Linear Spectral Unmixing Algorithms for Abundance Fraction Estimation in Spectroscopy

CHANGIN OH

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF ARTS

GRADUATE PROGRAM IN MATHEMATICS AND STATISTICS
YORK UNIVERSITY
TORONTO, ONTARIO

January 2023

© Changin Oh, 2023

Abstract

Fluorescence spectroscopy is commonly used in modern biological and chemical studies, especially for cellular and molecular analysis. Since the measured fluorescence spectrum is the sum of the spectrum of each fluorophore in a sample, a reliable separation of fluorescent labels is the key to the successful analysis of the sample. A technique known as linear spectral unmixing is often used to linearly decompose the measured fluorescence spectrum into a set of constituent fluorescence spectra with abundance fractions.

Various algorithms have been developed for linear spectral unmixing. In this work, we implement the existing linear unmixing algorithms and compare their results to discuss their strengths and drawbacks. Furthermore, we apply optimization methods to the linear unmixing problem and evaluate their performance to demonstrate their capabilities of solving the linear unmixing problem. Finally, we denoise noisy fluorescence emission spectra and examine how noise may affect the performance of the algorithms.

Table of Contents

List of Tables

List of Figures

Chapter 1

In this chapter, we present a general description of the theoretical background on linear spectral mixture analysis and a mathematical problem known as the linear unmixing problem. Furthermore, we provide a brief introduction about existing linear unmixing methods and optimization techniques applicable to the linear unmixing problem.

1.1 Motivation

Fluorescence spectroscopy is commonly used in modern biological and chemical studies, especially for cellular and molecular analysis [1]–[6]. Spectroscopy is primarily concerned with how matter interacts with electromagnetic radiation. A molecule undergoes transitions between discrete (or quantized) energy states by absorbing or emitting a photon, a packet of light. Fluorescence occurs when a molecule absorbs a photon at one wavelength and reemits a photon at a longer wavelength

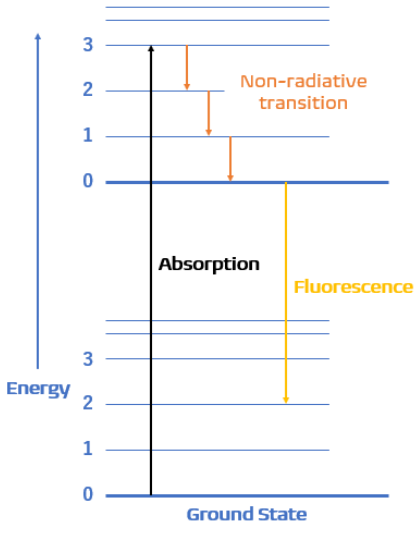

Figure 1.1. Jablonski diagram representing vibrational levels for absorbance, non-radiative decay, and fluorescence

(i.e., the energy or frequency of the incident light is different from that of the emitted light). The mechanism of fluorescence is described in the Jablonski diagram (Figure 1.1). In this process, a fluorophore (or a fluorescent molecule) is first excited from its ground electronic state to one of the many vibrational states in an excited electronic state. As it collides with other molecules, the molecule undergoes non-radiative vibrational relaxation until it reaches the lowest vibrational state in the excited electronic state. When it returns to any of several vibrational states in the ground electronic state, it fluoresces or emits a photon that has a lower energy (and thus frequency) than the incident photon. Afterwards, the photons are directed towards a filter and onto a detector for measurement and identification of the molecules; the detection and analysis of the intensities and frequencies of the photons yields fluorescence spectra which contain useful information to determine the structure of the vibrational energy levels of the molecules. Using the information, one can depict what the molecules are, how much of them are present, how they are interacting with other molecules within the sample, and so forth [7].

Many practical applications of fluorescence spectroscopy have been made in science and industry. In chemical industries, for instance, it has been used to detect polycyclic aromatic hydrocarbons and analyze dissolved organic carbon in water [1], [2]. It has been also used to characterize the emission of new synthesized materials to examine their electronic properties for optoelectronic applications [3]. Fluorescence spectroscopy also plays a vital role in biological studies since it enables one to analyze changes in the tertiary structure of proteins, detect specific bacterial strains using fluorescent assays, and determine the melting temperature of deoxyribonucleic acid (DNA) and ribonucleic acid (RNA) samples [4]–[6]. The application is still growing remarkably as a powerful and effective tool to study the physical and chemical behaviour of molecules.

In fluorescence spectroscopy, multiple fluorophores are generally used as markers for labeling [8]. Therefore, a reliable separation of fluorescent labels is the key to the successful characterization, identification, and analysis of a sample; the decomposition of the measured fluorescence spectrum is essential since it is the sum of the spectrum of each fluorophore in the sample. There is, however, an inherent problem with multiple fluorescent labeling due to the severe overlap of different fluorophore spectra [9]. This is caused by their wide emission ranges which results in the measured fluorescence spectrum possibly not being completely decomposed into the representations of the fluorophores in the sample. Therefore, an unambiguous identification may be impossible. To circumvent such a technical issue, a technique known as linear spectral unmixing is often used to decompose a fluorescence spectrum. It is known that nonlinear approaches can generate more accurate, robust abundance fractions than the linear approach [10]. Nevertheless, the linear unmixing technique is often used for solving spectral unmixing problem because of its simplicity and efficiency [11]. The obtained results are promising, despite the simplicity of the technique [12].

Spectral unmixing problem aims to decompose the measured fluorescence spectrum into a set of constituent fluorescence spectra and abundance fractions that indicate the contribution of each constituent fluorescence spectrum [13]. For this reason, the measured fluorescence spectrum is called a mixture fluorescence spectrum, and the constituent fluorescence spectra are called reference fluorescence spectra. Linear spectral unmixing is built on the assumption that a mixture fluorescence spectrum can be decomposed linearly into reference fluorescence spectra. Consequently, assuming that reference fluorescence spectra are linearly combined via a linear mixture model, a mixture fluorescence spectrum can be unmixed in terms of abundance fractions of the reference fluorescence spectra.

Fluorescence is a highly sensitive analytical technique [14] and hence a fluorescent signal may contain noise, an unintended fluctuation in a signal, due to the sensitivity of the spectrofluorometer. Even though some level of noise can never be removed on account of the particulate nature of light, excess noise arising from imperfections in equipment and conditions can be theoretically minimized or eliminated [15]. Denoising is a signal processing method that minimizes the effect of noise in a noisy signal (i.e., a mixture of signal and noise) to preserve useful information [16]. Since a fluorescence spectrum is a fluorescent signal, denoising may lead to better estimations of abundance fractions by linear unmixing if the spectrum incorporates unwanted noise.

1.2 Mathematical Formulation of the Problem

Linear spectral mixture analysis is widely used in fluorescence spectroscopy to estimate the abundance fractions for the decomposition of a mixture spectrum into a set of given reference spectra on the assumption that the mixture spectrum is the linear combination of the reference spectra [8]. Fluorescence emission follows the principle of linear superposition which states that a system can be decomposed into its constituent components and the behaviour of each component is independent of the other components [17], [18]. This fact enables the emission spectrum of a mixture of fluorophores to be expressed as the sum of its components, that is, the reference emission spectra of the fluorophores. Suppose that the $p \times 1$ column vector $\vec{r}$ represents the mixture emission spectrum and the $p \times 1$ column vector $\vec{m}_i$ represents the i^{th} reference emission spectrum of the mixture emission spectrum $\vec{r}$ with p emission wavelength detection channels. Then the mixture emission spectrum can be described as

$$\alpha_1\vec{m}_1 + \alpha_2\vec{m}_2 + \cdots + \alpha_l\vec{m}_l = \vec{r} \tag{1.1}$$

or simply

$$\sum_{i=1}^{l} \alpha_i\vec{m}_i = \vec{r} \tag{1.2}$$

where l is the number of fluorophores in the sample and α_i is the abundance fraction of the corresponding reference emission spectrum $\vec{m}_i$. Eq. (1.2) can be shown in matrix-vector form as

$$M\vec{\alpha} = \vec{r} \tag{1.3}$$

where M is the $p \times l$ matrix of reference emission spectra for the l individual reference emission spectra arranged in columns and $\vec{\alpha}$ is the $l \times 1$ column vector containing the abundance fractions of the l individual reference emission spectra. Considering additive noise in the emission spectra, the linear spectral mixture model is generalized as

$$M\vec{\alpha} + \vec{n} = \vec{r} \tag{1.4}$$

with the $p \times 1$ noise vector $\vec{n}$. The matrix M is called the mixing matrix of the linear spectral mixture model. The linear unmixing is a mathematical technique to solve the model for the abundance vector $\vec{\alpha}$ to estimate the abundance fractions of the l individual reference emission spectra [17].

For the uniqueness of the solution to the linear spectral mixture model, the number of emission wavelength detection channels in a spectrofluorometer must be greater than or equal to the number of fluorophores in the sample. Without this condition, multiple solutions are possible and hence no unique solution can be obtained for the linear unmixing. This condition thus implies that the linear spectral mixture model must be a determined or overdetermined system [8].

On the assumption that the linear spectral mixture model cannot be reduced to an underdetermined system, it is necessary to find an approximate solution to the model for abundance

fraction estimation. By interpreting the noise vector $\vec{n}$ of the model (1.4) as the least squares error, the linear unmixing problem can be cast as an optimization problem

$$\underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}) \tag{1.5}$$

where

$$J(\vec{\alpha}) = \frac{1}{2}\|\vec{n}\|^2 = \frac{1}{2}\|\vec{r} - M\vec{\alpha}\|^2 \tag{1.6}$$

Therefore, linear unmixing is now to find the abundance vector $\vec{\alpha}$ which minimizes the least squares error function (1.6).

1.3 Algorithms

Scharf and Friedlander [19] developed a simple algorithm to tackle the linear unmixing problem as an unconstrained least squares (ULS) problem. This algorithm may produce negative abundance values which are physically infeasible. By equating such negative fractions to zero, this method inevitably produces a suboptimal solution. To resolve this issue, Chang et al. [20] suggested an improved algorithm, sum-to-one constrained least squares (SCLS) method, and Chang and Heinz [21] proposed a method called the nonnegativity constrained least squares (NCLS) method. Each method generally yields better abundance estimations than the ULS method by imposing the abundance sum-to-one constraint (ASC) and the abundance nonnegativity constraint (ANC) on the linear mixture model, respectively, but they may produce suboptimal results when they contain negative abundance fractions in their procedures. By combining those two methods, Heinz et al. [22] devised the so-called fully constrained least squares (FCLS) method to find better estimated abundance fractions for the linear unmixing problem. Wong and Chang [23] found that the FCLS

algorithm can estimate the abundance fractions more accurately if they replace the ANC with an absolute abundance sum-to-one constraint (AASC), and their algorithm is called the modified fully constrained least squares (MFCLS) method.

Chen et al. and Theys et al. [24], [25] employed the gradient descent (GD) method, which is a local optimization technique, to solve the linear unmixing problem. They incorporated the ANC and the ASC into the gradient descent algorithm by using Karush-Kuhn-Tucker (KKT) conditions and Lagrange multipliers. This algorithm is called the fully constrained gradient descent (FC-GD) method. We, however, demonstrate that the original gradient descent scheme can be used without considering any constraints into the updating scheme if the linear unmixing problem satisfies two conditions: (i) the linear mixture model is an overdetermined system and (ii) the columns of the mixing matrix for the model are linearly independent. This is connected to the fact that the linear unmixing problem has a unique local solution and thus the original gradient descent method performs well on the linear unmixing problem if the abundance fractions are bounded between 0 and 1 while iterating over the problem.

Rather than local optimization methods, a global optimization algorithm can be considered to solve the linear unmixing problem, which guarantees robust convergence to an optimal solution to the linear unmixing problem. The standard Nelder-Mead (NM) method [26] locates an optimal solution by iteratively creating a simplex and moving it towards the optimum via four operations: reflection, expansion, contraction, and shrinkage. This technique is however very sensitive to dimensionality [27]; it performs very poorly in high dimensions. Gao and Han [28] improved its performance in high dimensions with adaptive parameters for the four operations. Their method is called the adaptive NM method.

Strictly speaking, the NM method is not a strong global optimization technique since it may fail to find the global optimum of a function with strong multimodality [29]. In this case, we should consider a strong global optimization technique. For instance, a metaheuristic technique has been designed to find a good approximate solution to the global optimization problem which is complex and difficult to solve computationally [30]. Rao et al. [31] developed the teaching-learning-based optimization (TLBO) method, which is a nature-inspired metaheuristic optimization algorithm motivated by teaching and learning processes in a classroom. The TLBO method consists of two stages, namely, teacher phase and learner phase. In the teacher phase, learners gain knowledge directly from a teacher and the quality of knowledge is dependent on the teaching skill of the teacher; the teacher phase performs a global search for optimization. In the learner phase, a learner can improve the gained knowledge with the help of other learners, which indicates that the learner phase conducts a local search for optimization. Repeating these two phases, the algorithm can finally locate the global optimum.

However, if the teacher is trapped in one of the local optima and thus cannot escape from it in the following iterations, TLBO requires too many iterations for global convergence or sometimes fails to locate the global optimum since all learners gradually moves towards the teacher. To avoid this issue, Akbari et al. [32] introduced a new phase called studying phase into TLBO; each member attempts to change and improve its position by properly changing each dimension of its position. The author named this algorithm teaching-learning-studying-based optimization (TLSBO).

Since a fluorescence spectrum is a fluorescent signal [14], the spectrum contains unintended noise as the spectrofluorometer captures it. It follows that denoising may help one obtain better estimations of abundance fractions via linear unmixing. Using the Fourier transform and the wavelet

transform [33], one can remove insignificant information (which is generally noise) from the noisy signal. These methods are known as the Fourier-based denoising method and the wavelet-based denoising method, respectively.

1.4 Contribution and Novelty of the Research

This work is the first comparative study on various linear unmixing algorithms used in fluorescence spectroscopy, and it will thus provide a useful insight into how to choose an optimal algorithm based on the given problem. In addition, we demonstrate that the typical GD scheme can be used without considering any constraints into the scheme if the linear unmixing problem satisfies two conditions: (i) the linear mixture model is an overdetermined system and (ii) the columns of the mixing matrix for the model are linearly independent. This is attributed to the fact that the linear unmixing problem has a unique local solution and thus the GD method performs well on the linear unmixing problem if the abundance fractions are bounded between 0 and 1, while iterating over the problem. The GD method is very simple to implement and finds the solution more rapidly than the FC-GD method. This study also shows how the NM method, TLBO, and TLSBO can be applied to the linear unmixing problem, since, to the best of our knowledge, these methods have not been employed for the problem despite their practical uses in the field of science and engineering. Furthermore, motivated by machine learning, we apply early stopping to TLBO and TLSBO as a termination condition so that it is no longer required to set the number of iterations for those algorithms. We also develop a metric to quantify the numbers of correct and incorrect probes detected by a linear unmixing algorithm. Lastly, we demonstrate how the Fourier-based and wavelet-based denoising methods can be used to handle the linear mixture model with random noise.

1.5 Outline of the Thesis

In this thesis, we will implement the existing linear unmixing algorithms and compare their results to discuss their strengths and drawbacks. Furthermore, we will apply optimization methods to the linear unmixing problem and evaluate their performance on the problem to demonstrate their capabilities of solving the linear unmixing problem. Finally, we will denoise noisy fluorescence emission spectra and examine how noise may affect the performance of the algorithms.

In Chapter 2, we establish the linear mixture model for linear spectral mixture analysis and present the existing linear unmixing methods with their derivations and algorithms. Chapter 3 presents an introduction to the gradient descent optimization technique, its variants, and their applications to linear spectral mixture analysis. Chapter 4 introduces global optimization techniques applicable to linear spectral mixture analysis and Chapter 5 discusses about how the Fourier-based and wavelet-based denoising methods can be used to handle noisy mixture emission spectra. In Chapter 6, we perform the comparative studies of the algorithms based on the resulting abundance estimations obtained from the algorithms. This thesis is closed in Chapter 7 with discussions on the implication of the results and on the extension of the work in future directions.

Chapter 2

Linear unmixing is a method to estimate the abundance fractions by solving the linear mixture model $M\vec{\alpha} + \vec{n} = \vec{r}$ as seen in Eq. (1.4). It can be solved with no constraints imposed on the abundance vector or with constraints on the abundance vector such as the abundance sum-to-one constraint (ASC), the abundance nonnegativity constraint (ANC), and the abundance absolute sum-to-one constraint (AASC). In this chapter, various algorithms which solve unconstrained and constrained least squares unmixing problems for the abundance estimation are presented.

2.1 Unconstrained Least Squares (ULS) Linear Unmixing Method

Scharf and Friedlander [19] proposed the ULS method to solve an unconstrained least squares problem for linear unmixing in Eq. (1.5). The least squares error function (1.6) can be expressed as

$$J(\vec{\alpha}) = \frac{1}{2}(\vec{r} - M\vec{\alpha})^T(\vec{r} - M\vec{\alpha}). \tag{2.1}$$

For the minimization of the objective function (2.1), differentiating $J(\vec{\alpha})$ with respect to $\vec{\alpha}$ and equating it to zero produces

$$\frac{\partial J(\vec{\alpha})}{\partial \vec{\alpha}} = M^T\vec{r} - M^TM\vec{\alpha} = \vec{0}. \tag{2.2}$$

Multiplying Eq. (2.2) by $(M^TM)^{-1}$, the optimal least-squares estimate of $\vec{\alpha}$ is derived as

$$\vec{\alpha}_{ULS} = (M^TM)^{-1}M^T\vec{r}. \tag{2.3}$$

However, abundances are required to be nonnegative numbers and, since it assumes no constraints imposed on the abundance vector $\vec{\alpha}$, the estimated abundance vector $\vec{\alpha}_{ULS}$ in (2.3) may contain

negative values which are meaningless in terms of abundance and may fail the sum to one. It is therefore necessary to remove the negative values in $\vec{\alpha}_{ULS}$ by setting them to zero and thus this method generally produces a suboptimal solution for the model (1.4). The algorithm to solve the ULS problem is presented in Algorithm 2.1.

2.2 Sum-to-one Constrained Least Squares (SCLS) Linear Unmixing Method

Chang et al. [20] suggested an improved algorithm, SCLS method, for linear unmixing. To obtain more accurate abundance fractions, the ASC, also known as the full additivity constraint,

$$\sum_{i=1}^{l} \alpha_i = 1 \tag{2.4}$$

must be applied to the abundance vector $\vec{\alpha}$. The linear unmixing problem with ASC can be defined as

$$\underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}) \text{ subject to } \sum_{i=1}^{l} \alpha_i = 1 \tag{2.5}$$

2.2.1 Direct Method

By introducing a Lagrange multiplier, λ, the least squares error function (2.1) can be modified to account for the ASC, yielding

$$J(\vec{\alpha}) = \frac{1}{2}(\vec{r} - M\vec{\alpha})^T(\vec{r} - M\vec{\alpha}) + \lambda\left(\vec{1}^T\vec{\alpha} - 1\right) \tag{2.6}$$

where $\vec{1}$ denotes an $l \times 1$ column vector of ones.

Differentiating in Eq. (2.6) with respect to $\vec{\alpha}$ and setting it to zero leads to

$$\frac{\partial J(\vec{\alpha})}{\partial \vec{\alpha}} = M^T\vec{r} - M^T M\vec{\alpha} + \lambda\vec{1} = \vec{0}, \tag{2.7}$$

and once again multiplying by $(M^T M)^{-1}$ yields,

$$(M^T M)^{-1}M^T\vec{r} - (M^T M)^{-1}M^T M\vec{\alpha} + \lambda(M^T M)^{-1}\vec{1} = \vec{0} \tag{2.8}$$

which simplifies to

$$\vec{\alpha}_{SCLSd} = \vec{\alpha}_{ULS} + \lambda(M^T M)^{-1}\vec{1} \tag{2.9}$$

where $\vec{\alpha}_{ULS} = (M^T M)^{-1}M^T\vec{r}$ is the solution to the unconstrained problem (2.3).

To find the value of the Lagrange multiplier λ analytically, multiplying Eq. (2.9) by $\vec{1}^T$ yields

$$\vec{1}^T\vec{\alpha}_{SCLSd} = \vec{1}^T\vec{\alpha}_{ULS} + \lambda\vec{1}^T(M^T M)^{-1}\vec{1}, \tag{2.10}$$

and due to the ASC, we know that

$$\vec{1}^T\vec{\alpha}_{SCLSd} = 1, \tag{2.11}$$

thus Eq. (2.10) becomes

$$1 = \vec{1}^T\vec{\alpha}_{ULS} + \lambda\vec{1}^T(M^T M)^{-1}\vec{1}. \tag{2.12}$$

The Lagrange multiplier λ is therefore given by

$$\lambda = \frac{1 - \vec{1}^T\vec{\alpha}_{ULS}}{\vec{1}^T(M^T M)^{-1}\vec{1}}. \tag{2.13}$$

Solving the SCLS may still produce an estimated abundance vector $\vec{\alpha}_{SCLSd}$ with negative entries because the positivity constraint is not enforced. It is thus essential to eliminate the negative values in $\vec{\alpha}_{SCLSd}$ by setting them to zero as in the ULS method, yielding a suboptimal solution. The algorithm for the direct SCLS method is summarized in Algorithm 2.1.

2.2.2 Iterative Method

We can incorporate the Lagrange multiplier slightly differently by rewriting the linear mixture model with the ASC,

$$\vec{r} = M\vec{\alpha} + \vec{n} \text{ subject to } 1 = \vec{1}^T\vec{\alpha} \tag{2.14}$$

as

$$\begin{bmatrix}\vec{r}\\ \lambda\end{bmatrix} = \begin{bmatrix}M\\ \lambda\vec{1}^T\end{bmatrix}\vec{\alpha} + \begin{bmatrix}\vec{n}\\ 0\end{bmatrix} \tag{2.15}$$

where λ is a scale variable. We define

$$\vec{r}' = \begin{bmatrix}\vec{r}\\ \lambda\end{bmatrix}, \qquad M' = \begin{bmatrix}M\\ \lambda\vec{1}^T\end{bmatrix}, \qquad \vec{n}' = \begin{bmatrix}\vec{n}\\ 0\end{bmatrix} \tag{2.16}$$

so that Eq. (2.15) becomes

$$\vec{r}' = M'\vec{\alpha} + \vec{n}'. \tag{2.17}$$

We can solve the new linear mixture model (2.17) using the ULS method with the new least squares error function,

$$J(\vec{\alpha}) = \frac{1}{2}(\vec{r}' - M'\vec{\alpha})^T(\vec{r}' - M'\vec{\alpha}) \tag{2.18}$$

and the estimated abundance vector

$$\vec{\alpha}_{SCLSi} = \left(M'^T M'\right)^{-1} M'^T\vec{r}'. \tag{2.19}$$

By increasing the value of λ in Eq. (2.15) iteratively, we can obtain the estimated abundance vector $\vec{\alpha}_{SCLSi}$. An initial parameter $\lambda = 10000$ is suggested in [34], [35]. However, they have not shown how to select a step size, h, to increase the value of λ. We found experimentally that $h = 1$ works well for this algorithm. The details about this will be discussed in Chapter 6.

Again, the estimated abundance vector $\vec{\alpha}_{SCLSi}$ may contain negative abundance fractions. Hence, by setting them to zero, the negative values in $\vec{\alpha}_{SCLSi}$ are removed. Due to this process, the method produces a suboptimal result. The algorithm for the iterative SCLS method is shown in

Algorithm 2.2

2.3 Nonnegativity Constrained Least Squares (NCLS) Linear Unmixing

Both the ULS and SCLS methods produce suboptimal solutions to the linear spectral mixture model (1.4) owing to the forced nonnegativity of the abundance fractions by equating negative values in the abundance vector to zero. Chang and Heinz [21] proposed a method called NCLS method to solve such this issue. In the nonnegativity constrained least squares problem, the ANC,

$$a_i \geq 0 \ \ (i = 1, 2, \dots, l) \tag{2.20}$$

is imposed on the abundance vector $\vec{\alpha}$. The ULS problem (1.5) with the nonnegativity constraint is expressed as

$$\underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}) \ \text{subject to} \ a_i \geq 0 \ \ (i = 1, 2, \dots, l) \tag{2.21}$$

Unlike the direct SCLS method, the NCLS method does not have an analytical solution since the ANC is formed by a set of l linear inequalities rather than equalities, implying that the Lagrange multiplier method is not applicable to solving optimal solutions [36]. In this case, the Karush-Kuhn-Tucker conditions must be used instead to develop a numerical algorithm for abundance estimation [37]. Instead, however, Chang and Heinz devised an efficient algorithm to generate NCLS linear unmixing solutions without considering the KKT conditions. To take care of the ANC, their algorithm performs dimensionality reduction on the mixing matrix of the linear mixture model. First, the NCLS algorithm uses the ULS method to estimate an abundance vector $\vec{\alpha}_{ULS}$. If the vector has negative fractions, the algorithm finds the negative value with the largest

magnitude, say α_j, and eliminates its corresponding column $\vec{m}_j$ in the mixing matrix M. The algorithm estimates a new abundance vector $\vec{\alpha}_{ULSr}$ with the reduced mixing matrix M_r, and conducts the dimensionality reduction process on M_r. Repeating this procedure, the algorithm terminates when no negative fractions are available in the abundance vector and therefore requires a maximum of $l-1$ iterations. However, since M becomes smaller and smaller due to the iterative removal of columns, so does the abundance vector $\vec{\alpha}_{ULS}$. Therefore, to obtain $\vec{\alpha}_{NCLS}$, it is necessary to record the position j of α_j in each iteration and place zeros in those positions of the final $\vec{\alpha}_{ULS}$. After this, we can finally obtain the estimated abundance vector $\vec{\alpha}_{NCLS}$. The NCLS algorithm is described in Algorithm 2.1.

2.4 Fully Constrained Least Squares (FCLS) Linear Unmixing

Heinz et al. [22] found that we can obtain more accurate solutions by simultaneously applying both constraints to the linear unmixing problem. We can express the fully constrained problem

$$\underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}) \text{ subject to } a_i \geq 0 \ (i = 1, 2, \dots, l) \text{ and } \sum_{i=1}^{l} \alpha_i = 1. \tag{2.22}$$

They developed the FCLS method which performs both the SCLS and NCLS algorithms to solve the fully constrained problem above.

2.4.1 Direct Method

The direct FCLS method is performed by implementing the direct SCLS method in conjunction with

the dimensionality reduction process in the NCLS method. The algorithm for the direct FCLS algorithm can be found in Algorithm 2.1.

2.4.2 Iterative Method

As with the direct FCLS method, the iterative method consists of the iterative SCLS method and the dimensionality reduction procedure of the NCLS method. The iterative FCLS algorithm is summarized in Algorithm 2.2.

2.5 Modified Fully Constrained Least Squares (MFCLS) Linear Unmixing

As mentioned previously, the main difficulty with solving constrained linear unmixing problems is the ANC prevents us from using the Lagrange multiplier method to find solutions analytically. Wong and Chang [23] proposed an alternative, MFCLS method, by modifying the ANC. Rather than directly handling the inequality-constraints $a_i \geq 0$ for $1 \leq i \leq l$, they are substituted with the AASC which is formulated as

$$\sum_{i=1}^{l} |\alpha_i| = 1. \tag{2.23}$$

The advantage of AASC is that the Lagrange multiplier method is applicable, enabling one to derive an algorithm to yield an optimal solution. Moreover, the AASC also allows us to preclude negative abundance fractions from the solution since all the abundance fractions become nonnegative if both the ASC and AASC are satisfied. The modified fully constrained least squares problem is then given by

$$\underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}) \text{ subject to } \sum_{i=1}^{l} \alpha_i = 1 \text{ and } \sum_{i=1}^{l} |\alpha_i| = 1 \tag{2.24}$$

2.5.1 Direct Method

Using two Lagrange multipliers, λ_1 and λ_2, we can obtain the least squares error function for the fully constrained ASC and AASC problem

$$J(\vec{\alpha}) = \frac{1}{2}\|\vec{r} - M\vec{\alpha}\|^2 + \lambda_1\left(\sum_{i=1}^{l} \alpha_i - 1\right) + \lambda_2\left(\sum_{i=1}^{l} |\alpha_i| - 1\right), \tag{2.25}$$

which we can write as

$$J(\vec{\alpha}) = \frac{1}{2}(\vec{r} - M\vec{\alpha})^T(\vec{r} - M\vec{\alpha}) + \lambda_1\left(\vec{1}^T\vec{\alpha} - 1\right) + \lambda_2(\text{sign}(\vec{\alpha})^T\,\vec{\alpha} - 1) \tag{2.26}$$

where

$$\text{sign}(\vec{\alpha}) = [\beta_1 \quad \beta_2 \quad \dots \quad \beta_l]^T = \begin{cases} \dfrac{\beta_i}{|\beta_i|} & \beta_i \neq 0 \\ 0 & \beta_i = 0 \end{cases}. \tag{2.27}$$

Differentiating $J(\vec{\alpha})$ in Eq. (2.26) with respect to $\vec{\alpha}$ and setting it to zero results in

$$\frac{\partial J(\vec{\alpha})}{\partial \vec{\alpha}} = M^T\vec{r} - M^TM\vec{\alpha} + \lambda_1\vec{1} + \lambda_2\,\text{sign}(\vec{\alpha}) = \vec{0} \tag{2.28}$$

and multiplying by $(M^TM)^{-1}$ yields

$$(M^TM)^{-1}M^T\vec{r} - (M^TM)^{-1}M^TM\vec{\alpha} + \lambda_1(M^TM)^{-1}\vec{1} + \lambda_2(M^TM)^{-1}\,\text{sign}(\vec{\alpha}) = \vec{0}. \tag{2.29}$$

Rearranging terms in Eq. (2.29) produces an analytical solution

$$\vec{\alpha}_{MFCLSd} = \vec{\alpha}_{ULS} + \lambda_1(M^TM)^{-1}\vec{1} + \lambda_2(M^TM)^{-1}\,\text{sign}(\vec{\alpha}_{MFCLSd}). \tag{2.30}$$

Multiplying Eq. (2.30) by $\vec{1}^T$,

$$\vec{1}^T\vec{\alpha}_{MFCLSd} = \vec{1}^T\vec{\alpha}_{ULS} + \lambda_1\vec{1}^T(M^TM)^{-1}\vec{1} + \lambda_2\vec{1}^T(M^TM)^{-1}\,\text{sign}(\vec{\alpha}_{MFCLSd}). \tag{2.31}$$

Since $\vec{1}^T\vec{\alpha}_{MFCLSd} = 1$ from the ASC, it can be seen that

$$1 - \vec{1}^T\vec{\alpha}_{ULS} = \lambda_1\vec{1}^T(M^TM)^{-1}\vec{1} + \lambda_2\vec{1}^T(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \tag{2.32}$$

Now multiplying Eq. (2.30) by $\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T$,

$$\begin{aligned}&\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{\alpha}_{MFCLSd}\\ &= \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{\alpha}_{ULS} + \lambda_1\,\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\vec{1}\\ &+\lambda_2\,\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd})\end{aligned} \tag{2.33}$$

By the AASC, $\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{\alpha}_{MFCLSd} = 1$ and thus Eq. (2.33) becomes

$$\begin{aligned}1 - \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{\alpha}_{ULS} &= \lambda_1\,\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\vec{1}\\ &+ \lambda_2\,\mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{1}^T(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd})\end{aligned} \tag{2.34}$$

Using Cramer's rule with Eq. (2.32) and Eq. (2.34),

$$\lambda_1 = \frac{\begin{vmatrix} 1 - \vec{1}^T\vec{\alpha}_{ULS} & \vec{1}^T(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \\ 1 - \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{\alpha}_{ULS} & \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \end{vmatrix}}{\begin{vmatrix} \vec{1}^T(M^TM)^{-1}\vec{1} & \vec{1}^T(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \\ \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\vec{1} & \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \end{vmatrix}} \tag{2.35}$$

and

$$\lambda_2 = \frac{\begin{vmatrix} \vec{1}^T(M^TM)^{-1}\vec{1} & 1 - \vec{1}^T\vec{\alpha}_{ULS} \\ \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\vec{1} & 1 - \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,\vec{\alpha}_{ULS} \end{vmatrix}}{\begin{vmatrix} \vec{1}^T(M^TM)^{-1}\vec{1} & \vec{1}^T(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \\ \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\vec{1} & \mathrm{sign}(\vec{\alpha}_{MFCLSd})^T\,(M^TM)^{-1}\,\mathrm{sign}(\vec{\alpha}_{MFCLSd}) \end{vmatrix}} \tag{2.36}$$

However, the analytical solution is not obtainable as given above since it requires the sign of an unknown solution as in Eq. (2.30), Eq. (2.35), and Eq. (2.36). Instead, for simplicity, we introduce an approximate solution by replacing it with $\vec{\alpha}_{SCLSd}$, but it may violate the ANC [23]. The direct MFCLS algorithm is described in Algorithm 2.1.

2.5.2 Iterative Method

As in the iterative SCLS method, we solve Eq. (2.17) with the ULS method by setting

$$\vec{r}' = \begin{bmatrix} \vec{r} \\ \lambda_1 \\ \lambda_2 \end{bmatrix}, \qquad M' = \begin{bmatrix} M \\ \lambda_1 \vec{1}^T \\ \lambda_2 \operatorname{sign}(\vec{\alpha})^T \end{bmatrix}, \qquad \vec{n}' = \begin{bmatrix} \vec{n} \\ 0 \\ 0 \end{bmatrix} \tag{2.37}$$

where λ_1 and λ_2 are the scale variables for the ASC and the AASC, respectively. Chang et al. propose $\lambda_1 = \lambda_2 = 10000$ for initial values in their experiments [20]. We observed that $h_1 = h_2 = 1$ were an available selection. The detailed discusses about the selection will be presented in Chapter 6. The iterative MFCLS algorithm can be found in Algorithm 2.2.

2.6 Algorithms

By specifying the parameter alg in Algorithm 2.1, we can estimate the abundance vector $\vec{\alpha}$ using the specified direct method. The parameter is chosen among "ULS", "NCLS", "SCLS", "FCLS", and "MFCLS".

Algorithm 2.1. Direct Least Squares Method

Input: Mixing matrix M, mixture emission spectrum $\vec{r}$ and algorithm alg

Initialize *termination* = *false*;
while *not termination* **do**
  Calculate $\vec{\alpha}_{ULS}$ using Eq. (2.3);
  if $alg = ULS$ or $alg = NCLS$ **then**
    Set $\lambda = 0$;
  else
    Calculate λ using Eq. (2.13);
  end if

Calculate $\vec{\alpha} = \vec{\alpha}_{SCLSd}$ using Eq. (2.10);
if all α_i in $\vec{\alpha} \geq 0$ **then**
Set *termination* = *true*;
else
if $alg = NCLS$ or $alg = FCLS$ **then**
Find the negative α_j with the largest $|\alpha_j|$ in $\vec{\alpha}$;
Store the position j of α_j;
Set $\alpha_j = 0$;
Remove $\vec{m}_j$ in M;
else
Set negative α_i in $\vec{\alpha}$ to zero;
Set *termination* = *true*;
end if
end if
end while
if $alg = NCLS$ or $alg = FCLS$ **then**
Restore $\vec{\alpha}$;
else if $alg = MFCLS$ **then**
Calculate λ_1 and λ_2 using Eq. (2.35) and Eq. (2.36);
Calculate $\vec{\alpha} = \vec{\alpha}_{MFCLSd}$ using Eq. (2.30);
Set negative α_i in $\vec{\alpha}$ to zero;
end if

Output: Abundance vector $\vec{\alpha}$

We can obtain the estimated abundance vector using a desired iterative method by setting the parameter alg in Algorithm 2.2. The parameter can be "SCLS", "FCLS", or "MFCLS".

Algorithm 2.2. Iterative Least Squares Method

Input: Mixing matrix M, mixture emission spectrum $\vec{r}$, scale variables λ_1 and λ_2, step sizes h_1 and h_2, error tolerance ε and algorithm alg

Initialize *termination* = *false*;

2 **while** *not termination* **do**
3 **if** $alg = SCLS$ or $alg = FCLS$ **then**
4 Set $\lambda = \lambda_1,\ h = h_1$;
5 Define M' and $\vec{r}'$ using Eq. (2.16);
6 Calculate $\vec{\alpha} = \vec{\alpha}_{SCLSi}$ using Eq. (2.19);
7 **else**
8 Calculate $\vec{\alpha}_{ULS}$ using Eq. (2.3);
9 Define M' and $\vec{r}'$ using Eq. (2.37);
10 Calculate $\vec{\alpha} = \vec{\alpha}_{MFCLSi}$ using Eq. (2.19);
11 **end if**
12 **if** all α_i in $\vec{\alpha} \geq 0$ **then**
13 **if** $alg = SCLS$ or $alg = FCLS$ **then**
14 **if** $\left|\vec{1}^T \vec{\alpha} - 1\right| < \varepsilon$ **then**
15 Set *termination* = *true*;
16 **else**
17 Set $\lambda = \lambda + h$;
18 **end if**
19 **else**
20 **if** $\left|\vec{1}^T \vec{\alpha} - 1\right| < \varepsilon$ and $\text{sign}(\vec{\alpha})^T \vec{\alpha} - 1 < \varepsilon$ **then**
21 Set *termination* = *true*;
22 **else**
23 **if** $|\vec{\alpha} - 1| \geq \varepsilon$ **then**
24 Set $\lambda_1 = \lambda_1 + h_1$;
25 **end if**
26 **if** $\text{sign}(\vec{\alpha})^T \vec{\alpha} - 1 \geq \varepsilon$ **then**
27 Set $\lambda_2 = \lambda_2 + h_2$;
28 **end if**
29 **end if**
30 **end if**
31 **else**
32 Find the negative α_j with the largest $\left|\alpha_j\right|$ in $\vec{\alpha}$;
33 Store the position j of α_j;
34 Set $\alpha_j = 0$;

Remove $\vec{m}_j$ in M;
end if
end while
if $alg = SCLS$ **then**
Set negative α_i in $\vec{\alpha}$ to zero;
Else
Restore $\vec{\alpha}$;
end if

Output: Abundance vector $\vec{\alpha}$

2.7 Conclusion

In this chapter, we have studied various linear unmixing algorithms. The ULS method solves an unconstrained least square linear unmixing problem to estimate abundance fractions, which results in a suboptimal solution since it may contain negative values which are meaningless in terms of abundance and may fail to sum to one. To produce more meaningful abundance fractions, modified ULS methods, the SCLS method and the NCLS method, have been developed by applying the ASC and the ANC to the unconstrained linear unmixing problem, respectively. It has been observed that the SCLS and NCLS methods may yield suboptimal solutions since the ASC and the ANC are not implemented simultaneously. By applying both constraints, the FCLS method can produce a more optimal solution. Thus far, the ANC has been implemented by performing dimensionality reduction on the mixing matrix, which prevents us from using the Lagrange multiplier method to obtain analytical solutions. Replacing the ANC with the AASC enables the Lagrange multiplier method and hence we can derive an algorithm, the MFCLS method, to produce an optimal solution. However, the method may result in a suboptimal solution because the SCLS solution is employed rather than the MFCLS solution to compute the analytical solution.

The iterative linear unmixing methods and the methods equipped with the dimensionality reduction process must generate matrices repeatedly in their procedures to estimate abundance fractions, increasing the computational complexity of the algorithms. To resolve this issue, numerous optimization techniques can be candidates for linear unmixing; particularly, the gradient descent method is applicable to solve the fully constrained linear unmixing problem. We will discuss this method in the next chapter.

Chapter 3

This chapter describes the gradient descent methods used to estimate abundance fractions for the linear mixture model (1.4). We will first show that applying the ASC and ANC to the gradient descent algorithm leads to the so-called FC-GD method. Also, we will demonstrate that, when the linear mixture model has a unique local solution, the original gradient descent algorithm can be employed by implementing a bounding process into the algorithm for optimal abundance fraction estimation.

