HSI DATA UNMIXING USING MACHINE LEARNING TECHNIQUES

CHAO ZHOU

UNIVERSITY COLLEGE LONDON

DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING

Submitted to University College London (UCL) in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy.

Word count: **44861**

Thesis submission date: 24 March 2023

Declaration

I, Chao Zhou, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Hyperspectral image (HSI) unmixing is a challenging research problem that tries to identify the constituent components, known as endmembers, and their corresponding proportions, known as abundances, in the scene by analysing images captured by hyperspectral cameras. Recently, many deep learning based unmixing approaches have been proposed with the surge of machine learning techniques, especially convolutional neural networks (CNN). However, most of these methods rely on the general-purpose CNN structures and it is unclear how to design an efficient network for unmixing purposes. In this work, we first address the structural issue by proposing new unmixing networks that leverage algorithm unrolling techniques to the Alternating Direction Method of Multipliers (ADMM) solver of a constrained sparse regression problem underlying a linear mixture model. However, like many other methods in the literature, there is no guarantee that the network could generate physically meaningful unmixing results. To solve this problem, we proposed a novel blind unmixing network using double DIP techniques (BUDDIP) which consists of two DIP sub-networks to estimate the endmember and abundance respectively, which are coined as EDIP and ADIP. The network is trained in an end-to-end manner by minimizing a novel composite loss function. Finally, we propose a novel unmixing algorithm that can address both issues, simultaneously. Specifically, we first propose a novel MatrixConv Unmixing (MCU) Model for endmember and abundance estimation, respectively, which can be solved via certain iterative solvers. We then unroll these

solvers to build two unfolding based sub-networks, which are coined as UEDIP and UADIP, to generate the estimation of endmember and abundance, respectively. The overall network is then constructed by assembling these two sub-networks. To further improve the unmixing quality, we also add explicitly a regulariser for endmember and abundance estimation, respectively. Experimental results on both synthetic and real HSI data show that the proposed method achieves state-of-the-art performance compared to other unmixing approaches.

Impact Statement

HSI data is a valuable tool for characterizing the materials present in a given scene, but the process of unmixing the data to accurately identify these materials can be complex and challenging. The successful application of machine learning techniques in various research areas has inspired the development of many machine learning-based algorithms for HSI unmixing purposes. However, most of these methods rely on the general-purpose CNN structure and lack the ability to guarantee the generation of physically meaningful unmixing results. Hence, it is necessary to develop new unmixing networks that are designed specifically for HSI unmixing problems and can effectively deliver physically meaningful unmixing results.

In this work, we solve the above problems by proposing a novel framework that can build network structures specifically for unmixing purposes and generate physically meaningful unmixing results by leveraging the guidance provided by existing unmixing algorithms. Our approach has the potential to significantly advance the state-of-the-arts in HSI data analysis, as it allows for more accurate and efficient characterization of the materials present in a given scene. This has numerous practical applications in various domains, such as remote sensing, art investigation and medical imaging, where accurate identification and quantification of materials is critical.

Our methodology will have implications in both academia and industry. In specific, the proposed framework provides a systematic way to build novel neural networks by unrolling the ADMM solver to the

underling physical model. Our results illustrate that it delivers state-of-the-art unmixing performance compared to the competing approaches. In addition, the proposed methodology provides a novel way to incorporate the existing algorithms to guide the training of neural networks, which, after training, can deliver not only physically meaningful results but also better results than the guidance itself. Overall, the work presented in this thesis has the potential to make a significant impact on a wide range of fields and applications, and we believe it represents a valuable contribution to the scientific community.

Research Paper Declaration Form

1. **For a research manuscript that has already been published** (if not yet published, please skip to section 2)**:**

   (a) **What is the title of the manuscript?**

   An ADMM Based Network for Hyperspectral Unmixing Tasks

   (b) **Please include a link to or doi for the work:**

   DOI: 10.1109/ICASSP39728.2021.9414555.

   (c) **Where was the work published?**

   ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

   (d) **Who published the work?**

   IEEE.

