Machine Learning Approaches for Modeling and Correction of Confounding Effects in Complex Biological Data

Chiung-Ting Wu

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

In

Electrical Engineering

Yue Wang, Chair

Guoqiang Yu

William Baumann

Thidapat Chantem

Chang-Tien Lu

May 6th, 2021
Arlington, Virginia

Keywords: bioinformatics, multiple alignment, deconvolution, unsupervised learning, convex analysis, feature selection, tissue heterogeneity

Machine Learning Approaches for Modeling and Correction of Confounding Effects in Complex Biological Data

Chiung-Ting Wu

ABSTRACT (Academic)

With the huge volume of biological data generated by new technologies and the booming of new machine learning based analytical tools, we expect to advance life science and human health at an unprecedented pace. Unfortunately, there is a significant gap between the complex raw biological data from real life and the data required by mathematical and statistical tools. This gap is contributed by two fundamental and universal problems in biological data that are both related to confounding effects. The first is the intrinsic complexities of the data. An observed sample could be the mixture of multiple underlying sources and we may be only interested in one or part of the sources. The second type of complexities come from the acquisition process of the data. Different samples may be gathered at different time and/or from different locations. Therefore, each sample is associated with specific distortion that must be carefully addressed. These confounding effects obscure the signals of interest in the acquired data. Specifically, this dissertation will address the two major challenges in confounding effects removal: alignment and deconvolution.

Liquid chromatography–mass spectrometry (LC-MS) is a standard method for proteomics and metabolomics analysis of biological samples. Unfortunately, it suffers from various changes in the retention time (RT) of the same compound in different samples, and these must be subsequently corrected (aligned) during data processing. Classic alignment methods such as in the popular XCMS package often assume a single time-warping function for each sample. Thus, the potentially varying RT drift for compounds with different masses in a sample is neglected in these methods. Moreover, the systematic change in RT drift across run order is often not considered by alignment

algorithms. Therefore, these methods cannot effectively correct all misalignments. To utilize this information, we develop an integrated reference-free profile alignment method, neighbor-wise compound-specific Graphical Time Warping (ncGTW), that can detect misaligned features and align profiles by leveraging expected RT drift structures and compound-specific warping functions. Specifically, ncGTW uses individualized warping functions for different compounds and assigns constraint edges on warping functions of neighboring samples. We applied ncGTW to two large-scale metabolomics LC-MS datasets, which identifies many misaligned features and successfully realigns them. These features would otherwise be discarded or uncorrected using existing methods.

When the desired signal is buried in a mixture, deconvolution is needed to recover the pure sources. Many biological questions can be better addressed when the data is in the form of individual sources, instead of mixtures. Though there are some promising supervised deconvolution methods, when there is no a priori information, unsupervised deconvolution is still needed. Among current unsupervised methods, Convex Analysis of Mixtures (CAM) is the most theoretically solid and strongest performing one. However, there are some major limitations of this method. Most importantly, the overall time complexity can be very high, especially when analyzing a large dataset or a dataset with many sources. Also, since there are some stochastic and heuristic steps, the deconvolution result is not accurate enough. To address these problems, we redesigned the modules of CAM. In the feature clustering step, we propose a clustering method, radius-fixed clustering, which could not only control the space size of the cluster, but also find out the outliers simultaneously. Therefore, the disadvantages of K-means clustering, such as instability and the need of cluster number are avoided. Moreover, when identifying the convex hull, we replace Quickhull with linear programming, which decreases the computation time significantly. To avoid the not only heuristic but also approximated step in optimal simplex identification, we propose a greedy search strategy instead. The experimental results demonstrate the vast improvement of computation time. The accuracy of the deconvolution is also shown to be higher than the original CAM.

Machine Learning Approaches for Modeling and Correction of Confounding Effects in Complex Biological Data

Chiung-Ting Wu

ABSTRACT (General Audience)

Due to the complexity of biological data, there are two major pre-processing steps: alignment and deconvolution. The alignment step corrects the time and location related data acquisition distortion by aligning the detected signals to a reference signal. Though many alignment methods are proposed for biological data, most of them fail to consider the relationships among samples carefully. This piece of structure information can help alignment when the data is noisy and/or irregular. To utilize this information, we develop a new method, Neighbor-wise Compound-specific Graphical Time Warping (ncGTW), inspired by graph theory. This new alignment method not only utilizes the structural information but also provides a reference-free solution. We show that the performance of our new method is better than other methods in both simulations and real datasets.

When the signal is from a mixture, deconvolution is needed to recover the pure sources. Many biological questions can be better addressed when the data is in the form of single sources, instead of mixtures. There is a classic unsupervised deconvolution method: Convex Analysis of Mixtures (CAM). However, there are some limitations of this method. For example, the time complexity of some steps is very high. Thus, when facing a large dataset or a dataset with many sources, the computation time would be extremely long. Also, since there are some stochastic and heuristic steps, the deconvolution result may be not accurate enough. We improved CAM and the experimental results show that the speed and accuracy of the deconvolution is significantly improved.

Acknowledgment

First, it was my pleasure working in the Computation Bioinformatics & Bio-imaging Laboratory (CBIL). I would like to express my foremost thanks and deep gratitude to my advisor, Dr. Yue Wang, for his constant support and encouragement throughout my Ph.D. journey. His sharp insights and profound knowledge helped me at various stages of my research. He is not only a great academic advisor but also a mentor and friend outside of the academic world. His caring attitude and words of wisdom have formed the finest part of my memories at Virginia Tech.

I would also like to extend my sincere gratitude to Dr. Guoqiang Yu, another very important mentor in CBIL. Dr. Yu's professional knowledge never fails us. The insightful suggestions from him always inspire me to view the problem from the most suitable perspective. Moreover, he is very willing to answer all my questions. No matter how busy he is, he always tries to find out some time to give me invaluable advices. Without Dr. Wang or Dr. Yu, I might have many more burdens during my Ph.D. journey.

Also, I am grateful to Dr. William Baumann, Dr. Thidapat Chantem, and Dr. Chang-Tien Lu, who have generously served on my dissertation committee. Their insightful questions and helpful comments have helped a lot. I would also like to thank all of our collaborators at Wake Forest University, Cedars-Sinai Heart Institute, Imperial College London, Johns Hopkins Medical School, and State University of New York Upstate Medical University for their generous help in the projects and publications.

My labmates at CBIL are also very important to me. Our group meetings and offline discussions were the sources of my inspirations. Though our research topics are not exactly the same, the points of view from different aspects are very precious. In particular, I want to thank Lulu Chen, Yizhi Wang, Yinxue Wang, Congchao Wang, Zuolin Cheng, Yingzhou Lu, and Minjie Shen; their contributions to my dissertation are greatly appreciated.

Last but not least, I would like to thank my parents and sisters for their unconditional love and for always being there for me. Their immense patience and silent support have helped me scale the peaks I have never imagined.

Table of Contents

List of figures

Chapter 1

Background and Introduction

1.1 Motivation

Recently, the advances in the biotechnologies and bioinformatics promote the importance of omics data in the biomedical field. Nowadays, doing research on the function of components on DNA, on the biological mechanism of disease, or on the health status of human beings, is nearly impossible without omics data. To give a brief introduction of omics data, we can follow the central dogma of molecular biology [1]. The information flow of an organism starts from DNA/gene, to RNA, protein, and metabolite, as shown in **Figure 1.1**. Genomics is the study of the functions of genomes [2], while transcriptomics focuses on RNA [3]. As the product of gene expression, protein is the subject of research in proteomics [4]. Metabolomics studies metabolites, which is the end product [5]. DNA, the genetic material and the source of information, leads to RNA during transcription. In the subsequent translation step, RNA is decoded to protein. Some proteins participate the chemical reactions of metabolism, which generates metabolites as intermediate or final products. Though DNA encodes the genetic information, we cannot predict all phenomena of the organism only with genomics data, since the biological phenotypes at different stages are also affected by environmental factors [6]. Thus, to fully understand the mechanisms in the organism, different stages of the information flow, in other words, different types of omics data, are all important and complementary.

To collect omics data, people developed various techniques to extract information from different sources. For the DNA and RNA data, microarray-type methods are widely used [7], which can measure the expression level of thousands of genes each time. In recent years next-generation sequencing methods have achieved unprecedented accuracy, cost and throughput [8, 9]. In general, these methods detect the nucleotides in the fragmented DNA/RNA, and then align the detections to decide which parts of DNA/RNA they belong to respectively. On the other hand, proteomics and metabolomics studies utilize mass spectrometry [4, 10] or nuclear magnetic resonance spectroscopy (NMR spectroscopy) [11, 12] to separate proteins/metabolites by their physical or

chemical properties. However, for mass spectrometry, there may be overlapping among the spectrum of some proteins/metabolites. To address the overlapping problem, gas chromatography or liquid chromatography can be combined with mass spectrometry to expand the 1D spectrum to the 2D spectrum, so that the impact of the overlapping problem decreases significantly. The two resulting methods, Liquid chromatography-mass spectrometry (LC-MS) [13] and Gas chromatography-mass spectrometry (GC-MS) [14] are important and popular techniques in proteomics/metabolomics.

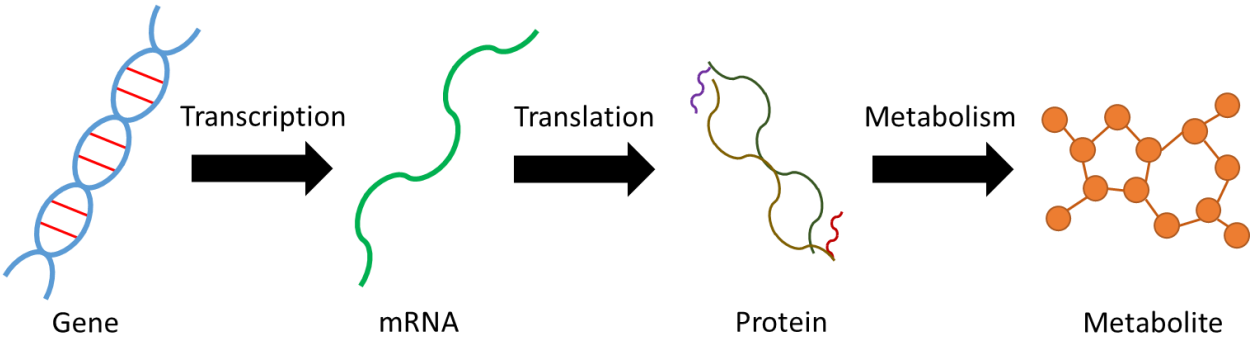

Figure 1.1 Information flow in biological systems by the central dogma of molecular biology. Roughly speaking, the information transfers from DNA to RNA by transcription, then from RNA to protein by translation, and from protein to metabolite by metabolism. The information flow may be affected by other factors, so genomic, proteomics, and metabolomics are all important and complementary for fully understanding the biological mechanism of the organism.

As different types of omics data give us complementary information for research, people developed many different techniques to detect various signals from different types of samples. Each of these different methods has its own technical flaws which make the detection not as accurate as we wished. Likewise, these omics data also have their innate properties which may block us from obtaining the real signal, so that mining information from detected signals can be a non trivial task. Considering the unavoidable technical flaws and the interference from the innate properties of biological samples, to obtain the clean data for further analysis, preprocessing is always needed. Generally, the motivation of preprocessing is to remove all the confounding factors. However, removing all confounding factors is a very hard problem, especially for unknown ones.

Thus, any preprocessing procedure for any type of data is trying instead to remove as many confounding factors as possible.

In general, as shown in **Figure 1.2**, there are at least two important steps in bioinformatics data pre-processing: alignment and deconvolution. For example, after signal detection, the data of DNA and RNA may need to be aligned to a reference to identify which parts of DNA/RNA are detected [15]. In addition, protein data and metabolite data also need the alignment step, in which the same compounds are aligned together across samples by adjusting their detection time (LC-MS) [16] or frequency (NMR) [17].

There are two different types of deconvolution for omics data. The first one is the deconvolution of the compound spectrum, which solves the overlapping problem of the spectrum of different compounds. This kind of overlapping often happens in NMR data as the spectrum is 1D. That is, if we have the similar number of peaks, it is natural that the overlapping will happen more frequently on the 1D spectrum than 2D. The second type is about tissue heterogeneity [18], which happens when the sample is a mixture of different tissues / cell types. For example, in the gene expression data, it is hard for us to separate different types of cells when measuring their expression level. Thus, the data we obtain is an expression mixture of several different cell types. If we are only interested in the gene expressions of some certain cell types, deconvolution is needed to extract such information.

After alignment and deconvolution, the data has almost no confounding factors, and ready for further analysis. For example, we can detect marker based on pure sources. However, these confounding factors sometimes may also contain important information. For example, in the tissue heterogeneity problem, the confounding factor is the portion of each pure cell type in the mixture. Nevertheless, the changing of the portions across samples may also be an important observation in the time series data [19]. Thus, after obtaining the confounding factors, we are not going to "drop" them. Instead, we are trying to detect the hidden information from them, "separate" them from the data, and keep them for further analysis if need.

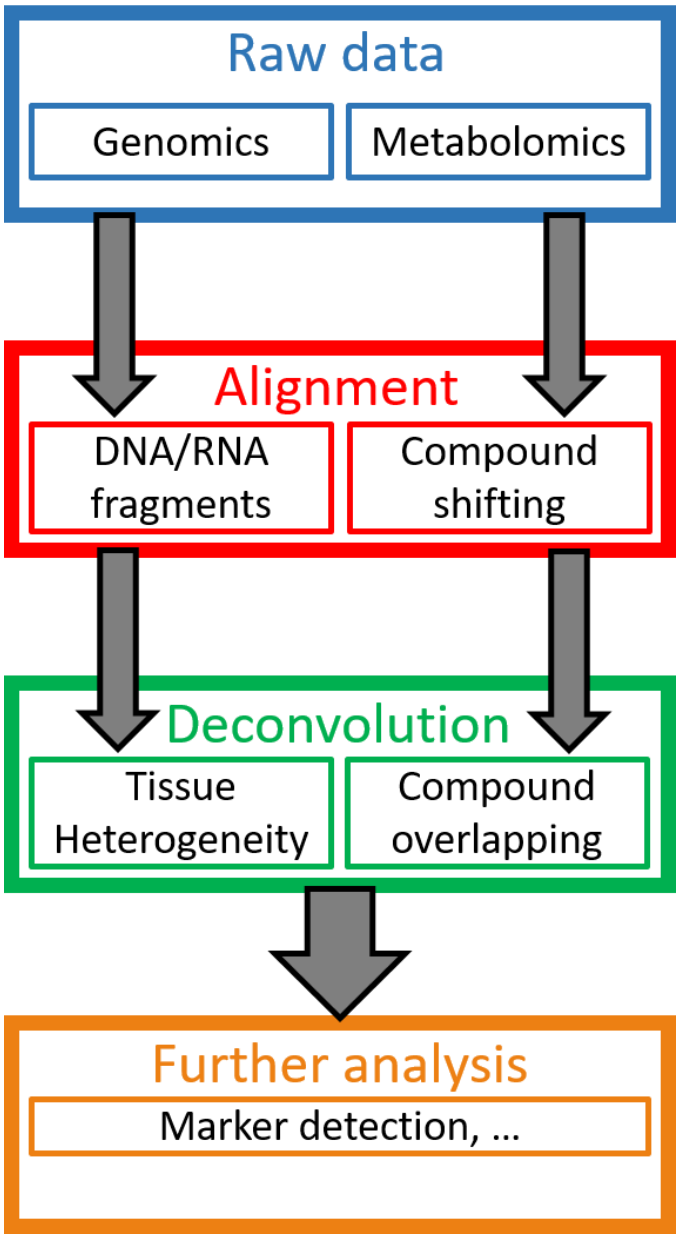

Figure 1.2 Two important steps in preprocessing for correcting the confounding factors. The genomics and metabolomics data are picked as examples. Most of the omics data need these two steps in the preprocessing.

1.1.1 Technical and biological background

Due to the technical limitation, in metabolomics research, people often use both NMR and LC-MS to acquire data [20]. However, the detected signanls from both techniques require alignment. For example, LC-MS always suffers from the "retention time drift" problem, which happens on

one dimension of the LC-MS data. In more detail, LC-MS is composed of liquid chromatography (LC) and mass spectrometry (MS). That is, LC-MS combines the physical separation capability and the mass analysis capability, so that the detected signal is like a 2D matrix. The first dimension is called retention time (RT), which represents when the detected signal comes out from the LC part. The second dimension is m/z, which is the ratio of the mass to charge of the compound. The whole process of LC-MS is as follows: first, inject the sample into the mobile phase stream. When the mixture goes through the LC column, the speed of the compounds is different, according to their physical and chemical properties. Thus, the compound will be separated by their RT. Next, the compounds will be ionized in the MS part, and will then be further separated by their m/z.

To compare the quantity of the same compound across samples, the straightforward idea is to look for the intensity of the signals with the same RT and m/z. However, the RT of the same compound is not stable across samples, as shown in **Figure 1.3**, since the interactions in the LC column may be affected by several factors, such as the room temperature changing [21]. Thus, when conducting a large study, which may include thousands of samples, the total data acquisition time may be more than one month. Therefore, it is hard to make sure all the factors are the same during such long-time duration. Hence, RT drifting is the major problem in the LC-MS data preprocessing. Comparing with RT, the m/z is relatively much more stable, so we can consider the RT drifting as a 1D-alignment problem [22]. By the way, because the separation of compounds is 2D in LC-MS, the overlapping problem rarely happens, so deconvolution is not needed in general.

Classic alignment methods often assume a single RT alignment function for each sample across different m/z [16]. Thus, the potentially varying RT drift for compounds with different m/z in a sample is neglected in these methods. Moreover, the systematic change in RT drift across run order is often not considered by alignment algorithms. Therefore, these methods cannot effectively correct all misalignments. For a large-scale experiment involving many samples, the existence of misalignment becomes inevitable.

Different from metabolomics data, the major problem of RNA data is the complex heterogeneity [23]. Though the high-throughput measurement technologies can detect thousands or even more RNAs simultaneously, most of them are global profiling methods [24]. For example, one of the most popular method of measuring gene expression level is RNA sequencing (RNA-seq) [25], and

one type of most common sample for RNA-seq is biopsy. Since a biopsy is usually obtained by human from a living organism, it is hard to guarantee that the biopsy contains only one type of tissue or cell. Thus, the estimated RNA quantity from RNA-seq is usually a "mixture", so the signal from the targeted cell type may be severely interfered. Therefore, it is crucial to find a way to "purify" the mixture.

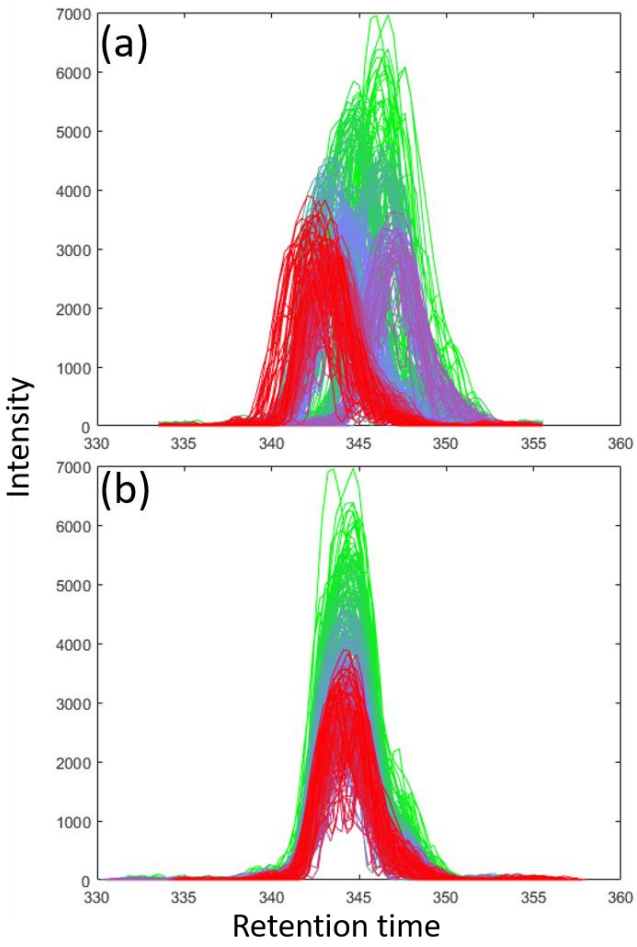

Figure 1.3 Examples of retention time drift. (a) The spectrum of LC-MS at a certain m/z value shows the problem of retention time drift. The peaks are not at the same RT before alignment. (b) After alignment, the peaks are aligned together, so that we can make sure these peaks corresponding to the same compound.

Though there are some methods which can mitigate tissue heterogeneity physically, they are not sufficiently reliable and cost-effective and is inapplicable to previously-assayed samples [26]. For

example, cell sorting [27], tissue microdissection [28], and single cell sequencing [29] are either expensive, time-consuming, or may not accurately detect the RNA signal during isolation [30, 31]. There are some computational methods which could deconvolve the mixed signal with *a priori* information [32-34]. However, these methods are supervised and may fail if there is not any prior information or the quality of the prior information is low for some cell types in the mixture.

There are also some unsupervised deconvolution methods, known as blind source separation (BSS) algorithms. Though these methods do not need prior information of the cell types, they have strong limitations or unsuitable assumptions. For example, Non-negative independent component analysis (nICA) [35] assumes that the source signals (from each cell types) are statistically independent, but it is common that the gene expressions of different cell types are correlated. Non-negative matrix factorization (NMF) [36], on the other hand, only assumes that each source is non-negative. However, its solution may change with different initialization (local optimum problem). Thus, the result of NMF may not be biologically meaningful. Also, both methods need to know the number of the cell types before deconvolution, which is usually not available.

Recently, deconvolution by convex analysis of mixtures (debCAM) is proposed as a software package, which can perform completely unsupervised deconvolution [37]. This core algorithm in the package is convex analysis of mixtures (CAM), which can estimate the proportion of each cell type in the mixture by identifying the optimal simplex for the dataset [23]. With the vertices of the simplex, the mixing proportions can be obtained. These vertices are corresponding to the "markers" of each cell type, so the idea of CAM is biologically plausible. However, a key step in CAM requires solving a combinatorial optimization problem. When the source number is larger than 10, the computation time would be unfeasible. Also, CAM needs K-means clustering as a pre-processing step. As K-means clustering is sensitive to random initialization, the results of CAM may be different if we apply it on the same dataset multiple times.

1.2 Objectives

1.2.1 Multiple alignment

For the alignment step, as mentioned above, there are several classical methods available for LC-MS data. However, these methods conveniently assume a single RT alignment function across all

m/z bins and perform multiple alignment that neglects the run order of each sample. For a large-scale experiment involving many samples, misalignment for certain features becomes inevitable. Moreover, no existing analytics tool includes a systematic way to detect misalignment and thus some misalignment is often undetectable or uncorrected.

To address the critical problem of the absence of validated methods for misalignment detection and structure-aware alignment, we would like to develop an alignment method with the following properties:

1. Align multiple samples simultaneously (multiple alignment) with no certain reference to avoid picking a reference.
2. Work on the profile of the signal (profile-based), not only on the detected peaks (feature based), to avoid peak-missing problem.
3. Consider the expected RT drift structures to improve the alignment accuracy.
4. Detect the misaligned features to improve the efficiency of the profile-based method.

1.2.2 Unsupervised deconvolution

Though debCAM is the most biologically-plausible unsupervised deconvolution method now, its unstable pre-processing step and time-consuming problem should be solved for more applications and more accurate results. Thus, we would like to address the problems of debCAM in the following aspects:

1. Find a more suitable clustering method to replace K-means clustering as a stable pre-processing step.
2. Improve the speed of finding the convex hull, the initial step of identifying the optimal simplex, to make the computation time feasible.
3. Replace the heuristic combinatorial optimization problem solver in debCAM, so that the computation time may decrease and the result may be more accurate.

1.3 Statement of problems

1.3.1 Multiple alignment

One of the difficulties when aligning metabolomics data is that the signal may vary significantly, and some signal may be very weak or even disappear in some samples. Thus, the traditional alignment methods, especially the feature-based ones, do not work very well on metabolomics data. Here, we consider it as a classical profile-based alignment problem, but take account of the structure of the dataset. Though the signal may vary or miss, their shifting trend is correlated with the structure of the dataset, which can be viewed as the relationship among the data in the dataset. Also, we formulate a whole multiple alignment problem as a large graph to avoid picking a certain reference. Thus, we developed an integrated reference-free profile alignment method, neighbor-wise compound-specific Graphical Time Warping (ncGTW) [38], that can detect misaligned features and align profiles by leveraging expected RT drift structures and compound-specific warping functions. Specifically, ncGTW uses individualized warping functions for different compounds and assigns constraint edges on warping functions of neighboring samples.

1.3.2 Unsupervised deconvolution

The basic idea of CAM is that it considers the observed gene expression data and the underlying pure gene expressions from different cell types as $\mathbf{X} = \mathbf{AS}$, where $\mathbf{X}$ is the expression level of each samples, $\mathbf{A}$ is the mixing proportions of each cell type, and $\mathbf{S}$ is the pure expression level of each cell type.

To deconvolve $\mathbf{X}$, CAM will first estimate $\mathbf{A}$ by finding the vertices of the simplex of $\mathbf{X}$. Before finding the simplex, we need to find the convex hull first. However, with noise, the dimension of the convex hull would be high and the resulting vertices can be noisy. Thus, CAM will first perform K-means clustering or affinity propagation clustering (APC) to suppress the noise. However, the clustering by K-means is unstable, and the computation time of APC is very high. Moreover, both methods cannot control the size of the clusters, which is related to the size of the vertices in the following simplex identification.

After clustering, the next step is to find the convex hull on the clusters. Quickhull is one of the most popular algorithms and is used in debCAM [39]. However, in Quickhull, the time complexity with respect to dimension is exponential. Therefore, the computation time of this step would be huge when the number of clusters is more than 50.

Moreover, when identifying the optimal simplex, CAM will try all possible combinations of clusters in a heuristic way. Thus, computation cost would be dramatically high if the source number is more than 10, and the result may still be inaccurate.

To solve the above-mentioned problems, we improved CAM in the following aspects. First, we propose a radius-fixed clustering to make sure the size of each cluster is similar and controllable. Then, for the algorithm of finding the convex hull, we use a linear-programming based approach to replace Quickhull, so that the time complexity of dimension is polynomial. For the combinatorial optimization problem, we use greedy search to replace the heuristic search to obtain result faster. Interestingly, on average the results are even more accurate.

1.4 Outline of the dissertation

The remainder of this dissertation is organized as follows. We will first introduce a new alignment method, ncGTW, in Chapter 2. Chapter 3 will demonstrate the application of ncGTW on LC-MS data, and introduce the pre-process and post-process as an automatic and complete alignment package. In fact, ncGTW is motivated by metabolomics data, but it can be applied on any alignment problem. Chapter 4 will discuss each improved step of CAM. In Chapter 5, the applications and improvements of CAM on real datasets are demonstrated. Finally, the contributions and the future works are summarized in Chapter 6.

Chapter 2

Neighbor-wise Compound-specific Graphical Time Warping

2.1 Introduction

Alignment of multiple time series or sequences is a ubiquitous problem and critical task in solving many real-world problems, such as proteomics [40], speech recognition [41], and DNA sequence analysis [42]. Although the alignment of two samples can be effectively solved by dynamic time warping (DTW), most existing approaches based on DTW exploits separate application of DTW to a sequence of sample pairs. A typical strategy is so-called "progressive alignment" [43], which aligns a pair of samples sequentially according to a heuristic order, or to align all samples to an initial reference and iteratively update the reference. On the other hand, alternative methods have also been developed based on probability model [44] where a prototype function is assumed and all the samples are generated from it. In addition, there is a distinct class of methods based on features [45], while the feature extraction is challenging and often application-specific or even ill-posed. Nevertheless, this paper is focused on profile-based methods.

While the existing profile-based methods work reasonably well for applications with high signal-to-noise ratio and simple patterns, more sophisticated and principled approach is needed for complex signals with significant uncertainty. Progressive alignment is sensitive to the processing order of samples. Iteratively aligning all samples to a reference requires an accurate initial reference that can be compromised by noise or missing peaks. The probability methods depend on the assumption that the stochastic model can describe faithfully the data yet simple to infer.

Critically, all existing methods lack a unified scheme to incorporate structural information among neighboring samples. This kind of structural information is frequently available in real applications. For example, when a liquid chromatography-mass spectrometry (LC-MS) experiment is conducted, two consecutive samples are expected to have more synchronized time series than two samples separated by a large time interval, because the physical properties of the equipment are gradually

changing. Thus, the neighborhood structure of these samples can be represented by a line, which is ordered by the time that the experiment is conducted. Dataset containing multiple batches is another example, where we expect that samples from the same batch are more similar than the ones from different batches. Thus, the structure can be represented by multiple separated cliques.

Overlooking structural information could lead to a compromised alignment accuracy, as we will show in the paper. This problem is particularly severe when the data is noisy or has missing peaks. However, it was largely unknown how to incorporate structural information in a principled manner into simultaneous alignment of multiple time series. In this chapter, we develop a new approach, called neighbor-wise compound-specific Graphical Time Warping (ncGTW) [38], to address the need. In the scenario of complex patterns and significant noises, ncGTW reasons that no single sample could serve well as the only reference curve. Instead, avoiding selecting a single sample as the reference, ncGTW considers each sample as a reference to others and thus jointly explore the information embedded in all samples. However, an independent alignment of each pair of samples is not reliable due to the noise and complexity. An innovation of ncGTW is to use the neighborhood structure to impose a smoothness constraint to the pairwise alignment, which guides the final multiple alignment. In summary, ncGTW converts the multiple alignment problem to two staged sub-problems. In the first stage, we consider jointly all pairwise alignments by incorporating structure information. In the second stage, we align all samples to a pseudo-reference and use the pairwise alignment results in the first stage as constraints. The sub-problem in each stage can be effectively solved by network flow algorithm.

The proposed approach ncGTW is deterministic and requires no heuristics. In addition, ncGTW is flexible to model and explore any structural information among samples as long as the relationship can be described by a weighted graph. Next, we will describe in details the methods, report the experimental results, and discuss potential issues and alternative solutions. Our results demonstrate that ncGTW can effectively explore structural information to improve alignment accuracy. Interestingly, when there is no ‘informative’ structure, ncGTW can still achieve a better performance than the three representative peer methods tested.

2.2 Problem Modeling and Methods

Given N samples (curves) $\{x_1, \dots, x_N\}$, the multiple alignment problem aims to find a set of warping functions $\{\Phi_{i,c}\}, i \in \{1 \dots N\}$, through which each sample can be aligned to a reference x_c. For the sake of clarity, all samples have the same number of points P. The subscript "i, c" means this function maps a set of points $\{x_{ip}\}, p \in \{1 \dots P\}$ in curve x_i to a corresponding set of points in curve x_c. That is, for any point x_{ip} in x_i, $\Phi_{i,c}$ can always map x_{ip} to at least one point in x_c, where p is from 1 to P. The main challenge in multiple alignment is the lack of a priori reference curve. Indeed, various strategies have been proposed to select a reference sample or estimate a reference using all samples. However, when the samples have complex patterns such as missing signals at some time points, or contains significant noise, no single sample merits a good reference while estimation of an ensemble reference requires a set of pre-aligned samples. Our ncGTW avoids these difficulties by exploring the observation that if curve x_1 is similar to x_2, $\Phi_{1,c}$ and $\Phi_{2,c}$ should be similar. Specifically, given $\Phi_{1,2}$ can be utilized as a constraint for estimating $\Phi_{1,c}$ and $\Phi_{2,c}$. Next, to reliably estimate all pairwise alignments $\Phi_{i,j}$ and to mathematically incorporate all the associated constraints into final multiple alignment, we adapt the framework of network flow algorithms that can produce a global and efficient solution. Accordingly, reliable pairwise alignment is achieved by incorporating neighborhood smoothness constraints between pairwise warping functions, constituting a minimum cut problem; and the final multiple alignment is also modeled as a minimum cut problem in which additional edges in the graph is induced by the pairwise pre-alignments.

Definition 1 – valid warping function

A valid warping function for the pair of curves (x_i, x_j) is a set of integer pairs $\Phi_{i,j} = \{(p, q)\}$, such that the following conditions are satisfied: (a) boundary conditions: $(1,1) \in \Phi_{i,j}$ and $(P, P) \in \Phi_{i,j}$; (b) continuity and monotonicity conditions: if $(p, q) \in \Phi_{i,j}$, then $(p-1, q) \in \Phi_{i,j}$ or $(p, q-1) \in \Phi_{i,j}$ or $(p-1, q-1) \in \Phi_{i,j}$. An example is shown in **Figure 2.1a**.

Definition 2 – the inverse of a valid warping function

Given a valid warping function $\Phi_{i,j} = \{(p, q)\}$, the inverse of $\Phi_{i,j}$ is ${\Phi_{i,j}}^{-1} = \{(q, p)\} = \Phi_{j,i}$

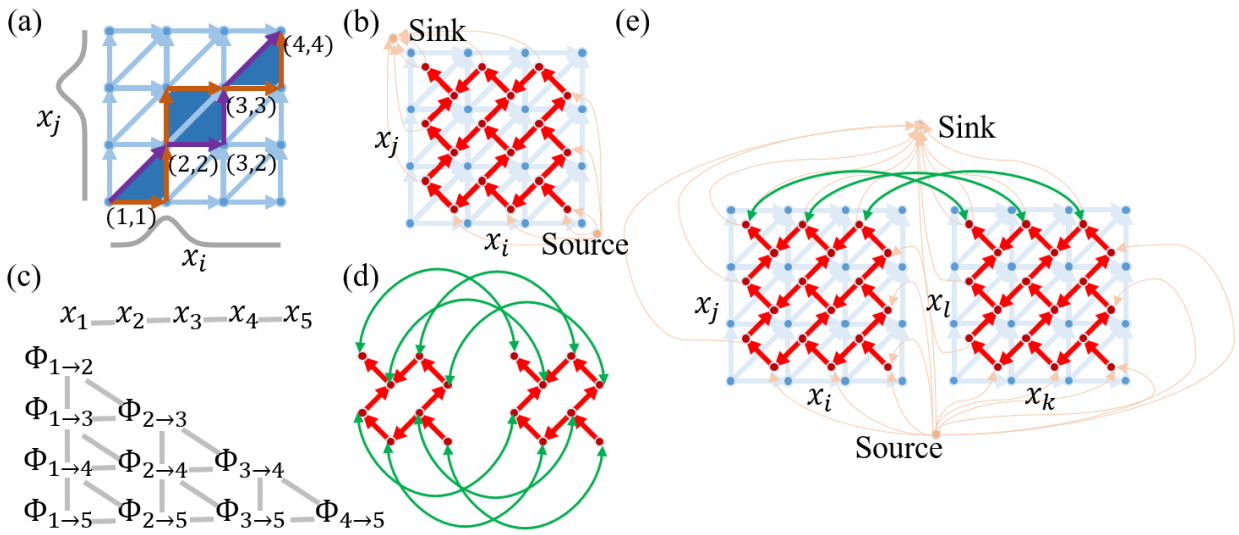

Figure 2.1 Figures of DTW grids, DTW graphs, structural information diagram for GTW, and GTW graph. (a) A DTW grid for aligning x_i to x_j. The purple path and the orange path correspond to two different warping functions. Each node in the grid corresponds to a pair of points, one from x_i and the other from x_j. For example, node (3, 2) corresponds to the third point on x_i (x_{i3}) and the second point on x_j (x_{j2}). The weight of an edge is determined by its starting node. For example, the weight of the edge ((3, 2), (3, 3)) is given by the distance between (x_{i3}, x_{j2}). The distance between the purple path and the orange path is defined as the area in dark blue (four triangles in this case). The corresponding warping function of the purple path is {(1, 1), (2, 2), (3, 2), (3, 3), (4, 4)}. (b) A DTW grid and the corresponding DTW graph. The blue lines and dots form the original DTW grid, and the red and orange lines and dots form the corresponding DTW graph. Note here orange lines link only the vertices (those red dots enclosed by the blue-lined exterior triangle) to a single source or sink. (c) An example of structural information between samples and the induced structural information between pairwise warping functions. Suppose there are five LC-MS samples and the run orders of which are exactly 1, 2, 3, 4, and 5. These samples are expected to have continuous changing profiles from smaller indices to larger indices. More importantly, the continuous changing gives rise to the similarities between warping function. For example, the warping function for curve pair (x_1, x_2) is similar to the warping function for (x_2, x_3). In general, curve pairs (x_i, x_{i+1}) and (x_{i+1}, x_{i+2}) are considered as neighbors. Similarly, curve pairs (x_i, x_j) and (x_i, x_{j+1}), along with curve pairs (x_i, x_j) and (x_{i+1}, x_j) are also neighbors. The diagram shows all warping function neighboring information. (d) Two small DTW graphs

connected together. Green lines are the additional edges linking the corresponding vertices of the DTW graphs. (e) A GTW graph formed by two linked DTW graphs. Two warping functions ($\Phi_{i,j}$ and $\Phi_{k,l}$) are neighbors. Orange edges connect vertices to source and sink. Green edges link the corresponding vertices in two graphs. For clarity, we only show links between the top three vertices. Other vertices are linked in the same way.

Definition 3 – alignment cost

For any given valid warping function $\Phi_{i,j}$ and its corresponding pair of curves (x_i, x_j), the associated alignment cost is defined as follows:

$$\text{cost}(\Phi_{i,j}) = \sum_{(p,q)\in\Phi_{i,j}} g(\mathrm{x}_{ip}, \mathrm{x}_{\mathrm{j}q}), \tag{2.1}$$

where $g(\mathrm{x}_{ip}, \mathrm{x}_{\mathrm{j}q})$ is any nonnegative function which computes the distance between x_{ip} and $\mathrm{x}_{\mathrm{j}q}$.

The flowchart of ncGTW is shown in Figure 2.1. We first find warping functions $\{\Phi_{i,j}\}$ to utilize the structural prior knowledge. Then we use this set of warping functions as constraints to estimate $\{\Phi_{i,c}\}$. These two subproblems are formulated and solved in details as the following.

2.2.1 Stage 1: jointly aligning all pairs with the structural prior incorporated

To estimate $\{\Phi_{i,j}\}$ jointly, ncGTW considers all possible sample pairs. For each pair, one sample is set as the reference. In other words, in the first stage, ncGTW tries to align each sample to the other samples. For N samples, the number of alignment pairs is $N(N-1)$. However, $\Phi_{i,j}$ is the inverse of $\Phi_{j,i}$, so only $N(N-1)/2$ pairs need to be considered in the real implement. That is, only $N(N-1)/2$ pairwise warping functions are needed.

In order to incorporate the structural information, as GTW, we need to convert the given structural information in the dataset into the warping function neighborhood information. Suppose (x_i, x_j) are neighbors by structural information, we consider the pair of warping functions, $(\Phi_{i,k}, \Phi_{j,k})$, as well as $(\Phi_{k,i}, \Phi_{k,j})$, as neighbors for all k. Also, if (x_i, x_j) are neighbors and (x_k, x_l) are also neighbors respectively, we can consider the pair of $\Phi_{i,k}$ and $\Phi_{j,l}$ are neighbors. **Figure 2.1c** gives an example of the warping function neighborhood information of five samples. The run orders of the five samples are exactly 1, 2, 3, 4, and 5. Thus, these five samples are expected to have a pattern

of continuous changing, so (x_1, x_2), (x_2, x_3), (x_3, x_4), and (x_4, x_5) are considered neighbors respectively.

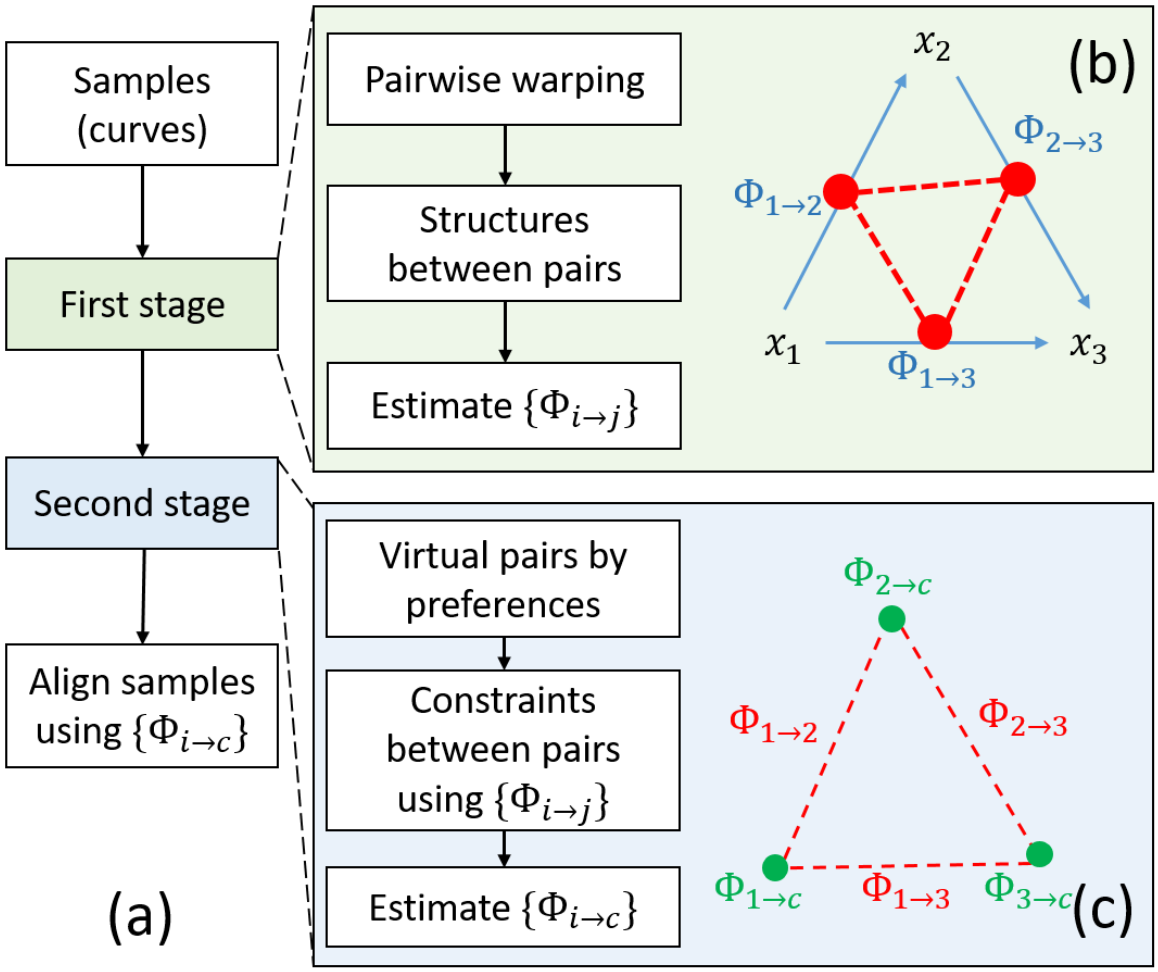

Figure 2.2 Flowchart of ncGTW algorithm. (a) With two-stage alignment strategy, all input samples (curves) are aligned simultaneously to a virtual reference. (b) Stage 1 of ncGTW with three illustrative samples. First, ncGTW builds a pairwise warping flow map (blue arrows). Then ncGTW incorporates structural information as the constraint and applies to all pairs (pair as red dot and constraint as a red dashed line). Lastly, ncGTW estimates all pairwise warping functions ($\Phi_{i,j}$) jointly with e.g., smoothness constraint on neighboring sample pairs. **(c)** Stage 2 of ncGTW with three illustrative samples. ncGTW aligns every sample to a common virtual reference x_c, where the warping functions $\{\Phi_{i,j}\}$ obtained in Stage 1 provide warping correspondences and final warping functions $\{\Phi_{i,c}\}$ are calculated by solving the maximum flow problem.

