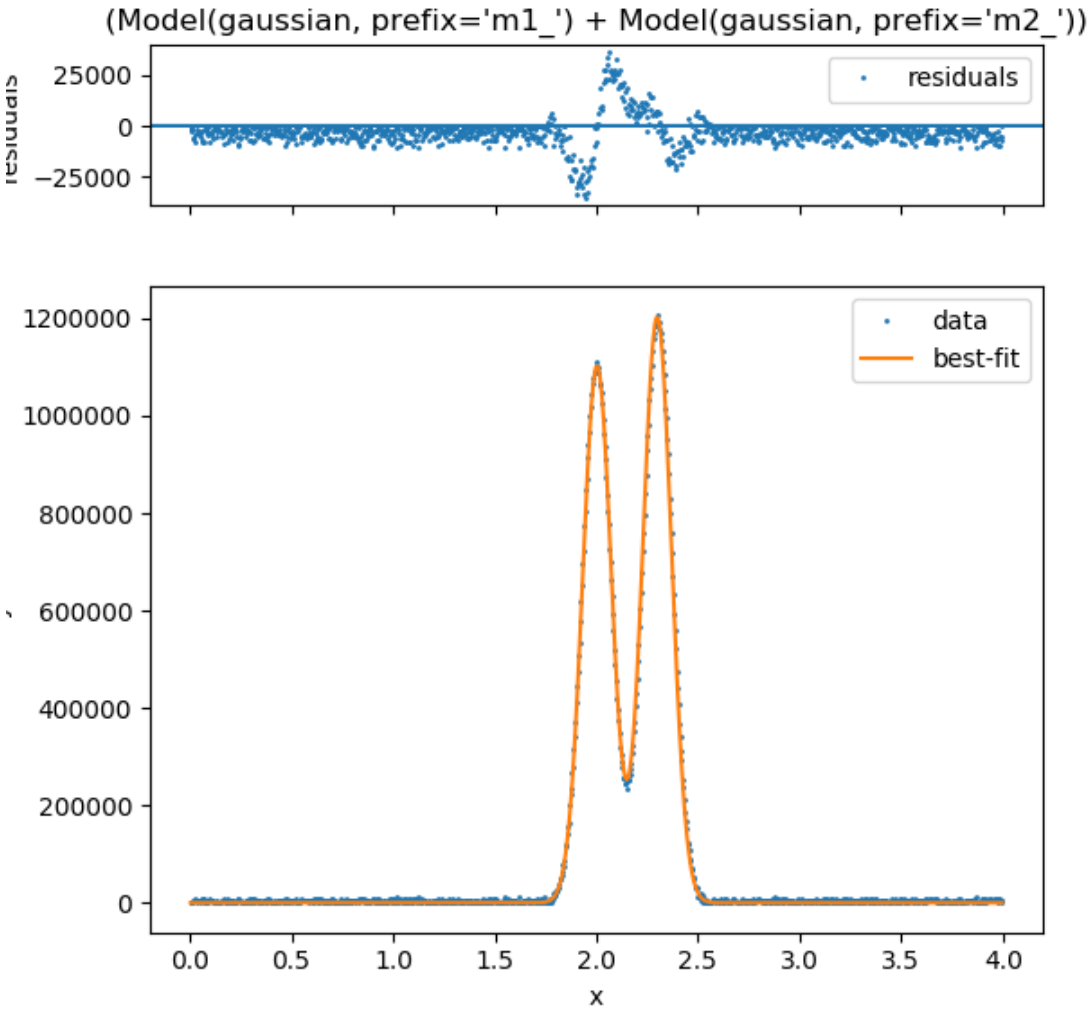

Figure 18. Best fit and residuals resulting from the least-squares minimization procedure.

The parameters found by the lmfit package are shown in Table 8. The table shows that the CWT is able to find positions of peaks accurately, and it can allow for accurate fitting with least-squares minimization or another appropriate curve fitting method.

Table 8

Results of Least-Squares Minimization Fit

Name	**Value**
σ_1	0.070589
center_1	2.003
height_1	1102213.07
σ_2	0.07028453
center_2	2.299
height_2	1202303.29
χ^2	6.4275×10^{10}

Performance testing. The algorithm was run on a Windows 10 desktop with an Intel Core i7-7700K 4.20GHz CPU and 32 gigabytes of RAM. Results were collected with the dotTrace performance profiler. Test results are shown in Table 9. For each data set, the algorithm ran on the original data set as well as the 1000-point set resulting from interpolation.

Table 9

Runtimes for the CWT Algorithm

Filename	Original Number of Points	Runtime for Original Number of Points (ms)	Runtime for n = 1000 Points (ms)
Hydrophilic_Double_5_185.0957.txt	54	32	6741
Hydrophilic_Double_65_215.0148.txt	77	5.6	8268
Hydrophilic_Triple_121_242.9807.txt	103	10	8174
Lipophilic_Double_865_524.3573.txt	45	1.1	8523
Lipophilic_Single_827_509.txt	22	17	7973
Lipophilic_Triple_253_288.1739.txt	522	1148	8446
Mixed_Single_1308_915.6949.txt	18	0.7	8284
Mixed_Double_985_585.3342.txt	43	1.2	8105
Mixed_Triple_1315_933.3913.txt	238	118	8174

Conclusion

The algorithm is very fast for the original number of points for each data set, but performance could be improved for the interpolated data sets. One potential improvement is to use an FFT for the convolution required by the calculation of the CWT matrix instead of a direct implementation of convolution. The bulk of the work of the

algorithm is done in the convolution step, and using an FFT can improve performance from $O(n^2)$ ro $O(n \log n)$.

Chapter 5

The Application of Deconvolution for Analysis of High-Resolution LC-MS Data

This chapter describes the applicability of the CWT algorithm to automated analysis of high-resolution LC-MS data collected in metabolomics experiments for rat plasma samples. Experimental LC-MS data used in this chapter were acquired on a Thermo Fisher Scientific Open Accela 1250 UHPLC system coupled with an Orbitrap mass spectrometer as described in Hnatyshyn and Shipkova [2]. Raw LC-MS data were preprocessed and converted to ASCII files. Each sample in the experiment has a corresponding folder that contains a collection of all detected signals. Each file stored in the sample folder represents an extracted ion chromatogram at a specific mass-to-charge ratio. Each ion chromatogram was extracted within a 10 ppm window of the selected mass-to-charge ratio. The collection of all extracted chromatograms represents a chemical makeup of a sample, where each extracted ion chromatogram is a measure of all detected isobaric chemicals in the sample makeup. Peaks in an extracted ion chromatogram represent a quantitative measurement of the contribution of the corresponding chemical in the sample composition [2]. Changing peak shapes, noise levels, and convolution states reflect the physio-chemical states of interactions of mobile and stationary phases of chromatographic separation throughout the duration of an LC-MS experiment [3].

An extracted ion chromatogram can be modeled simply as a sum of peaks, where each peak can be approximated by a Gaussian function [44]. The CWT algorithm was used to automatically analyze each extracted ion chromatogram data file to detect all peaks and calculate the parameters of Gaussian functions that approximate them.

To summarize the performance tests and validate the CWT algorithm, all experimental data were classified into 27 different cases reflecting all possible variations of peak properties in the experiment according to their elution time, noise levels, and convolution state.

The classification procedure for the entire input extracted ion chromatogram is based on the following differences: (1) presence of background noise, (2) elution time of the most intense peak of the extracted ion chromatogram, and (3) state of convolution. Signal-to-noise (S/N) ratio was calculated as a ratio between corresponding signals measured in a solvent blank and signals measured in plasma samples. Three arbitrary elution regions were established using elution profiles of compounds with known physical-chemical properties. The degree of peak convolution was measured by counting the number of peaks in an extracted ion chromatogram.

Classification of Extracted Ion Chromatograms

The most important determining factor for the quality of chromatographic data is the presence of noise. There are two major types of noise in a chromatographic experiment: (1) chemical noise and (2) random noise. A typical procedure to measure the presence of chemical noise in a chromatographic system is an experiment with a blank injection, which does not contain any sample but rather includes only the solvent used to dissolve the sample in the chromatographic experiment. A signal measured in the blank injection represents chemical noise (see Figure 19).

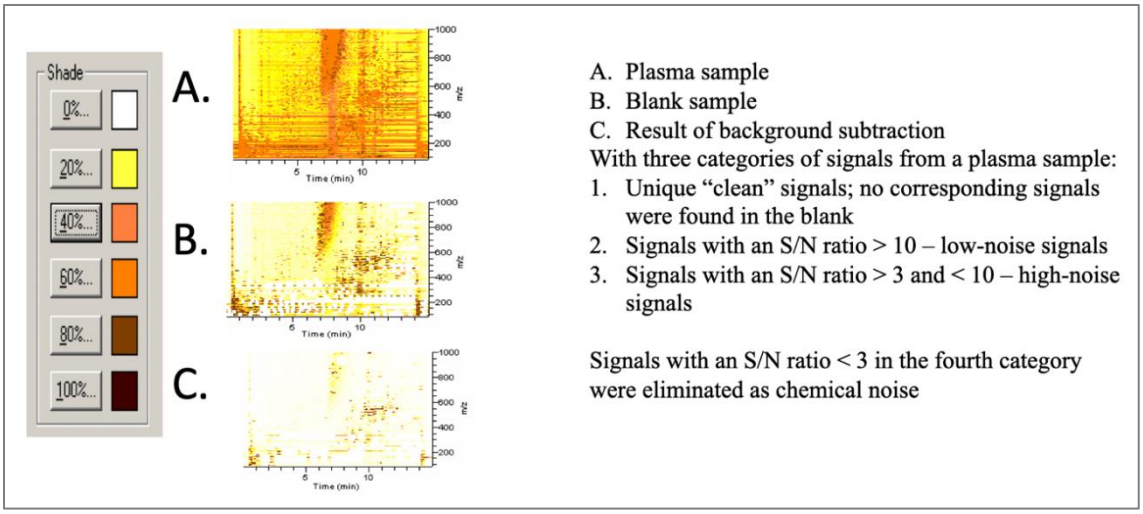

Figure 19. Comparison of rat plasma samples. Reproduced with permission from Dr. S. Hnatyshyn [2].

Classification by signal-to-noise (S/N) ratio. The comparison of correspondent signals matched by the mass-to-charge ratio in the blank injection and rat plasma sample allows the sorting of detected signals into four categories:

1. "Clean" unique to plasma samples
2. Low-noise signals with an S/N ratio greater than 10
3. High-noise signals with an S/N ratio between 3 and 10
4. Chemical noise signals with an S/N ratio less than 3 (signals in this category were removed from consideration)

Classification by physical-chemical properties. Physical-chemical properties of a substance define its behavior during a separation experiment in a chromatographic system. The behavior of a substance and its interactions inside the chromatographic system define elution time and shape of correspondent peaks on the extracted ion

chromatograms [44]. Substances can be classified into three categories based on the value of the elution time of the correspondent peak (see Figure 20):

1. Signals that correspond to substances with hydrophilic properties (retention time of 0-6 minutes)
2. Signals that correspond to substances with mixed hydrophilic/hydrophobic properties (retention time of 6-10 minutes)
3. Signals that correspond to substances with lipophilic properties (retention time of 10-16 minutes)

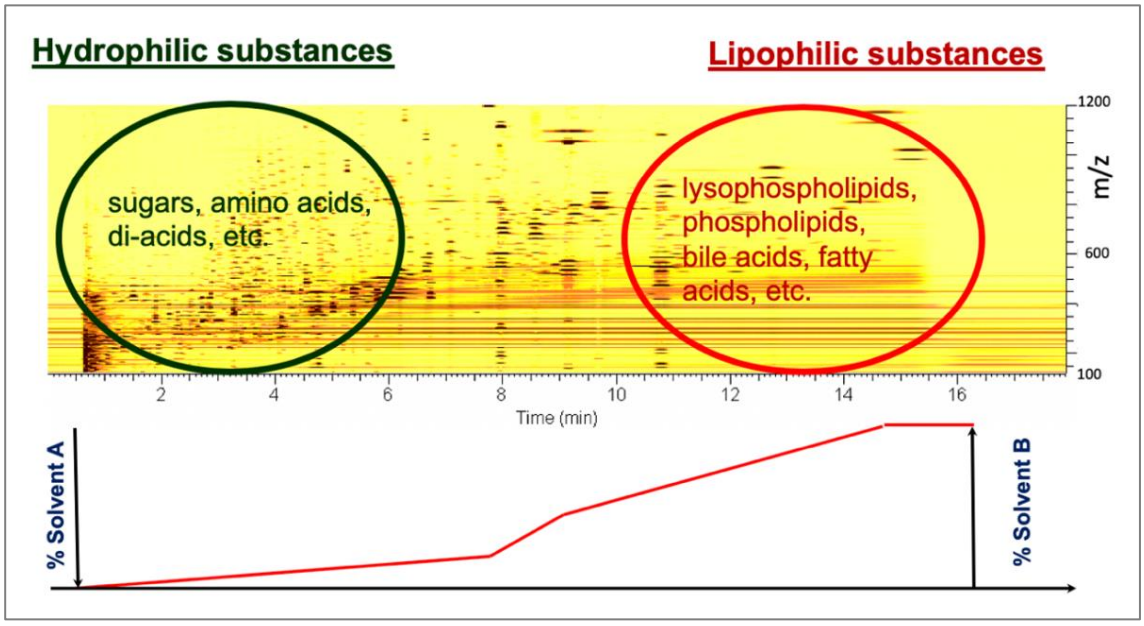

Figure 20. Substance properties over time. Reproduced with permission from Dr. S. Hnatyshyn [2].

Classification by degree of peak convolution. Finally, the categories based on peak elution time are further divided into categories determined by degree of peak convolution (see Figure 21):

1. Signals corresponding to chromatograms with a single peak
2. Signals corresponding to chromatograms with two overlapping peaks
3. Signals corresponding to chromatograms with three or more overlapping peaks

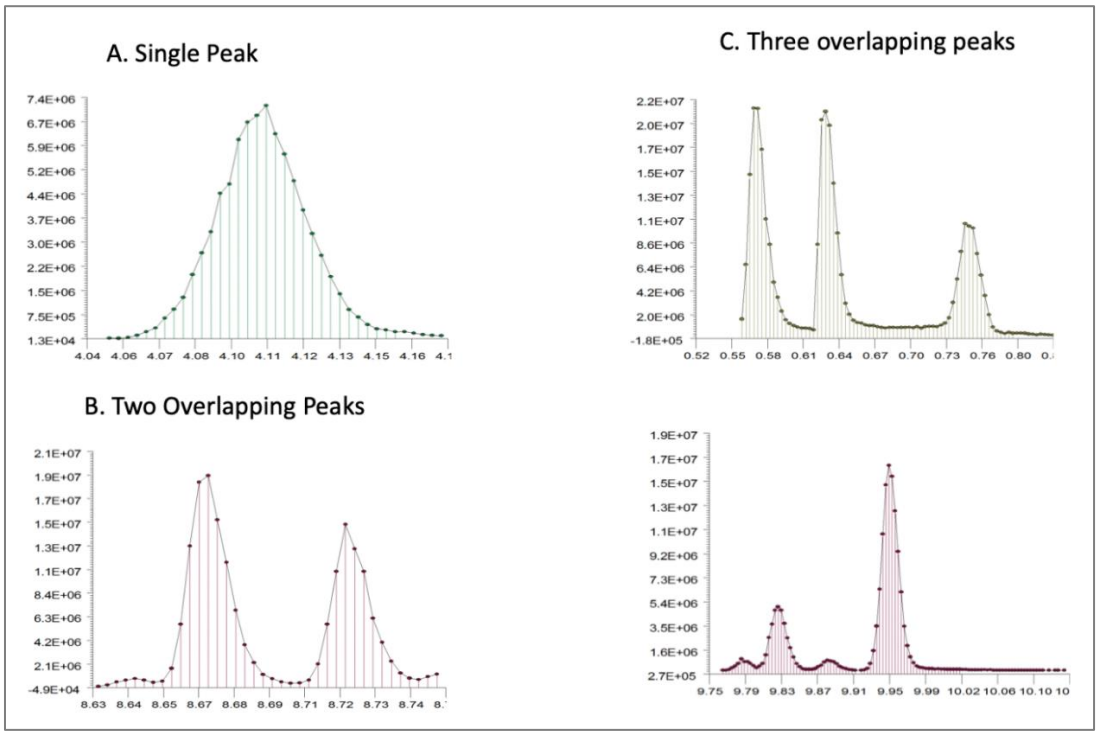

Figure 21. Degree of peak convolution.