3.1 Gradient Descent (GD) Method

The GD method is a local optimization method, which finds a local solution of an objective function using the gradient [38]. Even though it is not a global optimization method, it is employed in many situations due to its simplicity in implementation and robustness to converge to a local optimum. When the objective function is a convex function defined on a convex set, any local optimum automatically becomes the global optimum [39] and thereby, in this case, the GD method works as a global optimization method.

If an initial point $\vec{x}^{(0)}$ is selected in the neighbourhood of a local minimum, the method moves in consecutive points from $\vec{x}^{(k)}$ to $\vec{x}^{(k+1)}$ in the direction of the downhill gradient. Thus, it moves along the line extended from $\vec{x}^{(k)}$ in the direction opposite to the gradient $-\nabla f\left(\vec{x}^{(k)}\right)$. Since the search commences at a point $\vec{x}^{(0)}$ and then slides down the gradient, the iterative scheme of the GD method is described as

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - s^{(k)}\nabla f\left(\vec{x}^{(k)}\right) \tag{3.1}$$

where $s^{(k)}$ is a (positive) step size for the k^{th} iteration [40].

Now the step size $s \geq 0$ should be found such that $\vec{x}^{(k)} - s\nabla f\left(\vec{x}^{(k)}\right)$ improves $\vec{x}^{(k)}$. Then, if we define a function

$$\phi(s) = f\left(\vec{x}^{(k)} - s\nabla f\left(\vec{x}^{(k)}\right)\right), \tag{3.2}$$

it can be expanded at $s = 0$ using the Taylor expansion as

$$\phi(s) = \phi(0) + \phi'(0)s + O(s^2). \tag{3.3}$$

Since

$$\phi'(s) = -\nabla f\left(\vec{x}^{(k)} - s\nabla f\left(\vec{x}^{(k)}\right)\right) \cdot \nabla f\left(\vec{x}^{(k)}\right), \tag{3.4}$$

Eq. (3.3) becomes

$$f\left(\vec{x}^{(k)} - s\nabla f\left(\vec{x}^{(k)}\right)\right) = f\left(\vec{x}^{(k)}\right) - s\left\|\nabla f\left(\vec{x}^{(k)}\right)\right\|^2 + O(s^2). \tag{3.5}$$

If the value of s is sufficiently small, then it satisfies the following inequality,

$$f\left(\vec{x}^{(k)} - s\nabla f\left(\vec{x}^{(k)}\right)\right) \leq f\left(\vec{x}^{(k)}\right) \tag{3.6}$$

which leads to

$$f\left(\vec{x}^{(k+1)}\right) \leq f\left(\vec{x}^{(k)}\right). \tag{3.7}$$

The GD method is therefore locally convergent, which generates a sequence of $\vec{x}^{(k)}$ towards a minimum $\vec{x}^*$ if a starting point $\vec{x}^{(0)}$ is sufficiently close to $\vec{x}^*$. The appropriate selection of the initial point is significant to locate a desired minimum. The scheme terminates if there is no longer improvement in $\vec{x}^{(k)}$; that is, it stops when $\vec{x}^{(k)}$ becomes a stationary point of f with $\nabla f\left(\vec{x}^{(k)}\right) = \vec{0}$.

3.2 Fully Constrained Gradient Descent (FC-GD) Method

We can solve the linear unmixing problem by locating a minimizer of the least squares error function $J(\alpha)$ in Eq. (2.1). However, the gradient descent algorithm may find multiple local solutions according to its initial point since it is a local optimization method, and, because no constraints are considered, they may be suboptimal solutions. Chen et al. and Theys et al. [24], [25] suggested a gradient descent algorithm considering the ASC and ANC, which is called the FC-GD method; it still may locate multiple local solutions, but the solutions must satisfy the ASC and ANC and hence they are more accurate than those obtained from the original GD method.

To implement the ANC for the GD method [25], we need to minimize $J(\vec{\alpha})$ under inequality constraints. Introducing a Lagrange function L associated with the error function under nonnegativity constraints,

$$L(\vec{\alpha}, \vec{\lambda}) = J(\vec{\alpha}) - \vec{\lambda}^T \vec{g}(\vec{\alpha}) \tag{3.8}$$

where $\vec{\lambda} = [\lambda_1, \ \lambda_2, \ \ldots, \ \lambda_l]^T$ contains the Lagrange multiplies as elements and $\vec{g}(\vec{\alpha}) = [g(\alpha_1), \ g(\alpha_2), \ \ldots, \ g(\alpha_l)]^T$ with the function g to express the nonnegativity constraints for abundance fractions such that g is an increasing function that must be positive for inactive constraints $\alpha_i > 0$ and zero for active constraints $\alpha_i = 0$ [25]. The KKT conditions at the optimum $\vec{\alpha}^*$ and $\vec{\lambda}^*$ are described as

$$\nabla L(\vec{\alpha}^*, \vec{\lambda}^*) = \vec{0}; \tag{3.9}$$

$$g(\alpha_i^*) \geq 0 \ \ \forall i; \tag{3.10}$$

$$\lambda_i^* \geq 0 \ \ \forall i; \tag{3.11}$$

$$\lambda_i^* g(\alpha_i^*) = 0 \ \ \forall i. \tag{3.12}$$

Eq. (3.9) becomes

$$\nabla L(\vec{\alpha}^*, \vec{\lambda}^*) = \nabla\left[J(\vec{\alpha}^*) - \vec{\lambda}^{*T}\vec{g}(\vec{\alpha}^*)\right] = \nabla J(\vec{\alpha}^*) - \vec{\lambda}^{*T}\nabla\vec{g}(\vec{\alpha}^*) = 0 \tag{3.13}$$

where $\nabla\vec{g}(\vec{\alpha}) = [\nabla g(\alpha_1), \quad \nabla g(\alpha_2), \quad \ldots, \quad \nabla g(\alpha_l)]^T$. Since $\nabla J(\vec{\alpha}^*) = \vec{\lambda}^{*T}\nabla\vec{g}(\vec{\alpha}^*)$, we take the i^{th} element of each vector. This gives

$$\lambda_i^* = \frac{[\nabla J(\vec{\alpha}^*)]_i}{\nabla g(\alpha_i^*)}. \tag{3.14}$$

Multiplying Eq. (3.14) by $g(\alpha_i^*)$ and using Eq. (3.12) results in

$$\lambda_i^* g(\alpha_i^*) = \frac{[\nabla J(\vec{\alpha}^*)]_i g(\alpha_i^*)}{\nabla g(\alpha_i^*)} = 0 \tag{3.15}$$

or equivalently,

$$-[\nabla J(\vec{\alpha}^*)]_i g(\alpha_i^*) = 0. \tag{3.16}$$

By taking $g(\alpha) = \alpha$ for the nonnegativity constraints,

$$-[\nabla J(\vec{\alpha}^*)]_i \alpha_i^* = 0. \tag{3.17}$$

Considering an equation of the form $f(x) = 0$ as $x = x + f(x)$, we can apply the fixed-point iteration method [25] to find x iteratively, which gives $x^{(n+1)} = x^{(n)} + f\left(x^{(n)}\right)$. Hence, taking $f(\alpha_i^*) = -s[\nabla J(\vec{\alpha}^*)]_i \alpha_i^* = 0$ for some real number s yields the following component-wise scheme for the GD method considering the ANC

$$\alpha_i^{(k+1)} = \alpha_i^{(k)} - s^{(k)}\left[\nabla J\left(\vec{\alpha}^{(k)}\right)\right]_i \alpha_i^{(k)} \tag{3.18}$$

where $s^{(k)}$ is a step size that must be adjusted for convergence of the algorithm. (Refer to Section 3.4.)

In order to impose the ASC on the GD method [24], the following variable change is performed to ensure that the ASC is satisfied. Thus, set $w_i \geq 0$ for all i, and define $\vec{\alpha}$ as

$$\alpha_i = \frac{w_i}{\sum_{m=1}^{l} w_m}. \tag{3.19}$$

Indeed,

$$\sum_{i=1}^{l} \alpha_i = \sum_{i=1}^{l} \frac{w_i}{\sum_{m=1}^{l} w_m} = \frac{\sum_{i=1}^{l} w_i}{\sum_{m=1}^{l} w_m} = 1. \tag{3.20}$$

The partial derivative of the error function J with respect to the new variables w_i becomes

$$\frac{\partial J}{\partial w_i} = \sum_{j=1}^{l} \frac{\partial J}{\partial \alpha_j} \left(\frac{\partial \alpha_j}{\partial w_i} \right) \tag{3.21}$$

which, by the quotient rule yields,

$$\frac{\partial \alpha_j}{\partial w_i} = \frac{\frac{\partial w_j}{\partial w_i} \sum_{m=1}^{l} w_m - \frac{\partial \left(\sum_{m=1}^{l} w_m \right)}{\partial w_i} w_j}{\left(\sum_{m=1}^{l} w_m \right)^2}. \tag{3.22}$$

Therefore,

$$\begin{aligned} \frac{\partial J}{\partial w_i} &= \sum_{j=1}^{l} \frac{\partial J}{\partial \alpha_j} \left[\left(\frac{\frac{\partial w_j}{\partial w_i} \sum_{m=1}^{l} w_m}{\left(\sum_{m=1}^{l} w_m \right)^2} \right) - \left(\frac{\frac{\partial \left(\sum_{m=1}^{l} w_m \right)}{\partial w_i} w_j}{\left(\sum_{m=1}^{l} w_m \right)^2} \right) \right] \\ &= \sum_{j=1}^{l} \frac{\partial J}{\partial \alpha_j} \left[\left(\frac{\frac{\partial w_j}{\partial w_i}}{\sum_{m=1}^{l} w_m} \right) - \left(\frac{w_j}{\left(\sum_{m=1}^{l} w_m \right)^2} \right) \right] \\ &= \frac{1}{\sum_{m=1}^{l} w_m} \left[\frac{\partial J}{\partial \alpha_i} - \sum_{j=1}^{l} \alpha_j \left(\frac{\partial J}{\partial \alpha_j} \right) \right]. \end{aligned} \tag{3.23}$$

Eq. (3.18) formulates the component-wise update equation as

$$w_i^{(k+1)} = w_i^{(k)} - s^{(k)} \frac{w_i^{(k)}}{\sum_{m=1}^{l} w_m^{(k)}} \left[\frac{\partial J}{\partial \alpha_i^{(k)}} - \sum_{j=1}^{l} \alpha_j^{(k)} \left(\frac{\partial J}{\partial \alpha_j^{(k)}} \right) \right]. \tag{3.24}$$

Since $\sum_{m=1}^{l} w_m^{(k+1)} = \sum_{m=1}^{l} w_m^{(k)}$ for all k, $\sum_{m=1}^{l} w_m^{(k)}$ is constant and thus it can be absorbed into the step size $s^{(k)}$, which gives

$$w_i^{(k+1)} = w_i^{(k)} + s^{(k)} w_i^{(k)} \left[\frac{\partial J}{\partial \alpha_i^{(k)}} - \sum_{j=1}^{l} \alpha_j^{(k)} \left(\frac{\partial J}{\partial \alpha_j^{(k)}} \right) \right]. \tag{3.25}$$

Notice that it solves the unconstrained problem with respect to the ASC [24]. To impose the ASC on Eq. (3.25), we necessarily divide it by $\sum_{m=1}^{l} w_m^{(k+1)} = \sum_{m=1}^{l} w_m^{(k)}$. Thus,

$$\frac{w_i^{(k+1)}}{\sum_{m=1}^{l} w_m^{(k+1)}} = \frac{w_i^{(k)}}{\sum_{m=1}^{l} w_m^{(k)}} + s^{(k)} \frac{w_i^{(k)}}{\sum_{m=1}^{l} w_m^{(k)}} \left[\frac{\partial J}{\partial \alpha_i^{(k)}} - \sum_{j=1}^{l} \alpha_j^{(k)} \left(\frac{\partial J}{\partial \alpha_j^{(k)}} \right) \right] \tag{3.26}$$

and then using Eq. (3.19) produces

$$\alpha_i^{(k+1)} = \alpha_i^{(k)} + s^{(k)} \alpha_i^{(k)} \left[\frac{\partial J}{\partial \alpha_i^{(k)}} - \sum_{j=1}^{l} \alpha_j^{(k)} \left(\frac{\partial J}{\partial \alpha_j^{(k)}} \right) \right]. \tag{3.27}$$

Since $\sum_{m=1}^{l} \alpha_m^{(k+1)} = \sum_{m=1}^{l} \alpha_m^{(k)}$ for all k, the initial point $\vec{\alpha}$ must be selected such that $\sum_{m=1}^{l} \alpha_m^{(0)} = 1$ so that the algorithm satisfies the ASC. Considering the component-wise update equation (3.27) results in the following scheme of the FC-GD method

$$\vec{\alpha}^{(k+1)} = \vec{\alpha}^{(k)} + s^{(k)} \mathrm{diag}\left(\vec{\alpha}^{(k)}\right) \left[\nabla J\left(\vec{\alpha}^{(k)}\right) - \vec{1} \nabla J\left(\vec{\alpha}^{(k)}\right)^T \vec{\alpha}^{(k)} \right] \tag{3.28}$$

where $\vec{1}$ is the all-one vector and $\mathrm{diag}(\cdot)$ is a diagonal matrix. A proper choice of the step size is required for convergence. This will be discussed in Section 3.4. The algorithm of the FC-GD method is explained in Algorithm 3.1.

3.3 Special Case for Application of GD method

The FC-GD method minimizes the least squares error function $J(\vec{\alpha})$ to find an optimal solution satisfying both the ASC and the ANC simultaneously using the KKT conditions, which increases the complexity of the GD algorithm [23]. However, if there exists a unique local minimum

in $J(\vec{\alpha})$ on the domain $[0,1]^n$, we can use the original GD scheme instead of the FC-GD method because it must locate the same solution as the FC-GD method.

In fact, the linear unmixing problem has a unique local solution if (i) the linear mixture model is an overdetermined system and (ii) the columns of the mixing matrix for the model are linearly independent. Considering $J: \mathbb{R}^n \to \mathbb{R}$ where $J(\vec{\alpha}) = \frac{1}{2}(\vec{r} - M\vec{\alpha})^T(\vec{r} - M\vec{\alpha})$, we first show that $J(\vec{\alpha})$ is a quadratic function using Definition 3.1.

Definition 3.1. A quadratic function is a function $f: \mathbb{R}^n \to \mathbb{R}$ of form

$$f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} + \vec{b}^T\vec{x} + c,$$

where Q is an $n \times n$ square matrix, $\vec{b}$ is $n \times 1$ column vector and c is a real number [41].

Proposition 3.1. The least squares error function $J(\vec{\alpha})$ is quadratic.

Proof. The least squares error function is

$$J(\vec{\alpha}) = \frac{1}{2}(\vec{r} - M\vec{\alpha})^T(\vec{r} - M\vec{\alpha})$$

which results in

$$J(\vec{\alpha}) = \frac{1}{2}[\vec{r}^T\vec{r} - \vec{r}^T M\vec{a} - (M\vec{a})^T\vec{r} + (M\vec{a})^T(M\vec{a})].$$

Since $(M\vec{a})^T\vec{r} = [(M\vec{a})^T\vec{r}]^T = \vec{r}^T M\vec{a}$ and $(M\vec{a})^T(M\vec{a}) = \vec{a}^T M^T M\vec{a}$, it follows that

$$J(\vec{\alpha}) = \frac{1}{2}(\vec{r}^T\vec{r} - 2\vec{r}^T M\vec{a} + \vec{a}^T M^T M\vec{a})$$

$$= \frac{1}{2}\vec{a}^T M^T M\vec{a} - \vec{r}^T M\vec{a} + \frac{1}{2}\vec{r}^T\vec{r}.$$

By taking $Q = M^T M$, $\vec{b}^T = -\vec{r}^T M$, and $c = \frac{1}{2}\vec{r}^T\vec{r}$, $J(\vec{\alpha})$ becomes

$$J(\vec{\alpha}) = \frac{1}{2}\vec{\alpha}^T Q\vec{\alpha} + \vec{b}^T\vec{\alpha} + c.$$

Hence, $J(\vec{\alpha})$ is a quadratic function. Notice that

$$Q^T = (M^T M)^T = M^T M = Q,$$

which implies that Q is a symmetric matrix. ■

Proposition 3.1 implies that $J(\vec{\alpha})$ is a twice-differentiable function, and its respective gradient and Hessian matrix are obtained by

$$\nabla J(\vec{\alpha}) = Q\vec{\alpha} + \vec{b} \tag{3.29}$$

and

$$\nabla^2 J(\vec{\alpha}) = Q. \tag{3.30}$$

Now, we assume that the mixing matrix M forms an overdetermined system and the columns of M are linearly independent, enabling us to prove that the Hessian matrix must be positive definite using Definition 3.2.

Definition 3.2. An $n \times n$ real symmetric matrix A is said to be positive semi-definite if and only if $\vec{x}^T A\vec{x} \geq 0$ for all $\vec{x} \in \mathbb{R}^n$. It is said to be positive definite if and only if $\vec{x}^T A\vec{x} > 0$ for all $\vec{x} \in \mathbb{R}^n \setminus \{\vec{0}\}$ [42].

Proposition 3.2. Suppose $A \in M_{mn}(\mathbb{R})$ where $m > n$. Then $A^T A$ is positive definite if the columns of A are linearly independent.

Proof. Let $A \in M_{mn}(\mathbb{R})$ with $m > n$. Suppose that the columns of A are linearly independent.

Then $\text{rank}(A) = n$ and thus, by the rank-nullity theorem, $\text{nullity}(A) = 0$. This implies $\text{null}(A) = \{\vec{0}\}$. Note that

$$\vec{x}^T A^T A\vec{x} = (A\vec{x})^T(A\vec{x}) = (A\vec{x}) \cdot (A\vec{x}) = \|A\vec{x}\|^2 \geq 0$$

for all $\vec{x} \in \mathbb{R}^n$. Namely, $A^T A$ is positive semi-definite. Since $A^T A$ is positive definite if and only if $\vec{x}^T A^T A\vec{x} > 0$ for all $\vec{x} \in \mathbb{R}^n \backslash \{\vec{0}\}$, it suffices to show that $\vec{x} = \vec{0}$ if and only if $\vec{x}^T A^T A\vec{x} = 0$. Suppose that $\vec{x} = \vec{0}$. Then it is obvious that $\vec{x}^T A^T A\vec{x} = 0$. Conversely, suppose that $\vec{x}^T A^T A\vec{x} = 0$. Then $\|A\vec{x}\|^2 = 0$ and thus $A\vec{x} = \vec{0}$. But $A\vec{x} = \vec{0}$ implies $\vec{x} = \vec{0}$ by the rank-nullity theorem as shown above. Therefore, $A^T A$ is positive definite if the columns of A are linearly independent.

■

From the following theorem [43], we can see that the least squares error function $J: \mathbb{R}^n \to \mathbb{R}$ is strictly convex (where $\mathbb{R}^n$ is a convex set) because the Hessian matrix of $J(\vec{\alpha})$ is positive definite.

Theorem 3.1. Let Ω be a convex set. A twice-differentiable function $f: \Omega \to \mathbb{R}$ is strictly convex if and only if for every $x \in \Omega$, the Hessian matrix $\nabla^2 f(x)$ is positive definite.

We can see that $J: \mathbb{R}^n \to \mathbb{R}$ is strictly convex on $\mathbb{R}^n$. Now we want to show that $J: [0,1]^n \to \mathbb{R}$ is also strictly convex on the convex subset $[0,1]^n$ of $\mathbb{R}^n$. Before we proceed, we need to define a convex set as in [39] and verify that $[0,1]^n$ is such a set.

Definition 3.2. Let V be a vector space. A subset $\Omega \subseteq V$ is said to be convex if for every $a, b \in \Omega$ and $t \in [0,1]$, the convex combination $ta + (1-t)b \in \Omega$.

Proposition 3.3. $[0,1]^n$ is a convex set.

Proof. Note that $[0,1]^n \subseteq \mathbb{R}^n$ and $\mathbb{R}^n$ is a vector space. Suppose that $\vec{a}, \vec{b} \in [0,1]^n$. Then we can express them as $\vec{a} = (a_1, a_2, \dots, a_n)$ and $\vec{b} = (b_1, b_2, \dots, b_n)$ where $\max\{a_i, b_i\} \leq 1$ and $\min\{a_i, b_i\} \geq 0$ with $1 \leq i \leq n$. Without loss of generality, let $a_i \geq b_i$. Then $0 \leq b_i \leq a_i \leq 1$, we have

$$0 \leq ta_i + (1-t)b_i \leq a_i \leq 1$$

and thus

$$\vec{a} + (1-t)\vec{b} \in [0,1]^n$$

for all $t \in [0,1]$. Therefore, $[0,1]^n$ is a convex set. ■

Definition 3.3. Let Ω be a convex set. A function $f: \Omega \to \mathbb{R}$ is convex on Ω if

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$$

for every $x, y \in \Omega$ and $t \in (0,1)$. If the inequality holds strictly, then the function f is called strictly convex.

With the definition of convexity of a function [39], we can see that $J: [0,1]^n \to \mathbb{R}$ is strictly convex on $[0,1]^n$ as $J: \mathbb{R}^n \to \mathbb{R}$ is strictly convex on $\mathbb{R}^n$ and $[0,1]^n$ is a convex subset of $\mathbb{R}^n$. Therefore, the following proposition illustrates that it must contain at most one local minimizer on the given domain.

Proposition 3.5. If a function $f: \Omega \to \mathbb{R}$ is strictly convex where Ω is a convex set, then it has at

most one local minimizer on Ω.

Proof. Suppose that f is strictly convex but assume by contraction that it has at least two distinct local minimizers on Ω, say x_1 and x_2 with $f(x_1) \leq f(x_2)$ where $x_1 \neq x_2$. By the definition of strict convexity,

$$f(tx_1 + (1-t)x_2) < tf(x_1) + (1-t)f(x_2)$$

for all $t \in (0,1)$. Since $t > 0$, we have $tf(x_1) \leq tf(x_2)$ and thus

$$tf(x_1) + (1-t)f(x_2) \leq tf(x_2) + (1-t)f(x_2)$$

which becomes

$$tf(x_1) + (1-t)f(x_2) \leq f(x_2).$$

Applying this to the definition of strict convexity,

$$f(tx_1 + (1-t)x_2) < f(x_2).$$

If t is selected to be sufficiently close to 0, say $t < \varepsilon$, then $tx_1 + (1-t)x_2 \in B(x_2, \varepsilon)$, which contradicts the definition of the local minimizer x_2. Therefore, the assumption is false and thus the function must have at most one local minimizer. ■

However, according to the extreme value theorem [44], there must exist at least one local minimizer on the domain as well; $[0,1]^n$ is a closed and bounded subset of an open set $\mathbb{R}^n$ and, by Theorem 3.3, the strictly convex function J is continuous everywhere [45].

Theorem 3.2. (Extreme Value Theorem) Suppose that $f: \mathbb{R}^n \to \mathbb{R}$ is continuous on an open set U. If S is a closed and bounded subset of U, then f has a global minimum and global maximum on S.

Theorem 3.3. If $f: \mathbb{R}^n \to \mathbb{R}$ is a convex function, then f is continuous on $\mathbb{R}^n$.

We finally demonstrated that the least squares error function $J(\vec{\alpha})$ contains a unique local solution on $[0,1]^n$, which becomes a unique global solution of the function on the specified domain under the two conditions: (i) the linear mixture model is an overdetermined system and (ii) the columns of the mixing matrix are linearly independent. The original GD method therefore can be applied as a global optimization method to minimize $J(\vec{\alpha})$ defined on the domain.

To minimize $J(\vec{\alpha})$ on $[0,1]^n$ with the GD method, it is required to bound the abundance vector $\vec{\alpha}$ to the domain in each iteration so that it remains and finds a solution there; otherwise, the GD method solves unconstrained optimization problems. Thus, we need to implement a process pushing a point back iteratively to the domain whenever the point escapes from the domain. This process is called a bounding process and it indeed enables us to take any initial point on $[0,1]^n$. The algorithm of the original GD method for linear unmixing is shown in Algorithm 3.1.

3.4 Selection of Step Size

As seen in Eq. (3.1) and Eq. (3.28), the iterative scheme requires a step size. In fact, there are several options to choose a step size for the GDM. The simplest option is merely to take a fixed step size $s^{(k)} = s$ for all k. The proper selection of the fixed step size is however not simple; if s is too large, the method may overshoot the minima and diverge and, if s is too small, the method may converge very slowly.

If an objective function is a quadratic function, we can easily find an appropriate fixed step

size for the GD method [43]. Given a fixed step size $s > 0$, the gradient method is guaranteed to converge if and only if

$$0 < s < \frac{2}{\lambda_{max}(Q)} \tag{3.31}$$

where $\lambda_{max}(Q)$ is the largest eigenvalue of the matrix Q. Since the least squares error function $J(\alpha)$ is quadratic, we can find a proper step size by computing the largest eigenvalue of $M^T M$ where M is the mixing matrix.

3.5 Algorithms

We can use either the FC-GD method or the typical GD method to estimate the abundance vector $\vec{\alpha}$ by specifying the parameter alg in Algorithm 3.1. The parameter is selected between "FCGDM" and "GDM".

Algorithm 3.1. Gradient Descent Method for Linear Unmixing

Input: Mixing matrix M, mixture emission spectrum $\vec{r}$, initial point $\vec{\alpha}^{(0)}$, step size s, maximum number of iterations $maxIter$, convergence tolerance ε and algorithm alg

Initialize $k = 0$ and $convergence = false$;
Calculate $Q = M^T M$ and $\vec{b} = -M^T \vec{r}$;
while *not convergence* **do**
  Calculate $\nabla J(\vec{\alpha}^{(k)})$ using Eq. (3.29);
  if $\|\nabla J(\vec{\alpha}^{(k)})\| < \varepsilon$ or $k = maxIter$ **then**
    Set $convergence = true$;
  Else
    if $alg = GDM$ **then**
      Calculate $\vec{\alpha}^{(k+1)}$ using Eq. (3.1);
      Bound $\vec{\alpha}^{(k+1)}$ with lower bound 0 and upper bound 1;
    Else

Calculate $\vec{\alpha}^{(k+1)}$ using Eq. (3.28);
end if
Set $k = k + 1$;
end if
end while
Output: Abundance vector $\vec{\alpha}$

3.6 Conclusion

In general, a GD algorithm finds multiple local optima, which may produce suboptimal solutions for linear unmixing. By applying the ASC and ANC to the updated scheme of the GD method, the FC-GD method can locate an optimal solution. However, if the linear mixture model is an overdetermined system and the columns of the mixing matrix of the model are linearly independent, it is guaranteed that there exists only one local solution on the domain $[0, 1]^n$ and therefore the original GD method with the bounding process finds the same optimal solution as the FD-GD method. Due to the bounding process, we can select any initial point on the domain for the original GD method, whereas we should choose an initial point meeting the ASC for the FC-GD method. Furthermore, the simplicity of the original GD method leads to less computational cost, compared to the FC-GD method.

Considering the ASC and ANC, we know that an optimal solution for a linear unmixing problem must exist on the domain $[0, 1]^n$. Then a natural question arises: is a global solution on the domain an optimal solution for the linear unmixing problem? It is undoubtedly true. We will discuss about two global optimization methods applicable to linear unmixing in the next chapter.

Chapter 4

The purpose of this chapter is to represent global optimization methods to estimate abundance fractions for the linear spectral mixture model. To obtain an optimal solution with the GD method, we necessarily modify the updating equation in Eq. (3.1) by applying the ASC and ANC. In this chapter, two global optimization methods are introduced to locate an optimal solution without considering such constraints.

4.1 Standard Nelder-Mead (NM) Method

The NM method has been developed by Nelder and Mead to solve unconstrained optimization problems without any derivative information, which makes it suitable for optimization of non-smooth and even discontinuous functions [46]. Strictly speaking, the NM method is not a strong global optimization method; however, in practice it performs reasonably well for objective functions with weak multimodality [29], [47].

This method is characterized by the use of a simplex which is a geometric figure in n dimensions. A simplex is defined as the convex hull of $n+1$ vertices. The NM method iteratively produces a sequence of simplices to approximate an optimum; it improves a simplex by comparing the objective function values at the $n+1$ vertices and moves the simplex towards an optimum.

In each iteration, the vertex with the worst function value is eliminated and then replaced with another point with a better value. The new point is found by reflecting, expanding, or contracting the simplex along the line joining the worst vertex with the centroid of the other vertices. In the case

when the algorithm fails to find a better point, it maintains only the vertex with the best function value and shrinks the simplex by moving the remaining vertices towards this vertex. In this manner, a new simplex is iteratively formed, and the search is continued. As the iterations proceed, the function values at the vertices of the simplex get smaller and smaller, and hence its size diminishes and the optimum point is obtained at the end [48].

As stated above, reflection, expansion, contraction, and shrinkage are four possible operations in the algorithm. Each operation is associated with a parameter: α (reflection), β (expansion), γ (contraction), and δ (shrinking) where $\alpha > 0$, $\beta > 1$, $0 < \gamma < 1$, and $0 < \delta < 1$. For the standard NM method, the parameters are selected to be

$$\{\alpha, \beta, \gamma, \delta\} = \left\{1, 2, \frac{1}{2}, \frac{1}{2}\right\}. \tag{4.1}$$

The operations of the algorithm for minimization are described as below [48].

- **Ordering.** At each iteration, the vertices $\{\vec{x}_i\}_{i=0}^{n}$ of the simplex are ordered based on the objective function values

$$f(\vec{x}_0) \leq f(\vec{x}_1) \leq \cdots \leq f(\vec{x}_n) \tag{4.2}$$

  where $\vec{x}_0$ is the best vertex (where the function value is the smallest) and $\vec{x}_n$ is the worst vertex (where the function value is the largest).

- **Centroid.** The procedure uses the centroid $\vec{x}_m$ of the simplex. It is obtained as

$$\vec{x}_m = \frac{1}{n}\sum_{i=0}^{n-1} \vec{x}_i \tag{4.3}$$

- **Reflection.** This operation reflects the highest-valued point over the centroid. This typically moves the simplex from high regions toward lower regions. The reflection point $\vec{x}_r$ is given by

$$\vec{x}_r = \vec{x}_m + \alpha(\vec{x}_m - \vec{x}_n). \tag{4.4}$$

- **Expansion.** When the reflected point has a function value less than all points in the simplex, the reflected point is sent even further by this operation. The expansion point $\vec{x}_e$ is defined as

$$\vec{x}_e = \vec{x}_r + \beta(\vec{x}_r - \vec{x}_m). \tag{4.5}$$

- **Contraction.** The simplex is shrunk down by moving away from the worst point. The contraction point $\vec{x}_c$ is thus given by

$$\vec{x}_c = \vec{x}_m + \gamma(\vec{x}_n - \vec{x}_m). \tag{4.6}$$

- **Shrinkage.** All points are moved toward the best point, typically halving the separation distance. For $1 \leq i \leq n$, $\vec{x}_i$ is given by

$$\vec{x}_i = \vec{x}_0 + \delta(\vec{x}_i - \vec{x}_0). \tag{4.7}$$

The algorithm terminates when the simplex becomes sufficiently small, and the vertices are within a specific tolerance [28]. Namely,

$$\max_{1\leq i\leq n} |f(\vec{x}_i) - f(\vec{x}_0)| < \varepsilon \text{ and } \max_{1\leq i\leq n} \|\vec{x}_i - \vec{x}_0\|_\infty < \varepsilon \tag{4.8}$$

In practice, the standard deviation of the function values at the vertices is often employed as a termination condition for the algorithm [48].

$$\Delta = \sqrt{\frac{1}{n+1}\sum_{i=0}^{n}[f(\vec{x}_i) - f_m]^2} \tag{4.9}$$

where

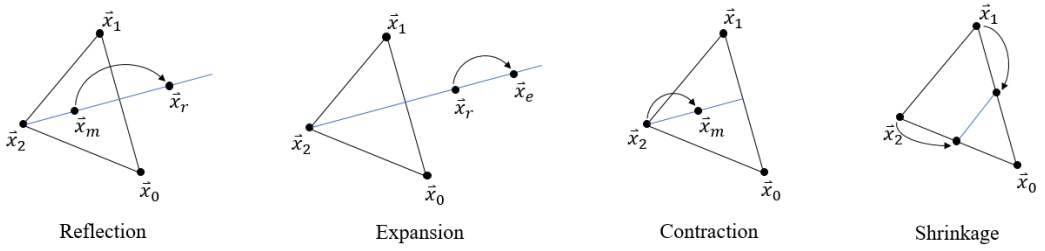

Figure 4.1. The Nelder-Mead simplex operations visualized in two-dimensions.

$$f_m = \frac{1}{n+1}\sum_{i=0}^{n} f(\vec{x}_i) \tag{4.10}$$

The final optimizer $\vec{x}_{best}$ is the point whose function value is the smallest among the vertices of the simplex. The algorithm of the standard NM method is summarized in Algorithm 4.1.

4.2 Adaptive Nelder-Mead (NM) Method

It is well-known that the standard NM algorithm is very sensitive to dimensionality of an objective function; in fact, it has been observed by researchers that the standard NM can become very inefficient for large dimensional problems, which is called the effect of dimensionality [27], [29]. Although the standard NM method generally performs well for low-dimensional problems and continuously remains as one of the most popular optimization techniques, it may show poor performance in high dimensions. Unfortunately, it is still an open question whether the simplices converge to a minimizer due to the lack of a satisfactory theoretical analysis for explaining the effect of dimensionality on the NM method [49].

It is however found that reducing the chances of using reflection steps and avoiding the rapid reduction in the simplex diameter should help improve the performance of the standard NM for large dimensional problems. Based on these observations, it is proposed to adaptively select the parameters for expansion, contraction, and shrinkage according to dimensionality of an objective function [28]. Specifically, for the dimension n,

$$\{\alpha, \beta, \gamma, \delta\} = \left\{1, 1+\frac{2}{n}, 0.75-\frac{1}{2n}, 1-\frac{1}{n}\right\}. \tag{4.11}$$

When $n = 2$, the adaptive NM method becomes the standard NM method discussed in the previous

section.

In high dimensions, expansion operations may distort the simplex badly, but the choice of β in (4.11) is helpful to prevent it from the bad distortion caused by expansion. The use of γ in (4.11) can avoid its diameter reduction when n is large. Similarly, taking δ in (4.11) can prevent the simplex diameter from drastic reduction, which enables the following expansion or contraction operations to contribute to minimize the objective function more rapidly. It is also observed that the use of the adaptive parameters (4.11) rather than (4.1) can help reduce the number of reflection operations for optimization.

4.3 Selection of Initial Simplex

The proper choice of an initial simplex influences the performance of the NM method significantly [50]. Gao and Han [28] suggested how to choose the initial simplex vertices for better convergence. For a search in n dimensions, by selecting a starting point $\vec{x}_0$ as one of the vertices, the remaining n vertices are obtained by the following rule

$$\vec{x}_i = \vec{x}_0 + \tau_i \vec{e}_i \tag{4.12}$$

where $1 \leq i \leq n$, and $\vec{e}_i$ denotes the i^{th} standard basis vector with a 1 in the i^{th} coordinate and 0 elsewhere. The value τ_i is chosen as

$$\tau_i = \begin{cases} 0.05 & \text{if } (\vec{x}_1)_i \neq 0 \\ 0.00025 & \text{if } (\vec{x}_1)_i = 0 \end{cases}. \tag{4.13}$$

4.4 Teaching-Learning-Based Optimization

For a large-scale problem, difficulties such as multimodality and dimensionality lie in conducting optimization to find an optimal solution. While solving such a problem, finding a feasible solution, and improving it to the global solution are often required in practice. However, the NM method may fail to locate the global solution since it is vulnerable at strong multimodality and high dimensionality [27], [29]. Sustainable development to overcome the disadvantages has resulted in metaheuristic optimization techniques, which are being employed extensively in academia and industry due to their strength in solving such difficulties [31].

A metaheuristic optimization is a search procedure designed to find a good approximate solution to an optimization problem which is complex and difficult to solve computationally [51]. Modern metaheuristic optimization algorithms tend to be suitable in most cases for global optimization, implementing stochastic search processes in their algorithms. Particularly, teaching-learning-based optimization is a nature-inspired metaheuristic optimization algorithm motivated by the teaching and learning process in a classroom and it has been designed to obtain a global solution with less computational cost and high consistency [31].

The main idea behind the TLBO algorithm is the simulation of a teaching-learning process of the classroom. It is built on the influence of a teacher on the outcome of learners in a class. The teacher is selected among learners as a highly educated person who help learners gain knowledge. The quality of a teacher is certainly a variable of the output of learners and hence it is evident that a good teacher can educate learners so well that they can achieve better outcomes. Also, learners interact with one another to further modify and improve their gained knowledge.

The TLBO algorithm is a stochastic population-based algorithm with a prescribed population size (N_P) and hence an initial population is randomly generated. An individual (X_i) in the

population denotes a single possible solution to a specified optimization problem. Here, X_i is a real D dimensional vector symbolizing the number of design variables associated with an individual. The algorithm attempts to improve each individual during the teacher phase and learner phase by replacing an individual with a better one; each individual accepts his new solution only when it is better than his previous one. This process is called greedy selection. The algorithm continues until it reaches the maximum number (T) of generations.

In the teacher phase, the individual with the best solution takes a role of a teacher ($X_{teacher}$). The algorithm pushes other individuals ($X_{current}$) towards the teacher using the current average of the individuals ($X_{average}$) which measures the qualities of all learners from the current generation. Eq. (4.14) shows how learning improvement of learners can be influenced by the difference between the knowledge of the teacher and the qualities of all learners in the algorithm. For stochastic purpose, randomness is applied to two parameters within the equation: r ranges from 0 and 1; and T_F, defined as a teaching factor, is either 1 or 2, highlighting the significance of the qualities of the learners.

$$X_{new} = X_{current} + r\left(X_{teacher} - T_F X_{average}\right) \tag{4.14}$$

During the learner phase, a learner ($X_{current}$) strives to enhance his knowledge through peer learning from an arbitrary learner ($X_{partner}$). If $X_{partner}$ is better than $X_{current}$ (that is, $f_{partner} < f_{current}$), then $X_{current}$ moves towards $X_{partner}$ as described in Eq. (4.15); otherwise, it moves away from $X_{partner}$ as shown in Eq. (4.16). In the case that a learner (X_{new}) shows a better performance by evaluating Eq. (4.15) and Eq. (4.16), he is welcome to be accepted into the population.

$$X_{new} = X_{current} - r\left(X_{current} - X_{partner}\right) \text{ if } f_{partner} < f_{current} \tag{4.15}$$

$$X_{new} = X_{current} + r\left(X_{current} - X_{partner}\right) \text{ if } f_{current} \leq f_{partner} \quad (4.16)$$

The algorithm repeats its iterations until it reaches the maximum number (T) of generations. The individual (X_{best}) whose function value is the smallest in the population becomes the global minimum of the function. The algorithm of the TLBO method is illustrated in Algorithm 4.2.

4.5 Teaching-Learning-Studying-Based Optimization

There are a variety of complicated real-world problems having many local optimal solutions. In optimizing such problems using the TLBO algorithm, it may require too many iterations to find the global solution or sometimes fail it if the teacher is trapped in one of the local optima and cannot escape from there in the following iterations. In such cases, according to Eq. (4.14), all of the population gradually moves towards the teacher and their positions would be equal to the teacher. This implies that the learning and teaching phases gradually lose their effectiveness in the optimization process and hence the algorithm requires too many iterations for global convergence. Since the position of the teacher affects the overall performance of the algorithm, a new appropriate strategy known as studying phase is proposed by Akbari et al. [32] for the TLBO algorithm to enhance the power of the algorithm. During this phase, each individual attempts to improve its position by appropriately changing each dimension of its position.

$$X_{studying,d} = r_d\left(X_{partner,d} - X_{current,d}\right) \text{ if } f_{partner} < f_{current} \quad (4.17)$$

$$X_{studying,d} = r_d\left(X_{current,d} - X_{partner,d}\right) \text{ if } f_{current} \leq f_{partner} \quad (4.18)$$

In the studying phase, a new partner $X_{partner}$ is randomly chosen and, again, randomness $r_d \in [0, 1]$ is applied to the parameter within the equation for each dimension d. This strategy merges into the global and local search equations (that is, the teaching and learning phases) so that it can

considerably increase the power of the algorithm by extricating the population from their bad positions; it helps effectively to add variety to the population and thus escape from local optima. Good exploration for the global solution can be achieved in this manner.