   (e) **When was the work published?**

   May 2021.

   (f) **List the manuscript's authors in the order they appear on the publication:**

   Chao Zhou; Miguel R.D. Rodrigues.

   (g) **Was the work peer reviewd?**

   Yes.

   (h) **Have you retained the copyright?**

   Yes.

   (i) **Was an earlier form of the manuscript uploaded to a**

preprint server (e.g. medRxiv)? If Yes, please give a link or doi

If No, please seek permission from the relevant publisher and check the box next to the below statement:

☒ *I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.*

2. **For a research manuscript prepared for publication but that has not yet been published** (if already published, please skip to section 3)**:**

   (a) **What is the current title of the manuscript?**

   (b) **Has the manuscript been uploaded to a preprint server 'e.g. medRxiv'?**
   If 'Yes', please please give a link or doi:

   (c) **Where is the work intended to be published?**

   (d) **List the manuscript's authors in the intended authorship order:**

   (e) **Stage of publication:**

3. **For multi-authored work, please give a statement of contribution covering all authors** (if single-author, please skip to section 4)**:**
   Chao Zhou finished this paper under the supervision of Prof. Miguel R.D. Rodrigues.

4. **In which chapter(s) of your thesis can this material be found?**
   In Chapter 3.

Research Paper Declaration Form

1. **For a research manuscript that has already been published** (if not yet published, please skip to section 2)**:**

   (a) **What is the title of the manuscript?**

   ADMM-Based Hyperspectral Unmixing Networks for Abundance and Endmember Estimation.

   (b) **Please include a link to or doi for the work:**

   DOI: 10.1109/TGRS.2021.3136336.

   (c) **Where was the work published?**

   IEEE Transactions on Geoscience and Remote Sensing.

   (d) **Who published the work?**

   IEEE.

   (e) **When was the work published?**

   December 2021.

   (f) **List the manuscript's authors in the order they appear on the publication:**

   Chao Zhou; Miguel R.D. Rodrigues.

   (g) **Was the work peer reviewd?**

   Yes.

   (h) **Have you retained the copyright?**

   Yes.

   (i) **Was an earlier form of the manuscript uploaded to a**

preprint server (e.g. medRxiv)? If Yes, please give a link or doi

https://arxiv.org/abs/2005.10856.

If No, please seek permission from the relevant publisher and check the box next to the below statement:

☐ *I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.*

2. **For a research manuscript prepared for publication but that has not yet been published** (if already published, please skip to section 3)**:**

   (a) **What is the current title of the manuscript?**

   (b) **Has the manuscript been uploaded to a preprint server 'e.g. medRxiv'?**

   If 'Yes', please please give a link or doi:

   (c) **Where is the work intended to be published?**

   (d) **List the manuscript's authors in the intended authorship order:**

   (e) **Stage of publication:**

3. **For multi-authored work, please give a statement of contribution covering all authors** (if single-author, please skip to section 4)**:**

   Chao Zhou finished this paper under the supervision of Prof. Miguel R.D. Rodrigues.

4. **In which chapter(s) of your thesis can this material be found?**

   In Chapter 3.

Research Paper Declaration Form

1. **For a research manuscript that has already been published** (if not yet published, please skip to section 2)**:**

   (a) **What is the title of the manuscript?**

   Blind Unmixing Using A Double Deep Image Prior.

   (b) **Please include a link to or doi for the work:**

   DOI: 10.1109/ICASSP43922.2022.9747545.

   (c) **Where was the work published?**

   ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

   (d) **Who published the work?**

   IEEE.

   (e) **When was the work published?**

   April 2022.

   (f) **List the manuscript's authors in the order they appear on the publication:**

   Chao Zhou; Miguel R.D. Rodrigues.

   (g) **Was the work peer reviewd?**

   Yes.

   (h) **Have you retained the copyright?**

   Yes.