Definition 4 – neighboring warping functions

Suppose the neighboring structure for a set of M valid warping functions is given by the graph $G_{struct} = \{V_s, E_s\}$, where V_s is the set of nodes, with each node corresponding to a warping function, and E_s is the set of undirected edges between nodes. If $v_{ij}, v_{kl} \in V_s$ and $(v_{ij}, v_{kl}) \in E_s$, we call $\Phi_{i,j}$ and $\Phi_{k,l}$ neighbors, denoted by $((i,j),(k,l)) \in Neib$.

As a profile-based method, ncGTW adapts the idea of DTW. DTW aligns two curves x_i and x_j by finding a warping function $\Phi_{i,j}$ that minimize the alignment cost (equation 2.1). The correspondence represented by $\Phi_{i,j}$ can be visualized as a path in a DTW grid, from bottom left to top right, and the weight of each edge is decided by the distance function $g(\cdot)$ with all possible point pairs (x_{ip}, x_{jq}), where $x_{ip} \in x_i$ and $x_{jq} \in x_j$. DTW estimates Φ in the DTW grid using dynamic programming and various additional constraints can be employed, such as the direction of the path.

Definition 5 – DTW grid for a single pair of curves

For each pair of curves, consistent with the cost function (equation 2.1), there is an induced directed planar graph [46], $G_{ij} \coloneqq \{V_{ij}, E_{ij}\}, 1 \le i < j \le N$, where V_{ij} and E_{ij} are the nodes and directed edges respectively. Each point pair (x_{ip}, x_{jq}) is corresponding to a node $V_{ij,pq} \in V_{ij}$, where $1 \le p, q \le P$. The weight of $(V_{ij,p_1q_1}, V_{ij,p_2q_2}) \in E_{ij}$ is the distance between the two points (x_{ip_1}, x_{jq_1}), measured by $g(x_{ip_1}, x_{jq_1})$. An example is shown in **Figure 2.1a**. Any directed path from the bottom-left corner to the upper-right corner is corresponding to a valid warping function $\Phi_{i,j}$.

Once the structure for warping functions is obtained, the joint alignment of all pairs of samples can be readily solved by the recently developed model – Graphical Time Warping (GTW) [47]. When we jointly align multiple pairs of curves with structural information, our goal is to minimize both the overall alignment cost and the distance between neighboring warping functions.

Definition 6 – the distance between two valid warping functions

For any two given valid warping functions $\Phi_{i,j}$ and $\Phi_{k,l}$, the distance between them is defined as follows:

$$\mathrm{dist}\left(\Phi_{i,j}, \Phi_{k,l}\right) = \frac{1}{2} \sum_{1 \le n \le P} \left| \max_{(p_i,n) \in \Phi_{i,j}} p_i - \max_{(p_k,n) \in \Phi_{k,l}} p_k \right| + \left| \min_{(p_i,n) \in \Phi_{i,j}} p_i - \min_{(p_k,n) \in \Phi_{k,l}} p_k \right|, \qquad (2.2)$$

which is equivalent to the area of the region bounded by the two corresponding paths in a DTW grid as shown in **Figure 2.1a**.

Mathematically, to balance the alignment cost (equation 2.1) and the distance between warping functions (equation 2.2), we want to minimize the following cost function for GTW problem:

$$\min_{\Phi} f\left(\Phi\right) = \min_{\Phi = \{\Phi_{i,j} | 1 \le i < j \le N\}} \sum_{1 \le i < j \le N} \mathrm{cost}\left(\Phi_{i,j}\right) + \kappa_1 \sum_{((\mathrm{i,j}),(\mathrm{k,l})) \in \mathrm{Neib}} \mathrm{dist}\left(\Phi_{i,j}, \Phi_{k,l}\right), \qquad (2.3)$$

where κ_1 is the parameter which balances the two terms. The first term is the overall alignment cost of the warping functions. The second term could be considered as the sum of the "dissimilarity" between each pair of the neighboring warping functions. Again, the neighboring warping functions should be similar. In other words, their distance should be small.

There are three major steps to solve the GTW problem. Firstly, GTW transforms the DTW grid for each pair of curves to an equivalent minimum cut problem as a DTW graph. Supposing there are *M* pairs of curves, where each pair contains two curves to be aligned with each other, then we will have *M* DTW grids, of which there are also *M* DTW graphs. Secondly, GTW adds in extra edges between DTW graphs, if two pairs of curves are considered as neighbors. Note that if edges are to be added between two DTW graphs, all the corresponding vertices in the two graphs need to be connected. Thus, the *M* separate DTW graphs become an extended graph (GTW graph). Thirdly, GTW proved that the joint alignment of multiple pairs with smoothness constraints imposed could be formulated as a minimum cut problem in the GTW graph. Hence, efficient network flow algorithms can be used to find a global optimal solution.

Definition 7 – DTW graph

Define $G'_{ij} \coloneqq \{V'_{ij}, E'_{ij}\}$ as the DTW graph of the DTW grid G_{ij}, where nodes V'_{ij} are all faces of G_{ij}. That is, for each $V_{ij,pq} \in V_{ij}$, where $2 \le p \le P-1$ and $1 \le q \le P-1$, there are two corresponding nodes $V'_{ij,pq+}$ and $V'_{ij,pq-}$; for each $V_{ij,pq}$ where $p = 1$ and $1 \le q \le P-1$, there is one corresponding node $V'_{ij,pq+}$; for each $V_{ij,pq}$ where $p = P$ and $1 \le q \le P-1$, there is one

corresponding node $V'_{ij,pq-}$. For each $e \in E_{ij}$, we have a new edge $e' \in E'_{ij}$ connecting the faces from the right side of e to the left side. This edge is directed (with positive direction by convention). The edge weights are the same as for the primal graph G_{ij}. An example is shown in **Figure 2.1b**.

Definition 8 – GTW graph

The GTW graph $G_{gtw} \coloneqq \{V_{gtw}, E_{gtw}\}$ is defined as the *integrated graph* of all DTW graphs $\{G'_{ij} | 1 \leq i < j \leq N\}$ with the integration guided by the neighborhood of warping functions, such that $V_{gtw} = \{V'_{ij} | 1 \leq i < j \leq N\}$ and

$$E_{gtw} = E'_{ij}\big|1 \leq i < j \leq N \cup \{\left(V'_{ij,pq+}, V'_{kl,pq+}\right), \left(V'_{ij,pq-}, V'_{kl,pq-}\right)\}\big|((\mathrm{i,j}),(\mathrm{k,l})) \in Neib.$$

All newly introduced edges $\left(V'_{ij,pq+}, V'_{kl,pq+}\right)$ and $\left(V'_{ij,pq-}, V'_{kl,pq-}\right)$ are bi-directional with capacity λ_1 as shown in **Figure 2.1d**. An example of a GTW graph with two pairs of curves is shown in **Figure 2.1e**.

Definition 9 – **Labeling of the graph**

L is a *labeling* of graph G if it assigns each node in G a binary label. L can induce a cut set $C = \{(s,t) | L(s) \neq L(t), (s,t) \in E_G\}$. The corresponding cut (or flow) is $cut(L) = cut(C) = \sum_{(s,t)\in C} weight(s,t)$, where $weight(s,t)$ is the weight on the edge between nodes s and t.

Based on its construction, a labeling L for the graph G_{gtw} can be written as $L = \{L_{ij} | 1 \leq i < j \leq N\}$, where L_{ij} is a labeling for the DTW graph G'_{ij}. Thus, we can express the minimum cut problem for the graph G_{gtw} as:

$$\min_{L} g(L) = \min_{L \coloneqq \{L_{ij} | 1 \leq i < j \leq N\}} \sum_{1 \leq i < j \leq N} cut\left(L_{ij}\right) + \lambda_1 \sum_{((i,j),(k,l)) \in Neib} cut\left(L_{ij}, L_{kl}\right), \qquad (2.4)$$

where $cut\left(L_{ij}\right)$ is the cut of all edges for G'_{ij} and $cut\left(L_{ij}, L_{kl}\right)$ is the number of the cut edges between two neighboring DTW graphs G'_{ij} and G'_{kl}.

As proved in [47], the GTW problem as stated in (equation 2.3) is equivalent to the minimum cut problem on the GTW graph G_{gtw} if we set $\lambda_1 = 2\kappa_1$. Once we apply any existing network flow

algorithm to solve this minimum cut problem, the GTW problem would be solved, and all pairwise warping functions are available.

2.2.2 Stage 2: Finding multiple alignment based on the constraint of pairwise alignments

In stage 2 of ncGTW, the result from stage 1, $\{\Phi_{i,j}\}$, is used to estimate $\{\Phi_{i,c}\}$, which are the final goal of multiple alignment problem. Here, *N* warping functions need to be estimated. On contrary to stage 1, where neighboring information as constraints, in stage 2 the warping function $\{\Phi_{i,j}\}$ are set as constraints to estimate $\{\Phi_{i,c}\}$. For example, according to the warping function $\Phi_{i,j}$, point x_{ip} in x_i should be aligned to point x_{jq} in x_j. Then, we expect that x_{ip} and x_{jq} should be aligned to the same position on the reference x_c. That is, we want the warping of x_{ip} and x_{jq} on the reference is *consistent*.

Definition 10 – inconsistency between two sample points

For any given integer pair $(p_i, p_j) \in \Phi_{i,j}$, where p_i is the point index of x_i and p_j is the point index of x_j. The inconsistency between x_{ip_i} and x_{jp_j} is defined as follows:

$$\text{incons}\left(x_{ip_i}, x_{jp_j}; \Phi_{i,j}\right) = \left| \max_{(p_i,q_i)\in\Phi_{i,c}} q_i - \max_{(p_j,q_j)\in\Phi_{j,c}} q_j \right| + \left| \min_{(p_i,q_i)\in\Phi_{i,c}} q_i - \min_{(p_j,q_j)\in\Phi_{j,c}} q_j \right|,$$

which could be considered as the distance between two continuous integer sets, as shown in **Figure 2.3a**. The inconsistency quantifies how much the alignment deviates from our expectation. If the inconsistency is zero, we call these two points are consistent.

Definition 11 – inconsistency between two warping functions

For any given valid warping function $\Phi_{i,j}$, the inconsistency between two warping functions $\Phi_{i,c}$ and $\Phi_{j,c}$ is defined as follows:

$$\text{incons}\left(\Phi_{i,c}, \Phi_{j,c}; \Phi_{i,j}\right) = \sum_{(p_i,p_j)\in\Phi_{i,j}} \left| \max_{(p_i,q_i)\in\Phi_{i,c}} q_i - \max_{(p_j,q_j)\in\Phi_{j,c}} q_j \right| + \left| \min_{(p_i,q_i)\in\Phi_{i,c}} q_i - \min_{(p_j,q_j)\in\Phi_{j,c}} q_j \right|, \quad (2.5)$$

which is equivalent to the sum of the inconsistency between two sample points x_{ip_i} and x_{jp_j}, where $(p_i, p_j) \in \Phi_{i,j}$. Likewise, if the inconsistency between these two warping functions is zero, then the two warping functions are consistent.

Intuitively speaking, stage 2 tries to identify the final warping functions from all pairwise ones, with the constraint that the sum of inconsistency between all warping function pairs as low as possible. However, due to the noise and other factors, in the real data, there are always some contradictions among the pairwise warping functions. For example, from $\Phi_{i,j}$, we know that x_{ip} and x_{jq} are aligned together, and from $\Phi_{j,k}$, we know that x_{jq} and x_{kr} are aligned together, but from $\Phi_{k,i}$, we know that x_{kr} and x_{io} (not x_{ip}) are aligned together. This kind of contradictions will make the alignment tend to be the trivial "all to one" mapping, if we want to minimize the inconsistency. To solve this problem, here we introduce another constraint that makes the warping functions tend to choose "one to one" mapping, to avoid the trivial mapping. In the real implement, we redesign the weight of the edges in the DTW graphs. Thus, the exact values of neither x_i nor x_c do no matter in this stage. In other words, the weights of the edges on the DTW graph of x_i and x_c is not based on the points of x_i and x_c. This property is the reason why our approach does not rely on the selection or estimate of the reference curve. Therefore, x_c is called a "virtual reference".

Definition 12 – non-diagonality of a warping function

For any given valid warping function $\Phi_{i,c}$, the associated non-diagonality is defined as follows:

$$\text{nondiag}(\Phi_{i,c}) = \sum_{(p,q)\in\Phi_{i,c}} \mathbb{1}\Big((p-1,q) \in \Phi_{i,c} \vee (p,q-1) \in \Phi_{i,c}\Big), \tag{2.6}$$

where $\mathbb{1}(\cdot)$ is the indicator function. This definition is the same as the total number of vertical and horizontal edges in the corresponding path. The smallest value of non-diagonality is zero (one-to-one mapping, the diagonal line). An example of non-diagonality is shown in **Figure 2.3a**.

In order to obtain the consensus final alignment from pairwise warping functions, we design ncGTW problem which tries to balance the non-diagonality (S12) and inconsistency (S11) among final warping functions:

$$\min_{\Phi} f(\Phi) = \min_{\Phi=\{\Phi_{i,c}|1\le i\le N\}} \sum_{1\le i\le N} \text{nondiag}(\Phi_{i,c}) + \kappa_2 \sum_{1\le i<j\le N} \text{incons}(\Phi_{i,c}, \Phi_{j,c}; \Phi_{i,j}), \qquad (2.7)$$

where the first term relates to the warping path for each final warping function, and the second term is the constraints between each warping function pair. The relation between these two terms is similar to the two terms in the GTW problem (equation 2.3). Thus, as same as GTW, an ncGTW problem could also be transformed into an ncGTW graph, and solved by maximum flow algorithms.

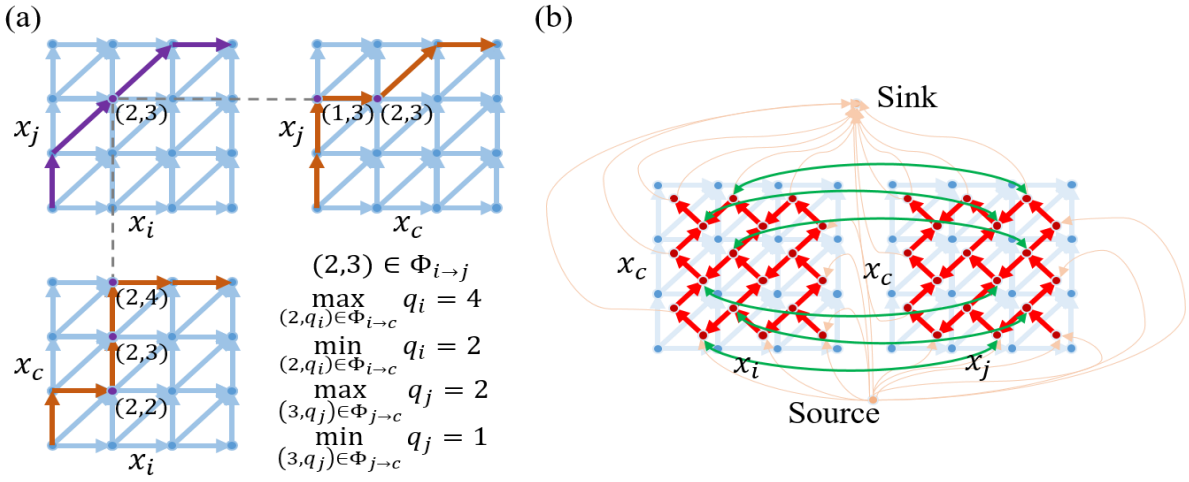

Figure 2.3 The illustration of Stage 2 of ncGTW. (a) Example of inconsistency and non-diagonality calculation. From the upper-left DTW grid, to achieve consistency, the 2nd point on x_i and the 3rd point on x_j should be aligned to the same position on the virtual reference, because $(2,3) \in \Phi_{i,j}$. It is clearly not the case by looking at the other two DTW grids. From $\Phi_{i,c}$ (lower-left DTW grid), we know that the 2nd point on x_i is aligned to points 2, 3, and 4 on the reference. From $\Phi_{c,j}$ (upper-right DTW grid, the inverse of $\Phi_{j,c}$), we know that the 3rd point on x_j is aligned to points 1 and 2 on the reference. Thus, by definition the inconsistency from (2, 3) is $|4-2| + |2-1| = 3$. The total inconsistency is calculated along all nodes in $\Phi_{i,j}$. The non-diagonality of $\Phi_{i,c}$ is 6, since there are totally 6 corresponding vertical and horizontal paths in the DTW grid. The non-diagonality of $\Phi_{i,j}$ is 2. (b) The graph of stage 2 of ncGTW. From stage 1, we have all pairwise warping functions. If from the warping function $\Phi_{i,j}$, we know that the 2nd point in x_i is aligned to the 3rd point in x_j, this graph shows how to link the related vertices, where x_c is the virtual reference.

Definition 13 – ncGTW graph

The ncGTW graph $G_{ncgtw} \coloneqq \{V_{ncgtw}, E_{ncgtw}\}$ is defined as the *integrated graph* of all DTW graphs $\{G'_{ic} | 1 \leq i \leq N\}$ with the integration guided by the pairwise warping functions, such that $V_{ncgtw} = \{V'_{ic} | 1 \leq i \leq N\}$ and $E_{ncgtw} = \{E'_{ic} | 1 \leq i \leq N \cup \left(V'_{ic,p_iq+}, V'_{jc,p_jq+}\right) | (p_i, p_j) \in \Phi_{i,j} \cup \left(V'_{ic,p_iq-}, V'_{jc,p_jq-}\right) | (p_i, p_j) \in \Phi_{i,j}\}$. That is, all newly introduced edges are guided by $\Phi_{i,j}$ as shown in **Figure 2.3b**. Also, all these new edges are bi-directional with capacity λ_2. For the edges in V'_{ic}, the capacity of edges corresponding to the vertical or horizontal path is one, and zero for edges corresponding to a diagonal path, so that the non-diagonality of the warping function $\Phi_{i,c}$ is equivalent to the cost of the warping path on the corresponding DTW grid G_{ic}.

Like GTW problem, the ncGTW problem as stated in equation (2.7) is equivalent to the minimum cut problem on the ncGTW graph G_{ncgtw} if we set $\lambda_2 = 2\kappa_2$. Moreover, a labeling L for the graph G_{ncgtw} can be written as $L = \{L_{ic} | 1 \leq i \leq N\}$, where L_{ic} is a labeling for the DTW graph G'_{ic}. Therefore, we can express the minimum cut problem for the graph G_{ncgtw} as:

$$\min_{L} g(L) = \min_{L \coloneqq \{L_{ic} | 1 \leq i \leq N\}} \sum_{1 \leq i \leq N} cut(L_{ic}) + \lambda_2 \sum_{1 \leq i < j \leq N} cut\left(L_{ic}, L_{jc}\right), \tag{2.8}$$

where $cut(L_{ic})$ is the cut of all edges for G'_{ic} and $cut\left(L_{ic}, L_{jc}\right)$ is the number of the cut edges between two connecting DTW graphs G'_{ic} and G'_{jc}.

Again, after applying any generic maximum flow algorithm, the ncGTW problem would be solved, and all final warping functions $\{\Phi_{i,c}\}$ are available.

2.2.3 Relationship between hyper-parameter and the solutions of Stage 1 and Stage 2

With a specific value of hyper-parameter $\lambda_1(\kappa_1)$ in Stage 1, we can obtain from equation (2.4) the corresponding label set L and the minimum cut. Similarly, in Stage 2, with $\lambda_2(\kappa_2)$ given, we can obtain the corresponding label set and the minimum cut from Equation S14. We also obtain the final warping functions corresponding to that minimum cut. If we change the value of λ_1 or λ_2, the minimum cut solution may or may not change. Since the solution space is discrete, a very minor change of λ_1 or λ_2 may not lead to the change of solution. If the change of the hyper-parameter is large enough, we may get a different minimum-cut solution.

The edges that are cut in any minimum cut solution can be grouped into two categories: those inside each DTW graph and those between DTW graphs. Therefore, the minimum cut obtained in Stage 1 or 2 can be viewed as a function of hyper-parameter λ (λ_1 in Stage 1 or λ_2 in Stage 2):

$$cut_{total}(\lambda) = cut_D(\lambda) + \lambda \times cut_B(\lambda), \tag{2.9}$$

where $cut_D(\lambda)$ is the cut for all DTW graphs and $cut_B(\lambda)$ is the total number of the cut edges between any two connecting DTW graphs. Note that λ (λ_1 or λ_2) can be replaced with κ (κ_1 or κ_2) since they are equivalent [47]. If we can test all possible values of the hyper-parameter, we can get all possible solutions of warping functions and we can choose the best one from them. However, practically this is intractable and consumes too much time. Instead, we hope to find some special properties of equation (2.9) and utilize those properties in the search for the optimal value of hyper-parameter.

Interestingly, we found that the total cut $cut_{total}(\lambda)$ in equation 2.9 is a concave non-decreasing piecewise linear function of λ, and each line segment is corresponding to a specific result (a set of warping functions) in Stage 1 or Stage 2. We will utilize those properties to reduce the search space of hyper-parameters and to obtain efficient approximate strategies. In this section, we will prove the property. We first introduce some lemmas.

Lemma 1 $cut_{total}(\lambda)$ is a non-decreasing function.

Proof: Assuming $\lambda'' > \lambda'$, if $cut_{total}(\lambda)$ is not a non-decreasing function, then there should be at least one pair of λ' and λ'' satisfies:

$$cut_D(\lambda') + \lambda' \times cut_B(\lambda') > cut_D(\lambda'') + \lambda'' \times cut_B(\lambda'').$$

Since $\lambda'' > \lambda'$, we can obtain:

$$cut_D(\lambda') + \lambda' \times cut_B(\lambda') > cut_D(\lambda'') + \lambda'' \times cut_B(\lambda'') > cut_D(\lambda'') + \lambda' \times cut_B(\lambda''),$$

and

$$cut_D(\lambda') + \lambda' \times cut_B(\lambda') > cut_D(\lambda'') + \lambda' \times cut_B(\lambda'').$$

By definition, $cut_D(\lambda')$ and $cut_B(\lambda')$ should be the cuts which give the minimum cut when λ is λ'. However, the above equation shows that when λ equals λ', $cut_D(\lambda'')$ and $cut_B(\lambda'')$ can give even lower total cut. Therefore, there is a contradiction. Thus, $cut_{total}(\lambda)$ is a non-decreasing function.

Lemma 2 $cut_D(\lambda)$ is a non-decreasing step function, and $cut_B(\lambda)$ is a non-increasing step function. The interval of each step of $cut_D(\lambda)$ is the same as the one of $cut_B(\lambda)$.

Proof: Assuming $\lambda'' > \lambda' \geq 0$, then

$$cut_D(\lambda') + \lambda' \times cut_B(\lambda') \leq cut_D(\lambda'') + \lambda' \times cut_B(\lambda''),$$

and

$$cut_D(\lambda'') + \lambda'' \times cut_B(\lambda'') \leq cut_D(\lambda') + \lambda'' \times cut_B(\lambda'),$$

since ($cut_D(\lambda')$, $cut_B(\lambda')$) and ($cut_D(\lambda'')$, $cut_B(\lambda'')$) should give the minimum cut when λ is λ' and λ'' respectively. Thus, from the above two inequalities, we can obtain:

$$\lambda'\big(cut_B(\lambda') - cut_B(\lambda'')\big) \leq cut_D(\lambda'') - cut_D(\lambda') \leq \lambda''\big(cut_B(\lambda') - cut_B(\lambda'')\big),$$

and thus

$$cut_B(\lambda') \geq cut_B(\lambda''),$$

$$cut_D(\lambda'') \geq cut_D(\lambda').$$

Therefore, $cut_D(\lambda)$ is a non-decreasing function, and $cut_B(\lambda)$ is a non-increasing function. Also, the possible values of $cut_D(\lambda)$ and $cut_B(\lambda)$ are discrete and countable, since the edges in DTW graphs and the edges between DTW graphs are countable. Thus, $cut_D(\lambda)$ is a non-decreasing step function, and $cut_B(\lambda)$ is a non-increasing step function.

Moreover, $cut_D(\lambda)$ and $cut_B(\lambda)$ will change together. When $cut_B(\lambda)$ increases/decreases, $cut_D(\lambda)$ should decrease/increase. If $cut_D(\lambda)$ does not decrease/increase, which means the previous $cut_B(\lambda)$ does not give the minimum cut and it is a contradiction. Therefore, the interval of each step of $cut_D(\lambda)$ is the same as the one of $cut_B(\lambda)$. □

Lemma 3 In each step of $cut_D(\lambda)$ and $cut_B(\lambda)$, $cut_{total}(\lambda)$ is a linear function with a positive slope.

Proof: From Lemma 2, we know that in each step of $cut_D(\lambda)$ and $cut_B(\lambda)$, $cut_D(\lambda)$ and $cut_B(\lambda)$ are both constants. Thus, Equation (2.8) becomes

$$cut_{total}(\lambda) = C_D + \lambda \times C_B,$$

where C_D and C_B are constants. Thus, in each step of $cut_D(\lambda)$ and $cut_B(\lambda)$, $cut_{total}(\lambda)$ is a linear function with a positive slope C_B. □

Theorem 1 The solved minimum cut from Stage 1 or Stage 2 is a concave non-decreasing piecewise linear function of λ_1 of λ_2, and each line segment corresponding to a specific alignment result in Stage 1 or Stage 2.

Proof: From **Lemma 2**, we know that $cut_D(\lambda)$ and $cut_B(\lambda)$ are not continuous since they are step functions. To prove Equation (2.6) is continuous, we need to prove that λ between each step of $cut_D(\lambda)$ (and also the same point of $cut_B(\lambda)$) is continuous for Equation (S12). That is, assuming C_D and C_B is the value of $cut_D(\lambda)$ and $cut_B(\lambda)$ for step i, and C_D' and C_B' is the value of $cut_D(\lambda)$ and $cut_B(\lambda)$ for step $i+1$, from **Lemma 3**, we need to prove:

$$C_D + \lambda \times C_B = C_D' + \lambda \times C_B',$$

for λ between each step of $cut_D(\lambda)$. If $C_D + \lambda \times C_B > C_D' + \lambda \times C_B'$, then $cut_{total}(\lambda)$ is not a non-decreasing function, which contradicts **Lemma 1**. If $C_D + \lambda \times C_B < C_D' + \lambda \times C_B'$, it means C_D and C_B give lower total cut than C_D' and C_B' for step $i+1$, which contradicts the fact that C_D' and C_B' always give the minimum cut for step $i+1$. Thus, $C_D + \lambda \times C_B = C_D' + \lambda \times C_B'$, and $cut_{total}(\lambda)$ is a continuous function. In addition, the slope of each line segment of $cut_{total}(\lambda)$ is non-increasing by **Lemma 2** and **Lemma 3**, so $cut_{total}(\lambda)$ is a concave non-decreasing piecewise linear function of λ. From **Definition 9** and **Definition 13**, we know that after obtaining $cut_D(\lambda)$ (the cut of DTW graphs), the warping function set is available for Stage 1 (if λ is λ_1) or Stage 2 (if λ is λ_2). Also, in each line segment of $cut_{total}(\lambda)$, $cut_D(\lambda)$ is the same. Thus, each line segment of $cut_{total}(\lambda)$ corresponds to a specific alignment result in Stage 1 or Stage 2. □

We can immediately obtain two corollaries:

Corollary 1 The first line segment of $cut_{total}(\lambda)$ is corresponding to solving each DTW graph separately.

The first line segment of $cut_{total}(\lambda)$ starts when λ is zero. Under such circumstance, there is no additional edge connecting DTW graphs, so we can solve each DTW graph separately to obtain $cut_D(\lambda)$.

Corollary 2 The last line segment of $cut_{total}(\lambda)$ is corresponding to solving a *single* DTW graph.

When λ is large enough (for example, larger than the maximum value of $cut_D(\lambda)$), no additional edge will be cut to avoid the large value of cut. As a result, $cut_B(\lambda)$ is zero. Hence, the warping functions (paths) of all the DTW graphs are the same to avoid cutting any additional edge. Therefore, we can create a new DTW graph whose topology is the same as any existing DTW graph. The capacity of each edge in the new graph is the summation of the capacities of corresponding edges from all existing graphs. The warping function obtained from the new DTW graph is the same as the warping functions of all original DTW graphs of the last segment of $cut_{total}(\lambda)$.

2.2.4 The strategy of finding the line segments of $cut_{total}(\lambda)$

To obtain the best alignment result of Stage 1 and Stage 2, we need to tune the hyperparameters λ_1 and λ_2 (κ_1 and κ_2). However, since the alignment results are countable (line segments), we can tune the hyperparameters much more efficiently by finding different line segments, instead of trying different values of λ_1 and λ_2 blindly. Assuming there are total k line segments, from **Corollary 1**, we can obtain the first line segment (segment 1) by solving each DTW graph separately, and from **Corollary 2**, we can obtain the last line segment (segment k) by solving a new DTW graph. As shown in **Figure 2.4**, we can find a crossing point by extending segment 1 and segment k respectively. The λ corresponding to the crossing point is the new hyperparameter we can try to obtain a new segment i. Again, we can find another two new segments by extending segment i to find the crossing points with segment 1 and segment k. Repeating the steps for newly obtained line segments, we can obtain different values of λ roughly uniformly along with line

segments. When this step is repeated with enough iterations, all line segments are identified if needed.

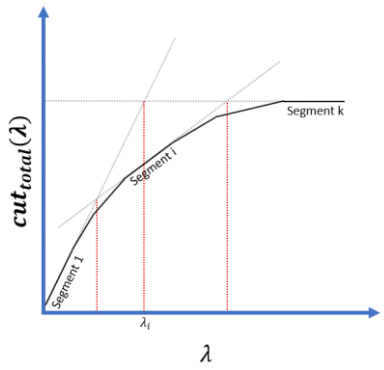

Figure 2.4 The illustration of the strategy of finding the line segments of $cut_{total}(\lambda)$. The first line segment (segment 1) can be found by solving each DTW graph separately (**Corollary 1**). The last line segment (segment k) can be found by solving the new DTW graph (**Corollary 2**). After extending segment 1 and segment k, we can obtain a crossing point, whose corresponding position on the λ axis is λ_i. With λ_i, we can solve the minimum cut problem to identify segment i. Similarly, if we extend segment i, we can find the crossing points with segment 1 and segment k and we can further identify new segments. If we repeat this step, more line segments can be identified.

2.2.5 Local or global optimum

In Stage 1 of ncGTW, we claimed that we could obtain the global optimum of a GTW problem. That is, we can solve the maximum flow problem from the GTW graph with the global optimum. One should notice that this global optimum is not the global optimum of the multiple alignment problem. In fact, the multiple alignment problem is an NP-hard problem. It is possible that the global optimum of GTW is just the local optimum of multiple alignment. In spite of that, ncGTW still has a superior advantage. In different fields, the ways of evaluation of the result of multiple alignment are very different. However, ncGTW can adjust the weights of the additional edges according to the evaluation method. Thus, with the structure information, ncGTW can approach to the global optimum of the multiple alignment better than other methods, with great flexibility to various evaluation criteria.

2.3 Experimental results

We evaluate the performance of ncGTW on both simulated and real datasets. Any neighborhood structure among samples can be incorporated into ncGTW as long as the structure can be represented as a graph. Although weights on edges can also be naturally integrated into ncGTW, prior knowledge on weights is very application-dependent and thus we assume the neighborhood structure has no weight. In this set of experiments, we test three different structures: line, block, and uniform (non-informative). Three peer methods, DBA, CPM, and GTW were selected for comparison due to their representativeness. DBA is a DTW based method that iteratively computes barycenter and aligns all samples to the barycenter [48]. CPM uses the hidden Markov model to learn the prototype function [44]. Since GTW does not provide the reference, we need to manually supply one if we want to apply GTW to the multiple alignment problem. In the experiments, each time we used one sample as a reference. We went through all samples one by one and take the average of all scores. For a visual demonstration, we choose the most informative one. In the following experiments, one can see that a bad reference may ruin the alignment of GTW. Also, other flaws of directly applying GTW on multiple alignment are demonstrated.

Both quantitative measurements and visual assessments were used to evaluate the performance. We adopt two quantitative criteria that are frequently used in the literature. They are the mean correlation coefficient (MCC) and simplicity (SP) [49]. After alignment, the correlation among samples is expected to increase. Thus, the mean of the correlation of all sample pairs can be considered as a quantification of the alignment quality. The definition of simplicity here is the sum of the fourth power of all singular values, where the sum of all singular values is normalized to be one. The singular values are computed based on the data matrix, where each row represents a sample. The idea is that if it is good alignment, the first singular value should dominate others. Hence, larger simplicity means better alignment. Note that the largest possible value for simplicity is one.

For the simulation data, since we have the ground truth, we test each peak separately so that we can know the alignment quality of each peak. For the real data, the numerical evaluations for each peak are not applicable since we do not have the ground truth.

2.3.1 Simulation data generation

To understand the performance of each method on different structures, we first conduct three case studies on synthetic datasets with a line, block, and non-informative structures respectively. To prevent the impact of the other factors, the shapes of all peaks are sinusoidal. Also, the distance between the neighboring peaks is set to be the same for all simulated samples, and the shifts between the neighboring simulated samples are all one (for block structure, the shift between the two batches is seven). The apex intensity of each peak is extracted from the samples with similar peak intensity in the MESA dataset. Moreover, to fully observe the impact of the missing peak problem, we randomly picked a simulated sample and remove one peak from it.

To compare all the methods on a small-scale real dataset, we picked samples from the MESA dataset. For the line structure, we picked samples with a clear linear structure. For the block structure, we picked two groups of samples. Within each group, the RT shifts are similar, but the RT shift between groups is obvious. For the non-informative structure, the samples for line structure are re-used but without the structure information.

2.3.2 Case study on Line Structure

When samples change gradually according to a certain variable (such as time, **Figure 2.5a**), we call it to have a line structure, which can be converted to triangles (**Figure 2.5b**) as an input structure between pairwise alignments. To evaluate the performance of different methods, we first designed a simulation dataset containing five samples. The first sample is shown in **Figure 2.6a** and all five samples are plotted in **Figure 2.6b**. Note that all the samples contain three peaks, except the fourth sample (purple dash line, the third peak of which is missing). This phenomenon of missing peaks occurs frequently in real applications.

Figure 2.6c-f shows the alignment result of DBA, CPM, GTW, and ncGTW, and **Table 2-1** shows the evaluation scores of each peak. As the ground truth, the two peaks pointed by black arrows belong to the first group and the second group, respectively. The MCC and SP of all peaks of all methods are improved compared with the original curves. However, DBA wrongly aligned the peaks in sample 4, which leads to bad scores of MCC and SP. CPM aligned all peaks correctly, so it got good scores for all groups in three evaluations. In **Figure 2.6e**, the reference of GTW is the

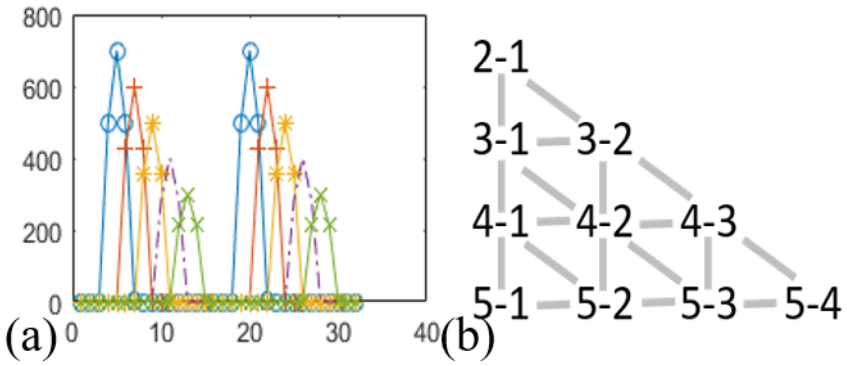

Figure 2.5 Example of five-curve with line structure. (a) Five continuously changing samples: the first one is drawn in "o", second in "+", third in "*", fourth in "-", and fifth in "x". The shifts between the directly neighboring samples are all two points. (b) The induced neighborhood structure between warping functions.

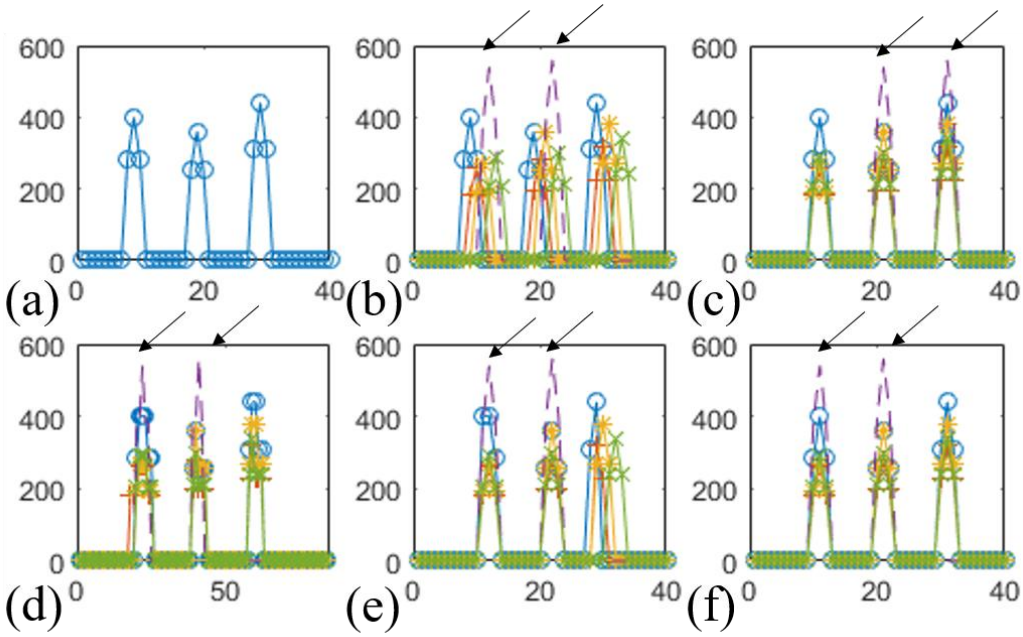

Figure 2.6 Case study on the line structure. (a) The first sample. (b) The five synthetic samples are drawn in the order of "o", "+", "*", "-", and "x". The shifts between neighboring samples are all one. All samples contain three peaks, except for the fourth sample, which misses the third peak (the other two are pointed by the arrows). (c) Alignment result of DBA. (d) CPM. (e) GTW. (f) ncGTW.

Table 2-1 Peak scores for four methods of line structure. 'Before alignment' serves as the baseline. MCC and SP represent mean correlation coefficient and simplicity respectively. The range of either score is between 0 and 1 with a higher score indicating better performance.

Peak1		
Scores / Methods	MCC	SP
Before alignment	0.3071	0.4240
DBA	0.5683	0.5095
CPM	0.9697	0.9390
GTW	0.9988	0.9986
ncGTW	0.9999	0.9999
Peak2		
Scores / Methods	MCC	SP
Before alignment	0.3072	0.4407
DBA	0.5671	0.5118
CPM	0.8943	0.8169
GTW	0.9999	0.9999
ncGTW	0.9999	0.9999
Peak3		
Scores / Methods	MCC	SP
Before alignment	0.0848	0.3959
DBA	0.4912	0.9999
CPM	0.5042	0.9384
GTW	0.3986	0.8791
ncGTW	0.5099	0.9999

fourth sample. Since the fourth sample lacks the third peak, we can see from the figure that the third peaks of all samples are not aligned well. This is an example showing that the reference may have a huge effect on alignment. Moreover, the variance of the fourth sample is the largest one. In fact, many existing methods posit that the sample with the largest variance should be selected as the reference. As we have shown here, reference selection is indeed a hard problem. Even with averaging, the scores of peak 3 of GTW are still much worse than other methods. ncGTW produced accurate alignment as evidenced by both visual assessment and quantitative scores. One may notice that the MCC of peak 3 is all smaller than 0.6, which is due to the missing peak in the fourth sample.