Classification results. Table 10 shows the number of chromatograms in each category. For each of the 27 categories, a representative sample was chosen and used as input to the CWT deconvolution algorithm. Obtained results were compared with the results of manual convolution for the same traces.

Table 10

LC-MS Extracted Ion Chromatogram Categories

Noise Level	Hydrophilic			Mixed			Lipophilic		
	Single	**Two**	**Three or More**	**Single**	**Two**	**Three or More**	**Single**	**Two**	**Three or More**
Clean	333	29	9	257	10	14	515	97	86
Low	173	25	39	284	7	8	212	24	62
High	91	39	160	157	29	72	393	49	360

Discussion

The CWT was tested with a representative data set from each of the 27 categories. The results of the tests are shown in Table A1. The Data File column contains the file name for each data set. Each file was interpolated with a step of 0.0015 and had a peak width of 0.2. The Model Fit Statistics column shows the measurements of the model generated by lmfit. In the case where multiple models were generated, the ones with the lowest Bayesian and Akaike information criteria were chosen. The Model View column shows the resulting variables for each of the Gaussian peaks. In the model view, A_n represents the height of the nth peak, B_n represents the position of the nth peak, and C_n represents the width of the nth peak. Table A1 explains the model fit statistics. The figures corresponding to the models for each data file are shown in Appendix B.

Clean and hydrophilic samples. *CHS_5_185.0957* shows a sample with a single peak and no noise. The peak is close to being symmetric, and the model generated creates a Gaussian peak with minimal residual error compared to the original. *CHD_65_215*

shows a sample with two visually distinct peaks and a small bump at the right end of the data. The model contains one Gaussian peak for the larger of the distinct peaks, a high-width Gaussian peak for the smaller of the peaks, and a low-height Gaussian for the bump at the end of the data. *CHT_121_242* shows a sample with two visually distinct peaks at the left end of the data, and multiple smaller, less distinguishable peaks in the right half of the data. The model generated includes Gaussians for the two visually distinct peaks in the left half of the data and has five Gaussian peaks for the right half of the data (two pairs from peaks that were convoluted and one at the right end of the data).

Clean and mixed samples. *CMS_1308_915* shows a sample with a tall, thin peak at the beginning of the data and a wider, visually distinct peak in the first half of the data from the left. The model computed includes both of these peaks as well as two small peaks for the second half of the data. *CMD_985_585* shows a set of peaks of which each appears to consist of two or three overlapping peaks. The computed model finds Gaussians for each of the overlapping peaks, as well as wider peaks for the baseline. *CMT_1315_933* has multiple overlapping peaks, each of which is narrow and close together. The resulting model is a set of multiple thin Gaussian peaks. Some peaks from the original data were not considered significant and thus were excluded from the model.

Clean and lipophilic samples. *CLS_827_509* shows a single, symmetric peak that has no noise. The model computed accurately represents the peak. *CLD_865_524* shows two peaks with slight overlap that are each symmetric and that have a small amount of noise. The resulting model accurately finds two overlapping Gaussian peaks. *CLT_253_288* shows two main visually distinct peaks in the first half of the data from the

left and one small peak in the second half of the data. The larger of the two peaks has some tailing, and both are narrow. The resulting model accurately finds the three peaks.

Low-noise and hydrophilic samples. *LHS_49_230* shows a single peak that has a slight amount of noise and is slightly asymmetric. The model computed accurately represents the peak. *LHD_108_86* shows two slightly overlapping peaks with surrounding noise. The model computed finds two overlapping peaks as well as one distinct peak to the right in the data. *LHT_75_261* shows a single visually distinct peak with significantly more height than the remainder of the data along with two smaller peaks that overlap and are only slightly higher than the noise. The model finds the high peak as well as the two overlapping peaks.

Low-noise and mixed samples. *LMS_7_189* shows a single, slightly asymmetric peak with a low amount of noise. The model calculates the peak accurately. *LMD_151_297* shows a single, visually distinct peak in the middle data with two smaller peaks at the front and tail ends of the data. The model finds these peaks accurately. *LMT_163_300* shows a single, visually distinct peak at the front end of the data with several small, overlapping peaks at the end of the data. The model finds the visually distinct peak and four peaks for the tail end.

Low-noise and lipophilic samples. *LLS_89_272* shows a single, slightly asymmetric peak that is narrow and has low noise. There is a small bump of noise to the left of the peak. The model calculates a peak with lower height than the peak in the data. *LLD_81_263* shows two sets of peaks of which each is composed of two low-noise overlapping peaks. The model accurately identifies the left pair of overlapping peaks, but the position it calculates for the smaller peak of the right pair of peaks is to the right of

the actual position of the peak. *LLT_57_238* has six peaks of varying heights that are each narrow and are all visually distinct. The model marks five of the six peaks as being legitimate peaks, and each of these has its position, height, and width calculated correctly.

High-noise and hydrophilic samples. *HHS_507_306* shows a peak with peaklike oscillations to its right. The model finds two peaks: one to represent the peak itself, and one to represent the baseline noise of the data. *HHD_507_306* is similar to the previous sample, which has a single visually distinct peak with peaklike oscillations to its right. The model calculates four peaks to account for the given peak and its noise. *HHT_23_185* has three narrow peaks that are each surrounded by high-frequency noise. The model finds the three peaks as well as several groups of small peaks that are visually indiscernible from the noise in the data.

High-noise and mixed samples. *HMS_125_211* shows a single symmetric peak and the left half of a peak at the tail end of the data. The model finds the visually distinct peak and two peaks for the tail end of the data. *HMD_152_217* shows two visually distinct peaks of which each has noisy oscillations to its right. The model successfully calculates two peaks that correspond with those in the data. *HMT_85_199* appears to have an asymmetric peak that spreads throughout the data, with one narrow, high peak in the middle of the data. The model actually finds 13 narrow peaks to represent the data.

High-noise and lipophilic samples. *HLS_609_343* shows a single, slightly asymmetric peak that has a low baseline to its right. The model calculates two peaks: one narrow peak representing the peak in the data, and one very wide peak that represents the baseline. *HLD_159_219* has several noisy peaks that are slightly asymmetric and

overlapping. The model finds one wide, high peak and three smaller peaks. *HLT_83_199* shows several noisy peaks that appear to be symmetric and slightly overlapping. The model calculates several overlapping peaks that mix to produce a representation of the data.

Conclusion

The CWT provides an effective automated procedure for the analysis of extracted ion chromatograms. The CWT is robust to different numbers of peaks and levels of noise in input data. Additionally, a mixture of symmetric Gaussian functions provides an adequate model for chromatographic data.

Chapter 6

Conclusions

The goal of this thesis was to develop an algorithm that provides robust peak deconvolution with completely automated output without the need for manual verification of results. The deconvolution algorithm presented consists of preprocessing steps, noise removal, peak detection, and function fitting. For noise removal, both a Fourier Transform and Continuous Wavelet Transform method of noise removal were examined. Testing of the algorithm involved running the automated algorithm on data divided into distinct categories based on amount of noise and peak types.

The presence of noise in images causes deconvolution to be a difficult problem, the solution to which is to identify and separate overlapping peaks. The research presented in this thesis began with prototyping an algorithm for processing modeled data composed of custom *x* and *y* values that form a mixture of Gaussian functions with "known" initial parameters. These data were used to develop and test the expectation-maximization (EM) portion of the algorithm for deconvolution. When tested on modeled data, the implementation of the EM algorithm quickly converged to the correct parameter values, providing an accurate estimation of individual function components in the mixture. However, when run on real data, the algorithm did not converge and could not accurately compute individual functions. A k-means clustering algorithm was implemented with the assumption that more accurate knowledge of point membership would lead to better estimates of peak parameter values. Testing of the prototype implementation revealed that the algorithm continued to work well with custom data but still failed to converge for real data input. It was found that the assumption that the initial

mixture is described by a Gaussian distribution of evenly distributed data points is incorrect, as spectroscopic data points are not guaranteed to be evenly distributed. Thus, the prototype could not correctly identify the number of peaks and initial values for individual function parameters. This showed the necessity of preprocessing the input data before running the EM algorithm.

To address the problem of noise, various algorithms, including the preprocessing steps of interpolation, smoothing, and spline calculation, were introduced. After noise continued to be present in the data, a 5-step polynomial smoothing function, the Savitzky-Golay filter, was added to increase the signal-to-noise ratio of the interpolation algorithm's output. The Savitzky-Golay smoothing algorithm resulted in a uniformly spread, low-noise set of discrete points. The remainder of the algorithm required a continuous function as input; thus, the *y* value for a given *x* value was calculated using two linearly independent cubic polynomial terms in the spline function. After data preprocessing, peaks were found by calculating the curve's derivative. Each peak, after being found, was fit into a Gaussian function. The function fitting procedure described in chapter 2 assumes Gaussian functions are symmetrical in nature. However, asymmetric Gaussian functions are more commonly found in experimental data. Asymmetry broadens the base of a peak and increases peak overlap, thereby resulting in more difficult measurement. The work presented in chapter 2 resulted in software that reads raw data and performs data preprocessing. After preprocessing, the software detected peaks and fit experimental data into a Gaussian approximation and then optimized Gaussian parameters with the EM algorithm. Stability testing revealed that the algorithm continued to falter in the presence of noise.

A major improvement to the algorithm was made with the addition of noise elimination through low-pass filtering with Fourier transforms. Tests were conducted with a low-pass filter implemented with a DFT to determine the impact of decreased peak spacing and of noise. Two sets of test data were examined: Test 1 describes the result of increasing the width of peaks, while Test 2 gives the result of moving peak centers closer together. The tests conducted showed that increased overlap in input data resulted in increased error in estimated parameter values after the DFT was applied. While the DFT is useful for most applications requiring deconvolution, there are still some limitations: Peaks that are not visually distinct will not be found. Potential improvements to the algorithm include (a) applying a more complex filter that produces fewer artifacts than the simple low-pass filter and (b) using curve fitting for types of functions other than Gaussian.

An alternative to the Fourier transform is the continuous wavelet transform (CWT). The main advantage of the wavelet transform as a method for time-frequency analysis is that it is able to perfectly reconstruct functions [45]. Testing of the algorithm involved testing the algorithm's accuracy on modeled data for which the Gaussian parameters were already known and measuring the runtime of the algorithm on several data sets with differing numbers of data points. The CWT was able to find positions of peaks accurately, and it can allow for accurate fitting with least-squares minimization or another appropriate method of curve fitting. The algorithm is very fast for the data sets without interpolation, but could have improved performance for interpolated data sets. The algorithm's performance could be improved by using an FFT implementation of convolution instead of a direct implementation.

To test the performance of the CWT algorithm, all input data were classified based on the following differences: (1) presence of background noise, (2) elution time on the extracted ion chromatogram, and (3) state of peak convolution. Data were classified by signal-to-noise (S/N) ratio, by physical-chemical properties, and by degree of peak convolution. Extracted ion chromatograms were sorted into 27 categories that reflect all possible combinations of classification differences. The CWT was tested with representative data sets from each of the 27 categories. The results, presented in the form of a sum of Gaussian function models, are shown in Appendices A and B. Appendix A shows the values of Gaussian function parameters and the details of the statistical evaluation of fits. Appendix B is a graphical representation of the experimental data, models, and residuals. Presented results for the CWT's application to experimental data illustrate that the algorithm is an effective method for estimating the locations of peaks in chromatographic data and that a sum of symmetric Gaussian curves is a reasonable model that approximates all types of extracted ion chromatograms.

Future improvements can include expanding the model-fitting capabilities of the CWT algorithm by utilizing different functions to describe chromatographic peaks (e.g, asymmetrical Gaussians; see chapter 2) and creating an unbiased model optimizer that will automatically select the most adequate model based on the values of the Akaike information criterion or Bayesian information criterion [46].

References

[1] E. Holmes, I. D. Wilson, and J. K. Nicholson, "Metabolic phenotyping in health and disease," *Cell*, vol. 134, pp. 714–717, 5 Sept. 2008.

[2] S. Hnatyshyn and P. Shipkova, "Automated and unbiased analysis of LC-MS metabolomic data," *Bioanalysis*, vol. 4, no. 5, pp. 541–554, 2012.

[3] F. E. Lytle and R. K. Julian, "Automatic processing of chromatograms in a high-throughput environment," *Clin. Chem.*, vol. 62, no. 1, pp. 144–153, 2016.

[4] H. P. Blok, J. D. de Lange, and J. W. Schotman, "A new peak search method for an automatic spectrum analysis program," *Nucl. Instrum. Methods*, vol. 128, no. 3, pp. 545–556, 1975.

[5] C. A. Smith, E. J. Want, G. O'Maille, R. Abagyan, and G. Siuzdak, "XCMS: Processing mass spectrometry data for metabolite profiling using nonlinear peak alignment, matching, and identification," *Anal. Chem.*, vol. 78, no. 3, pp. 779–787, 1 Feb. 2006.

[6] R. A. Scheltema, A. Jankevics, R. C. Jansen, M. A. Swertz, and R. Breitling, "PeakML/mzMatch: A file format, Java library, R library, and tool-chain for mass spectrometry data analysis," *Anal. Chem.*, vol. 83, no. 7, pp. 2786–2793, 2011.

[7] S-Q Zhang, X. Zhou, H. Wang, A. Suffredini, D. Gonzales, W-K Ching, M. K. Ng, and S. Wong, "Peak detection with chemical noise removal using short-time FFT for a kind of MALDI data," *The First International Symposium on Optimization and Systems Biology*, 2007, pp. 222–231.

[8] D. P. A. Kilgour et al., "Autopiquer–A robust and reliable peak detection algorithm for mass spectrometry," *J. Am. Soc. Mass Spectrom.*, vol. 28, no. 2, pp. 253–262, Feb. 2017.

[9] N. Dyson, "Errors in peak area measurement," in *Chromatic Integration Methods*, 2nd ed. Cambridge, UK: The Royal Society of Chemistry, 1998, ch. 2, pp. 35–56.

[10] ———, "Measurement and models," in *Chromatic Integration Methods*, 2nd ed. Cambridge, UK: The Royal Society of Chemistry, 1998, ch. 1, pp. 1–34.