The modified global and local search equations can be expressed as

$$X_{new} = X_{current} + r\left(X_{teacher} - T_F X_{average}\right) + r_n X_{studying} \tag{4.19}$$

$$X_{new} = X_{current} - r\left(X_{current} - X_{partner}\right) + r_n X_{studying} \text{ if } f_{partner} < f_{current} \tag{4.20}$$

$$X_{new} = X_{current} + r\left(X_{current} - X_{partner}\right) + r_n X_{studying} \text{ if } f_{current} \leq f_{partner} \tag{4.21}$$

where r_n is a normally distributed random number. The algorithm of the TLSBO method is shown in Algorithm 4.2.

4.6 Termination Condition for TLBO and TLSBO

The major disadvantage of the TLBO and TLSBO methods is that different control parameters are required for proper working of these algorithms [52]. A proper selection of the parameters is essential for these algorithms to search the global solution since convergence of the solution is highly dependent of the initial parameters. Especially, an inappropriate selection may affect the convergence rate, and it is hence necessary to tune the parameters by trial and error for fast, robust convergence to the solution, which is a very tedious process [53], [54]. To reduce such tediousness, we have developed a strategy to avoid tuning the maximum number of generations (T), which motivated by the so-called early stopping method in machine learning.

In machine learning, overfitting refers to a modelling error that occurs when a statistical model fits exactly against its training data [55]. Early stopping is a regularization technique used to prevent overfitting while training a model with an iterative optimization method such as the gradient

descent method. Such a method updates the model to make it fit the training data better with each iteration. Up to a certain point, this enhances the performance of the model on the validation data which is outside of the training data. However, after that point, improving the model on the training data leads to an increase in generalization error, which reduces the ability of the model to generate accurate predictions for previously unseen data [56]. To circumvent this issue, early stopping terminates training when the model updates do not yield improvements anymore on the validation data.

By implementing a similar technique into TLBO and TLSBO, we can impose a termination condition on them. Storing the best minimum solution at each generation, if the norm of the difference of the best minimum solutions between two consecutive generations is less than a prescribed error tolerance, then one point is given to the algorithm as a reward; otherwise, the accumulated points are reset to zero. When the score reaches a desired number (for example, 100 in this study), then the solution is considered as a convergent solution and the algorithm is terminated.

4.7 Algorithms

We can solve the linear unmixing problem (1.12) using the NM method in order to estimate the abundance fractions. As seen in Chapter 3, however, it is necessary to implement the bounding process to the algorithm so that the NM method does not solve the unconstrained linear unmixing problem. We can choose either the standard NM method or the adaptive NM method by specifying the parameter alg between "SNM" and "ANM" in Algorithm 4.1.

Algorithm 4.1. Nelder-Mead Method for Linear Unmixing

Input: Mixing matrix M, mixture emission spectrum $\vec{r}$, initial vertex $\vec{\alpha}_0{}^{(0)}$, maximum number of iterations $maxIter$, convergence tolerance ε and algorithm alg

Initialize $k = 0$ and $convergence = false$;
Generate remaining n initial vertices using Eq. (4.12) and Eq. (4.13);
if $alg = SNM$ **then**
  Set $\alpha,\ \beta,\ \gamma,\ \delta$ using Eq. (4.1);
Else
  Set $\alpha,\ \beta,\ \gamma,\ \delta$ using Eq. (4.11);
end if
while *not convergence* **do**
  Sort $\left\{\vec{\alpha}_i{}^{(k)}\right\}_{i=0}^{n}$ using Eq. (4.2);
  Calculate $\vec{\alpha}_m{}^{(k)}$ using Eq. (4.3);
  Calculate $\vec{\alpha}_r{}^{(k)}$ using Eq. (4.4);
  Calculate $J\left(\vec{\alpha}_r{}^{(k)}\right)$, $J\left(\vec{\alpha}_0{}^{(k)}\right)$, $J\left(\vec{\alpha}_{n-1}{}^{(k)}\right)$ and $J\left(\vec{\alpha}_n{}^{(k)}\right)$ using Eq. (2.1);
  if $J\left(\vec{\alpha}_r{}^{(k)}\right) < J\left(\vec{\alpha}_0{}^{(k)}\right)$ **then**
    Calculate $\vec{\alpha}_e{}^{(k)}$ using Eq. (4.5);
    Calculate $J\left(\vec{\alpha}_e{}^{(k)}\right)$ using Eq. (2.1);
    if $J\left(\vec{\alpha}_e{}^{(k)}\right) < J\left(\vec{\alpha}_r{}^{(k)}\right)$ **then**
      Set $\vec{\alpha}_n{}^{(k)} = \vec{\alpha}_e{}^{(k)}$;
    Else
      Set $\vec{\alpha}_n{}^{(k)} = \vec{\alpha}_r{}^{(k)}$;
    end if
  elseif $J\left(\vec{\alpha}_r{}^{(k)}\right) > J\left(\vec{\alpha}_{n-1}{}^{(k)}\right)$ **then**
    if $J\left(\vec{\alpha}_r{}^{(k)}\right) \leq J\left(\vec{\alpha}_n{}^{(k)}\right)$ **then**
      Set $\vec{\alpha}_n{}^{(k)} = \vec{\alpha}_r{}^{(k)}$;
    end if

Calculate $\vec{\alpha}_c{}^{(k)}$ using Eq. (4.6);

Calculate $J\left(\vec{\alpha}_c{}^{(k)}\right)$, $J\left(\vec{\alpha}_n{}^{(k)}\right)$ using Eq. (2.1);

if $J\left(\vec{\alpha}_c{}^{(k)}\right) > J\left(\vec{\alpha}_n{}^{(k)}\right)$ **then**

Perform shrinkage using Eq. (4.7);

Else

Set $\vec{\alpha}_n{}^{(k)} = \vec{\alpha}_c{}^{(k)}$;

end if

Else

Set $\vec{\alpha}_n{}^{(k)} = \vec{\alpha}_r{}^{(k)}$;

end if

Bound $\left\{\vec{\alpha}_i{}^{(k)}\right\}_{i=0}^{n}$ with lower bound 0 and upper bound 1;

Calculate $\Delta^{(k)}$ using Eq. (4.9);

if $\Delta^{(k)} < \varepsilon$ or $k = maxIter$ **then**

Set $convergence = true$;

end if

end while

Select $\vec{\alpha}_{best}$;

Output: Abundance vector $\vec{\alpha}$

Similar to the NM method, the bounding process is required for the TLBO method and the TLSBO method so that they do not find an unconstrained solution to the problem. By specifying the parameter alg in Algorithm 4.2, one can select either the TLBO method or the TLSBO method to obtain optimal abundance fractions.

Algorithm 4.2. Teaching-Learning-Based-Optimization Method with Termination Condition for Linear Unmixing

Input: Mixing matrix M, mixture emission spectrum $\vec{r}$, number of abundance fractions D,

population size N_p, maximum number of iterations $maxIter$, convergence tolerance ε and algorithm alg

Initialize $point = 0$, $k = 0$ and $convergence = false$;
Generate a random population;
Calculate the function values of the population using Eq. (2.1);
Choose $\vec{\alpha}_{best}$;
Set $\vec{\alpha}_{old} = \vec{\alpha}_{best}$;
while *not convergence* **do**
  for $n = 1$ **to** N_p **do**
    if $alg = TLSBO$ **then**
      {Studying Phase}
      for $d = 1$ **to** D **do**
        Choose $\vec{\alpha}_{partner}$ randomly;
        Calculate $J(\vec{\alpha}_{current})$ and $J(\vec{\alpha}_{partner})$ using Eq. (2.1);
        if $J(\vec{\alpha}_{partner}) < J(\vec{\alpha}_{current})$ **then**
          Set $\vec{\alpha}_{studying,d}$ using Eq. (4.17);
        Else
          Set $\vec{\alpha}_{studying,d}$ using Eq. (4.18);
        end if
      end for
    end if
    {Teacher Phase}
    Choose $\vec{\alpha}_{teacher}$;
    Calculate $\vec{\alpha}_{average}$;
    if $alg = TLSBO$ **then**
      Generate $\vec{\alpha}_{new}$ using Eq. (4.19);
    Else
      Generate $\vec{\alpha}_{new}$ using Eq. (4.14);
    end if
    Bound $\vec{\alpha}$ with lower bound 0 and upper bound 1;
    Calculate $J(\vec{\alpha}_{current})$ and $J(\vec{\alpha}_{new})$ using Eq. (2.1);
    if $J(\vec{\alpha}_{new}) < J(\vec{\alpha}_{current})$ **then**
      Set $\vec{\alpha}_{current} = \vec{\alpha}_{new}$;

```
    end if
    {Learner Phase}
    Choose $\vec{\alpha}_{partner}$ randomly;
    Calculate $J(\vec{\alpha}_{current})$ and $J(\vec{\alpha}_{partner})$ using Eq. (2.1);
    if $J(\vec{\alpha}_{partner}) < J(\vec{\alpha}_{current})$ then
      if $alg = TLSBO$ then
        Generate $\vec{\alpha}_{new}$ using Eq. (4.20);
      Else
        Generate $\vec{\alpha}_{new}$ using Eq. (4.15);
      end if
    Else
      if $alg = TLSBO$ then
        Generate $\vec{\alpha}_{new}$ using Eq. (4.21);
      Else
        Generate $\vec{\alpha}_{new}$ using Eq. (4.16);
      end if
    end if
    Bound $\vec{\alpha}$ with lower bound 0 and upper bound 1;
    Calculate $J(\vec{\alpha}_{current})$ and $J(\vec{\alpha}_{new})$ using Eq. (2.1);
    if $J(\vec{\alpha}_{new}) < J(\vec{\alpha}_{current})$ then
      Set $\vec{\alpha}_{current} = \vec{\alpha}_{new}$;
    end if
    Set $k = k + 1$;
  end for
  Choose $\vec{\alpha}_{best}$;
  if $\|\vec{\alpha}_{best} - \vec{\alpha}_{old}\| < \varepsilon$ then
    Set $point = point + 1$;
  Else
    Set $point = 0$;
  end if
  Set $\vec{\alpha}_{old} = \vec{\alpha}_{best}$;
  if $point = 100$ or $k = maxIter$ then
    Set $convergence = true$;
```

65 **end if**
66 **end while**
67 Choose $\vec{\alpha}_{best}$;
Output: Abundance vector $\vec{\alpha}$

4.8 Conclusion

We have seen that global optimization algorithms, the NM method and the TLBO method, are applicable to the linear unmixing problem. The NM method is a global optimization technique, which generates and improves iteratively simplices to locate an optimum. It is, however, known as a weak global optimization method because it is vulnerable at strong multimodality. Furthermore, the inherent problem of the NM method is the effect of dimensionality; indeed, it may find a suboptimal solution to a high-dimensional problem if we choose the standard parameters in Eq. (4.1). To overcome this issue, the adaptive parameters in Eq. (4.11) are suggested, enabling the algorithm to approximate an optimal solution even in high dimensions.

The TLBO method is a nature-inspired metaheuristic optimization algorithm motivated by the teaching and learning process in a classroom. As a strong global optimization technique, it has been developed to obtain a global solution regardless of dimensionality and multimodality. In this algorithm, the role of the teacher is very significant since all learners constantly move their positions towards the teacher. If the teacher is confined in the local optimum, the algorithm may take too many iterations for global convergence. The TLSBO method solves this issue. Applying some changes to the positions of all population helps them to flee from their bad positions, which results in faster convergence. However, finding the parameters, the population size, and the number of generations, by trial and error is a tedious, time-consuming task. By implementing the early stopping technique

into the TLBO and TLSBO algorithms, we can reduce effort on finding the proper number of generations.

A fluorescence spectrum is a fluorescent signal. This fact implies that it may contain unintended noise while the spectrofluorometer captures it. We then may want to ask if the linear unmixing algorithms work properly on noisy spectra. If it is not true, can they then unmix filtered spectra when we denoise such noisy spectra? In the next chapter, we will investigate denoising algorithms to filter noisy spectra and their applications to the linear unmixing algorithms.

Chapter 5

Since fluorescence is a highly sensitive analytical technique, emission spectra, which are fundamentally fluorescent signals, may contain unintended noise due to the sensitivity of the spectrofluorometer. This chapter discusses denoising techniques that rely on Fourier transform and wavelet transform to filter noise from the signals and presents their algorithms.

5.1 Fourier Transform (FT)

Wave functions consist of energy at a fundamental frequency and at harmonic frequencies. The shape of the wave function is determined by the proportions of energy at the fundamental and the harmonic frequencies, implying that the wave function can be represented as a sum of sine and cosine functions with unique constants. The summation and the constants are called a Fourier series and Fourier coefficients, respectively [57].

The FT is a mathematical method of transforming a function of time (or space) to a function of frequency. Thus, it decomposes a waveform in a time domain into a combination of sinusoidal terms, each with a unique magnitude, frequency, and phase [58]. The FT process converts the time-based waveform expressed in complex functions into sinusoidal functions, which when combined, can exactly replicate the original waveform. In particular, the relationship between the FT of a continuous function f at frequency ξ and its inverse FT is described as

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i \xi t} dt \tag{5.1}$$

$$f(t) = \int_{-\infty}^{\infty} F(\xi) e^{2\pi i \xi t} d\xi \tag{5.2}$$

where $e^{i\theta}$ can be expressed as a sum of sines and cosines according to Euler's formula

$$e^{i\theta} = \cos\theta + i\sin\theta. \tag{5.3}$$

The FT and its inverse are thus said to be in a one-to-one mapping between the time and frequency domains.

However, a computer cannot work with a continuous-time signal, and it is hence necessary to take some samples of the signal and analyze these samples instead of the original signal. Moreover, since the computer can process only a finite number of samples, it is also necessary to make an approximation and use a limited number of samples. Therefore, a finite-duration sequence is generally used to represent a continuous-time signal which may extend to positive infinity on the time axis [33]. Considering a continuous-time signal as such a discrete-time signal, the discrete Fourier transform (DFT) becomes a powerful tool used to convert a finite sequence of waveform data in the time domain into equally spaced data in the frequency domain. The original data are restored through an additional Fourier analysis, known as the inverse DFT, using FT samples as the coefficients of complex sinusoids at the corresponding FT frequencies [59].

The DFT is the most common technique of Fourier analysis applied to a discrete complex-valued series [60]. The DFT to transform a sequence of N complex-valued samples $\{x_n\}$ into $\{X_k\}$ is defined as

$$X_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i k n / N} \tag{5.4}$$

and its inverse is given by

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{2\pi i k n/N}. \tag{5.5}$$

Thus, the DFT is a linear operator that maps the data points in the time domain $\{x_n\}$ to the frequency domain $\{X_k\}$. Letting $\omega_N = e^{-2\pi i/N}$, the DFT may be computed by matrix multiplication as follows [33].

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_{N-1} \end{bmatrix} = \begin{bmatrix} \omega_N^0 & \omega_N^0 & \omega_N^0 & \cdots & \omega_N^0 \\ \omega_N^0 & \omega_N^1 & \omega_N^2 & \cdots & \omega_N^{N-1} \\ \omega_N^0 & \omega_N^2 & \omega_N^4 & \cdots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \omega_N^0 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \cdots & \omega_N^{(N-1)^2} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix} \tag{5.6}$$

where ω_N^k for $k = 0, 1, \ldots, N-1$ are the N^{th} roots of unity with the property $\omega_N^k = \omega_N^{k+lN}$ for all integers l. The output column vector $\vec{X}$ contains the Fourier coefficients for the input vector $\vec{x}$, and the N-point DFT matrix is a complex valued matrix, thus one can obtain both magnitude and phase information from the output $\vec{X}$.

Even though the DFT is tremendously useful for numerical approximation and computation, the simple formulation in Eq. (5.6) involves multiplication requiring $O(N^2)$ operations, which is often computationally too expensive to be practical especially when the value of N is very large. Fortunately, the fast Fourier transform (FFT) algorithm, an optimized approach for implementing DFT, was developed to solve the problem of high computational cost. By using FFT, the computational complexity of DFT can be reduced from $O(N^2)$ to $O(N \log_2 N)$ [61]. Thus, as N becomes very large, the term $\log_2 N$ grows slowly and the algorithm approaches a linear scaling. Although the different computational complexity between the DFT and FFT algorithms may seem like a small difference, FFT is widely used in many practical applications due to the relatively inexpensive cost of $O(N \log_2 N)$.

The FFT algorithm is built on a symmetry in the Fourier transform that allows an N dimensional DFT to be solved with a number of smaller dimensional DFT computations [33]. Splitting the terms in Eq. (5.4) into even terms and odd terms, the following equation is obtained.

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-2\pi i k(2m)/N} + \sum_{m=0}^{N/2-1} x_{2m+1} e^{-2\pi i k(2m+1)/N} \tag{5.7}$$

It follows that

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i k m}{N/2}} + \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i k(m+1/2)}{N/2}} \tag{5.8}$$

and so

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i k m}{N/2}} + \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i k m}{N/2} - \frac{\pi i k}{N/2}}, \tag{5.9}$$

which results in

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i k m}{N/2}} + e^{-\frac{2\pi i k}{N}} \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i k m}{N/2}}. \tag{5.10}$$

Now consider $X_{k+N/2}$ by substituting $k + N/2$ into k. Then

$$X_{k+N/2} = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i (k+N/2) m}{N/2}} + e^{-\frac{2\pi i (k+N/2)}{N}} \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i (k+N/2) m}{N/2}} \tag{5.11}$$

leading to

$$X_{k+N/2} = \sum_{m=0}^{N/2-1} x_{2m} e^{-2\pi i - \frac{2\pi i k m}{N/2}} + e^{-\pi i - \frac{2\pi i k}{N}} \sum_{m=0}^{N/2-1} x_{2m+1} e^{-2\pi i - \frac{2\pi i k m}{N/2}}. \tag{5.12}$$

Since $e^{-\pi i} = -1$, it follows that

$$X_{k+N/2} = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi ikm}{N/2}} - e^{-\frac{2\pi ik}{N}} \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi ikm}{N/2}}. \quad (5.13)$$

It is remarkable that Eq. (5.10) and Eq. (5.13) have a very similar structure except the sign between two summations. Due to the similarity, it is possible to compute X_k and $X_{k+N/2}$ symmetrically for DFT and hence numerous computations can be avoided. The FFT algorithm uses this useful fact to reduce the complexity of the DFT. The detailed explanation can be found in [62].

5.2 Denoising with FFT

Analyzing measured data has its own challenges involving unpredictable conditions and systematic measurement errors. These factors introduce noise into the data, thereby complicating analysis and possibly causing biased or incorrect conclusions.

One of the most useful Fourier analysis applications is determining the noise frequencies and ascertaining noise sources in experimental data [60]. Since this can be achieved using FT, the FFT is often used for noise filtering in digital signal processing, offering a computationally rapid and efficient method for DFT computation.

Given a noisy signal, assuming that the signal and the noise are non-correlated and that the noise is not dominant in the signal, the main idea is to find the real signal frequencies and to obtain a reconstructed signal by using only the significant frequencies of the signal. The non-relevant frequencies are set to zero.

When computing the FFT of the noisy signal, the power spectral density (PSD), which is the normalized squared magnitude of $\vec{X}$ in Eq. (5.6), can be obtained [33].

$$PSD(X_k) = \frac{|X_k|^2}{N} = \frac{X_k \bar{X}_k}{N} \tag{5.14}$$

where N is the number of samples and $0 \leq k \leq N-1$. The PSD indicates how much power (i.e., significant information) the signal contains in each frequency. By wiping out all the components that have power below a certain threshold, one can remove noise from the signal. After inverse transforming the filtered signal, we can obtain the denoised signal.

The threshold value is determined experimentally by computing the correlation value between the original signal and the denoised signal [63]. The correlation value γ is defined as

$$\gamma = \frac{\sum_{i=0}^{N-1}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=0}^{N-1}(X_i - \bar{X})^2 (Y_i - \bar{Y})^2}} \tag{5.26}$$

where X_i are the samples of the original signal X, Y_i are the samples of the denoised signal Y, and $\bar{X}$ and $\bar{Y}$ are the mean values of the samples X_i and Y_i, respectively. As the value of γ approaches 1, the selection becomes more appropriate. The algorithm of noise reduction in a noisy signal using the FFT is presented in Algorithm 5.1.

5.3 Wavelet Transform (WT)

Although the FT provides detailed information about the frequency of a given signal, it does not give any information about when those frequencies occur in time. In fact, in Fourier analysis, there is a fundamental uncertainty principle that temporal information is not obtainable as the frequency content is specified, and vice versa [64]. The more concentrated a signal is in the time domain, the more spread out it is in the frequency domain. Measuring the uncertainty of a function in terms of its variance, the Fourier uncertainty principle indicates that there exists a lower bound on the

product of the variances of a function and its Fourier transform [65].

If a function f is a complex-valued function, then its Fourier variance is defined as

$$Var(f) = \int_{-\infty}^{\infty} t^2 |f(t)|^2 \, dt \tag{5.15}$$

where the function $t^2|f(t)|^2$ is the dispersion about $t = 0$. Eq. (5.15) can be viewed as the variance of a random variable with mean 0 and probability density function $|f(t)|^2$. Unlike the conventional probability theorem, the Fourier variance can be applied to any functions for which the integral converges; it is not restricted to the case when $|f(t)|^2$ integrates to 1.

The Fourier uncertainty principle states that

$$Var(f)Var\left(\hat{f}\right) \geq C\|f\|_2^2 \left\|\hat{f}\right\|_2^2 \tag{5.16}$$

where $\hat{f}$ is the FT of f and C is a constant. The inequality in Eq. (5.16) gives a lower bound on how spread out the two quantities $Var(f)$ and $Var\left(\hat{f}\right)$ must be; if the width of f is very narrow, then $\hat{f}$ becomes a very broad function, and vice versa. As an extreme example, when f is the Dirac delta function δ, $\hat{f}$ becomes a constant function since $f(0) = 1$ by Eq. (5.1). If f is confined in a small region in time domain so that it is highly localized, then the spread of $\hat{f}$ in frequency domain becomes very large, and vice versa. In this extreme limit, a time series is perfectly resolved in time, but provides no information about frequency content, and the FT perfectly resolves frequency content, but provides no information about when in time these frequencies occur [33].

In Fourier analysis, the uncertainty principle indicates that we lose information about time as the frequency content is specified, and vice versa. One comes at the expense of the other between the time and frequency domains. To solve such an issue, an alternative approach known as a multi-resolution analysis was introduced. Wavelet analysis extends the concepts in Fourier analysis and partially overcome the uncertainty principle by using a multi-resolution decomposition [64]. The

fundamental idea in wavelet analysis is the use of the mother wavelet and generate a family of scaled and translated versions of the wavelet. The mother wavelet ψ is given by

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \tag{5.17}$$

where a is a scale parameter and b is a translation parameter. The continuous wavelet transform (CWT) is defined as

$$W_{\psi_{a,b}}(f) = \int_{-\infty}^{\infty} f(t)\bar{\psi}_{a,b}(t)dt \tag{5.18}$$

with its inverse

$$f(t) = \frac{1}{C_\psi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{1}{a^2}W_{\psi_{a,b}}(f)\psi_{a,b}(t)da\,db \tag{5.19}$$

where

$$C_\psi = \int_{-\infty}^{\infty}\frac{\left|\hat{\psi}(\xi)\right|^2}{|\xi|}d\xi < \infty. \tag{5.20}$$

Eq. (5.20) is called the boundedness property and the mother wavelet must satisfy this property [33].

Since the results of CWT are wavelet coefficients which are a function of a and b, the signal can be expressed as the combination of wavelets of different scales and positions; that is, the wavelet transform decomposes the signal into different scales with different levels of resolution by scaling the mother wavelet. The compressed scaled wavelet captures all the high frequency components available in the signal since it has a high frequency, and the stretched wavelet captures the low frequency contents in the signal because it has a low frequency. At high frequencies, it provides good time resolution and poor frequency resolution. Eq. (5.18) can be viewed as the correlation between a signal and the scaled and translated mother wavelets, which enables the multi-resolution analysis (MRA) since it analyzes a signal into scales with different time and frequency resolution.

In CWT, however, calculating wavelet coefficients at every possible scale is often redundant, generating too much data. By selecting scales and positions to be discrete, the original signal can be completely reconstructed by a sample version of WT and analysis also becomes much easier. For a time series $f(t)$, the discrete wavelet transform (DWT) is given by

$$W_{\psi_{j,k}}(f) = \int_{-\infty}^{\infty} f(t)\bar{\psi}_{j,k}(t)dt \tag{5.21}$$

with a discrete family of wavelets $\psi_{j,k}$:

$$\psi_{j,k}(t) = a_0^{-j/2}\psi\left(a_0^{-j}t - b_0k\right) \tag{5.22}$$

where j is the decomposition level and k is the time translation factor, and a_0 and b_0 are constants. In practice, $W_{\psi_{j,k}}(f)$ is typically sampled in a dyadic grid, that is, $a_0 = 2$ and $b_0 = 1$. It is known that this sampling method provides good time-frequency localization [66]. Thus, Eq. (5.22) becomes

$$\psi_{j,k}(t) = 2^{-j/2}\psi\left(2^{-j}t - k\right). \tag{5.23}$$

The sub-signals f_j of the original signal under the level j can be reconstructed by

$$f_j(t) = \sum_{k=-\infty}^{\infty} W_{\psi_{j,k}}(f)\psi_{j,k}(t) \tag{5.24}$$

and the sum of the sub-signals becomes the original signal as below.

$$f(t) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} W_{\psi_{j,k}}(f)\psi_{j,k}(t) \tag{5.25}$$

The maximum possible decomposition level L can be computed as $L = \lfloor log_2 N \rfloor$ where N is the number of samples. As the decomposition level increases, more sub-signals and detailed information of the original signal are attainable, but more computational cost is required.

There are several methods to implement the DWT algorithm. The commonly used method is the dyadic algorithm. In this algorithm, two filters which are high-pass filter and low-pass filter are

constructed from the wavelet coefficients and those filters are recurrently used to obtain data for all the scales. If the total number of data $N = 2^m$ is used and the signal length is L, $N/2$ data at scale $L/2^{m-1}$ are computed in the first level, and then $(N/2)/2$ data at scale $L/2^{m-2}$ in the second level, and so forth. This procedure continues up to finally obtaining 2 data at scale $L/2$. Therefore, the dyadic DWT algorithm performs a multi-level decomposition, resulting in approximation and detail coefficients according to the decomposition level. The result of this algorithm is an array of the same length as the input signal, where the data are usually sorted from the largest scales to the smallest ones. The details of the dyadic DWT algorithm can be found in [67].

5.4 Denoising with WT

By decomposing a noisy signal down to a specific level, we can obtain its wavelet coefficients. We can reconstruct the denoised signal by transforming back a limited number of highest magnitude wavelet coefficients with the wavelet basis into time domain. This method is known as wavelet-based denoising technique, and it has demonstrated its efficiency in noise removal in a noisy signal [68].

To filter a noisy signal using the wavelet-based technique, we should determine the mother wavelet and decomposition level to be used for noise reduction [66]. Theoretically, we can select any mother wavelet if its family meets both the orthogonality property and the boundedness property in Eq. (5.20). There are several methods to choose a proper mother wavelet, but the choice is generally based on the visual resemblance between a noisy signal and a mother wavelet [69], [70]. For a proper decomposition level, level 4, 5, 6, or 7 is typically preferrable in signal processing experiments [68], [70].

Similar to the Fourier-based denoising algorithm, we can eliminate noise from the noisy signal by erasing all the wavelet coefficients that have power below a specific threshold, and the threshold value is based on the correlation value the original signal and the denoised signal. The denoising algorithm using the wavelet-based technique is summarized in Algorithm 5.2.

5.5 Algorithms

It is not an easy task to find a proper threshold value for denoising algorithms. Alternatively, we will determine the percentage of high-power coefficients to keep rather than the exact threshold value, as in [33].

Algorithm 5.1. Noise Reduction using Fast Fourier Transform

Input: Noisy signal samples $\{\tilde{x}_n\}_{n=0}^{N-1}$ and coefficient percentage P

Calculate the number of samples N;
Calculate $\{X_k\}_{k=0}^{N-1}$ using Eq. (5.4) with FFT;
Calculate $\{PSD(X_k)\}_{k=0}^{N-1}$ using Eq. (5.14);
Sort $\{PSD(X_k)\}$ by magnitude and store it as $\{\widetilde{PSD}(X_k)\}$;
Calculate $I = \lfloor 0.01 * P * N \rfloor$;
Set the I^{th} component of $\{\widetilde{PSD}(X_k)\}$ as threshold T;
for $i = 0$ **to** $N - 1$ **do**
  if $PSD(X_i) < T$ **then**
    Set $X_i = 0$;
  end if
end for
Calculate $\{x_n\}_{n=0}^{N-1}$ using Eq. (5.5) with FFT;

Output: Filtered signal samples $\{x_n\}_{n=0}^{N-1}$

The wavelet-based method requires two additional parameters, mother wavelet and decomposition level.

Algorithm 5.2. Noise Reduction using Wavelet-based Method

Input: Noisy signal samples $\{\tilde{f}(t_n)\}_{n=0}^{N-1}$, mother wavelet ψ, decomposition level L and coefficient percentage P

Calculate $\{W_{\psi_{L,k}}\}$ using Eq. (5.21) and Eq. (5.23) with the dyadic DWT algorithm;
Calculate the number S of $\{W_{\psi_{L,k}}\}$;
Sort $\{|W_{\psi_{L,k}}|\}$ by magnitude and store it as $\{|\widetilde{W}_{\psi_{L,k}}|\}$;
Calculate $I = \lfloor 0.01 * P * N \rfloor$;
Set the I^{th} component of $\{|\widetilde{W}_{\psi_{L,k}}|\}$ as threshold T;
for $i = 1$ **to** S **do**
  if $|W_{\psi_{L,k}}|_i < T$ **then**
    Set $\{W_{\psi_{L,k}}\}_i = 0$;
  end if
end for
Compute $\{f(t_n)\}_{n=0}^{N-1}$ using Eq. (5.25);

Output: Filtered signal samples $\{f(t_n)\}_{n=0}^{N-1}$

5.6 Conclusion

The Fourier-based method and the wavelet-based method share one common fact: after transforming a noisy signal, denoising is accomplished by maintaining significant coefficients, erasing insignificant coefficients, and then inverse transforming the filtered coefficients to obtain the denoised signal. However, the wavelet-based method allows more flexibility and thus better denoising performance due to more options on the choice of the mother wavelet and decomposition level, whereas the Fourier-based method is restricted to cosine and sine functions without any coefficient

decomposition. The proper selection of the mother wavelet is therefore necessary for the wavelet-based method, which is achieved based on visual similarity between the original signal and the denoised signal.

We have investigated various linear unmixing algorithms to estimate abundance fractions for the linear spectral mixture model, and denoising algorithms to filter noisy spectra. In the next chapter, we will analyze the performance of each linear unmixing method by performing it on the real dataset and observe how much denoising methods can improve the estimates obtained from noisy spectra.

Chapter 6

In this chapter, we will implement linear unmixing algorithms to estimate the abundance fractions of a linear spectral mixture model and evaluate their performance using the resultant abundance fractions. By adding uniformly random noise to the mixture emission spectra, we will also probe how noise affects their performance and then examine whether denoising may lead to a better estimation of abundance fractions.

6.1 Data Set

The laboratory data set considered in [17] will be used as an input to the 14 different algorithms described in the previous chapters: namely, ULS, SCLS (direct and iterative), NCLS, FCLS (direct and iterative), MFCLS (direct and iterative), FC-GDM, GDM, NMM (standard and adaptive), TLBO, and TLSBO methods.

The data set contains 9 reference emission spectra and 48 mixture emission spectra. The reference emission spectra were measured from individual or combined probes (EBFP2, EBFP2-ECFP, EBFP2-mTFP1, EBFP2-LSSmOrange, ECFP, mTFP1, mTFP1-mVenus, mVenus, and LSSmOrange) which are constructed with 5 fluorescent proteins (EBFP2, ECFP, mTFP1, mVenus, and LSSmOrange). The 48 different mixture emission spectra were created from those 9 individual probes spanning two-way probe combinations to all probes present. The pure probes were combined into mixtures with known amounts ($1/n$ where n is the number of probes combined). Mixture samples 1 to 14 are a set of two-way probe combinations, 15 to 23 are three-way, 24 to 28 are four-

way, 29 to 31 are five-way, 32 to 34 are six-way, 35 to 38 are seven-way, 39 to 46 are eight-way, and 47 and 48 are nine-way. The detailed explanation about the experimental method can be found in [17]. The graphs of the emission spectra are presented in Appendix A. From these spectral emission scanning data (See Appendix B for the description about data preprocessing), we solve the linear unmixing problem (1.12) to estimate the abundance fractions in each mixture. These estimated fractions are then compared to the actual fractions in the mixture for performance analysis.

A mixing matrix M is formed from the 9 reference emission spectra detected by 2761 emission wavelength channels with their associated abundance fractions expressed by a vector $\vec{\alpha}$. Since each reference emission spectrum $\vec{m}$ is a 2761×1 real column vector, the mixing matrix M becomes a 2761×9 real matrix and the abundance vector $\vec{\alpha}$ is a 9×1 real column vector. It is known that fluorescence emission follows the principle of linear superposition and thus the columns of the mixing matrix M are linearly independent to one another. In fact, $\text{rank}(M) = 9$ and therefore our linear mixture model is an overdetermined system whose mixing matrix has linearly independent columns.

6.2 Performance Measurements

To characterize the performance of different linear unmixing algorithms, some performance metrics must be employed to compare the actual and estimated counterpart. These metrics include: (1) The numbers of correct probes identified by the algorithm; (2) The number of incorrect probes detected by the algorithm; (3) Least square error using L2 norm of the linear mixture model with estimated abundance fractions; (4) The processing time averaged over 10 simulation runs.

We have developed a metric to quantify the numbers of correct and incorrect probes found

by a linear unmixing algorithm. Given an abundance vector $\vec{\alpha}$, an indicator vector $\vec{T}$ is defined as follows:

$$T_i = \begin{cases} 1 & \text{if } a_i > \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

where ε is a prescribed error tolerance (we will set $\varepsilon = 10^{-3}$ in this work). Now assume that $\vec{T}_{\text{act}}$ and $\vec{T}_{\text{est}}$ are the indicator vectors corresponding to the actual and the estimated abundance vectors, respectively. Then, we define an error vector $\vec{E}$ satisfying

$$E_i = \begin{cases} 1 & \text{if } T_{\text{act},i} > T_{\text{est},i} \\ 0 & \text{if } T_{\text{act},i} = T_{\text{est},i} \\ -10^{-m} & \text{otherwise} \end{cases}$$

where m is the number of digits of the maximum value of n among n-way mixture emission spectra. Lastly, we define a score S such that

$$S = n - \sum_i E_i.$$

Here, it follows that the floor value $\lfloor S \rfloor$ of S is the number of correct probes found by a specific linear unmixing algorithm, and $S - \lfloor S \rfloor$ is the number of additional probes located by the algorithm. Therefore, if we have $S = \lfloor S \rfloor$, then the algorithm perfectly finds the correct probes to be used for a certain mixture emission spectrum. If $S > \lfloor S \rfloor$, then it finds additional unused probes as well as the correct probes.

6.3 Experiment 1

In this study, experiments are designed to demonstrate the performance of all the linear unmixing methods mentioned in Chapter 2. For iterative methods, we will take 1 and 10^{-6} for step size and error tolerance, respectively. We will also investigate proper step sizes for the iterative

methods.

6.3.1 ULS Linear Unmixing Method

The first and second plots in Figure 6.1 show the actual abundance fractions (AAFs) and the estimated abundance fractions (EAFs) of the probes in the mixture samples, respectively, in the dataset using colour intensities. The comparison of these plots reveals how properly the linear unmixing method (the ULS method in this case) unmixes each given n-way mixture sample into n probes, and how accurately the method estimates the abundance fractions. Thus, the similarity of the colour intensities between these two plots indicates the performance of the method for abundance fraction estimation.

The two plots in Figure 6.1 show different colour intensities; the colour pattern in the first plot is monotone, whereas that in the second plot is more polychrome, which occurs when $S \neq \lfloor S \rfloor$, or it estimates wrong abundance fractions although it unmixes a sample into n probes correctly. These plots provide insight into the overall performance of the linear unmixing algorithm, but we still

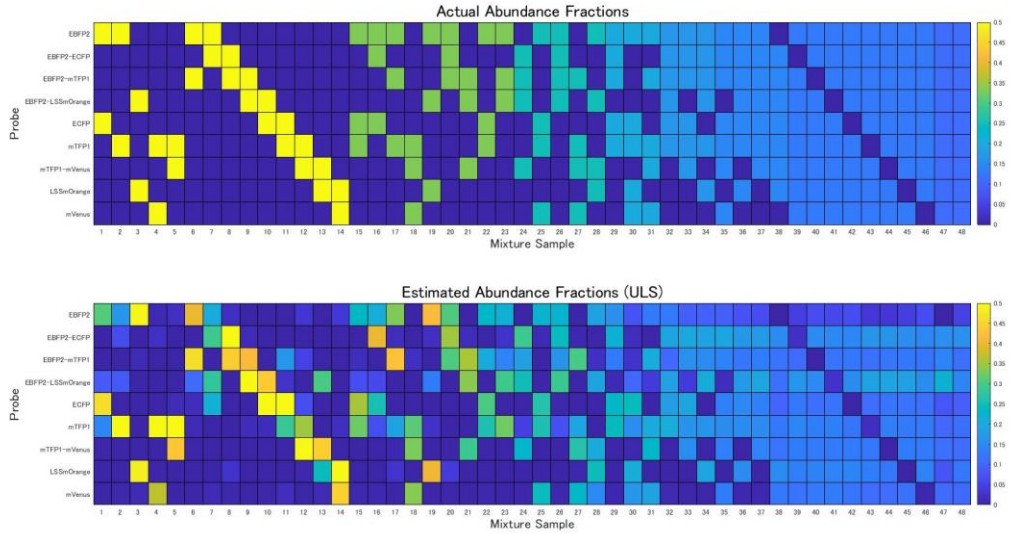

Figure 6.1. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by ULS (lower panel) on mixture samples

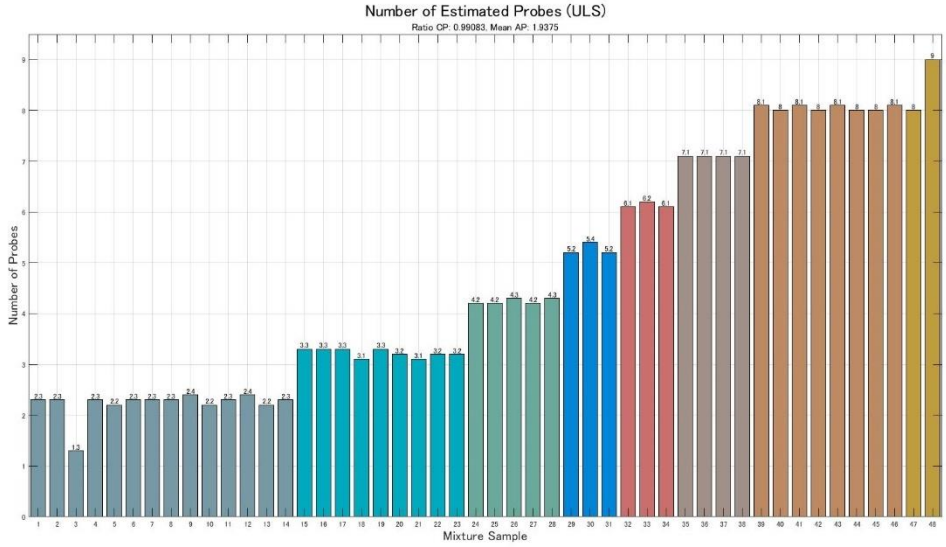

Figure 6.2. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by ULS (lower panel) on mixture samples

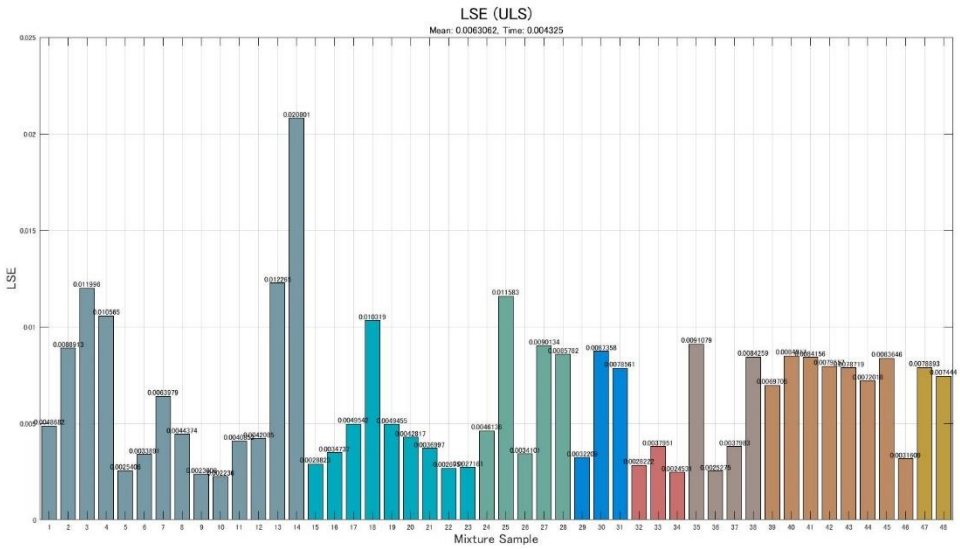

Figure 6.3. Bar graph of least square errors by ULS on mixture samples

need detailed information about these plots to quantify it.