2.3.3 Case study on the block structure

Block structure means there are several blocks formed by samples in the dataset. Within each block, the shifts are small. Between blocks, the shift is larger. Suppose we have ten samples in a dataset where every five samples form a block. **Figure 2.7** shows how to connect these pairs. We can separate these pairs into three types. The first type is the alignment within the first block. The second type is between the two blocks. The third type is within the second block. Only the same type of neighbors will be connected. To test this structure, we generated a ten-sample dataset, and every five samples form a block. **Figure 2.8a** shows the first sample, which contains three peaks. **Figure 2.8b** shows the eighth sample, of which the third peak is missing. **Figure 2.8c-d** shows the two blocks in the dataset. **Figure 2.8e** shows all the samples.

Figure 2.8f- i show the results of DBA, CPM, GTW, and ncGTW, and **Table 2-2** shows the peak scores for each method. DBA wrongly aligned the two peaks in the eighth sample, due to the same reason as in the previous section. CPM has the worst performance since CPM separated all the peaks into four groups, not three. The reason may be that in **Figure 2.8e**, it somehow shows four peak groups, and in each group, there are at least five peaks. Thus, CPM incorrectly treated them as four peak groups, not three. The reference of GTW is the tenth sample, which contains three peaks. With a good reference, in **Figure 2.8h**, GTW has the performance as good as ncGTW. However, since there is a missing peak in the eighth sample, after averaging, the MCC of peak 3 is a little bit lower than ncGTW. Again, the MCC of peak 3 is not close to one for all methods, and this is also due to the missing peak in the eighth sample.

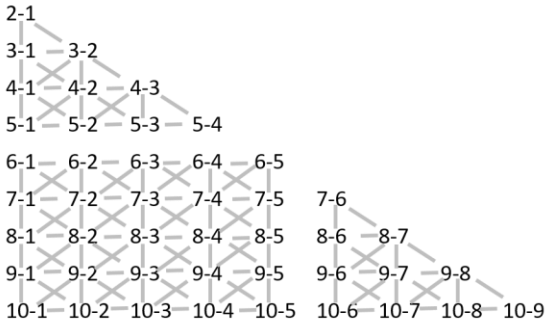

Figure 2.7 The way to connect samples in a dataset with two blocks (5 samples in each block). One can see there are three types of pairs: within the first group, between 2 groups, and within the second group.

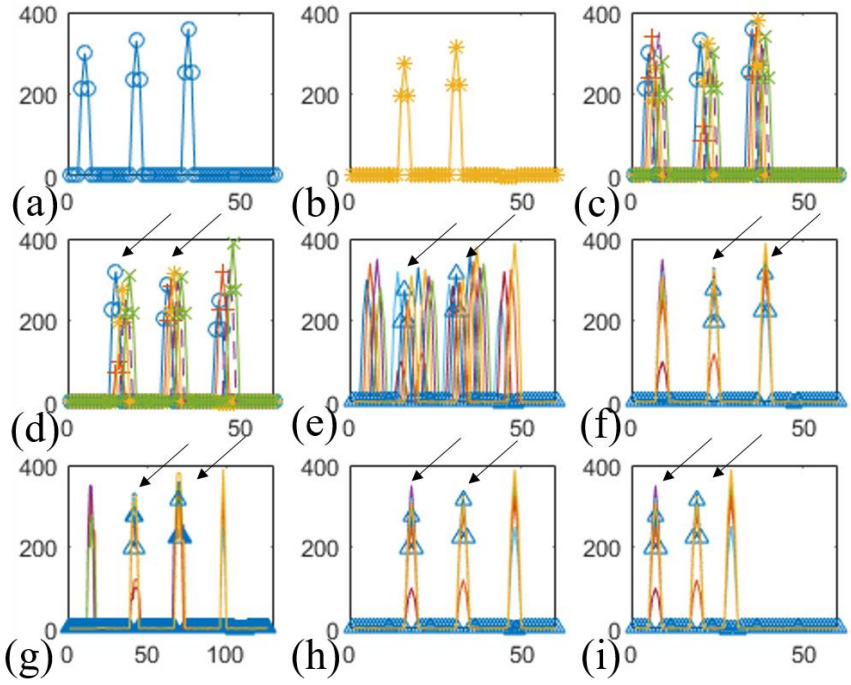

Figure 2.8 Case study on block structure. (a) The first sample. (b) The eighth sample. The third peak is missing. (c) The first block of the dataset. The samples are drawn in the order of "o", "+", "*", "-", and "x". The shifts between neighboring samples are all one. (d) The second block of the dataset. The five samples are drawn the same way as the previous subfigure. In the following subfigures, only the eighth sample is drawn with marks (triangle). Its peaks are pointed by arrows. (e) All ten samples. The shift between the two blocks is seven points. (f) Alignment result of DBA. (g) CPM. (h) GTW. (i) ncGTW.

Table 2-2 Peak scores for four methods of block structure.

Peak1		
Scores / Methods	MCC	SP
Before alignment	0.1163	0.1937
DBA	0.7894	0.8326
CPM	0.3923	0.4603
GTW	0.9835	0.9778
ncGTW	0.9998	0.9998
Peak2		
Scores / Methods	MCC	SP
Before alignment	0.1165	0.1937
DBA	0.7885	0.7957
CPM	0.3988	0.4801
GTW	0.9301	0.9379
ncGTW	0.9999	0.9998
Peak3		
Scores / Methods	MCC	SP
Before alignment	0.0859	0.2002
DBA	0.7521	0.9999
CPM	0.3408	0.5831
GTW	0.6926	0.9407
ncGTW	0.7481	0.9998

2.3.4 Case study on the non-informative structure

Sometimes we may know nothing about the structure of the dataset. In this section, we demonstrate the experiments on the dataset without any structural information. For simulation, here we consider a ten-sample dataset. Without structural information, in the first stage of ncGTW, we add edges between the pairs that have the same aligning sample or reference sample. For example, for $G'_{1,2}$, we will connect it to $G'_{1,i}$, where i is from 3 to 10. Likewise, for $G'_{2,1}$, we will connect it to $G'_{i,1}$, where i is from 3 to 10. In this dataset with 10 simulated samples, there are two samples without peak 1, another two without peak 2, and still another two without peak 3. Thus, there are only four samples with all three peaks. **Figure 2.9a** shows a sample without peak 2. **Figure 2.9b** shows all ten samples. **Figure 2.9c-f** show the results of DBA, CPM, GTW, and ncGTW, and **Table 2-3** shows the peak scores for each method. DBA again wrongly aligned the sample shown in **Figure 2.9a** (pointed by arrows). CPM aligned all the peaks well but with some small drifts in some samples and distortions, so the scores are not as good as ncGTW but still better than DBA. With the sample shown in **Figure 2.9a** as a reference, GTW again misaligned the peak 2 group as shown in **Figure 2.9e**. Since there are six samples with a peak missing, the average scores of GTW are all relatively low. Even without any structure information, ncGTW has the best performance.

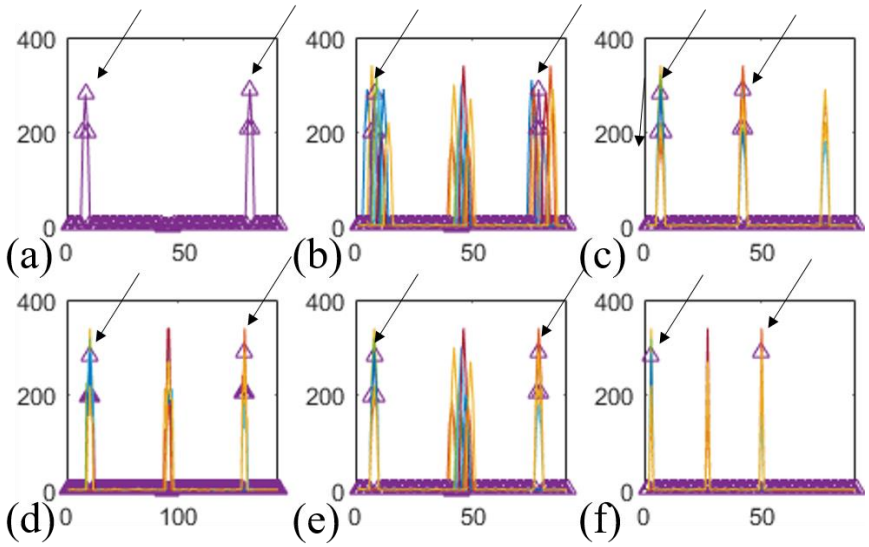

Figure 2.9 Case study on the non-informative structure. (a) A sample without peak 2 (marked as triangles in the following subfigures). (b) All ten simulated samples. (c) Alignment result of DBA. (d) CPM. (e) GTW. (f) ncGTW.

Table 2-3 Peak scores for four methods of non-informative structure.

Peak1		
Scores / Methods	MCC	SP
Before alignment	0.0686	0.2293
DBA	0.5475	0.9996
CPM	0.5931	0.9090
GTW	0.5312	0.8433
ncGTW	0.6750	0.9991
Peak2		
Scores / Methods	MCC	SP
Before alignment	0.0695	0.2545
DBA	0.2809	0.7712
CPM	0.5041	0.8091
GTW	0.4759	0.8482
ncGTW	0.6673	0.9989
Peak3		
Scores / Methods	MCC	SP
Before alignment	0.0542	0.2184
DBA	0.3108	0.5238
CPM	0.5980	0.9419
GTW	0.4707	0.8427
ncGTW	0.7435	0.9990

2.3.5 Case study on small scale real LC-MS dataset

In this section, we conducted tests on a real LC-MS experiments. First, ten samples were selected and ordered by the time as they were assayed. Assuming the properties of the equipment were gradually changing, we impose a line structure among these samples. **Figure 2.10a** shows one sample from the ten samples. Clearly, there are nine peaks in the sample. **Figure 2.10b** shows all ten samples together. Note that the intensities of the corresponding peaks between samples are very different, and some peaks are missing.

Figure 2.10c-f show the results of DBA, CPM, GTW, and ncGTW. DBA wrongly aligned the peak pointed by the arrow to the third group of peaks (that peak should be aligned to the second group). The reason is that DBA is based on DTW. At some steps, DTW aligned a peak to a peak with a similar intensity, but these two peaks are not in the same group. DTW based methods have the tendency to align the peaks with similar intensities together. CPM also aligned the same peak wrongly to the third group. Similar to DBA, the intensity of the peak is so strong that CPM decided to align this peak to the group with a higher intensity. The reference sample of GTW is just the one shown in **Figure 2.10a** and GTW aligned the arrow-pointed peak correctly, while the fourth and fifth peak groups were misaligned. Again, ncGTW aligned all the peaks well, so that the nine peak groups are clearly separated and none of them is clumped together.

Next, we chose 20 samples from two batches where each batch contains 10 samples. Among the samples in each batch, the retention time drift is small. However, between the first and the second batches, the drift is larger. **Figure 2.11a-b** show an example of each batch, and there are three peaks for each two sample. However, the third peak of most samples is missing. **Figure 2.11c-d** show the first and second batches in the dataset. There are three peak groups in both batches. However, for the third peak group, only two samples in the first batch and only three samples in the second batch have the peak. To see other peaks clearer, in **Figure 2.11e**, we show all the samples by changing the scale of the y-axis.

Figure 2.11f-i show the results of DBA, CPM, GTW, and ncGTW. DBA aligned more than ten peaks to the third peak group with serious distortions. There should be five peaks in that peak group. CPM also aligned wrongly the third peak group, since there are too many missing peaks in the third peak group. The reference of GTW is the first sample of the first batch, as shown in

Figure 2.11a. Though there is no missing peak in the reference, GTW still tends to align the peaks with similar intensity together. Thus, the third peak group has more than five peaks and the shapes of the peaks are distorted. Only ncGTW aligned correctly the five peaks to the third peak group. This example demonstrated the significant improvement of ncGTW compared with GTW. We can see the advantage that ncGTW does not need a certain reference, and there is no distortion after alignment.

To test ncGTW on data with no structure among samples (or in the scenario that we are not sure about its structure), we use the same ten LC-MS samples in **Figure 2.10b** but we do not incorporate the a priori structural information. Instead, we apply the non-informative structure by adding edges between the pairs with the same aligning sample or reference sample in the first stage of ncGTW.

Figure 2.10a-d are the same as **Figure 2.12a-d**, since DBA and CPM do not consider structural information already. As shown in **Figure 2.12e-f**, unlike DBA and CPM, GTW and ncGTW still well aligned the pointed peak. However, without the structural information, GTW misaligned peak 3 and peak 4. Even without structural information, ncGTW accurately aligned all the peaks.

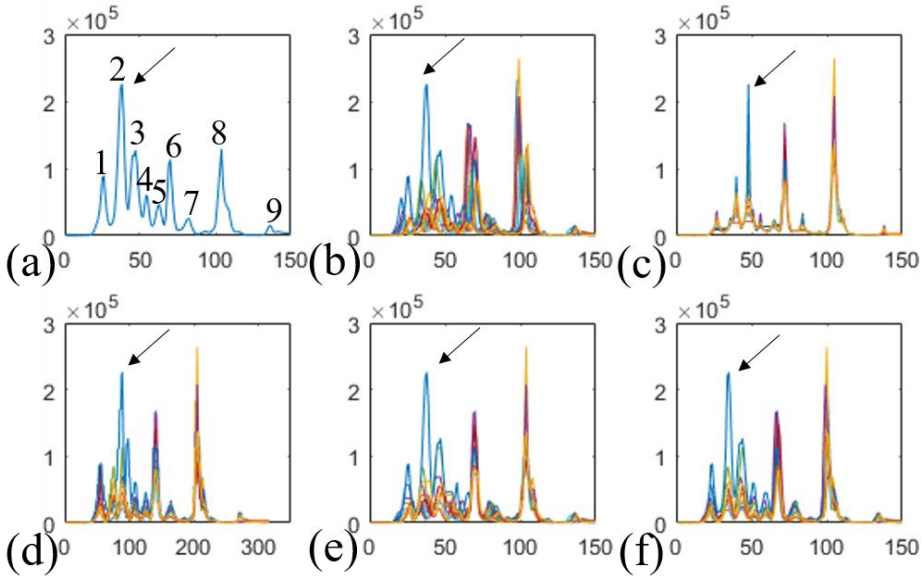

Figure 2.10 Case study on real LC-MS dataset of ten samples. (a) One exemplary sample with nine clear peaks marked with indexes. The second peak (pointed by an arrow) was wrongly aligned by most methods except GTW and ncGTW. (b) All ten samples. (c) Result of DBA. (d) CPM (e) GTW (f) ncGTW.

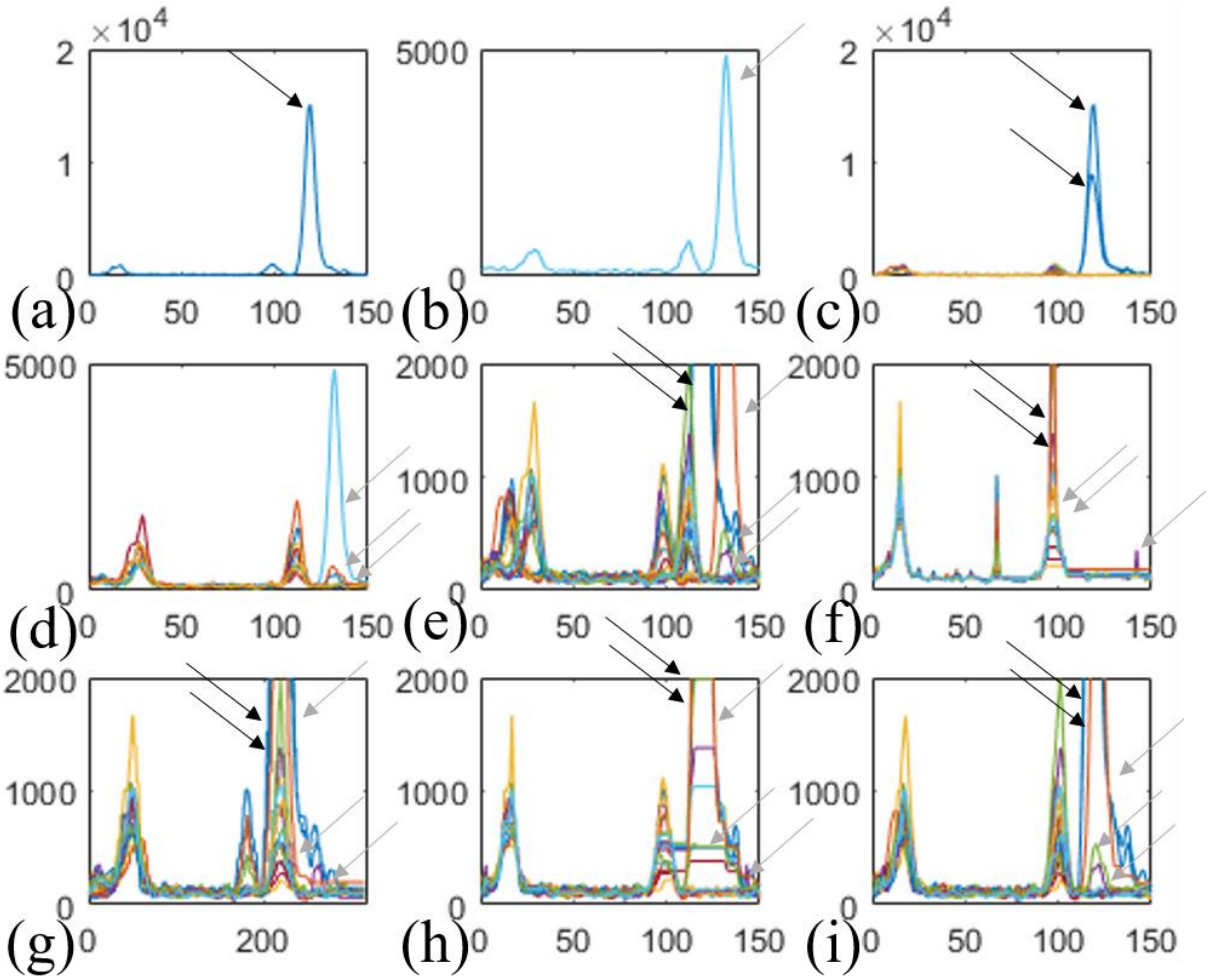

Figure 2.11 Case study on real LC-MS dataset of twenty samples from two batches. (a) The first sample from the first batch, in which there are three peaks. (b) The sixth sample from the second batch, in which there are also three peaks. (c) The first batch. For the third peak group, only two samples have the peak (pointed by the black arrows). (d) The second batch. Only three samples have a peak in the third peak group (pointed by the gray arrows). (e) All twenty samples. (f) Result of DBA. (g) CPM. (h) GTW. (i) ncGTW.

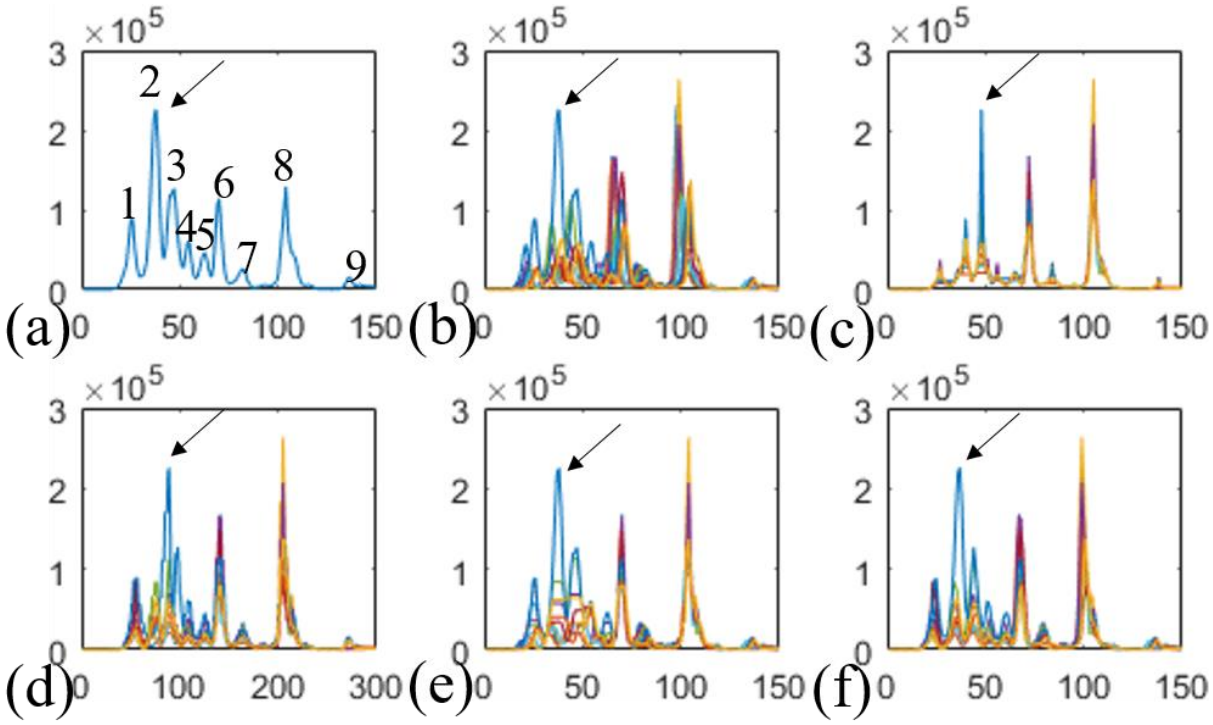

Figure 2.12 Case study on real LC-MS dataset of ten samples without prior knowledge of the structure information. (a) One exemplary sample with nine peaks. The second peak (pointed by an arrow) was wrongly aligned by most methods except GTW and ncGTW. (b) All ten samples. (c) Result of DBA. (d) CPM. (e) GTW. (f) ncGTW.

2.4 Discussions

Readers may be aware of the significant progress on aligning multiple DNA sequences. One may wonder why similar progress has not been seen in the general category of multiple alignment and why good ideas for DNA sequence alignment cannot be directly transferred to other applications. In our view, this is not surprising because many effective approaches for DNA sequences explicitly or implicitly rely on the assumption of the existence of the evolution tree. Yet, the tree structure is very special and many applications cannot be described by a tree structure. Our ncGTW can be understood as an extension of tree structure to any graph structure. In addition, unlike methods designed for DNA sequences, our method models all samples simultaneously. Though we did not test our algorithm on DNA sequencing data, we expect to see favorable performance due to the integrative modeling nature.

Our approach is built on GTW, a recent extension of DTW to multiple pairs. GTW retains all desirable properties of DTW such as monotonicity of time shifts and polynomial efficient solution, and yet flexibly models any graph-encoded structure among pairs. However, GTW cannot be directly applied to solve multiple alignment problem. GTW takes multiple pairs of samples as input and finds alignment for each pair with consistency between pairs considered. The problem considered in this paper takes multiple samples as input and aims to find consistent alignment among samples. Though we can manually specify one certain sample as a reference to construct pairs of samples for input of GTW, to our knowledge, there is no established method that can always help user to identify which sample is the most suitable one as the reference. Moreover, it is likely that none of the samples contains enough information to serve as a good reference. That is, no matter we choose any sample as the reference, we can never obtain an accurate alignment. Thus, ncGTW is a significant improvement of GTW, since ncGTW can model all samples simultaneously and deal with multiple alignment problem without manually setting a reference.

The experiments clearly demonstrated the power of ncGTW. When structural information is available, it is anticipated to see increased accuracy of alignment due to the use of extra information. It is a little bit counterintuitive to observe that ncGTW still performed better when there was no informative structure. From a hindsight, this is also expected because one can always borrow information from other samples if we use a Bayesian perspective and consider all samples forming a prior distribution. This phenomenon is also related to Stein's paradox in the estimation theory [50].

2.5 Summary

In this chapter, we illustrated structure aware multiple alignment (SAMA), a novel method for multiple alignment. Inherited from GTW, SAMA incorporates the structure information in the dataset when alignment. Also, as an extension of GTW, the graph problems built by SAMA can be solved by network flow algorithms easily. Moreover, SAMA does not need a certain reference, which avoids the unwanted variance due to different reference. Through the simulations and the real LC-MS datasets, SAMA demonstrates its superiority over the classic alignment methods. With the flexibility and structure incorporation, SAMA may handle any kind of multiple alignment problem, especially for omics dataset.

Chapter 3

LC-MS signal alignment

3.1 Introduction

For proteomics or metabolomics analysis of biological samples, liquid chromatography coupled with mass spectrometry (LC-MS) is a standard method [51, 52] that produces two-dimensional profiles of constituent compounds over retention time (RT) and mass-to-charge ratio (m/z). The identity and quantity of a particular compound (known [53] or unknown [54]) may be inferred by analyzing the associated characteristic peak/curve profile (RT, m/z & intensity information). When analyzing multiple samples, the RT of each compound must be aligned accurately across different samples.

Many software packages for LC-MS data analysis include a tool kit that performs RT alignment, such as XCMS [16, 45]. However, due to varying RT drift over different m/z bins in a sample and significant RT drift across distant samples (samples with larger run order difference), often nonlinear, accurate RT alignment remains a challenging task [45]. Unfortunately, classic alignment methods conveniently assume a single warping function across all m/z bins and perform multiple alignment that neglects the run order of each sample [55-57]. These methods largely ignore the aforementioned two factors, and thus are prone to various types of misalignment. For a large-scale experiment involving many samples, some degree of misalignment for certain features becomes inevitable. For some types of misalignment, an algorithmic effort to mitigate them is to optimize parameter values in complex alignment algorithms. However, current strategies for handling a large number of parameters or features (peaks from the same compound at a single m/z bin with aligned RT across samples) are ad hoc, labor-intensive, subjective, and often fail to achieve a desired performance. Furthermore, for misaligned features whose true warping functions are different over different m/z bins, the misalignment cannot be corrected simply by adjusting the parameters of a single warping function (**Figure 3.1**). Moreover, no existing analytics tool includes a systematic way to detect misalignment and thus previously acknowledged misalignment is often undetectable or uncorrected.

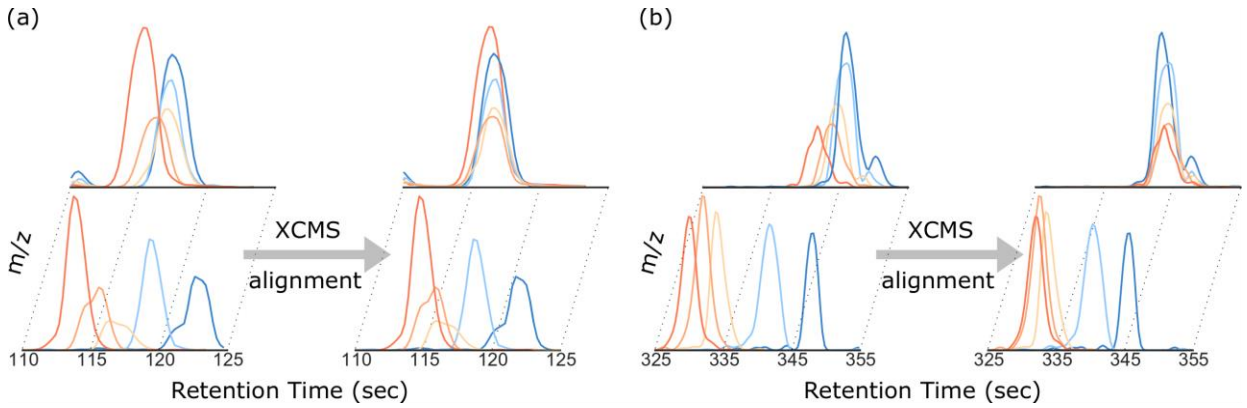

Figure 3.1 Examples of the observed misalignments due to single warping function assumption. Five samples over two m/z bins from each dataset are shown here for demonstration, where the upper and lower rows represent two different m/z bins respectively (see details in sub-section 3.3.1). (a) An example from the Rotterdam dataset, shows that even with similar RT, the drift of each sample could be significantly different in two m/z bins. Using only a single warping function, XCMS can only align one bin (the upper one) well but not the other one as shown in the right part. (b) A similar example is also observed in MESA dataset.

To address the critical problem of the absence of validated methods for misalignment detection and structured alignment, we developed an integrated reference-free profile-based alignment method, neighbor-wise compound-specific Graphical Time Warping (ncGTW), that first detects misaligned features and then aligns affiliated profiles. In contrast to feature-based methods that only align detected peaks, fundamental to the success of our approach is the incorporation of expected RT drift structures across both different m/z bins or distant samples. Specifically, under the GTW framework [47], ncGTW uses individualized warping functions for different m/z bins and assigns constraints on warping functions of neighboring samples. Furthermore, ncGTW utilizes a two-stage algorithm to achieve a reference-free alignment, where combinatorial pair-wise alignments are first performed and these aligned profiles are then coordinately aggregated into a pseudo-reference.

The input data to be analyzed by ncGTW includes the peak information extracted by XCMS and the raw data profiles corresponding to misaligned features. First, a statistically-principled misalignment detection scheme is applied to identify features requiring realignment. Then, each of the two-stage alignment procedures in ncGTW is solved efficiently by network flow-based

algorithms [58]. The ncGTW software tool takes full advantage of feature-based alignment methods and is provided currently as a plug-in to XCMS package.

3.2 Method

Generally, our package includes two steps. The first step is to detect the misaligned features with the feature information provided by any pre-processing tool. The user can tune the parameters of the pre-processing tool depending on the result of this step. After adjusting the parameters, if the user wants to correct the misalignment with our package, then one can send the result from the first step to the second step, the realignment part. Due to the global warping function assumption, XCMS is easily to have misaligned peaks, so we use XCMS as the preprocessing tool to demonstrate our package. The features (groups of peaks) from XCMS are the input of ncGTW. Figure 3.3 is the workflow of ncGTW as post-processing of XCMS. The details of each step are explained in the following sections. After the realignment, ncGTW can send the result back to XCMS so that XCMS can regroup the misaligned peaks more accurately. Moreover, for the peak-filling step of XCMS, due to the alignment is improved, XCMS has higher chance to retrieve the missing peaks back. That is, our package can significantly improve the accuracy of the peak-filling step of XCMS when there are many missing peaks.

3.2.1 Detection of misaligned features

Accurate alignment of RT over a large number of samples remains a challenging task, particularly across distant samples due to significant yet varying RT drift. RT drifts between neighboring samples are small and gradual, even though random RT drifts may occur. More explicitly, the gradual RT drifts among neighboring samples, mainly due to temperature or column age [21, 59], can be modeled by the closer or similar run orders [60]. For a given feature corresponding to a set of peaks detected in several samples (e.g. by XCMS), we assume that the abundances of corresponding compound are independent of sample indices (this is an implication of randomizing the run order – standard practice in analytical science) and alignment relies on sufficient compound abundance in the relevant samples. Thus, for an accurately aligned feature, the samples associated with this feature should be a subset randomly drawn from the entire sample set, including both neighboring and very distant samples in the run-order domain (sample index). In contrast, a

misaligned feature often splits into multiple features by the feature detection algorithm with non-ideal parameter setting (**Figure 3.2**). Since most of alignment algorithms tend to group the peaks with similar RT drift together [45], regardless of the alignment accuracy, neighboring samples tend to be aligned together due to their similar RT shift. In other words, when a detected feature includes only neighboring samples, it would highly likely constitute a misaligned feature (one of the most common forms of misalignment). Thus, even though the RT drift goes back and forth, and features are misaligned, the neighboring samples would still be grouped into a feature, as shown in **Figure 3.2**. Accordingly, we designed a statistically-principled approach to detect misaligned features. We associate the null hypothesis with correct alignment, use the range of sample indices within feature as the test statistic, and detect misaligned features by rejecting the null hypothesis.

Assume there are N samples in total. For a provisionally-aligned feature for which the peak is detected in a total of n samples, as the first criterion, we denote the indices (run-order) of these samples as a set $\{l_1, l_2, \dots, l_n\}$, where $1 \le l_i \le N$ and $l_1 < l_2 < \cdots < l_n$. For this feature, we define the test statistic

$$t = \max_{1 \le i,j \le n} |l_i - l_j|, \tag{3.1}$$

which is the same as the *range* of the n indices. It is clear that $n - 1 \le t \le N - 1$. Under the null hypothesis, these n samples are randomly drawn from the N samples. Therefore, each index in this feature can be viewed as a random variable, and we assume it follows a discrete uniform distribution. Our goal is to find the null distribution of t, which is obtained by subtracting the smallest of these n random variables from the largest one. To achieve this, we sort these n random variables from small to large. Each *sorted* random variable is no longer uniformly distributed. Instead, its distribution needs to be calculated using order statistics [61]. For l_i, the ith smallest index in the feature, the probability mass function (pmf) is

$$f_{l_i}(k) = \frac{\binom{k-1}{i-1}\binom{N-k}{n-i}}{\binom{N}{n}}, i \le k \le N - n + i, \tag{3.2}$$

and the joint pmf of l_i and l_j is given by

$$f_{l_i,\,l_j}(k,l)=\frac{\binom{k-1}{i-1}\ \ \binom{l-k-1}{j-i-1}\ \ \binom{N-l}{n-j}}{\binom{N}{n}},\ i\le k<l\le N-n+j,\ l-k\ge j-i. \quad (3.3)$$

To obtain the null distribution of the test statistics, we let $i=1, j=n, l=k+t$. Thus, equation (3.3) becomes

$$f_{l_1,\,l_n}(k,k+t)=\frac{\binom{t-1}{n-2}}{\binom{N}{n}},\ 1\le k<k+t\le N,\ t\ge n-1. \quad (3.4)$$

Therefore, under the null hypothesis, the pmf of t is

$$f_T(t)=\sum_{k=1}^{N-t} f_{l_1,\,l_n}(k,k+t)=(N-t)\frac{\binom{t-1}{n-2}}{\binom{N}{n}},\ n-1\le t\le N-1. \quad (3.5)$$

For an observed feature, we can calculate its *p*-value:

$$p\text{-}value=\Pr\{t\le t_{obs}\}=\sum_{t=n-1}^{t_{obs}}(N-t)\frac{\binom{t-1}{n-2}}{\binom{N}{n}}, \quad (3.6)$$

where t_{obs} is the test statistic calculated in that feature [62]. The summation is over all ts that are smaller than t_{obs}, which reflects the assumption that the more concentrated the indices are, the more likely misalignment exists. The smaller the p-value, the more unlikely the observed feature follows the null distribution, and the more likely misalignment occurs.

To address the additional layer of complication concerning varying RT drift over different m/z bins, for a candidate misaligned feature, we further check whether there exists neighboring feature(s) in the same m/z bin with sufficiently small p-value while with disjoint sets of sample indices, and if so, consider these features as needing to be realigned together. The rationale behind this second criterion is that applying the same warping function cross different m/z bins would likely, yet wrongly, split the complete feature of a single compound into several pieces.

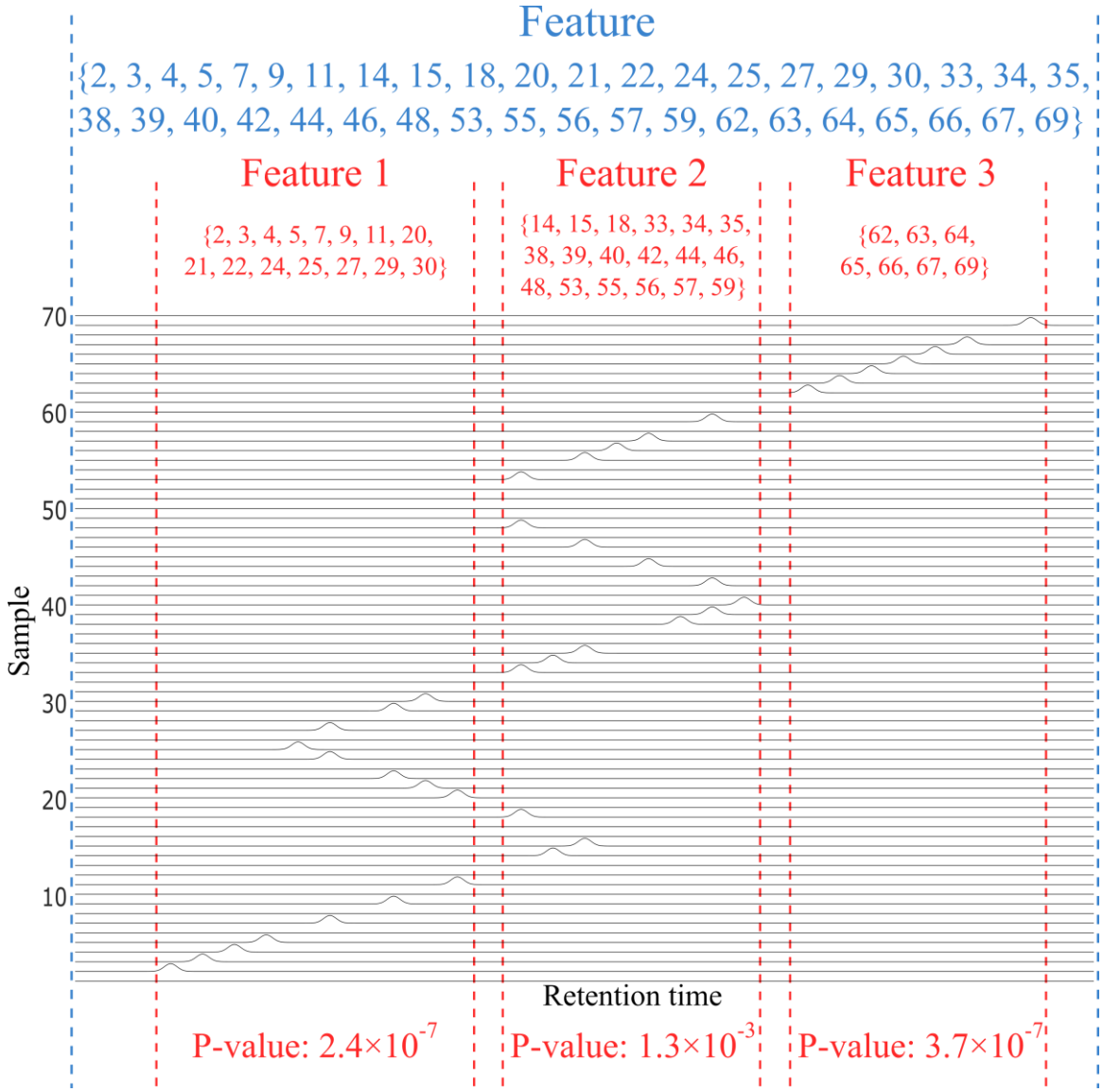

Figure 3.2 Illustrative example on detecting misaligned features. After initial alignment, among the total 70 samples, relevant peaks are detected only in some samples (the indices in blue), and some of the feature(s) are obviously misaligned. With a lower resolution grouping by XCMS, these peaks are all grouped into one single feature, as shown between the two blue dashed lines. While with higher resolution grouping, this feature is split into three features 1-3 as separated by the red dashed lines. The sample index sets of these features are shown in red respectively. The p-values of features 1-3 are all smaller than 0.05, thus pass the first criterion. Because the sample index sets of these three features are also disjoint, they pass the second criterion. Accordingly, ncGTW will detect the misalignment and realign the whole blue feature produced by the lower resolution grouping.

In the ncGTW algorithm, these two criteria are combined to detect misaligned features. The initial alignment is performed using the existing XCMS alignment module. After XCMS alignment, two separate grouping results are produced using different RT window parameter values (bandwidth, bw) in the XCMS peak-grouping module (feature detection). More precisely, the lower resolution grouping uses an RT window corresponding to the expected maximal RT drift, while the higher resolution grouping uses an RT window near the RT sampling resolution (the inverse of scan frequency). First, ncGTW algorithm estimates the p-value of each feature using higher resolution grouping result and identifies all features with sufficiently small p-values and disjoint sample subsets. Then, the ncGTW algorithm matches the neighboring features to the corresponding features produced by lower resolution grouping, and considers realigning these features. An illustration of misalignment detection is given in **Figure 3.2**.

The true causes for a misaligned feature may be complex and hidden, and may involve multiple yet unknown factors. It may be arguably suspected that the observed misalignment by existing alignment methods is at least partially due to the unstructured cost distributions over neighboring versus distant samples adopted by these methods. The joint optimization may thus be overly influenced by neighboring samples with intrinsically less costs. However, such a biased solution is clearly in conflict with the very purpose of a globally optimal alignment that should be able to simultaneously correct larger and complex RT drift over samples that are widely separated in run order.

3.2.2 Feature realignment

After misalignment detection, if the user decides to correct the misalignments with our package, one can directly send the detected misalignments from the previous step to the next step to realign these features. As mentioned above, the introduction of neighbor information makes our package can align features with limited information. If we even have only one m/z bin information, our package can still align the peaks well, with the help of structure information. As described in the last chapter, the core algorithm is ncGTW, which do not have the global warping function assumption, so that it can fix the misalignments. For the peak distortion problem, we take the advantage of the information from the input, the alignment result from other feature-based packages. Thus, the whole procedure is a feature-profile hybrid alignment method.

3.2.2.1 Implementation of ncGTW on a large dataset

For a large dataset, it is very time-consuming to simultaneously align all the samples for a profile-based alignment method. When the sample number is large, the ncGTW graph becomes extremely huge, and it may take hours or even days to solve the maximal flow problem. Thus, in the practical implementation, ncGTW splits the whole dataset into several sub-datasets, and performs alignment on each small dataset. In this way, the numbers of nodes and edges in the graph for each small dataset will decrease significantly comparing to the original graph. Also, since these small datasets are aligned independently, the computation time can be further reduced with parallel computing. After the alignment of each small dataset, we build a "super-sample" for each small dataset. Then, align these super-samples to obtain the warping functions of super-samples. With the warping functions within each small dataset and of the super-samples, we can obtain the final warping functions for each sample. We called this hierarchical alignment process as "two-layer ncGTW". **Figure 3.3** shows the diagram of two-layer ncGTW.

The first layer of two-layer ncGTW

In the first layer, to begin with, we need to decide how many sub-datasets we should split the original dataset into. For the computational efficiency, we recommend in each sub-dataset there should be at least 10 samples. For example, if there are 500 samples in the original dataset, it is recommended to split the dataset into 10 sub-datasets (50 samples in each). Then, we apply ncGTW on these sub-datasets independently. If the CPU cores the user has are more than 10, the computation time of this layer is equivalent to apply ncGTW on a 50-sample dataset once. After the alignment of each sub-dataset is done, we obtain the warping functions of all samples. With the warping functions, within each sub-dataset, the samples can be aligned to the same RT axis, and a “super-sample” will be generated by taking the average on the aligned samples. After the super sample of each sub-dataset is generated, we build a super-dataset with these super samples and send the super-dataset to the second layer with the warping functions of all samples.