[11] Š. Hatrík and J. Hrouzek, "The use of general exponential function for the deconvolution of fused chromatographic peaks," *Chem. Papers*, vol. 48, no. 6, pp. 376–380, 1994.

[12] T. O'Haver. (2018, July). *A pragmatic introduction to signal processing* [Online]. Available: https://terpconnect.umd.edu/~toh/spectrum/Integration.html

[13] M. G. Grotenhuis. (2006). *An overview of the maximum entropy method of image deconvolution* [Online]. Available: http://homepages.spa.umn.edu/~tjj/Grotenhuis/Michael_Grotenhuis_MEM_masters_paper_final3.pdf

[14] N. Agmon, Y. Alhassid, and R. D. Levine, "An algorithm for finding the distribution of maximal entropy," *J. Comput. Phys.*, vol. 30, pp. 250–258, 1979.

[15] N. Chamoli, S. Kukreja, and M. Semwal, "Survey and comparative analysis on entropy usage for several applications in computer vision," *Int. J. Comput. Appl.*, vol. 97, no. 16, 2014.

[16] A. Wiesel, Y. C. Eldar, and A. Beck, "Maximum likelihood estimation in linear models with a Gaussian model matrix," *IEEE Signal Process. Lett.*, vol. 13, no. 5, pp. 292–295, May 2006.

[17] R. A. Chylla, K. Hu, J. J. Ellinger, and J. L. Markley, "Deconvolution of two-dimensional NMR spectra by fast maximum likelihood reconstruction: Application to quantitative metabolomics," *Anal. Chem.*, vol. 83, pp. 4871–4880, 2011.

[18] C. Simon. (2012, Dec. 24). *The principle of maximum entropy* [Online]. Available: http://corysimon.github.io/articles/the-principle-of-maximum-entropy/

[19] J. L. Starck, E. Pantin, and F. Murtagh, "Deconvolution in astronomy: A review," *Publ. Astron. Soc. Pac.*, vol. 114, pp. 1051–1069, Oct. 2002.

[20] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. San Diego: California Technical Publishing, 1997.

[21] D. Gaffney and W. J. Burke, "Robust analysis of spectroscopic datasets using the maximum entropy principle," STEM poster, presented at Rowan University, Glassboro, NJ, Apr. 19, 2016.

[22] T. K. Moon. (1996, Nov.). "The expectation-maximization algorithm," *IEEE Signal Processing Mag.* [Online]. Available: https://www.eecs.yorku.ca/course_archive/2007-08/W/6328/Reading/EM_tutorial.pdf

[23] A. Ng. Part IX: *The EM algorithm*, lecture notes [Online]. Available: http://cs229.stanford.edu/notes/cs229-notes8.pdf

[24] P-N Tan, M. Steinbach, A. Karpatne, and V. Kumar, *Introduction to Data Mining*, 2nd ed. New York: Pearson, 2018.

[25] C. Piech. *K means* [Online]. Available: http://stanford.edu/~cpiech/cs221/handouts/kmeans.html

[26] C. Bauer, R. Cramer, and J. Schuchhardt, "Evaluation of peak-picking algorithms for protein mass spectrometry," in *Data Mining in Proteomics: From Standards to Applications*, M. Hamacher, M. Eisenacher, and C. Stephan, Eds. New York: Humana Press, 2011, pp. 341–352.

[27] J. N. Gregg. *Polynomial interpolation* [Online]. Available: http://www2.lawrence.edu/fast/GREGGJ/Math420/Section_3_1.pdf

[28] B. Archer and E. W. Weisstein. "Lagrange interpolating polynomial," *MathWorld–A Wolfram Web Resource* [Online]. Available: http://mathworld.wolfram.com/LagrangeInterpolatingPolynomial.html

[29] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, 1964.

[30] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. New York: Cambridge University Press, 1992.

[31] Nucleomatica. inmr.net software. *Peak-picking* [Online]. Available: http://www.inmr.net/Help3/ref/picking.html

[32] C. R. Nave. *Gaussian distribution function* [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/Math/gaufcn.html

[33] E. W. Weisstein. "Normal distribution," *MathWorld*–A Wolfram Web Resource [Online]. Available: http://mathworld.wolfram.com/NormalDistribution.html

[34] J. Jacquelin. (2009, Jan. 14). "Régressions et équations intégrales" [Online]. https://www.researchgate.net/.../14674814-Regressions-et-equations-integrales.pdf

[35] R. Matusiak. (2001, Aug.). *Implementing fast Fourier transform algorithms of real-valued sequences with the TMS320 DSP platform* [Online]. Available: http://www.ti.com/lit/an/spra291/spra291.pdf

[36] DATAQ Instruments, Inc. *FFT (fast Fourier transform) waveform analysis* [Online]. Available: https://www.dataq.com/resources/pdfs/article_pdfs/an11.pdf

[37] J. O. Smith, III., *Mathematics of the discrete Fourier transform (DFT): With audio applications*, 2nd ed. W3K Publishing, 2007.

[38] R. Polikar. *The wavelet tutorial* (2nd ed.) [Online]. Available: http://web.iitd.ac.in/~sumeet/WaveletTutorial.pdf

[39] D. Ozog. (2007, May 11). *Signal analysis* [Online]. Available: https://www.whitman.edu/Documents/Academics/Mathematics/ozogdm.pdf

[40] P. Du, W. A. Kibbe, and S. M. Lin, "Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching," *Bioinformatics*, vol. 22, no. 17, pp. 2059–2065, 2006.

[41] E. Lange, C. Gropl, K. Reinert, O. Kohlbacher, and A. Hildebrandt, "High-accuracy peak picking of proteomics data using wavelet techniques," *Pac. Symp. Biocomput.*, vol. 11, pp. 243–254, 2006.

[42] J. L. Starck and A. Bijaoui, "Filtering and deconvolution by the wavelet transform," *Signal Process*, vol. 35, pp. 195–211, 1994.

[43] M. Newville, T. Stensitzki, et al. (2018, Nov. 29). *Non-linear least-squares minimization and curve-fitting for Python* [Online]. Available: http://cars9.uchicago.edu/software/python/lmfit/lmfit.pdf

[44] J. V. Hinshaw, "Anatomy of a peak," *LCGC NORTH AMERICA*, vol. 22, no. 3, pp. 253-260, 2004.

[45] R. Büssow. (2007). "An algorithm for the continuous Morlet wavelet transform," *Institute of Fluid Mechanics and Engineering Acoustics*, p. 15 [Online]. Available: http://www.tu-berlin.de/fb6/ita

[46] K. P. Burnham and D. R. Anderson, *Model selection and multimodel inference: A practical information-theoretic approach*, 2nd ed. New York: Springer, 2002, p. 488.

Appendix A

Continuous Wavelet Transform (CWT) Test Categorical Results

Appendix Table A1 includes the data files and parameters, Gaussian model fit statistics, and Gaussian model view for the following CWT test categories:

Table A1

CWT Test Categorical Results

Data File	**Model Fit Statistics**	**Model View**
Clean + hydrophilic		
CHS_5_185.0957	Fitting method = least sq. Function evals = 65 Data points = 165 Variables = 2 Chi-square = 1.11 x 10^{13} Reduced chi-sq. = 6.80 x 10^{10} Akaike info crit = 4117.45 Bayesian info crit = 4124.66	C1: 0.016 B1: 4.052 A1: 291944.773
CHD_65_215	Fitting method = least sq. Function evals = 1516 Data points = 214 Variables = 6 Chi-square = 1.41 x 10^{12} Reduced chi-sq. = 6.80 x 10^{9} Akaike info crit = 4850.89 Bayesian info crit = 4871.08	C1: 0.177 B1: 0.685 A1: 49982.973 C2: 0.010 B2: 0.778 A1: 14044.863 C3: 0.010 B3: 0.943 A3: 789.664
CHT_121_242	Fitting method = least sq. Function evals = 23 Data points = 530 Variables = 14 Chi-square = 2.72 x 10^{12} Reduced chi-sq. = 5.26 x 10^{9} Akaike info crit = 11877.26 Bayesian info crit = 11937.08	C1: 0.010 B1: 0.671 A1: 4526.952 C2: 0.010 B2: 0.777 A2: 14942.912 C3: 0.010 B3: 0.957 A3: 2250.164 C4: 0.010 B4: 1.017 A4: 1987.361 C5: 0.010 B5: 1.169 A5: 1559.910 C6: 0.010 B6: 1.247 A6: 1739.057 C7: 0.010. B7: 1.439 B7: 1063.765

Table A1 (continued)

Data File	Model Fit Statistics	Model View
Clean + Mixed		
CMS_1308_915	Fitting method = least sq Function evals = 925 Data points = 14902 Variables = 8 Chi-square = 4.14×10^{14} Reduced chi-sq. = 2.78×10^{10} Akaike info crit = 358380.27 Bayesian info crit = 358441.14	C1: 0.011 B1: 7.719 A1: 66051.343 C2: 0.050 B2: 8.195 A2: 94222.523 C3: 0.050 B3: 8.767 A3: 30271.728 C4: 0.050 B4: 9.653 A4: 31836.843
CMD_985_585	Fitting method = least sq Function evals = 62012 Data points = 2773 Variables = 30 Chi-square = 1.64×10^{12} Reduced chi-sq. = 5.97×10^{8} Akaike info crit = 56063.42 Bayesian info crit = 56241.25	C1: 0.005 B1: 9.106 A1: 4459.497 C2: 0.008 B2: 9.115 A2: 3528.697 C3: 0.005 B3: 9.171 A3: 2418.612 C4: 0.005 B4: 9.186 A4: 3124.540 C5: 0.017 B5: 9.193 A5: 8708.966 C6: 0.020 B6: 9.203 A6: -5981.197 C7: 0.050 B7: 9.247 A7: 8361.033 C8: 0.050 B8: 9.264 A8: -6932.100 C9: 0.032 B9: 9.274 A9: 10844.050 C10: 0.006 B10: 9.342 A10: 1084.792 C11: 0.009 B11: 9.372 A11: 2428.465 C12: 0.031 B12: 9.484 A12: 11104.174 C13: 0.036 B13: 9.494 A13: 167.365 C14: 0.008 B14: 9.504 A14: 1642.195 C15: 0.026 B15: 9.513 A15: -6249.097
CMT_1315_933	Fitting method = least sq Function evals = 4289 Data points = 4547 Variables = 24 Chi-square = 9.38×10^{13} Reduced chi-sq. = 2.07×10^{10} Akaike info crit = 108041.08 Bayesian info crit = 108195.21	C1: 0.032 B1: 8.082 A1: 99157.843 C2: 0.047 B2: 8.214 A2: 132816.410 C3: 0.028 B3: 8.349 A3: 33673.731 C4: 0.050 B4: 8.456 A4: 65173.659 C5: 0.050 B5: 8.703 A5: 65360.402 C6: 0.050 B6: 8.874 A6: 42017.234 C7: 0.028 B7: 9.363 A7: 17232.963 C8: 0.050 B8: 9.518 A8: 40076.604 C9: 0.026 B9: 10.587 A9: 41516.848 C10: 0.050 B10: 11.586 A10: 76087.124 C11: 0.025 B11: 11.681 A11: 34369.312 C12: 0.050 B12: 12.792 A12: 37864.713

Table A1 (continued)

Data File	Model Fit Statistics	Model View
Clean + Lipophilic		
CLS_827_509	Fitting method = least sq Function evals = 52 Data points = 50 Variables = 2 Chi-square $= 2.81 \times 10^{11}$ Reduced chi-sq. $= 5.85 \times 10^{9}$ Akaike info crit = 1126.48 Bayesian info crit = 1130.30	C1: 0.010 B1: 11.223 A1: 37010.756
CLD_865_524	Fitting method = least sq Function evals = 108 Data points = 1939 Variables = 4 Chi-square $= 6.91 \times 10^{14}$ Reduced chi-sq. $= 3.57 \times 10^{11}$ Akaike info crit = 51584.08 Bayesian info crit = 51606.36	C1: 0.010 B1: 10.955 A1: 137044.144 C2: 0.017 B2: 10.995 A2: 112675.968
CLT_253_288	Fitting method = least sq Function evals = 127 Data points = 1411 Variables = 6 Chi-square $= 3.88 \times 10^{14}$ Reduced chi-sq. $= 2.76 \times 10^{11}$ Akaike info crit = 37177.00 Bayesian info crit = 37208.51	C1: 0.011 B1: 10.603 A1: 115890.697 C2: 0.025 B2: 10.774 A2: 349301.416 C3: 0.016 B3: 11.676 A3: 31714.947
Low Noise + Hydrophilic		
LHS_49_230	Fitting method = least sq Function evals = 34 Data points = 41 Variables = 2 Chi-square $= 8.92 \times 10^{9}$ Reduced chi-sq. $= 2.29 \times 10^{8}$ Akaike info crit = 791.11 Bayesian info crit = 794.54	C1: 0.013 B1: 1.744 A1: 10217.278

Table A1 (continued)

Data File	Model Fit Statistics	Model View
Low Noise + Hydrophilic (Continued)		
LHD_108_86	Fitting method = least sq Function evals = 370 Data points = 1278 Variables = 6 Chi-square $= 1.13 \times 10^{11}$ Reduced chi-sq. $= 8.86 \times 10^{7}$ Akaike info crit = 23393.39 Bayesian info crit = 23424.31	C1: 0.032 B1: 0.927 A1: 3585.962 C2: 0.010 B2: 1.009 A2: 2274.590 C3: 0.050 B3: 1.296 A3: 3053.554
LHT_75_261	Fitting method = least sq Function evals = 126 Data points = 1355 Variables = 6 Chi-square $= 4.26 \times 10^{11}$ Reduced chi-sq. $= 3.16 \times 10^{8}$ Akaike info crit = 26524.05 Bayesian info crit = 26555.32	C1: 0.011 B1: 3.201 A1: 1956.141 C2: 0.050 B2: 3.260 A2: 7977.781 C3: 0.015 B3: 3.533 A3: 20517.349
Low Noise + Mixed		
LMS_7_189	Fitting method = least sq Function evals = 52 Data points = 43 Variables = 2 Chi-square $= 9.06 \times 10^{11}$ Reduced chi-sq. $= 2.21 \times 10^{10}$ Akaike info crit = 1026.17 Bayesian info crit = 1029.70	C1: 0.008 B1: 7.589 A1: 33439.582
LMD_151_297	Fitting method = least sq Function evals = 78 Data points = 327 Variables = 6 Chi-square $= 1.97 \times 10^{10}$ Reduced chi-sq. $= 6.13 \times 10^{7}$ Akaike info crit = 5869.33 Bayesian info crit = 5892.07	C1: 0.016 B1: 6.147 A1: 1946.591 C2: 0.010 B2: 6.321 A2: 3797.443 C3: 0.030 B3: 6.585 A3: 2036.545

Table A1 (continued)