Figure 6.2 depicts the scores defined in Eq. (6.3) for all the mixture samples. It shows that the ULS method generally finds the minimum probes for each mixture sample except mixture samples 3 and 47; it fails to locate one of the necessary probes for those samples. Also, it can find the probes correctly only on mixture samples 40, 42, 44, and 45, but it locates additional unnecessary probes on the other mixture samples. In fact, the success ratio of correct probes (rCP) to be found by the ULS method is 0.99083, and the average numbers of additional probes (mAP) is 1.9375, indicating that the method can at least locate the necessary probes very well, but approximately two more unnecessary probes.

Figure 6.3 shows the least square errors of the linear mixture model calculated with the estimated abundance fractions obtained by the ULS method. The LSE of mixture sample 14 is much larger than those of mixture samples 3 and 47 even though the method finds the correct number of

probes for mixture sample 14 and does not for mixture samples 3 and 47. For the ULS method, the average LSE is 0.0063062 with unmixing time 0.004325s.

6.3.2. Direct SCLS Linear Unmixing Method

The EAF plot in Figure 6.4 still shows a polychromatic pattern as in Figure 6.1. Figure 6.5 tells that it has the same rCP value 0.99083 and a slightly greater mAP value 1.9792. Thus, the method finds the necessary probes properly with approximately two more unnecessary probes. According to Figure 6.6, the direct SCLS method yields, on average, the LSE of 0.0062933, which is somewhat smaller than the ULS method. The direct SCLS method also requires more computing time, 0.0056273s, for unmixing than the ULS method.

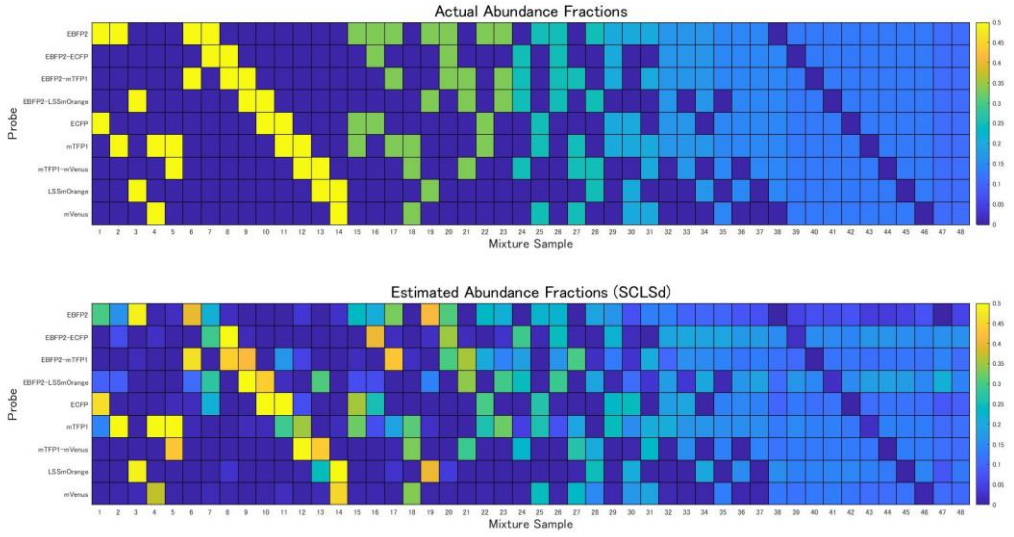

Figure 6.4. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct SCLS (lower panel) on mixture samples

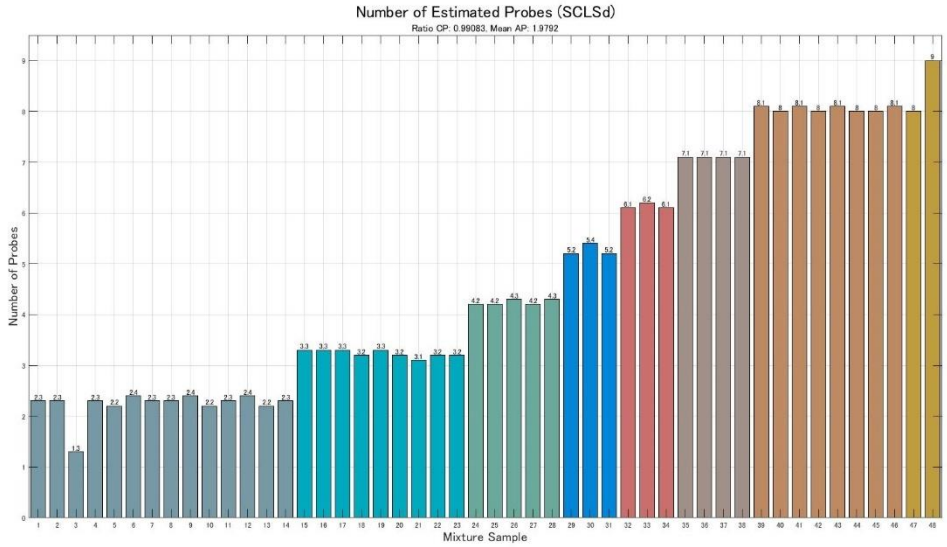

Figure 6.5. Bar graph of the numbers of estimated probes by direct SCLS on mixture samples

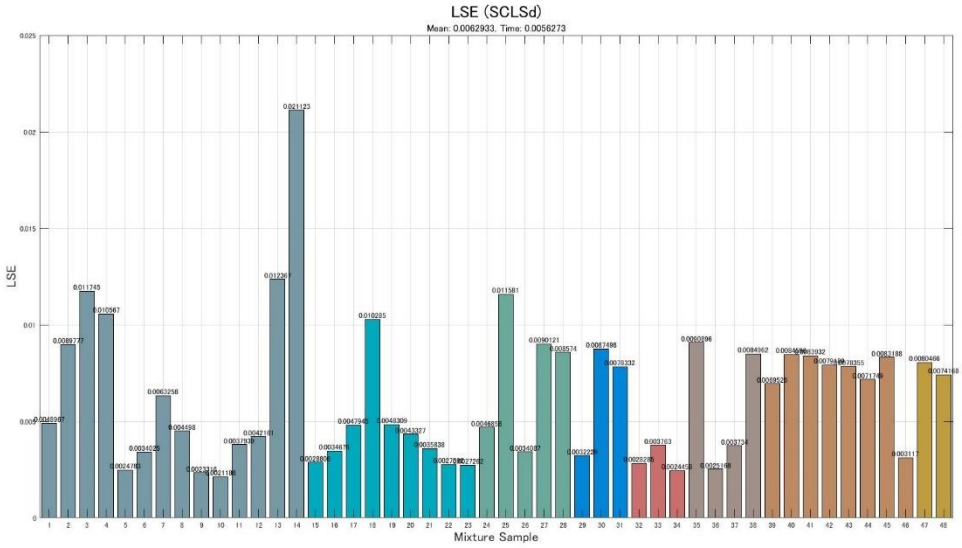

Figure 6.6. Bar graph of least square errors by direct SCLS on mixture samples

6.3.3 Iterative SCLS Linear Unmixing Method

No significant difference is observed in the EAF plot in Figure 6.7 compared to the previous EAF plots. In Figure 6.8, the rCP value, 0.99083, of the SCLSi method is the same as the previous methods, but the mAP value, 2.0625, is larger than those methods. The method tends to find the correct probes, but also locate two more incorrect probes. Figure 6.9 shows that the indirect SCLS method produces the average LSE of 0.0063233, which is greater than both the ULS method and the direct SCLS method. The iterative SCLS method takes even more time, 0.0080092s, compared to those methods.

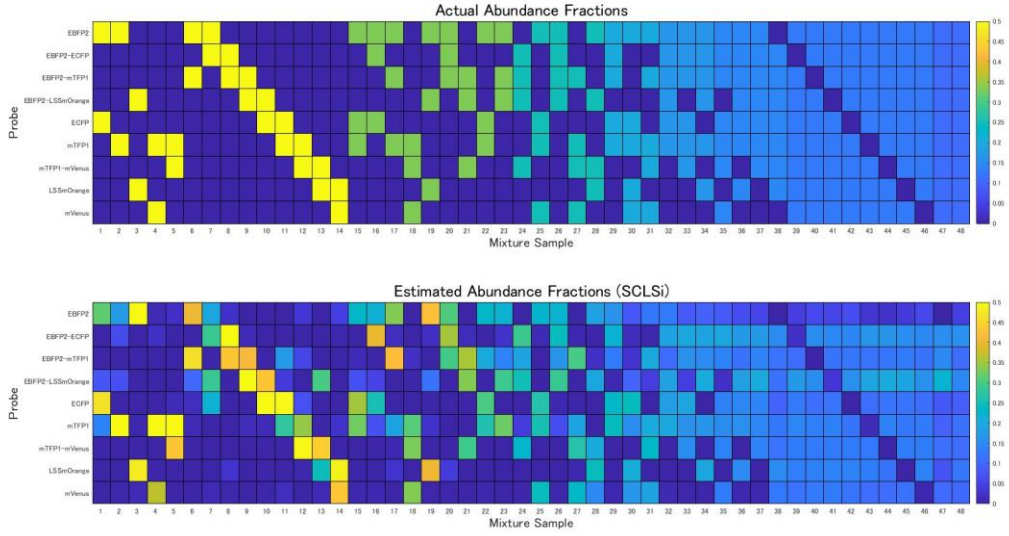

Figure 6.7. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by iterative SCLS (lower panel) on mixture samples

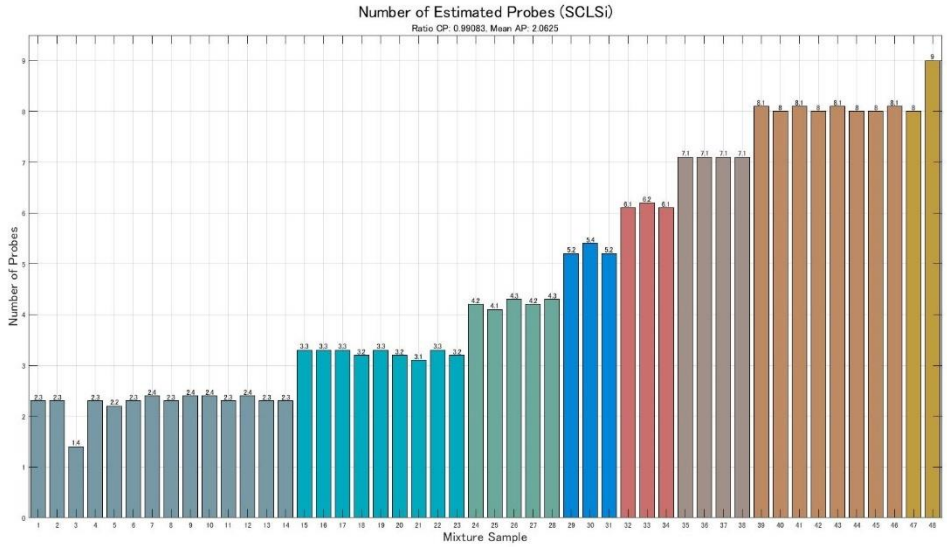

Figure 6.8. Bar graph of the numbers of estimated probes by iterative SCLS on mixture samples

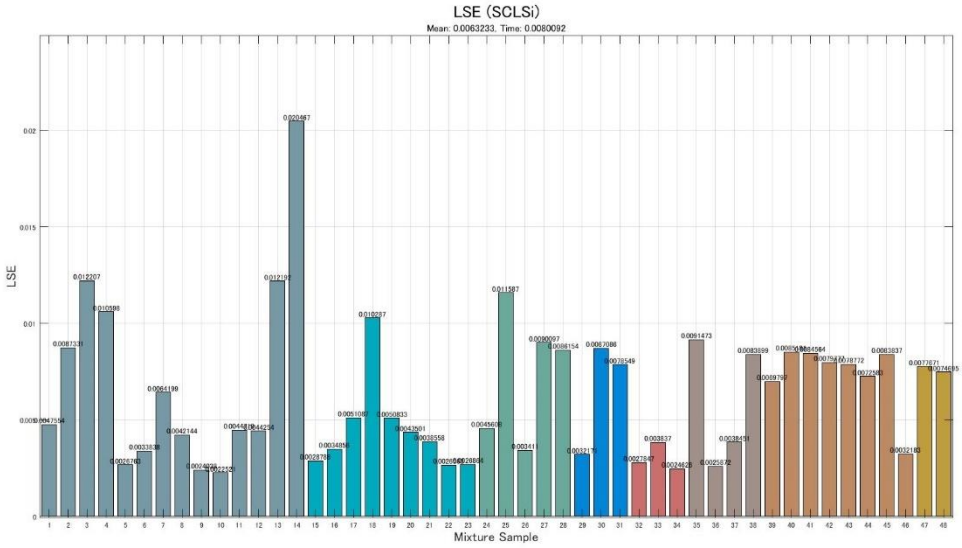

Figure 6.9. Bar graph of least square errors by iterative SCLS on mixture samples

6.3.4 NCLS Linear Unmixing Method

In Figure 6.10, the EAF plot for the NCLS method shows a relatively less chaotic pattern (and thus more similarity to the AAF plot) than the previous methods. In Figure 6.11, the rCP value is still the same, but the mAP value, 0.85417, becomes much smaller. This illustrates that the method generally finds the necessary probes with less than one unnecessary probe. We can find many integer-valued scores in the figure. Figure 6.12 shows that the NCLS method produces the average LSE of 0.0060268 with time 0.018064s. We can thus observe an improvement in the LSE with a longer unmixing time for the NCLS method.

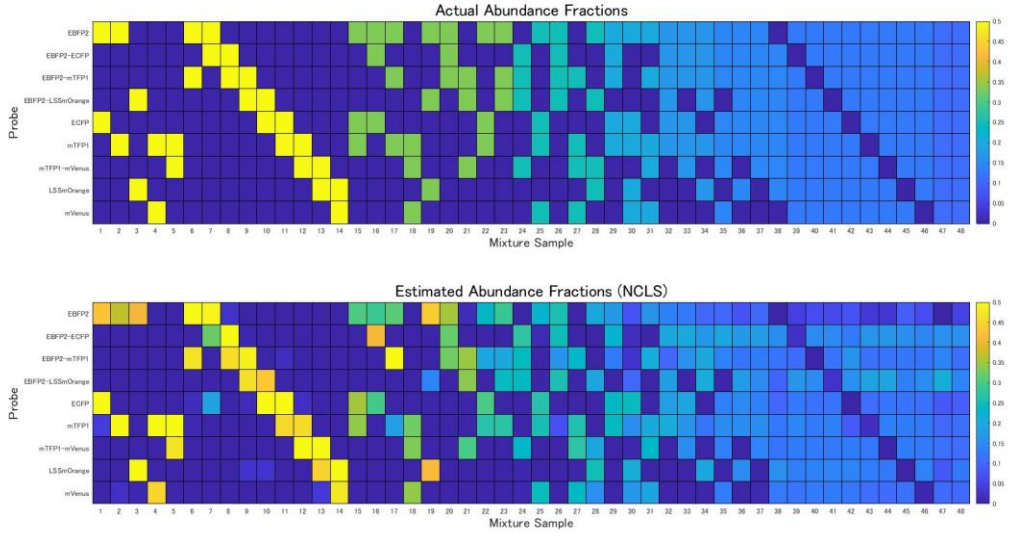

Figure 6.10. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by NCLS (lower panel) on mixture samples

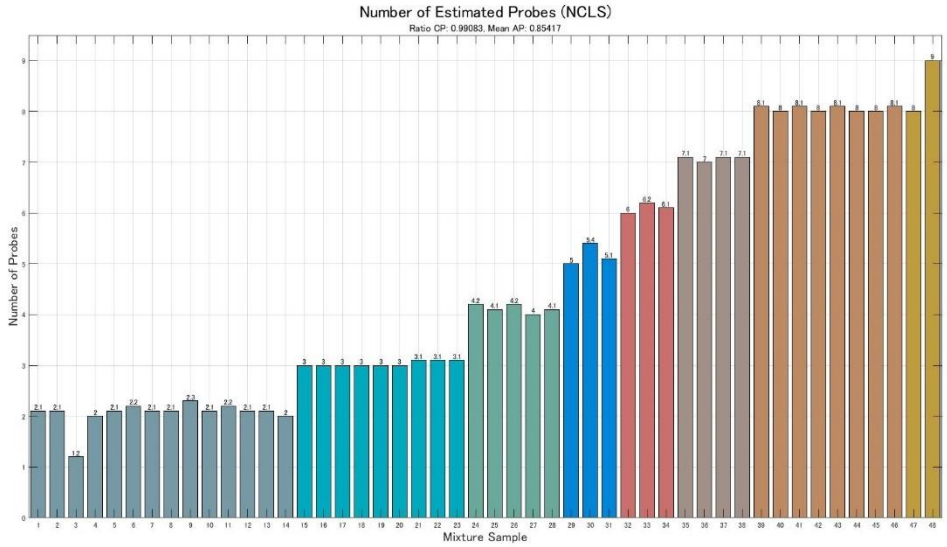

Figure 6.11. Bar graph of the numbers of estimated probes by NCLS on mixture samples

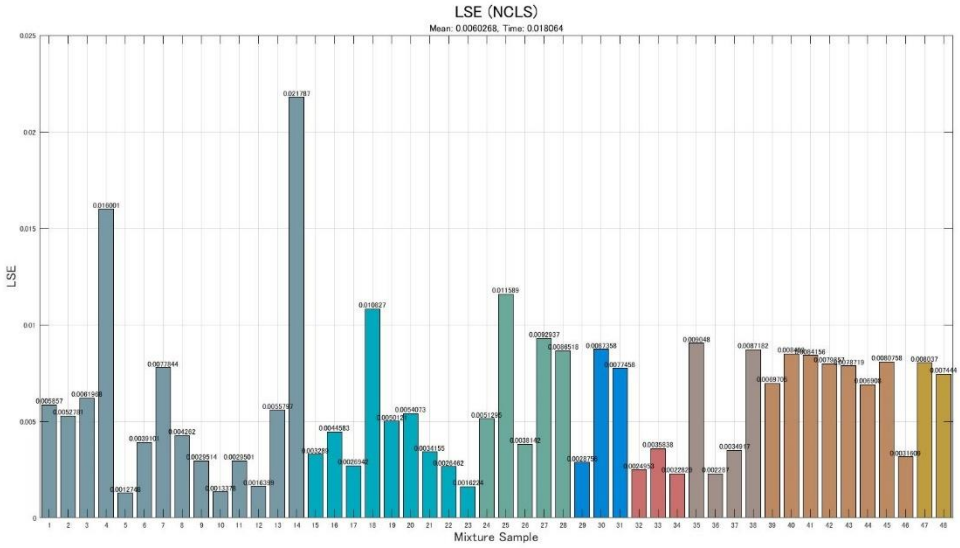

Figure 6.12. Bar graph of least square errors by NCLS on mixture samples

6.3.5 Direct FCLS Linear Unmixing Method

In Figure 6.13, the EAF plot for the direct FCLS method a similar pattern to the NCLS method. According to Figure 6.14, it still has the same rCP value 0.99083, but the mAP value 1.0625 is somewhat larger than the NCLS method. Thus, the method locates the necessary probes with approximately one unnecessary probe. Figure 6.15 shows that the direct FCLS method produces the average LSE of 0.0060189 which is slightly smaller than the NCLS method, but the unmixing time, 0.029525s, is fairly longer.

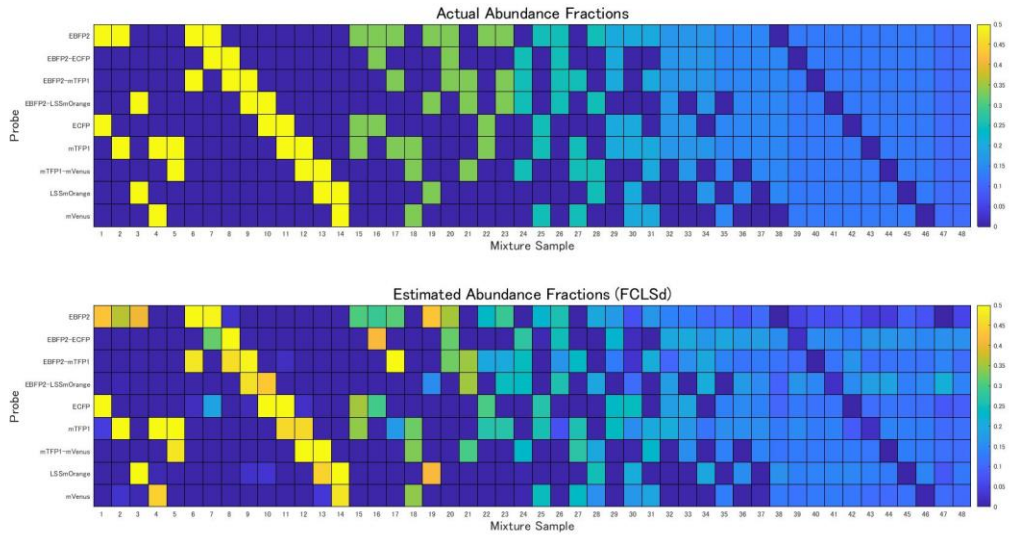

Figure 6.13. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on mixture samples

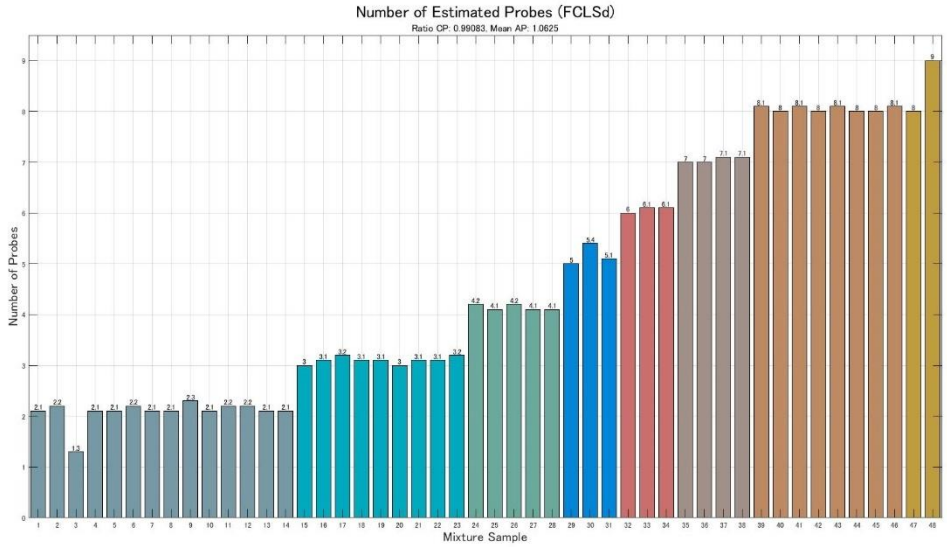

Figure 6.14. Bar graph of the numbers of estimated probes by direct FCLS on mixture samples

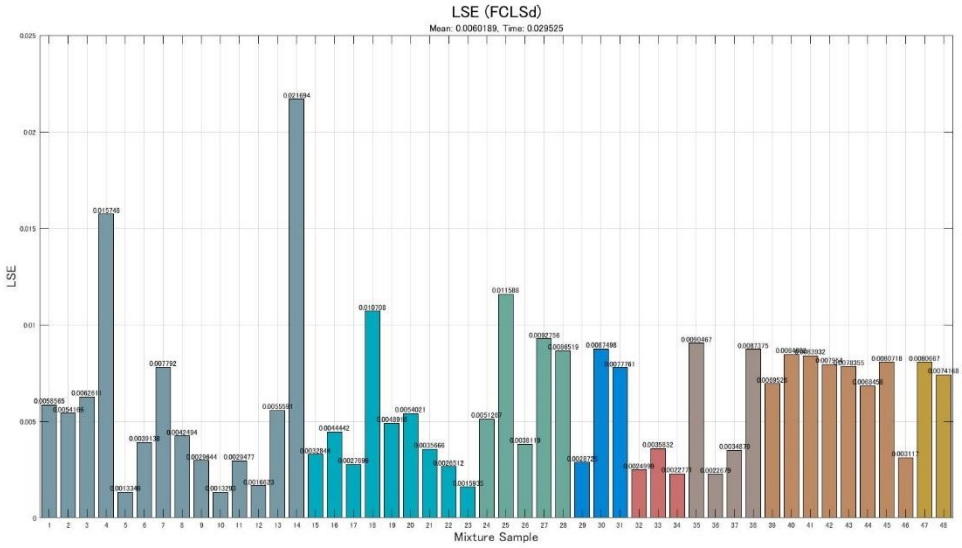

Figure 6.15. Bar graph of least square errors by direct FCLS on mixture samples

6.3.6 Iterative FCLS Linear Unmixing Method

The similar pattern in the EAF plot in Figure 6.16 is observed as in the direct FCLS method. In Figure 6.17, it yields the same rCP value 0.99083 with the smaller mAP value 0.875. Hence, the method finds the correct probes very well with less than one incorrect probe. There are indeed many integer-values scores in the figure. Figure 6.18 shows the average of LSE 0.0060422 and the processing time 0.035986s when we use the iterative FCLS method for unmixing. The average LSE of the iterative method is somewhat greater than the direct method. The unmixing process is however slower than the direct method.

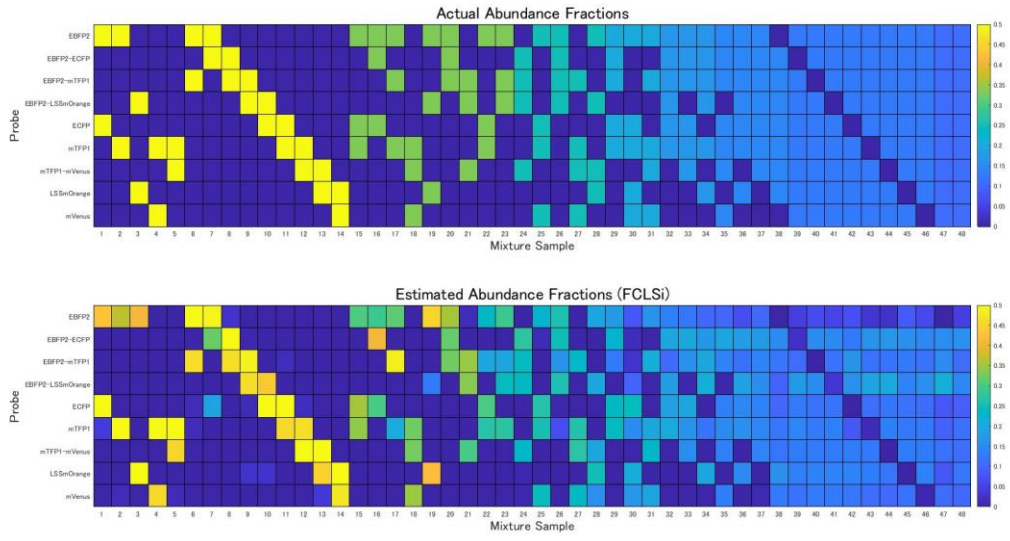

Figure 6.16. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by iterative FCLS (lower panel) on mixture samples

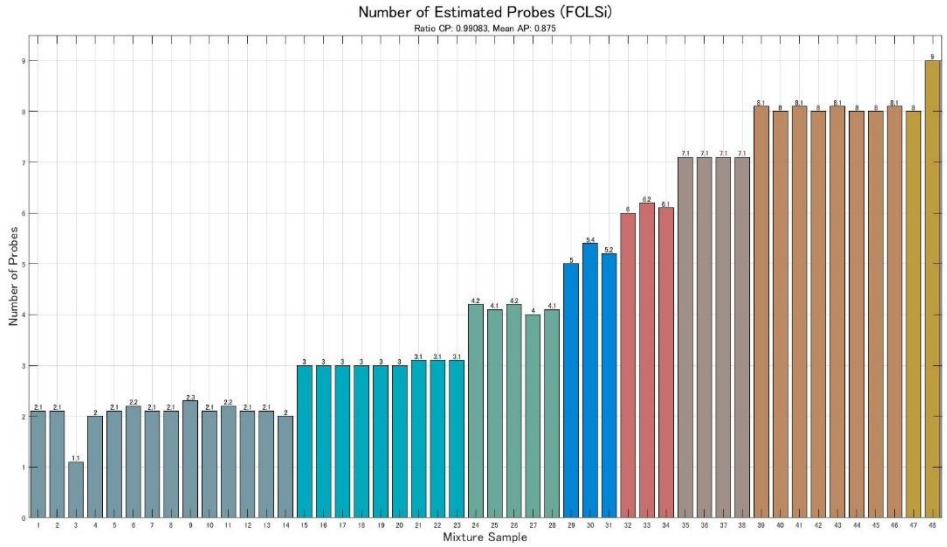

Figure 6.17. Bar graph of the numbers of estimated probes by iterative FCLS on mixture samples

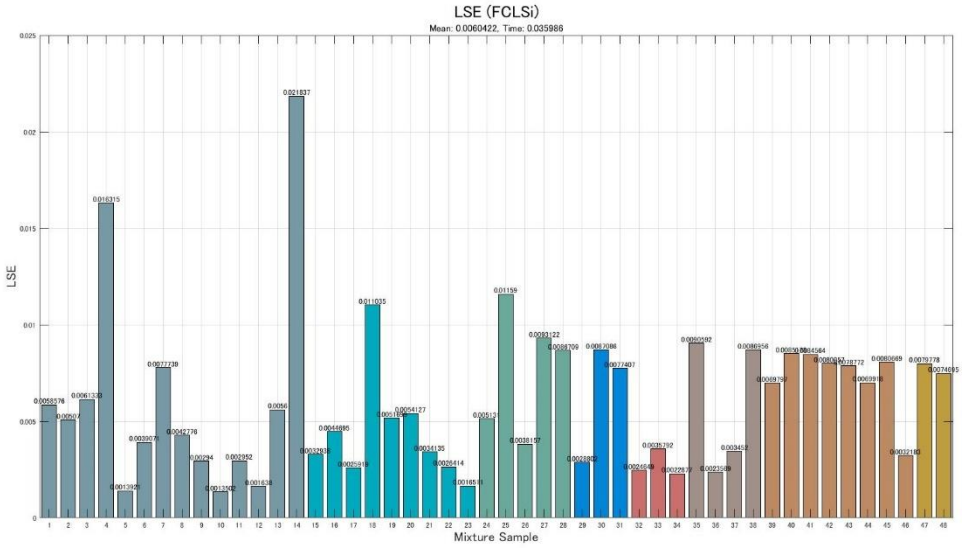

Figure 6.18. Bar graph of least square errors by iterative FCLS on mixture samples

6.3.7 Direct MFCLS Linear Unmixing Method

Compared to the FCLS methods, the EAF plot in Figure 6.19 contains more colourful squares. Indeed, we can see in Figure 6.20 that it has the same rCP value 0.99083 and a considerably large mAP value 2.1667, which implies that the method locates the necessary probes properly with two more unnecessary probes. As expected, there are only few integer-values scores in the figure. According to Figure 6.21, the direct SCLS method produces, on average, the LSE of 0.0060847, which is somewhat larger than the FCLS methods with the reduced unmixing time 0.007856s.

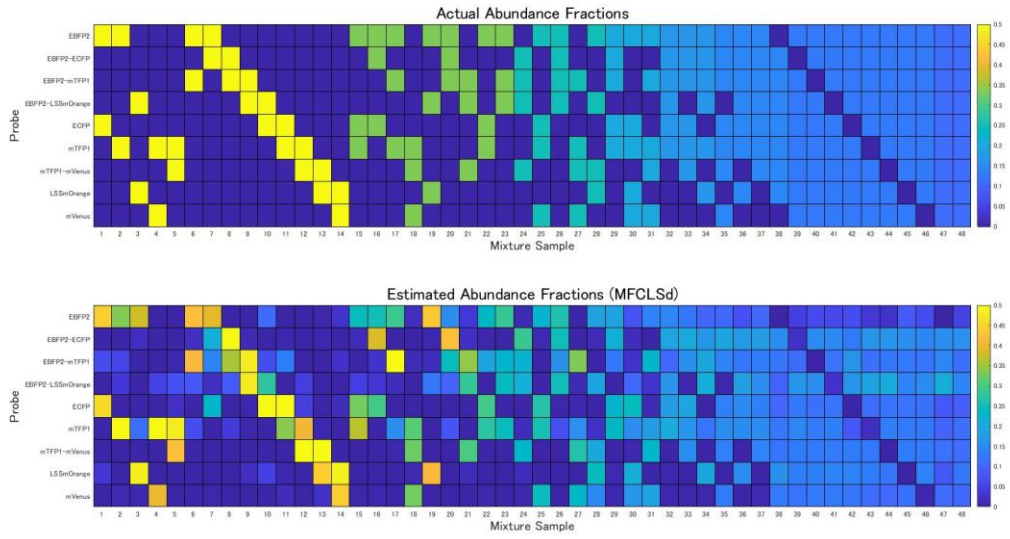

Figure 6.19. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct MFCLS (lower panel) on mixture samples

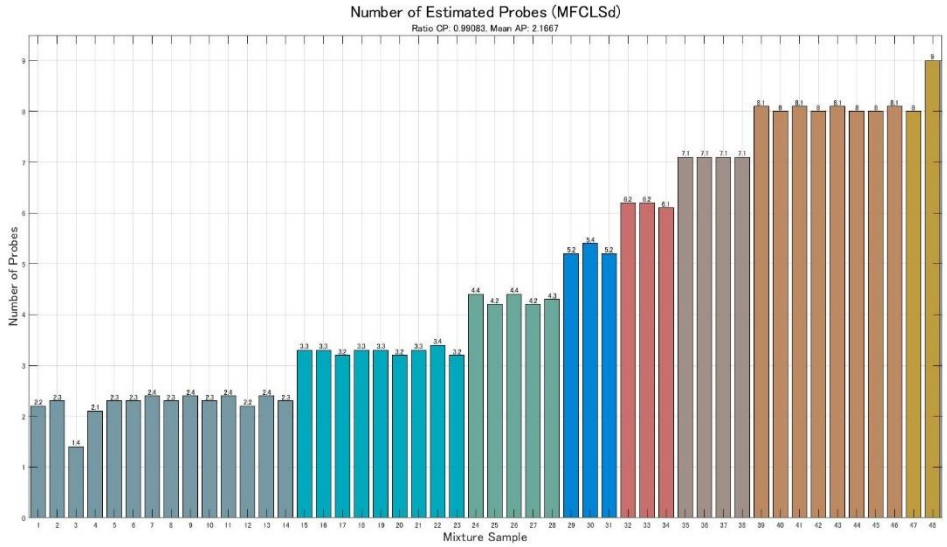

Figure 6.20. Bar graph of the numbers of estimated probes by direct MFCLS on mixture samples

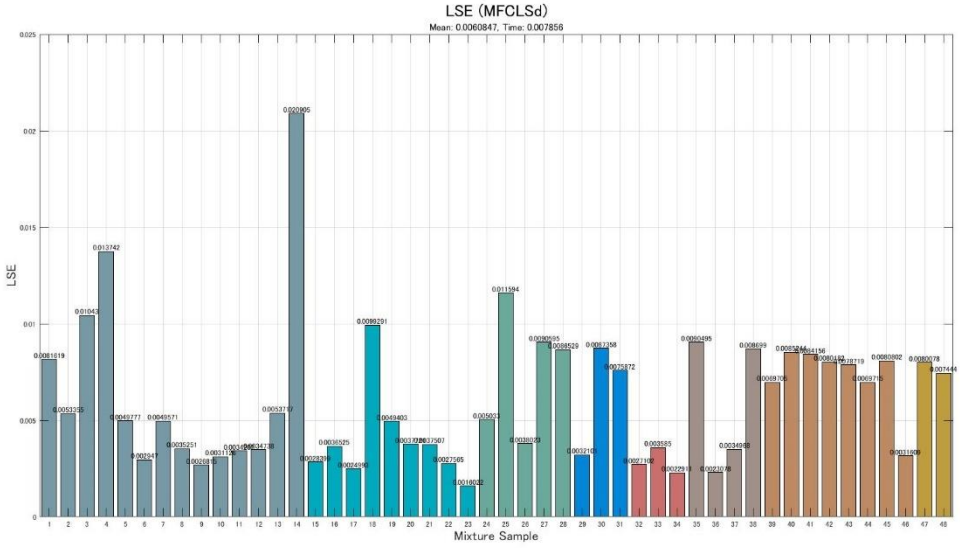

Figure 6.21. Bar graph of least square errors by direct MFCLS on mixture samples

6.3.8 Iterative MFCLS Linear Unmixing Method

In Figure 6.22, the squares in the EAF plot for the iterative MFCLS method are less colourful than the direct MFCLS method. The pattern is more similar to the FCLS methods. In Figure 6.23, we can observe that the rCP value, 0.99083, is still the same as the previous methods, but the mAP value, 0.875, is smaller. The method thus finds the correct probes with less than one incorrect probe, generating integer-valued scores in the figure. Figure 6.24 illustrates that the indirect MFCLS method yields the average LSE of 0.0060329, which is smaller than the direct MFCLS method with the longer processing time 0.041165s.