The second layer of two-layer ncGTW

In the second layer, ncGTW is applied to the super-dataset to align the super-samples. Since the super-samples can be viewed as samples, we can directly apply ncGTW on them. Therefore, we can obtain a set of warping functions for super-samples. With the warping functions from the first

layer (sample to super-sample) and the ones for super samples (super-sample to the final axis), all the samples can be aligned to the final axis, so that the alignment of all samples is done.

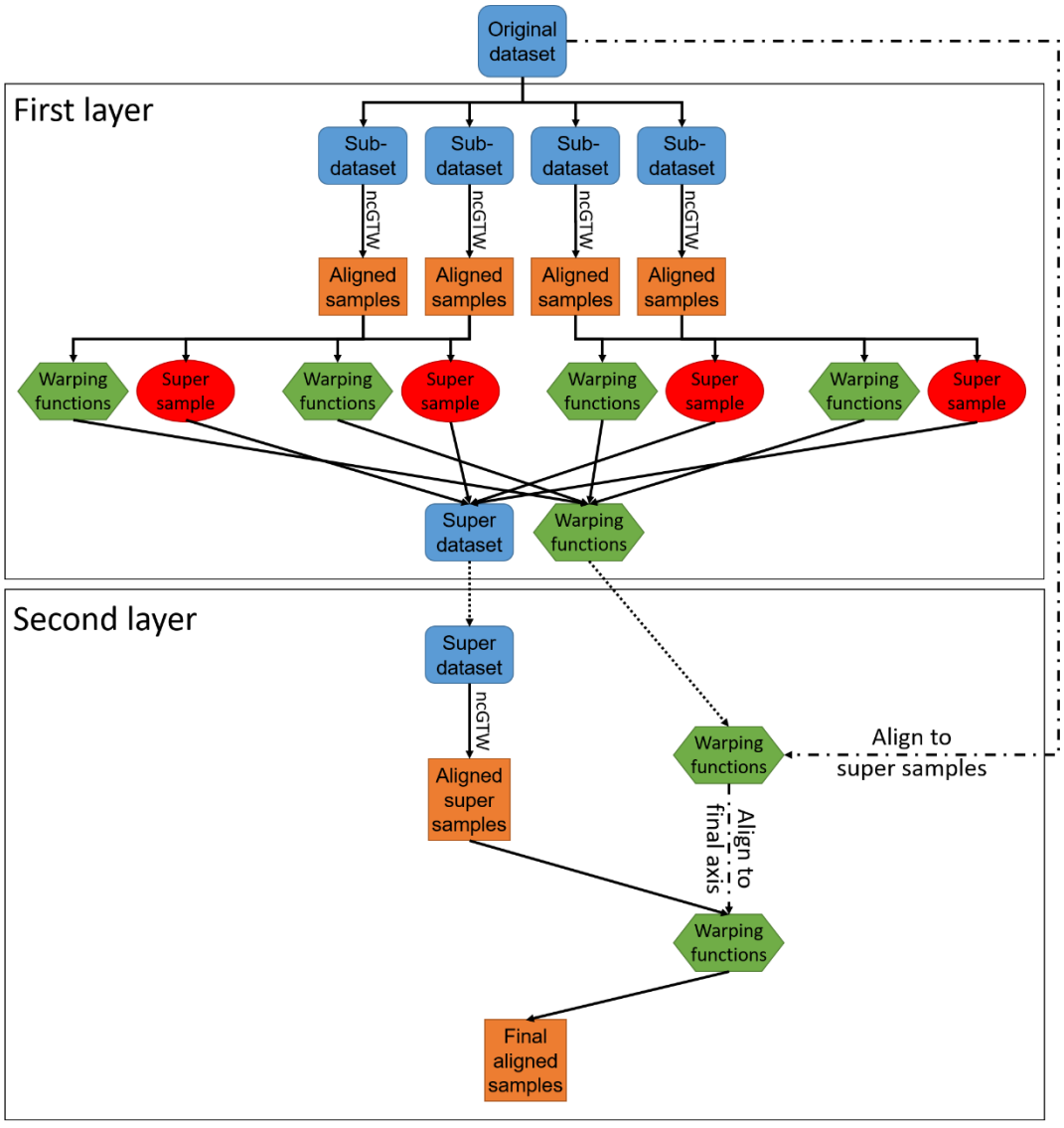

Figure 3.3 Diagram of two-layer ncGTW.

3.2.2.2 Correction of potential peak distortion

Peak distortion is a potential problem associated with DTW-based alignment, that is, while warping functions are optimally estimated by network flow algorithm, the shape of peaks may be altered. In other words, DTW-based methods do not guarantee one-to-one mapping for the points in the peak area (may be many-to-one or one-to-many), so the peak may become broader or narrower with shape changing. The reason is that DTW-based methods are trying to map the points

with similar intensity. When aligning peaks with different shapes, the peak in the sample may be distorted to fit the shape of the peak in the reference. The requirement of maintaining the shape of peaks cannot be easily incorporated into the cost functions of existing methods and ncGTW alike. Therefore, while warping functions are optimally estimated, the shape of peaks may still be altered. To avoid such distortion, based on multiple alignment of all samples, we first extract the information of each peak from XCMS. Within the starting and ending points of each peak, only the apex point will be aligned by the warping function $\Phi_{i,c}$, and the rest of the points are simply shifted by the same amount as the apex. In other words, the whole peak area will be aligned together with shifting the same amount, so the shape of the peak will be the same after alignment. For the samples with peaks undetected by XCMS, we borrow information from samples with detected peaks. The apices on virtual reference aggregated by samples with no missing peaks are first identified, and then a representative peak can be constructed from them. The apex of the representative peak is the median of the locations of the detected apices, and the width is the median of the detected peak widths. The range and apex location of the representative peak is mapped back to each sample with undetected peaks by reversely applying the warping function. Thus, the samples with undetected peaks will be finally aligned without peak distortion. An illustrative experimental result is given in **Fig. 6c-d** using the MESA dataset, demonstrating the improved alignment by ncGTW where distorted peaks are effectively corrected.

3.2.3 Integration of ncGTW into XCMS

The ncGTW R package is developed currently as a complementary plug-in to XCMS tool. The unique features of ncGTW algorithm include misalignment detection, individualized warping function (over m/z bins), incorporation of structural information (run order), reference-free multiple alignment, and correction of peak distortion. The functional integration of ncGTW into XCMS also allows ‘iteration’ (or ‘interaction’) between ncGTW and XCMS. For example, information about misaligned features obtained by ncGTW may be used to guide parameter retuning in XCMS. Then, those misaligned features may be realigned by ncGTW and regrouped by XCMS. Moreover, using the correct locations of missing peaks specified by ncGTW warping functions, those missing peaks may be accurately retrieved by the peak-filling procedure in XCMS. The workflow of XCMS-ncGTW pipeline is given in Fig. 7.

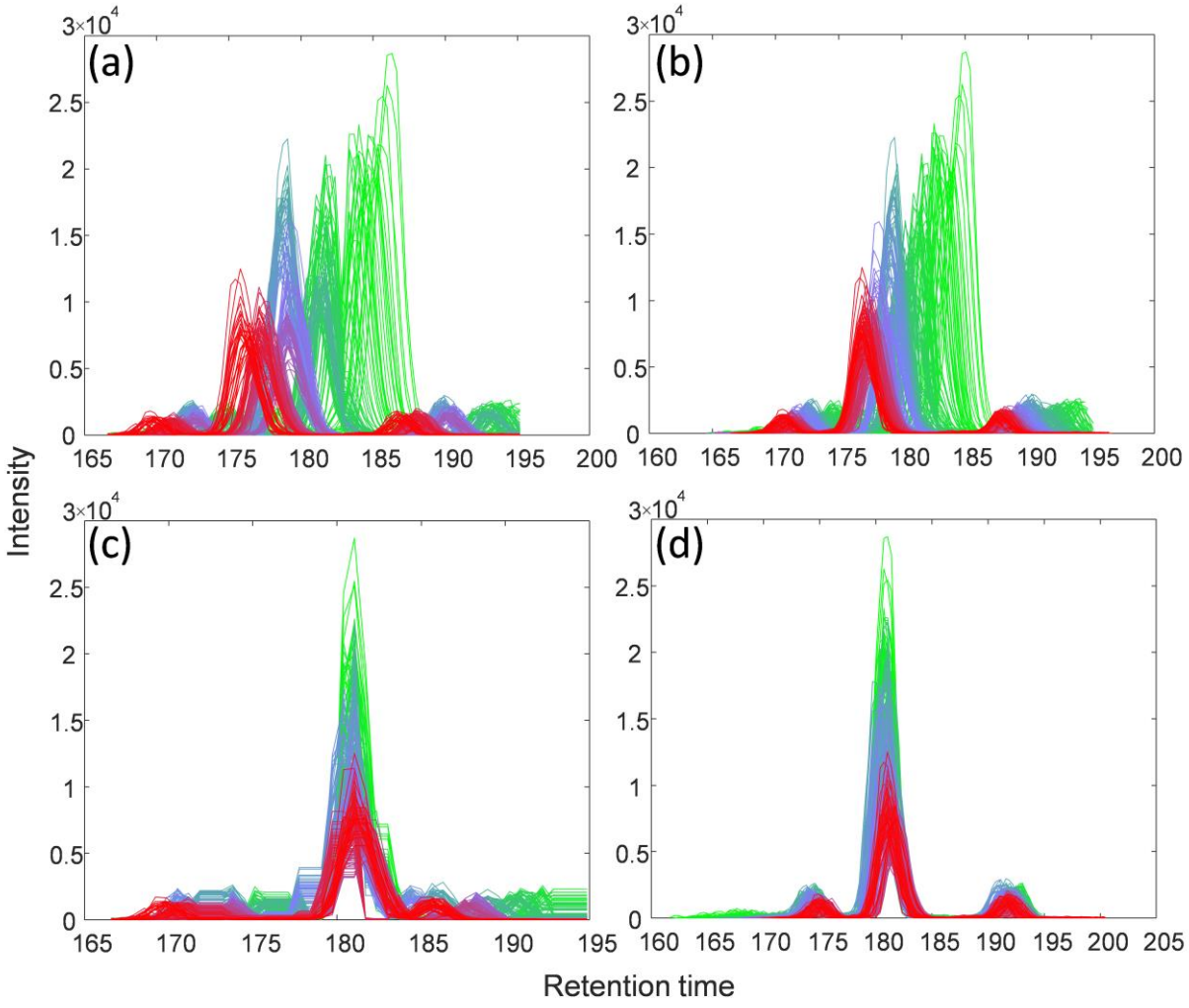

Figure 3.4 An illustrative experimental result on realignment and peak distortion correction by ncGTW, where a feature from the MESA dataset was initially misaligned by XCMS. The color mapping (green to blue and blue to red) corresponds to the sample index. (a) Raw LC-MS data associated with the feature of interest (before alignment). (b) The misaligned feature by XCMS that has been correctly detected and reported by the misalignment detection module of ncGTW package. (c) Realignment by ncGTW where apices are well aligned but with observable peak shape distortion. (d) Peak shape distortion is efficiently corrected by the post-processing module of ncGTW package.

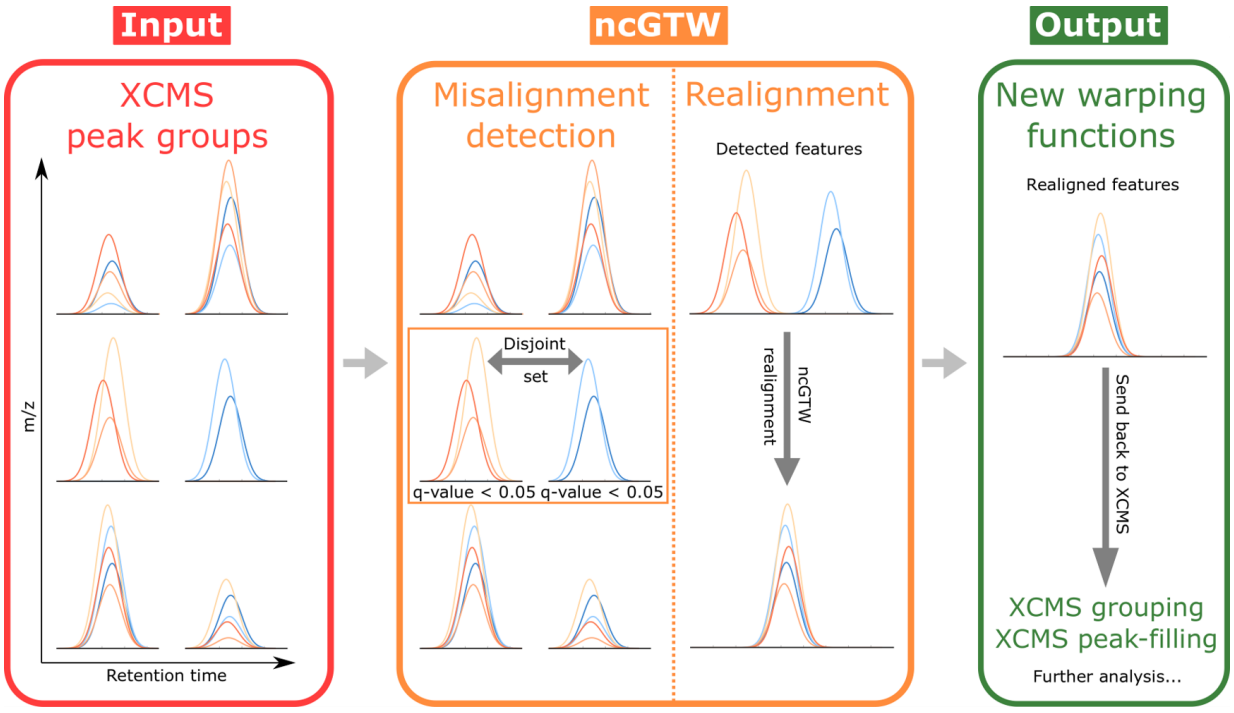

Figure 3.5 Workflow of ncGTW. As a plug-in to XCMS, ncGTW uses the grouping results provided by XCMS as the inputs (one lower resolution and one higher resolution, as explained in **Figure 3.2**). Then, ncGTW detects all misaligned features using the aforementioned criteria and performs realignment on these features. Lastly, ncGTW calculates final warping functions for each sample that can be sent back to XCMS for re-grouping or peak-filling.

3.3 Experimental results on real LC-MS datasets

3.3.1 Datasets

We apply ncGTW pipeline to two large-scale real metabolomics LC-MS datasets, namely the Rotterdam (The Rotterdam Study, The Netherlands) and MESA (The Multi-Ethnic Study of Atherosclerosis, USA) cohorts [63, 64], first to detect misalignment and then to realign those misaligned features.

Serum lipid profiling datasets were acquired by C8 reversed-phase liquid chromatography after serum protein precipitation using isopropanol, using protocols adapted from [65] and [66]. LC-MS profiles of the serum samples from the Rotterdam and MESA cohorts were generated using a Waters Acquity Ultra Performance LC system (Waters, Milford, MA, USA) for chromatographic

separation and a Xevo G2S Q-TOF (Waters, Milford, MA, USA) for mass spectrometry detection in positive ionization mode.

The Rotterdam dataset contains 1,000 study samples and 44 internal quality control (iQC) samples and the MESA dataset contains 1,977 study samples and 335 iQC samples. Each iQC sample is an aliquot of a pool of all the study samples, used to monitor and correct instrument performance in long runs. Because each of these two cohorts contains many samples (> 1k), the total time duration on data acquisition would be in the range of weeks - thus significant RT drift across experiments is expected. Indeed, on the iQC samples, the global warping function assumption made by XCMS is clearly and evidently violated in both Rotterdam (**Figure 3.1a**) and MESA (**Figure 3.1b**) datasets (using five samples from each dataset for demonstration). Moreover, in our observation, we estimate that there are around 3% of features which are misaligned due to the global warping function assumption in each dataset. Given that these two datasets were acquired from different cohorts at different labs with probably different experimental settings, this kind of misalignment may happen in any large dataset, and is not a rare occurrence.

In the subsequent experiments on both iQC and study samples, the major preprocessing steps (peak detection, RT alignment, peak grouping) are done by XCMS, and the ncGTW pipeline uses the default parameter settings.

3.3.2 Detection of misaligned features

Application of XCMS to the Rotterdam iQC samples alone generates total 1,872 features, among which 57 features are detected as potentially misaligned by the misalignment detection module in ncGTW package. With a closer visual inspection (performed independently by two MS experts), 41 features are confirmed as misaligned (true positives, **Figure 3.4b** as an example), and 16 remaining features are considered well-aligned (false positives, see **Figure 3.6** as an example). On Rotterdam study samples (excluding iQC samples), XCMS generates total 1,689 features, of which 45 features are detected as misaligned. Visual screening identifies 32 true positives and 13 false positives. On MESA iQC samples alone, XCMS generates total 1,951 features, of which 61 features are detected as misaligned. Visual inspection identifies 58 true positives and 3 false positives. On MESA study samples (excluding iQC samples), XCMS generates total 1,861 features, of which 49 features are detected as misaligned. Visual screening identifies 48 true positives and

1 false positive. All the p-value thresholds of the misalignment detection here are set as 0.05. While the false discovery rate is higher than theoretically expected, these false positives are mainly due to signal intensity fluctuation and will be eliminated in a post-realignment step.

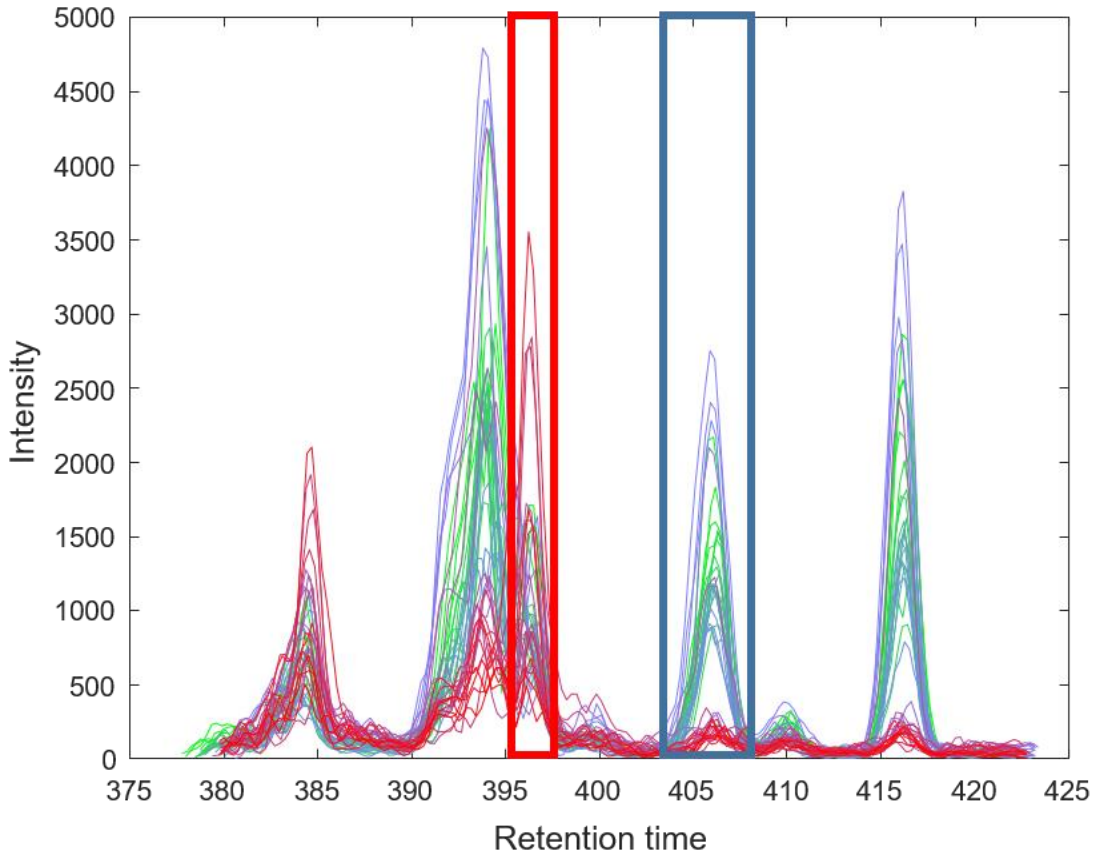

Figure 3.6 An XCMS aligned feature from the Rotterdam dataset as an example of false positives of the misalignment detection algorithm. One can see that all the peaks are aligned well. That is, unlike **Figure 3.4b** in the main article, no peaks are spread consecutively across RT, Thus, this detected feature is considered as a false positive. The sample indexes in the red box are 34, 36, and 40, and the p-value is 0.034. Apparently, there are more than three peaks in the red box, but only three of them are detected. Moreover, the sample indexes in the blue box are 1, 4, 9, 21, 23, and 24, and the p-value is 0.047. Again, there are many peaks that are not detected in the blue box. Both of the p-values are lower than the threshold and the index sets are disjoint, so this feature is reported as misaligned.

3.3.3 Realignment by ncGTW

We apply the ncGTW tool to realign the features flagged as misaligned in the previous step. While we have applied ncGTW to all study samples to show the scalability, our reports are focused on the results only on iQC samples mainly attributed to the feasibility of performing quantitative assessment. The evaluation criteria are the average pairwise correlation coefficient [49] and the average pairwise total overlapping area [67]. We use these two benchmark quantitative measures to assess the comparative performances by ncGTW and XCMS.

Comparative experimental results are detailed using two performance measures (**Figure 3.7**), where the features with circles represent true positives and the features with crosses represent the false positives. These comparative experimental results, on both Rotterdam and MESA datasets, consistently show that ncGTW effectively and accurately realigns those misaligned features. Specifically, ncGTW realignment on most true positives achieves much higher performance scores than that of XCMS, and on most false positives, produces comparable performance scores as expected (near the diagonal lines, **Figure 3.7**). Note that the default parameter setting of ncGTW in this step is purposely designed for the true positives, and probably not suitable for the false positives.

Experimental results on iQC samples also show that the two performance scores can well separate the true positives and false positives (**Figure 3.7**). Accordingly, via a post-alignment step we use these two performance scores to screen out true positives for further analysis. This strategy is also applicable to handling study samples.

3.3.4 Evaluation of ncGTW via post-realignment peak-filling performance

Accurate alignment of RT drift has significant impact on the performance of peak-grouping and peak-filling that define the features. In the XCMS pipeline, detected peaks are first grouped into features, and when there are undetected/missing peaks, peak-filling is then performed to retrieve those peaks. In our experiments, the coefficient of variation (CV) of intensity calculated over the samples, with and without ncGTW guided peak-filling, is adopted to assess the beneficial impact of ncGTW realignment [68].

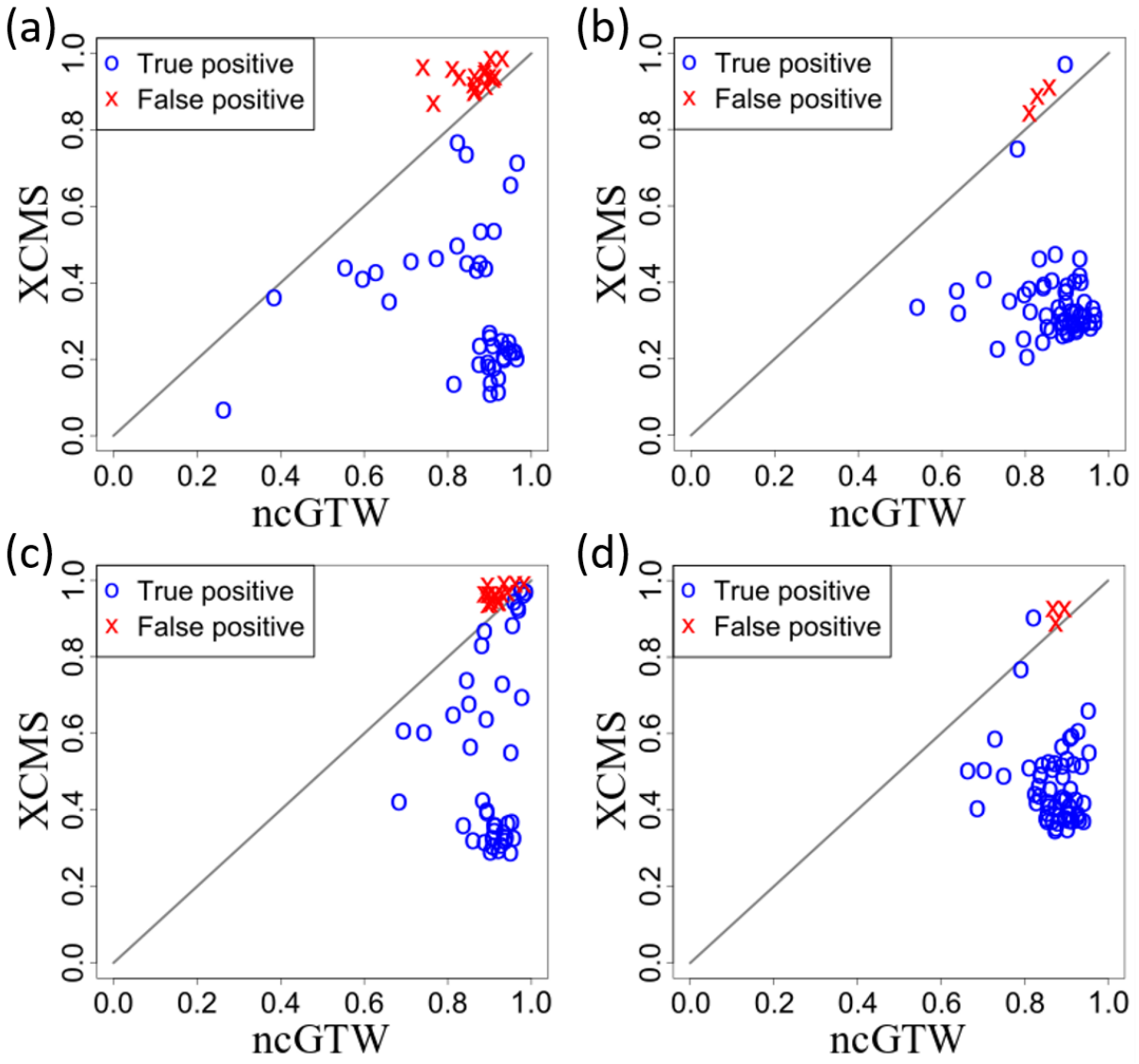

Figure 3.7 Application of ncGTW realignment method to Rotterdam and MESA datasets, where among the detected misaligned features, the blue circles represent true positives, and the red crosses represent false positives, respectively. (a) The average pairwise correlation coefficients on the Rotterdam dataset. (b) The average pairwise correlation coefficients on the MESA dataset. (c) The average pairwise total overlapping area on the Rotterdam dataset. (d) The average pairwise overlapping area on the MESA dataset.

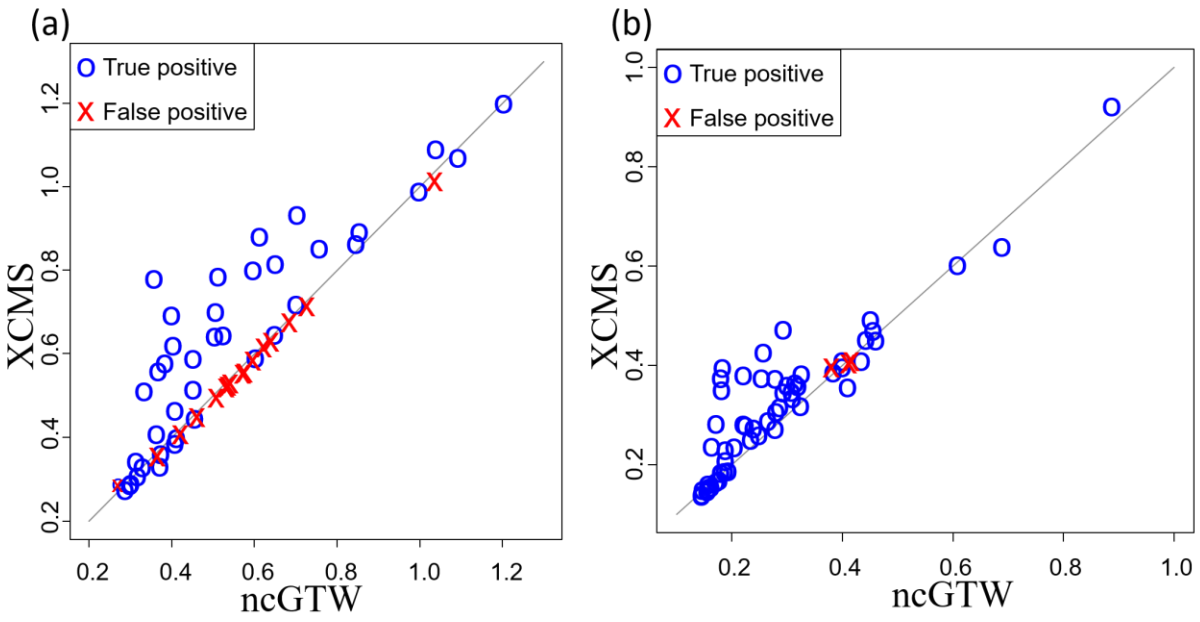

Figure 3.8 The comparisons of CV with versus without ncGTW realignment after the peak-filling step of XCMS. The blue circles represent the true positives, and the red crosses represents the false positives. (a) The CV comparison on Rotterdam dataset. (b) The CV comparison on MESA dataset.

Though we proposed that we can screen out the false positives by the two performance scores, we still include the false positives in the peak-filling step to observe the impact of the realignment on them. Post-realignment peak-filling results show that, measured by CV over the iQC sample features, ncGTW realignment consistently reduces the CV as compared with that derived from the initial XCMS alignment in both Rotterdam and MESA datasets (**Figure 3.8**). Note that here again the circles represent the true positives and the crosses represent the false positives. More importantly, post-realignment peak-filling supported by ncGTW has led to the significantly reduction of CV on many 'hard-to-define' features, demonstrating the beneficial contribution of ncGTW realignment to improved feature generation.

3.3.5 Biological or clinical importance of the detected misaligned features

When significant misalignment occurs, it is possible that some peaks will be incorrectly assigned to the wrong compound and the information from those peaks comingled with other compounds. In addition, in samples where a peak has been improperly aligned and therefore misclassified, the

true compound of interest may appear to be completely absent. These errors may obscure important information about molecular pathways and lead to biased inferences or corrupt statistical analyses concerning relationships between specific compounds and other traits associated with the samples. In **Table 2-1**, we provided the detailed annotation for the five features associated with underlying compounds that were initially misaligned in MESA and Rotterdam datasets and later corrected by ncGTW (measured by the reduction in CV values). For the whole list of annotation, please refer to **Table 3-2**. Indeed, each of these compounds plays important roles in specific and clinically relevant metabolic pathways. For instance, Alpha-tocopherol-glucuronide is a conjugation metabolite of the biological anti-oxidant alpha-tocopherol (vitamin E); Ganglioside GM3 is a phospholipid found predominantly in cell surfaces, and this molecule has important role in cell-to-cell recognition [69]. Phosphatidylinositol is a glycerophospholipid and an important component of cell membranes, found predominantly in the inner surface of the cell membranes [70]; Choline is an important precursor of Phosphatidylcholine and Sphingomyelin phospholipids, and also the precursor of the neurotransmitor acetylcholine and participates as methyl group donor in several biochemical reactions [71]; Lysophosphatidylinositol is a metabolite of phosphatidylinositol resulting from the cleavage of one the two fatty acyl chains by the action of a phospholipase-type A enzyme [72]. Given the important biological roles of these compounds, potential misclassification or corruption of the signals concerning these metabolites, due to misalignment, could seriously hinder the ability to understand important aspects of disease biology.

Table 3-1 Annotation details on the representative features that are associated with biologically important compounds with their m/z and RT positions, where the improvements in the targeted ncGTW realignment are quantitatively measured by the reduction in CV values.

Dataset	m/z	RT	Metabolite annotation	CV XCMS	CV ncGTW
MESA	629.4	184.9	Alpha-tocopherol-glucuronide	0.29	0.25
MESA	844.6	275.6	Ganglioside GM3	0.39	0.29
MESA	879.5	293.8	Phosphatidylinositol	0.39	0.19
Rotterdam	104.1	24.2	Choline	0.66	0.51
Rotterdam	342.3	111.5	Lysophosphatidylinositol	0.71	0.41

Table 3-2 The annotations of features in MESA and Rotterdam datasets with their m/z and RT positions.

Dataset	m/z	RT	Metabolite annotation
MESA	431.4	184.9	alpha-tocopherol-glucuronide
MESA	629.4	184.9	alpha-tocopherol-glucuronide
MESA	430.4	187.1	alpha-tocopherol-glucuronide
MESA	1176.7	275.1	Ganglioside GM3
MESA	1175.7	275.2	Ganglioside GM3
MESA	520.5	275.6	Ganglioside GM3
MESA	844.6	275.6	Ganglioside GM3
MESA	599.5	293.3	Phosphatidylinositol
MESA	879.5	293.8	Phosphatidylinositol
MESA	575.5	293.9	Phosphatidylinositol
MESA	857.5	293.9	Phosphatidylinositol
MESA	576.5	293.9	Phosphatidylinositol
MESA	881.5	294.4	Phosphatidylinositol
MESA	903.5	294.6	Phosphatidylinositol
MESA	882.5	294.7	Phosphatidylinositol
MESA	600.5	294.9	Phosphatidylinositol
MESA	907.5	307.5	Phosphatidylinositol
MESA	625.5	307.8	Phosphatidylinositol
MESA	908.5	332.6	Phosphatidylinositol
MESA	907.5	333.6	Phosphatidylinositol
MESA	886.5	334.0	Phosphatidylinositol
MESA	885.5	334.0	Phosphatidylinositol
MESA	603.5	334.3	Phosphatidylinositol
MESA	604.5	334.3	Phosphatidylinositol
MESA	605.5	334.4	Phosphatidylinositol
MESA	910.5	334.4	Phosphatidylinositol
MESA	909.5	334.4	Phosphatidylinositol
MESA	931.5	334.5	Phosphatidylinositol
MESA	932.5	334.5	Phosphatidylinositol
MESA	628.5	334.6	Phosphatidylinositol
MESA	341.3	334.7	Phosphatidylinositol
MESA	342.3	334.7	Phosphatidylinositol
MESA	627.5	334.7	Phosphatidylinositol
MESA	269.2	335.2	Phosphatidylinositol
MESA	925.5	335.3	Phosphatidylinositol
MESA	911.6	338.7	Phosphatidylinositol
MESA	629.5	344.0	Phosphatidylinositol
MESA	630.6	346.8	Phosphatidylinositol
Rotterdam	104.1	24.2	Choline
Rotterdam	342.3	111.5	Lysophosphatidylinositol
Rotterdam	623.3	112.4	Lysophosphatidylinositol
Rotterdam	341.3	112.5	Lysophosphatidylinositol
Rotterdam	583.3	115.4	Lysophosphatidylinositol
Rotterdam	909.5	284.1	Phosphatidylinositol
Rotterdam	603.5	285.0	Phosphatidylinositol
Rotterdam	604.5	285.0	Phosphatidylinositol
Rotterdam	627.5	285.0	Phosphatidylinositol
Rotterdam	628.5	285.0	Phosphatidylinositol
Rotterdam	885.5	284.8	Phosphatidylinositol

3.4 Discussion

Feature-based and profile-based alignment methods are complementary to each other. Because of high efficiency on large datasets, the majority of existing alignment methods are feature-based. However, due to the challenging nature of accurate peak detection particularly when there are some missing peaks and significant RT drift, misalignment occurs on some features that also causes incorrect peak-grouping and/or peak-filling. Here we develop the ncGTW method to first detect misaligned features and then to realigned them. One unique advantage of ncGTW is the incorporation of structural information (i.e run-order) in our multiple alignment method. To the best of our knowledge, most existing alignment methods have overlooked structural information related to experimental design and batch duration. Moreover, the novel design of a reference-free multiple alignment strategy and utility of individualized warping functions across m/z bins all contributed to produce superior performance of ncGTW.

Our approach is built on GTW, a recent extension of DTW to multiple pairs. GTW retains all the desirable properties of DTW such as monotonicity of time shifts and polynomial efficient solution, and yet flexibly models any graph-encoded structure among pairs. However, GTW cannot be directly applied to solve the multiple alignment problem. GTW takes multiple pairs of samples as input and finds alignment for each pair with consistency between pairs considered. The problem considered in this paper takes multiple samples as input and aims to find consistent alignment among all samples. Though we can manually specify one certain sample as a reference to construct pairs of samples for input of GTW, to our knowledge, there is no established method that can always help the user to identify which sample is the most suitable one as the reference. Moreover, it is likely that none of the samples contains enough information to serve as a good reference. Thus, ncGTW is a significant improvement of GTW, since ncGTW can model all samples simultaneously and deal with the multiple alignment problem without manually setting a reference.

Currently, there is no consensus method to detect misalignment other than simple visual inspection. Thus, misalignment is often undetected and therefore may not even benefit from the application of (substandard) conventional alignment methods. Specifically designed to address the problem of misalignments complementary to existing alignment software tools, our proposed ncGTW method focuses on correcting only those misaligned features. Toward this objective with high efficiency,

the ncGTW package includes a unique functional module that specifically aims to detect misaligned features. We validated ncGTW using both realistic synthetic data and internal quality control samples. The performance of ncGTW is particularly attractive when processing large-scale datasets consisting of hundreds or thousands of samples, because the RT drifts between distant samples may be significant and warping functions over different m/z bins are not guaranteed to be the same (**Figure 3.1**). Explicit incorporation of the RT structural information by the ncGTW method helps to achieve accurate realignments on misaligned features. While we have only demonstrated ncGTW as a plug-in package to XCMS, in fact, two major functions of the ncGTW tool can serve as a plug-in jointly or independently to other alignment tools as well, and thus have broad applicability, including to other spectral data types beyond LC-MS.

Regarding the peak distortion problem associated with warping functions, there are at least two potentially effective solutions. First, the peak information provided by XCMS can be utilized by ncGTW to correct peak distortion as discussed in sub-section 3.2.2.2. Second, parameter settings in ncGTW can be adjusted or optimized to reduce the likelihood of peak distortion. Note that the current ncGTW tool package already includes a peak distortion correction module, and our experiments have also shown that interim peak-distortion correction can help optimize ncGTW parameter settings that will in turn reduce the likelihood of peak distortion.

In our misalignment detection step in sub-section 3.3.2, we have observed that the false positive rate in the Rotterdam dataset is much higher than the theoretical threshold of 0.05. By a closer look at the peak detection results, we found that many peaks were actually missed by XCMS, mainly due to significant yet irregular signal intensity fluctuating over the course of data acquisition as shown in **Figure 3.6** with all samples. Considering such relatively higher false positive rate does not create significant computational burden on ncGTW yet may be uncontrollable, we have opted to first ‘accept’ these false positives and then screen them out at a later stage.

3.5 Summary

In conclusion, we develop a software package, which can help any feature-based LC-MS alignment method to detect the misalignment. After the detection, we propose a new feature-profile hybrid alignment algorithm, ncGTW, to realignment the misalignments. ncGTW is an

improvement of GTW, and ncGTW does not need a certain reference when aligning. ncGTW can utilize the structure information of the dataset, so that it can easily align the sample in only one m/z bin accurately. To correct the peak distortion, ncGTW can utilize the information from the preprocessing feature-based tools to recover the peak shape, and this step makes ncGTW as a hybrid alignment method. We evaluate ncGTW by the average pairwise correlation coefficient and the average pairwise overlapping area. Both of them show ncGTW improves the alignment. After sending the realignment result back to XCMS, the peak-filling step validates that with ncGTW realignment, the further analysis would be more accurate.

Chapter 4

Unsupervised Deconvolution - CAM

4.1 Introduction

The primary objective of mathematical deconvolution is to computationally detect subtype-specific markers, determine the number of constituent subtypes, calculate subtype proportions in individual samples, and estimate subtype-specific expression profiles [26]. Complex tissues – where formation and remodeling require productive interactions between multiscale subtypes – are intrinsically dynamic. Thus, supervised methods are poorly suited to finding molecularly distinctive subtypes that are subtle, condition-specific (their distinctive signatures change when the cells are present in different microenvironments), or previously unknown [31, 32, 73]. Unsupervised deconvolution methods, which are based on regularized matrix factorization, are an appropriate tool to characterizing the molecular landscape of complex tissues. Supported by advanced machine learning algorithms and proven theorems, unsupervised deconvolution methods can decompose the mixed molecular signals into many latent variables; these subtypes are biological interpretable and functionally enriched [23, 74-76]. However, the question remains as to whether it is possible to develop fully unsupervised deconvolution methods that can concisely define molecular latent variable ecosystem of complex tissues.

Recently, Convex Analysis of Mixtures (CAM) is proposed as a fully unsupervised method, which works by detecting the vertices of the scatter simplex geometrically, i.e., determining the multifaceted simplex that most tightly encloses the globally measured expression mixtures. Subsequently, CAM identify the molecular markers residing at the vertices, and estimate the proportions and specific expression profiles of constituent subtypes [23]. The number of subpopulations present is determined by the newly-derived minimum description length (MDL) criterion. Ideally, a molecularly distinctive subtype would contain molecular signatures (molecular markers) that are exclusively expressed in the cognate cell or tissue subtype of interest while in no others. Importantly, CAM deconvolution pipeline requires no a priori information on the number, signatures, or compositions of the subtypes present in heterogeneous samples, and does not require

the presence of pure subtype samples [74, 77]. This advantage is significant in that CAM can achieve all of its goals using only a small number of heterogeneous samples, and provides a powerful means to distinguish among phenotypically similar subtypes.