   (i) **Was an earlier form of the manuscript uploaded to a**

preprint server (e.g. medRxiv)? If Yes, please give a link or doi

If No, please seek permission from the relevant publisher and check the box next to the below statement:

☒ *I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.*

2. **For a research manuscript prepared for publication but that has not yet been published** (if already published, please skip to section 3)**:**

   (a) **What is the current title of the manuscript?**

   (b) **Has the manuscript been uploaded to a preprint server 'e.g. medRxiv'?**
   If 'Yes', please please give a link or doi:

   (c) **Where is the work intended to be published?**

   (d) **List the manuscript's authors in the intended authorship order:**

   (e) **Stage of publication:**

3. **For multi-authored work, please give a statement of contribution covering all authors** (if single-author, please skip to section 4)**:**
   Chao Zhou finished this paper under the supervision of Prof. Miguel R.D. Rodrigues.

4. **In which chapter(s) of your thesis can this material be found?**
   In Chapter 4.

Research Paper Declaration Form

1. **For a research manuscript that has already been published** (if not yet published, please skip to section 2)**:**

   (a) **What is the title of the manuscript?**

   Hyperspectral Blind Unmixing using a Double Deep Image Prior.

   (b) **Please include a link to or doi for the work:**

   DOI: 10.1109/TNNLS.2023.3294714.

   (c) **Where was the work published?**

   IEEE Transactions on Neural Networks and Learning Systems.

   (d) **Who published the work?**

   IEEE.

   (e) **When was the work published?**

   August 2023.

   (f) **List the manuscript's authors in the order they appear on the publication:**

   Chao Zhou; Miguel R.D. Rodrigues.

   (g) **Was the work peer reviewd?**

   Yes.

   (h) **Have you retained the copyright?**

   Yes.

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If Yes, please give a link or doi**

If No, please seek permission from the relevant publisher and check the box next to the below statement:

☒ *I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.*

2. **For a research manuscript prepared for publication but that has not yet been published** (if already published, please skip to section 3)**:**

   (a) **What is the current title of the manuscript?**

   (b) **Has the manuscript been uploaded to a preprint server 'e.g. medRxiv'?**

   If 'Yes', please please give a link or doi:

   (c) **Where is the work intended to be published?**

   (d) **List the manuscript's authors in the intended authorship order:**

   (e) **Stage of publication:**

3. **For multi-authored work, please give a statement of contribution covering all authors** (if single-author, please skip to section 4)**:**

   Chao Zhou finished this paper under the supervision of Prof. Miguel R.D. Rodrigues.

4. **In which chapter(s) of your thesis can this material be found?**

   In Chapter 4 and Chapter 6.

Research Paper Declaration Form

1. **For a research manuscript that has already been published** (if not yet published, please skip to section 2)**:**

   (a) **What is the title of the manuscript?**

   (b) **Please include a link to or doi for the work:**

   (c) **Where was the work published?**

   (d) **Who published the work?**

   (e) **When was the work published?**

   (f) **List the manuscript's authors in the order they appear on the publication:**

   (g) **Was the work peer reviewd?**

   (h) **Have you retained the copyright?**

   (i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If Yes, please give a link or doi**

   If No, please seek permission from the relevant publisher and check the box next to the below statement:

   ☐ *I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.*

2. **For a research manuscript prepared for publication but that**

has not yet been published (if already published, please skip to section 3)**:**

(a) **What is the current title of the manuscript?**

Neural Network for Blind Unmixing: a novel MatrixConv Unmixing (MCU) Model.

(b) **Has the manuscript been uploaded to a preprint server 'e.g. medRxiv'?**

If 'Yes', please please give a link or doi:

No.

(c) **Where is the work intended to be published?**

IEEE Transactions on Geoscience and Remote Sensing.

(d) **List the manuscript's authors in the intended authorship order:**

Chao Zhou; Wei Pu, Miguel R.D. Rodrigues.

(e) **Stage of publication:**

In submission.

3. **For multi-authored work, please give a statement of contribution covering all authors** (if single-author, please skip to section 4)**:**

Chao Zhou finished this paper under the supervision of Prof. Miguel R.D. Rodrigues. Dr. Wei Pu assisted in the proofreading process of this manuscript.