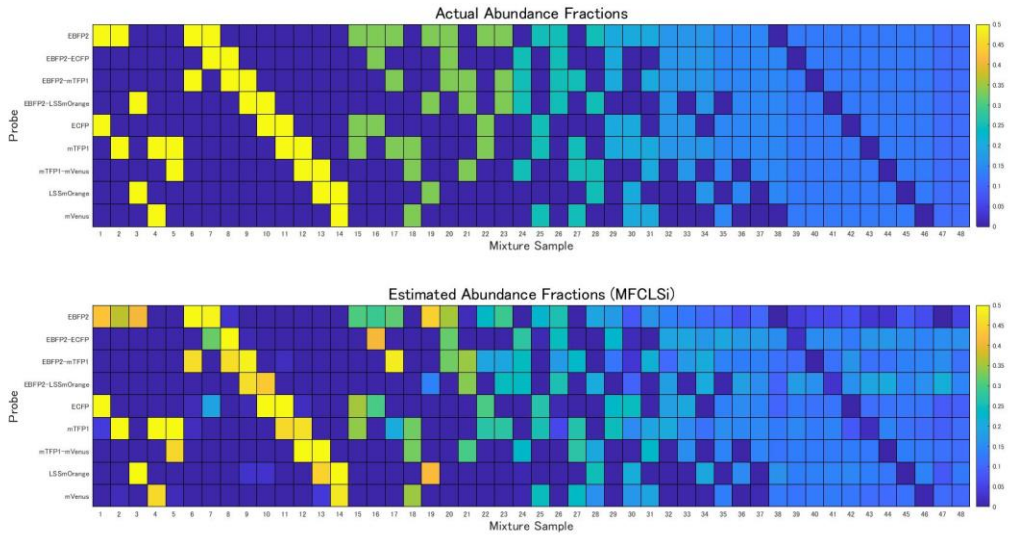

Figure 6.22. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by iterative MFCLS (lower panel) on mixture samples

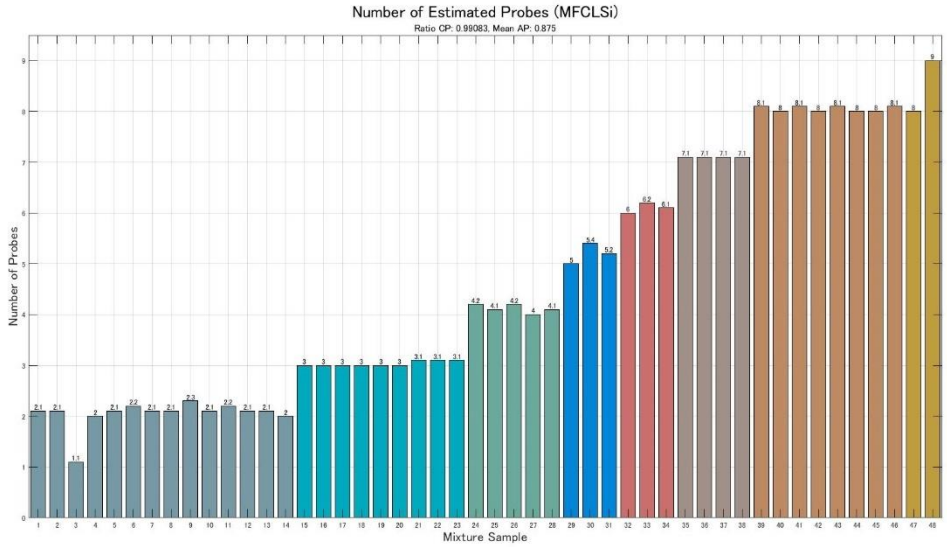

Figure 6.23. Bar graph of the numbers of estimated probes by iterative MFCLS on mixture samples

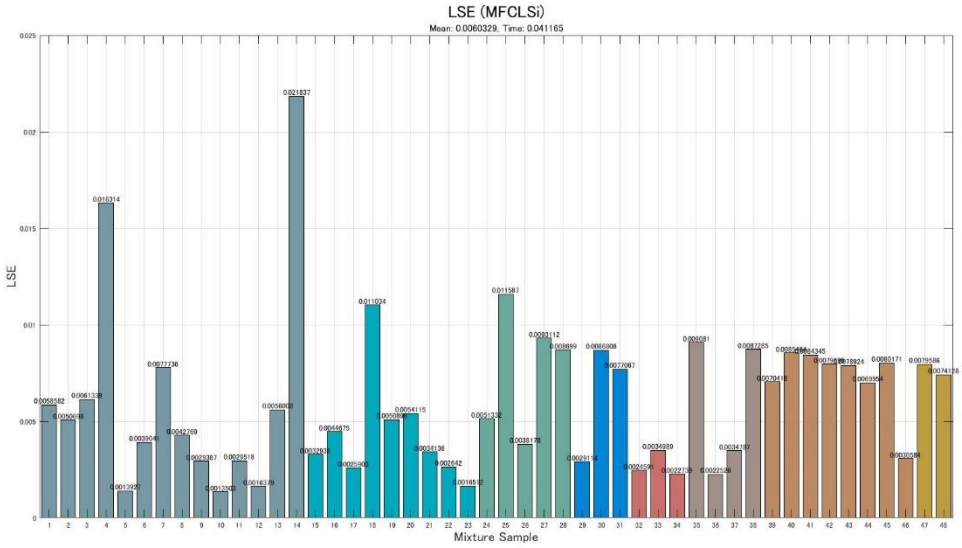

Figure 6.24. Bar graph of least square errors by direct MFCLS on mixture samples

6.3.9 Step Sizes for Iterative Methods

Tables 6.1 and 6.2 show the average LSE and unmixing time of the iterative SCLS and FCLS methods, respectively, when specific values are given for step size h. The average LSE and unmixing time are independent of the value of the step size in our data set. This is because the value of the scale variable $\lambda = 10000$ for the ASC is already very close to its appropriate value, demonstrating that it is a proper selection as proposed in [34], [35]. Thus, the choice of step size is not significant here, enabling us to select $h = 1$.

Step Size h	Average LSE	Time (s)
0.01	0.0063233	0.0080092
0.1		0.0081056
1		0.0080669
10		0.0079025
100		0.0079197

Table 6.1. Average LSE and unmixing time of iterative SCLS with step sizes

Step Size h	Average LSE	Time (s)
0.01	0.0060422	0.035986
0.1		0.034389
1		0.034612
10		0.036464
100		0.034581

Table 6.2. Average LSE and unmixing time of iterative FCLS with step sizes

In Table 6.3, the average LSE and unmixing time of the iterative MFCLS method are dependent of the selection of step size h_2 which is a parameter for the AASC, but they are

independent of step size h_1 for the ASC as shown above. Moreover, the elapsed time generally decreases as the average LSE increases, but there are no significant differences among the times. Therefore, we can choose $h_1 = h_2 = 1$ as proper step sizes for the iterative MFCLS method.

Step Size h_2	Average LSE	Time (s)
0.01	0.0060352	0.039613
0.1	0.0060329	0.041477
1	0.0060329	0.041165
10	0.0060328	0.041848
100	0.0060399	0.038819

Table 6.3. Average LSE and unmixing time of iterative MFCLS with step sizes

Overall, considering both average LSE and processing time, we conclude that 1 is an appropriate selection as a step size for iterative methods from the experiments.

6.3.10 Conclusion

Figure 6.25 summarizes the rCP and mAP values of the LS methods described in Chapters 2. As the rCP value is close to 1, the linear unmixing algorithm finds correct probes well (with high reliability) for a mixture sample. In this work, the value can be interpreted as a probability of locating correct probes. Thus, if it has a low rCP value, it shows poor performance of unmixing. Interestingly, all of the methods have the rCP value close to 0.99; they can locate correct probes with probability of 99%. The mAP denotes the average number of unnecessary probes located by the method. As the mAP increases, the method finds more unnecessary probes reducing its performance level. We observe that the mAP values of the ULS method, the SCLS methods, and the

direct MFCLS method are approximately 2, whereas those of the other methods are nearly 1. Hence, we expect that the former will show relatively worse performance than the latter.

Figure 6.26 agrees that the NCLS method, FCLS methods, and the indirect MFCLS method are the four highest-LSE methods. Also, we notice the outstanding difference in the average LSE between the SCLS methods and the NCLS method. Therefore, the ANC is more significant than the ASC in obtaining optimal solutions in our system. The direct FCLS method yields a slightly smaller LSE on average than the NCLS method since both ANC and ASC are imposed on the FCLS method. Thus, we can obtain more accurate solutions if more constraints are imposed. Although the direct MFCLS method is theoretically supposed to locate a more accurate solution than the direct FCLS method as its solution is obtained analytically from the ASC and AASC (and thus ANC), its average LSE is greater than that of the direct FCLS method because its algorithm produces suboptimal solutions by taking advantage of the solution obtained from the SCLS method alternatively. For iterative methods, however, the average LSE decreases as the algorithm becomes more constrained as we expected. Also, except the MFCLS methods, each direct method produces a smaller LSE than its corresponding indirect method.

We observe that, for the direct methods, the NCLS method and the FCLS method require more time for unmixing than the ULS method and the SCLS method since they perform dimensionality reduction on the mixing matrix for the ANC, which is computationally expensive. However, the MFCLS method is faster since it uses the AASC for the ANC to obtain the analytical solution using the Lagrange multiplier. Also, in general, the more constraints the LS algorithm handles, the slower the unmixing process is. The direct MFCLS method however shows an exceptional result due to its analytical approach. Furthermore, each direct method is faster than its

indirect analogue. This is attributed to the fact that the indirect methods create new matrices iteratively during the unmixing procedures, significantly increasing a computational cost. For the FCLS methods, both methods conduct matrix manipulations (that is, dimensionality reduction for the direct method and iterative matrix generation for the indirect method), but generating matrices is generally more computationally expensive since it is repeated until the loop terminates, while the maximum number of repetitions of dimensionality reduction is based on the total number of probes (in this work, less than or equal to 9 times).

In summary, the direct FCLS method is the best unmixing algorithm with respect to LSE. However, considering its processing time, we conclude that the direct MFCLS method shows the best performance among the LS methods.

There are two results contrary to our expectations: step sizes for the iterative methods and the processing time of the indirect methods; indeed, the smallest step size yields more average LSE, and the indirect methods are slower than the direct methods. We believe that these unexpected results may be attributed to our unoptimized algorithms and therefore optimization for the indirect unmixing algorithms should be performed sufficiently to improve those results.

For the direct MFCLS method, we used the solution of the direct SCLS method alternatively, which leads to more average LSE. To our best knowledge, such an alternative is often employed for simplicity [23] and we cannot find a method to solve such an issue.

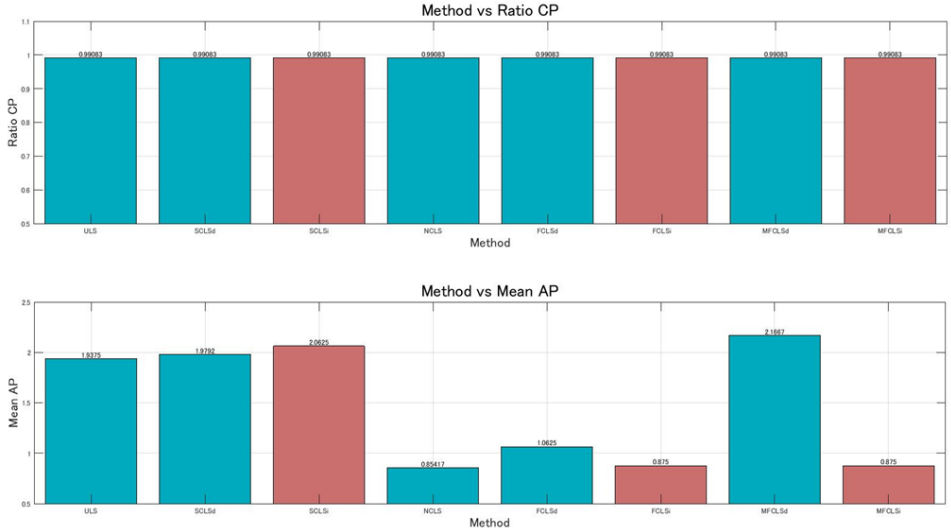

Figure 6.25. Bar graphs of ratios of detected correct probes (upper panel) and average detected incorrect probes (lower panel) by LS methods

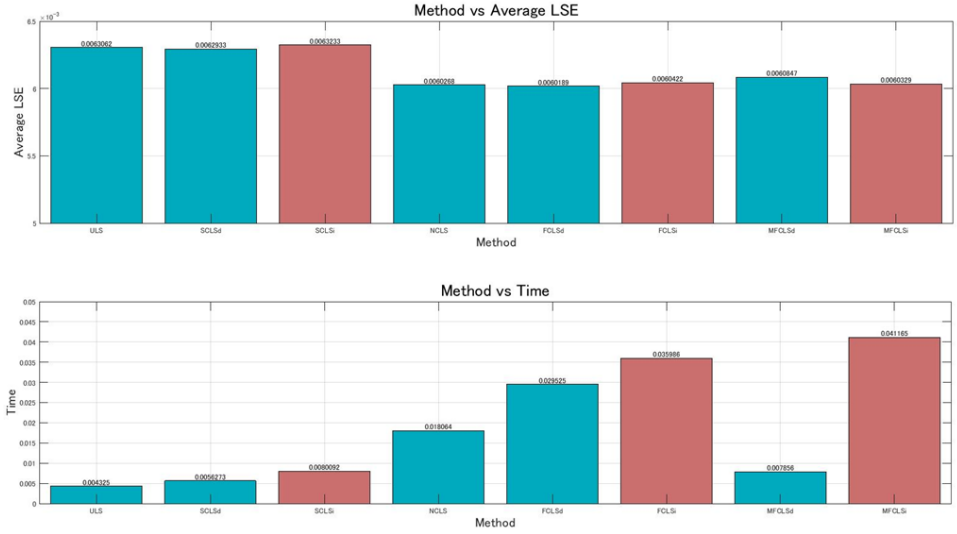

Figure 6.26. Bar graphs of average least square errors (upper panel) and processing time (lower panel) by LS methods

6.4 Experiment 2

In this experiment, we will perform the performance analysis of the optimization techniques for linear unmixing discussed in Chapters 3 and 4. For all of the optimization methods, we will use 10^5 and 10^{-6} as the maximum number of iterations and error tolerance, respectively. We will set the step size for the GD methods to 200. For the GD and NM methods, the starting point is set to $\overrightarrow{1/9}$ whose entries are all $1/9$. On the other hand, random initial points are generated for the TLBO and TLSBO methods because they are a stochastic population-based algorithms. Also, the respective values 10 and 100 are given to them for the size of population and maximum point.

6.4.1 FC-GD Method

There is no noticeable difference in the EAF plot in Figure 6.27 than the previous EAF plot for the iterative MFCLS method. In Figure 6.28, however, the rCP value has been increased to 1 and the mAP value to 1.25. The method finds the necessary probes and may locate one or more unnecessary probes; indeed, it successfully finds all the correct probes even on mixture samples 3 and 47, while the other methods cannot. Figure 6.29 shows that the FC-GD method produces the average LSE of 0.0060104, which is smaller than all the linear unmixing methods in Chapter 2. The method however requires much more unmixing time, 2.5822s.

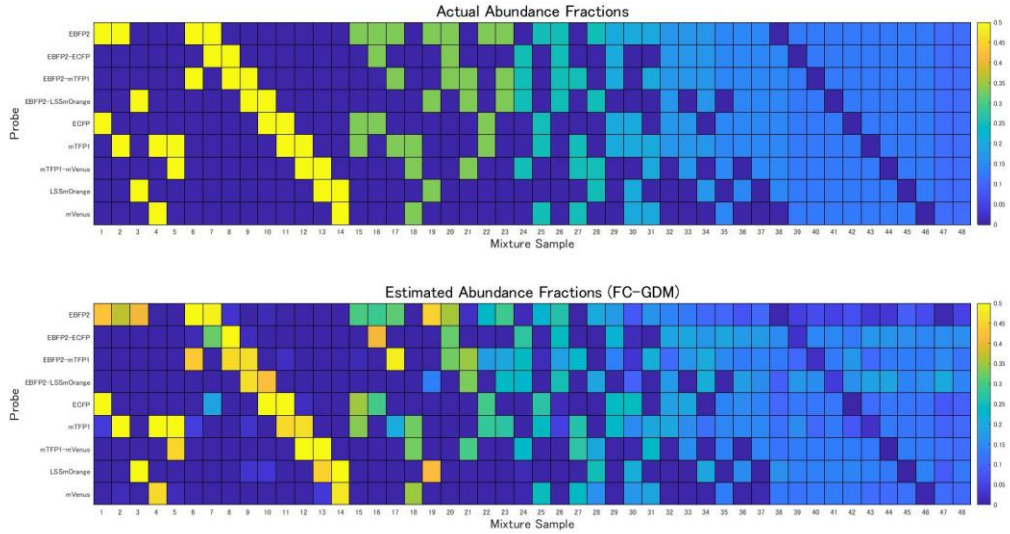

Figure 6.27. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by FC-GD (lower panel) on mixture samples

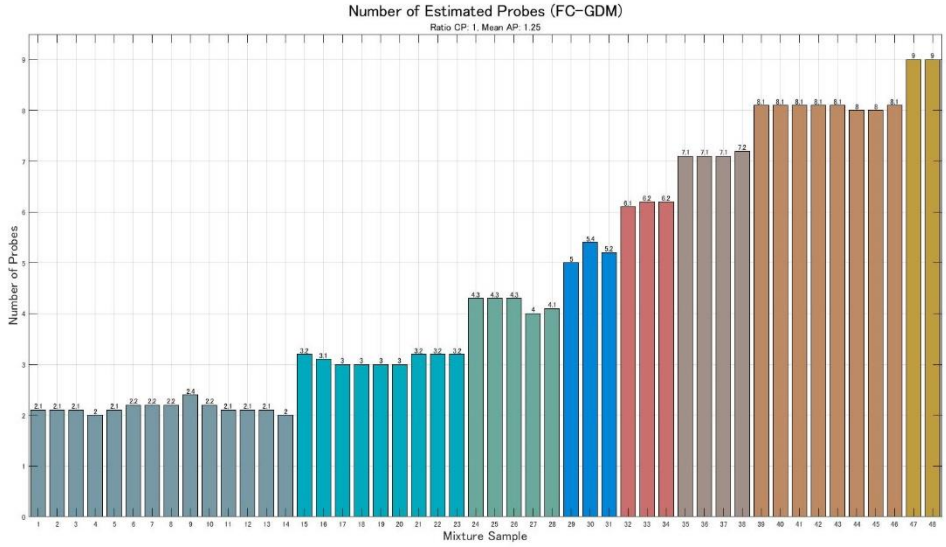

Figure 6.28. Bar graph of the numbers of estimated probes by FC-GD on mixture samples

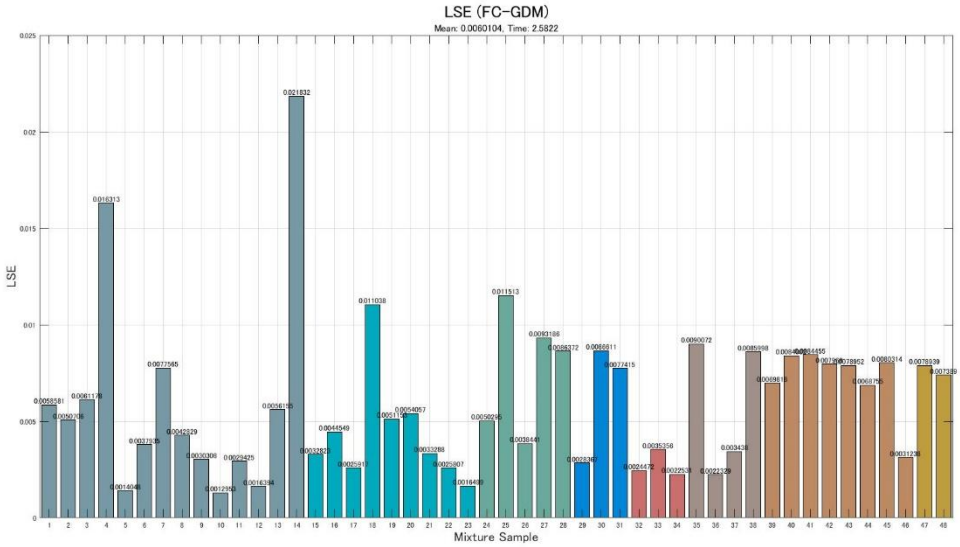

Figure 6.29. Bar graph of least square errors by FC-GD on mixture samples

6.4.2 GD Method with Bounding Process

The EAF plot in Figure 6.30 shows a similar pattern to the FC-GD method. In Figure 6.31, we can see that the rCP is 0.99083 and the mAP is 0.85417, which are smaller than the FC-GD method, implying that it locates the correct probes very well with less than one incorrect probe. Figure 6.32 dictates that the GD method yields, on average, the LSE of 0.0060244, which is slightly larger than the FC-GD method. The GD method takes 0.27807s and thus it is much faster than the FC-GD method.

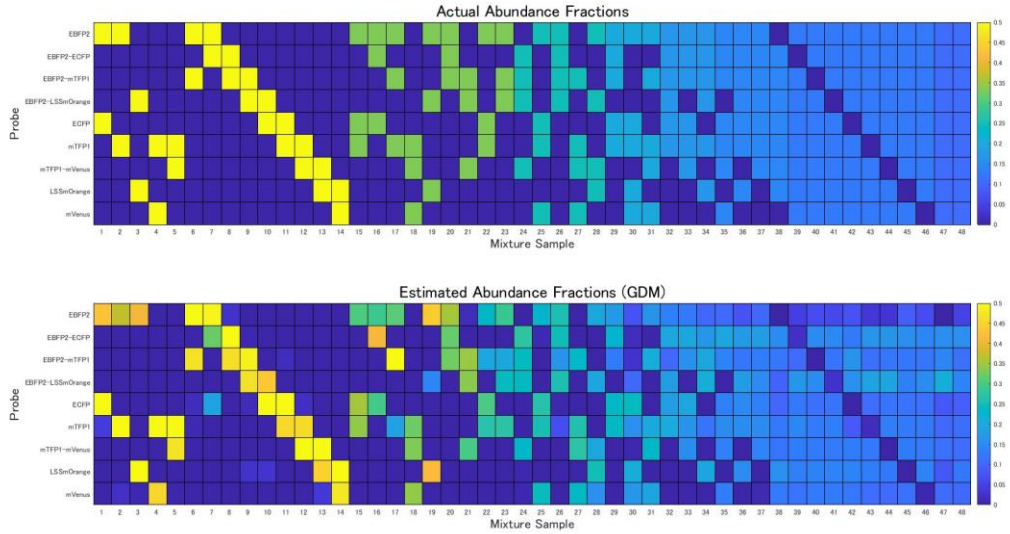

Figure 6.30. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by GD (lower panel) on mixture samples

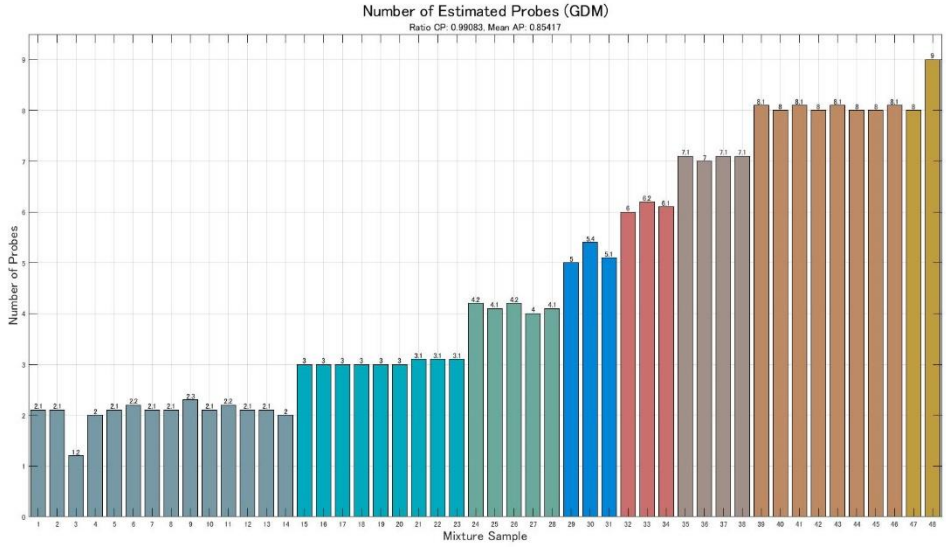

Figure 6.31. Bar graph of the numbers of estimated probes by GD on mixture samples

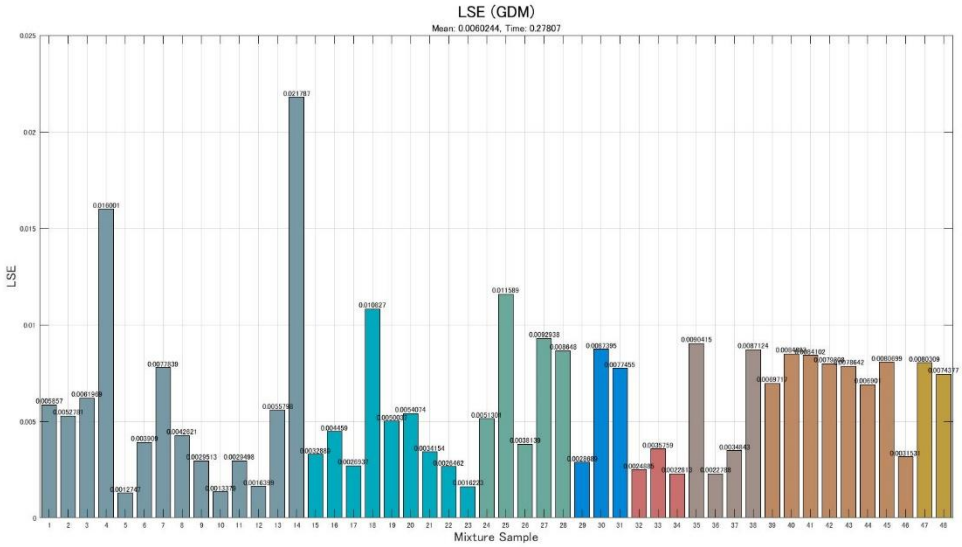

Figure 6.32. Bar graph of least square errors by GD on mixture samples

6.4.3 Standard NM Method

In Figure 6.33, the colour pattern in the EAF plot is very similar to the GD methods. Figure 6.34 illustrates that the rCP is 0.99083 and the mAP is 0.91667, indicating that it finds the necessary probes very well with approximately one unnecessary probe. According to Figure 6.35, the average LSE, 0.0059838, becomes somewhat smaller and the unmixing time, 31.8805s, becomes much longer, compared to the GD methods.

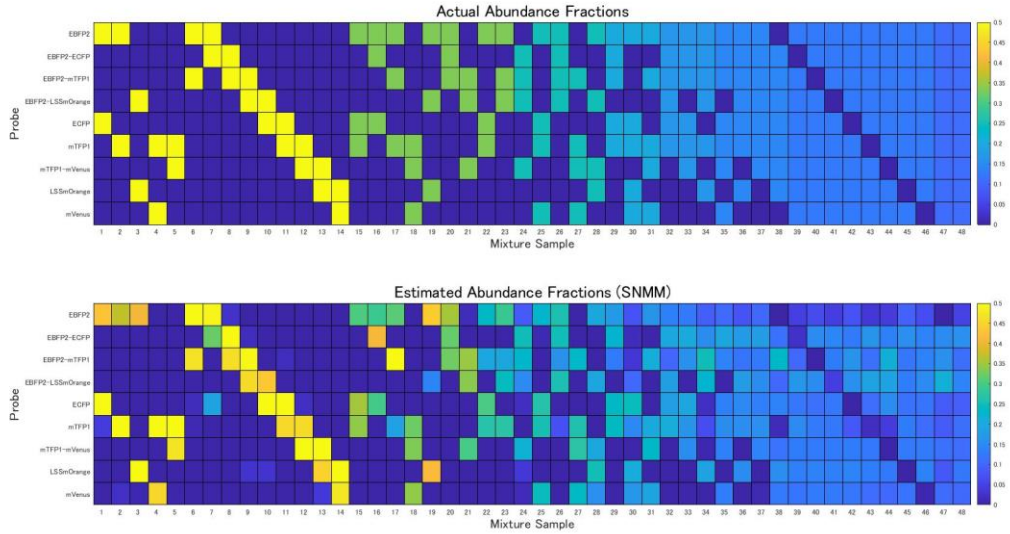

Figure 6.33. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by standard NM (lower panel) on mixture samples

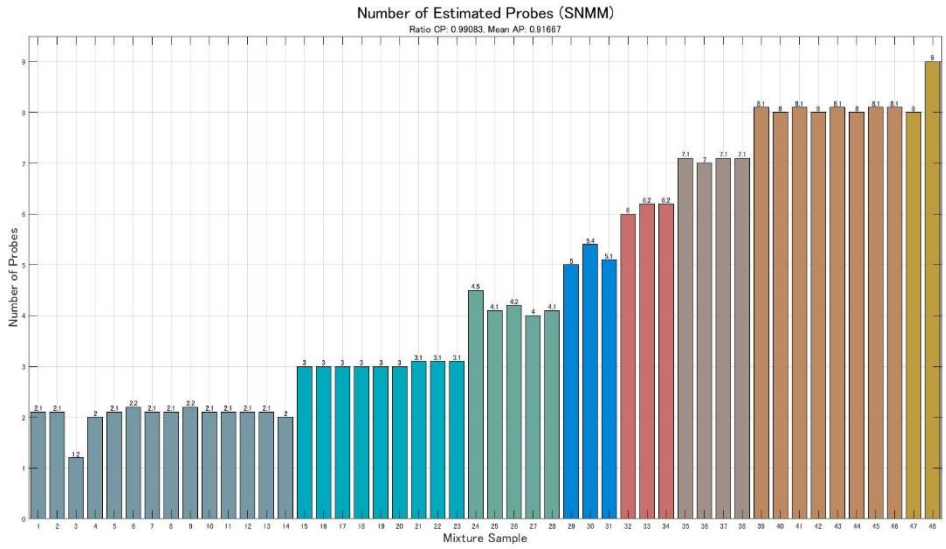

Figure 6.34. Bar graph of the numbers of estimated probes by standard NM on mixture samples

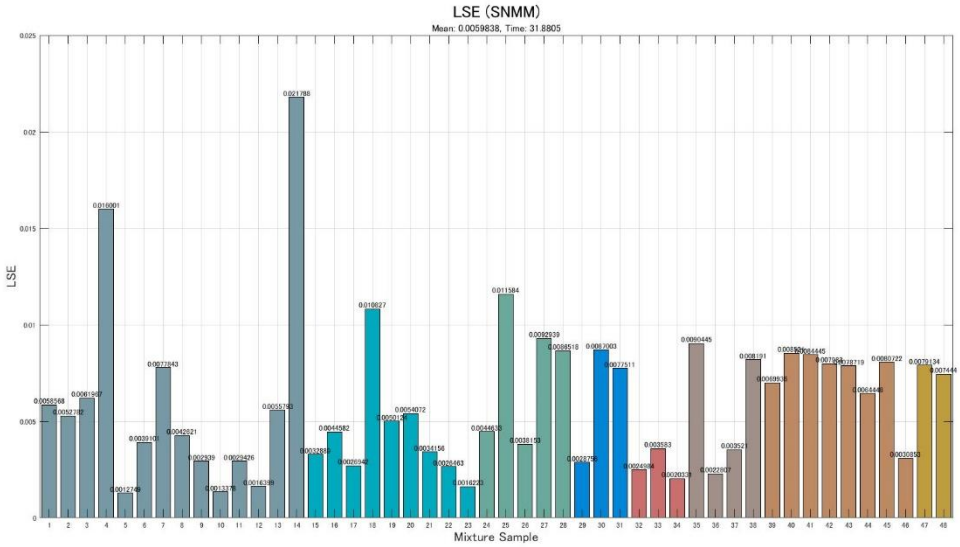

Figure 6.35. Bar graph of least square errors by standard NM on mixture samples

6.4.4 Adaptive NM Method

Figure 6.36 illustrates high similarity of the colour pattern between the standard NM method and the adaptive NM method in the EAF plot. We can see that the rCP is 0.9633 and the mAP is 1.0417 in Figure 6.37; the method can locate the correct probes fairly well with approximately one additional incorrect probe. Indeed, it cannot find correct probes for mixture samples 3, 16, 36, 38, 40, and 48. Figure 6.38 shows that the average LSE is 0.0059733 with unmixing time 43.2687s. Compared to the standard method, the LSE becomes slightly smaller, but the processing time becomes longer.

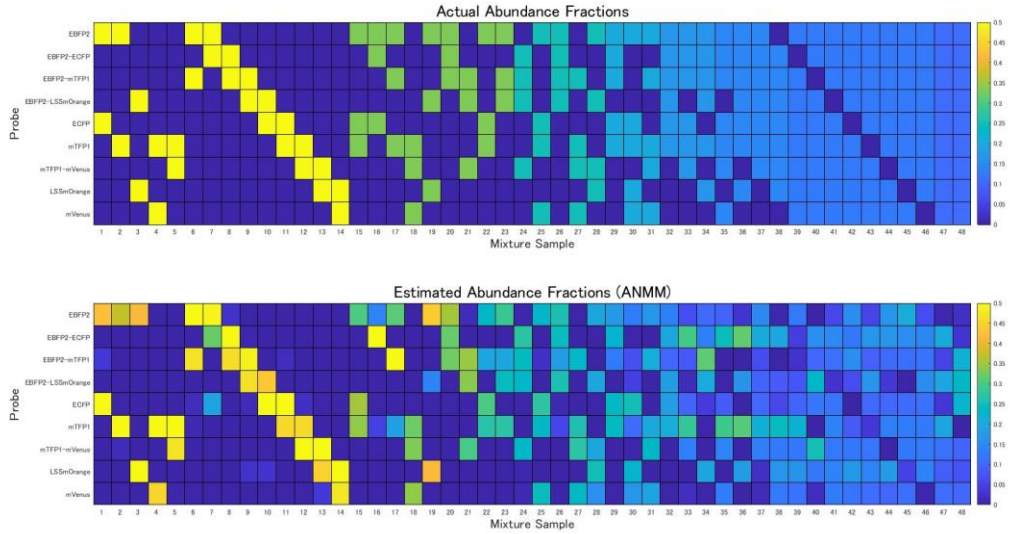

Figure 6.36. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by adaptive NM (lower panel) on mixture samples

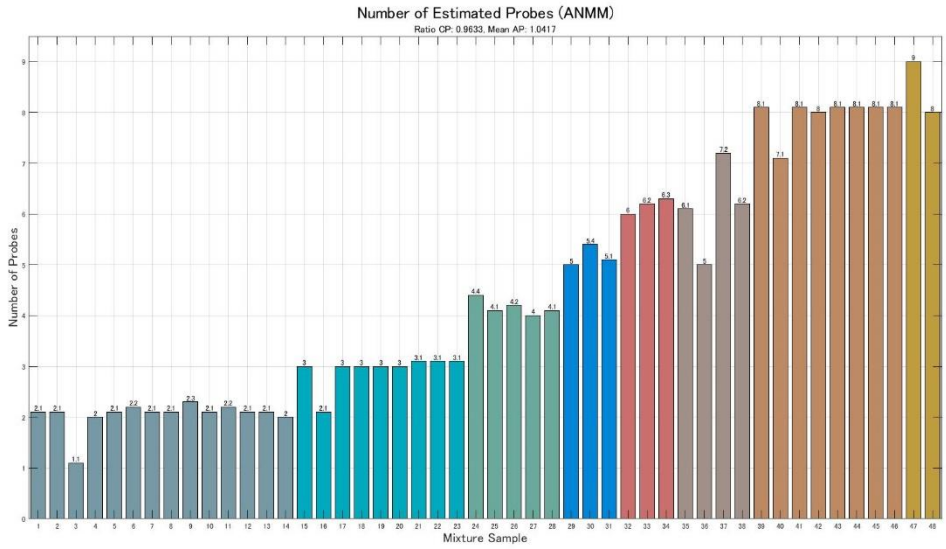

Figure 6.37. Bar graph of the numbers of estimated probes by adaptive NM on mixture samples

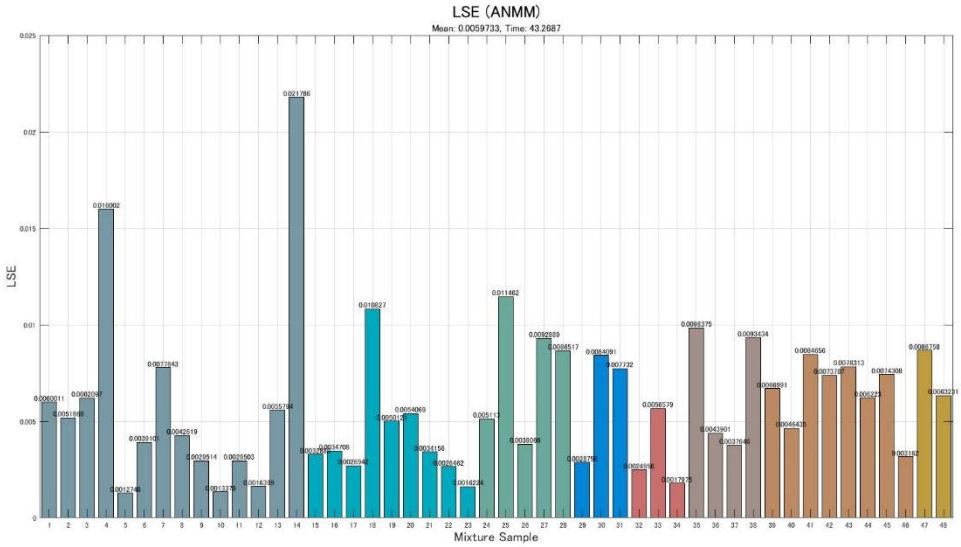

Figure 6.38. Bar graph of least square errors by adaptive NM on mixture samples

6.4.5 TLBO Method

The EAF plot of the TLBO method in Figure 6.39 shows no remarkable difference from those of the NM methods. Indeed, the rCP and the mAP are 0.9633 and 0.85417 in Figure 6.40, which implies that the method can find the necessary probes properly with less than one unnecessary probe. We can observe that it fails to find all the correct probes for mixture samples 3, 9, 36, 38, 40, 42, 44, and 47. According to Figure 6.41, the average LSE and the unmixing time are 0.005989 and 51.4505s. Both values become larger than the adaptive NM method.