Data File	Model Fit Statistics	Model View
Low Noise + Mixed (Continued)		
LMT_163_300	Fitting method = least sq Function evals = 338 Data points = 767 Variables = 10 Chi-square = 5.2×10^{12} Reduced chi-sq. = 6.8×10^{9} Akaike info crit = 17376 Bayesian info crit = 17423	C1: 0.008 B1: 6.704 A1: 94190.825 C2: 0.009 B2: 7.300 A2: 18086.199 C3: 0.050 B3: 7.411 A3: 35197.100 C4: 0.009 B4: 7.583 A4: 16768.698 C5: 0.010 B5: 7.694 A5: 13284.838
Low Noise + Lipophilic		
LLS_89_272	Fitting method = least sq Function evals = 94 Data points = 45 Variables = 2 Chi-square = 8.43×10^{10} Reduced chi-sq. = 1.96×10^{9} Akaike info crit = 964.81 Bayesian info crit = 968.42	C1: 0.016 B1: 11.192 A1: 8614.819
LLD_81_263	Fitting method = least sq Function evals = 614 Data points = 895 Variables = 8 Chi-square = 2.20×10^{11} Reduced chi-sq. = 2.48×10^{9} Akaike info crit = 17305.87 Bayesian info crit = 17344.25	C1: 0.041 B1: 10.605 A1: 10655.028 C2: 0.018 B2: 10.701 A2: 1086.379 C3: 0.050 B3: 11.712 A3: 9214.079 C4: 0.050 B4: 11.889 A4: 6968.989
LLT_57_238	Fitting method = least sq Function evals = 232 Data points = 1299 Variables = 10 Chi-square = 4.80×10^{12} Reduced chi-sq. = 3.73×10^{9} Akaike info crit = 28638.01 Bayesian info crit = 289689.70	C1: 0.021 B1: 10.451 A1: 37614.454 C2: 0.011 B2: 10.788 A2: 56351.928 C3: 0.024 B3: 10.967 A3: 19148.912 C4: 0.023 B4: 11.490 A4: 10254.435 C5: 0.013 B5: 11.673 A5: 41481.673

Table A1 (continued)

Data File	Model Fit Statistics	Model View
High Noise + Hydrophilic		
HHS_507_306	Fitting method = least sq Function evals = 263 Data points = 142 Variables = 4 Chi-square $= 1.66 \times 10^{10}$ Reduced chi-sq. $= 1.20 \times 10^{8}$ Akaike info crit = 2645.53 Bayesian info crit = 2657.36	C1: 0.050 B1: 0.662 A1: 7444.985 C2: 0.050 B2: 0.839 A2: 3095.042
HHD_507_306	Fitting method = least sq Function evals = 303 Data points = 142 Variables = 8 Chi-square $= 8.76 \times 10^{9}$ Reduced chi-sq. $= 6.54 \times 10^{7}$ Akaike info crit = 2563.14 Bayesian info crit = 2586.79	C1: 0.022 B1: 0.662 A1: 3683.899 C2: 0.032 B2: 0.734 A2: 2755.046 C3: 0.017 B3: 0.803 A3: 762.169 C4: 0.016 B4: 0.839 A4: 652.332
HHT_23_185	Fitting method = least sq Function evals = 5061 Data points = 10002 Variables = 50 Chi-square $= 3.36 \times 10^{13}$ Reduced chi-sq. $= 3.37 \times 10^{9}$ Akaike info crit = 219488.90 Bayesian info crit = 219849.43	C1: 0.050 B1: 0.011 A1: 11791.121 C2: 0.013 B2: 0.526 A2: 13004.850 C3: 0.050 B3: 1.412 A3: 11263.337 C4: 0.050 B4: 1.778 A4: 10789.312 C5: 0.050 B5: 2.008 A5: 10484.735 C6: 0.050 B6: 2.216 A6: 10556.967 C7: 0.050 B7: 2.623 A7: 11257.386 C8: 0.050 B8: 4.067 A8: 12622.162 C9: 0.050 B9: 4.319 A9: 12882.175 C10: 0.050 B10: 4.720 A10: 13605.874 C11: 0.050 B11: 5.075 A11: 14118.873 C12: 0.050 B12: 5.491 A12: 15426.992 C13: 0.050 B13: 6.556 A13: 16331.154 C14: 0.050 B14: 6.938 A14: 16838.927 C15: 0.050 B15: 7.555 A15: 29414.749 C16: 0.050 B16: 8.203 A16: 16077.473 C17: 0.050 B17: 8.930 A17: 13888.402

Table A1 (continued)

Data File	Model Fit Statistics	Model View
HHT_23_185 (Continued)		C18: 0.050 B18: 9.280 A18: 12551.084 C19: 0.050 B19: 9.703 A19: 11233.640 C20: 0.050 B20: 10.588 A20: 9103.178 C21: 0.050 B21: 10.910 A21: 9134.124 C22: 0.050 B22: 11.215 A22: 7682.866 C23: 0.050 B23: 11.392 A23: 9955.576 C24: 0.050 B24: 14.366 A24: 11532.549 C25: 0.050 B25: 14.995 A25: 10489.416
High Noise + Mixed		
HMS_125_211	Fitting method = least sq Function evals = 78 Data points = 408 Variables = 6 Chi-square $= 3.79 \times 10^{9}$ Reduced chi-sq. $= 9.42 \times 10^{6}$ Akaike info crit = 6557.88 Bayesian info crit = 6581.94	C1: 0.020 B1: 6.333 A1: 2283.317 C2: 0.019 B2: 6.519 A2: 770.352 C3: 0.016 B3: 6.558 A3: 780.250
HMD_152_217	Fitting method = least sq Function evals = 86 Data points = 158 Variables = 4 Chi-square $= 1.28 \times 10^{11}$ Reduced chi-sq. $= 8.32 \times 10^{8}$ Akaike info crit = 3249.14 Bayesian info crit = 3261.39	C1: 0.013 B1: 9.609 A1: 6321.669 C2: 0.011 B2: 9.741 A2: 7646.806
HMT_85_199	Fitting method = least sq Function evals = 207 Data points = 259 Variables = 4 Chi-square $= 4.93 \times 10^{11}$ Reduced chi-sq. $= 1.93 \times 10^{9}$ Akaike info crit = 5542.10 Bayesian info crit = 5556.33	C1: 0.016 B1: 9.608 A1: 5865.250 C2: 0.011 B2: 9.739 A2: 13416.975

Table A1 (continued)

Data File	Model Fit Statistics	Model View
High Noise + Lipophilic		
HLS_609_343	Fitting method = least sq Function evals = 116 Data points = 41 Variables = 4 Chi-square $= 1.05 \times 10^{10}$ Reduced chi-sq. $= 2.83 \times 10^{8}$ Akaike info crit = 801.69 Bayesian info crit = 808.54	C1: 0.005 B1: 10.867 A1: 1335.793 C2: 0.050 B2: 10.905 A2: 5654.655
HLD_159_219	Fitting method = least sq Function evals = 214 Data points = 95 Variables = 8 Chi-square $= 2.01 \times 10^{10}$ Reduced chi-sq. $= 2.32 \times 10^{8}$ Akaike info crit = 1837.35 Bayesian info crit = 1857.78	C1: 0.005 B1: 14.085 A1: 522.131 C2: 0.039 B2: 14.124 A2: 10167.009 C3: 0.005 B3: 14.166 A3: 462.063 C4: 0.006 B4: 14.199 A4: 534.156
HLT_83_199	Fitting method = least sq Function evals = 12383 Data points = 440 Variables = 18 Chi-square $= 1.73 \times 10^{10}$ Reduced chi-sq. $= 4.11 \times 10^{7}$ Akaike info crit = 7731.24 Bayesian info crit = 7804.8	C1: 0.050 B1: 14.314 A1: 3967.250 C2: 0.019 B2: 14.395 A2: 752.273 C3: 0.050 B3: 14.452 A3: 6711.188 C4: 0.048 B4: 14.534 A4: -321.497 C5: 0.050 B5: 14.588 A5: 5745.159 C6: 0.050 B6: 14.750 A6: 6112.604 C7: 0.048 B7: 14.792 A7: -1745.740 C8: 0.050 B8: 14.860 A8: 2957.700 C9: 0.036 B9: 14.960 A9: 1734.996

Table A2

Explanation of Model Fit Statistics

Attribute Name	**Description / Formula**
nfev	Number of function evaluations
nvarys	Number of variables in N_{varys}
ndata	Number of data points: N
nfree	Degrees of freedom in fit: N – N_{varys}
residual	Residual array, returned by the objective function: {$Resid_i$}
chisqr	Chi-square: $\chi^2 = \sum_i^N [\, Resid_i]^2$
redchi	Reduced chi-square: $\chi_v^2 = \chi^2 / (N - N_{varys})$
aic	Akaike information criterion statistic (see below)
bic	Bayesian information criterion statistic (see below)
var_names	Ordered list of variable parameter names used for init_vals and covar
covar	Covariance matrix (with rows/columns using var_names)
init_vals	List of initial values for variable parameters

The *MinimizerResult* includes the traditional chi-square and reduced chi-square statistics, shown in Equations (A1) and (A2):

$$\chi^2 = \sum_i^N r_i^2 \tag{A1}$$

$$\chi_v^2 = \chi^2 / (N - N_{varys}), \tag{A2}$$

where *r* is the residual array returned by the objective function, which, for data modeling usages, is likely to be (*data-model / uncertainty*), *N* is the number of data points (*ndata*), and N_{varys} is number of variable parameters.

The Akaike Information Criterion (*aic*) and Baeysian Information Criterion (*bic*) statistics are also included. These each give slightly different measures for the relative quality of a fit. These statistics attempt to balance the quality of the fit with the number of variable parameters the fit uses. The equations for the *aic* and *bic* are shown in Equations (A3) and (A4), respectively:

$$\text{aic} = N \ln\left(\frac{\chi^2}{N}\right) + 2N_{varys} \tag{A3}$$

$$\text{bic} = N \ln\left(\frac{\chi^2}{N}\right) + \ln(N)N_{varys}. \tag{A4}$$

One typically selects the model with the lowest reduced chi-square, the Akaike Information Criterion, and/or the Bayesian Information Criterion, when comparing fits with different numbers of varying parameters. The most conservative of these statistics is the Bayesian Information Criterion.

Appendix B

CWT Test Categories

Appendix B includes images of the Gaussian Model Fit and Model View for the CWT test categories.

Clean-Hydrophilic Category

Results are shown for the following categories: clean-hydrophilic, clean-mixed, and clean-lipophilic.

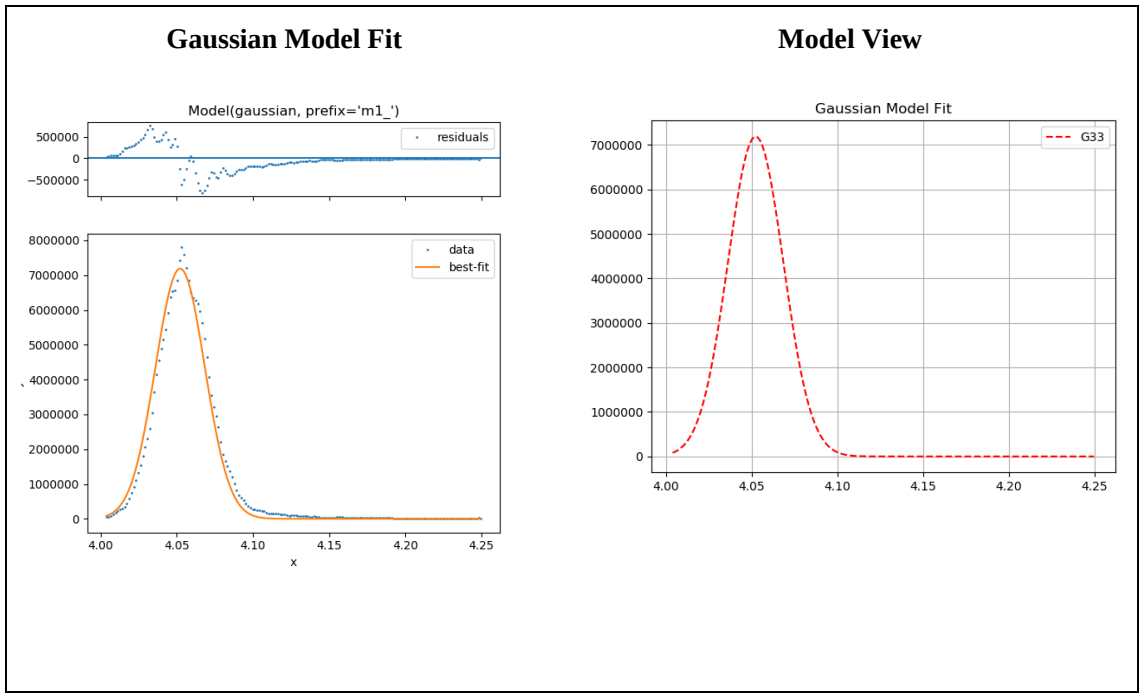

Figure B1. CHS_5_185.0957.

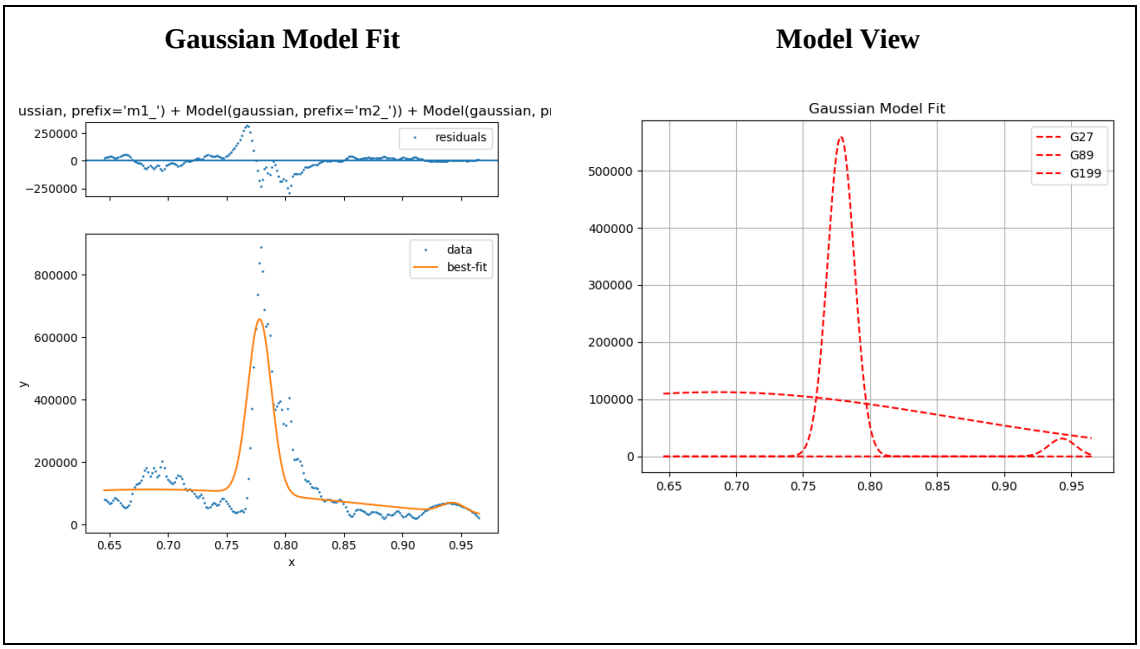

Figure B2. CHD_65_215.