Comparing with other unsupervised methods, such as nICA and NMF, CAM has the advantage that its assumption is biologically plausible. However, the computation complexity makes CAM could not deconvolve the mixtures with more than 10 subtypes efficiently. Also, the pre-processing step of CAM is not full controllable and sometimes unstable. Thus, in this chapter, we first briefly introduce CAM, and then analyze its shortages and provide the improvements in the following sections.

4.2 Problem modeling

Here we formulate the deconvolution task as a blind source separation (BSS) problem. Supposing the number of samples (observations) is *M*, number of genes (features) is *N*, and number of cell types (sources or subtypes) is *K*, with several assumptions for the application on the tissue heterogeneity deconvolution, the problem can be stated as

$$\mathbf{X} = \mathbf{AS}, \tag{4.1}$$

where $\mathbf{X}$ is a $M \times N$ gene expression matrix, where each row corresponding to a sample (observation), and each column corresponding to a gene (feature). $\mathbf{A}$ is a $M \times K$ mixing matrix, where each row corresponding to a sample, and each column corresponding to a cell type (source). That is, each row in $\mathbf{A}$ represents the portion of each cell type in a sample. $\mathbf{S}$ is a $K \times N$ source matrix, where each row corresponding to a cell type, and each column corresponding to a gene. Thus, each row in $\mathbf{S}$ represents the expression in a cell type. Our target is to find out $\mathbf{A}$ and $\mathbf{S}$ separately. The only information we have is the observation matrix $\mathbf{X}$, so this is a blind source separation problem.

4.2.1 Convex Analysis of Mixtures

According to the biological properties of genomics data, CAM has major three assumptions to help solve the BSS problem:

(1) The number of samples is larger than the number of cell types ($M > K$).

(2) The mixing matrix **A** is full column rank.

(3) Each cell type is well-grounded, that is, in each cell type, there exists at least one gene which only expresses in this cell type (marker genes).

(4) The values in **A** and **S** are all non-negative, and so does **X**.

The first assumption is easy to be satisfied, since in the real applications, it is common that we will have tens of samples, and the cell types are often less than ten. The second assumption is needed, or some of the cell types are indistinguishable, since they could be expressed by the linear combinations of the other cell type mathematically. The third assumption is the core of CAM, and we know that there are always "marker genes" for a cell type, so this assumption is also satisfied. The fourth assumption is usual for the biological data, since when measuring RNAs, metabolites, and so on, their quantities should always be positive, and so does the mixing portions of cell types.

If we view each column in **S** as a vector, then these *N* gene points are in a *K*-dimension space. With assumption 3, we can assume that for each dimension (each subtype), there is at least one gene as the Cartesian vector (marker gene), though the length is not one. Thus, if we normalize all the genes by their expression sum (the column sums are all 1 after normalization), all the genes are the convex combinations of the marker genes from each subtype. That is, the normalization is corresponding to project the genes onto a hyper plane, where the genes form a (*K*-1)-simplex, and the marker genes are on the vertices of the simplex (assumption 2 and 3). Thus, if we multiply **A** to **S**, which is similar to rotate and project the (*K*-1)-simplex into a higher dimensional space (assumption 1 and 4), and the vertices of the simplex are the column vectors of **A**. That is, after projection (normalization), the column vectors (genes) in **X** can be viewed as convex combination of the column vectors in **A**, so the vertices of the simplex are exactly the vectors in **A**. Thus, after projecting the column vectors (genes) of **X**, we just need an algorithm, such as Quickhull [39], to find the convex hull, and points on the convex hull are the candidates of the vertices. The relation of the simplex of **S** and **X** is illustrated in **Figure 4.1**.

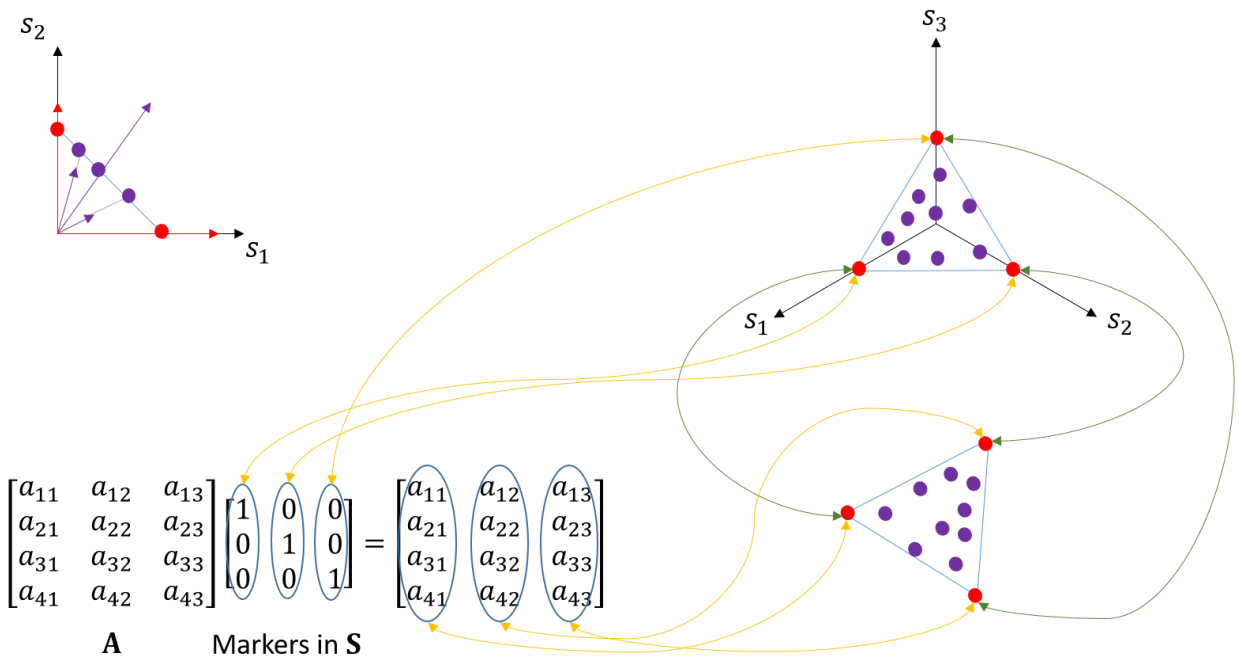

Figure 4.1 Illustration of the relation between S and X (K = 3 for example). The upper left corner is an example of sum-to-1 normalization in 2D space, which project the genes onto 2 1-simplex (a line). The reds are the marker genes, and the purples are the normal genes. The right part is the simplex in 3D space (a triangle). After multiplying with A, the simplex is rotated and projected into a higher dimension space (four in this figure), and the vertices (markers) become the column vector of A, as shown in the figure.

However, with noise, the dimension of the convex hull would be high and noisy. That is, there would be too many vertex candidates (false positives) which are due to noise. Also, even the marker genes at the same vertex may be split into many candidates due to noise. Thus, to suppress noise, CAM will first perform K-means clustering or affinity propagation clustering (APC) to suppress the noise. In real applications, K-means clustering is always applied since there may be dozens of thousands of genes, where the computation time of APC would be extremely large.

After finding Q candidates of the vertices of the simplex, supposing we know the source number is K, there is C_K^Q combinations to form a (K-1)-simplex. To decide which combination is the best one, CAM use reconstruction error to evaluate each combination. That is, for each combination, CAM set it as $\mathbf{A}'$, and with $\mathbf{X}$, we can estimate $\mathbf{S}'$ with non-negative least square (NNLS,

assumption 4). With $\mathbf{A}'$, $\mathbf{S}'$, and $\mathbf{X}$, we can have the reconstruction error of this candidate combination:

$$reconstruction\ error = \sum_{j=1}^{N} \|\boldsymbol{x}(j) - \mathbf{A}'\mathbf{s}'(j)\|_2^2, \tag{4.2}$$

where $\mathbf{A}'$ is formed by the combination of the candidates (*K*-columns), $\boldsymbol{x}(j)$ denotes the *j*th column vector of $\mathbf{X}$, and $\mathbf{s}'(j)$ denotes the *j*th column vector of $\mathbf{S}'$. However, it is a combinatorial optimization problem, so computation time would be extremely large if *Q* is not small. Also, since the NNLS is need for reconstruction error, the number of genes also affect the computation time. Here, CAM uses a heuristic way to find the best combination. First, to simplify the NNLS step, CAM builds a new matrix $\mathbf{X}'$, which is formed by all the vertex candidates, so the dimension of $\mathbf{X}'$ is $M \times Q$ (the dimension of $\mathbf{X}$ is $M \times N$). *N* could be more than 1000 times larger than *Q* (depends on the cluster number setting in K-means). Also, with $\mathbf{A}'$ and $\mathbf{X}'$, we can apply NNLS to obtain $\widehat{\boldsymbol{S}}$. Next, to measure the combination with $\mathbf{X}'$, CAM computes the reconstruction error on $\mathbf{A}'$ and $\mathbf{X}'$ as

$$margin\ of\ error = \sum_{j=1}^{N} \|\boldsymbol{x}'(j) - \mathbf{A}'\hat{\boldsymbol{s}}(j)\|_2^2, \tag{4.3}$$

where $\hat{\boldsymbol{s}}(j)$ denotes the *j*th column vector of $\hat{\mathbf{S}}$. In this way, it would be 1000 times faster than computing the original reconstruction error. However, since the column vectors in $\mathbf{X}'$ are after projection (comparing the ones in $\mathbf{X}$) and do not include all the genes, the best margin of error does not mean the best reconstruction error. To mitigate this problem, after computing the margin of error of each combination, CAM will further compute the reconstruction error of the top 200 (a parameter in CAM) combinations decided by the margin of error. Then, the best one is decided in the 200 combinations with the lowest reconstruction error. That is, the optimal (*K*-1)-simplex is founded.

Then, the last problem is how to decide the value of *K*, as a model selection problem. CAM applies Minimum Description Length (MDL) [78] to find the optimal *K*. MDL is a widely-accepted method, which consider the idea of information theory. In general, the idea of MDL is to find the

balance between low reconstruction error and low model complexity. The formula of MDL adopted by CAM is:

$$MDL(K) = \frac{NM}{2}\log\left(\frac{1}{NM}\sum_{j=1}^{N}\|\boldsymbol{x}(j) - \mathbf{A}'\mathbf{s}'(j)\|_2^2\right) + \frac{(K-1)M}{2}log(N) + \frac{KN}{2}log(M)\,. \quad (4.4)$$

The first term corresponds to the model fitting error (reconstruction error), and the second and the third terms corresponding to the model complexity. That is, MDL is trying to find a model (*K*) where the reconstruction error is low (first term), and the number of sources (*K*) is also low (second and third terms). We can also view the second and the third terms as penalty terms, which can avoid overfitting problem. With MDL, unlike nICA or NMR, CAM is a fully unsupervised deconvolution method, since the value of *K* could be estimated by CAM automatically, which is not true for the other methods.

4.3 Methods

As mentioned in the previous sections, as an unsupervised deconvolution method, there are still some disadvantages in CAM: unstable pre-processing result due to K-means clustering, high time complex when finding convex full, and also high time complex when identifying the optimal simplex. Thus, in the following sub-sections, we will discuss how to solve these problems and provide solutions one by one.

4.3.1 Radius-fixed clustering

Though CAM provides two clustering methods in the package, K-means clustering is usually selected. The reason is that when data contains thousands of genes (features) with dozens of samples, the computation time of APC is not feasible. However, as we know, there are some disadvantages of K-means clustering. For example, K-means clustering needs initialization, and which does not guarantee global optimum [79]. That is, if K-means clustering may give different results with different initializations. Since the clusters from K-means clustering are for simplex vertex identification in CAM, CAM may produce unstable deconvolution results. Also, the reason CAM needs a clustering method for pre-processing is to suppress the noise before finding the convex hull. However, since we cannot control the size of each cluster from K-means clustering,

the noise may be suppressed at varying degrees in different clusters, which may distort the shape of the convex hull. Though we can change the number of clusters to affect the size of clusters in K-means, it is not guarantee that the size of all clusters will change with the number of clusters, and the size of all cluster may be still not the same usually. Also, it is well-known that K-means clustering is sensitive to the outliers, so if there are many outliers, K-means clustering is not a good choice.

For example, there is a dataset which contains three tissues (sources), so the genes could be projected onto a 2D plane to form a 2-simplex (triangle), as shown in **Figure 4.2**. One can see that there are some outliers around the simplex.

If we apply K-means clustering on the data and set cluster number as 50, as shown in **Figure 4.3**, it is obvious the sizes of the clusters are not similar. Also, the outliers are included in some large clusters, so the mean of these clusters may be shifted. If we increase the number of the clusters, may be the outliers would be clustered in a cluster with very few members, and then we could remove these clusters to suppress the effect of outliers. However, if we increase the number of clusters to 100, as shown in **Figure 4.4**, the outliers are still included in some large clusters. Also, the sizes of the clusters are still not very similar.

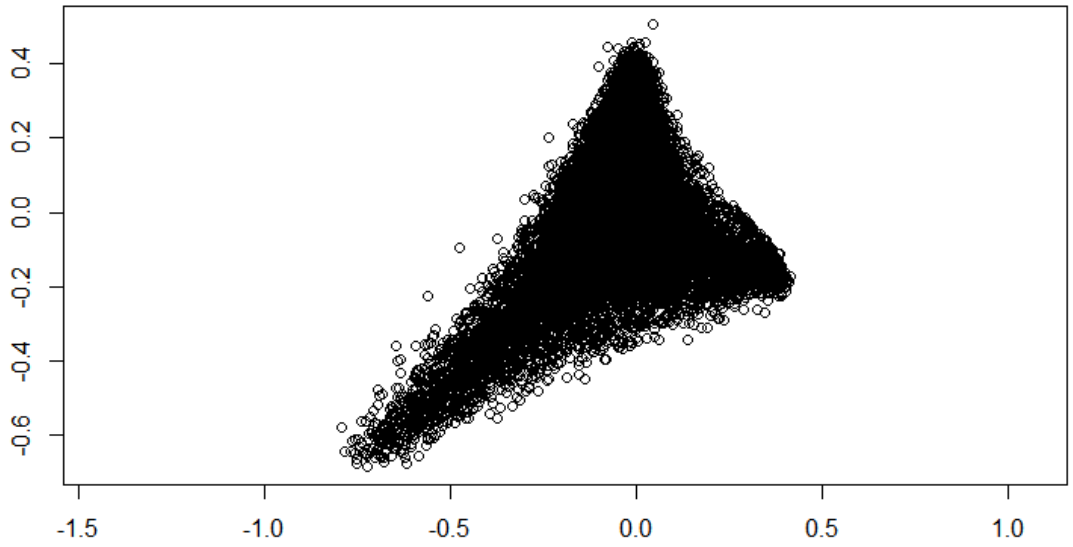

Figure 4.2 a 2-D simplex from a real dataset. ~30,000 points (genes) projected onto a 2D plane. Since there are three sources in the dataset, the points form a 2-simplex (triangle). There are some outliers around the simplex.

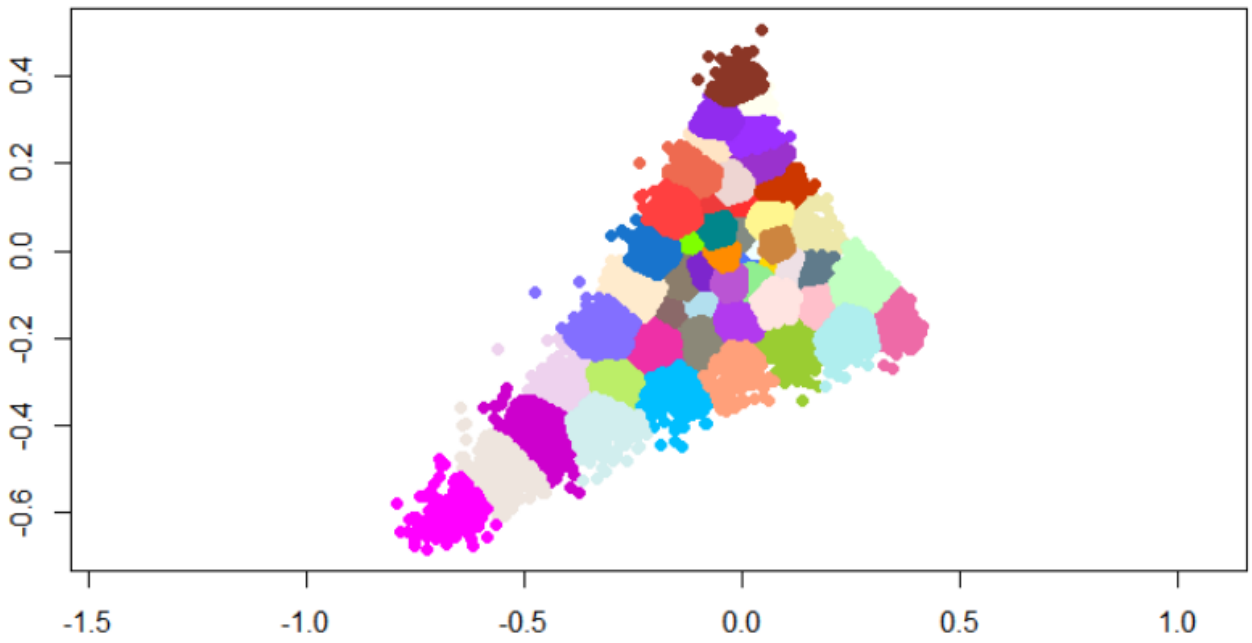

Figure 4.3 K-means clustering: 50 clusters. The sizes of the clusters are not very similar, and the outliers are included in some large clusters.

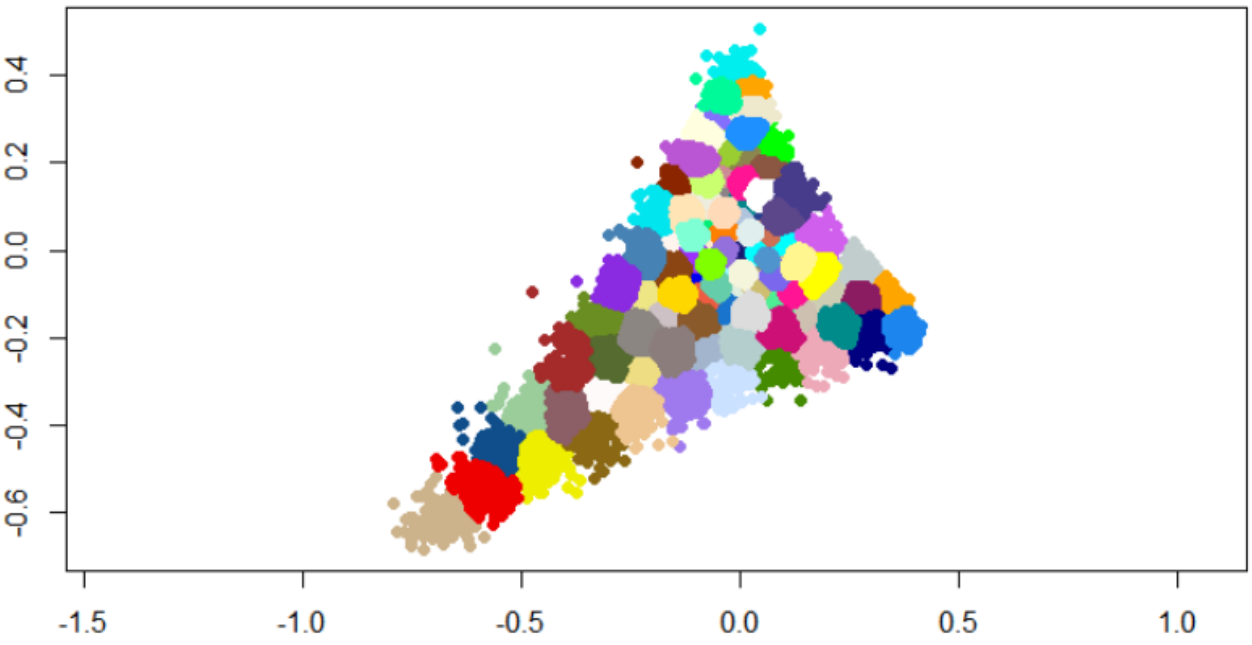

Figure 4.4 K-means clustering: 100 clusters. The sizes of the clusters are still not similar, and the outliers are still included in the large clusters.

To solve the problems of K-means clustering, we want a method which can control the size of each cluster no more than a certain threshold, and the threshold can be control by a parameter easily. Also, it should be a deterministic method so that there is no randomness in the result. Moreover, it would be great that if this method is not sensitive to the outliers. Here we propose another way for clustering, called Radius-fixed clustering. The idea of radius-fixed clustering is that first we need to decide the value of the threshold, then remove the clusters satisfying some requirements one by one from the dataset until there is no clusters which meet the requirements. The threshold here we selected is cosine similarity, since Euclidian distance is distorted after projection, but cosine similarity will not change. The steps of radius-fixed clustering are as following and also illustrated in **Figure 4.5**:

1. Set a radius (though we choose cosine similarity for CAM, in other applications, any other kinds of distance or similarity are acceptable).

2. For each gene (point), compute how many genes are within the radius. That is, set each gene as the center of a hyper ball, the radius of which is the one set in step 1, and then compute how many genes are within the hyper ball.

3. Find out the hyper ball with the greatest number of genes inside, and then remove all the genes in the hyper ball from the data.

4. Repeat Step 2 and 3 until there is no gene left or the number of genes in the hyper balls are below a certain value.

5. All the hyper balls removed from the data are the clustering results by radius-fixed clustering.

Since we have set the “radius” of each cluster at the beginning, it is guaranteed that no cluster would be larger than the radius. Thus, we can control the size of each cluster efficiently, or at least we can make sure that when computing the mean of each cluster, there is no gene which is far from the center of the cluster. Also, if we set a threshold for the number of genes in each cluster, there may be some genes which do not belong to any cluster. Since the target of the clustering step

is to suppress the noise, it is OK that if there are some genes which are not clustered. That is, the target of radius-fixed clustering is to "down-sampling" the simplex for de-noising, not for "clustering" all the genes to clusters. Also, if these "not clustered" genes are outliers, actually it would even make our de-noising more accurate. Moreover, in biology, genes always work together (pathways) [80]. That is, a bunch of genes will have similar patterns since they have related functions. Thus, when using radius-fixed clustering, setting a threshold for member number of a cluster can not only remove the outliers, but also follow the biology principle.

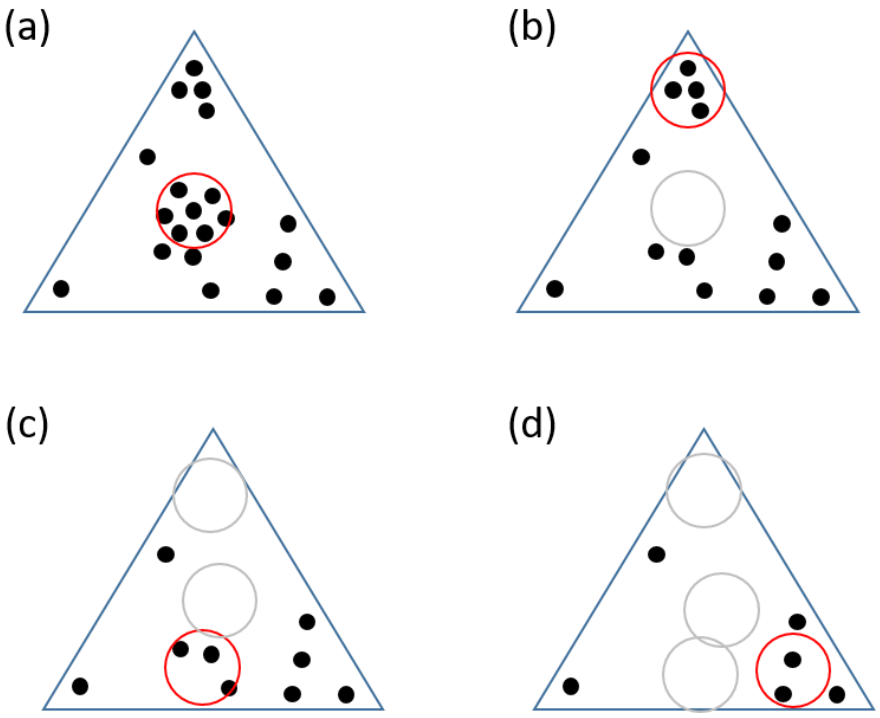

Figure 4.5 illustration of radius-fixed clustering. (a) After deciding the radius (the radius of the red circle), we can set each gene as center, and then find which circle contains the greatest number of gene (red circle, seven points). (b) Since the points in the red circle in (a) are removed, the next one contains four. (c) The next one contains three points. (d) The final one contains two, if we set all the clusters should contain at least two points.

As shown in **Figure 4.6** and **Figure 4.7**, the cluster sizes of radius-fixed clustering are more similar. Also, if we set the number of members in each cluster should be no less than 5, most of the outliers are not included. Thus, with radius-fixed clustering, the outlier removal and clustering can be done at the same time, and the idea follows the biology principle.

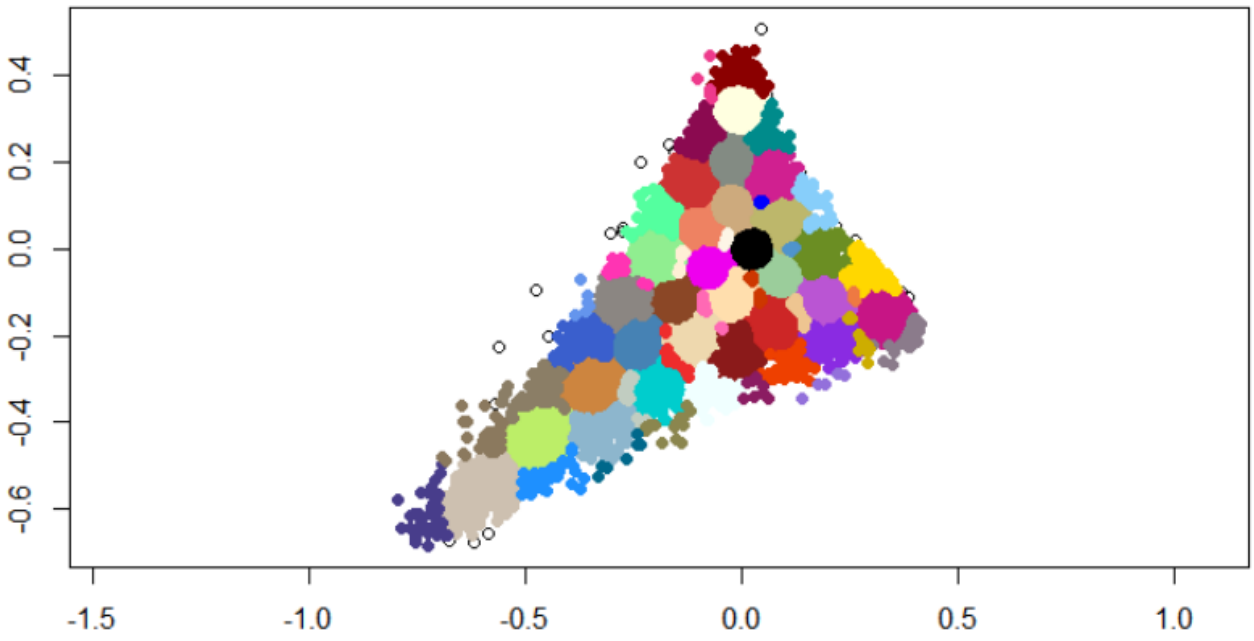

Figure 4.6 Radius-fixed clustering with cosine similarity as 0.998 (59 clusters). Comparing with K-means clustering with 50 clusters, the sizes of clusters are more similar. Also, some outliers are not clustered.

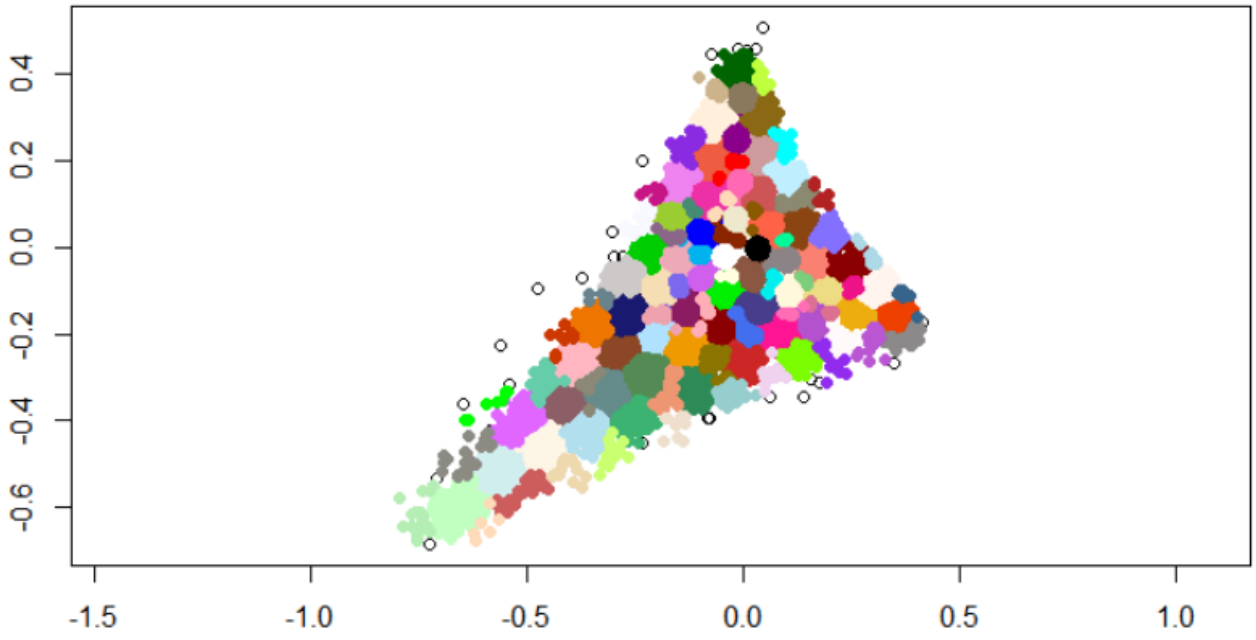

Figure 4.7 Radius-fixed clustering with cosine similarity as 0.999 (106 clusters). Again, comparing with K-means clustering with 100 clusters, the sizes of clusters are more similar. Also, more outliers are not clustered.

4.3.2 Convex hull identification

The core idea of CAM is to identify the marker genes, which are located on the vertices of the simplex. However, it is hard to find the optimal simplex directly, so CAM finds the convex hull of the data first. That is, the optimal simplex should be in the convex hull, and the vertices of the simplex should be included in the extreme points of the convex hull. Thus, after clustering, the next step of CAM is to find the convex hull.

Currently, the most popular method is Quickhull [39], a divide and conquer approach, the idea of which is similar to quicksort. Basically, Quickhull is trying to find a triangle and remove the internal points iteratively, until no points can be removed. The steps of Quickhull for 2D space are as follows:

1. Find the leftmost and rightmost points (the points with the minimum and maximum value of x coordinate).

2. With the line formed by the two points in Step 1, the plane is divided into two sub-planes, so the points are also divided into two subsets.

3. In one sub-plane, find the point which is the farthest to the line. The line and the point form a triangle.

4. The points in the triangle can be removed, since they are not on the convex hull obviously.

5. Since there are two more lines, repeat Step 3 and Step 4 on these lines iteratively.

6. Stop the iteration when there is no point can be removed, the left points form the convex hull. That is, the left points are the extreme points.

The time complexity of Quickhull is $O(N \log N)$, where N is the number of points. In the 2D or 3D space, Quickhull is a quite efficient algorithm. However, when the dimension increases, Quickhull may not be so "quick" anymore, since the time complexity of dimension is $O(\frac{N^{\frac{d}{2}}}{(\frac{d}{2})!})$. As

we know, the dimension of the data may be more than 100 (the number of samples). Though we can apply some dimension reduction methods first, the dimension could be still more than 10, so the time complexity may be extremely high. For example, after clustering, there may be about 50 clusters. If the dimension is 10, then the computation time for Quickhull to identify the convex hull on the 50 clusters is about eight seconds. If we increase the dimension to 11, then the computation time significantly increases to 72 seconds. If we further increase the dimension to 14, it will take more than nine hours. In the real applications, it is always that we do not know the exact source number but only some estimations. If we guess there may be more than 10 sources in the mixtures, we should at least keep 20 dimensions after dimension reduction, but the computation time of Quickhull would be extremely long. Thus, we need another method the complexity of which on dimension is much lower.

The core problem now is to find the convex hull more efficiently. To be more specific, we are trying to find the extreme points in the data. In fact, any point p_i in the data can be the convex combination of all points:

$$\sum_{i=1}^{N} \lambda_i p_i \ : \ \lambda_i \geq 0 \ for \ all \ i, and \ \sum_{i=1}^{N} \lambda_i = 1. \tag{4.5}$$

However, the definition of the extreme point is that the point can be constructed by just itself (the coefficients of the other points are all zero). That is, we are finding some points the linear combination of which are only themselves with some limitations on the coefficients. Thus, this problem can be formulated as a linear programming problem [81] as:

$$\min \lambda_j \ \ s.t. \ \sum_{i=1}^{N} \lambda_i p_i = p_j, \sum_{i=1}^{N} \lambda_i = 1, \lambda_i \geq 0 \ for \ all \ i. \tag{4.6}$$

We can test all p_j in the equation 4.6, and if λ_j is not zero, then the corresponding p_j is an extreme point. There are several advantages if we use linear programming to find the convex hull. First, though the time complexity of linear programming is just weak polynomial, is still better than the one of Quickhull on dimension. Second, though the complexity of number of points is worse than Quickhull, in CAM, the number of clusters is usually only around 100, and all clusters can be tested simultaneously. That is, for linear programming, we can even accelerate the computation with parallel computing. Third, if we want to "relax" the convex hull, for example, the points near

the boundary could be also identified as the extreme points, we can manipulate the constraints of the linear programming.

Table 4-1 shows some computation time comparison between Quickhull and linear programming. When the dimension number is 14 and the cluster number is 50, the computation time of Quickhull increases to about 9 hours, but only 1 second for linear programming. Thus, it is obvious that the computation time of linear programming is much faster than Quickhull. Even we increase the number of clusters to 1000 and dimension to 200, linear programming needs still only 4 minutes, which meets the demand in CAM sufficiently.

Table 4-1 Computation time of Quickhull and linear programming. The computation time of linear programming is always lower than Quickhull with combinations of different cluster number and dimension.

Computation time	Quickhull	Linear programming
Cluster number: 50 Dimension: 10	~ 8 seconds	< 1 second
Cluster number: 50 Dimension: 11	~ 1 minute	< 1 second
Cluster number: 50 Dimension: 12	~ 10 minutes	< 1 second
Cluster number: 50 Dimension: 13	~ 1 hour	< 1 second
Cluster number: 50 Dimension: 14	~ 9 hours	< 1 second
Cluster number: 1000 Dimension: 20	NA	~ 15 seconds
Cluster number: 1000 Dimension: 200	NA	~ 4 minutes

4.3.3 Optimal simplex identification

Supposed we know the number of sources *K*, then after finding out the clusters on the convex hull, the problem becomes which *K* clusters from the candidates are the vertices of the *K*-1 simplex. As mentioned in the section 4.2, it is a combinatorial optimization problem. That is, there is C_K^Q combinations to form a (*K*-1)-simplex. To decrease the computation time of this combinatorial optimization problem, instead of reconstruction error (equation 4.2), CAM adopts margin of error (equation 4.3) to rank all the combinations first, and then find the lowest 200 (a parameter can be set by the user) combinations. Then, in these 200 combinations, CAM computes their reconstruction error and selects the one with the lowest reconstruction error as the vertices of the (*K*-1)-simplex. However, though computing margin of error is much faster than reconstruction, when the number of *Q* or *K* increases, the computation time is still unaffordable. Moreover, in the real application, usually we do not have the information of source number or just have a range, so CAM need to solve the combinatorial problem independently to evaluate different values of *K*, which make the computation time a more severe problem. Also, since the margin of error is not equivalent to the reconstruction error, it is possible that CAM cannot find the best combination, though the top 200 combinations are tested.

To improve the computation speed problem, and directly use reconstruction error as the measurement, we propose the Sequential Forward Floating Search (SFFS) and Sequential Backward Floating Search (SBFS) [82] with reconstruction error to replace the margin of error. Though SFFS and SBFS are greedy methods, we directly use the reconstruction error and combine the both results, so sometimes the selected combination may have lower reconstruction error than the one picked by the margin of error. Moreover, as a greedy search, SFFS or SBFS tries different values of *K* when searching, which automatically meets our requirement for testing different values of *K* in the real application, so the computation time decreased significantly. The detailed steps of SFFS are as following:

1. Test all *Q* candidates, and select the one with lowest reconstruction error with **X** by NNLS. Then move the selected one from the candidates to the feature set.

2. Select one from the remaining candidates, and use this one and the feature set to compute the reconstruction error. After testing everyone in the remaining candidates, move the one with the lowest reconstruction error to the feature set. Record the reconstruction error for the current number of features in the feature set.

3. Remove one from the feature set temporarily, compute the reconstruction error for the current feature set, and then move the one back. After testing everyone in the feature set, if the lowest reconstruction error is lower than the recorded for the current number minus one of features in the feature set, remove the corresponding feature and replace the recorded reconstruction error.

4. Repeat Step 3 until the reconstruction error is not lower than the recorded one.

5. Repeat Step 2 – 4 until there is no candidate left.

Likewise, the steps of SBFS are as following:

1. At the beginning, all *Q* are in the feature set.

2. Remove one from the feature set temporarily, compute the reconstruction error for the current feature set, and then move the one back. After testing everyone in the feature set, move the one with the lowest reconstruction error from feature set to the candidate set. Record the reconstruction error for the current number of features in the feature set.

3. Select one from the remaining candidates, and use this one and the feature set to compute the reconstruction error. After testing everyone in the remaining candidates, if the lowest reconstruction error is lower than the recorded for the current number plus one of features in the feature set, move the one with the lowest reconstruction error to the feature set. Record the reconstruction error for the current number of features in the feature set.

4. Repeat Step 3 until the reconstruction error is not lower than the recorded one.

5. Repeat Step 2 – 4 until there is only one in the feature set.

Both SFFS and SBFS will test all the possibilities of *K* (1 to Q) in their steps, so we can compare the reconstruction error from SFFS and SBFS for the same source number, and pick the one with lower reconstruction error as the final result for the certain source number. The illustration of SBFS is shown in **Figure 4.8b.**

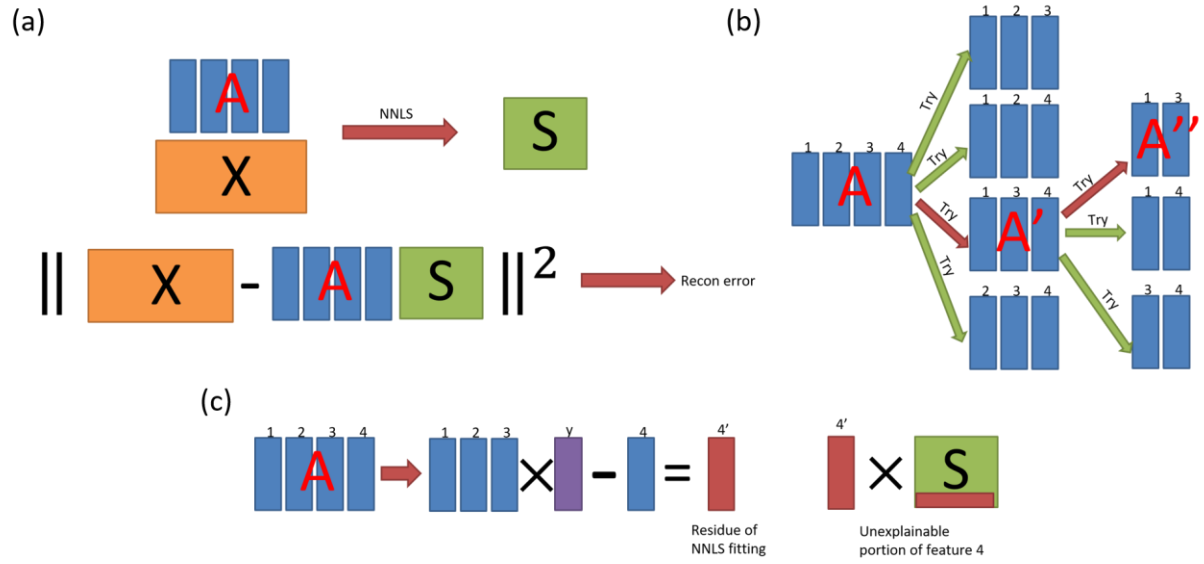

Figure 4.8 The illustration of SBFS with reconstruction error and unexplainable portion with four features in the feature set. (a) The reconstruction error the sum of the square of the residue by NNLS fitting. (b) In the Step 2 of SBFS, NNLS for fitting X is performed for excluding each feature in the feature set (four times in this figure) to compute the reconstruction error. (c) Though we still need to perform NNLS four times, but we just need to fit a matrix with only one column, not the whole X, so the computation time decreases significantly.

Moreover, when the total cluster number is larger (for example, > 100), the computation time of greedy search may still be high. Thus, to further improve the speed, besides the greedy search, we also try to approximate the reconstruction error with less computation. To be more specific, when deciding which one should be removed from the feature set, we pick the one with the lowest "unexplainable portion" to the others. That is, for each one in the feature set, we can use the others to fit it with NNLS, and the residue multiplying the corresponding row in the S matrix (computed by NNLS with the whole feature set) is the portion of **X** which can be explained by the feature, called unexplainable portion. Though calculating the unexplainable portions needs the same times

as calculating the reconstruction errors (the number in the feature set), the computation time of unexplainable portion is much lower, since we just need to fit the selected feature from the feature set, not the whole **X**. Though the accuracy may be lower with unexplainable portion, we can still use it for a quick check for the dataset. **Figure 4.8** is the illustration of the SBFS and the corresponding unexplainable portion for four features in the feature set.