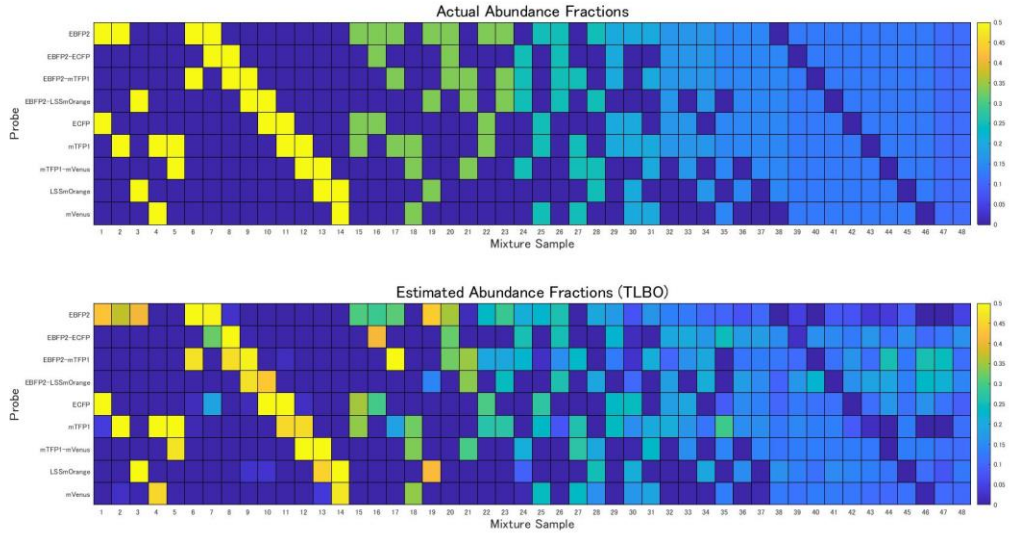

Figure 6.39. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by TLBO (lower panel) on mixture samples

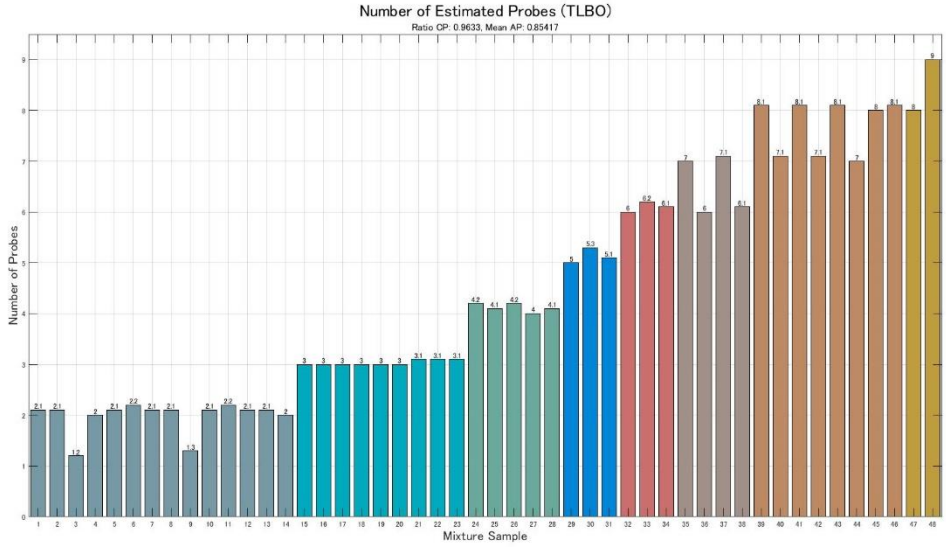

Figure 6.40. Bar graph of the numbers of estimated probes by TLBO on mixture samples

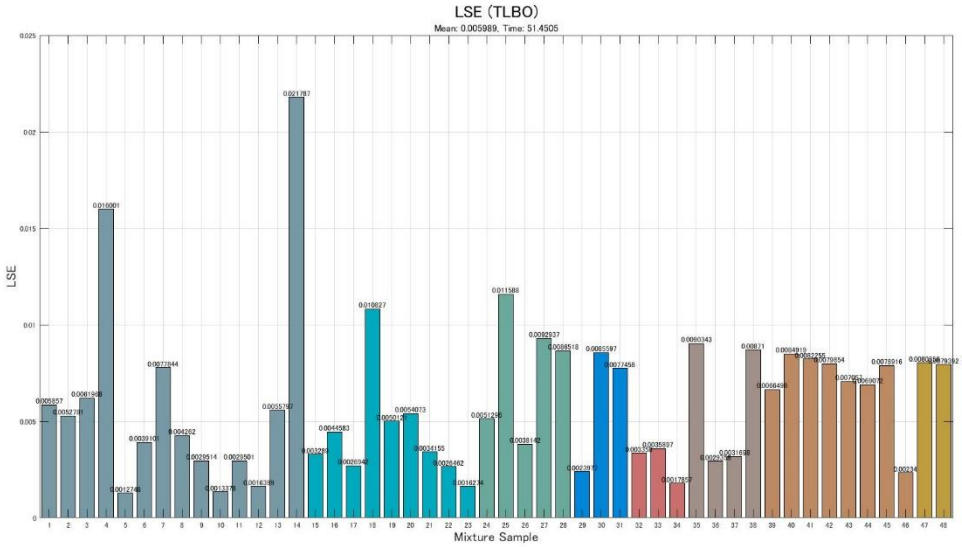

Figure 6.41. Bar graph of least square errors by TLBO on mixture samples

6.4.6 TLSBO Method

In Figure 6.42, there is high similarity of the colour pattern in the EAF plot between the TLBO method and the TLSBO method. Figure 6.43 shows that the rCP is 0.97248 and the mAP is 0.83333, indicating that the method locates the correct probes quite well with less than one incorrect probe. As observed in the figure, it does not find the correct probes on mixture samples 3, 32, 39, and 47. Figure 6.44 illustrates that the method produces the average LSE of 0.0059749 and requires 32.0968s for unmixing. We can see that these values have been decreased, compared to the TLBO method.

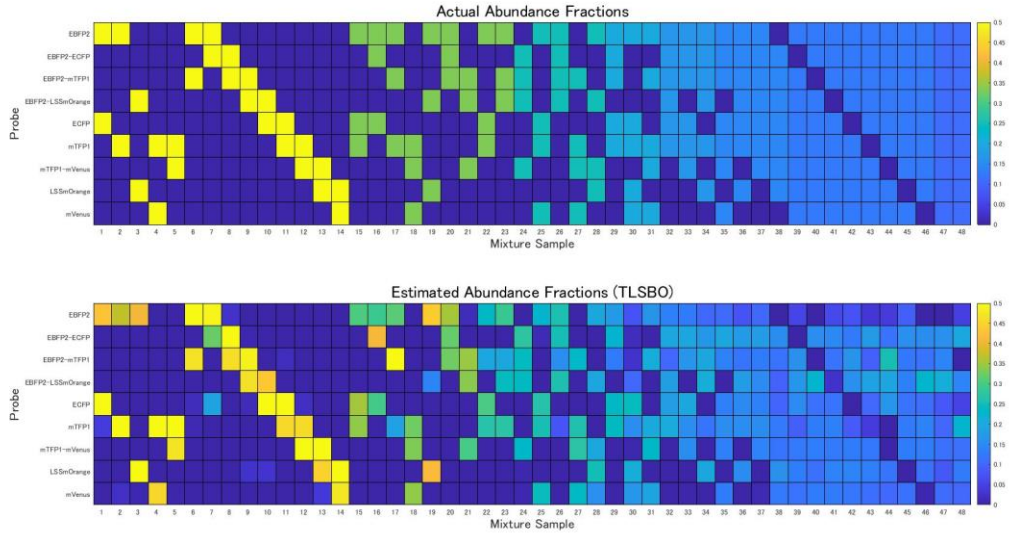

Figure 6.42. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by TLSBO (lower panel) on mixture samples

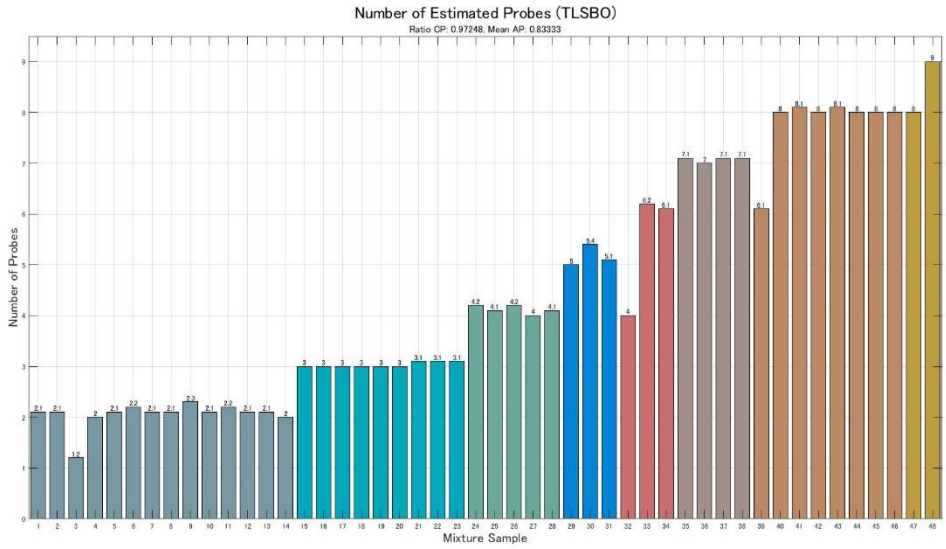

Figure 6.43. Bar graph of the numbers of estimated probes by TLSBO on mixture samples

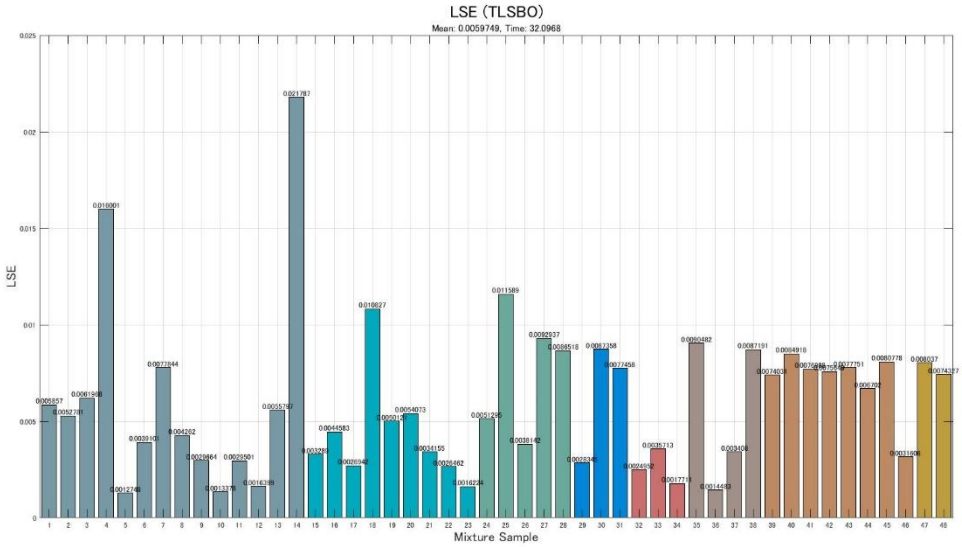

Figure 6.44. Bar graph of least square errors by TLSBO on mixture samples

6.4.7 Conclusion

Figure 6.45 summarizes the rCP and mAP values of the optimization methods described in Chapters 3 and 4. In the case of the FC-GD method, it finds all the correct probes for each mixture sample. The adaptive NM method, the TLBO method, and the TLSBO method produce relatively lower rCP values (0.9633, 0.9633, and 0.9724, respectively). They can find correct probes with probability of more than 96%. Also, all of the methods have the mAP values close to 1. From these results, we expect that the optimization methods will show relatively better performance than the LS methods with respect to LSE. Indeed, the optimization techniques generally yield less average LSE than the LS methods. We believe that such better performance is attributed to the fact that our linear unmixing problem is a strictly convex problem which has a unique local optimum. As aforementioned, the FC-GD method can find an optimal solution since the ASC and ANC are

imposed on the GD algorithm. However, we can use the GD method with the bounding process due to strict convexity. In this case, both GD methods converge to the same solution and hence they produce the approximately same LSE. For the NM methods, the adaptive method decreases the chances of using reflection steps and circumventing the rapid reduction in the simplex diameter by choosing adaptively the parameters for expansion, contraction, and shrinkage. Therefore, it generally finds a more optimal solution than the standard NM method. Since our dataset is not considered as high dimensions (that is, 9 dimensions), no significant improvement was found in terms of accuracy. The TLBO performs global and local searches using repetitive stochastic procedures in its algorithm, which makes it computationally expensive. To maximize the power of the global and local searches, the TLSBO gives some random changes to the coordinates of each member. Thus, the TLBO and TLSBO find the same solution (they yield the approximately same LSE), but TLSBO shows a better convergence rate to the solution.

Figure 6.46 illustrates the unmixing times of the optimization methods. The GD method with the bounding process is faster than the FC-GD method due to the simplicity of its algorithm; the FC-GD algorithm is more complicated since the ASC and the ANC are applied to the update equation. However, the use of the GD method is more restricted because it requires two conditions (that is, an overdetermined linear system model with mixing matrix consisting of linearly independent columns), whereas the FC-GD method can be employed in any situations if its initial point satisfies the sum-to-one property. We observe that the adaptive NM method is much slower than the standard NM method, because the adaptive method relieves the effect of dimensionality and thus requires more iterations for convergence. The TLBO method is typically a slow global optimization algorithm due to its stochastic procedures. Thus, the TLSBO algorithm improves the

speed of convergence by adding the studying phase into the TLBO algorithm. During the phase, it gives changes to the positions of each member, resulting in faster convergence to the global solution. It is notable that the NM methods and TLBO methods require much more time than the GD methods as they are global optimization algorithms that have higher computational complexity, while the GD methods are local optimization algorithms with less computational complexity.

Even though the global optimization methods (that is, the NM, TLBO and TLSBO methods) can locate as optimal solutions as the local optimization methods (that is, the FC-GD and GD methods), they require more processing time due to their algorithmic complexity. Also, the GD method is faster than the FC-GD method, but in order to use it, the linear unmixing problem necessarily satisfies two conditions: the mixing matrix forms an overdetermined system, and its columns are linearly independent. Therefore, we conclude that the FC-GD method is the best linear unmixing algorithm among the optimization methods.

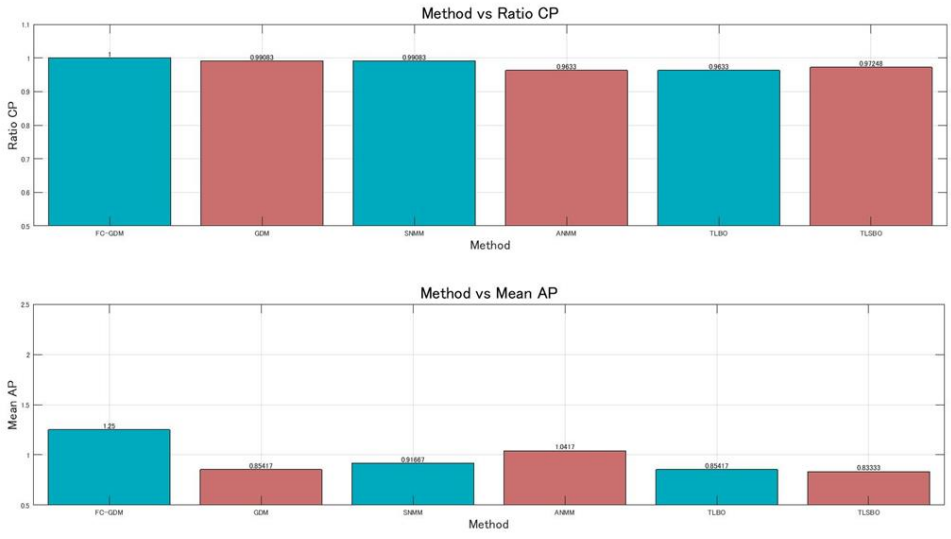

Figure 6.45. Bar graphs of ratios of detected correct probes (upper panel) and average detected incorrect probes (lower panel) by optimization methods

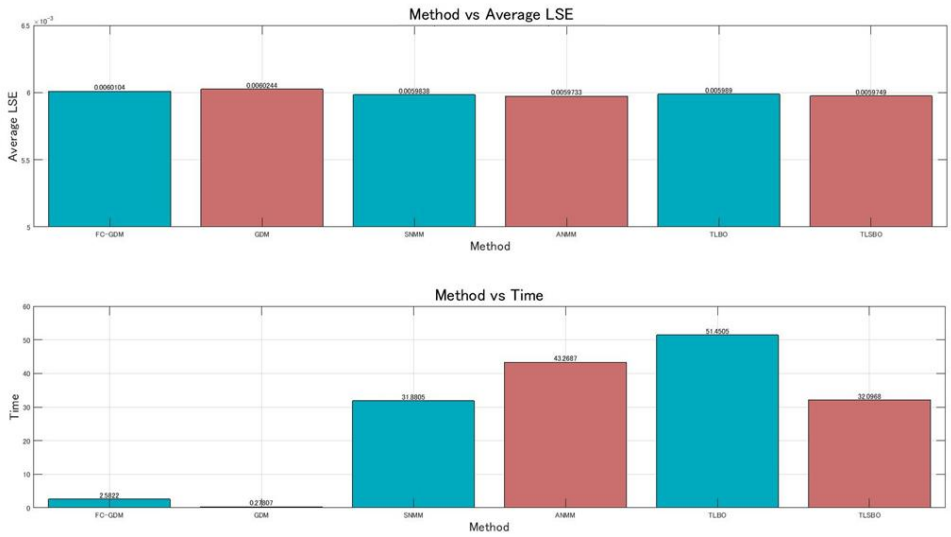

Figure 6.46. Bar graphs of average least square errors (upper panel) and processing time (lower panel) by optimization methods

6.5 Experiment 3

In this experiment, we investigate the influence of additive uniformly random noise on abundance recovery. We will examine how significantly the noise can affect the performance of the linear unmixing methods. For this purpose, we will generate three random noises with different ratios, add them to each mixture emission spectrum (See Figure 6.47), and attempt to unmix the noisy spectra with the direct FCLS method. We will then denoise the noisy spectra using the Fourier-based and wavelet-based algorithms discussed in Chapter 5 to compare the performance measurements on the filtered spectra and on the real spectra. To compare the level of each spectrum to the level of background noise, we compute signal-to-noise ratio (SNR), defined as the ratio of

signal power to the noise power (See Appendix C).

6.5.1 Selection of Parameters for Denoising Algorithms

By visual inspection, we will choose the Fejér-Korovkin wavelet of order 6 (fk6) as a mother wavelet for the wavelet-based denoising method (See Figure 6.48), which is defined as

$$F_n(x) = \frac{1}{n+1} \frac{\sin^2 \frac{(n+1)x}{2}}{\sin^2 \frac{x}{2}}$$

for order n [71]. For a decomposition level, we will select level 5 as in [70].

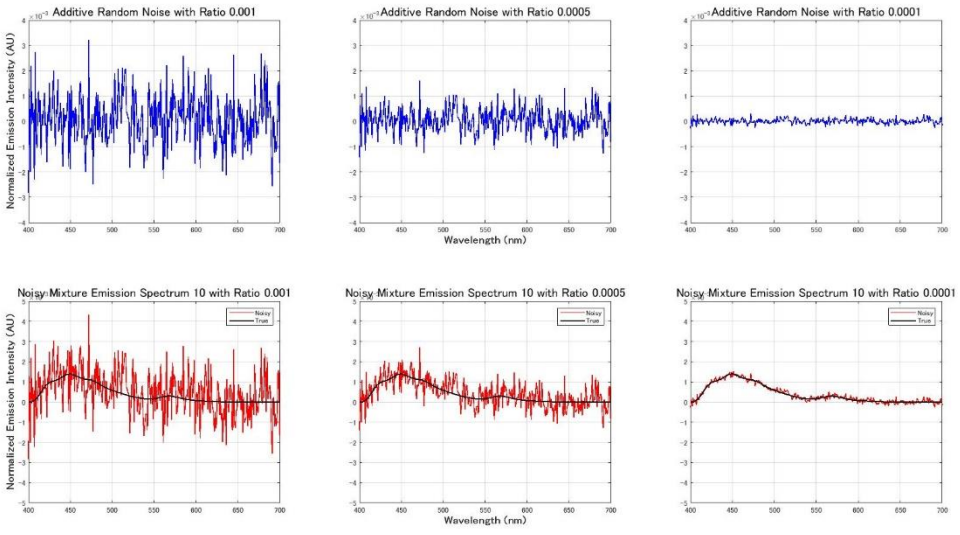

Figure 6.47. Plots of additive random noises with ratio 0.001, 0.0005, and 0.0001, respectively, and their corresponding noisy mixture samples

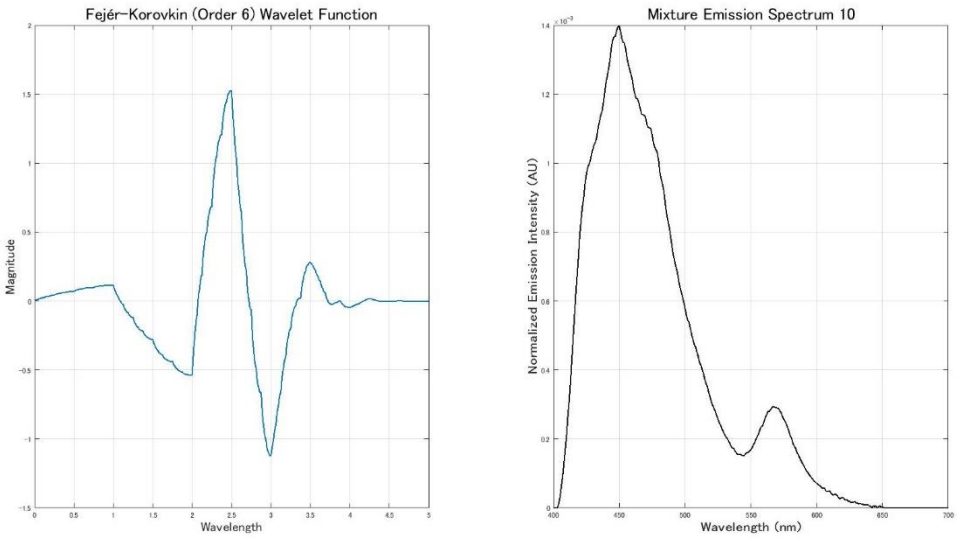

Figure 6.48. Plots of mother wavelet FK6 and mixture sample 10

6.5.2 Linear Unmixing on Noisy Mixture Samples with Ratio 0.001

We observe a polychrome pattern with many dark blue squares in the EAF plot in Figure 6.49. Specifically, the method does not find EBFP2 and mTFP1 for mixture samples 29 to 48.

In Figure 6.50, its rCP and mAP values are 0.80734 and 0.8125, respectively. Compared to the original mixture samples, their values have been decreased. In Figure 6.51, the average LSE is 0.056128 and the processing time is 0.037315s. The noisy mixture samples yield more average LSE than the original mixture samples with more unmixing time.

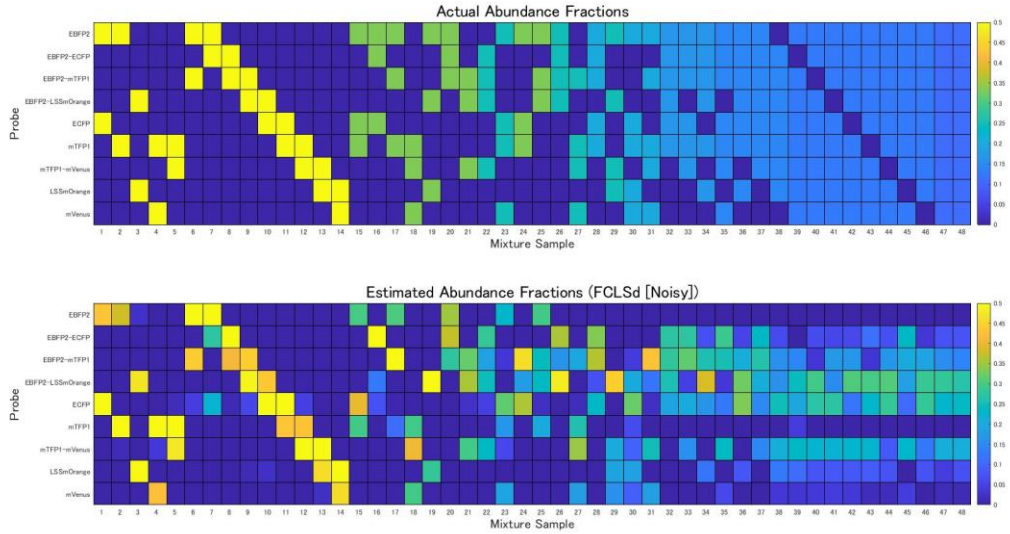

Figure 6.49. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on noisy mixture samples with ratio 0.001

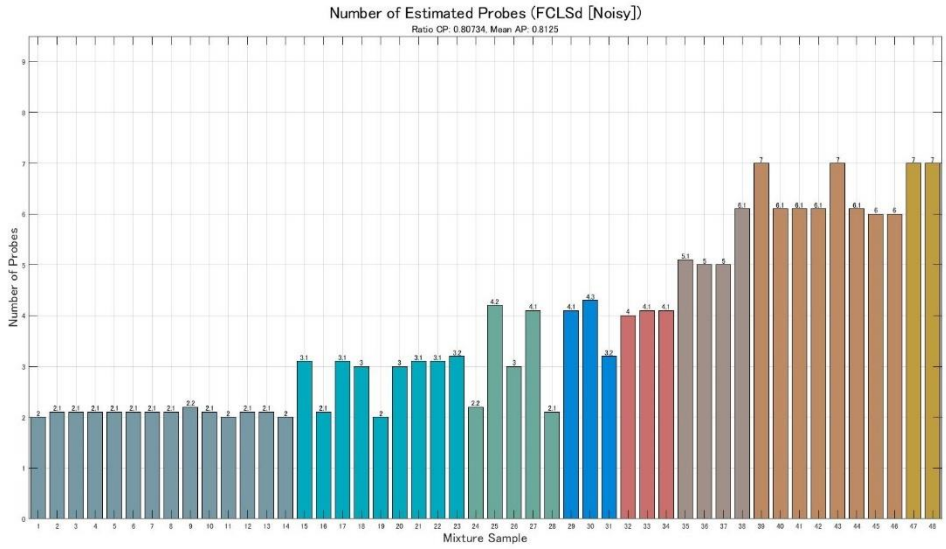

Figure 6.50. Bar graph of the numbers of estimated probes by direct FCLS on noisy mixture samples with ratio 0.001

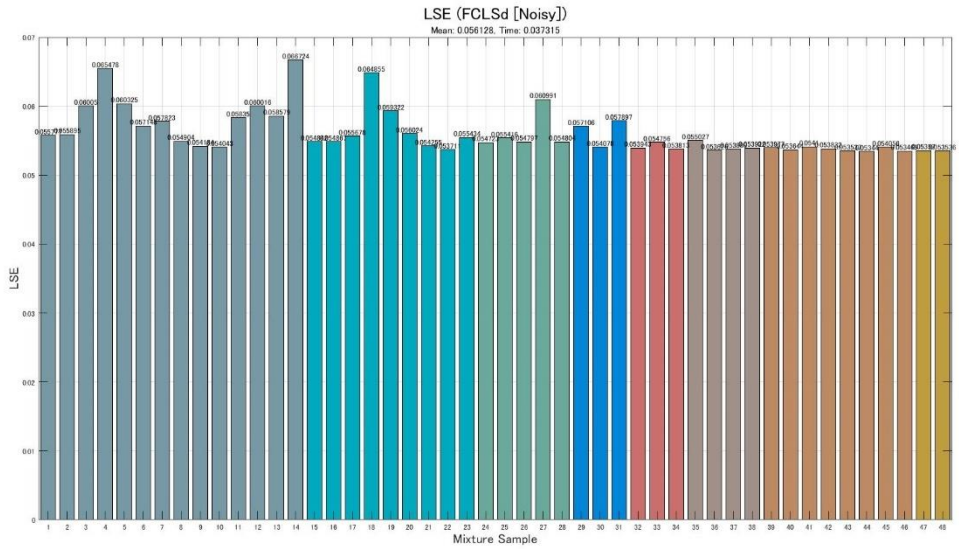

Figure 6.51. Bar graph of least square errors by direct FCLS on noisy mixture samples with ratio 0.001

6.5.3 Linear Unmixing on Denoised Mixture Samples with Noise Ratio 0.001 by Fourier-based Denoising

The pattern in the EAF plot in Figure 6.52 is still polychrome. We can observe many dark blue squares on mixture samples 29 to 48. Figure 6.53 tells that the rCP is 0.62844 and the mAP is 1.5417. We notice that the rCP has been decreased and the mAP has been increased. Nevertheless, according to Figure 6.54, the average LSE has been reduced to 0.03137 with unmixing time 0.056534s.

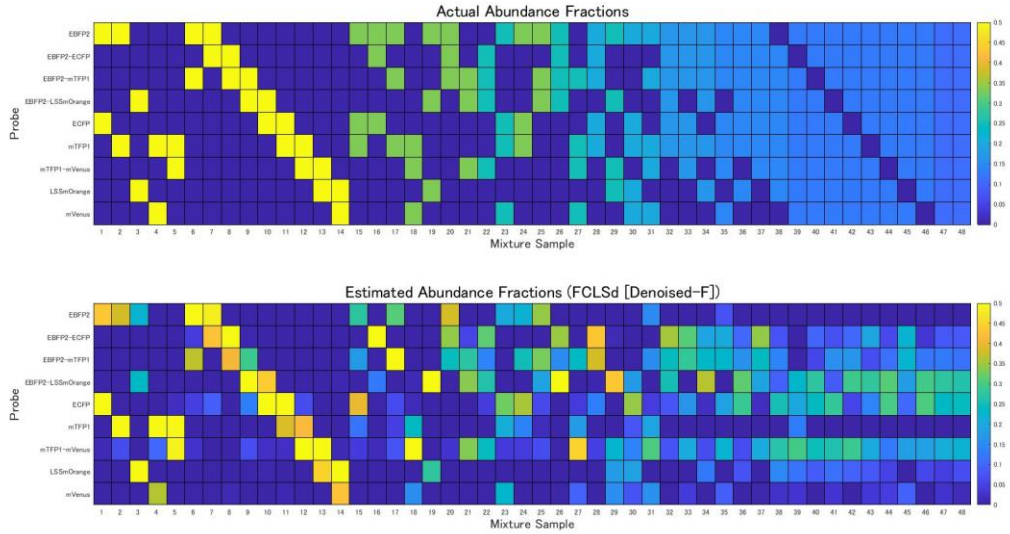

Figure 6.52. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on denoised mixture samples with noise ratio 0.001 by Fourier-based denoising

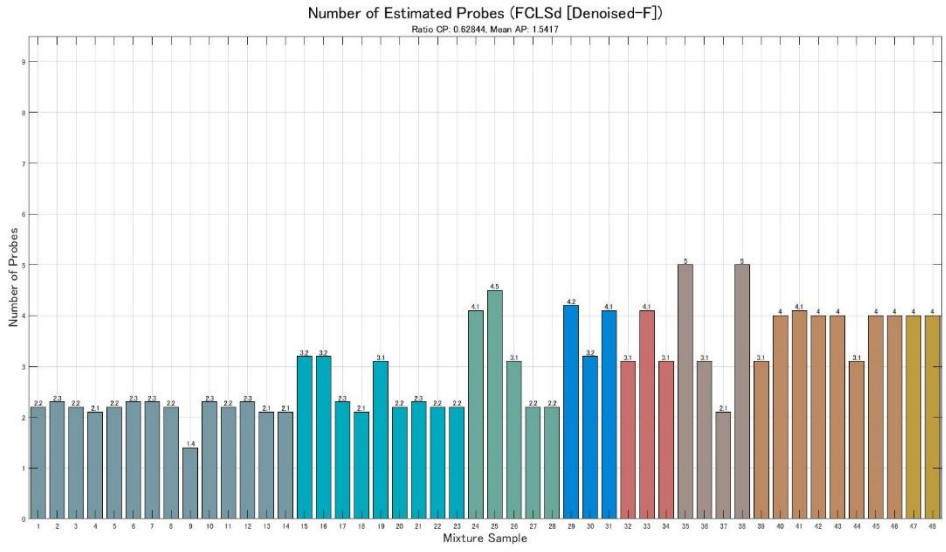

Figure 6.53. Bar graph of the numbers of estimated probes by direct FCLS on denoised mixture samples with noise ratio 0.001 by Fourier-based denoising

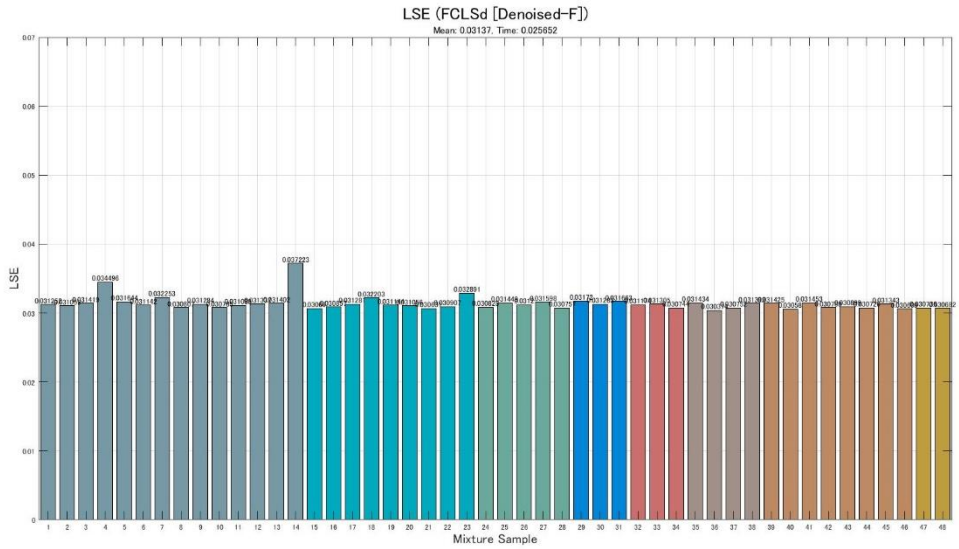

Figure 6.54. Bar graph of least square errors by direct FCLS on denoised mixture samples with noise ratio 0.001 by Fourier-based denoising

6.5.4 Linear Unmixing on Denoised Mixture Samples with Noise Ratio 0.001 by Wavelet-based Denoising

In Figure 6.55, a polychrome pattern, especially on the top of mixture samples 29 to 48, is observed with many dark blue squares on the bottom side. Figure 6.56 illustrates that the rCP value is 0.70642 and mAP 0.83333. The rCP value has been increased and the mAP value has been decreased. In Figure 6.57, the method yields the average LSE of 0.014348 and the unmixing time of 0.15655s. The LSE has been slightly reduced, but the unmixing time has been significantly increased.

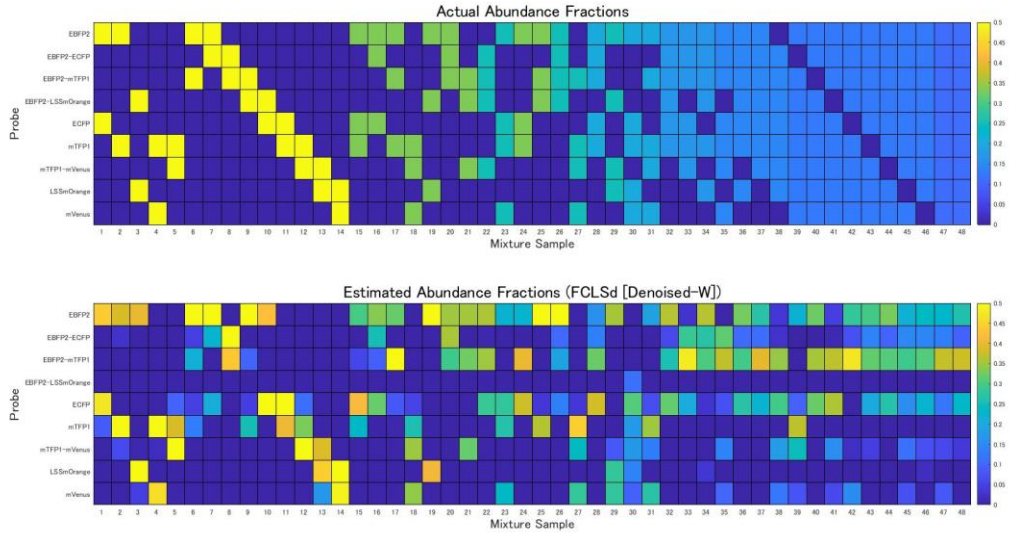

Figure 6.55. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on denoised mixture samples with noise ratio 0.001 by wavelet-based denoising

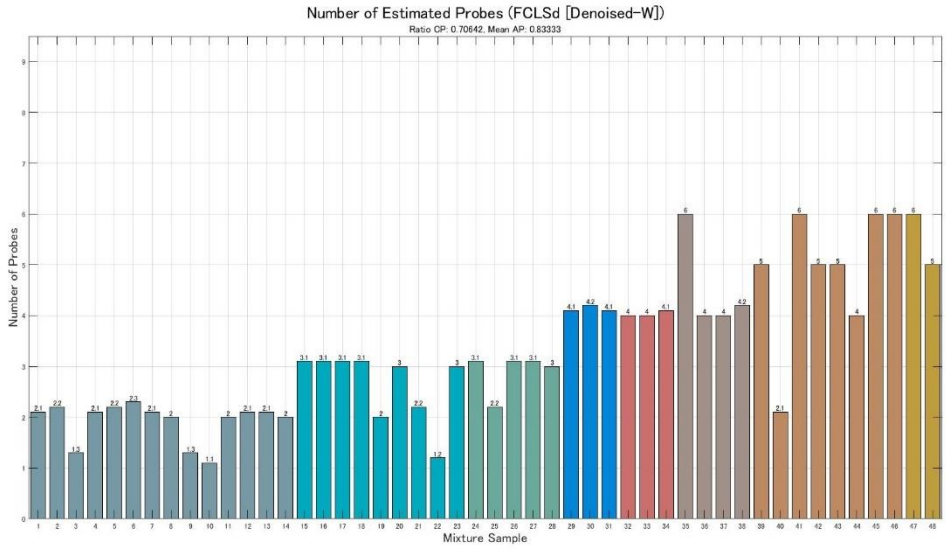

Figure 6.56. Bar graph of the numbers of estimated probes by direct FCLS on denoised mixture samples with noise ratio 0.001 by wavelet-based denoising

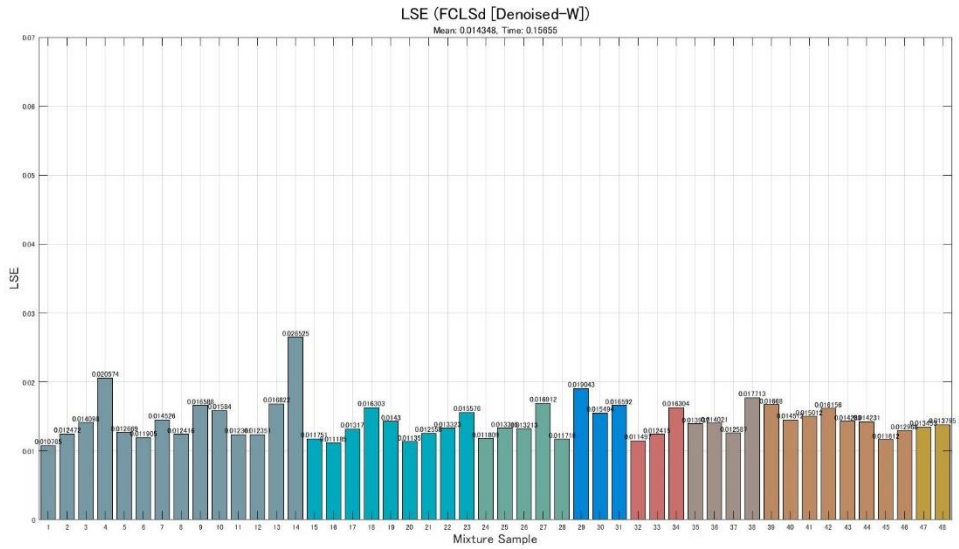

Figure 6.57. Bar graph of least square errors by direct FCLS on denoised mixture samples with noise ratio 0.001 by wavelet-based denoising

6.5.5 Linear Unmixing on Noisy Mixture Samples with Ratio 0.0005

In Figure 6.58, we observe a less polychrome pattern compared to that with ratio 0.001. However, the method still does not locate EBFP2 and mTFP1 properly for mixture samples 29 to 48. Figure 6.59 shows that the rCP value 0.8578 and the mAP value 0.66667. The values have been reduced, compared to the original mixture samples. In Figure 6.60, the method produces, on average, the LSE of 0.031613 with unmixing time 0.038857s. These values are larger than those of the original mixture samples, but smaller than those with ratio 0.001.