4. **In which chapter(s) of your thesis can this material be found?**

In Chapter 5.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Prof. Miguel R.D. Rodrigues, for his unwavering support, guidance, and mentorship throughout my doctoral journey. His expertise, enthusiasm, and encouragement have been instrumental in shaping my research direction, refining my analytical skills, and broadening my academic horizons.

I am grateful to my colleagues and collaborators, Wei Pu, Zhaoyan Lyu, Afroditi Papadaki, Martin Ferianc, Jaweria Amjad and countless others, for their camaraderie, inspiration, and intellectual exchange, which have made my academic experience more enjoyable, fulfilling, and rewarding.

I would like to acknowledge the financial support from China Scholarship Council (CSC), which has provided me with the resources and opportunities to pursue my research goals and to attend conferences and workshops.

Last but not least, I would like to express my heartfelt appreciation to my family and friends, for their unconditional love, encouragement, and moral support, which have sustained me during the challenging times and celebrated my achievements and milestones.

Thank you all for your contributions to my academic and personal growth, and for being part of this incredible journey.

Contents

Table of Acronyms

Notation	Description
AA	Archetypal Analysis.
AAD	Abundance Angle Distance.
AID	Abundance Information Divergence.
ADAM	Adaptive Moment Estimation.
ADMM	Alternating Direction Method of Multipliers.
AE	Abundance Estimation.
ANC	Abundance Non-negative Constraint.
ASC	Abundance sum-to-one Constraint.
AWGN	Additive White Gaussian Noise.
BSS	Blind Source Separation.
CNN	Convolutional Neural Networks.
CSC	convolutional sparse coding.
CSR	Constrained Sparse Regression.
DIP	Deep Image Prior.
DNN	Deep Neural Networks.
EE	Endmember Estimation.
FM	Fan Model.
HSI	Hyperspectral Image.
ISTA	Iterative Shrinkage-Thresholding Algorithm.
LMM	Linear Mixing Model.

Notation	Description
MSE	Mean Square Error.
NLMM	Non-Linear Mixing Model.
RED	Regularizer by Denoising.
ReLU	Rectified Linear Unit.
RMSE	Root Mean Square Error.
SAD	Spectral Angle Distance.
SNR	Signal-to-Noise Ratio.

Table of Symbols

Notation	Description
p	number of spectral bands.
r	number of endmembers.
n	number of pixels.
J	number of Iterations/Layers.
K	number of training epochs..
N	the size of training set.
$\mathbf{Y}$	the HSI reflectance for all n pixels.
$\mathbf{E}$	the endmember matrix.
$\mathbf{A}$	the abundance for all n pixels.
$\mathbf{N}$	the AWGN noise for all n pixels.
$\mathbf{O}$	the nonlinear mixing terms for all n pixels.
$\mathbf{y}_i$	the HSI reflectance for i^{th} pixel.
$\mathbf{a}_i$	the abundance vector for i^{th} pixel.
$\mathbf{e}_i$	the signature of i^{th} endmember.
$\mathbf{n}_i$	the AWGN noise for i^{th} pixel.
$\mathbf{x}^j$	the value of variable $\mathbf{x}$ at j^{th} iteration/layer.
$\mathbf{1}_r$	all-one vector with size $r \times 1$.
$\mathbf{\Theta}$	the set of learnable parameters for the whole network.
$\mathbf{\Theta}^j$	the learnable parameters at j^{th} layer.
$R(\mathbf{X})$	the regulariser for variable $\mathbf{X}$.
$f_{\boldsymbol{\theta}}$	a network with learnable parameters $\boldsymbol{\theta}$.