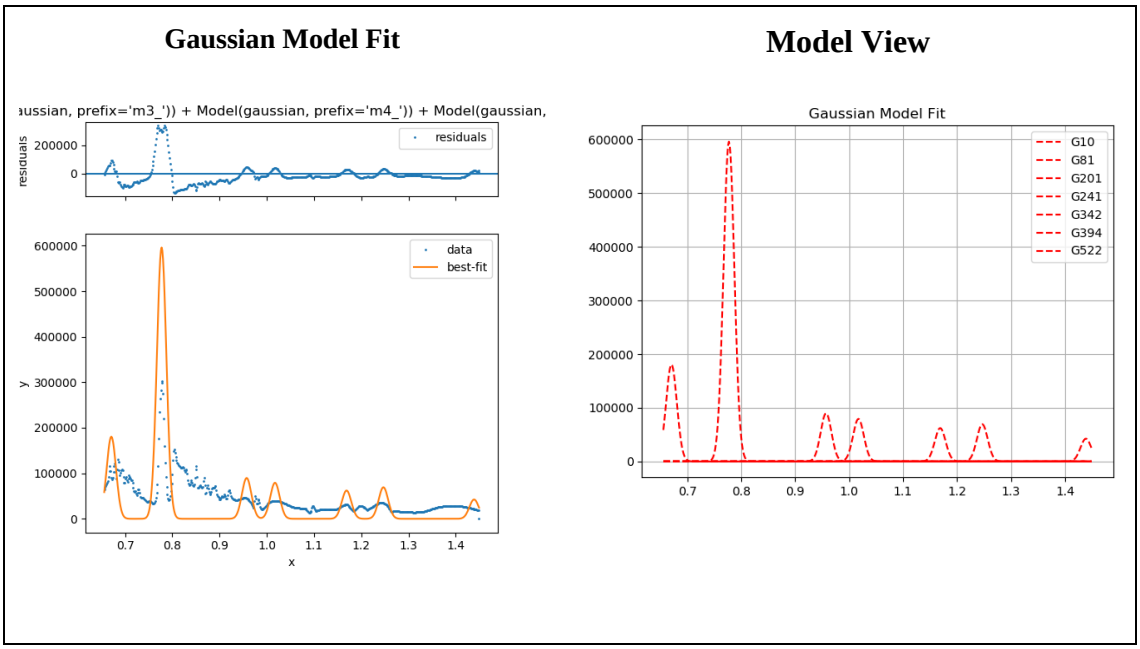

Figure B3. CHT_121_242.

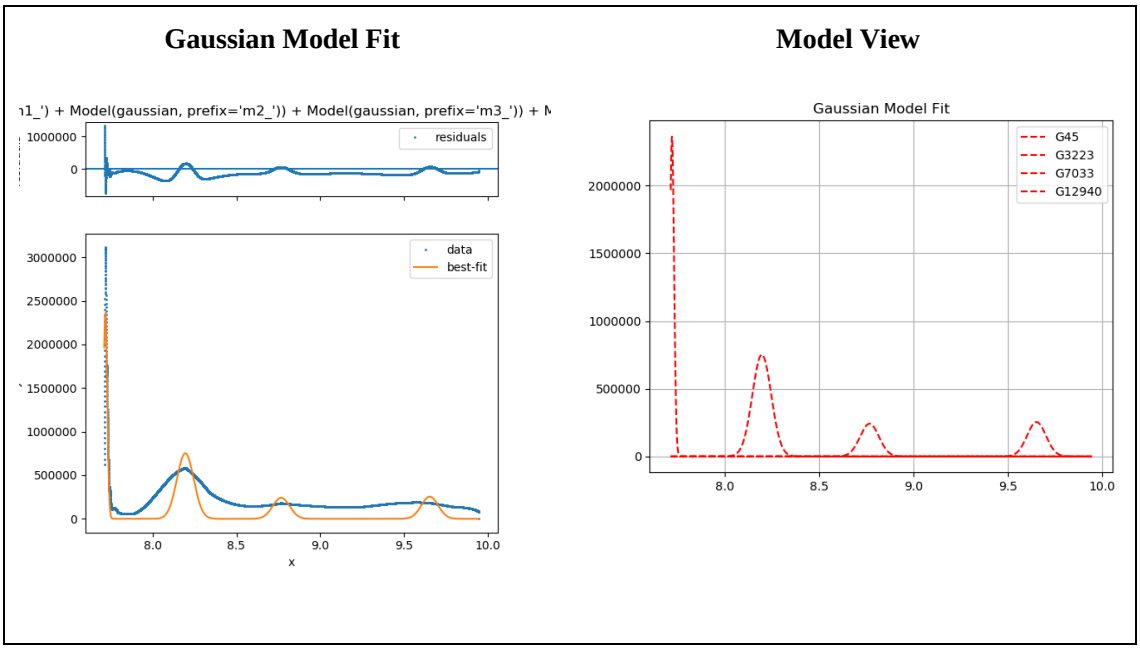

Figure B4. CMS_13089_915.

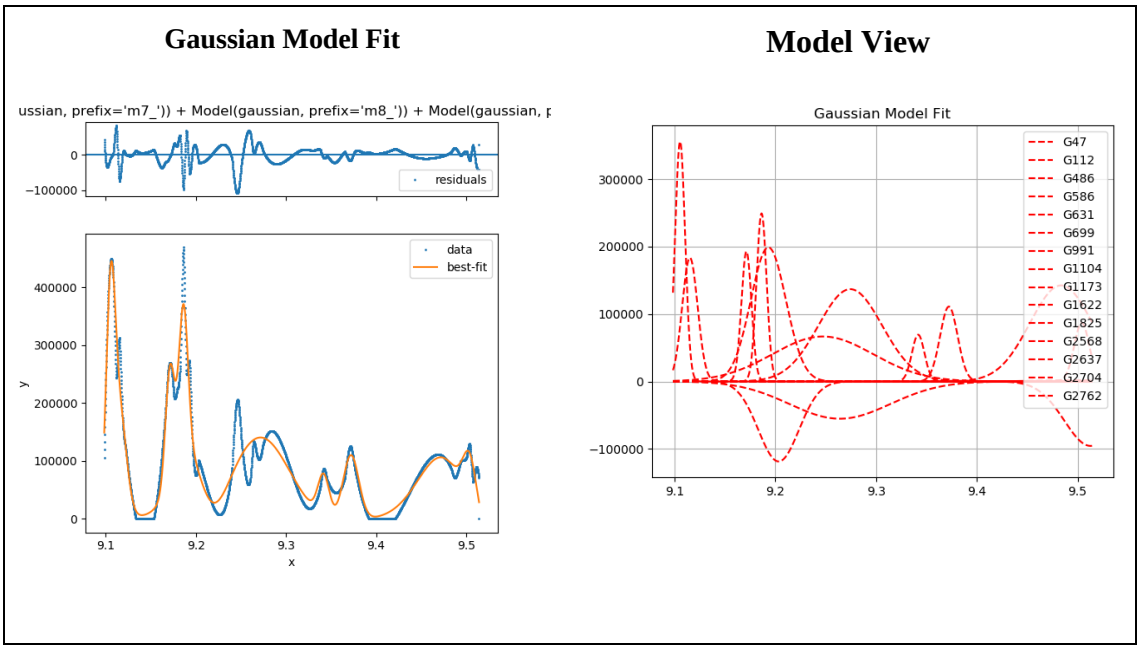

Figure B5. CMD_985_585.

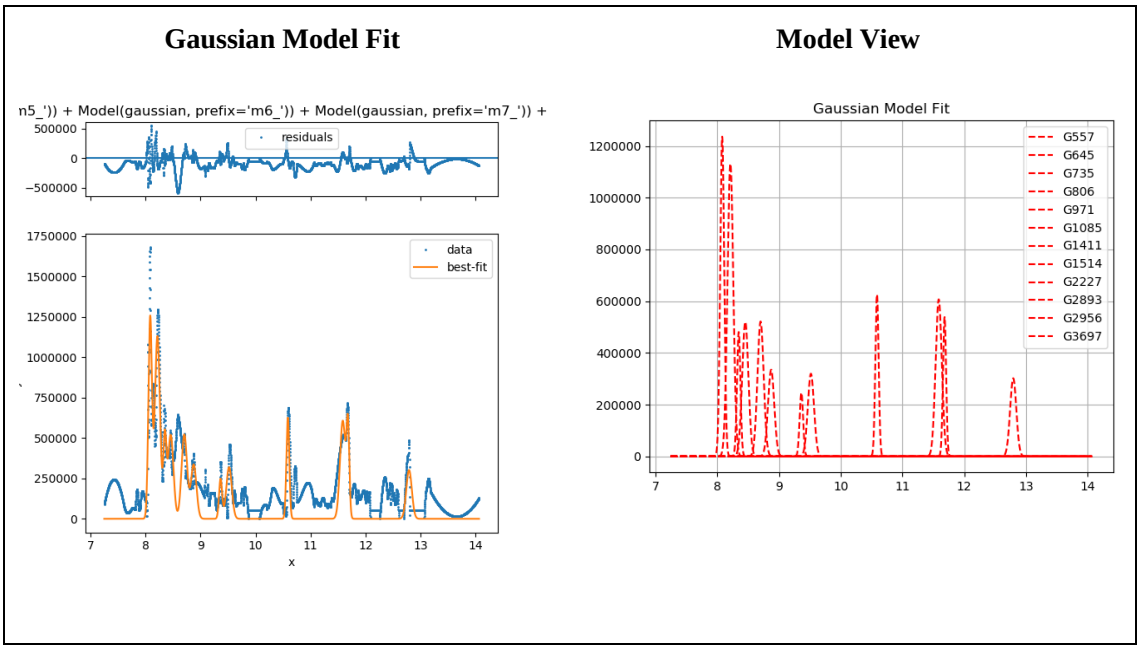

Figure B6. CMT_1315_933.

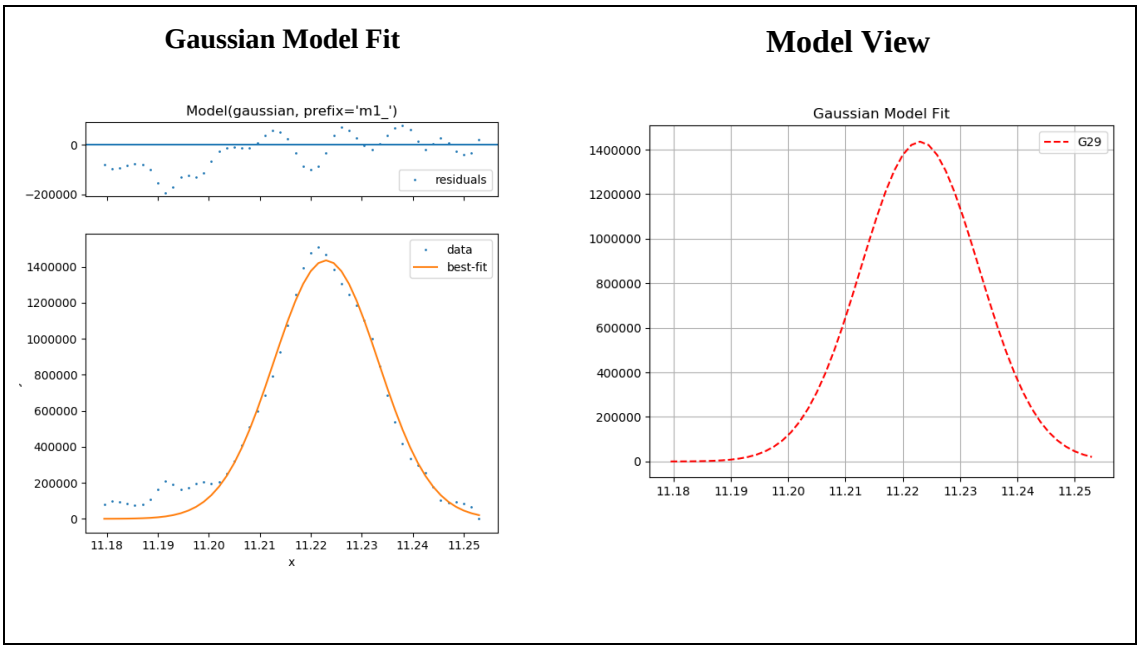

Figure B7. CLS_827_509.

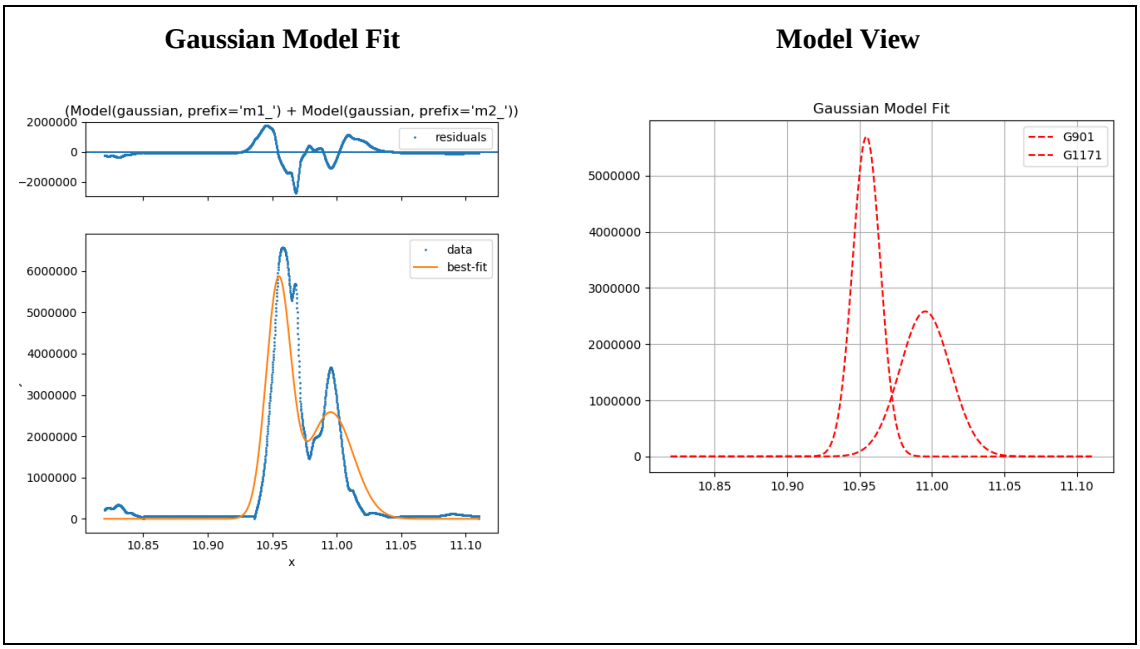

Figure B8. CLD_865_524.