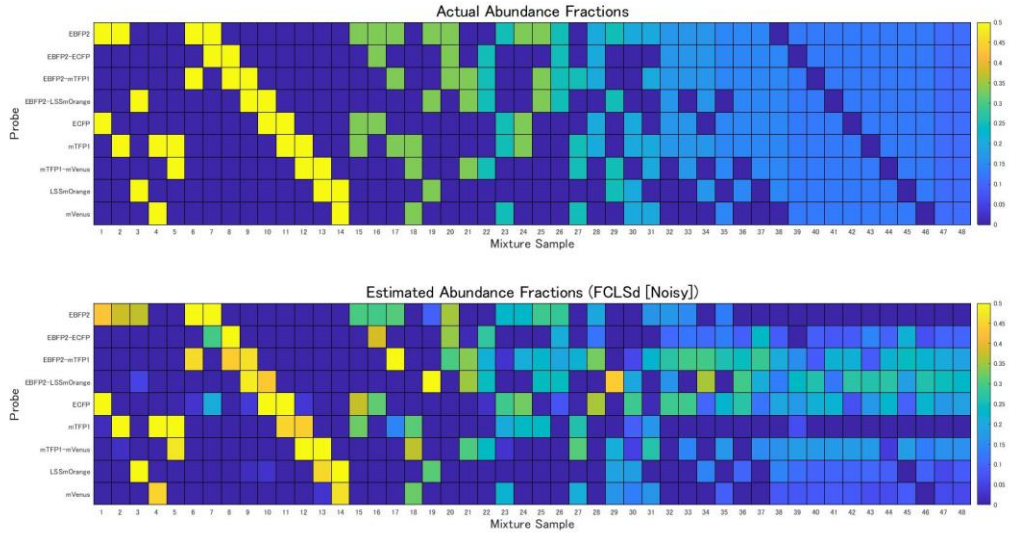

Figure 6.58. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on noisy mixture samples with ratio 0.0005

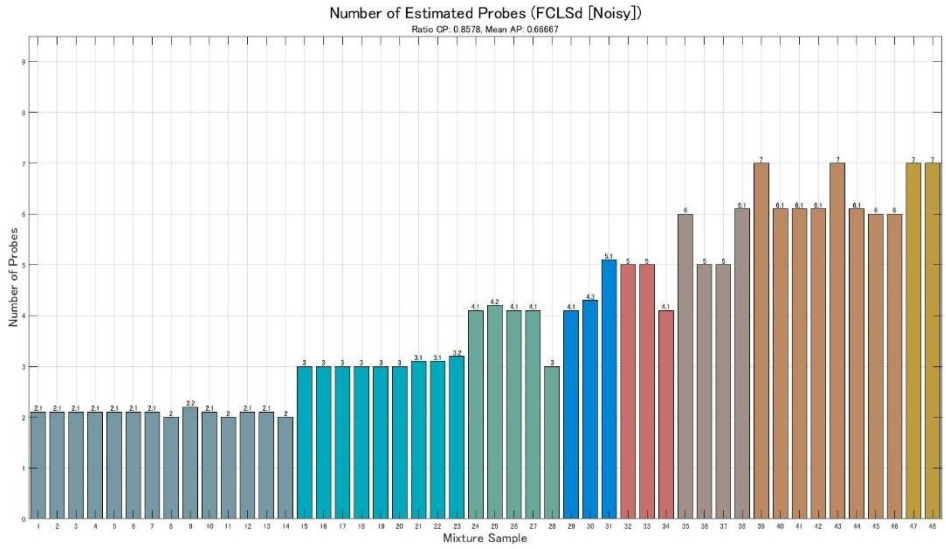

Figure 6.59. Bar graph of the numbers of estimated probes by direct FCLS on noisy mixture samples with ratio 0.0005

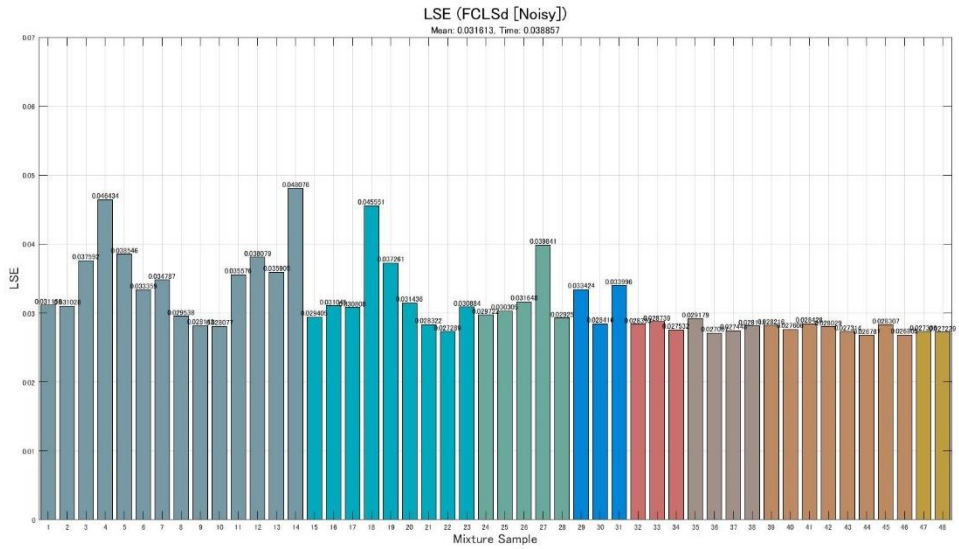

Figure 6.60. Bar graph of least square errors by direct FCLS on noisy mixture samples with ratio 0.0005

6.5.6 Linear Unmixing on Denoised Mixture Samples with Noise Ratio 0.0005 by Fourier-based Denoising

The pattern in the EAF plot in Figure 6.61 is polychrome with many dark blue squares on mixture samples 29 to 48. In Figure 6.62, compared to the noisy mixture samples, the rCP has been slightly reduced to 0.85321 and the mAP has been increased to 0.875. However, Figure 6.63 illustrates that the average LSE has been reduced to 0.020138 and the processing time has been increased to 0.048555s.

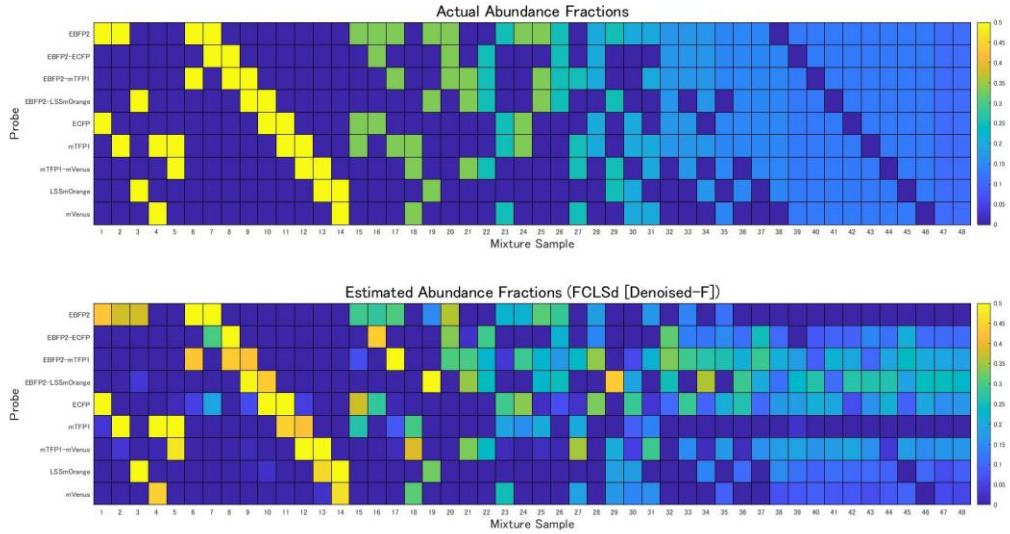

Figure 6.61. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on denoised mixture samples with noise ratio 0.0005 by Fourier-based denoising

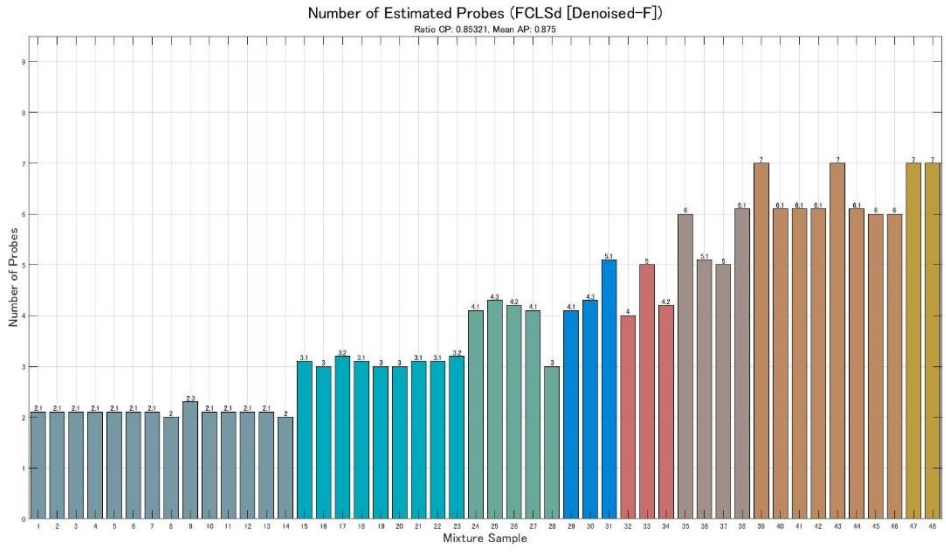

Figure 6.62. Bar graph of the numbers of estimated probes by direct FCLS on denoised mixture samples with noise ratio 0.0005 by Fourier-based denoising

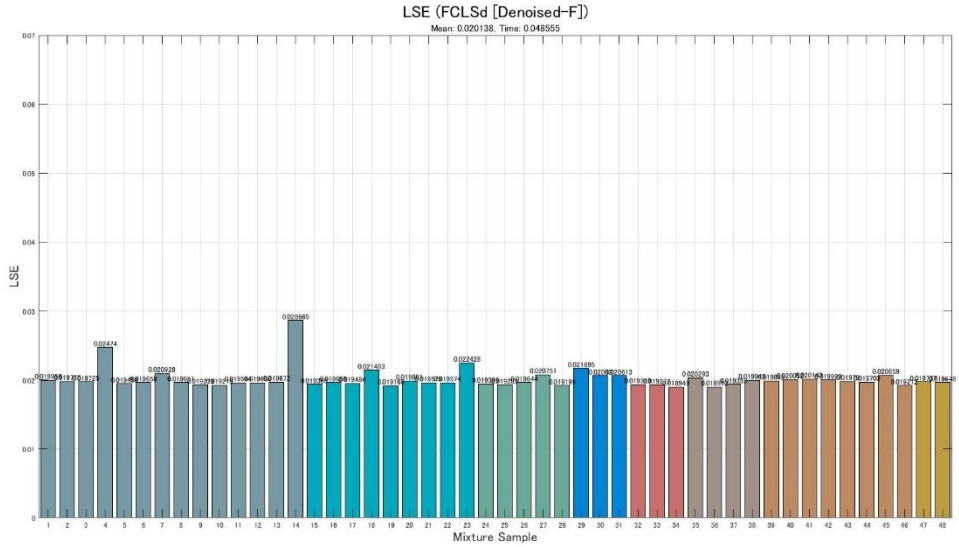

Figure 6.63. Bar graph of least square errors by direct FCLS on denoised mixture samples with noise ratio 0.0005 by Fourier-based denoising

6.5.7 Linear Unmixing on Denoised Mixture Samples with Noise Ratio 0.0005 by Wavelet-based Denoising

In comparison with the colour pattern in the EAF plot of the Fourier-based denoising method, we can observe fewer dark blue squares on mixture samples 29 to 48 in the EAF plot in Figure 6.64. Figure 6.65 depicts that the rCP and mAP values have been increased to 0.86697 and 0.95833, respectively, compared to those by Fourier-based denoising. In Figure 6.66, the method produces the average LSE of 0.011872 and the unmixing time 0.15511s. The average LSE has been decreased, but the processing time has been increased.

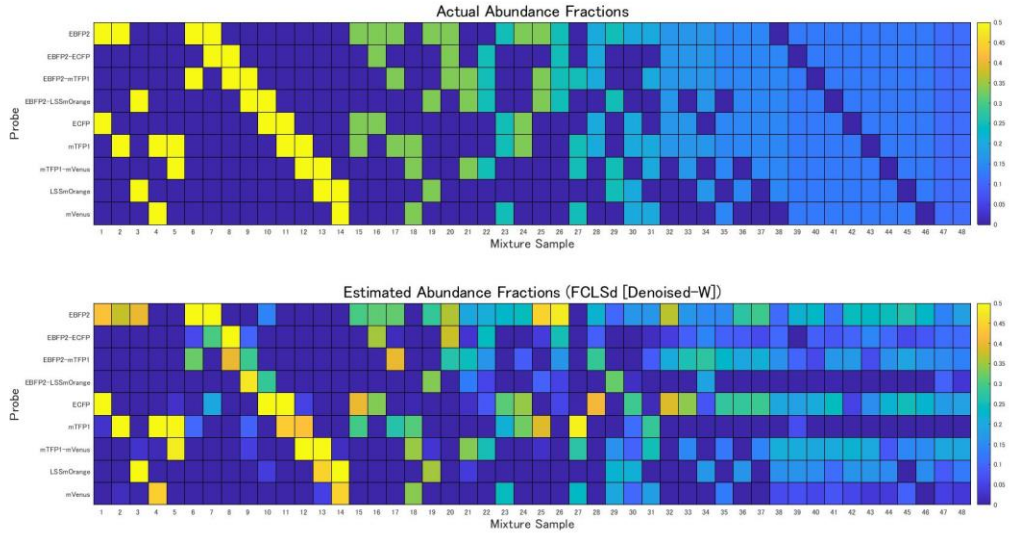

Figure 6.64. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on denoised mixture samples with noise ratio 0.0005 by wavelet-based denoising

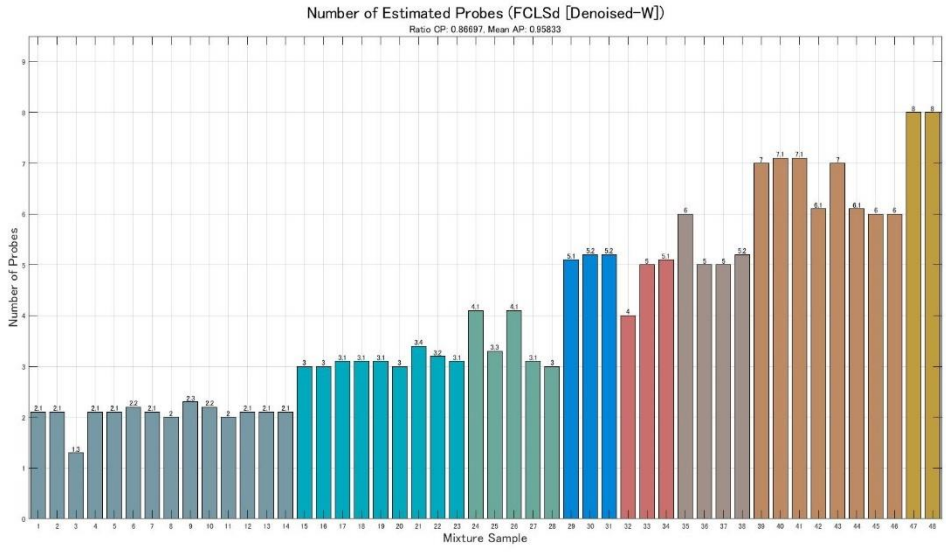

Figure 6.65. Bar graph of the numbers of estimated probes by direct FCLS on denoised mixture samples with noise ratio 0.0005 by wavelet-based denoising

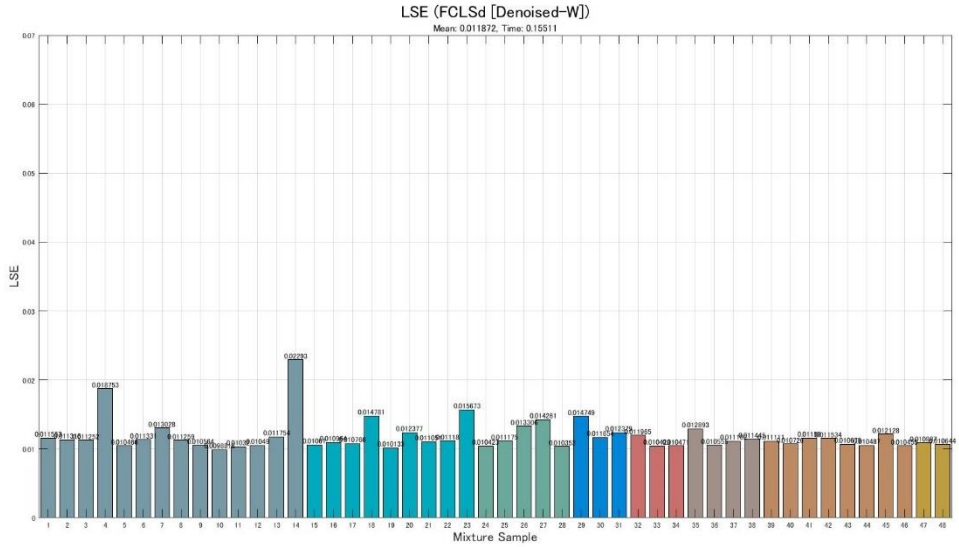

Figure 6.66. Bar graph of least square errors by direct FCLS on denoised mixture samples with noise ratio 0.0005 by wavelet-based denoising

6.5.8 Linear Unmixing on Noisy Mixture Samples with Ratio 0.0001

Figure 6.67 illustrates a less polychrome pattern compared to that with ratio 0.0005. Indeed, there are few dark blue squares on mixture samples 29 to 48. Figure 6.68 shows that the rCP value 0.96789 and the mAP value 0.72917. These values have been decreased, compared to the original mixture samples. In Figure 6.69, the average LSE is 0.016588 with unmixing time 0.028276s. The LSE is greater than that of the original mixture samples, but the unmixing is as fast as that is.

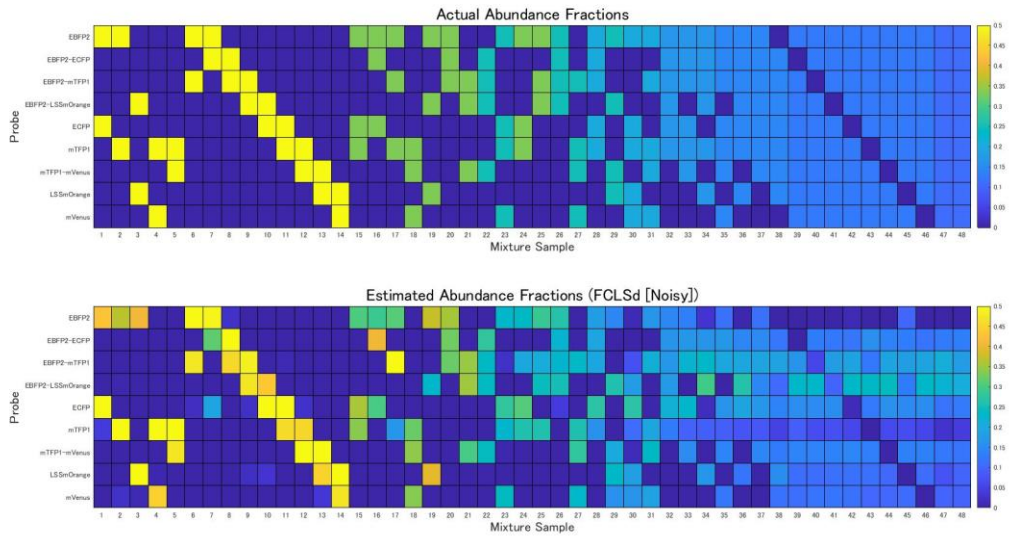

Figure 6.67. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on noisy mixture samples with ratio 0.0001

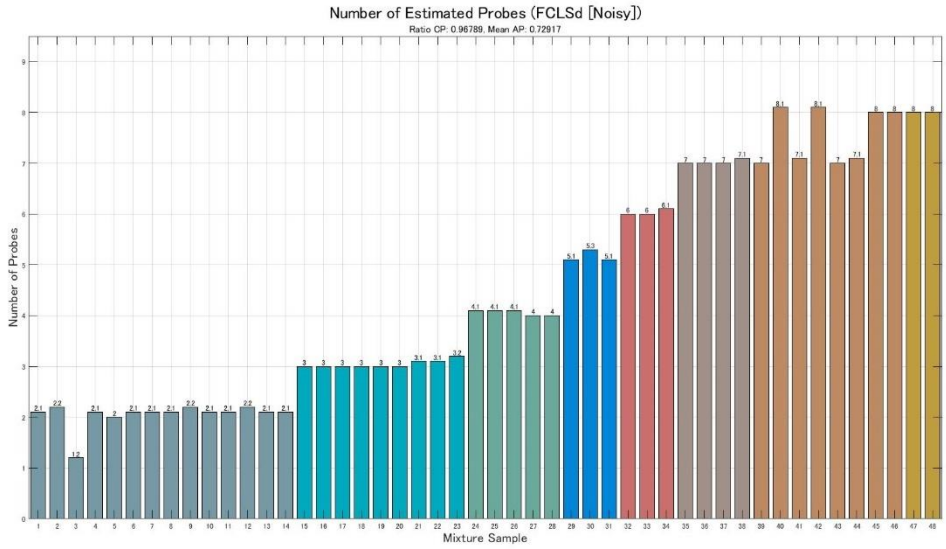

Figure 6.68. Bar graph of the numbers of estimated probes by direct FCLS on noisy mixture samples with ratio 0.0001

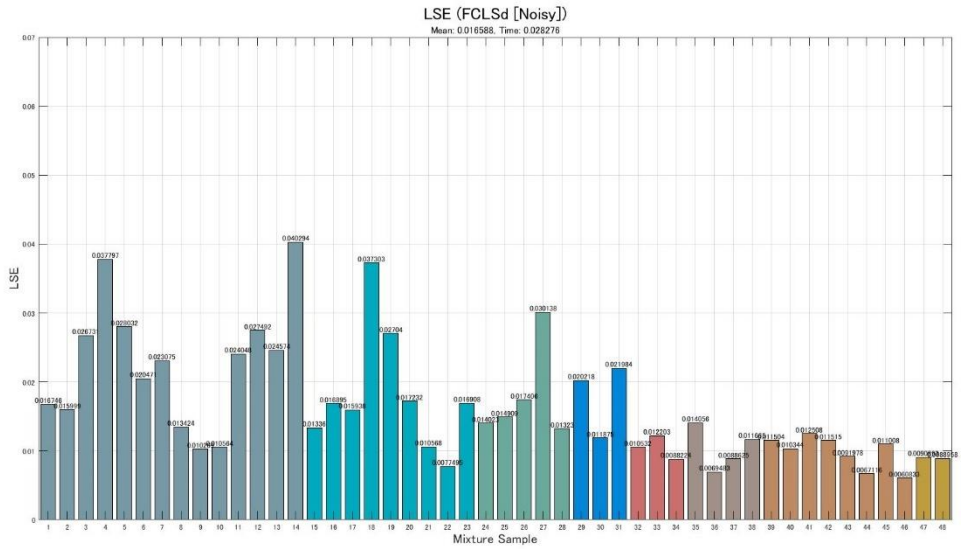

Figure 6.69. Bar graph of least square errors by direct FCLS on noisy mixture samples with ratio 0.0001

6.5.9 Linear Unmixing on Denoised Mixture Samples with Noise Ratio 0.0001 by Fourier-based Denoising

In Figure 6.70, the pattern in the EAF plot is similar to that for the original mixture samples. In Figure 6.71, compared to the noisy mixture samples, the rCP and mAP values have been increased to 0.99083 and 1.0208, respectively; these values are also similar to those for the original mixture samples. Figure 6.72 illustrates that the average LSE has been reduced to 0.0061004, which is close to that of the original mixture samples. The processing time however has been increased to 0.044528s.

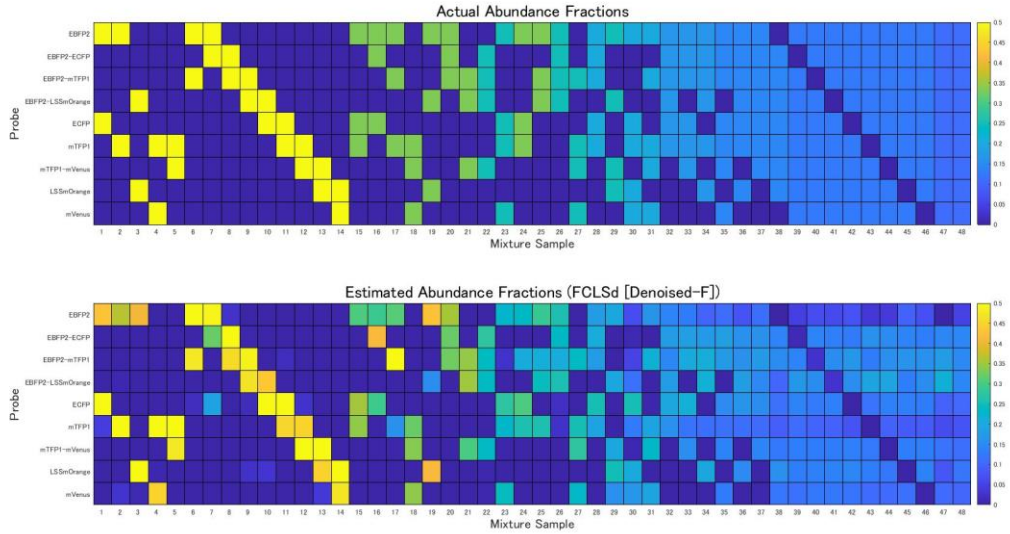

Figure 6.70. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on denoised mixture samples with noise ratio 0.0001 by Fourier-based denoising

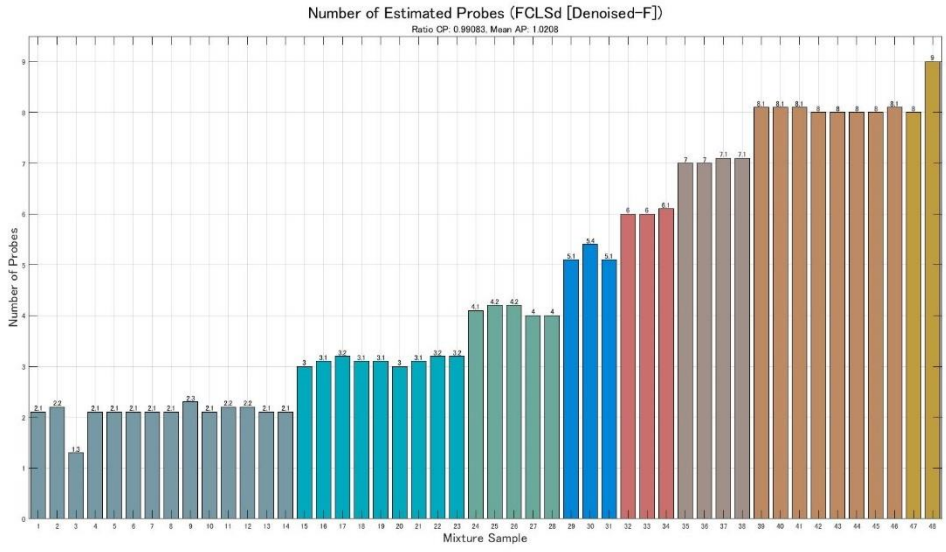

Figure 6.71. Bar graph of the numbers of estimated probes by direct FCLS on denoised mixture samples with noise ratio 0.0001 by Fourier-based denoising

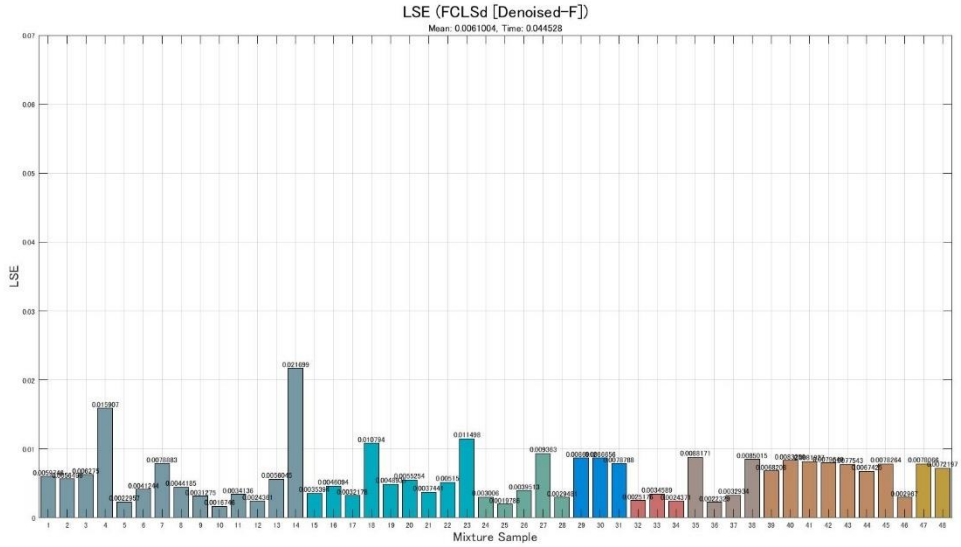

Figure 6.72. Bar graph of least square errors by direct FCLS on denoised mixture samples with noise ratio 0.0001 by Fourier-based denoising

6.5.10 Linear Unmixing on Denoised Mixture Samples with Noise Ratio 0.0001 by Wavelet-based Denoising

In Figure 6.73, the colour pattern in the EAF plot of the wavelet-based denoising method is very similar to that of the Fourier-based denoising method. Figure 6.74 illustrates that the rCP and mAP values, 0.99083 and 1.0417, respectively, are also similar to those with Fourier-based denoising. In Figure 6.75, the average LSE has been slightly reduced to 0.0060256 whereas the unmixing time has been increased to 0.14625s.

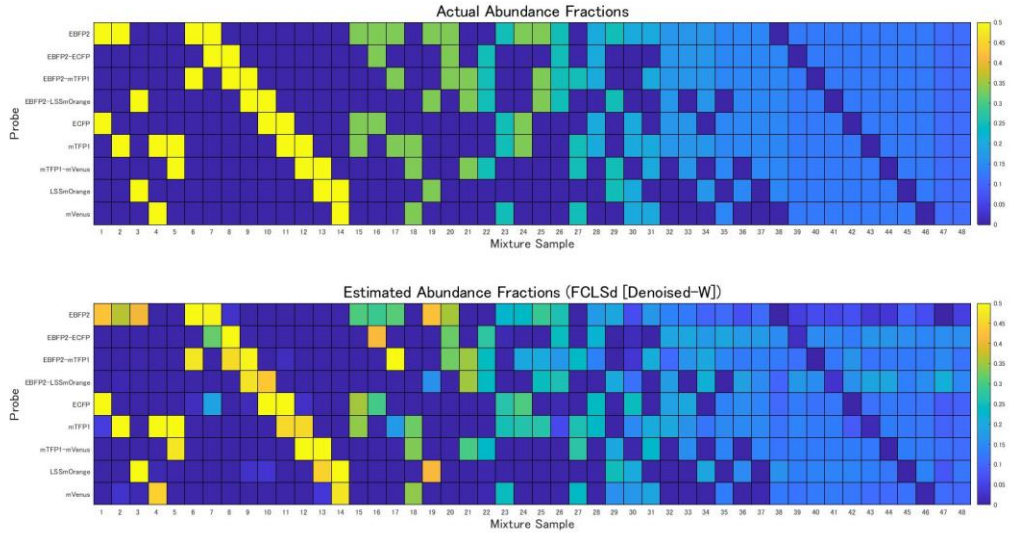

Figure 6.73. Colour maps of actual abundance fractions (upper panel) and estimated abundance fractions by direct FCLS (lower panel) on denoised mixture samples with noise ratio 0.0001 by wavelet-based denoising

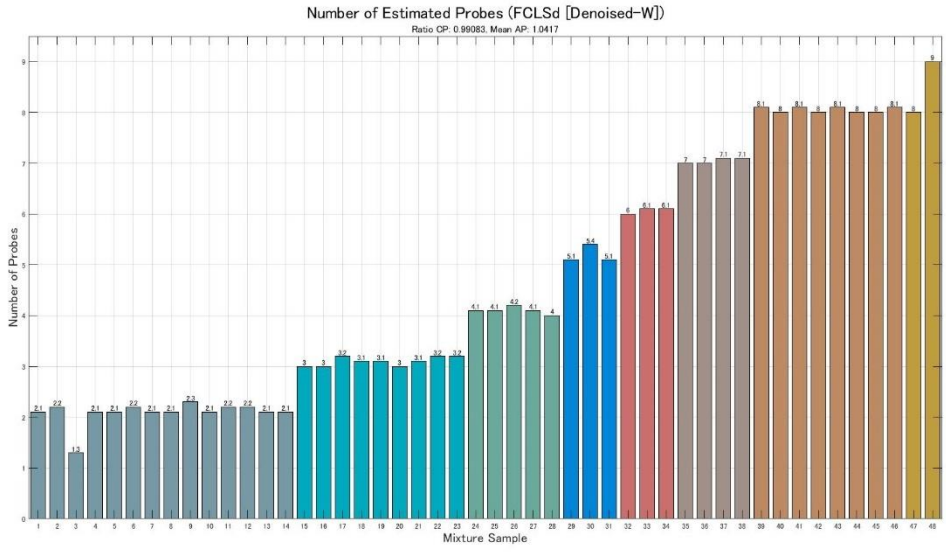

Figure 6.74. Bar graph of the numbers of estimated probes by direct FCLS on denoised mixture samples with noise ratio 0.0001 by wavelet-based denoising

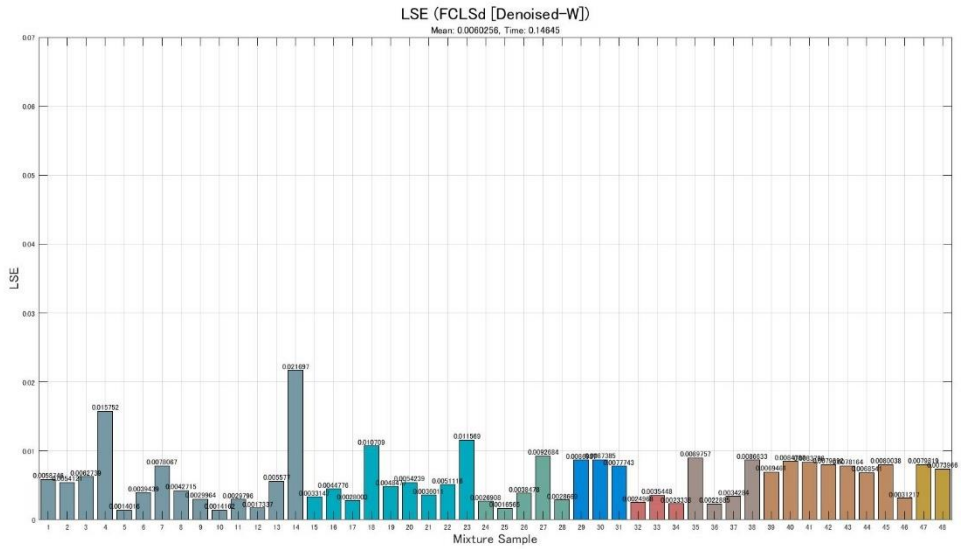

Figure 6.75. Bar graph of least square errors by direct FCLS on denoised mixture samples with noise ratio 0.0001 by wavelet-based denoising

6.5.11 Conclusion

Figure 6.76 summarizes the values of rCP, mAP, average LSE, and unmixing time of the direct FCLS method on the original mixture samples, noisy mixture samples with ratio 0.001, and denoised mixture samples with the Fourier-based method and the wavelet-based method. As noise is present, the rCP value decreases to 0.80734 on the noisy mixture samples. The mAP value also decreases, but it is still close to 1. Therefore, the average LSE increases as our expectations. The processing time on the noisy mixture samples also increases since there are less zeros in the noisy emission spectra due to random noise and thus the algorithm requires more computations for unmixing. However, it is unexpected that the denoised mixture samples produce smaller rCP values and larger mAP values than the noisy mixture samples even though they yield smaller average LSE

values. This is because the unmixing algorithm finds more correct probes on the noisy mixture samples with worse abundance fraction estimates. Furthermore, the wavelet-based denoising shows better performance than the Fourier-based denoising with respect to accuracy. This is a predictable result; the wavelet-based method is more flexible since there are more options on the choice of the mother wavelet and decomposition level, resulting in better denoising performance, while the Fourier-based method is limited to cosine and sine functions without coefficient decomposition. The algorithm also requires more processing time for the denoised mixture samples than for the noisy mixture samples due to the denoising procedures. Specifically, the wavelet-denoising method takes more time than the Fourier-based denoising method because it is more computationally expensive in compensation for flexibility.

Figure 6.77 describes the values of rCP, mAP, average LSE, and unmixing time of the direct FCLS method on the original mixture samples, noisy mixture samples with ratio 0.0005, and denoised mixture samples with the Fourier-based method and the wavelet-based method. The rCP and mAP values decrease to 0.8578 and 0.66667, respectively, on the noisy mixture samples. The rCP values are not changed much on denoised mixture samples, but the mAP values increased to 0.875 and 0.95833 for the respective Fourier-based denoising and wavelet-based denoising. The average LSE on the noisy mixture samples is larger than that on the denoised mixture samples since, as mentioned before, the unmixing algorithm locates more inaccurate solutions for the noisy mixture samples. As expected, the wavelet-based denoising produces less average LSE and requires more processing time than the Fourier-based denoising.

Figure 6.78 illustrates the values of rCP, mAP, average LSE, and unmixing time of the direct FCLS method on the original mixture samples, noisy mixture samples with ratio 0.0001, and

denoised mixture samples with the Fourier-based method and the wavelet-based method. The rCP value decreases slightly to 0.96789 for the noisy mixture samples. After denoising, however, the rCP values become 0.99083 which is the same value as the original mixture samples. The mAP value reduces to 0.72917 on the noisy mixture samples and they become close to 1.0625 on the denoised mixture samples. Even the average LSE values are very close to 0.0060189 after denoising. Therefore, both the denoising methods eliminate noise properly in this case (that is, ratio 0.0001). The unmixing algorithm requires approximately 0.3s for both the original and noisy mixture samples, leading to the fact that the noise ratio 0.0001 is so small that it does not affect the unmixing time. As our expectations, the wavelet-based denoising takes more processing time than the Fourier-based denoising.

We conclude that the wavelet-based denoising method is better than the Fourier-based denoising method with respect to accuracy, especially for noise ratio 0.0005 or more. However, the Fourier-based denoising method can be considered for rapid denoising at the expense of accuracy. In the case of noise ratio 0.0001 or less, the Fourier-based denoising method is a better choice because both the denoising methods produce small LSE values, but the wavelet-denoising method requires more processing time than the Fourier-denoising method.