Figure 4.9 is the plot of the reconstruction errors of three different methods, margin of error, SFFS and SBFS with reconstruction error, and SFFS and SBFS with unexplainable portion, for different number of *K*. The dataset is as same as the one demonstrated in the figure 4.2. The reconstruction errors for all the three methods are very similar. Though it is just one example, it is still an example that SFFS and SBFS could be an option to replace the margin of error, even with the unexplainable portion. Moreover, table 4.2 is the comparison of the computation time of the three methods. It is obvious that the computation time of SFFS and SBFS is significantly lower when cluster number more than 35. With unexplainable portion, we can even further decrease the computation time, though may sacrifice some accuracy. However, when the cluster number is more than 50, the computation time of the margin of error is unaffordable, and it could happen when the source number is larger than 10. Thus, though not be very accurate, it is still crucial that we have a faster method for identifying the optimal simplex.

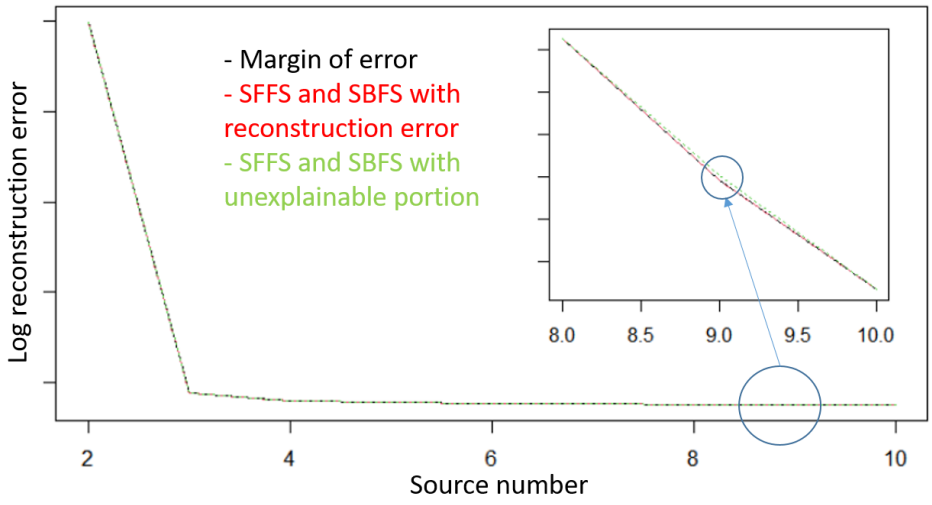

Figure 4.9 The reconstruction errors of the three different method. For source number from 2 to 10 of the data in **Figure 4.1**, the errors of the three methods are almost the same, except the ones when the source number is 9. The unexplainable portion is slightly higher than the other two.

Table 4-2 Computation time of the three different methods with different cluster number. Though the computation time is similar when the cluster number is 30, the differences are obvious when the cluster number increased to 35 or more. Though unexplainable portion may not be accurate, it is still good for a quick check, given the such low computation time.

Computation time	Margin of error	SFFS and SBFS with reconstruction error	SFFS and SBFS with unexplainable portion
Cluster number: 30	~ 30 minutes	~ 30 minutes	~ 10 minutes
Cluster number: 35	~ 3 hours	~ 40 minutes	~ 12 minutes
Cluster number: 40	~ 1 day	~ 1 hour	~ 15 minutes

4.3.4 Marker detection

After deconvolution, one way to examine the deconvolution results is to compare the marker genes with the prior knowledge (if there is any), for example, the ones tested by the biological experiments. However, currently, there is still no well-accepted algorithm for finding marker genes in a dataset, That is, detecting marker genes is still a challenging task even with purified gene expressions of tissues [83]. Similar to the idea of radius-fixed clustering, we propose a cosine-value based one-sample test method (COT) to detect the marker genes among two or more subtypes (sources) [84]. A marker gene, or to be more specific, a subtype-specific marker gene (SMG), is a gene which uniquely expressed in a certain subtype but not in the others. Mathematically, a SMG of subtype *k* can be expressed as [23, 85]:

$$s_k(n_{SMG,l}) = \begin{cases} s_k(n_{SMG,l}) \gg 0, & l = k, \\ 0, & l \neq k, \end{cases} \tag{4.7}$$

where $s_k(n_{SMG,l})$ is the expression of gene *n* in subtype *l*. Accordingly, the cross-subtype expression patterns of ideal SMG can be concisely represented by the Cartesian unit vectors $\hat{\boldsymbol{e}}_k$. Fundamental to the success of the COT method is the newly-proposed score $\cos(\boldsymbol{s}(n), \hat{\boldsymbol{e}}_k)$ that measures directly the distance between the cross-subtype expression patterns of gene *n* and the ideal SMG of subtype *k* in scatter space. The COT score is given by:

$$\mathrm{COT}(n) = \max_k \cos(\boldsymbol{s}(n), \hat{\boldsymbol{e}}_k) = \max_k \frac{s_k(n)}{\sqrt{\sum_{j=1}^{K}\left[s_j(n)\right]^2}}, \tag{4.8}$$

where $\boldsymbol{s}(n)$ is the *n*th column of the source matrix (the deconvolution result of **X**), that is, the expression of gene *n* across different subtypes, and $s_k(n)$ is the expression of gene *n* in subtype *k*. The known cross-subtype expression patterns of ideal SMG associated with the alternative hypothesis permit the use of a one-sample test to detect significant SMG, with $0 < \cos(\boldsymbol{s}(n), \hat{\boldsymbol{e}}_k) < 1$. Because $\boldsymbol{s}(n)$ is confined within the first quadrant whose central vector is the all-ones vector $\vec{\mathbf{1}}$, we have $\frac{\mathbf{1}}{\sqrt{\mathrm{K}}} < \mathrm{COT}(n) < 1$.

In relation to previous work, the effort to detect SMG can be traced back to One-Versus-Rest (OVR) test [86], One-Versus-One (OVO) test [85], and most recently One-Versus-Everyone (OVE) test [23]. We and others have recognized that the test statistics used by most existing methods do not exactly satisfy SMG definition (equation 4.7) and often require ad hoc OVE set intersection [83, 85, 86]. We then conducted a benchmark assessment of COT, using a one-sample test setting and the same gene expression data, in comparison with OVO test and the markers from prior knowledge.

In the GSE28490 dataset [87], we picked out five subtypes with prior knowledge 144 markers for the assessment [88]. The geometric proximity of the 144 SMG detected by a priori, OVO, and COT, to the vertices of scatter simplex and the heatmaps are given in **Figure 4.10**. It is clear that the markers detected by all three methods are close to the vertices of the simplex. However, for a priori and OVO, it is obvious that the detected markers are not exactly located at the vertices. On the other hand, the markers detected by COT are all confined around the vertices. Moreover, the heapmap of COT shows the markers detected by COT expressed significantly only in the corresponding subtype, but which is not true for a priori and OVO. As shown in the heatmaps, there are some reds (highly expressed) in more than one subtype in a priori and OVO t-test. Given the definition of marker genes, **Figure 4.10** is an example that sometimes prior knowledge may not be accurate enough. Thus, a good data-driven method, such as COT, is needed for the marker detection.

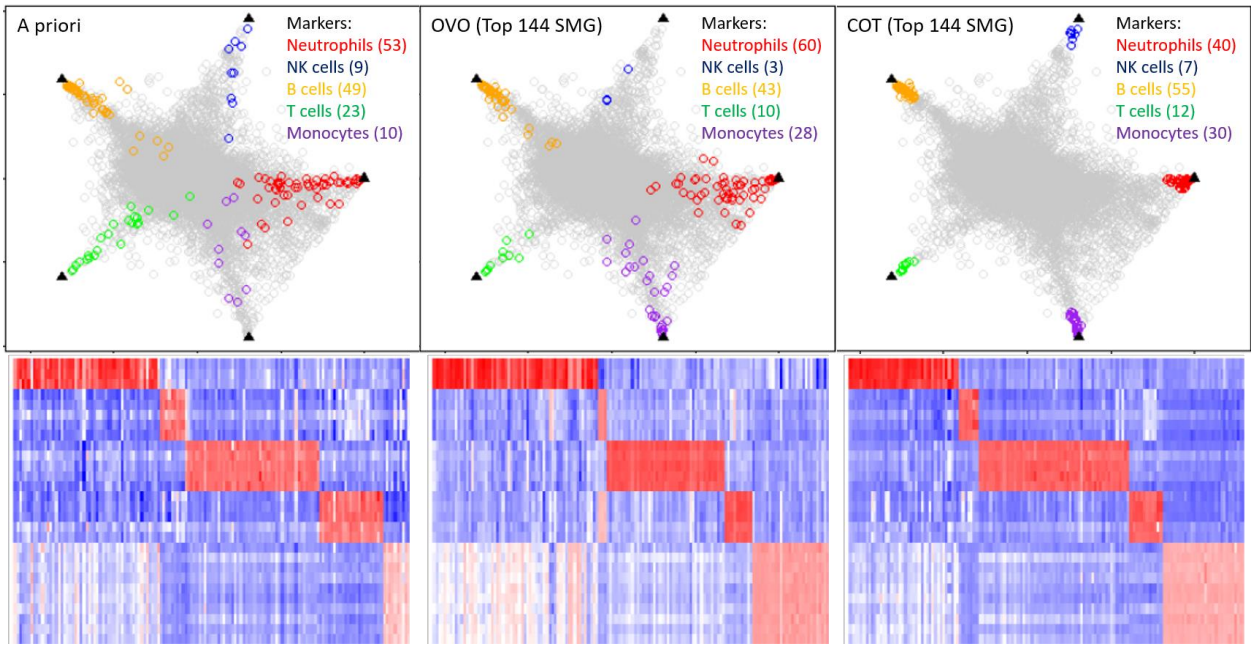

Figure 4.10 Simplex plots and heapmaps of GSE28490. The black triangles are the exact positions of the vertices of the simplex. In prior knowledge, there are 144 markers in these five subtypes (neutrophils, NK cells, B cells, T cells, and monocytes). The markers detected by a priori and OVO are close to the vertices, but not close enough. However, the markers detected COT are all confined around the vertices. Moreover, in the heatmap of a priori and OVO, it is clear that there some reds (high expression) across different subtypes for some marker genes. In the heapmap of COT, it is clear that except the corresponding subtype, the expressions of the marker genes are all almost blues or whites (low expression).

4.4 Discussion

Though we propose several ideas to improve the current design of CAM, these improvements are still not perfect. First, while we can control the size of the cluster in the radius-fixed clustering, the computation time is much slower than K-means clustering (though much faster than APC). Since radius-fixed clustering decides the clusters one by one, but K-means clustering decides all the clusters at the same time. Thus, if we can combine the advantages of these two methods, the integrated clustering method would be very suitable for initialization of CAM. For example, we can use K-means clustering as an initialization, and then apply the idea of the radius-fixed clustering to make sure the size of each cluster is no more than a certain threshold. However, in

this way, the uncertainty of the K-means clustering is still an issue, so we still need to think the other method to solve it.

In the convex hull identification step, we use linear programming to replace Quickhull for much faster speed and the same result. Since linear programming is much faster, it is even possible to use all the genes to identify the convex hull. However, as discussed in the subsection 4.3.2, the identified convex hull may be noisy. Thus, if we can adjust the constraints of linear programming, we may formulate a way to identify an approximated convex hull, which may be free to or not sensitive to the noise. That is, it is possible that we can drop the clustering step to detect the convex hull directly. Moreover, we even can consider to use robust linear programming [89] to find convex hull with some randomness.

For the optimal simplex identification problem, since CAM formulates it as a combinatorial optimization problem, it is an NP-hard problem [90], so there is no perfect solution. However, besides the greedy search algorithms, there is another type method which use lasso-related techniques [91]. That is, with a suitable penalty term (usually including L1-norm and L2-norm), by adjusting the parameter of the penalty term, we can obtain different number of candidates left, and the left candidates are corresponding to the vertices of the simplex. However, since we need to test a range of possible source number in the real applications, it would be time-consuming to try a lot of parameters to obtain all the needed numbers of the sources. That is, the computation time of this type of method is not guaranteed, but the accuracy may be better than ours, since it is not a greedy algorithm.

COT provides an accurate data-driven marker detection method where the score matches exactly the definition of SMG and permits the novel formulation of a one-sample test. Moreover, COT is efficient in that neither OVE set intersection nor intractable sample permutation is needed [83, 85]. While the case study here involves only mRNA, COT is applicable to any types of data with the same or similar marker definition, for example, protein or metabolomics data.

Besides the clustering step, the other steps in the improved CAM are suitable for parallel computing. Though radius clustering cannot be paralleled now, the computation time is much lower in the clustering step, comparing to convex hull identification and optimal simplex

identification. Thus, when dealing with a very large dataset, the improved CAM may be the most suitable unsupervised deconvolution method.

4.5 Summary

In this chapter, we introduce an unsupervised deconvolution method, CAM, and its improvements. In the clustering step, since the size of the clusters by K-means is hard to control, and the speed of APC is extremely slow, we propose radius-fixed clustering to control the size of the clusters easily with reasonable speed. Also, clustering and outlier removal are done by radius-fixed clustering at the same time.

In the convex hull identification step, we use linear programming to replace Quickhull, and the computation time decreased dramatically. Also, the constraints of linear programming provide us more flexibility when finding the convex hull.

To avoid the exhaustive search for the combinatorial optimization problem, we use SFFB and SBFS to identify the optimal simplex. Also, we directly use reconstruction error in the search, so the accuracy is comparable the original margin of error approach but much faster speed.

Inspired by the idea of radius-fixed clustering, we propose a new method for detecting the marker genes. This cosine-based method matches the definition of SMGs better than the other methods, as shown in the simplex plot and heatmap of the gene expressions.

Chapter 5

Biomedical Case Study Using improved CAM

5.1 Introduction

The functions of complex tissues are orchestrated by a productive interplay between many specialized tissue, cell, and task subtypes [92]. Characterizing the presence and dynamics of these components is important to understanding many physiological or pathophysiological processes. For example, shifts in the relative composition of neuron or glia cell subtypes is central to the developmental processes of the human brain [93, 94]. Likewise, understanding the genuine changes of subtype-specific molecular expressions is of direct etiological interest for many diseases [75]. Biologists have amassed a large body of information about complex tissues and have some powerful insights into how they remodel under different conditions [95]. However, most of our knowledge relies on mixed readouts with many unknown confounders. Experimental solutions to mitigate tissue heterogeneity are to isolate individual cells or to microdissect tissue subtypes before molecular profiling. While promising, physical separation is clearly not the most reliable and cost-effective method and is inapplicable to previously-assayed samples [26]. Would it not be better if we could frame a tissue ecosystem in precise mathematical models and use computational deconvolution to analyse, or re-analyse, the wealth of publicly available multi-omics bulk data?

In this chapter, we demonstrate some biomedical applications of the improved CAM. Our deconvolution pipeline is built on the strong parallelism between linear latent variable models and the theory of convex sets. Tissue samples to be analyzed by CAM contain more than two and varying proportions of molecularly distinctive subtypes present across various scales (tissue, cell, or process). Molecular expression in a specific subtype is modeled as being linearly proportional to the abundance of that subtype. According to the theory of convex sets [96] and the improved steps in CAM, every molecular feature within the scatter simplex can be uniquely determined by the linear combination of the vertices. Thus, the number of the vertices corresponds to the number of molecularly distinctive subtypes present in the bulk samples and the molecular features residing at the vertices are the molecular markers defining such subtypes [23].

Our CAM based deconvolution works by detecting the vertices of the scatter simplex geometrically, *i.e.*, determining the multifaceted simplex that most tightly encloses the globally measured expression mixtures. Subsequently, we identify the molecular markers residing at the vertices, and estimate the proportions and specific expression profiles of constituent subtypes [23]. The number of subpopulations present is determined by the minimum description length (MDL) criterion. Ideally, a molecularly distinctive subtype would contain molecular signatures (molecular markers) that are exclusively expressed in the cognate cell or tissue subtype of interest while in no others. Importantly, our deconvolution pipeline requires no a priori information on the number, signatures, or compositions of the subtypes present in heterogeneous samples, and does not require the presence of pure subtype samples [74, 77]. This advantage is significant in that CAM can achieve all of its goals using only a small number of heterogeneous samples, and provides a powerful means to distinguish among phenotypically similar subtypes.

While there may be many ways to subdivide a complex tissue ecosystem, there is clearly a need to dissect the staggering complexity of many multiscale molecular landscapes. We have shown the performance and biomedical utility of CAM based deconvolution using gene expression [23], methylation [37], proteomics [75, 97], and imaging data [98]. These applications have led to novel findings and hypotheses. However, the source numbers of the above applications are all only around five. With the improved CAM, the computation time is affordable when the source number is around 10 or even more, which means that we can use CAM in more applications.

A molecular latent variable model of complex tissues, particularly in the presence of disease lesions, is not yet available, but we can provide a roadmap of how a machine learning approach might uncover, in mathematical forms, the molecular events controlling tissue remodeling in many biomedical contexts. The value of this deep characterization is illustrated by the fully unsupervised deconvolution results obtained from spatial-temporal molecular expression data of various complex tissues, and will be measured ultimately by the emerged new insights or hypotheses.

5.2 Implementation in biological data

As explained in the last chapter, the major steps in CAM including clustering, convex hull identification, optimal simplex identification, and A and S matrix estimation. However, in the real

application, the number of samples are usually much larger than the number of sources. Thus, CAM will apply principle component analysis (PCA) for dimension reduction first. The general pipeline of CAM is shown in **Figure 5.1**.

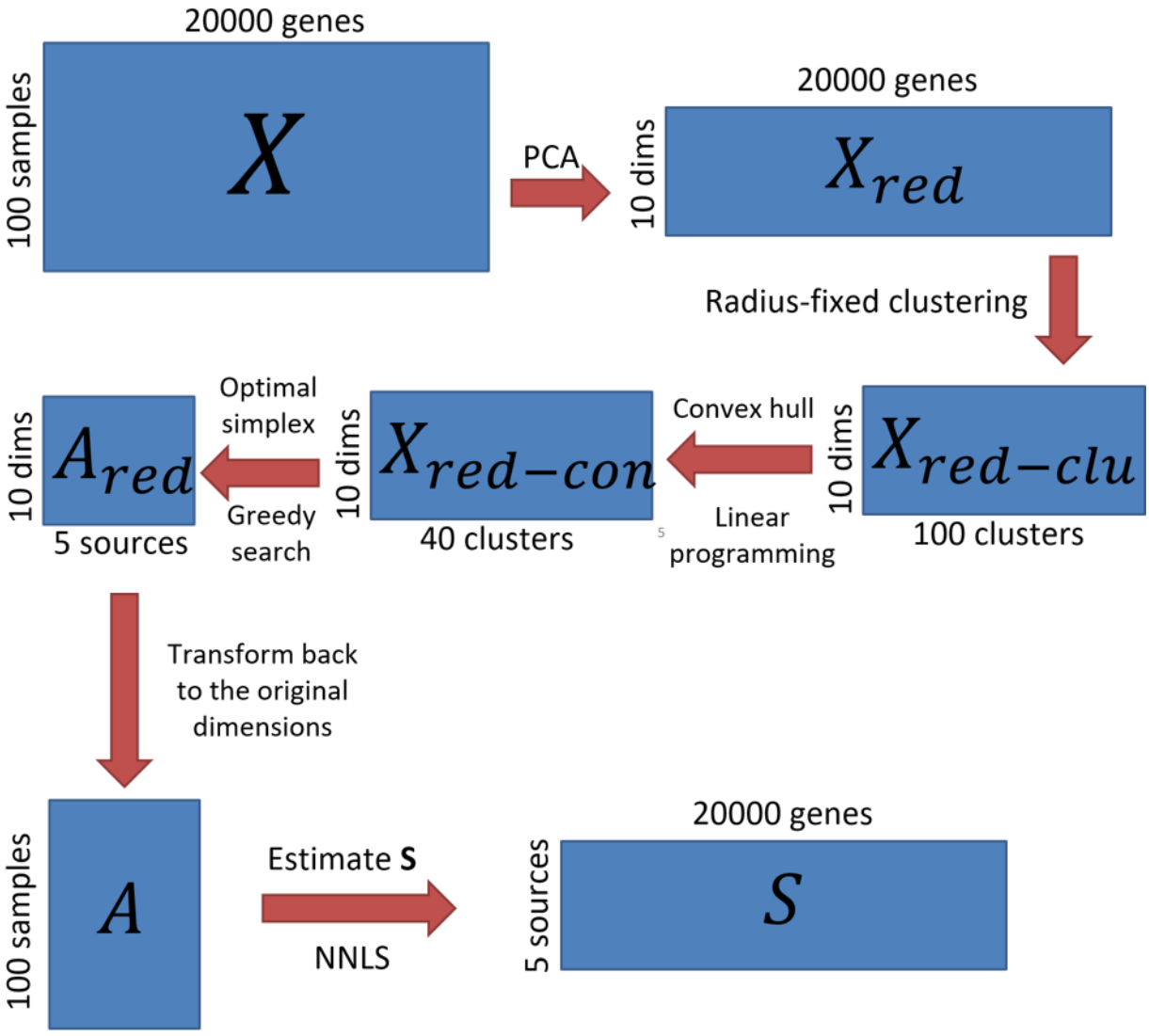

Figure 5.1 General pipeline of CAM. The number of samples, genes, sources, dimension after PCA, clusters, are examples only in this figure for illustration.

5.2.1 Source number pre-estimation

As mentioned, in the general CAM pipeline, we will apply PCA for dimension reduction. However, the first question is how to decide the number of the principle components we should keep after

PCA. This is a hard question and we can just "guess" with some prior knowledge if there are some. However, recently, some deep-learning based methods are getting noticed [99]. For example, Yang et al. use the eigenvalues of the received signal covariance matrix to estimate the source number with fully-connected neural network [100]. Though their application is on the signal received by the antennas, we tried their idea on the gene expression data due to the similar properties. That is, intrinsically, both are blind source separation problems. While the estimation may not be as accurate as we wished, it is a good pre-estimation for us to decide the number of the principle components we should keep after PCA. The structure of the neural network we use are shown in **Figure 5.2**.

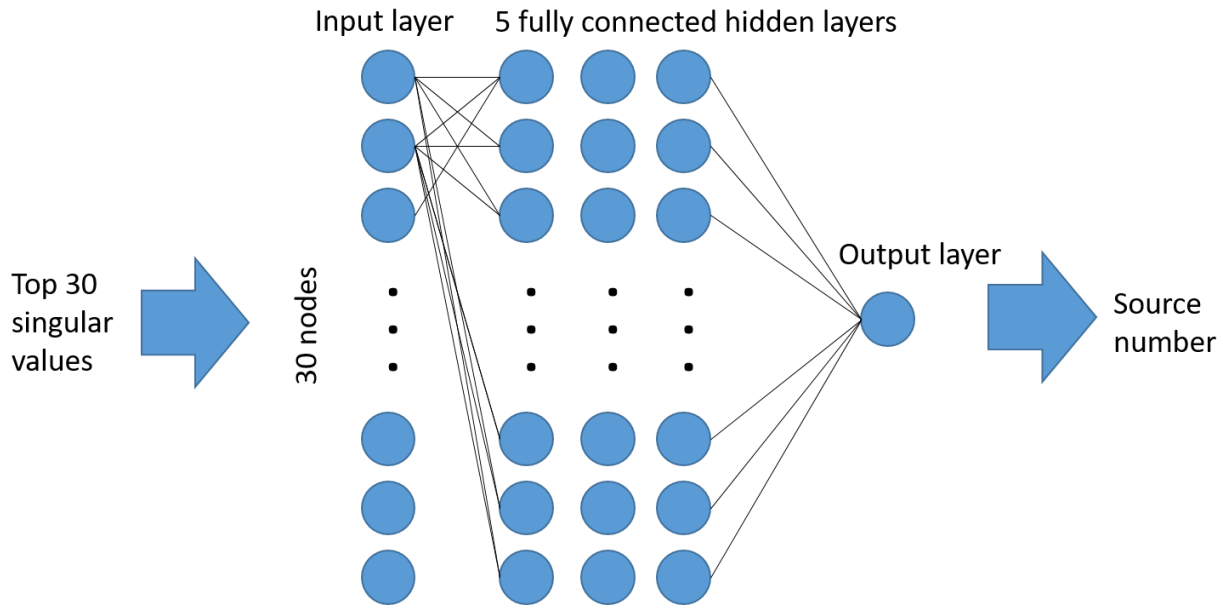

Figure 5.2 Structure of the source number pre-estimation neural network. In the real application, the source number may not exceed 30 in general, so here we choose the top 30 singular values as the input. The number of the fully connected hidden layers is choosing as five.

Since the size of each dataset may be very different, it is impossible to set the input size to fit all possibilities. Thus, in our implementation, we set the input node number as 30 for the top 30 singular values of the data matrix. In the mixtures of gene expressions, the number of sources is usually less than 30, so we pick 30 as the input size and focus on the top 30 singular values only. In this way, we can pre-estimate the source number of the dataset with any size. Moreover, since the intensity of different dataset may be very different, we will normalize the singular values by

divided by the largest singular value in the data. Therefore, the input data will be at the similar levels.

While we can deal with any size of the data now, another problem is we need data with ground truth as the training dataset. However, there are not too many such data available, not to mention they are with totally different sizes and source numbers. Thus, also follow the idea of Yang [100], we produce the simulation data as the training dataset. To be more specific, the range of the number of samples is from 10 to 1,000, gene is from 100 to 10,000, and source number is from 2 to 30. Therefore, the sizes of the simulated data would be a lot of combinations. In this way, we can simulate the situation in the real application that the sizes of different data are very different. Also, we add different levels of noise to the simulation data to further simulate the real data. Note that all the values in the simulation data should keep all positive (the fourth assumption in subsection 4.2.1).

After training with the simulation data, the model is ready for source number pre-estimation. However, currently the pre-estimation is only for deciding the dimension after PCA. For example, if the pre-estimation is five, then the dimension after PCA should be around 10 (about 2 times of the pre-estimation). The final estimation is still done by MDL, as mentioned in the sub-section 4.2.1.

5.2.2 Model selection

As mentioned in sub-section 4.2.1, after trying all possible values for the source number, CAM applies MDL to find the optimal one, as shown in equation 4.4. However, in some experiments, we notice that it seems that sometimes the source number K is underestimated. Thus, we re-examined equation 4.4 for possible reasons.

Assuming that there are some 'valid/wanted' sources with sufficient number of marker genes and satisfied vertex criteria, but which are highly correlated with some others and have small proportions, will MDL miss these sources due to high slope of penalty line? In equation 4.4, the statistical model behind MDL is actually a simple isotropic Gaussian model which follows a Gaussian distribution, and we account for model complexity by dealing with **A** and **S** separately (considering they are model parameters). If over-parameterization indeed occurs, it may be mainly

due to the high dimensionality of the vectors (sample-space), i.e., the number of features (N or M). A practical place to look into is the 2nd/middle term in equation 4.4, i.e., complexity related **A**. While we theoretically 'use' all features to identify marker genes and search for optimal simplex (for a particular K), the estimate of **A** relies only on marker genes. Thus, we use $log(N_{markers})$ ($N_{markers}$, the total number of genes in the selected clusters for the simplex) instead of $log(N)$ in equation 4.4:

$$MDL(K) = \frac{NM}{2}\log\left(\frac{1}{NM}\sum_{j=1}^{N}\|\boldsymbol{x}(j) - \mathbf{A}'\mathbf{s}'(j)\|_2^2\right) + \frac{(K-1)M}{2}log(N_{markers}) + \frac{KN}{2}log(M)\,. \quad (5.1)$$

However, since N is usually larger than M (the number of genes is larger than the number of samples), the third term is larger than the second term in equation 5.1. Thus, the effect of the changing is not obvious. For comparison, for both version of MDL (equation 4.4 and equation 5.1) are all included in the improved CAM package.

5.2.3 Simplex visualization

After deconvolution by CAM, we use MDL criterion to decide the number of sources, so that the optimal (K-1)-simplex are obtained. Visualization of the identified simplex may help us to further understand the data, and we can see the deconvolution result is consistent to the visualization or not. However, for the classical 2D visualization methods, such as PCA, or the more advanced ones, such as t-SNE [101] and Umap [102], none of them can guarantee the vertices of the simplex will be kept in their visualization. Though after PCA, the simplex would become a convex hull (the convexity is kept), some of the vertices may hide in the convex hull, not on the convex hull anymore. PCA only keeps the dimension with the top two largest variances (2D visualization), which will not consider the position of the vertices. Umap and t-SNE consider the local structure rather than the global structure. That is, these more advanced methods focus on the relation of the neighboring points, but the simplex is a global structure.

In the CAM package, the 2D visualization of simplex is done by projecting the vertices onto a circle with equal arc length. For example, for a 3-simplex, the projection is always as square, as

shown in the **Figure 5.3**. After the projection of the vertices is decided, the projection matrix **P** can be obtained by:

$$\mathbf{P} = \begin{bmatrix} 1 & \cos\left(\frac{2\pi}{K}\right) & \cos\left(\frac{2\pi}{K}\times 2\right) & \dots & \cos\left(\frac{2\pi}{K}\times (K-1)\right) \\ 0 & \sin\left(\frac{2\pi}{K}\right) & \sin\left(\frac{2\pi}{K}\times 2\right) & \dots & \sin\left(\frac{2\pi}{K}\times (K-1)\right) \end{bmatrix} \times \mathbf{A}^{-1}, \qquad (5.2)$$

where cosines and sines are the projections of the vertices, and **A** is the mixing matrix from the deconvolution result by CAM.

However, this projection will only keep the information of number of the vertices. That is, information of the distance among the vertices is totally neglected in this projection. Moreover, the order of the position of the vertices on the circle is decided by the column order of **A** matrix. That is, if we reorder the column in **A**, the vertices order will change after projection, though the shape of the simplex is not changed.

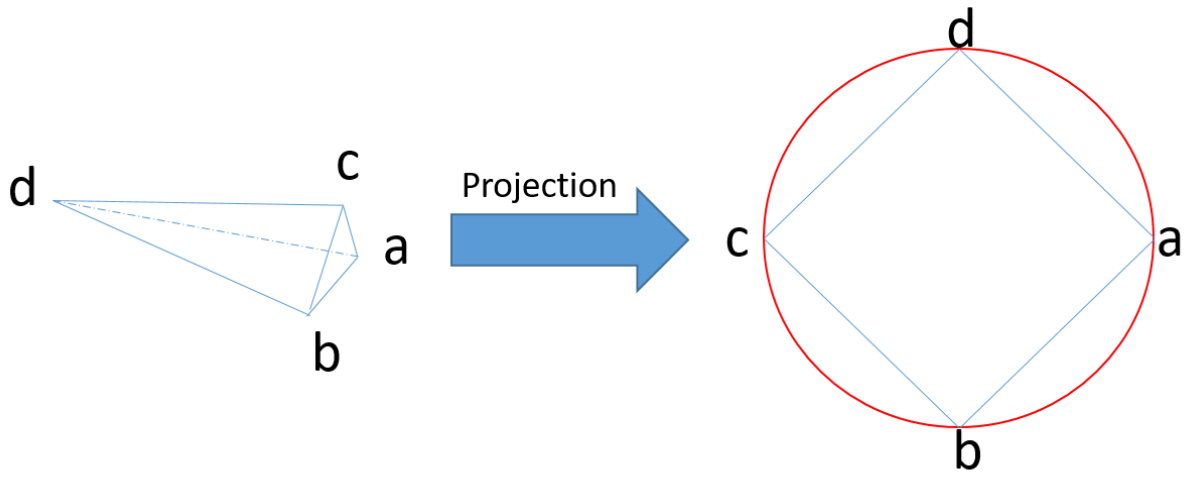

Figure 5.3 3-simplex projection example. No matter what shape is of a 3-simplex, the projection of it by the function in the CAM package is always a square (regular triangle for any 2-simplex), and the order of the vertices on the circle is not related to the distance among the vertices. Thus, some information lost in the projection.

Thus, based on such equal-arc-length projection, we propose another to project the simplex with the information of the distance among the vertices. The basic idea is that first we decide the order of the vertices on the circle one by one according to the distance among the vertices, and then decide the arc length by the distance between the directly neighboring point pairs. The steps of the newly proposed projection are as following and also shown in **Figure 5.4**:

1. For the *K* vertices of the (*K*-1)-simplex, find the three points (for example, b, d, and f) which form the triangle with the maximum area among all possible C_3^K combinations exhaustively.

2. After finding out the three points, project them onto a circle temporarily (b', d', and f'), note that the arc length is not important in this step.

3. For the remaining *K*-3 points, sort them by the sum of distance to the three points (b, d, and f) detected in the first step decreasingly in a list.

4. According to the order after sorting in the third step, find the first point (for example, a), and compare the distance between this point (a) to the points not in the list (b, d, and f). Find out the two points which are the closest to the point (d and f), and then the point is projected onto the arc between the projection of the two points (a' is between d' and f'). Remove the point from the list.

5. Repeat the fourth step until there is no points left in the list.

Assign the arc length by the distance of the two end points of each arc before projection (the original space).

Thus, in our newly proposed simplex projection method, the relative positions of the vertices are reserved largely, and the projection matrix of the points inside the simplex can be obtained by equation 5.2 with the projections of the vertices too.

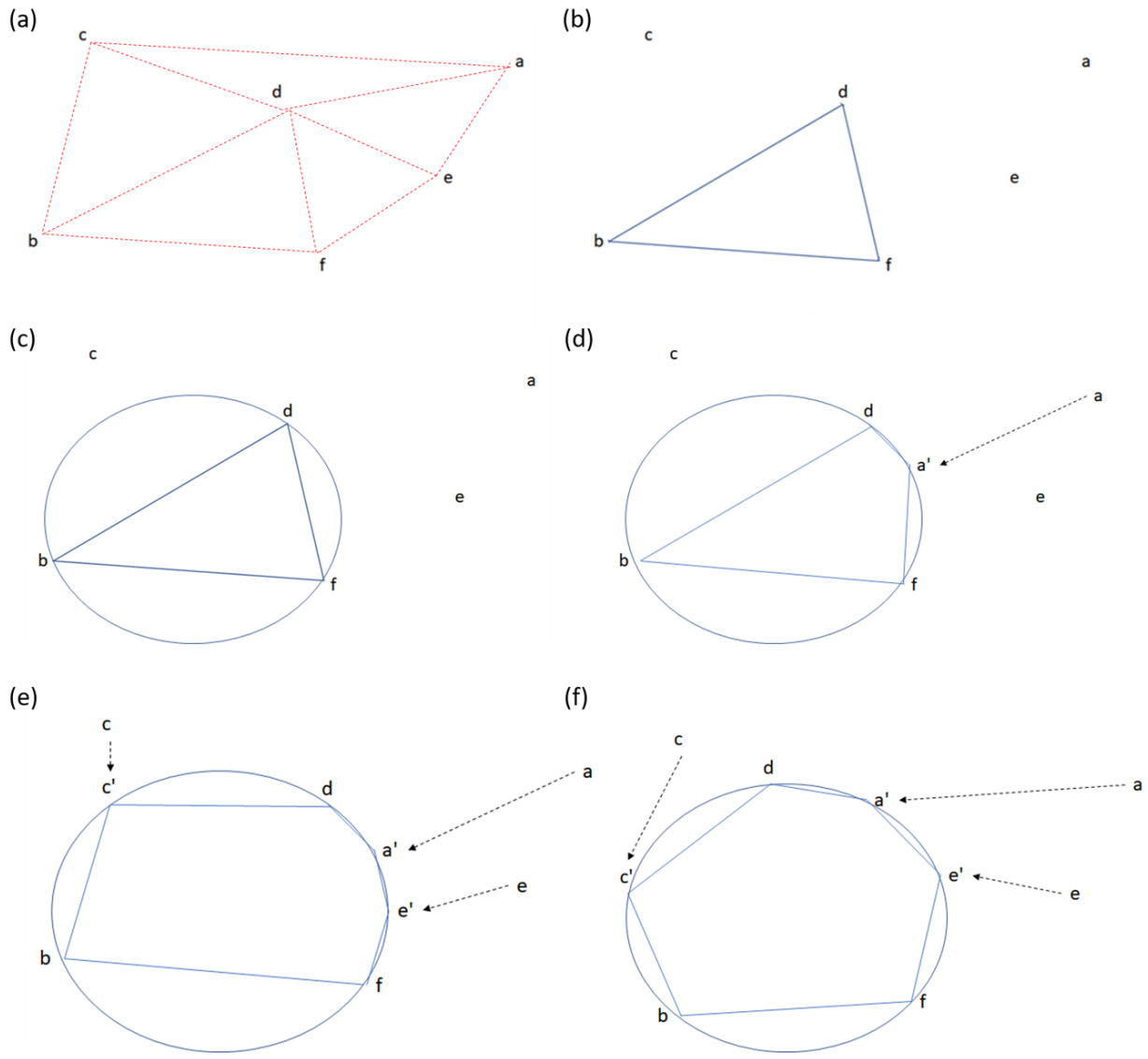

Figure 5.4 Newly proposed projection method. The example in the figure is a 5-simplex (6 vertices). (a) The original simplex in a high dimensional space. (b) Among the all vertices, find the three which form the triangle with the maximum area. The arc lengths now are just temporary ones. (c) We can view the triangle in a circle. (d) For the other points, project them one by one by their sum of the distances to the three points on the circle (furthest one first). In this example, point a is the farthest one, and which is close to d and f in the original space, so a' is between d' and f'. (e) Project the rest points (c and e) onto the circle (follow the same rules in (d)). (f) Assign the arc length by the length of $\overline{cb}, \overline{bf}, \overline{fe}, \overline{ea}, \overline{ad}, \overline{dc}$ in the original space.

5.2.4 Imputation and deconvolution

The phenomenon of missing values is very common in the omics data [103, 104]. Thus, before deconvolution, it is highly possible that we need to deal with the missing values first. If the portion of genes (or samples) with missing values is not high, for example, less than 5%, then we can simply remove the genes (or samples) so that there are no missing values in the remaining ones. However, when the portion of genes (or samples) with missing values is too high, then it is not appropriate to remove those genes (or samples), or the resolution of the dataset may not be enough. Thus, we need to solve the missing value problem in the other way.

In the algorithms of CAM, it is possible to edit them to be miss-value tolerable. For example, in the dimension reduction step, we use a clustering method to replace PCA. That is, set the number of dimensions after PCA as the cluster number, and then use the cluster centers of samples as the dimension reduction results. When clustering, we can choose a distance / similarity measure which can accept the missing values. Thus, it is also applicable to the following clustering step (gene / feature clustering).

Therefore, with appropriate editions, CAM can accept the missing values in the input, which means that CAM can be a missing value imputation method potentially. That is, after deconvolution by CAM, the estimated **A** and **S** matrix can multiply together as the fitting to the original data matrix **X**. Then, the missing values in **X** can be imputed by the values at the corresponding positions in the fitting matrix.

However, we found that the best result is based on the initialization by the other imputation method [105]. That is, before CAM, we can impute the missing values in **X** by the other imputation method, for example, NIPALS [106], as the initialization. Then apply CAM on the data with such initialization. Moreover, we can still replace the initial imputation by the corresponding ones in the fitting matrix, and it is possible that the ones imputed by CAM are more accurate than the ones by the initial imputation method [105]. The reason may be that the assumptions of CAM follows the biological properties, which may be not considered by the other imputation methods, so the result of CAM may be better.

5.3 Results

To validate the superiority of the improved CAM, we compare the results from the original CAM and the improved version on the datasets with ground truth. Moreover, in this section, to show the application of the improved CAM on the biological data, we will demonstrate the deconvolution results of the improved CAM, and the results are consistent with the prior knowledge, which is also a kind of validation that the deconvolution by the improved CAM is reasonable, and the results can be utilized for further analysis.

5.3.1 Validation on mixtures of mouse tissues

The first dataset for the comparison is RNA mixtures from the rat tissues (GSE19830) [30], which is from one animal. The mixtures contain rat brain, liver and lung biospecimens (three tissue types) with different proportions. For each proportion combination, there are three technical replicates, totally 33 samples, and 31,099 RNAs. This dataset is the one demonstrated in the sub-section 4.3.1.

First, we send the dataset to the source number pre-estimation module (deep-learning model), and the estimation is exactly three, which proves that our pre-estimation for the source number is very accurate. Then, supposing we do not know the actual number of the source, based on the pre-estimation, we applied PCA to reduce the number of dimension (samples) to 10 (a parameter in the CAM package). Since there are only three sources, the corresponding 2-simplex is a triangle, which can be shown in a 2D space as shown in **Figure 4.2**, using the top 2 components after PCA dimension reduction.

After the data processed by the original and improved CAM package, first we can compare their MDL curves to see whether the source number they estimated is correct or not. As shown in **Figure 5.5a** and **b**, both of the lowest points are at three. That is, both of them give the correct estimation, which is also an evidence that our MDL criterion works pretty well.

Next, we compared the deconvolution results directly. Since the mixtures in the dataset are mixed artificially, the mixing proportions of each sample are known. Thus, we can compute the correlation coefficients and cosine similarities for each source (the columns in **A** matrix) to evaluate the deconvolution results, as shown in **Table 5-1** and **Table 5-2**. The correlation

coefficients and cosine similarities of both are quite high. However, the computation time of the improved version is only 7 minutes 18 seconds, but the original version needs 26 minutes and 12 seconds (**Table 5-3**). That is, the improved version needs only one third time to obtain similar and accurate result.