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Hyperspectral imaging (HSI), also known as imaging spectroscopy, is a technique that captures and analyzes the light reflected or emitted from a scene or object [3]. It has been widely applied in fields such as remote sensing [4], mineralogy [5], environmental monitoring [6] and art investigation [7]. Unlike traditional imaging methods, which capture just several narrow band of wavelengths, an HSI image generally contains hundreds of narrow wavelength bands, covering a wide range of wavelengths across the electromagnetic spectrum, typically from the visible to the infrared range [8], which is depicted in Fig. 1.1. This results in a "spectral signature" for each pixel in the image, allowing for the preservation of important information in the spectrum, such as the shape of narrow absorption bands. This rich spectral information makes it possible to identify and analyze a wide range of materials and substances that are invisible or difficult to detect using other imaging methods, without physical contact. This relies on the fact that different materials within a scene reflect electromagnetic radiation differently, so that, when the radiation captured by sensors is measured at each wavelength over a sufficiently broad spectral band, the resulting spectral signature can be used to uniquely identify and characterize the constituent materials within the scene [9].

Due to the low spatial resolution of hyperspectral sensors and the

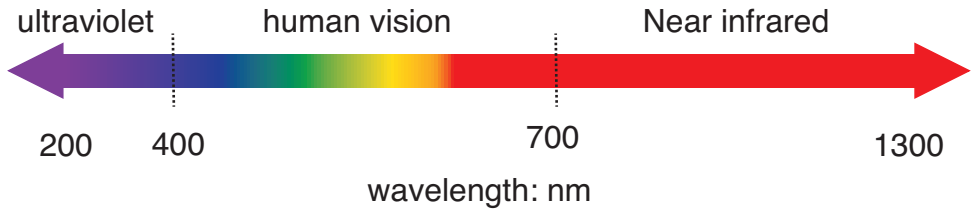


Figure 1.1: Human Vision *vs.* Hyperspectral Camera. HSI can capture richer spectral information than human vision.

heterogeneity of the landscape, the spectral vectors obtained are not pure, but rather a mixture of the spectral signatures of the materials present in the scene [8]. This calls for methods capable of quantitatively decomposing, or unmixing, the captured spectral signature onto the pure spectral signatures of the constituent materials in the scene also known as endmembers and their proportions within the mixture also known as abundances [10].

As outlined in [8], the HSI unmixing problem generally consists of two main tasks: (a) endmember estimation (EE) and (b) abundance estimation (AE). EE algorithms focus on identifying endmember spectral signatures that are present within the HSI image. Many EE methods are geometrically-based approaches, which assume that the data is embedded in a simplex, the vertices of which are the endmembers. Popular examples of such methods include vertex component analysis (VCA) [11] and simplex volume maximization (SiVM) [1]. AE algorithms, on the other hand, aim to determine the proportion of each endmember within each pixel in the HSI image. There are popular AE methods such as fully constrained least square (FCLS) [2] which assumes the endmembers are known beforehand, for example, extracted by the EE methods. In addition, there are also methods known as blind unmixing methods [12], which perform endmember estimation and abundance estimation simultaneously. For example, nonnegative matrix factorization (NMF) [13–16] and nonnegative tensor factorization (NTF) [17] are very popular blind unmixing methods that map

the HSI unmixing problem onto a matrix/tensor factorization problem by imposing various constraints, such as total-variation constraint, on endmember signatures or their abundances. However, these traditional model-based approaches can be computationally complex.

More recently, there has been a surge of interest in the application of machine learning, particularly neural networks, for addressing the HSI unmixing challenge [18–23]. These approaches can be broadly divided into two categories: supervised learning-based and unsupervised learning-based methods. Supervised learning-based approaches assume the availability of a labeled dataset, consisting of pairs of HSI reflectance and corresponding abundance [24–26]. Whereas, unsupervised learning-based approaches [27–33] instead attempt to learn a function that can estimate both the endmembers and abundances from the HSI reflectance data alone. These blind unmixing methods are usually based on an autoencoder network structure with a linear decoder, which takes only the HSI spectra as input and enforces the output to reconstruct HSI spectra. Despite the promising developments in this area, it is still unclear how to design network architectures specifically for the purpose of unmixing. Moreover, without proper guidance, deep learning-based methods may not generate physically meaningful unmixing results [4].