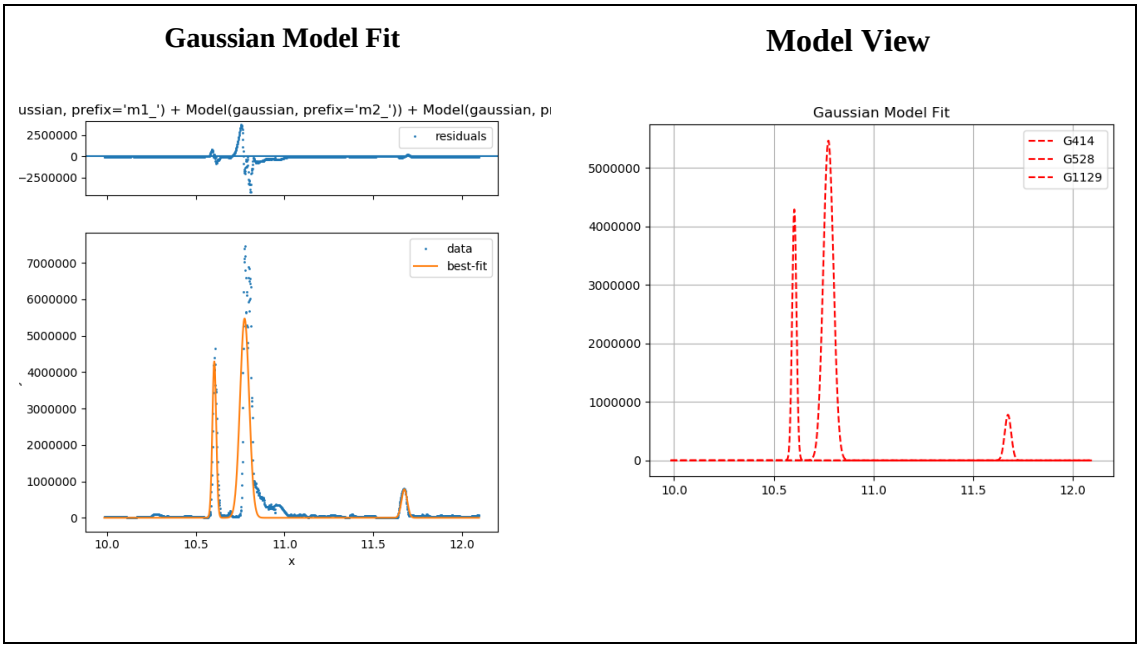

Figure B9. CLT_253_288.

Low-Noise–Hydrophilic Category

Results are shown for the following categories: low-noise–hydrophilic, low-noise–mixed, and low-noise–lipophilic.

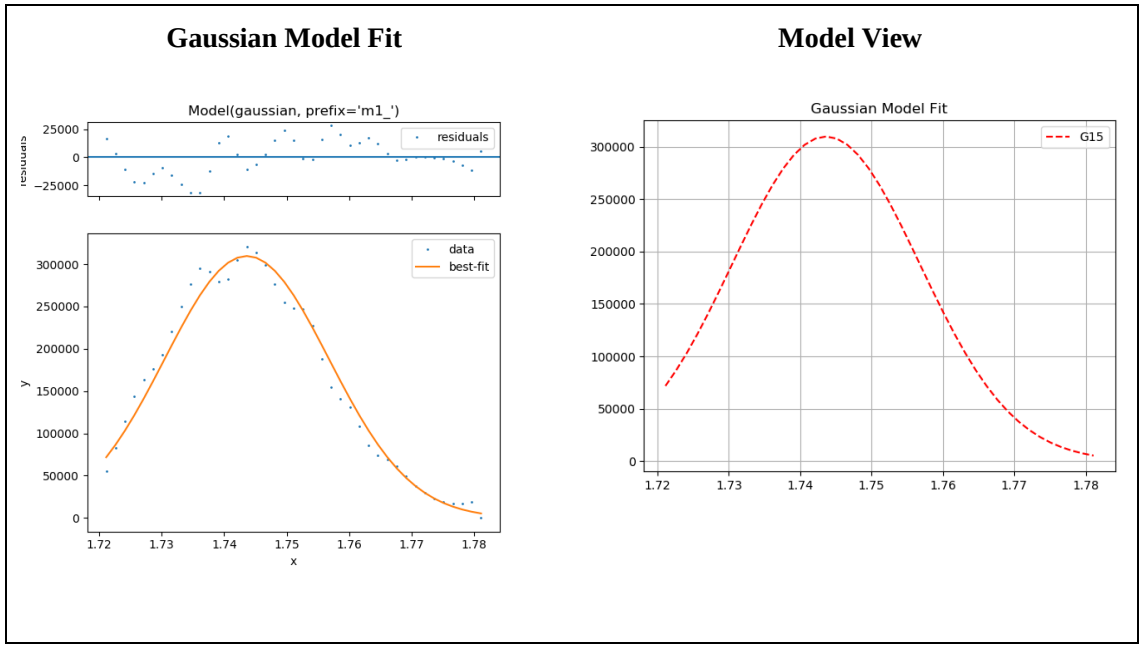

Figure B10. LHS_49_230.

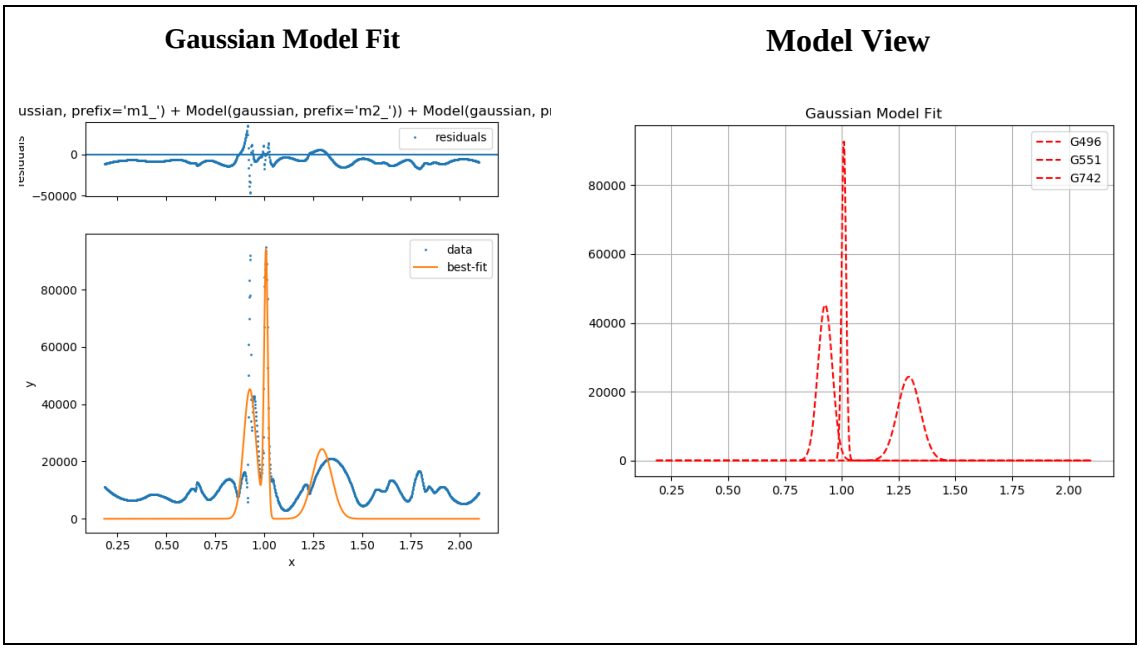

Figure B11. LHD_108_86.

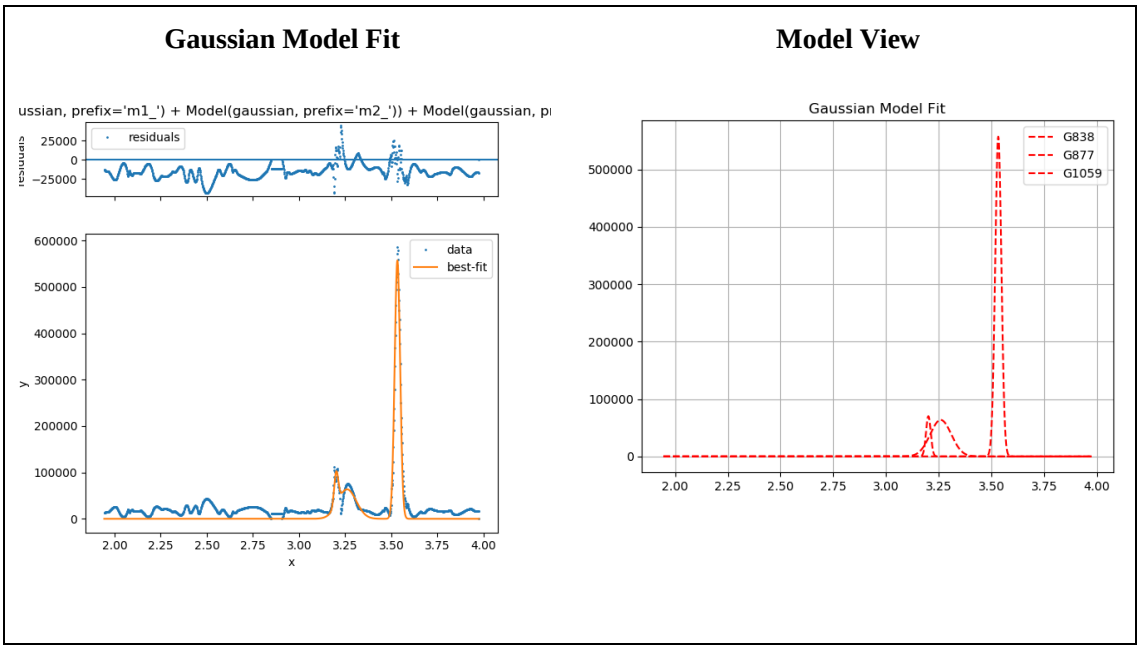

Figure B12. LHT_75_261.

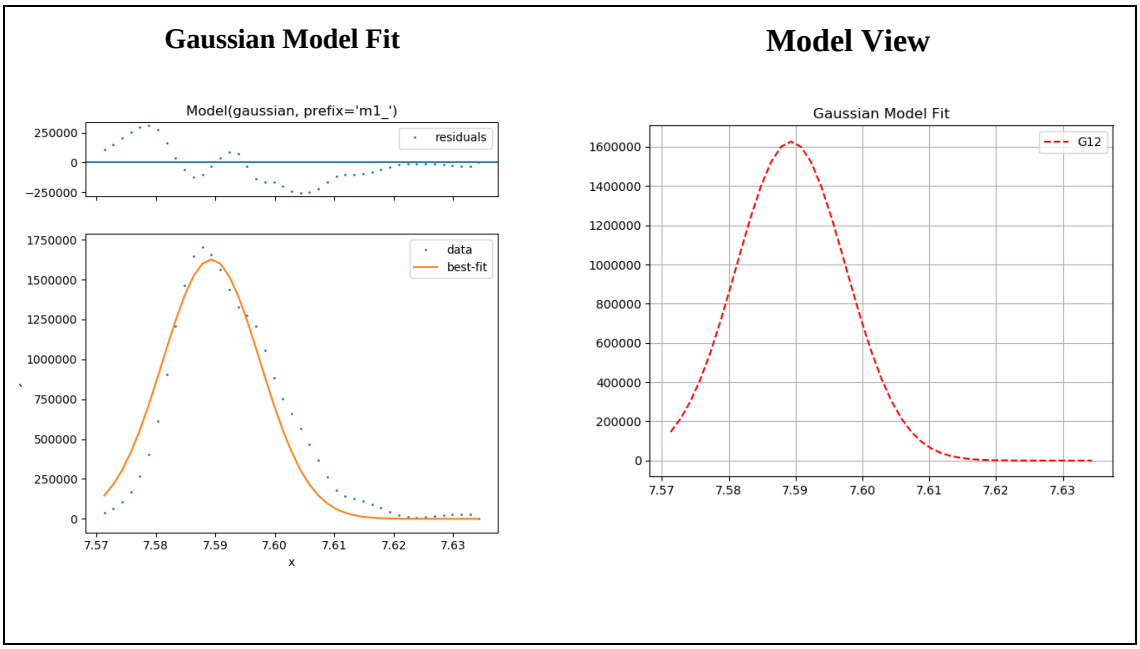

Figure B13. LMS_7_189.

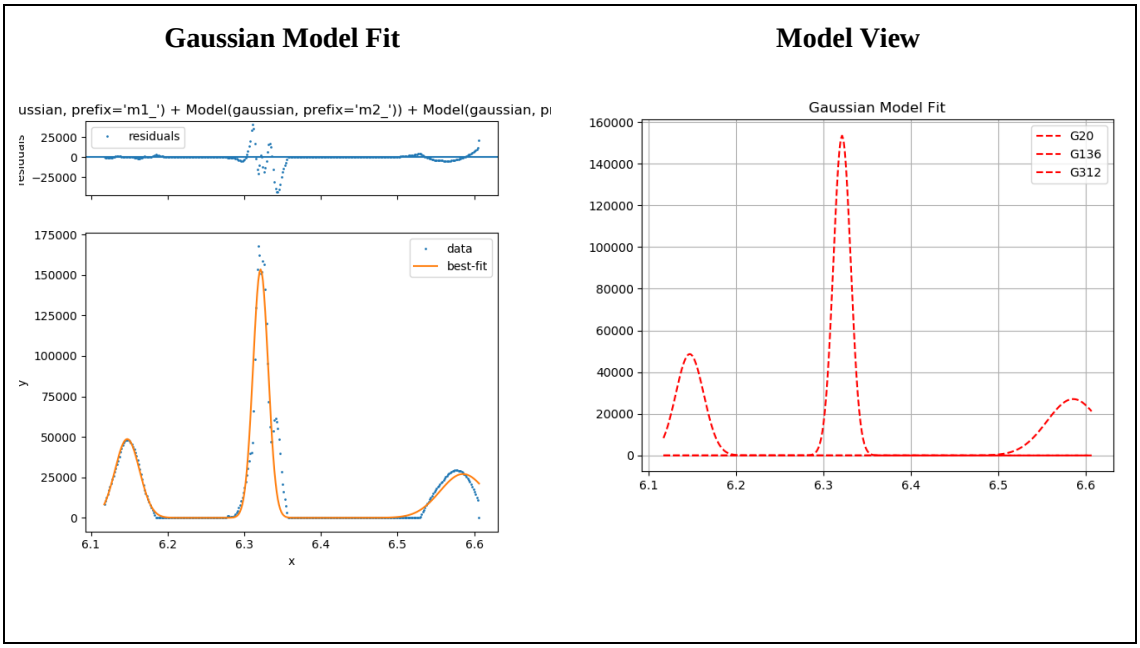

Figure B14. LMD_151_297.

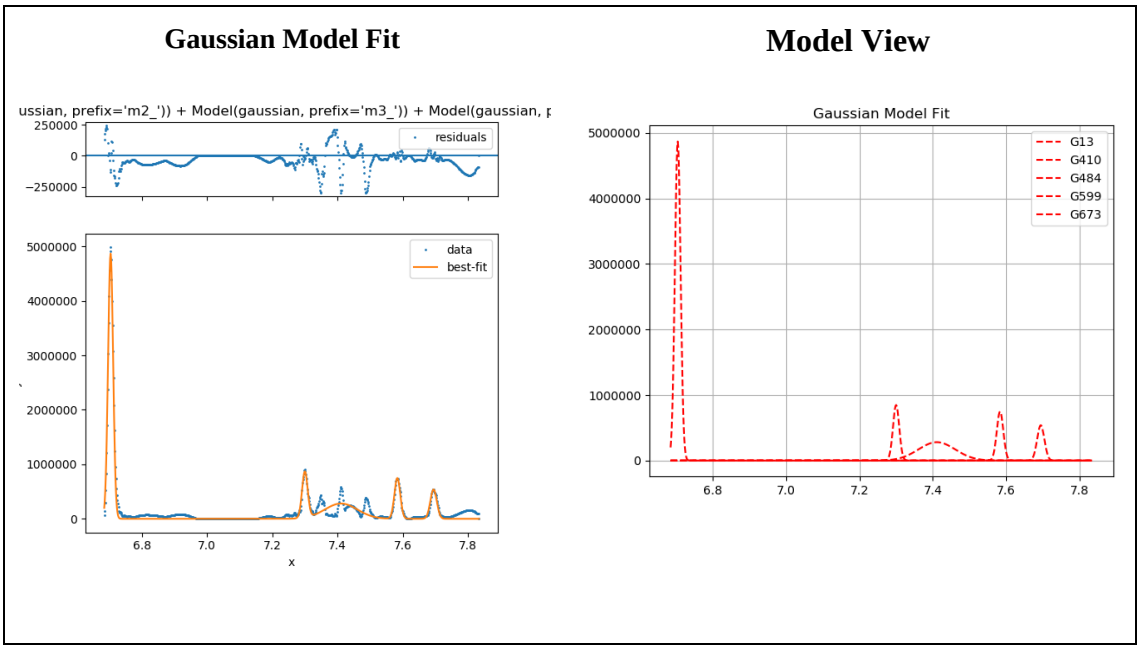

Figure B15. LMT_163_300.