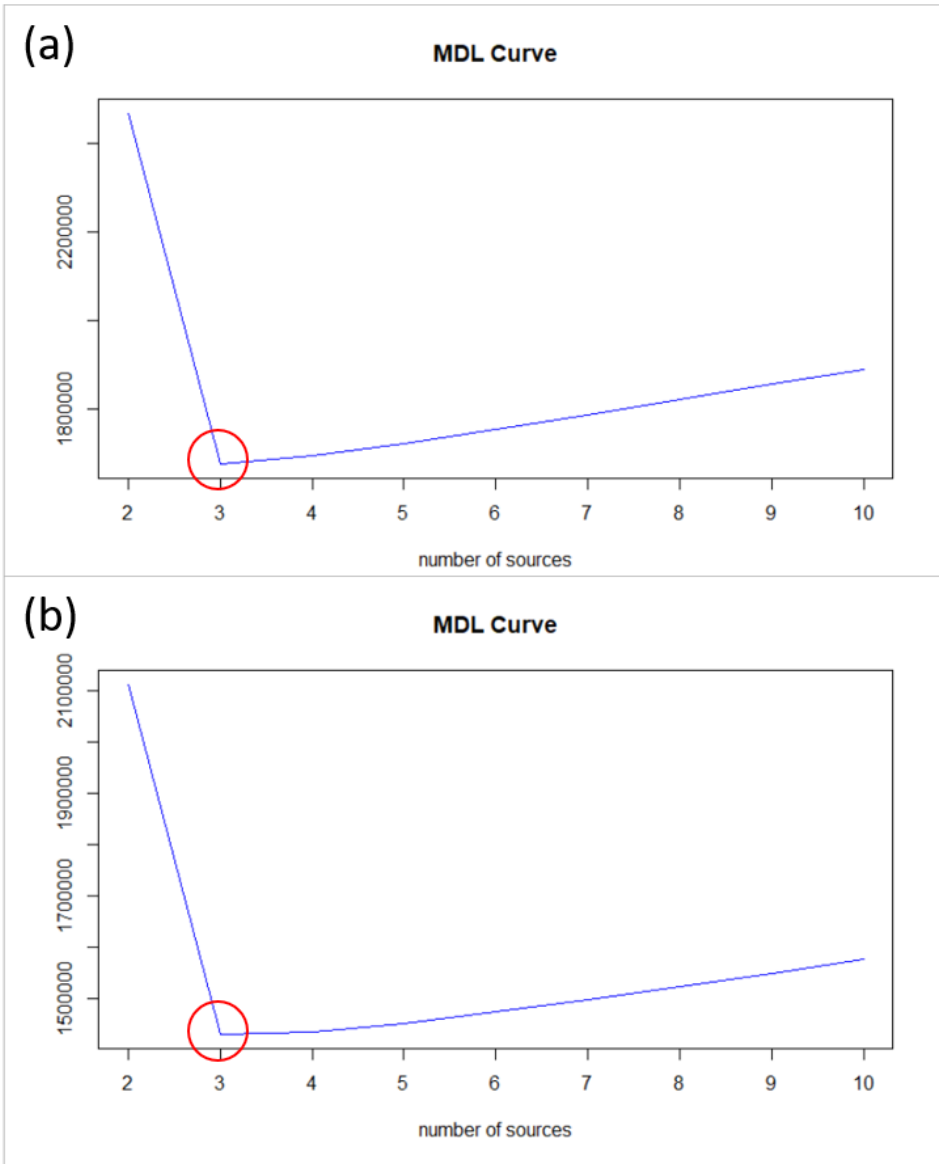

Figure 5.5 MDL curves of two version CAM for GSE19830. The MDL curve in the upper figure is from the original version, and the lower one is from the improved version. (a) The lowest point of the original is at three, which is as same as the ground truth. (b) Again, the lowest point of the improved version is at three, which is as same as the ground truth. That is, both versions estimate the source number accurately.

Table 5-1 Correlation coefficients between estimated A matrix and ground truth A matrix of original CAM and improved CAM. The correlation coefficients of both versions are quite similar and high.

Correlation coefficients	**Source 1**	**Source 2**	**Source 3**	**Average**
Original CAM	0.9797	0.9699	0.9845	0.9780
Improved CAM	0.9788	0.9690	0.9842	0.9773

Table 5-2 Cosine similarities between estimated A matrix and ground truth A matrix of original CAM and improved CAM. The cosine similarities of both versions are quite similar and high.

Correlation coefficients	**Source 1**	**Source 2**	**Source 3**	**Average**
Original CAM	0.9971	0.9908	0.9859	0.9913
Improved CAM	0.9972	0.9904	0.9885	0.9921

Table 5-3 General performance of original CAM and improved CAM. Though the errors of A matrix are similar, the computation time of the improved CAM is much shorter. Note that the range of values in A is from zero to one.

Performance	**Computation time**	**Root-mean-square error of A matrix**
Original CAM	26 minutes and 12 seconds	1.337%
Improved CAM	7 minutes 18 seconds	1.330%

Moreover, we can compare the root mean square error of A matrix. Again, as shown in table 5.3, the root-mean-square errors of both versions are quite similar. Thus, we can see that in this simple

dataset, the original and improved version CAM have very similar performance, but the computation speed of the improved version is much faster.

Next, we apply our newly proposed SMG detection method, COT, on the estimated S matrix to identify the SMGs, as shown in **Figure 5.6**. One can see that the detected SMGs (red, green, and blue circles) are all located in the vertices of the simplex, which meets the requirements of SMG, so the performance of COT is quite good on this dataset. Thus, COT is very suitable for detecting SMGs after deconvolution.

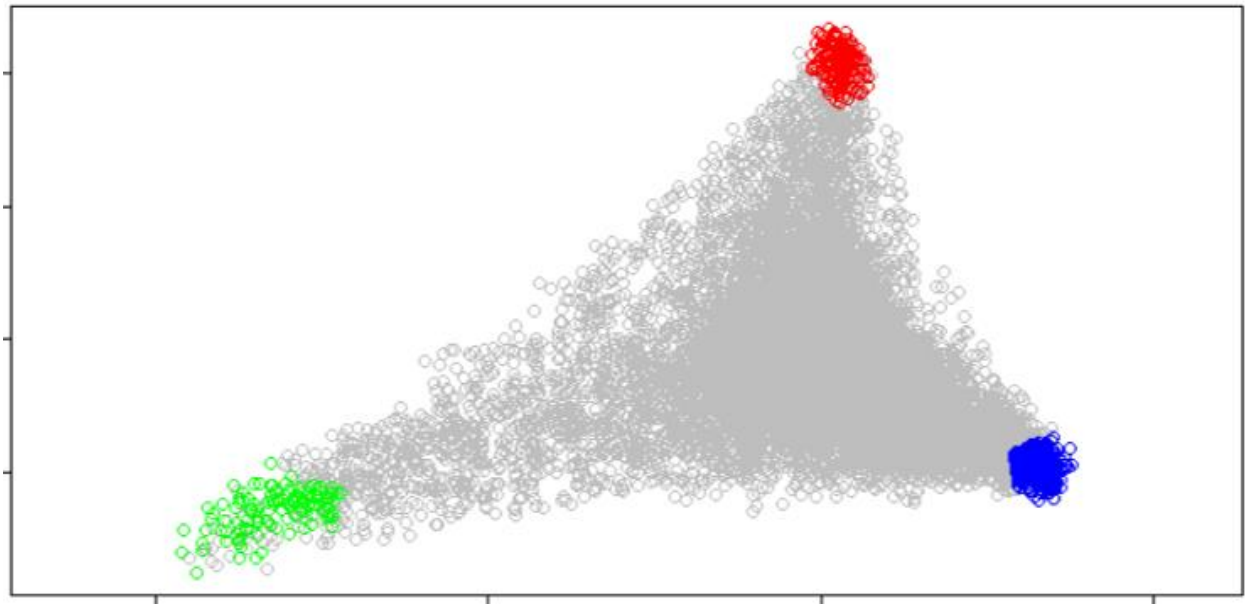

Figure 5.6 SMG detection by COT. The 2-simplex (triangle) is the projection of the dataset. Since the dimension of the 2-simplex is exactly two, there is no high-dimensional projection problem after projection onto a 2D space. The red, green, blue circles are the SMGs detected by COT (each color corresponding to one source), and one can see that they are all located in the vertices of the simplex, which meets the requirements of SMG.

5.3.2 Validation on mixtures of immune and cancer cell lines

Next, we compare the original CAM and the improved CAM on a biologically mixed gene expression dataset (GSE64385) [88]. In the mixtures, there are five immune cell types and one cancer cell line (six sources). In the dataset, there are 12 samples with different proportions of cell types and 54675 RNAs. At the beginning, we can visualize the dataset by PCA as a 2D plot, as

shown in **Figure 5.7**, and the "simplex" is similar to a triangle. However, since there are six sources, there should be six vertices on the simplex. Thus, we can see the limitation of PCA. That is, it is not easy to visualize a high dimensional simplex.

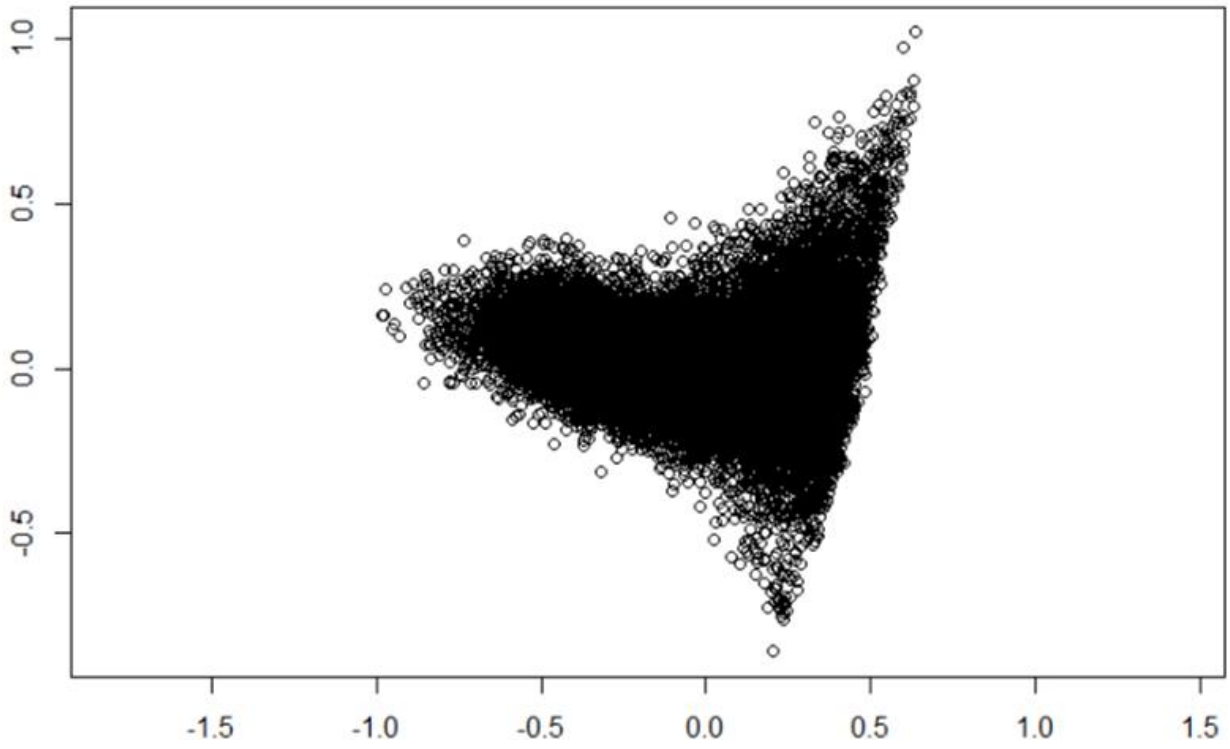

Figure 5.7 2D visualization by PCA. After the dimension reduction by PCA, the "simplex" here is similar to a triangle, however, which should contain six vertices (six sources). Thus, this example shows that PCA is not a good method for visualizing a high dimensional simplex.

Likewise, before we apply CAM, we use our deep-learning model for the source number pre-estimation, and the estimation is exactly six, again, which proves our pre-estimation for the source number is very accurate. Also, supposing we do not know the actual number of the source, based on the pre-estimation, we applied PCA to reduce the number of dimension (samples) to 10.

After the data processed by the original and improved CAM package, first we can compare their MDL curves to see whether the source number they estimated is correct or not. As shown in **Figure 5.8a** and **b**, both of the lowest points are at six. That is, both of them give the correct estimation of the source number, which again shows that that our MDL criterion is accurate.

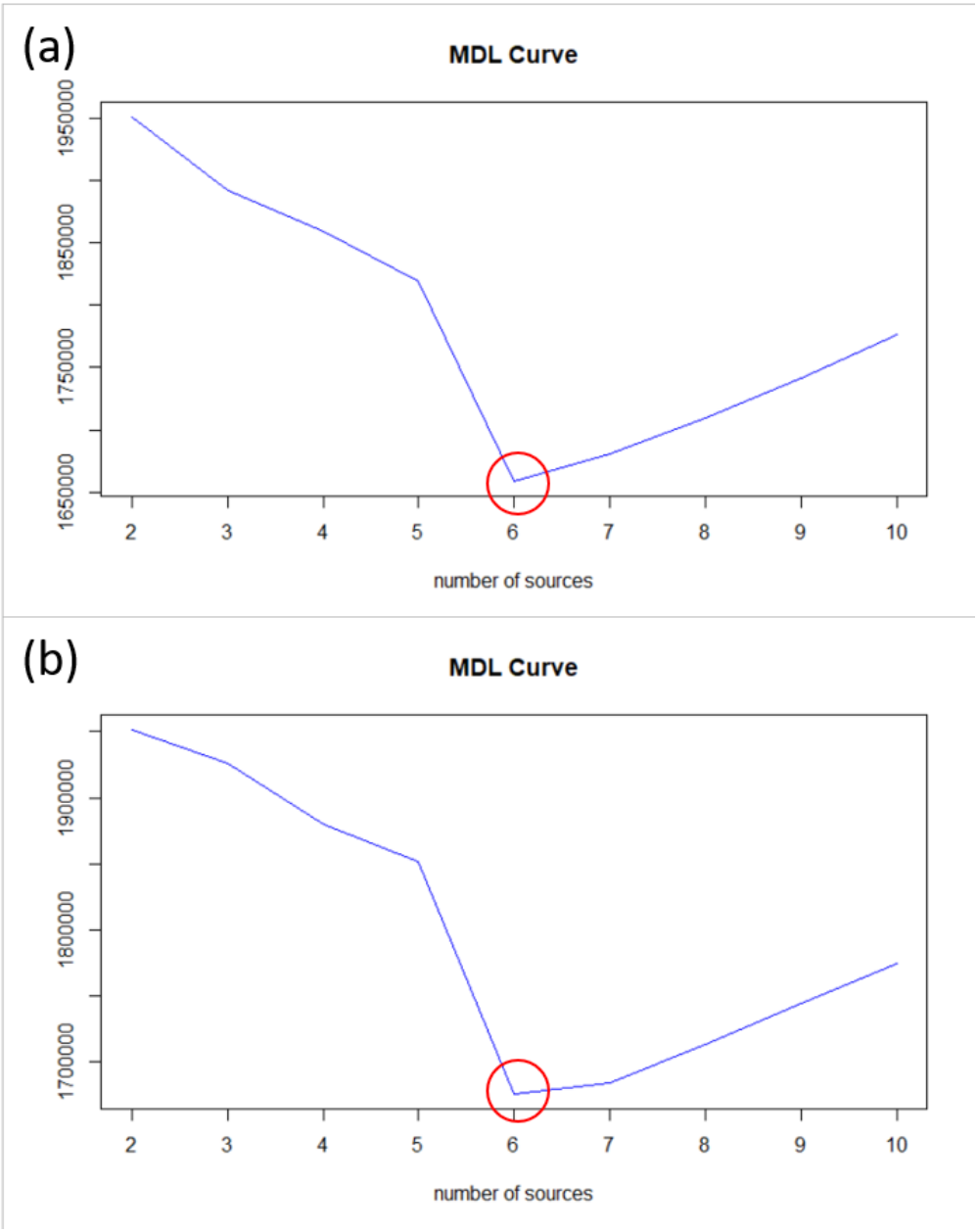

Figure 5.8 MDL curves of two version CAM for GSE64385. The MDL curve in the upper figure is from the original version, and the lower one is from the improved version. (a) The lowest point of the original is at six, which is as same as the ground truth. (b) Again, the lowest point of the improved version is at six, which is as same as the ground truth. That is, both versions estimate the source number accurately.

Next, we compared the deconvolution results directly. Since the mixtures in the dataset are mixed artificially, the mixing proportions of each sample are known. Thus, we can compute the

correlation coefficients and cosine similarities for each source (the columns in A matrix) to evaluate the deconvolution results, as shown in **Table 5-4** and **Table 5-5**. The correlation coefficients and cosine similarities of both are quite high. However, the computation time of the improved version is only 14 minutes 6 seconds, but the original version needs 2 hours 47 minutes 57 seconds (**Table 5-6**). That is, the improved version needs only one tenth time to obtain similar and accurate result, which is much faster than before.

Table 5-4 Correlation coefficients between ground truth and estimated A matrix of original and improved CAM. The correlation coefficients of both versions are quite similar and high.

Correlation coefficients	**Source 1**	**Source 2**	**Source 3**	**Source 4**	**Source 5**	**Source 6**	**Average**
Original CAM	0.9938	0.9815	0.9929	0.9933	0.9917	0.9802	0.9889
Improved CAM	0.9948	0.9608	0.9921	0.9950	0.9895	0.9884	0.9868

Table 5-5 Cosine similarities between ground truth and estimated A matrix of original and improved CAM. The cosine similarities of both versions are quite similar and high.

Correlation coefficients	**Source 1**	**Source 2**	**Source 3**	**Source 4**	**Source 5**	**Source 6**	**Average**
Original CAM	0.9824	0.9761	0.9962	0.9854	0.9772	0.9776	0.9825
Improved CAM	0.9816	0.9719	0.9949	0.9929	0.9745	0.9939	0.9850

Table 5-6 General performance of original CAM and improved CAM. The computation time of the improved CAM is much shorter, which is only about one tenth to the one of the original version. Moreover, the root-mean-square error is about 80% to the one of the original version

Performance	**Computation time**	**Root-mean-square error of A matrix**
Original CAM	2 hours 47 minutes 57 seconds	9.15%
Improved CAM	14 minutes 6 seconds	7.17%

Moreover, we can compare the root-mean-square error of A matrix. As shown in table 5.6, the root-mean-square errors of the original version is 9.15%, and the root-mean-square error of the improved version is only 7.17%. That is, the improved version is only about 80% to the original version. Thus, we can see that in this dataset, the improved CAM is not only much faster, but also more accurate.

As mentioned in the first paragraph, PCA is not suitable for visualizing the high dimensional simplex. Thus, here we apply the simplex visualization method mentioned in the sub-section 5.2.4 to this dataset, and the result is shown in figure 5.9. One can see that this time there are clear six vertices in the simplex, which is the correct number for the simplex. Also, from the figure, it is obvious that the distances between the neighboring vertices are not the same, which are decided by the distances in the original space.

Next, we again apply COT on the estimated S matrix to identify the SMGs, also shown in **Figure 5.9**. One can see that the detected SMGs (red, blue, green, orange, purple, and cyan circles) are all located in the vertices of the simplex, which meets the requirements of SMG, so the performance of COT is also quite good on this dataset.

In these two sub-sections, we compare the deconvolution results of original CAM and the improved version for two different datasets with ground truth. The results show that the computation speed of the improved version is much faster. Moreover, the result of the improved version is comparable to the original one, and sometimes even more accurate. Thus, the improved version is a better choice, especially for faster speed when dealing with a large dataset.

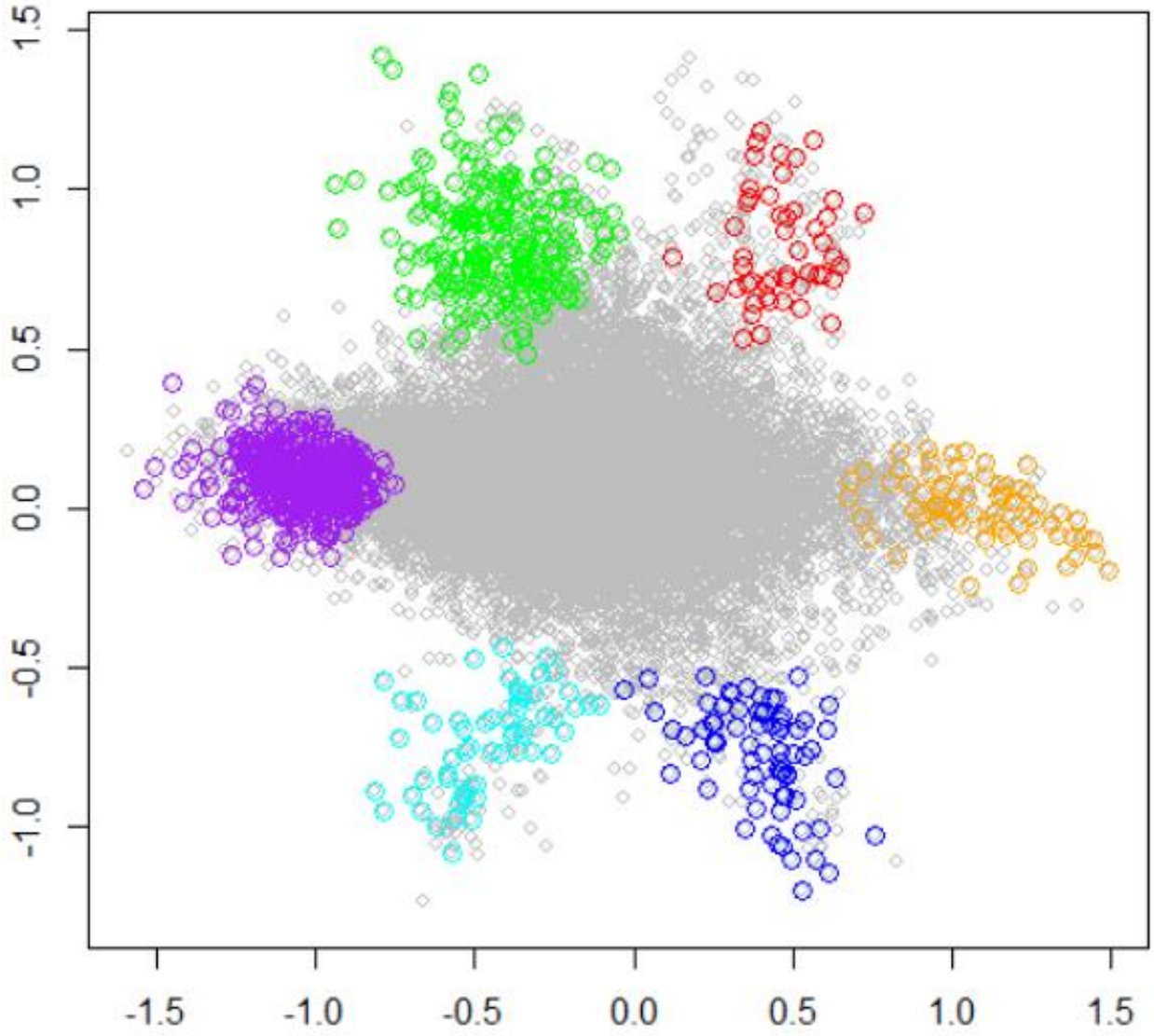

Figure 5.9 simplex plot and SMG detected by COT. The simplex plot is by the method mentioned in the sub-section 5.2.4, which can reserve the vertices of the simplex, and also consider the distances among the vertices. Thus, the six vertices are reserved after projection, which is better than PCA (**Figure 5.7**). The red, blue, green, orange, purple, and cyan circles are the SMGs detected by COT (each color corresponding to one source), and one can see that they are all located in the vertices of the simplex, which meets the requirements of SMG. Therefore, this figure not only shows our simplex plot method is good, but also demonstrate our COT is a good SMG detection method.

5.3.3 Application on RNA dataset of cortex tissues

In this sub-section, we apply the improved CAM on a benchmark human dataset, Braincloud (GSE30272) [94]. The samples in this dataset are all human prefrontal cortex of post-mortem brains at different ages, which include fetus, infant, child, teen, adult, and aged. There are 269 samples and 30176 RNAs in this dataset. This is a very precious dataset, since it is from before birth to 70s with enough sample number at each stage, which is across the whole human lifespan. Thus, applying CAM on this dataset can help us understand the tissue population development in brain at each stage. The definition at each stage [93] and the corresponding sample number is shown in table 5.7.

Table 5-7 Period definition and sample number. The definition of each period is from [93]. Though there some period of fetus with no samples, we still have enough samples for fetus at the other periods.

Period	Description	Age (year)	Sample
1	Embryonic	-0.65 ~ -0.58	0
2	Early fetal	-0.58 ~ -0.54	0
3	Early fetal	-0.54 ~ -0.48	4
4	Early mid-fetal	-0.48 ~ -0.42	9
5	Early mid-fetal	-0.42 ~ -0.37	25
6	Late mid-fetal	-0.37 ~ -0.27	0
7	Late fetal	-0.27 ~ 0	0
8	Neonatal and early infancy	0 ~ 0.5	17
9	Late infancy	0.5 ~ 1	1
10	Early childhood	1 ~ 6	12
11	Middle and late childhood	6 ~ 12	4
12	Adolescence	12 ~ 20	49
13	Young adulthood	20 ~ 40	53
14	Middle adulthood	40 ~ 60	73
15	Late adulthood	60 ~	22

First, we can still try to visualize the dataset by PCA as a 2D plot, as shown in **Figure 5.10**, and the "simplex" is again similar to a triangle. However, it is not reasonable that there are only three tissue / cell types in human prefrontal cortex. Thus, we again show that PCA is not a good tool for visualizing high-dimensional simplex.

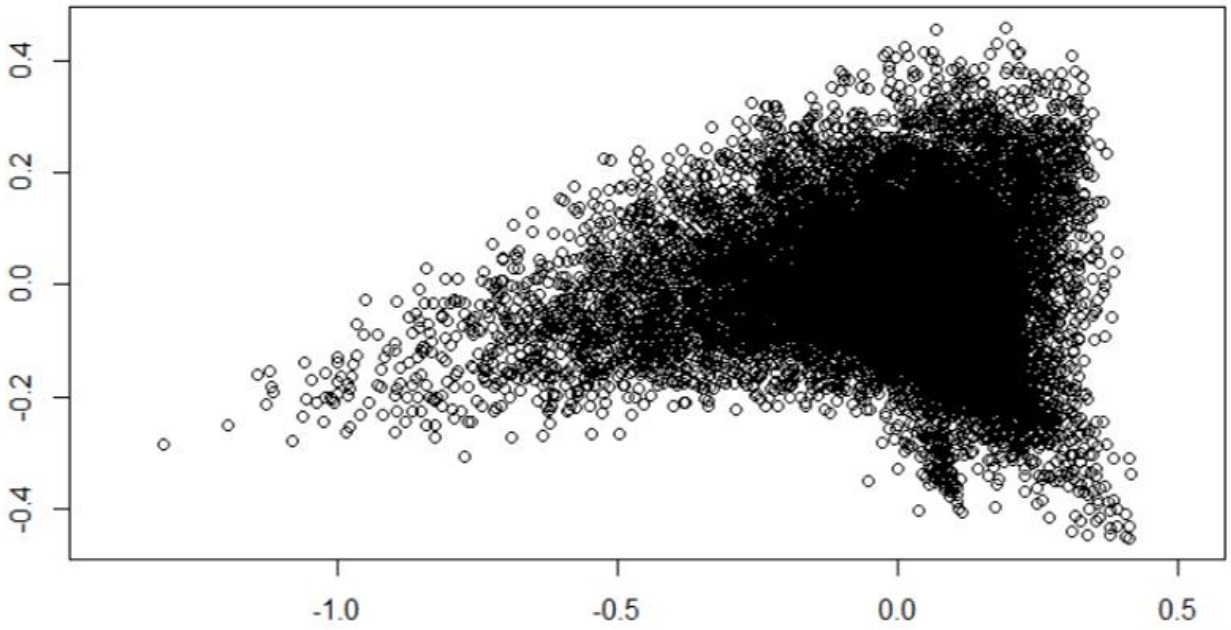

Figure 5.10 2D visualization by PCA. After the dimension reduction by PCA, the "simplex" here is similar to a triangle, however, which should contain six vertices (six sources). Thus, this example shows that PCA is not a good method for visualizing a high dimensional simplex.

Likewise, before CAM, we apply our deep-learning model for the source number pre-estimation. This time the pre-estimation of the source number is six. Thus, since we do not know the actual number of the source, based on the pre-estimation, we applied PCA to reduce the number of dimensions to ten. That is, if the pre-estimation is not accurate, our MDL criterion could still help us identify the number of sources from two to ten.

Next, after data processing, convex hull identification, optimal simplex identification, and **A** and **S** matrix estimation by the improved CAM package, we can view the MDL curve to decide the number of sources. As shown in **Figure 5.11**, the lowest point of the MDL curve is at seven, which is different from the one (six) by the pre-estimation. Though the estimations by the deep learning model and the MDL curve are not exactly the same, their difference is only one, which still shows

the both estimations are similar. That is, they cross-validate each other. Moreover, the one more source by the MDL curve may show that the MDL criterion is more sensitive to the deep learning model. That is, the MDL criterion may detect one more subtype which is very similar to the one among the six.

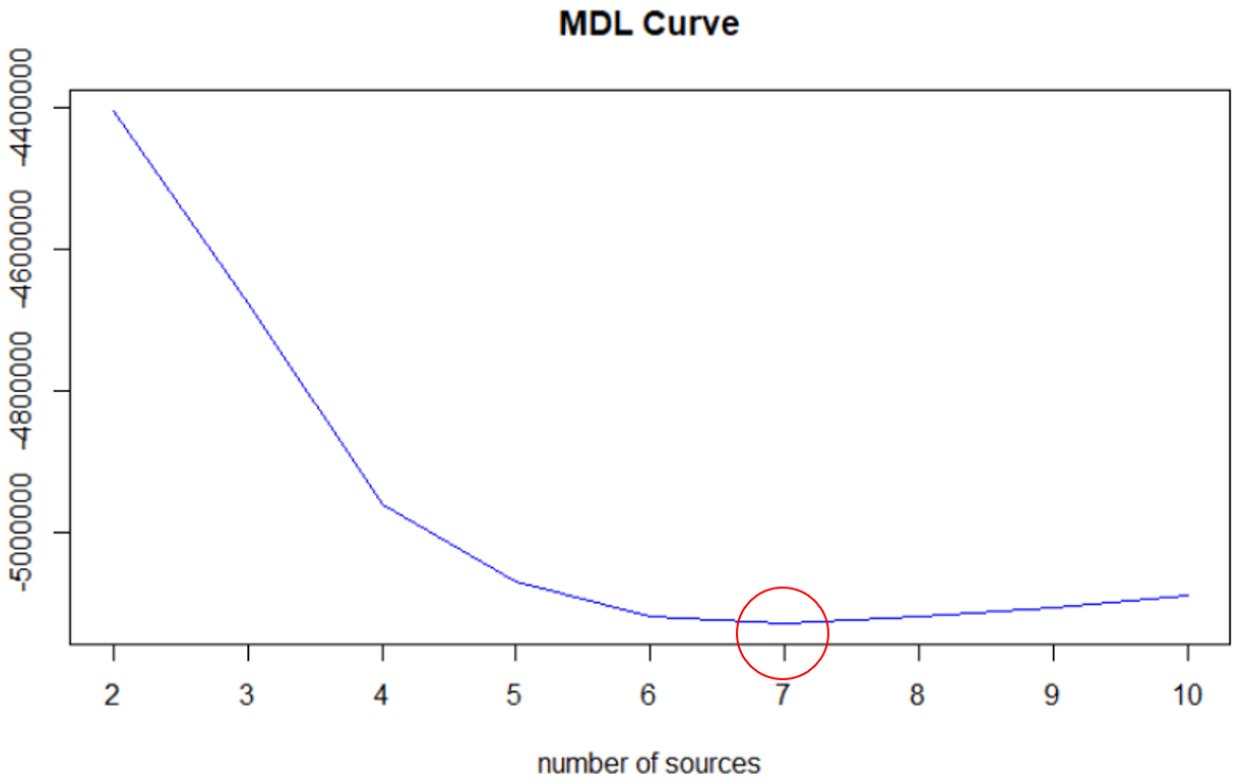

Figure 5.11 MDL curve for GSE30272. The lowest point of the MDL curve is at seven. Though it is different from the pre-estimation by the deep learning model, they are still very similar (only one difference).

After deconvolution and source number estimation, we can again visualize the dataset by our simplex plot, and identify the markers by COT, as shown in **Figure 5.12**. This time it is not a triangle anymore, and we can see that there are about seven vertices. Also, the markers detected by COT are all located around the vertices.

Since there is no ground truth for this dataset, to understand which cell types identified by CAM, we collect some "prior markers" from the literature [107] to compare the ones we identify by COT. The prior markers are from astrocytes, mature oligodendrocyte, progenitor cell, and neuron, totally

four cell types. The marker comparison of the four cell types between the seven sources identified by CAM is shown in **Table 5-8**.

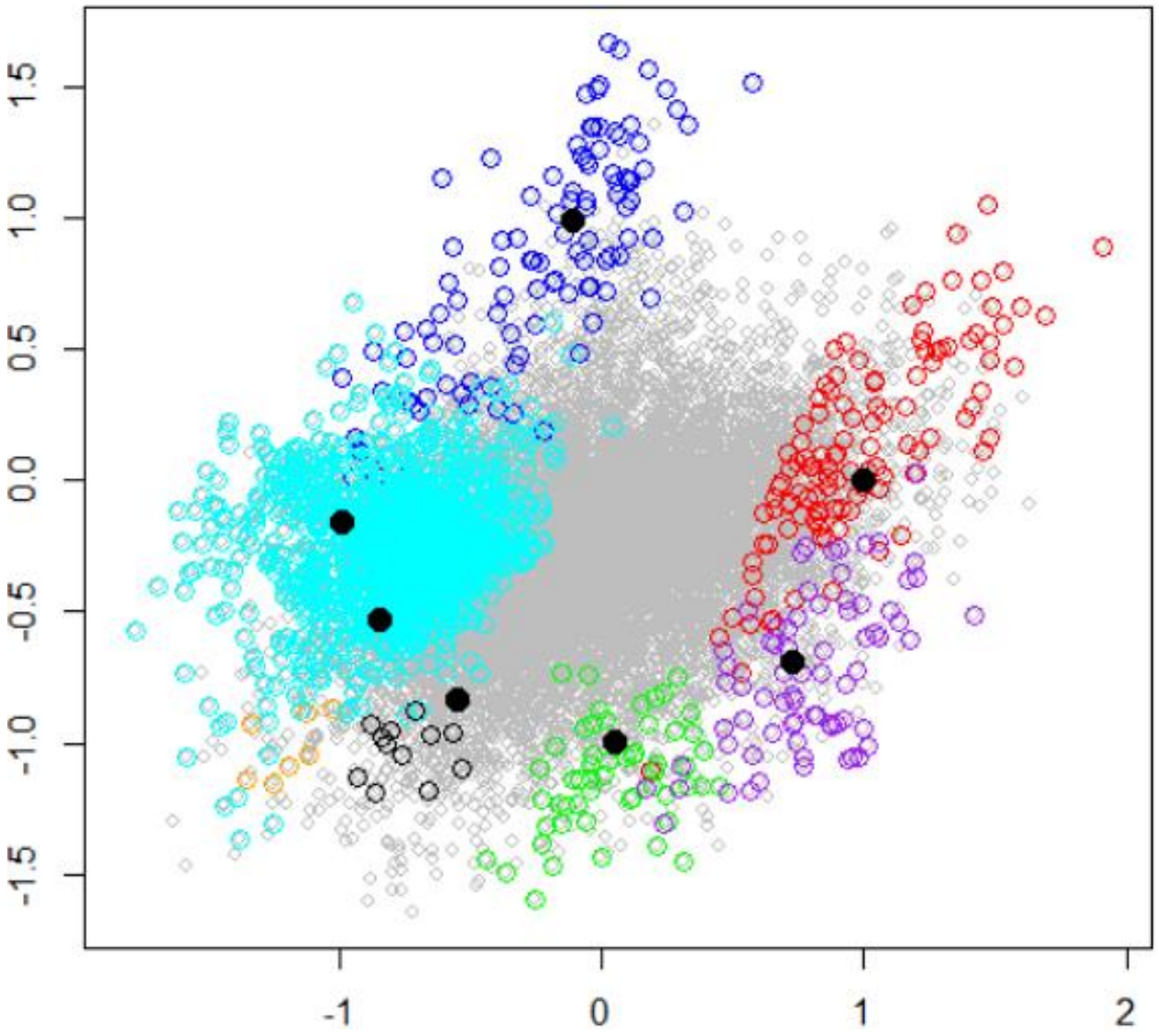

Figure 5.12 simplex plot and SMG detected by COT. The simplex plot is by the method mentioned in the sub-section 5.2.4. The red, blue, green, orange, purple, cyan, and black circles are the SMGs detected by COT (each color corresponding to one source), and one can see that they are all located in the vertices of the simplex, which meets the requirements of SMG. The black dots are the positions of the estimated A matrix (each column). From the figure, it is clear that the blue source is far from the others, so we can know that the proportions of the blue source should be very different to the other six sources.

Table 5-8 Overlapping between prior markers and COT markers. From the table, we can know that the blue source could be progenitor cell, the green source could be oligodendrocyte, and the orange one could be astrocytes.

Overlapping markers	Astrocytes (50)	Oligodendrocyte (31)	Progenitor cell (35)	Neuron (21)
Red	0	0	0	1
Blue	0	0	7	0
Green	0	21	0	0
Orange	10	0	1	0
Purple	0	0	0	1
Cyan	10	1	2	0
Black	0	0	0	0

From the **Table 5-8**, we can see that there seven overlapping markers between the blue source and the progenitor cell, so we can assume that the blue source could be the progenitor cell. Also, since there are 21 overlapping markers, the green source could be oligodendrocyte. Moreover, there are ten overlapping markers, the orange source could be astrocyte. The cyan source is very interesting, since the overlapping markers include astrocytes, oligodendrocyte, and progenitor cell. Thus, one possibility is that the cyan source represents the transition stage of the progenitor cell. That is, the progenitor cell is differentiating. The red and purple sources have only one overlapping markers with neuron, so perhaps they are two different subtypes of neuron. The black source has no overlapping to any prior markers, so it may be a novel unknown subtype, or simply a known cell type but lack of known markers.

As mentioned, as a time series dataset, we can see the cell type proportion transitions across time after deconvolution. That is, we can plot the **A** matrix with different time period, as shown in figure 5.13. The x-axis is the period defined in the **Table 5-7**, and the y-axis is the proportion of the sources. The points represent the samples in the same period after averaging for each source. From the figure, we can see that the blue source (progenitor cell) decreases significantly after birth, which is reasonable since most of the progenitor cells differentiate during the fetus stage [108].

Thus, the increasing of green (oligodendrocyte), orange (astrocyte), and red (neuron) sources is also expected. The cyan source is always in low proportion, the reason may be that the differentiating progenitor cell would not be too many at the same time. Nevertheless, the proportion of the black source is always no lower than 15%. Thus, this unknown cell type in human brain deserves further investigation. In conclusion, the improved CAM successfully identified several meaningful sources which are corresponding to known subtypes, and identified some novel subtypes. Also, the deconvolution result of CAM may help us understand the development of human brain at different period.

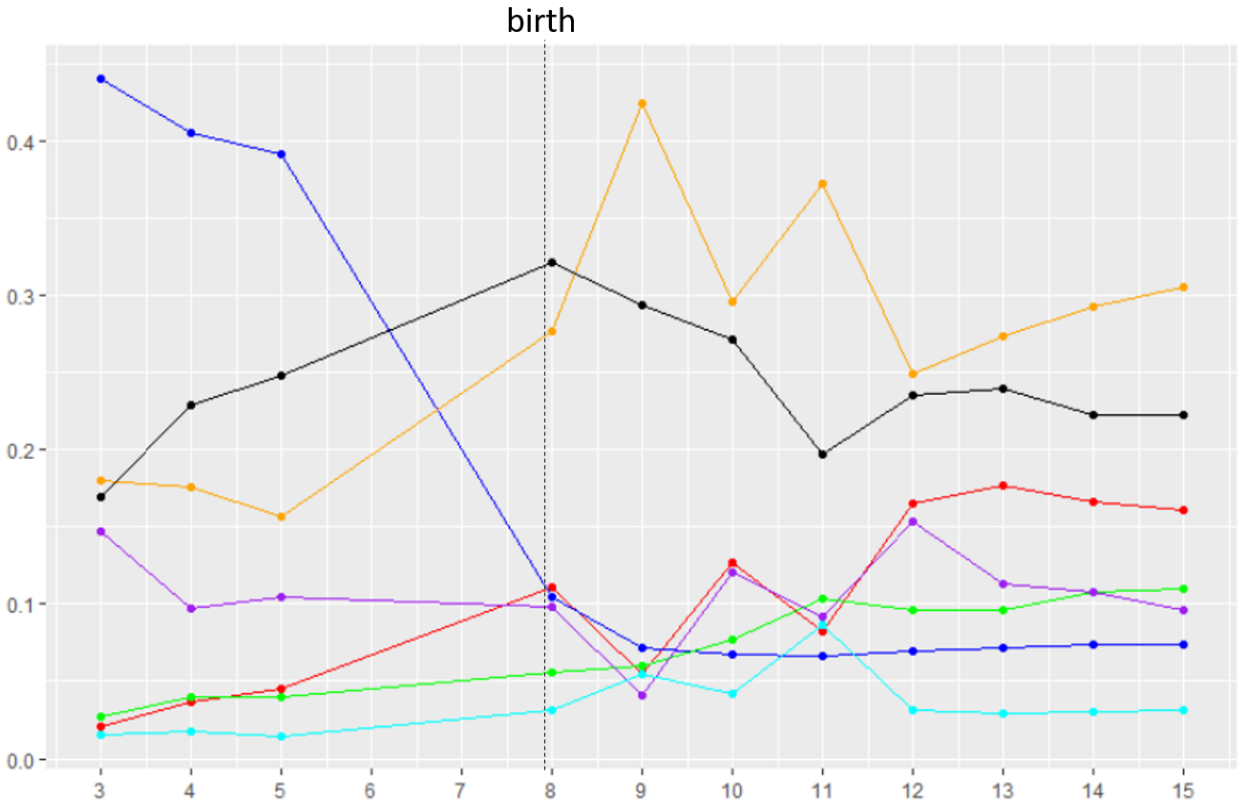

Figure 5.13 Proportion of each source (A matrix) at different period. The period is defined in the **Table 5-7**. The points represent the samples in the same period after averaging for each source. The blue source (progenitor cell) decreases significantly after birth. The green (oligodendrocyte), orange (astrocyte), and red (neuron) sources is increasing after birth. The cyan source is always in low proportion, the reason may be that the differentiating progenitor cell would not be too many at the same time. Nevertheless, the proportion of the black source is always no lower than 15%. Thus, this unknown cell type in human brain deserves further investigation.