Recently, algorithm unrolling or unfolding techniques [34, 35] have been proposed as a potential solution to design interpretable network structures. With these techniques, the unmixing task is first modelled as an optimization problem, which is typically solved by an iterative solver. Each step of the iterative solver is then converted to a network operation and the network is constructed by concatenating several iterations (or layers) of such operations. For example, [18, 26] proposed unmixing networks, MNN-AE and MNN-BU, which apply the unfolding technique to an iterative shrinkage-thresholding algorithm (ISTA)

for HSI unmixing. However, ISTA based unmixing networks [18, 26] are not very efficient and there are many iterative solvers other than ISTA. To avoid generating physically meaningless unmixing results, some recent works like UnDIP [36] have proposed to use existing algorithms like simplex volume maximization to extract endmembers and use them to guide the training of an abundance estimation network using a deep image prior (DIP). However, existing unmixing networks trained with guidance can be limited by the quality of the guidance and most learning algorithms lack the ability to generalize from linear unmixing problems to nonlinear ones because they are based on the autoencoder structure to solve either linear or nonlinear problems.

This work endeavors to investigate the following questions in the field of HSI unmixing:

- *Can we improve the unmixing performance by leveraging other iterative algorithms in algorithm unrolling approaches to deliver state-of-the-art unmixing neural network architectures?*
- *Can we surpass the performance limitations of existing guidance based unmixing approaches?*
- *Can we design an unmixing network that can be applied to both linear and nonlinear blind unmixing problems?*

This work aims to bridge the gap in the current literature by addressing these questions, and contribute to the advancement of more efficient and effective techniques for HSI unmixing.

1.1 Contributions

In particular, we make the following contributions:

1. We propose a novel neural network architecture for HSI linear unmixing, which builds upon the sparsity-driven model and is de-

rived from unrolling an ADMM algorithm. The network is trained using supervised principles and a composite loss function that incorporates additional terms such as an abundance angle distance (AAD) and an abundance information divergence (AID) term. The proposed network has a richer and lighter structure compared to state-of-the-art methods, achieving faster convergence, better unmixing performance, robustness, and interpretability.

2. Furthermore, we propose an autoencoder-like neural network trained using unsupervised principles to yield both endmembers and abundances directly from HSI data. The proposed blind unmixing network offers improved performance compared to state-of-the-art algorithms.

3. We propose a general framework, BUDDIP, for linear and nonlinear blind unmixing using deep image prior (DIP) techniques [37]. BUDDIP consists of three modules: an endmember estimation DIP (EDIP) module, an abundance estimation DIP (ADIP) module, and a mixing module (MM). Unlike other DIP-based methods [36, 37] that use random input, BUDDIP uses a meaningful input by leveraging existing unmixing methods, resulting in more efficient network structures and improved performance.

4. Furthermore, we propose a composite loss function for BUDDIP that ensures physically meaningful unmixing results and outperforms existing guidance based methods. Additionally, for nonlinear unmixing, we introduce an adaptive loss weight strategy for better results.

The above contributions have resulted in the following manuscripts during the course of the PhD study:

1. C. Zhou and M. R. D. Rodrigues, "An ADMM Based Network

for Hyperspectral Unmixing Tasks," ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 2021, pp. 1870-1874, doi: 10.1109/ICASSP39728.2021.9414555.

2. C. Zhou and M. R. D. Rodrigues, "ADMM-Based Hyperspectral Unmixing Networks for Abundance and Endmember Estimation," in IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1-18, 2022, Art no. 5520018, doi: 10.1109/TGRS.2021.3136336.

3. C. Zhou and M. R. D. Rodrigues, "Blind Unmixing Using A Double Deep Image Prior," ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, Singapore, 2022, pp. 1665-1669, doi: 10.1109/ICASSP43922.2022.9747545.

1.2 Organisation

This thesis is organised as follows:

In Chapter 2, we provide relevant background information about the HSI unmixing problem. We introduce both linear and nonlinear mixing models and discuss traditional model-based approaches as well as modern learning-based approaches. Furthermore, we examine the challenges in current research on HSI unmixing to provide a comprehensive understanding of the topic.