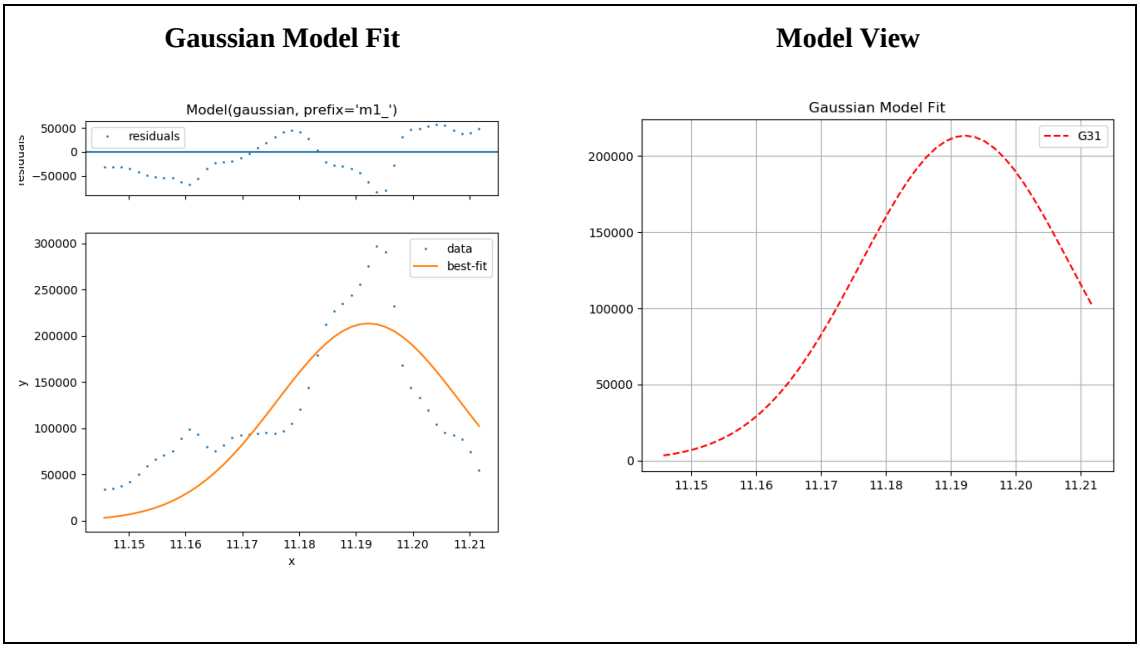

Figure B16. LLS_89_272.

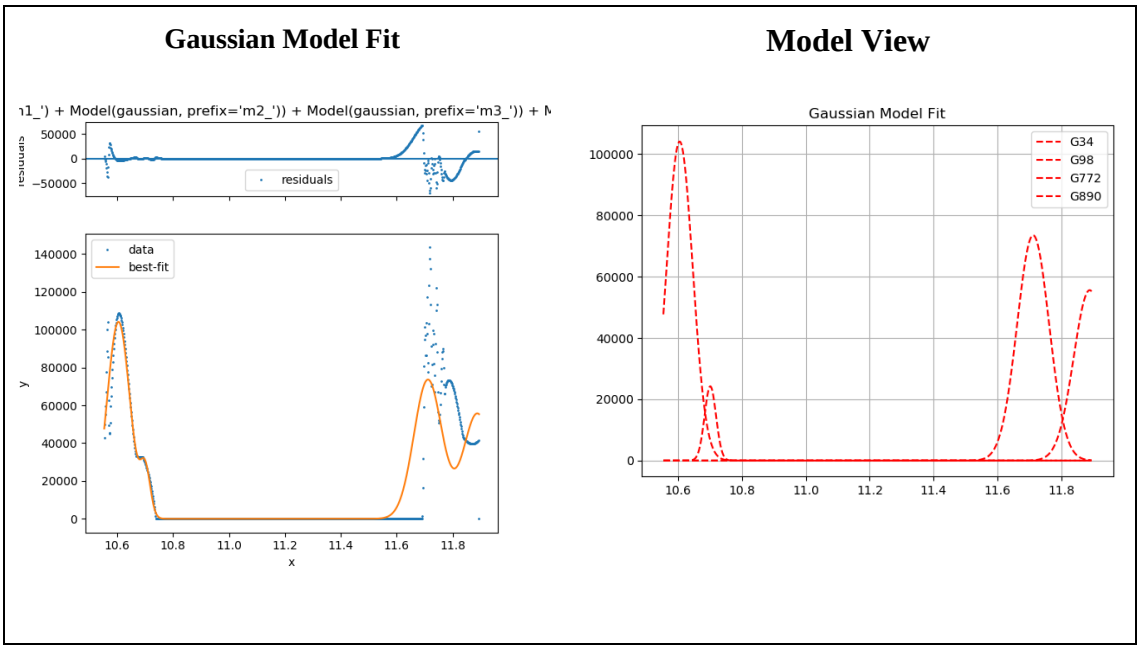

Figure B17. LLD_81_263.

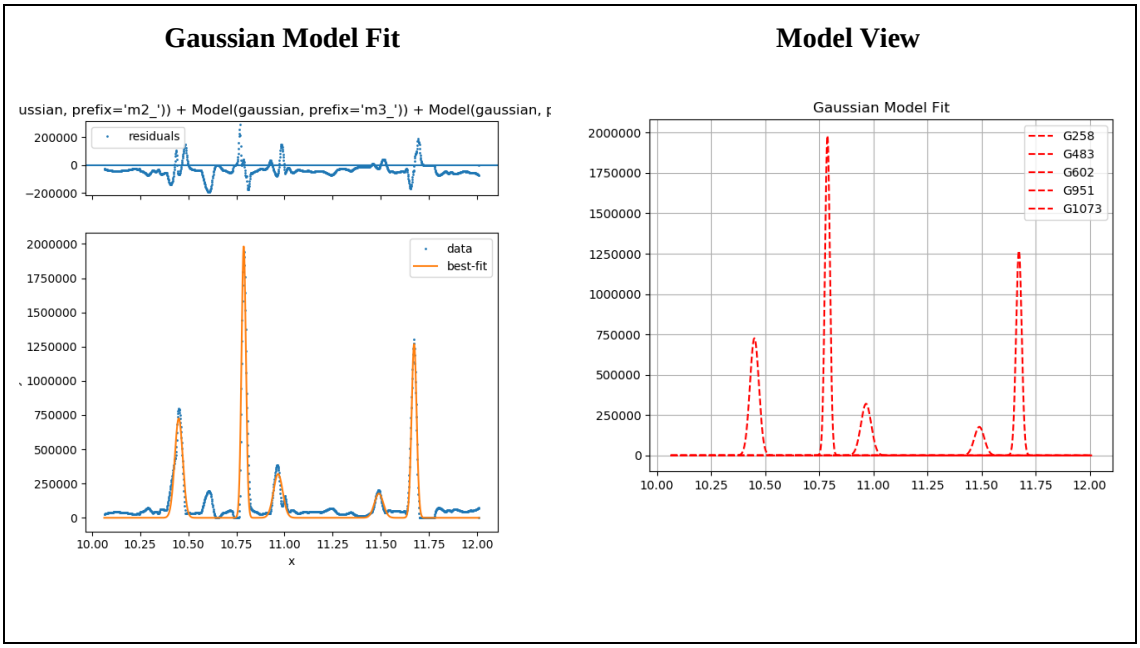

Figure B18. LLT_57_238.

High-Noise–Hydrophilic Category

Results are shown for the following categories: high-noise–hydrophilic, high-noise–mixed, and high-noise–lipophilic.

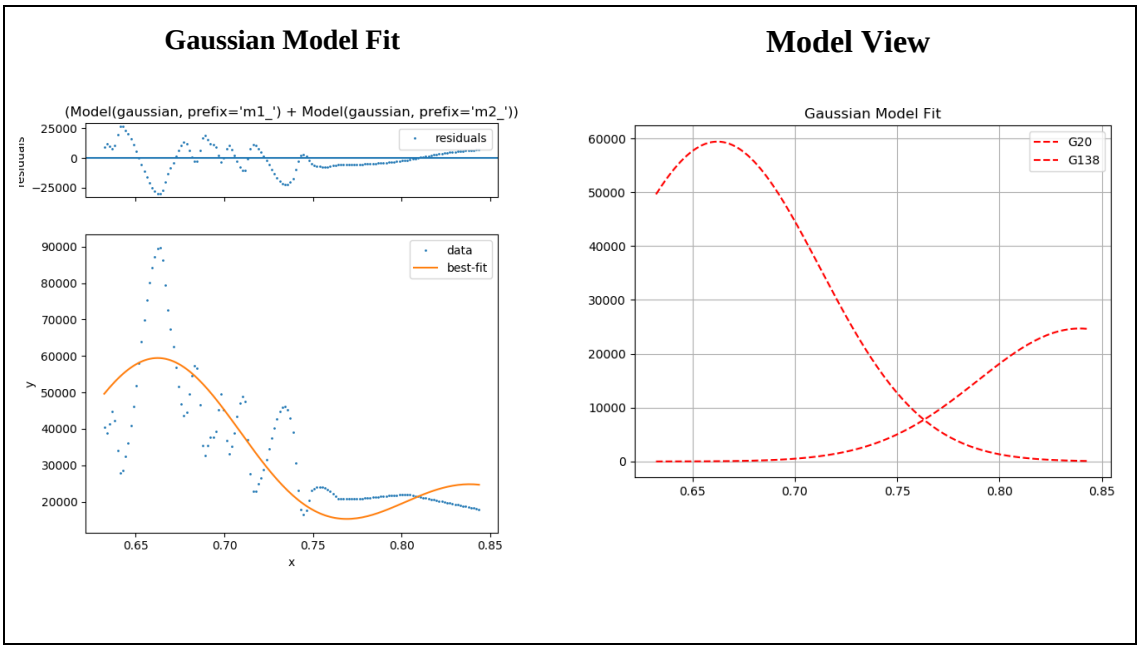

Figure B19. HHS_507_306.

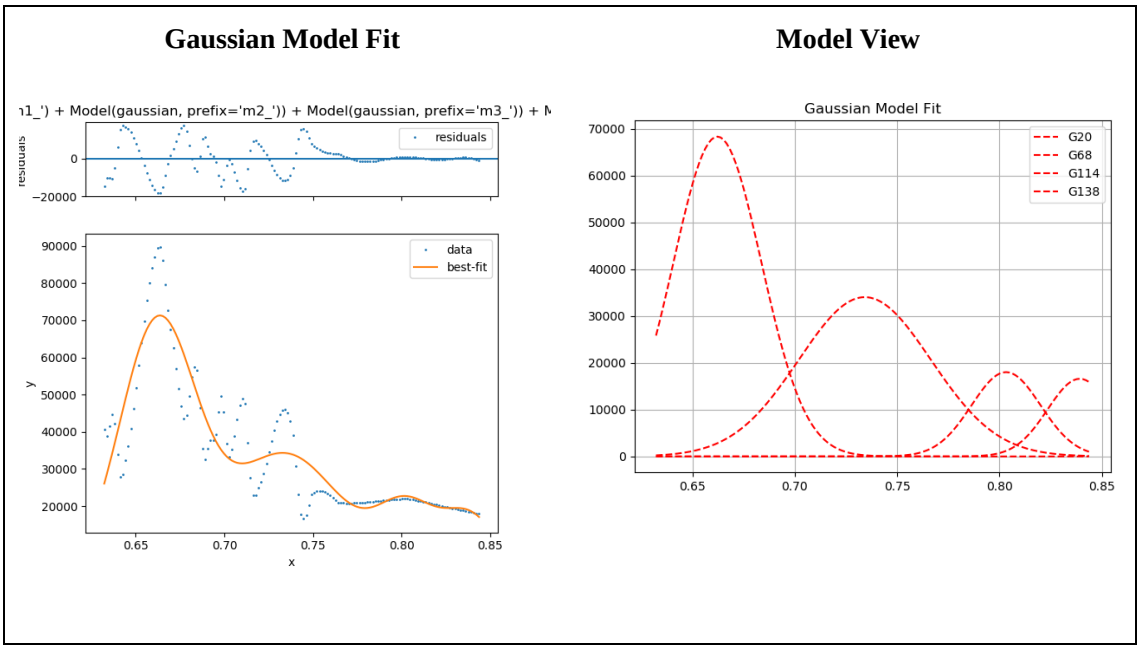

Figure B20. HHD_507_306.

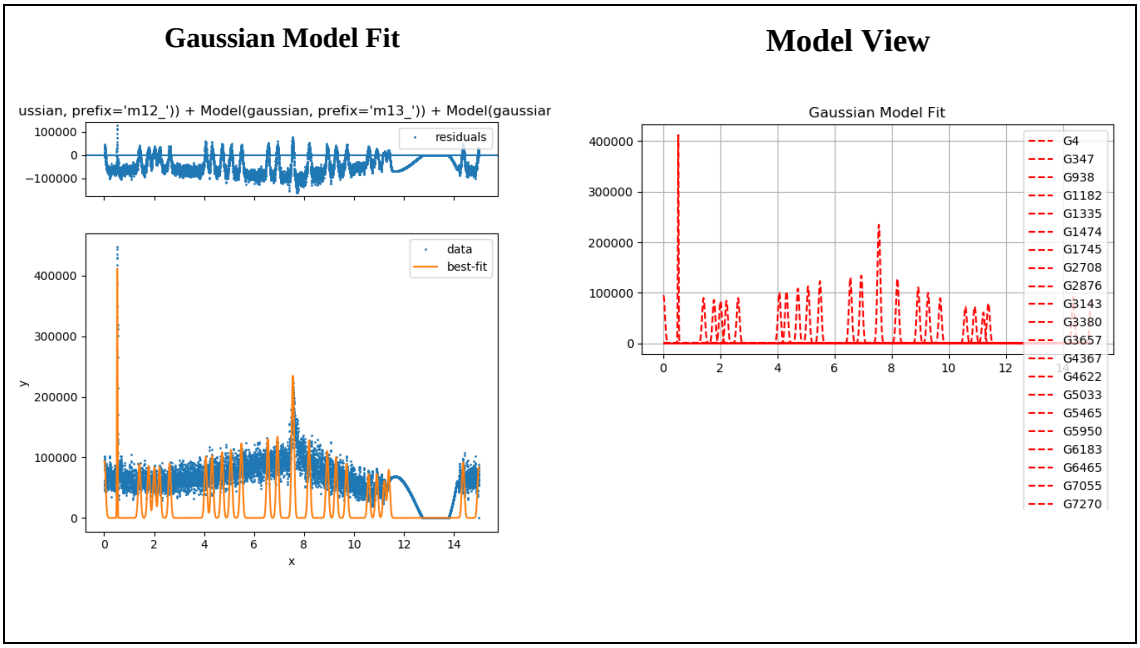

Figure B21. HHT_23_185.