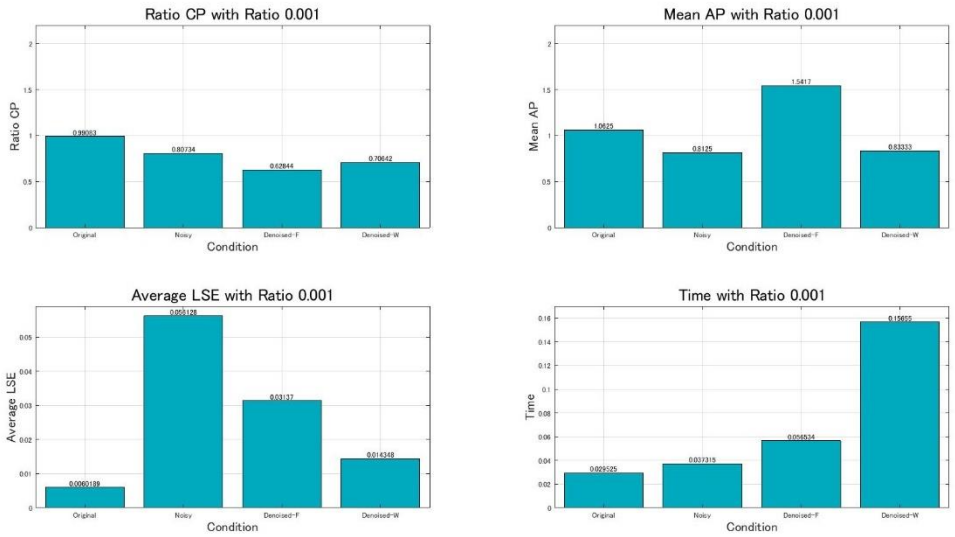

Figure 6.76. Bar graphs of ratios of detected correct probes (upper left panel), average detected incorrect probes (upper right panel), average least square errors (lower left panel), and processing times (lower right panel) by direct FCLS method on original, noisy, Fourier-denoised, and wavelet-denoised mixture samples with noise ratio 0.001

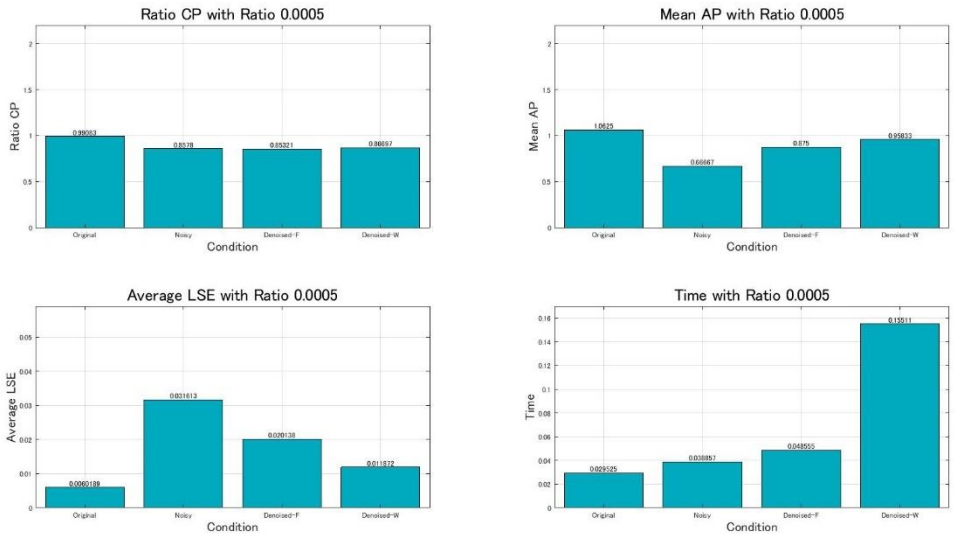

Figure 6.77. Bar graphs of ratios of detected correct probes (upper left panel), average detected incorrect probes (upper right panel), average least square errors (lower left panel), and processing times (lower right panel) by direct FCLS method on original, noisy, Fourier-denoised, and wavelet-denoised mixture samples with noise ratio 0.0005

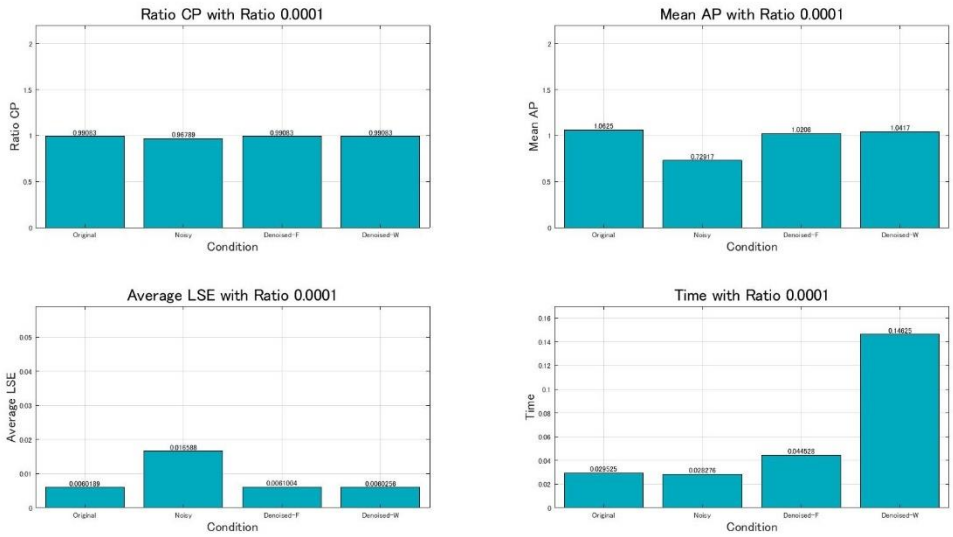

Figure 6.78. Bar graphs of ratios of detected correct probes (upper left panel), average detected incorrect probes (upper right panel), average least square errors (lower left panel), and processing times (lower right panel) by direct FCLS method on original, noisy, Fourier-denoised, and wavelet-denoised mixture samples with noise ratio 0.0001

Chapter 7

We have analyzed linear unmixing algorithms using the results obtained from experiments on the real dataset. In this chapter, we will conclude our work and discuss future works.

7.1 Conclusions

Linear unmixing is significant in fluorescence spectroscopy to estimate the abundance fractions for the decomposition of a mixture spectrum into a set of given reference spectra under the assumption that the mixture spectrum is the linear combination of the reference spectra. The linear unmixing problem is thus to find an optimal solution to the linear mixture model and the problem can be recast as an optimization problem to find an abundance vector which minimizes the least squares error. This work has presented the comparative studies to examine the behaviour of linear unmixing algorithms using the real dataset. To this end, we first provided theoretical backgrounds of the LS methods (ULS, SCLS, FCLS, and MFCLS) and the optimization techniques (GD, FC-GD, NM, TLBO, and TLSBO) and then implemented their algorithms to solve the linear unmixing problem and compared their performance.

Based on the results obtained from the experiments, we have seen that all the linear unmixing methods located correct probes with probability of 96% or more. However, the ULS, SCLS, and direct MFCLS methods located approximately two incorrect probes, while the other methods found nearly one incorrect probe.

For the LS methods, we have found that, as more constraints such as ASC and ANC were

imposed on the method, it yielded a more accurate solution, and ANC was more significant than ASC for accuracy. Also, we have observed that the direct methods found more optimal solutions than the iterative methods. However, the MFCLS methods were exceptional since the direct MFCLS method employed the SCLS solution for unmixing, which resulted in a suboptimal solution. It was remarkable that all the optimization techniques produced optimal solutions.

The LS methods required more unmixing time as more constraints were applied to their algorithms. Especially, the NCLS and direct FCLS methods used dimensionality reduction on the mixing matrix for ANC, which was computationally expensive. To avoid this, the direct MFCLS method employed AASC instead of ANC to obtain an analytical solution and thus it was much faster than them. Furthermore, the iterative methods should generate new matrices at every iteration, requiring more processing time than the direct methods.

Even though the direct FCLS method is the best unmixing algorithm for accuracy, the direct MFCLS method is regarded as the best choice among the LS methods if we consider its processing time.

The FC-GD method located an optimal solution by imposing ASC and ANC on the update equation of the GD method. However, we could use the GD method with the bounding process for the linear mixture model whose mixing matrix formed an overdetermined system with linearly independent columns. Due to the simplicity of the GD algorithm, it was much faster than the FC-GD method. Also, we have seen that, compared to the standard NM method, the adaptive method required more iterations for convergence, resulting in slow convergence. There was however no significant improvement in accuracy for the adaptive NM method. The TLBO method is a strong global optimization technique using stochastic procedures for global and local searches, which requires

much time for convergence. For better, faster convergence, the TLSBO method has been introduced; the studying phase has been added to the TLBO algorithm to give random changes to the positions of the members, enabling the TLSBO method to converge faster to the solution than the TLBO method. Still, it takes long unmixing time, compared with other optimization methods.

Overall, the FC-GD method is the best linear unmixing algorithm among the optimization methods due to its flexibility to use and rapid convergence. If the system, however, satisfies two conditions (that is, overdetermined system and linearly independent columns in the mixing matrix), then the GD method with the bounding process is the algorithm of choice.

We have observed that the wavelet-based denoising method was slower but yielded more accurate estimates than the Fourier-based denoising method; the wavelet-based denoising method has more options to choose parameters for denoising, whereas the Fourier-based denoising method is restrictive, leading to the fact that the wavelet denoising algorithm has higher complexity than the Fourier-based denoising algorithm.

In general, the wavelet-based denoising method shows better performance in terms of accuracy, and the Fourier-based denoising method in terms of speed. However, if noise is sufficiently small, the Fourier-denoising method is a better choice because both denoising methods remove the noise very well but the wavelet-denoising method requires much processing time.

7.2 Future Works

We have made many assumptions in this research such as the linearity of the mixture model, the overdetermined system formed by the mixture model, strict convexity for the GD method, etc. Future works will therefore include the following topics.

7.2.1 Nonlinear Unmixing Method

The linear unmixing method is preferred to solve spectral unmixing problems due to its simplicity and efficiency [11]. Even though nonlinear spectral unmixing is a challenging problem, nonlinear approaches can estimate more accurate, robust abundance fractions than the linear approach [10]. Therefore, we believe that nonlinear unmixing methods, for example, the nonnegative matrix factorization (NMF) method [72], will be an interesting research topic.

7.2.2 Linear Unmixing Method for Underdetermined System

In spectroscopy, the linear spectral mixture model is often an overdetermined system [8]. However, the model can be an underdetermined system (especially, when the number of detection channels to capture emission spectra is less than the number of fluorophores in the sample [73]) and the widely used linear unmixing generally fails in underdetermined cases. In this case, we should consider another approach, for instance, the similarity-unmixing algorithm SIMI [74]. This method will be another interesting research topic.

7.2.3 Nonconvex Linear Unmixing Problem

We have seen that, when the linear unmixing problem is a strictly convex problem, then we could use the GD method (with the bounding process); otherwise, we should use the FC-GD method. For nonconvex linear unmixing problems, there are other approaches to find an optimal solution. For

example, the NMF method with a generalized minimax concave regularization [75] can be considered. This method will be also an interesting research topic.

7.2.4 Application of Deep Learning

Advances in computing technology have fostered the development of new and powerful deep learning techniques, which have demonstrated promising results in a wide range of applications. Particularly, deep learning methods have been successfully used to abundance estimations [76]. As an example, deep generative endmember modelling has been developed to estimate abundance fractions for spectral mixture models [77]. We believe that deep learning methods will be another interesting research topic.

References

[1] E. Cloutis, P. Szymanski, D. Applin, and D. Goltz, "Identification and discrimination of polycyclic aromatic hydrocarbons using Raman spectroscopy," *Icarus*, vol. 274, pp. 211–230, Aug. 2016, doi: 10.1016/j.icarus.2016.03.023.

[2] W. Shi, W.-E. Zhuang, J. Hur, and L. Yang, "Monitoring dissolved organic matter in wastewater and drinking water treatments using spectroscopic analysis and ultra-high resolution mass spectrometry," *Water Res.*, vol. 188, p. 116406, 2021.

[3] A. D. Martinez, A. N. Fioretti, E. S. Toberer, and A. C. Tamboli, "Synthesis, structure, and optoelectronic properties of II–IV–V 2 materials," *J. Mater. Chem. A*, vol. 5, no. 23, pp. 11418–11435, 2017.

[4] F. Scheirlinckx, R. Buchet, J.-M. Ruysschaert, and E. Goormaghtigh, "Monitoring of secondary and tertiary structure changes in the gastric H+/K+-ATPase by infrared spectroscopy," *Eur. J. Biochem.*, vol. 268, no. 13, pp. 3644–3653, 2001.

[5] S. A. Yoon, S. Y. Park, Y. Cha, L. Gopala, and M. H. Lee, "Strategies of detecting bacteria using fluorescence-based dyes," *Front. Chem.*, p. 668, 2021.

[6] C. J. Wienken, P. Baaske, S. Duhr, and D. Braun, "Thermophoretic melting curves quantify the conformation and stability of RNA and DNA," *Nucleic Acids Res.*, vol. 39, no. 8, pp. e52–e52, 2011.

[7] A. Sharma and S. G. Schulman, *Introduction to Fluorescence Spectroscopy*. Wiley, 1999.

[8] T. Zimmermann, "Spectral imaging and linear unmixing in light microscopy," *Microsc. Tech.*, pp. 245–265, 2005.

[9] T. Zimmermann, J. Rietdorf, and R. Pepperkok, “Spectral imaging and its applications in live cell microscopy,” *FEBS Lett.*, vol. 546, no. 1, pp. 87–92, 2003.

[10] D. Shah, Y. N. Trivedi, and T. Zaveri, “Non-Linear Spectral Unmixing: A Case Study On Mangalore Aviris-Ng Hyperspectral Data,” in *2020 IEEE Bombay Section Signature Conference (IBSSC)*, 2020, pp. 11–15.

[11] R. Rajabi and H. Ghassemian, “Hyperspectral data unmixing using GNMF method and sparseness constraint,” in *2013 IEEE International Geoscience and Remote Sensing Symposium-IGARSS*, 2013, pp. 1450–1453.

[12] H. Deborah, M. O. Ulfarsson, and J. Sigurdsson, “Fully Constrained Least Squares Linear Spectral Unmixing of The Scream (Verso, 1893),” in *2021 11th Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2021, pp. 1–5.

[13] C. Quintano, A. Fernández-Manso, Y. E. Shimabukuro, and G. Pereira, “Spectral unmixing,” *Int. J. Remote Sens.*, vol. 33, no. 17, pp. 5307–5340, 2012.

[14] H. Franz and V. Jendreizik, “Fluorescence Method Development Handbook,” *Thermo Fish. Sci. Germering Ger.*, pp. 1–2, 2013.

[15] D. W. Ball, *Field guide to spectroscopy*, vol. 8. Spie Press Bellingham, Washington, 2006.

[16] F. Jin and F. Sattar, “Enhancement of recorded respiratory sound using signal processing techniques,” in *Encyclopedia of Information Communication Technology*, IGI Global, 2009, pp. 291–300.

[17] H. Y. Holzapfel *et al.*, “Fluorescence multiplexing with spectral imaging and combinatorics,” *ACS Comb. Sci.*, vol. 20, no. 11, pp. 653–659, 2018.

[18] R. C. Bishop, “Metaphysical and epistemological issues in complex systems,” in *Philosophy of*

complex systems, Elsevier, 2011, pp. 105–136.

[19] L. L. Scharf and B. Friedlander, "Matched subspace detectors," *IEEE Trans. Signal Process.*, vol. 42, no. 8, pp. 2146–2157, Aug. 1994, doi: 10.1109/78.301849.

[20] C.-I. Chang, H. Ren, F. M. D'Amico, and J. O. Jensen, "Subpixel target size estimation for remotely sensed imagery," in *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery IX*, 2003, vol. 5093, pp. 398–407.

[21] C.-I. Chang and D. C. Heinz, "Constrained subpixel target detection for remotely sensed imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 38, no. 3, pp. 1144–1159, 2000.

[22] D. Heinz, C.-I. Chang, and M. L. Althouse, "Fully constrained least-squares based linear unmixing [hyperspectral image classification]," in *IEEE 1999 International Geoscience and Remote Sensing Symposium. IGARSS'99 (Cat. No. 99CH36293)*, 1999, vol. 2, pp. 1401–1403.

[23] E. Wong and C.-I. Chang, "Modified full abundance-constrained spectral unmixing," in *High-Performance Computing in Remote Sensing II*, 2012, vol. 8539, pp. 72–83.

[24] J. Chen, C. Richard, H. Lantéri, C. Theys, and P. Honeine, "A gradient based method for fully constrained least-squares unmixing of hyperspectral images," in *2011 IEEE Statistical Signal Processing Workshop (SSP)*, 2011, pp. 301–304.

[25] C. Theys, N. Dobigeon, J.-Y. Tourneret, and H. Lantéri, "Linear unmixing of hyperspectral images using a scaled gradient method," in *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*, 2009, pp. 729–732.

[26] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, 1965.

[27] L. Han and M. Neumann, "Effect of dimensionality on the Nelder–Mead simplex method,"

Optim. Methods Softw., vol. 21, no. 1, pp. 1–16, 2006.

[28] F. Gao and L. Han, "Implementing the Nelder-Mead simplex algorithm with adaptive parameters," *Comput. Optim. Appl.*, vol. 51, no. 1, pp. 259–277, 2012.

[29] M. Selvam, R. Manickam, and V. Saravanan, "Nelder-Mead Simplex Search Method -A Study," in *Data Analytics and Artificial Intelligence*, vol. 2, 2022, p. 2022. doi: 10.46632/daai/2/2/7.

[30] S. Chattopadhyay, A. Marik, and R. Pramanik, "A Brief Overview of Physics-inspired Metaheuristic Optimization Techniques." arXiv, Jan. 30, 2022. doi: 10.48550/arXiv.2201.12810.

[31] R. V. Rao, "Teaching-learning-based optimization algorithm," in *Teaching learning based optimization algorithm*, Springer, 2016, pp. 9–39.

[32] E. Akbari, M. Ghasemi, M. Gil, A. Rahimnejad, and S. Andrew Gadsden, "Optimal Power Flow via Teaching-Learning-Studying-Based Optimization Algorithm," *Electr. Power Compon. Syst.*, vol. 49, no. 6–7, pp. 584–601, 2022.

[33] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[34] S. Rosario-Torres, "Iterative algorithms for abundance estimation on unmixing of hyperspectral imagery," PhD Thesis, 2004.

[35] C.-I. Chang, H. Ren, C.-C. Chang, F. D'Amico, and J. O. Jensen, "Estimation of subpixel target size for remotely sensed imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 6, pp. 1309–1320, 2004.

[36] D. C. Heinz, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 3, pp. 529–545, 2001.

[37] H.-C. Li, M. Song, and C.-I. Chang, "Finding analytical solutions to abundance fully-constrained linear spectral mixture analysis," in *2014 IEEE Geoscience and Remote Sensing Symposium*, 2014, pp. 3682–3685.

[38] M. French, "Fundamentals of Optimization," *Springer Int. Publ. DOI*, vol. 10, pp. 978–3, 2018.

[39] D. Bertsekas, *Convex optimization algorithms*. Athena Scientific, 2015.

[40] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[41] R. Horst and P. M. Pardalos, *Handbook of global optimization*, vol. 2. Springer Science & Business Media, 2013.

[42] N. Johnston, *Advanced Linear and Matrix Algebra*. Springer, 2021.

[43] R. L. Burden, J. D. Faires, and A. M. Burden, *Numerical analysis*. Cengage learning, 2015.

[44] D. Sloughter, *Calculus From Approximation to Theory*. Providence, Rhode Island: American Mathematical Society, 2020.

[45] R. T. Rockafellar, *Convex analysis*, vol. 18. Princeton university press, 1970.

[46] J.-F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal, *Numerical optimization: theoretical and practical aspects*. Springer Science & Business Media, 2006.

[47] N. Hansen, "Benchmarking the Nelder-Mead downhill simplex algorithm with many local restarts," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, 2009, pp. 2403–2408.

[48] M. J. Kochenderfer and T. A. Wheeler, *Algorithms for optimization*. Mit Press, 2019.

[49] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the Nelder–Mead simplex method in low dimensions," *SIAM J. Optim.*, vol. 9, no. 1, pp. 112–147, 1998.

[50] S. Wessing, “Proper initialization is crucial for the Nelder–Mead simplex search,” *Optim. Lett.*, vol. 13, no. 4, pp. 847–856, 2019.

[51] C. Blum and A. Roli, “Metaheuristics in combinatorial optimization: Overview and conceptual comparison,” *ACM Comput. Surv.*, vol. 35, no. 3, pp. 268–308, Spring 2003, doi: 10.1145/937503.937505.

[52] Z. Zhai, G. Jia, and K. Wang, “A Novel Teaching-Learning-Based Optimization with Error Correction and Cauchy Distribution for Path Planning of Unmanned Air Vehicle,” *Comput. Intell. Neurosci.*, vol. 2018, p. 5671709, Aug. 2018, doi: 10.1155/2018/5671709.

[53] M. Črepinšek, S.-H. Liu, and L. Mernik, “A note on teaching–learning-based optimization algorithm,” *Inf. Sci.*, vol. 212, pp. 79–93, 2012.

[54] A. Rajasekhar, R. Rani, K. Ramya, and A. Abraham, “Elitist teaching learning opposition based algorithm for global optimization,” in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2012, pp. 1124–1129.

[55] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O’Reilly Media, Inc., 2019.

[56] L. Prechelt, “Early stopping-but when?,” in *Neural Networks: Tricks of the trade*, Springer, 1998, pp. 55–69.

[57] G. P. Tolstov, *Fourier Series*, Illustrated edition. Dover Publications, 2012.

[58] I. Kanatov, D. Butusov, A. Sinitca, V. Gulvanskii, and D. Kaplun, “One Technique to Enhance the Resolution of Discrete Fourier Transform,” *Electronics*, vol. 3, p. 330, Mar. 2019, doi: 10.3390/electronics8030330.

[59] D. Song, A. M. C. Baek, and N. Kim, “Forecasting stock market indices using padding-based

fourier transform denoising and time series deep learning models," *IEEE Access*, vol. 9, pp. 83786–83796, 2021.

[60] M. F. Wahab, F. Gritti, and T. C. O'Haver, "Discrete Fourier transform techniques for noise reduction and digital enhancement of analytical signals," *TrAC Trends Anal. Chem.*, vol. 143, p. 116354, Oct. 2021, doi: 10.1016/j.trac.2021.116354.

[61] Q. Kong, T. Siauw, and A. Bayen, *Python Programming and Numerical Methods: A Guide for Engineers and Scientists*, $ {nombre}er édition. London: Academic Press, 2020.

[62] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput.*, vol. 19, no. 90, pp. 297–301, 1965.

[63] M. Masoori and M. L. Greenfield, "Reducing noise in computed correlation functions using techniques from signal processing," *Mol. Simul.*, vol. 43, no. 18, pp. 1485–1495, Dec. 2017, doi: 10.1080/08927022.2017.1321753.

[64] A. Boggess and F. J. Narcowich, *A first course in wavelets with Fourier analysis*. John Wiley & Sons, 2015.

[65] J. D. Cook, "Fourier uncertainty principle," Mar. 17, 2021. https://www.johndcook.com/blog/2021/03/17/fourier-uncertainty-principle/ (accessed Oct. 09, 2022).

[66] M. Yang, Y.-F. Sang, C. Liu, and Z. Wang, "Discussion on the choice of decomposition level for wavelet based hydrological time series modeling," *Water*, vol. 8, no. 5, p. 197, 2016.

[67] D. B. Percival and A. T. Walden, *Wavelet methods for time series analysis*, vol. 4. Cambridge university press, 2000.

[68] Y. I. Jang, J. Y. Sim, J.-R. Yang, and N. K. Kwon, "The Optimal Selection of Mother Wavelet

Function and Decomposition Level for Denoising of DCG Signal," *Sensors*, vol. 21, no. 5, p. 1851, Mar. 2021, doi: 10.3390/s21051851.

[69] T. Zikov, S. Bibian, G. A. Dumont, M. Huzmezan, and C. R. Ries, "A wavelet based de-noising technique for ocular artifact correction of the electroencephalogram," in *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society][Engineering in Medicine and Biology*, 2002, vol. 1, pp. 98–105.

[70] F. Santamaria, C. A. Cortés, and F. Roman, "Noise reduction of measured lightning electric fields signals using the wavelet transform," in *X International Symposium on Lightning Protection*, 2009.

[71] G. Bachman, L. Narici, and E. Beckenstein, *Fourier and wavelet analysis*, vol. 586. Springer, 2000.

[72] H. Ikoma, B. Heshmat, G. Wetzstein, and R. Raskar, "Nonlinear fluorescence spectra unmixing," in *2014 Conference on Lasers and Electro-Optics (CLEO) - Laser Science to Photonic Applications*, Jun. 2014, pp. 1–2. doi: 10.1364/CLEO_AT.2014.JTh2A.9.

[73] J. Rietdorf and T. Gadella, *Microscopy techniques*, vol. 1. Springer, 2005.

[74] A. Rakhymzhan *et al.*, "Synergistic strategy for multicolor two-photon microscopy: application to the analysis of germinal center reactions in vivo," *Sci. Rep.*, vol. 7, no. 1, pp. 1–16, 2017.

[75] F. Xiong, J. Zhou, J. Lu, and Y. Qian, "Nonconvex nonseparable sparse nonnegative matrix factorization for hyperspectral unmixing," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 13, pp. 6088–6100, 2020.

[76] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep learning classifiers for hyperspectral imaging: A review," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 279–317, 2019.

[77] R. A. Borsoi, T. Imbiriba, and J. C. M. Bermudez, “Deep generative endmember modeling: An application to unsupervised spectral unmixing,” *IEEE Trans. Comput. Imaging*, vol. 6, pp. 374–384, 2019.

Appendix A

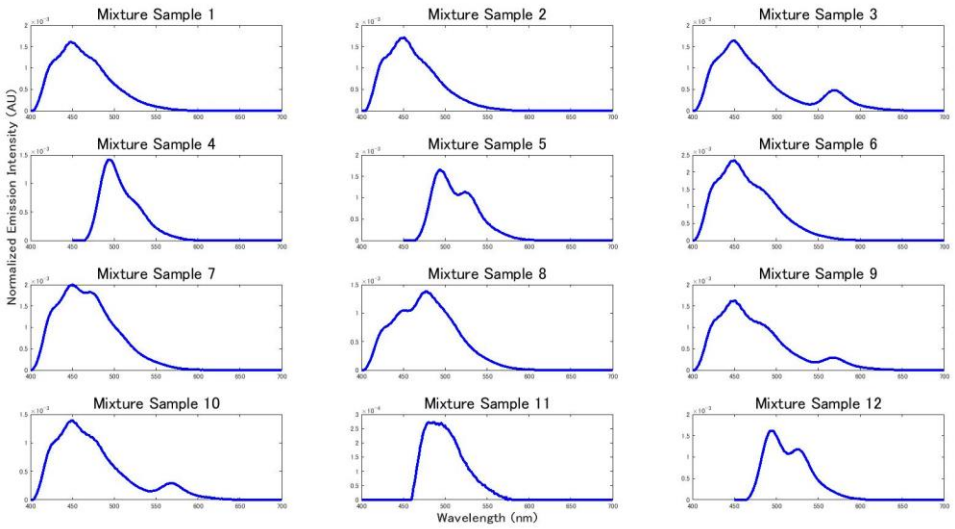

Figure A.1. Plots of mixture emission spectra 1 to 12

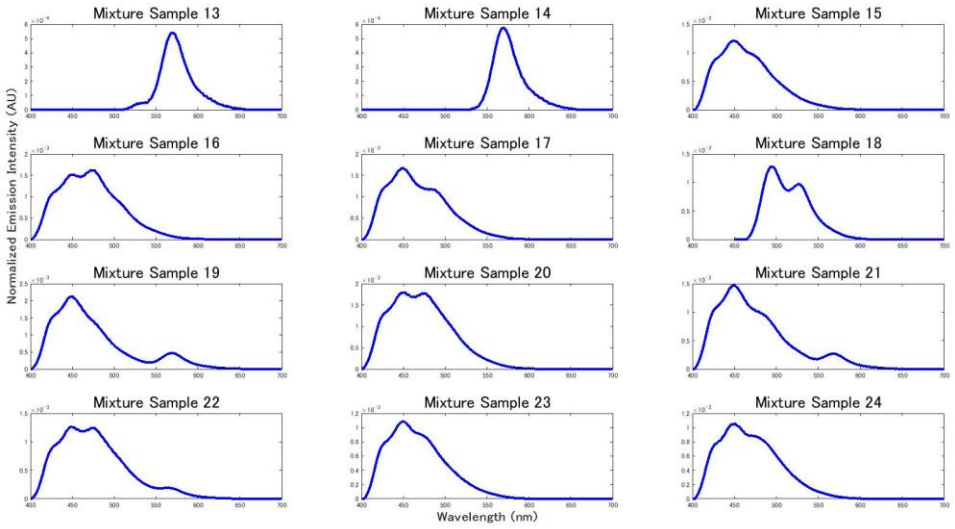

Figure A.2. Plots of mixture emission spectra 13 to 24

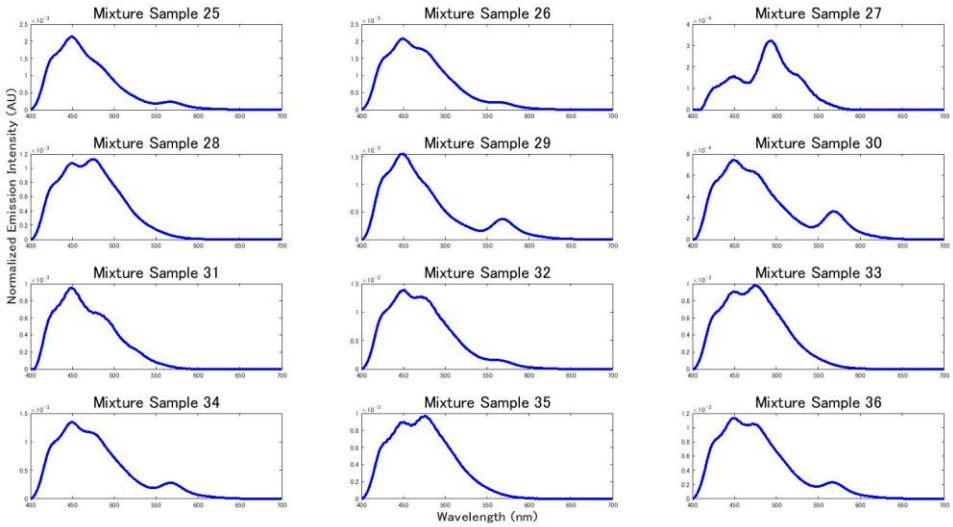

Figure A.3. Plots of mixture emission spectra 25 to 36

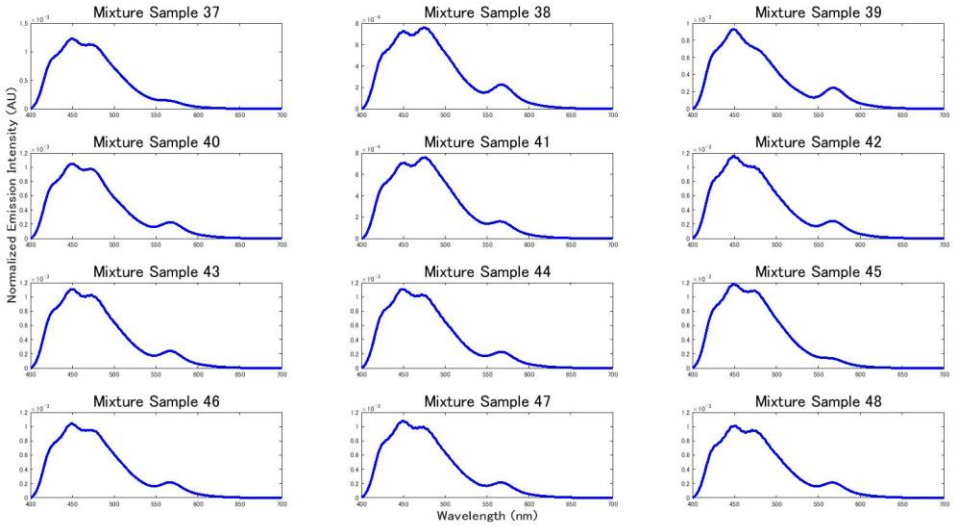

Figure A.4. Plots of mixture emission spectra 37 to 48

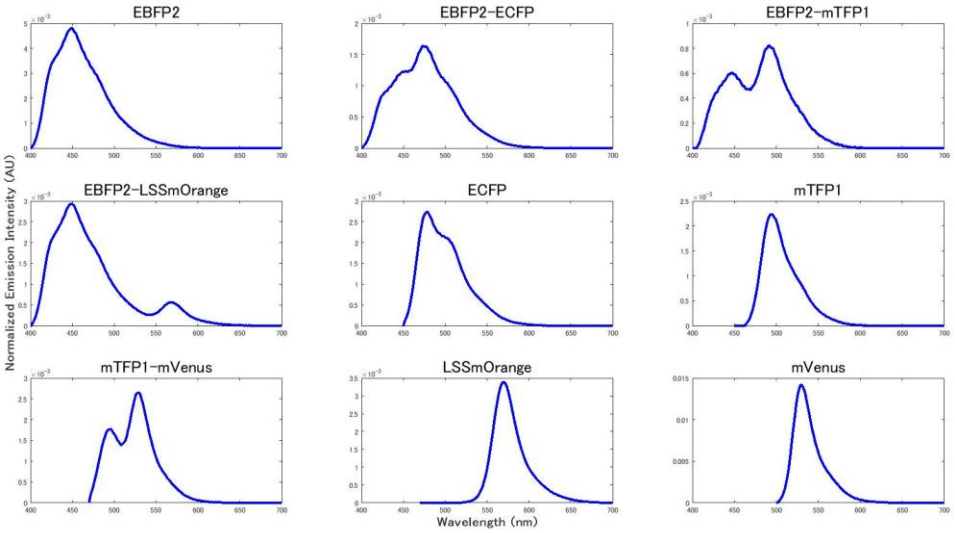

Figure A.5. Plots of reference emission spectra

Appendix B

Since the given raw data contain all detailed information about fluorescent proteins, it is necessary to extract significant information (i.e., fluorescence emission spectra) from the data. By specifying the wavelength to 400nm – 700nm, emission spectra data from the raw data are obtainable. Since each sample was generated in triplicate, the averages across triplicates are taken and then both the reference and mixture data are blank and background subtracted. Lastly, emission intensity for each data is normalized such that the area under the curve is 1. The procedure is described in Figure B.1.

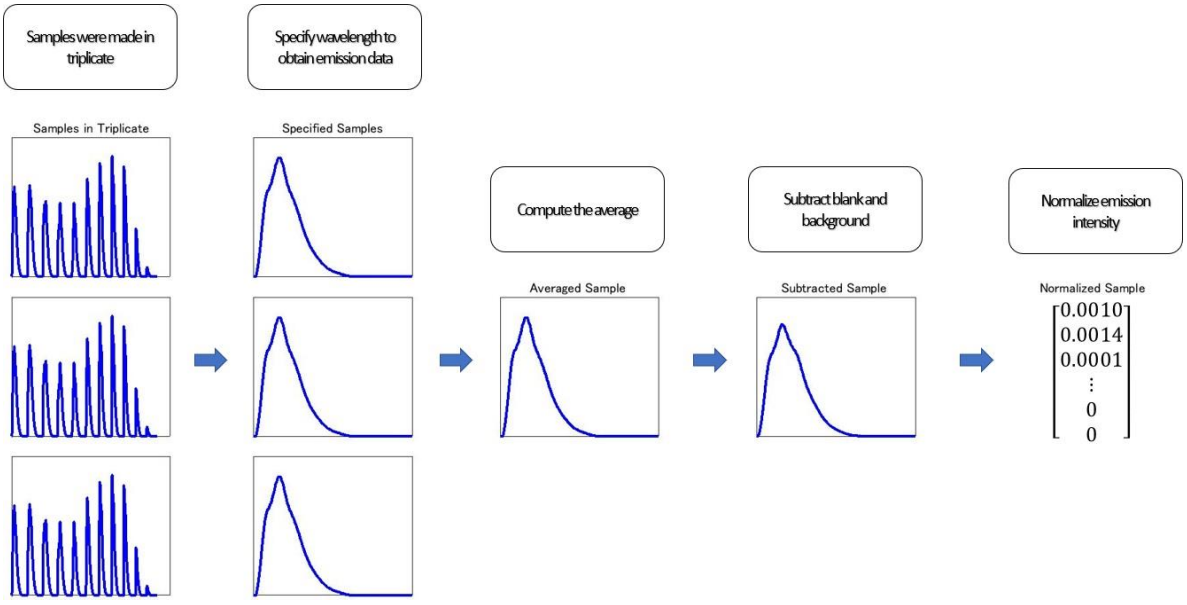

Figure B.1. Description of key steps for data preprocessing

Appendix C

Mixture Sample	SNR (dB)	Mixture Sample	SNR (dB)	Mixture Sample	SNR (dB)
1	1.4249	17	1.5214	33	1.3291
2	1.6456	18	2.2022	34	0.9389
3	1.3533	19	1.3162	35	1.2896
4	2.4121	20	1.3402	36	0.9879
5	2.1299	21	1.0641	37	1.1529
6	1.4989	22	1.0664	38	0.9777
7	1.4291	23	1.3248	39	0.9499
8	1.3450	24	1.4661	40	0.9250
9	1.0356	25	1.2420	41	1.0404
10	1.0679	26	1.2246	42	0.9231
11	1.8909	27	1.9380	43	0.9207
12	2.0555	28	1.3818	44	0.9607
13	1.3095	29	1.0187	45	1.1309
14	1.7373	30	0.9286	46	0.9651
15	1.4168	31	1.5109	47	0.9751
16	1.3785	32	1.1920	48	0.9615

Table C.1. SNR values of mixture emission spectra and random noise with ratio 0.001

Mixture Sample	SNR (dB)	Mixture Sample	SNR (dB)	Mixture Sample	SNR (dB)
1	4.1096	17	4.3631	33	3.9213
2	4.6398	18	5.6997	34	3.0003
3	3.9506	19	3.8675	35	3.8413
4	6.0866	20	3.9423	36	3.1172
5	5.5592	21	3.3265	37	3.5203
6	4.3016	22	3.3168	38	3.1067
7	4.1131	23	3.9248	39	3.0431
8	3.9536	24	4.2229	40	2.9682
9	3.2516	25	3.7405	41	3.2614
10	3.2815	26	3.6721	42	2.9756
11	5.0776	27	5.2072	43	2.9543
12	5.4172	28	4.0317	44	3.0561
13	3.8817	29	3.2028	45	3.4753
14	4.7778	30	2.9786	46	3.0680
15	4.1117	31	4.3524	47	3.0975
16	4.0020	32	3.6042	48	3.0632

Table C.2. SNR values of mixture emission spectra and random noise with ratio 0.0005

Mixture Sample	SNR (dB)	Mixture Sample	SNR (dB)	Mixture Sample	SNR (dB)
1	16.1127	17	16.5641	33	15.8393
2	17.0002	18	18.4724	34	14.2302
3	15.8553	19	15.7192	35	15.7192
4	18.9952	20	15.8682	36	14.4417
5	18.2747	21	14.8552	37	15.1708
6	16.4532	22	14.8166	38	14.4436
7	16.1113	23	15.8614	39	14.3334
8	15.8874	24	16.3208	40	14.1735
9	14.7126	25	15.5622	41	14.7279
10	14.6979	26	15.4101	42	14.2058
11	17.5756	27	17.8003	43	14.1424
12	18.0772	28	16.0093	44	14.3374
13	15.7800	29	14.6135	45	15.1023
14	17.1479	30	14.1952	46	14.3608
15	16.1407	31	16.5610	47	14.4225
16	15.9332	32	15.3048	48	14.3582

Table C.3. SNR values of mixture emission spectra and random noise with ratio 0.0001