In Chapter 3, we propose a novel HSI unmixing algorithm that combines both model- and learning-based approaches. Specifically, we employ algorithm unrolling techniques to the Alternating Direction Method of Multipliers (ADMM) solver of a constrained sparse regression problem underlying a linear mixture model.

We then introduce a neural network structure for abundance estimation that can be trained using supervised learning techniques with a composite loss function. Additionally, we propose another neural network structure for blind unmixing that can be trained using unsupervised learning techniques.

In Chapter 4, we propose a general unsupervised framework inspired by Deep-Image-Prior (DIP), which is applicable to both linear and nonlinear blind unmixing problems. Specifically, our framework involves three modules: (1) an Endmember estimation module using a DIP (EDIP), (2) an Abundance estimation module using a DIP (ADIP), and (3) a Mixing module (MM). The EDIP and ADIP modules are responsible for generating endmembers and abundances respectively, whereas the MM, which is constructed based on the postulated mixing model, generates a reconstruction of the HSI observations. In order to generate meaningful unmixing results, we also propose a composite loss function, which, again, applies to both linear and nonlinear unmixing models. We also propose an adaptive loss weight strategy to yield better unmixing results in nonlinear mixing scenarios.

In Chapter 5, we offer several variations of BUDDIP aimed at further enhancing its performance. One approach involves constructing the EDIP and ADIP structure through the incorporation of unfolding techniques applied to a novel MatrixConv Unmixing (MCU) Model. Another approach entails the addition of explicit regularizations to the network, resulting in improved performance.

In Chapter 6, we evaluate the effectiveness of the proposed methods by comparing them with some state-of-the-art approaches on three real HSI datasets.

In Chapter 7, we provide a summary of this thesis and suggest future directions for research.

1.3 Notation

In this thesis, we use lowercase letters, bold lowercase letters, and bold capital letters to represent scalars, vectors, and matrices respectively. Specifically, a scalar is represented by x, a vector is represented by $\mathbf{x}$, and a matrix is represented by $\mathbf{X}$. $\mathbf{x}_k$ correspond to the k^{th} column vector of matrix $\mathbf{X}$. $x_{i,j}$ correspond to the element of i^{th} row and j^{th} column of matrix $\mathbf{X}$. The symbol $*$ represents the convolutional operator, while $\times$ represents the matrix multiplication operator. The symbol $\odot$ represents the element-wise (Hadamard) product. The Frobenius norm of matrix $\mathbf{X}$ is represented as $\|\mathbf{X}\|_F$, and the ℓ_1 norm of matrix $\mathbf{X}$ is represented as $\|\mathbf{X}\|_1$. The transform of matrix $\mathbf{X}$ is represented as $\mathbf{X}^T$.

Chapter 2

Background

In this Chapter, we first define the HSI unmixing problem and introduce the general mixing model: Linear mixing model (LMM) and Nonlinear mixing model (NLMM). We then briefly review various model-based and learning-based unmixing approaches in the literature.

2.1 HSI unmixing Problem

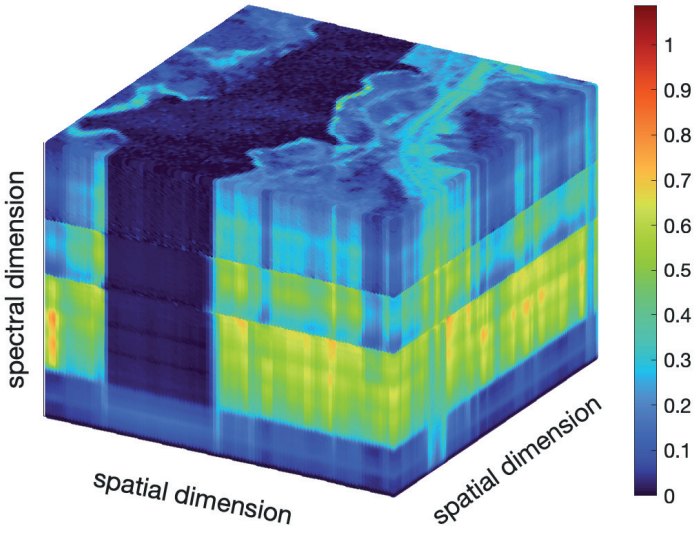


Figure 2.1: HSI Data cube. It can be visualised as a collection of images, where each image represents the radiance measured in a particular spectral band.