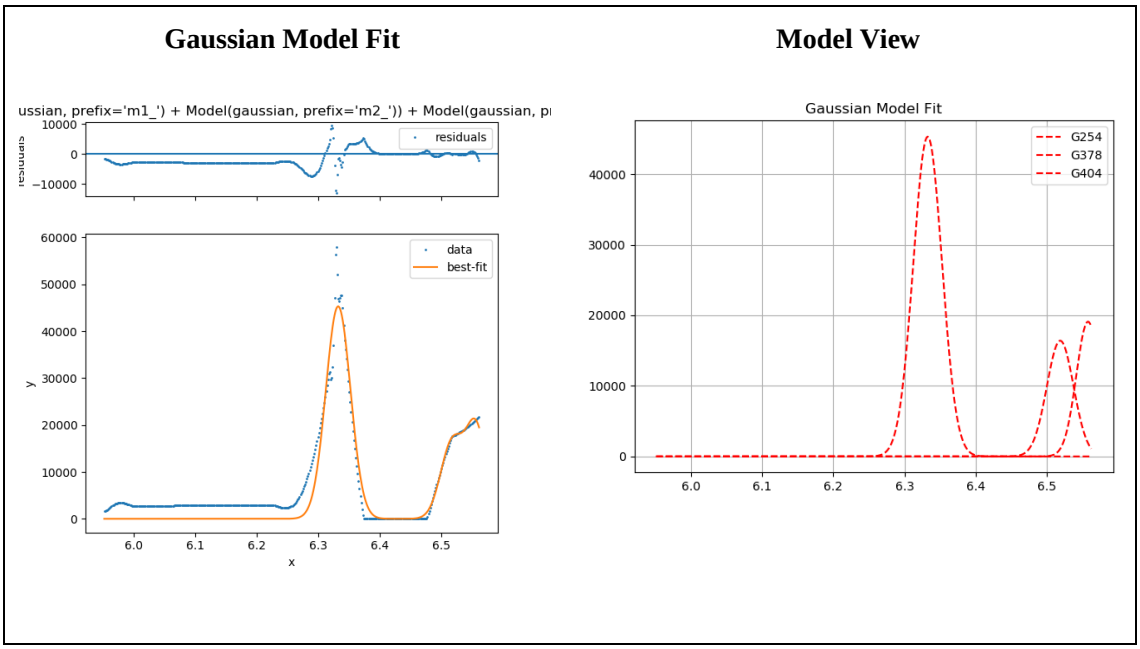

Figure B22. HMS_125_211.

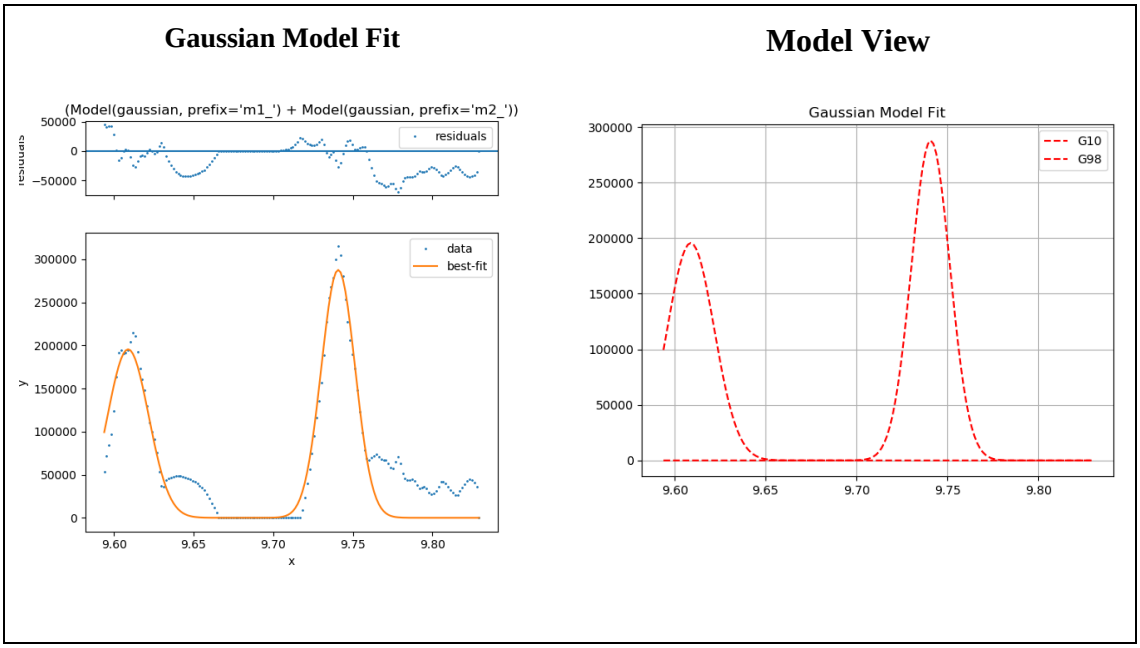

Figure B23. HMD_153_217.

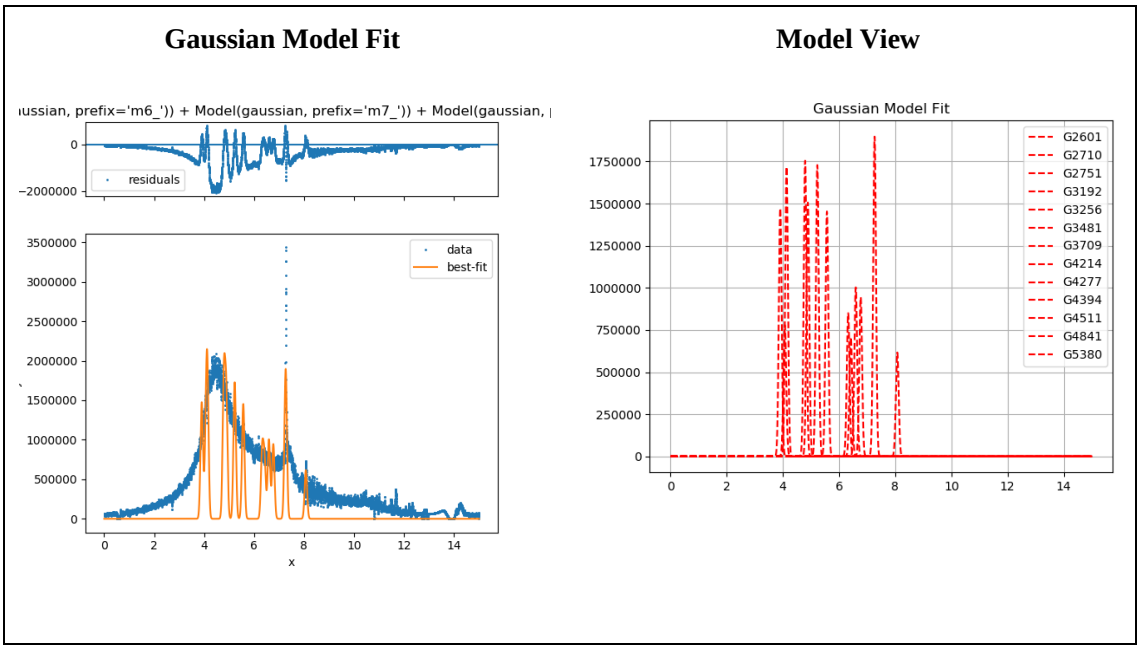

Figure B24. HMT_85_199.

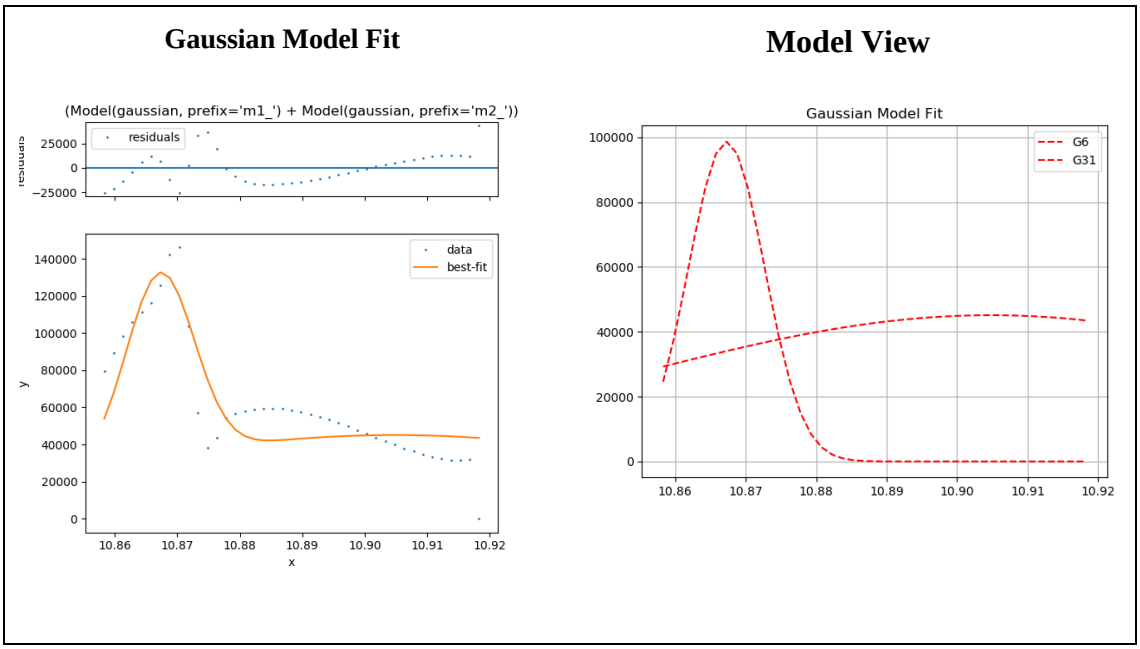

Figure B25. HLS_609_343.

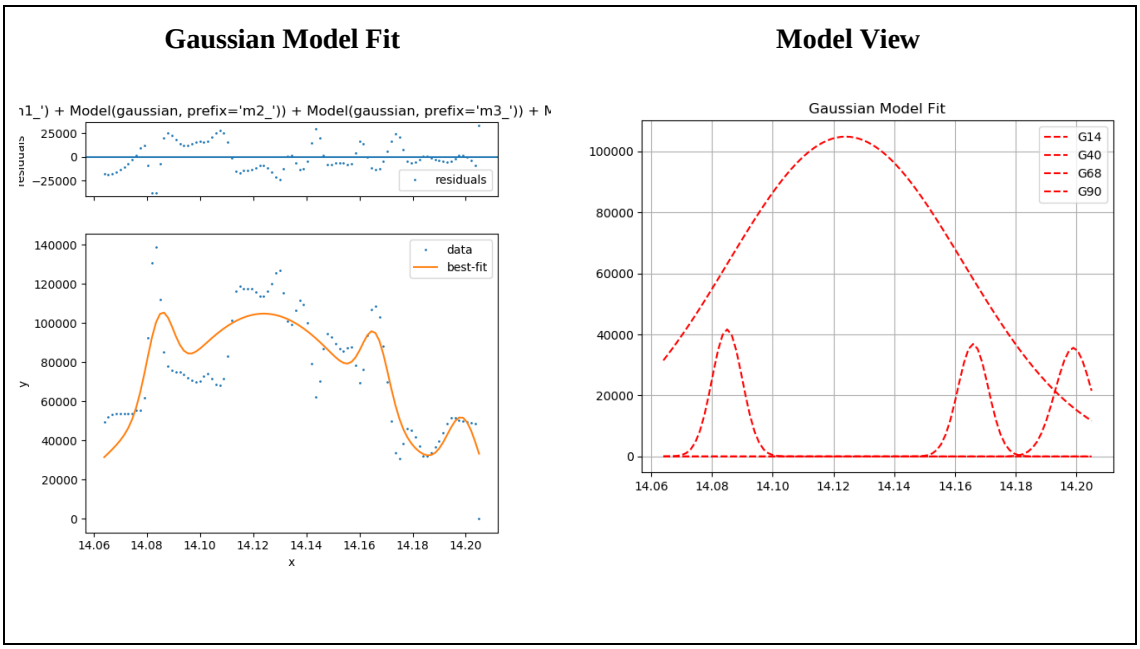

Figure B26. HLD_159_219.

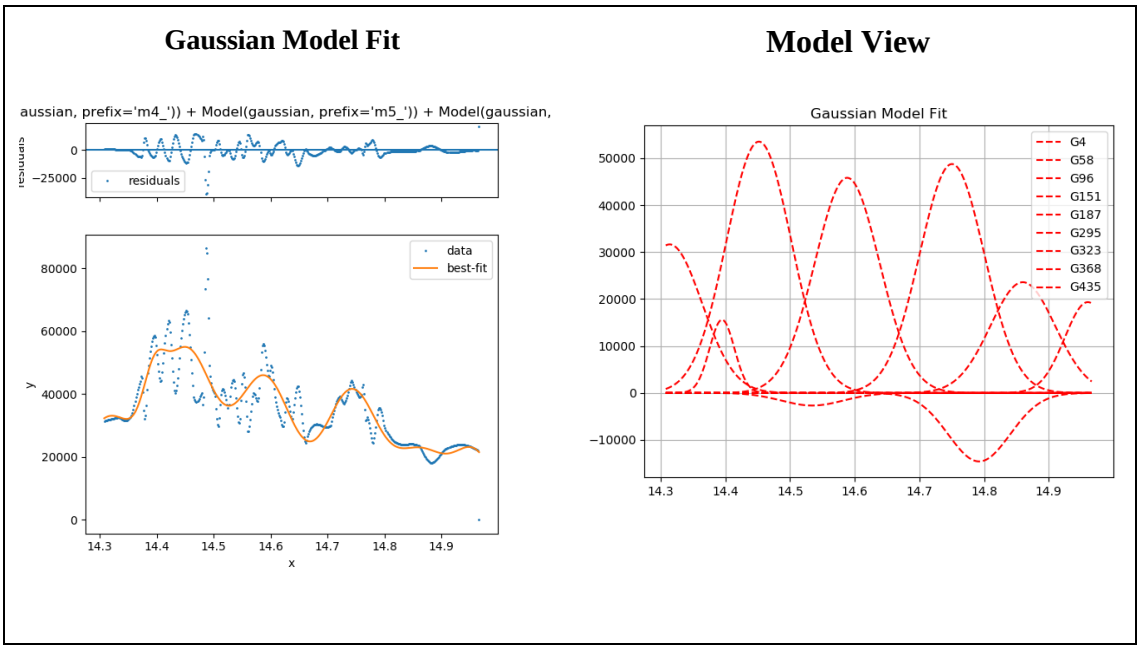

Figure B27. HLT_83_199.