5.4 Discussions

Though our deep-learning model is effective and accurate, the information we really use is not too much. That is, we just consider the singular values of the data in the model, so the application of the model is very limited. Thus, if we want to use deep-learning techniques for deconvolution, we need to set the whole dataset as input. While we can still produce training data by simulation, the size of different dataset could be very different. Therefore, it is hard to train a model which can accept any kind of dataset. It is also the reason that we only use the top 30 singular values, which is a compromise when dealing with datasets with very different sizes.

It is possible that our MDL criterion is not always accurate. However, with our pre-estimation, the source number indicated by the MDL curve is still a good reference. That is, after deconvolution, we can examine the **A** and **S** matrixes for different source numbers around the estimation by the MDL curve. With the expectations and knowledge of biology, we may further decide the source number more accurately. Also, with the estimation, the workload for the experts may decrease significantly.

To demonstrate the global structure of the simplex, we propose a new way to plot the simplex in 2D space, which is better than PCA. However, there are still some limitations of this method. For example, we need to first detect the vertices of the simplex, so that our method can decide how to project the whole simplex, which could be very time-consuming (all steps in CAM package) comparing with PCA or the other visualization methods. Moreover, the current idea for the projection is to project all the vertices onto a circle, which limit the shape of the projected simplex significantly. Therefore, if it is possible that the projected vertices can be onto a more flexible shape, the visualization may be more convincing and meaningful.

For the dataset with missing values, currently we suggest to use any existing missing value imputation method as initialization for CAM. With the deconvolution result from CAM, the imputed missing values can be replaced if needed. However, it is possible that CAM can perform imputation without any initialization by the other methods. One of the examples is that one peer method, NMF, can impute the missing values without any prior knowledge [109]. Thus, if we can also find ways to deal with missing values in each step of CAM, CAM can perform the

deconvolution without any problem. For example, the in the clustering step, we may neglect the missing values when computing the distance / similarity. Since the assumptions of CAM all follow biology knowledge, the imputation by CAM may better than the other methods. However, the details of the algorithm and performance should be further discussed.

In the results section, we apply CAM on three different datasets. However, for the different purpose of deconvolution, the preprocessing steps may need further discussion. For example, in the default preprocessing step, we suggest PCA for the dimension reduction. While in the sub-section 5.3.3, the purpose of the deconvolution is to view the proportions of different sources at different period. After deconvolution, we take average for the samples in the same period for **A** matrix. Since our target is for the "averaging proportions", it may be better to average the samples in the same period at the beginning, which also reduce the dimensions of samples, so the PCA is not needed. However, the effectiveness also needs further examinations.

5.5 Summary

In this chapter, we introduce the whole pipeline of the improved CAM package. The first step is source number pre-estimation. With the help of the deep-learning model, we can give a quick estimation of the source number with the singular values of the dataset. After gene (feature) clustering, convex hull identification, optimal simplex identification, and **A** and **S** matrix estimation with different source number, we can plot a curve with the MDL criterion. The lowest point of the MDL curve indicates the final estimation of the source number. When the source number is decided, we can identify the SMGs for each source and plot them on the simplex visualization with the corresponding estimated **A** and **S** matrix.

In the section 5.3, we compare the deconvolution results from the original CAM and improved CAM on two datasets with ground truth, and the evaluations show that their results are comparable, and sometimes the improved CAM is more accurate. Moreover, the computation time of the improved CAM is much lower than the original one, which can be three times to ten times or even faster, depending on the size of the dataset.

In the sub-section 5.3.3, we apply the improved CAM on the Braincould dataset, and the **A** matrix shows that the proportions of different sources across time are meaningful and biologically

reasonable. Also, CAM may identify several interesting subtypes which may need further investigations. Thus, in conclusion, the deconvolution by CAM is very useful and with wide applications for different biological purposes.

Chapter 6

Contributions and Future Work

6.1 Summary of Contributions

In this dissertation, we proposed several ideas to improve the preprocessing steps of the biological data. To eliminate different types of confounding factors thoroughly, we concluded the two major steps of preprocessing, and suggested methods to improve each of them respectively. These two steps, alignment and deconvolution, are common in the pre-processing, especially in the omics data. Thus, the improvement of all these two steps should increase the quality of further analysis significantly for any kinds of data. To test our ideas, we chose metabolomics data and gene expression data to examine our methods.

6.1.1 Multiple alignment

For alignment, we proposed a new alignment method, ncGTW, which can utilize the structure information in the dataset when aligning. The structure information here means the relationship among the samples in the data. For example, the running order of each sample or the samples from the same batch or not. Moreover, this new alignment method does not need to select a certain reference. Instead, ncGTW considers all the samples as references and extracts all pairwise information. These two properties make ncGTW very suitable for the alignment of biological data, since biological data often suffer from high noise and large variation across samples. Structure information can help ncGTW insensitive to the noise, and reference-free allows ncGTW to avoid the reference-picking problem. Also, comparing with other profile-based methods, the performance of ncGTW is the best in our comparisons, no matter for simulations or real datasets. Though we just demonstrated the results of LC-MS data, ncGTW can apply on any alignment-needed dataset in principle, even if there is no structure information.

For the LC-MS data, we designed several methods to overcome the disadvantages of ncGTW. Since the computation time is too long, comparing with feature-based methods, we proposed an algorithm to detect where the misalignments may happen to avoid aligning the whole dataset. In

addition, to fix the peak shape distortion, we utilize the information of peak position to make sure that the shape of the peaks would not change after alignment. In other words, we build a complete software package, which can detect and fix the misalignments in the LC-MS dataset. The comparison of XCMS and ncGTW also shows that with the help of ncGTW, we can obtain more accurate alignment after the alignment done by XCMS.

6.1.2 Unsupervised deconvolution

For deconvolution, we improved an unsupervised deconvolution method, CAM. In the clustering step of CAM, since the size of the clusters by K-means is hard to control, and the speed of APC is extremely slow, we propose radius-fixed clustering to control the size of the clusters easily with reasonable speed. Also, clustering and outlier removal are done by radius-fixed clustering at the same time. In the convex hull identification step, we use linear programming to replace Quickhull, and the computation time decreased dramatically. To avoid the exhaustive search for the combinatorial optimization problem, we use SFFB and SBFS to identify the optimal simplex. Nevertheless, we directly use reconstruction error in the search, so the accuracy is comparable the original margin of error approach but much faster speed. Inspired by the idea of radius-fixed clustering, we propose a new method for detecting the marker genes. This cosine-based method matches the definition of SMGs better than the other methods, as shown in the simplex plot and heatmap of the gene expressions.

To apply CAM on a real dataset, we further improved the whole pipeline of the improved CAM package. The first step is source number pre-estimation. With the help of the deep-learning model, we can give a quick estimation of the source number with the singular values of the dataset. After gene (feature) clustering, convex hull identification, optimal simplex identification, and **A** and **S** matrix estimation with different source number, we can plot a curve with the MDL criterion. The lowest point of the MDL curve indicates the final estimation of the source number. When the source number is decided, we can identify the SMGs for each source and plot them on the simplex visualization with the corresponding estimated **A** and **S** matrix.

We compared the deconvolution results from the original CAM and improved CAM on two datasets with ground truth, and the evaluations show that their results are comparable, and sometimes the improved CAM is more accurate. Moreover, the computation time of the improved

CAM is much lower than the original one, which can be three times to ten times or even faster, depending on the size of the dataset. Moreover, we apply the improved CAM on the Braincould dataset, and the **A** matrix shows that the proportions of different sources across time are meaningful and biologically reasonable. Also, CAM may identify several interesting subtypes which may need further investigations. Thus, in conclusion, the deconvolution by CAM is very useful and with wide applications for different biological purposes.

6.2 Future work

Based on the research results demonstrated in Chapter 2-5, there are still several potential research topics that may deserve to be further explored. Here we outline some directions which may improve the current methods for alignment and deconvolution. Also, there may be some new applications for our ncGTW and CAM.

6.2.1 Multiple alignment

6.2.1.1 Decrease the computation time of ncGTW

Since ncGTW needs to consider all possible sample pairs when aligning, the computation time of ncGTW is significantly high, especially comparing with feature-based methods. The core algorithm of ncGTW is solving the maximum flow minimum cut problem, which is a classic graph problem and currently there are tons of methods to solve it. However, the graph built by ncGTW is very special, which is like a linked DTW problem. Thus, if we can investigate some special properties for such graph, and then utilize them, perhaps we can design an algorithm for such special graph with a much faster computation time.

6.2.1.2 Apply ncGTW on other types of data

In this dissertation, we just demonstrated the results of LC-MS data, but in principle, ncGTW can align any kind of data, even without the structure information. Thus, it is very interesting to see the performance of ncGTW on other types of data, especially for those with different structure information. For example, in some studies, people need to compare the DNA from different species, but their DNA may be different so that we cannot tell which part corresponding to which part. One method is to align them first. If we apply ncGTW with the evolution stage as the structure

information (tree-like structure), it would be interesting to see the performance comparing with other methods.

6.2.1.3 Utilize the second stage of ncGTW

The second stage of ncGTW is to combine the pairwise alignments from the first stage, and these two stages work separately. That is, the second stage of ncGTW can also accept the information from any other kinds of alignment method, if they can provide the pairwise alignment information. For example, we can apply DTW to obtain all pairwise alignments and send them to the second stage of ncGTW, where the second stage can also give the final alignment results. In other words, the second stage of ncGTW can help any alignment method to avoid the reference-picking problem, which is a hard problem for many methods. Moreover, by adding different sets of edges, the second stage of ncGTW can accept pairwise alignments from not only one method, but as many as the user wish. That is, the second stage can combine the results from different alignment method, and we can adjust the weight of each method by setting different costs on those different sets of edges.

6.2.2 Unsupervised deconvolution

6.2.2.1 Improve the clustering step in CAM

While we propose radius-fixed clustering for better controlling the size of each cluster, the computation time of which is much slower than K-means clustering. Thus, if we can combine the advantages of these two methods, the integrated clustering method would be faster and very suitable for initialization of CAM. For example, we can use K-means clustering as an initialization, and then apply the idea of the radius-fixed clustering to make sure the size of each cluster is no more than a certain threshold. However, in this way, the uncertainty of the K-means clustering is still an issue, so we still need to think the other method to solve it. The other possibility is that we can initialize the K-means clustering with some seeds picking by the idea similar to the radius-fixed clustering. For example, select the feature with the most neighbors as the starting point, and then obtain the next one by finding the one which is farthest to the first one. The third one should be again the farthest to the first and the second one…… until the enough number of seeds is obtained. However, this idea still needs further refinement.

The other aspect is that if we can obtain the optimal simplex directly on the original data (with clustering), then we do not need the clustering step. For example, if we can adjust the constraints

of linear programming, we may formulate a way to identify an approximated convex hull, which may be free to or not sensitive to the noise. Moreover, we can consider to apply robust linear programming to find convex hull with some randomness, so that the noise may not affect the detected simplex significantly.

6.2.2.2 Try alternative methods for optimal simplex identification

For the optimal simplex identification problem, since CAM formulates it as a combinatorial optimization problem, it is an NP-hard problem, so there is no perfect solution. However, besides the greedy search algorithms, there is another type method which use lasso-related techniques. That is, with a suitable penalty term (usually including L1-norm and L2-norm), by adjusting the parameter of the penalty term, we can obtain different number of candidates left, and the left candidates are corresponding to the vertices of the simplex. However, since we need to test a range of possible source number in the real applications, it would be time-consuming to try a lot of parameters to obtain all the needed numbers of the sources. That is, the computation time of this type of method is not guaranteed, but the accuracy may be better than ours, since it is not a greedy algorithm.

6.2.2.3 Apply the encoder part of transformer for source number pre-estimation

We apply the deep-learning technique for source number pre-estimation with the singular values from the dataset. However, if we can analyze the whole dataset directly, we may obtain more accurate result. One of the promising deep learning techniques for dealing the whole dataset is transformer [110]. Though transformer is for natural language processing, it is possible that we can apply it on the omics datasets. When transformer dealing a translation problem, for example, translating a sentence in English to French, transformer will first use encoder part to "extract" the needed information from the data, and then decoder part to translate it to French. Thus, if we consider the dataset is as a sentence, and a gene is a word in the sentence, we may also utilize the encoder to extract the "number of sources" information from the dataset.

Moreover, if we tune the "self-attention" successfully, it is possible that the self-attention can help us to identify the SMGs in the dataset, which will make the deconvolution problem much easier. If we have the SMGs, it is equivalent to we have the vertices of the simplex, so the problem becomes a simplex NNLS problem. Also, it is possible that with the decoder part of transformer,

we can perform the deconvolution by transformer itself, but this idea still needs further refinement and investigation.

6.2.2.4 Improve the visualization of high-dimensional simplex

To demonstrate the global structure of the simplex, we propose a new way to plot the simplex in 2D space. However, one of the limitations is that the current idea for the projection is to project all the vertices onto a circle, which limit the shape of the projected simplex significantly. One possible way for improvement is that we can initialize it as a triangle (the first three points), and add the other points one by one to form more triangles sequentially. That is, every time we add a new point, we can consider the distances between the closest two points in the projected points to form a new triangle, and make sure that the whole projection is still "convex" after adding the new triangle (the projection is still a convex polygon). In this way, the projection will not be limited on the circle anymore. However, we still need to develop an algorithm which can make sure the projection is always a convex polygon during the steps.

6.2.2.5 Apply CAM for confounding variable identification

Though in the experiments we always apply CAM for identifying the different tissue / cell types in the mixture, it is possible that CAM can identify the confounding variables. That is, the confounding variables, for example, age and gender can also be sources in the mixture if they have some contribution. In the other words, there may be some genes which are highly correlated with age or gender, and these are the causes of the confounding variables, which may affect the further analysis if we do not correct them. If we have some information of the confounding variable, such as the age or gender of every sample, we can compare the information with the vertex candidates in the optimal simplex identification step. If there are some candidates which are highly correlated with the information, then they may be caused by the confounding variables and we should remove them. Also, after deconvolution, if there are some sources which are very strange or unexpected, we can further examine them, since they may be some unknown confounding variables, and we should remove them right away.

Appendix A

Personal Information

A.1 Biographical sketch

Chiung-Ting Wu received the B.S. degree in physics and the M.S. degree in photonics and optoelectronics from National Taiwan University, Taipei, Taiwan, in 2010 and 2012, respectively. Since August 2014, he has been pursuing a Ph.D. degree in electrical engineering with the Bradley Department of Electrical and Computer Engineering at Virginia Polytechnic Institute and State University (Virginia Tech), Virginia, United States, under the supervision of Dr. Yue Wang. His research focuses on applications of machine learning and signal processing to computational biology.

A.2 Publications

*Equal contribution;

Lulu Chen, **Chiung-Ting Wu**, Chia-Hsiang Lin, Rujia Dai, Chunyu Liu, Robert Clarke, Guoqiang Yu, Jennifer E. Van Eyk, David M. Herrington, and Yue Wang. "swCAM: estimation of subtype-specific expressions in individual samples with unsupervised sample-wise deconvolution." *Bioinformatics* (in revision).

Minjie Shen*, Yi-Tan Chang*, **Chiung-Ting Wu***, Sarah J Parker, Georgia Saylor, Yizhi Wang, Guoqiang Yu, Jennifer E Van Eyk, Robert Clarke, and David M Herrington. "Comparative Assessment and Outlook on Methods for Imputing Proteomics Data." *Research Square* (2021).

Lulu Chen, Yingzhou Lu, **Chiung-Ting Wu**, Robert Clarke, Guoqiang Yu, Jennifer E Van Eyk, David M Herrington, and Yue Wang. "Data-Driven Detection of Subtype-Specific Differentially Expressed Genes." *Scientific reports* 11.1 (2021): 1-12.

Yingzhou Lu*, **Chiung-Ting Wu***, Sarah J. Parker, Lulu Chen, Georgia Saylor, Jennifer E. Van Eyk, David M. Herrington, and Yue Wang. "COT: An Efficient Python Tool for Detecting Marker Genes among Many Subtypes." *bioRxiv* (2021): 2021.01.10.426146.

Lulu Chen, **Chiung-Ting Wu**, Niya Wang, David M Herrington, Robert Clarke, and Yue Wang. "Debcam: A Bioconductor R Package for Fully Unsupervised Deconvolution of Complex Tissues." *Bioinformatics* 36.12 (2020): 3927-29.

Chiung-Ting Wu, Yizhi Wang, Yinxue Wang, Timothy Ebbels, Ibrahim Karaman, Gonçalo Graça, Rui Pinto, David M Herrington, Yue Wang, and Guoqiang Yu. "Targeted Realignment of Lc-Ms Profiles by Neighbor-Wise Compound-Specific Graphical Time Warping with Misalignment Detection." *Bioinformatics* 36.9 (2020): 2862-71.

Yizhi Wang, Congchao Wang, Petter Ranefall, Gerard Joey Broussard, Yinxue Wang, Guilai Shi, Boyu Lyu, **Chiung-Ting Wu**, Yue Wang, Lin Tian, and Guoqiang Yu. "Synquant: An Automatic Tool to Quantify Synapses from Microscopy Images." *Bioinformatics* 36.5 (2020): 1599-606.

Guoqiang Yu, David J Miller, **Chiung-Ting Wu**, Eric P Hoffman, Chunyu Liu, David M Herrington, and Yue Wang. "Asymmetric Independence Modeling Identifies Novel Gene-Environment Interactions." *Scientific reports* 9.1 (2019): 1-9.

Congchao Wang, Yizhi Wang, Yinxue Wang, **Chiung-Ting Wu**, and Guoqiang Yu. "Mussp: Efficient Min-Cost Flow Algorithm for Multi-Object Tracking." *Advances in Neural Information Processing Systems* 32 (2019): 425-34.

Yingzhou Lu, Yi-Tan Chang, Eric P. Hoffman, Guoqiang Yu, David M. Herrington, Robert Clarke, **Chiung-Ting Wu**, Lulu Chen, and Yue Wang. "Integrated Identification of Disease Specific Pathways Using Multi-Omics Data." *bioRxiv* (2019): 666065.

Bibliography

[1] G. J. Patti, O. Yanes, and G. Siuzdak, "Innovation: Metabolomics: the apogee of the omics trilogy," *Nat Rev Mol Cell Biol,* vol. 13, no. 4, pp. 263-9, Mar 22 2012, doi: 10.1038/nrm3314.

[2] D. J. Lockhart and E. A. J. N. Winzeler, "Genomics, gene expression and DNA arrays," *Natrue,* vol. 405, no. 6788, p. 827, 2000.

[3] Z. Wang, M. Gerstein, and M. Snyder, "RNA-Seq: a revolutionary tool for transcriptomics," *Nature reviews genetics,* vol. 10, no. 1, p. 57, 2009.

[4] R. Aebersold and M. Mann, "Mass spectrometry-based proteomics," *Nature,* vol. 422, no. 6928, p. 198, 2003.

[5] R. Goodacre, S. Vaidyanathan, W. B. Dunn, G. G. Harrigan, and D. B. Kell, "Metabolomics by numbers: acquiring and understanding global metabolite data," *Trends Biotechnol,* vol. 22, no. 5, pp. 245-52, May 2004, doi: 10.1016/j.tibtech.2004.03.007.

[6] M. J. J. C. d. Meaney, "Epigenetics and the biological definition of gene× environment interactions," *Child development,* vol. 81, no. 1, pp. 41-79, 2010.

[7] M. J. Heller, "DNA microarray technology: devices, systems, and applications," *Annu Rev Biomed Eng,* vol. 4, pp. 129-53, 2002, doi: 10.1146/annurev.bioeng.4.020702.153438.

[8] A. T. Bankier, "Shotgun DNA Sequencing," in *DNA Sequencing Protocols*, C. A. Graham and A. J. M. Hill Eds. Totowa, NJ: Humana Press, 2001, pp. 89-100.

[9] J. Shendure and H. Ji, "Next-generation DNA sequencing," *Nat Biotechnol,* vol. 26, no. 10, pp. 1135-45, Oct 2008, doi: 10.1038/nbt1486.

[10] B. Wilcken, V. Wiley, J. Hammond, and K. Carpenter, "Screening newborns for inborn errors of metabolism by tandem mass spectrometry," *New England Journal of Medicine,* vol. 348, no. 23, pp. 2304-2312, 2003.

[11] M. Kainosho, T. Torizawa, Y. Iwashita, T. Terauchi, A. M. Ono, and P. J. N. Güntert, "Optimal isotope labelling for NMR protein structure determinations," *Nature,* vol. 440, no. 7080, p. 52, 2006.

[12] A. M. Weljie, J. Newton, P. Mercier, E. Carlson, and C. M. Slupsky, "Targeted profiling: quantitative analysis of 1H NMR metabolomics data," *Analytical chemistry,* vol. 78, no. 13, pp. 4430-4442, 2006.

[13] W. M. Niessen, *Liquid chromatography-mass spectrometry*. Crc Press, 2006.

[14] J. Lisec, N. Schauer, J. Kopka, L. Willmitzer, and A. R. Fernie, "Gas chromatography mass spectrometry-based metabolite profiling in plants," *Nat Protoc,* vol. 1, no. 1, pp. 387-96, 2006, doi: 10.1038/nprot.2006.59.

[15] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg, "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome," *Genome Biol,* vol. 10, no. 3, p. R25, 2009, doi: 10.1186/gb-2009-10-3-r25.

[16] 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," *Analytical chemistry,* vol. 78, no. 3, pp. 779-787, 2006.

[17] T. N. Vu and K. Laukens, "Getting your peaks in line: a review of alignment methods for NMR spectral data," *Metabolites,* vol. 3, no. 2, pp. 259-76, Apr 15 2013, doi: 10.3390/metabo3020259.

[18] M. J. Lee, Y. Wu, and S. K. Fried, "Adipose tissue heterogeneity: implication of depot differences in adipose tissue for obesity complications," *Mol Aspects Med,* vol. 34, no. 1, pp. 1-11, Feb 2013, doi: 10.1016/j.mam.2012.10.001.
[19] S. Pantalacci and M. Sémon, "Transcriptomics of developing embryos and organs: a raising tool for evo–devo," *Journal of Experimental Zoology Part B: Molecular and Developmental Evolution,* vol. 324, no. 4, pp. 363-371, 2015.
[20] S. Moco, J. Forshed, R. C. H. De Vos, R. J. Bino, and J. Vervoort, "Intra- and inter-metabolite correlation spectroscopy of tomato metabolomics data obtained by liquid chromatography-mass spectrometry and nuclear magnetic resonance," *Metabolomics,* journal article vol. 4, no. 3, pp. 202-215, September 01 2008, doi: 10.1007/s11306-008-0112-8.
[21] A. S. Benk and C. Roesli, "Label-free quantification using MALDI mass spectrometry: considerations and perspectives," *Analytical and bioanalytical chemistry,* vol. 404, no. 4, pp. 1039-1056, 2012.
[22] K. Podwojski *et al.*, "Retention time alignment algorithms for LC/MS data must consider non-linear shifts," *Bioinformatics,* vol. 25, no. 6, pp. 758-764, 2009.
[23] N. Wang *et al.*, "Mathematical modelling of transcriptional heterogeneity identifies novel markers and subpopulations in complex tissues," *Sci Rep,* vol. 6, p. 18909, Jan 7 2016, doi: 10.1038/srep18909.
[24] E. P. Hoffman, "Guidelines: Expression profiling-best practices for data generation and interpretation in clinical trials," *Nature Reviews Genetics,* vol. 5, no. 3, 2004.
[25] S. Marguerat and J. Bähler, "RNA-seq: from technology to biology," *Cellular and molecular life sciences,* vol. 67, no. 4, pp. 569-579, 2010.
[26] F. Avila Cobos, J. Vandesompele, P. Mestdagh, and K. De Preter, "Computational deconvolution of transcriptomics data from mixed cell populations," *Bioinformatics,* vol. 34, no. 11, pp. 1969-1979, 2018.
[27] L. A. Herzenberg, D. Parks, B. Sahaf, O. Perez, M. Roederer, and L. A. Herzenberg, "The history and future of the fluorescence activated cell sorter and flow cytometry: a view from Stanford," *Clinical chemistry,* vol. 48, no. 10, pp. 1819-1827, 2002.
[28] E. Kummari, S. X. Guo-Ross, and J. B. Eells, "Laser capture microdissection-a demonstration of the isolation of individual dopamine neurons and the entire ventral tegmental area," *Journal of visualized experiments: JoVE,* no. 96, 2015.
[29] T. Nawy, "Single-cell sequencing," *Nature methods,* vol. 11, no. 1, pp. 18-18, 2014.
[30] S. S. Shen-Orr *et al.*, "Cell type–specific gene expression differences in complex tissues," *Nature methods,* vol. 7, no. 4, pp. 287-289, 2010.
[31] A. Kuhn, D. Thu, H. J. Waldvogel, R. L. Faull, and R. Luthi-Carter, "Population-specific expression analysis (PSEA) reveals molecular changes in diseased brain," *Nature methods,* vol. 8, no. 11, pp. 945-947, 2011.
[32] A. M. Newman *et al.*, "Robust enumeration of cell subsets from tissue expression profiles," *Nature methods,* vol. 12, no. 5, pp. 453-457, 2015.
[33] S. Mohammadi, N. Zuckerman, A. Goldsmith, and A. Grama, "A critical survey of deconvolution methods for separating cell types in complex tissues," *Proceedings of the IEEE,* vol. 105, no. 2, pp. 340-366, 2016.
[34] N. S. Zuckerman, Y. Noam, A. J. Goldsmith, and P. P. Lee, "A self-directed method for cell-type identification and separation of gene expression microarrays," *PLoS Comput Biol,* vol. 9, no. 8, p. e1003189, 2013.

[35] E. Oja and M. Plumbley, "Blind separation of positive sources by globally convergent gradient search," *Neural Computation,* vol. 16, no. 9, pp. 1811-1825, 2004.
[36] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature,* vol. 401, no. 6755, pp. 788-791, 1999.
[37] L. Chen, C.-T. Wu, N. Wang, D. M. Herrington, R. Clarke, and Y. Wang, "debCAM: a bioconductor R package for fully unsupervised deconvolution of complex tissues," *Bioinformatics,* vol. 36, no. 12, pp. 3927-3929, 2020.
[38] C.-T. Wu *et al.,* "Targeted realignment of LC-MS profiles by neighbor-wise compound-specific graphical time warping with misalignment detection," *Bioinformatics,* vol. 36, no. 9, pp. 2862-2871, 2020.
[39] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS),* vol. 22, no. 4, pp. 469-483, 1996.
[40] N.-P. V. Nielsen, J. M. Carstensen, and J. Smedsgaard, "Aligning of single and multiple wavelength chromatographic profiles for chemometric data analysis using correlation optimised warping," *Journal of chromatography A,* vol. 805, no. 1-2, pp. 17-35, 1998.
[41] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, 2014, pp. 1764-1772.
[42] F. Sievers *et al.,* "Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega," *Mol Syst Biol,* vol. 7, p. 539, Oct 11 2011, doi: 10.1038/msb.2011.75.
[43] R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic Acids Res,* vol. 32, no. 5, pp. 1792-7, 2004, doi: 10.1093/nar/gkh340.
[44] J. Listgarten, R. M. Neal, S. T. Roweis, and A. Emili, "Multiple alignment of continuous time series," in *Advances in neural information processing systems*, 2005, pp. 817-824.
[45] R. Smith, D. Ventura, and J. T. Prince, "LC-MS alignment in theory and practice: a comprehensive algorithmic review," *Brief Bioinform,* vol. 16, no. 1, pp. 104-17, Jan 2015, doi: 10.1093/bib/bbt080.
[46] B. Korte, J. Vygen, B. Korte, and J. Vygen, *Combinatorial optimization*. Springer, 2012.
[47] Y. Wang, D. J. Miller, K. Poskanzer, Y. Wang, L. Tian, and G. Yu, "Graphical time warping for joint alignment of multiple curves," in *Advances in Neural Information Processing Systems*, 2016, pp. 3648-3656.
[48] F. Petitjean, A. Ketterlin, and P. Gançarski, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern Recognition,* vol. 44, no. 3, pp. 678-693, 2011.
[49] W. Jiang *et al.,* "Comparisons of five algorithms for chromatogram alignment," *Chromatographia,* vol. 76, no. 17-18, pp. 1067-1078, 2013.
[50] B. Efron and C. Morris, "Stein's paradox in statistics," *Scientific American,* vol. 236, no. 5, pp. 119-127, 1977.
[51] L. N. Mueller *et al.,* "SuperHirn - a novel tool for high resolution LC-MS-based peptide/protein profiling," *Proteomics,* vol. 7, no. 19, pp. 3470-80, Oct 2007, doi: 10.1002/pmic.200700057.
[52] G. Theodoridis, H. G. Gika, and I. D. Wilson, "LC-MS-based methodology for global metabolite profiling in metabonomics/metabolomics," *TrAC Trends in Analytical Chemistry,* vol. 27, no. 3, pp. 251-260, 2008, doi: 10.1016/j.trac.2008.01.008.

[53] W. Lu, B. D. Bennett, and J. D. Rabinowitz, "Analytical strategies for LC-MS-based targeted metabolomics," *J Chromatogr B Analyt Technol Biomed Life Sci*, vol. 871, no. 2, pp. 236-42, Aug 15 2008, doi: 10.1016/j.jchromb.2008.04.031.

[54] M. Vinaixa, S. Samino, I. Saez, J. Duran, J. J. Guinovart, and O. Yanes, "A Guideline to Univariate Statistical Analysis for LC/MS-Based Untargeted Metabolomics-Derived Data," *Metabolites*, vol. 2, no. 4, pp. 775-95, Oct 18 2012, doi: 10.3390/metabo2040775.

[55] A. Prakash *et al.*, "Signal maps for mass spectrometry-based comparative proteomics," *Mol Cell Proteomics*, vol. 5, no. 3, pp. 423-32, Mar 2006, doi: 10.1074/mcp.M500133-MCP200.

[56] J. T. Prince and E. M. Marcotte, "Chromatographic alignment of ESI-LC-MS proteomics data sets by ordered bijective interpolated warping," *Analytical Chemistry*, vol. 78, no. 17, pp. 6140-6152, 2006.

[57] X. Zhang, J. M. Asara, J. Adamec, M. Ouzzani, and A. K. Elmagarmid, "Data pre-processing in liquid chromatography-mass spectrometry-based proteomics," *Bioinformatics*, vol. 21, no. 21, pp. 4054-9, Nov 1 2005, doi: 10.1093/bioinformatics/bti660.

[58] A. V. Goldberg, S. Hed, H. Kaplan, R. E. Tarjan, and R. F. Werneck, "Maximum flows by incremental breadth-first search," in *European Symposium on Algorithms*, 2011: Springer, pp. 457-468.

[59] M. Palmblad, D. J. Mills, L. V. Bindschedler, and R. Cramer, "Chromatographic alignment of LC-MS and LC-MS/MS datasets by genetic algorithm feature extraction," *Journal of the American Society for Mass Spectrometry*, vol. 18, no. 10, pp. 1835-1843, 2007.

[60] E. Tengstrand, J. Lindberg, and K. M. Å berg, "TracMass 2 A Modular Suite of Tools for Processing Chromatography-Full Scan Mass Spectrometry Data," *Analytical chemistry*, vol. 86, no. 7, pp. 3435-3442, 2014.

[61] B. C. Arnold, N. Balakrishnan, and H. N. Nagaraja, *A first course in order statistics*. Siam, 1992.

[62] R. J. Connor, "The Sampling Distribution of the Range from Discrete Uniform Finite Populations and a Range Test for Homogeneity," *Journal of the American Statistical Association*, vol. 64, no. 328, pp. 1443-1458, 1969, doi: 10.1080/01621459.1969.10501070.

[63] D. E. Bild *et al.*, "Multi-Ethnic Study of Atherosclerosis: objectives and design," *Am J Epidemiol*, vol. 156, no. 9, pp. 871-81, Nov 1 2002, doi: 10.1093/aje/kwf113.

[64] A. Hofman *et al.*, "The Rotterdam Study: 2014 objectives and design update," *Eur J Epidemiol*, vol. 28, no. 11, pp. 889-926, Nov 2013, doi: 10.1007/s10654-013-9866-z.

[65] M. H. Sarafian *et al.*, "Objective set of criteria for optimization of sample preparation procedures for ultra-high throughput untargeted blood plasma lipid profiling by ultra performance liquid Chromatography–Mass spectrometry," *Analytical chemistry*, vol. 86, no. 12, pp. 5766-5774, 2014.

[66] M. R. Lewis *et al.*, "Development and Application of Ultra-Performance Liquid Chromatography-TOF MS for Precision Large Scale Urinary Metabolic Phenotyping," *Anal Chem*, vol. 88, no. 18, pp. 9004-13, Sep 20 2016, doi: 10.1021/acs.analchem.6b01481.

[67] C. Christin, A. K. Smilde, H. C. Hoefsloot, F. Suits, R. Bischoff, and P. L. Horvatovich, "Optimized time alignment algorithm for LC− MS data: correlation optimized warping using component detection algorithm-selected mass chromatograms," *Analytical chemistry*, vol. 80, no. 18, pp. 7012-7021, 2008.

[68] B. Matuszewski, M. Constanzer, and C. Chavez-Eng, "Matrix effect in quantitative LC/MS/MS analyses of biological fluids: a method for determination of finasteride in human plasma at picogram per milliliter concentrations," *Analytical Chemistry*, vol. 70, no. 5, pp. 882-889, 1998.
[69] P. H. Lopez and R. L. Schnaar, "Gangliosides in cell recognition and membrane protein regulation," *Current opinion in structural biology*, vol. 19, no. 5, pp. 549-557, 2009.
[70] G. Van Meer, D. R. Voelker, and G. W. Feigenson, "Membrane lipids: where they are and how they behave," *Nature reviews Molecular cell biology*, vol. 9, no. 2, p. 112, 2008.
[71] W. R. Parrish *et al.*, "Modulation of TNF release by choline requires α7 subunit nicotinic acetylcholine receptor-mediated signaling," *Molecular medicine*, vol. 14, no. 9-10, pp. 567-574, 2008.
[72] H. Ueda, T. Kobayashi, M. Kishimoto, T. Tsutsumi, and H. Okuyama, "A possible pathway of phosphoinositide metabolism through EDTA-insensitive phospholipase A1 followed by lysophosphoinositide-specific phospholipase C in rat brain," *Journal of neurochemistry*, vol. 61, no. 5, pp. 1874-1881, 1993.
[73] E. A. Houseman, M. L. Kile, D. C. Christiani, T. A. Ince, K. T. Kelsey, and C. J. Marsit, "Reference-free deconvolution of DNA methylation data and mediation by cell composition effects," *BMC bioinformatics*, vol. 17, no. 1, pp. 1-15, 2016.
[74] Y. Hart *et al.*, "Inferring biological tasks using Pareto analysis of high-dimensional data," *Nature methods*, vol. 12, no. 3, pp. 233-235, 2015.
[75] D. M. Herrington *et al.*, "Proteomic architecture of human coronary and aortic atherosclerosis," *Circulation*, vol. 137, no. 25, pp. 2741-2756, 2018.
[76] R. A. Moffitt *et al.*, "Virtual microdissection identifies distinct tumor-and stroma-specific subtypes of pancreatic ductal adenocarcinoma," *Nature genetics*, vol. 47, no. 10, p. 1168, 2015.
[77] R. Schwartz and S. E. Shackney, "Applying unmixing to gene expression data for tumor phylogeny inference," *BMC bioinformatics*, vol. 11, no. 1, pp. 1-20, 2010.
[78] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 33, no. 2, pp. 387-392, 1985.
[79] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1967, vol. 1, no. 14: Oakland, CA, USA, pp. 281-297.
[80] R. Kelley and T. Ideker, "Systematic interpretation of genetic interactions using protein networks," *Nature biotechnology*, vol. 23, no. 5, pp. 561-566, 2005.
[81] P. M. Pardalos, Y. Li, and W. Hager, "Linear programming approaches to the convex hull problem in Rm," *Computers & Mathematics with Applications*, vol. 29, no. 7, pp. 23-29, 1995.
[82] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern recognition letters*, vol. 15, no. 11, pp. 1119-1125, 1994.
[83] L. Chen *et al.*, "Data-driven detection of subtype-specific differentially expressed genes," *Scientific reports*, vol. 11, no. 1, pp. 1-12, 2021.
[84] Y. Lu *et al.*, "COT: an efficient Python tool for detecting marker genes among many subtypes," *bioRxiv*, p. 2021.01.10.426146, 2021, doi: 10.1101/2021.01.10.426146.
[85] G. J. Hunt, S. Freytag, M. Bahlo, and J. A. Gagnon-Bartsch, "dtangle: accurate and robust cell type deconvolution," *Bioinformatics*, vol. 35, no. 12, pp. 2093-2099, 2019.

[86] G. Yu *et al.*, "Matched gene selection and committee classifier for molecular classification of heterogeneous diseases," *The Journal of Machine Learning Research,* vol. 11, pp. 2141-2167, 2010.
[87] F. Allantaz *et al.*, "Expression profiling of human immune cell subsets identifies miRNA-mRNA regulatory relationships correlated with cell type specific expression," *PloS one,* vol. 7, no. 1, p. e29979, 2012.
[88] E. Becht *et al.*, "Estimating the population abundance of tissue-infiltrating immune and stromal cell populations using gene expression," *Genome biology,* vol. 17, no. 1, pp. 1-20, 2016.
[89] A. Ben-Tal and A. Nemirovski, "Robust solutions of linear programming problems contaminated with uncertain data," *Mathematical programming,* vol. 88, no. 3, pp. 411-424, 2000.
[90] I. Charon and O. Hudry, "The noising method: a new method for combinatorial optimization," *Operations Research Letters,* vol. 14, no. 3, pp. 133-137, 1993.
[91] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* vol. 68, no. 1, pp. 49-67, 2006.
[92] B. B. Lake *et al.*, "Neuronal subtypes and diversity revealed by single-nucleus RNA sequencing of the human brain," *Science,* vol. 352, no. 6293, pp. 1586-1590, 2016.
[93] H. J. Kang *et al.*, "Spatio-temporal transcriptome of the human brain," *Nature,* vol. 478, no. 7370, pp. 483-489, 2011.
[94] C. Colantuoni *et al.*, "Temporal dynamics and genetic control of transcription in the human prefrontal cortex," *Nature,* vol. 478, no. 7370, pp. 519-523, 2011.
[95] S. Dadgar *et al.*, "Asynchronous remodeling is a driver of failed regeneration in Duchenne muscular dystrophy," *Journal of Cell Biology,* vol. 207, no. 1, pp. 139-158, 2014.
[96] T.-H. Chan, W.-K. Ma, C.-Y. Chi, and Y. Wang, "A convex analysis framework for blind separation of non-negative sources," *IEEE Transactions on Signal Processing,* vol. 56, no. 10, pp. 5120-5134, 2008.
[97] S. J. Parker *et al.*, "Identification of Putative Early Atherosclerosis Biomarkers by Unsupervised Deconvolution of Heterogeneous Vascular Proteomes," *Journal of proteome research,* vol. 19, no. 7, pp. 2794-2806, 2020.
[98] L. Chen *et al.*, "CAM-CM: a signal deconvolution tool for in vivo dynamic contrast-enhanced imaging of complex tissues," *Bioinformatics,* vol. 27, no. 18, pp. 2607-2609, 2011.
[99] W. Hu, R. Liu, X. Lin, Y. Li, X. Zhou, and X. He, "A deep learning method to estimate independent source number," in *2017 4th International Conference on Systems and Informatics (ICSAI)*, 2017: IEEE, pp. 1055-1059.
[100] Y. Yang, F. Gao, C. Qian, and G. Liao, "Model-aided deep neural network for source number detection," *IEEE Signal Processing Letters,* vol. 27, pp. 91-95, 2019.
[101] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research,* vol. 9, no. 11, 2008.
[102] E. Becht *et al.*, "Dimensionality reduction for visualizing single-cell data using UMAP," *Nature biotechnology,* vol. 37, no. 1, pp. 38-44, 2019.
[103] A. D. Polpitiya *et al.*, "DAnTE: a statistical tool for quantitative analysis of-omics data," *Bioinformatics,* vol. 24, no. 13, pp. 1556-1558, 2008.

[104] A. W.-C. Liew, N.-F. Law, and H. Yan, "Missing value imputation for gene expression data: computational techniques to recover missing data from available information," *Briefings in bioinformatics,* vol. 12, no. 5, pp. 498-513, 2011.
[105] M. Shen *et al.*, "Comparative Assessment and Outlook on Methods for Imputing Proteomics Data," 2021.
[106] H. Wold, "Soft modelling by latent variables: the non-linear iterative partial least squares (NIPALS) approach," *Journal of Applied Probability,* vol. 12, no. S1, pp. 117-142, 1975.
[107] X. Xu, A. Nehorai, and J. D. Dougherty, "Cell type-specific analysis of human brain transcriptome data to predict alterations in cellular composition," *Systems Biomedicine,* vol. 1, no. 3, pp. 151-160, 2013.
[108] M. W. Ferenczy, K. R. Johnson, L. J. Marshall, M. C. Monaco, and E. O. Major, "Differentiation of human fetal multipotential neural progenitor cells to astrocytes reveals susceptibility factors for JC virus," *Journal of virology,* vol. 87, no. 11, pp. 6221-6231, 2013.
[109] R. Chen and L. R. Varshney, "Non-negative matrix factorization of clustered data with missing values," in *2019 IEEE Data Science Workshop (DSW)*, 2019: IEEE, pp. 180-184.
[110] A. Vaswani *et al.*, "Attention is all you need," *arXiv preprint arXiv:1706.03762,* 2017.