Hyperspectral imaging (HSI) can be interpreted as the acquisition of a stack of images, each of which represents the radiance in a spe-

cific band [3]. These images form a data cube, as shown in Fig. 2.1, where each pixel provides a spectrum that characterizes the materials within that pixel. The objective of HSI is to identify and segregate materials based on their unique reflective properties when observed over a wide range of wavelengths. However, the observed reflectance is typically a mixture of the spectral signatures of the materials present in the scene, due to the heterogeneity of the scene [8]. This necessitates the use of methods capable of quantitatively decomposing, or unmixing, the captured spectral signature into its spectral constituents, also known as "endmembers", and their corresponding proportions within the mixture, also known as "abundances" [9]. In the application of remote sensing, as illustrated in Fig. 2.2, endmembers can include various materials such as Tree, Water, Dirt, Road, etc [9].

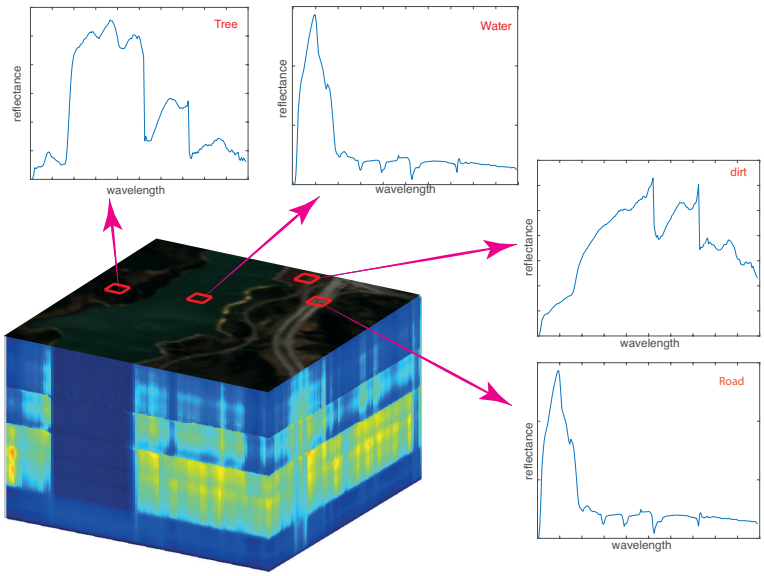


Figure 2.2: Hyperspectral imaging spectroscopy in remote sensing. The spectra of each pixel provide a characterization of the materials present within the corresponding pixel.

In general, HSI unmixing analysis involves three main steps as illustrated in Fig. 2.3:

Dimension reduction. Hyperspectral data is frequently characterized by high dimensionality, as a result of the oversampling of the spectral and spatial signal performed by the sensors to ensure

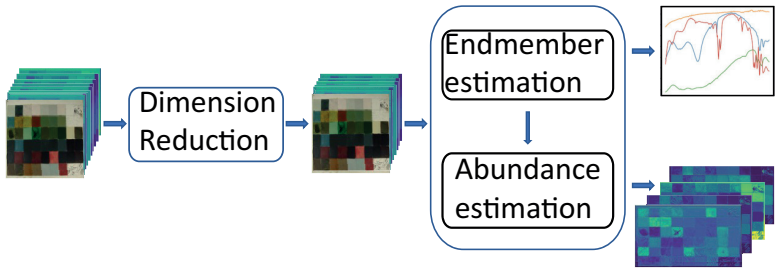


Figure 2.3: Procedures of spectral unmixing analysis

the capturing of any narrow features. In order to mitigate this issue, a pre-processing step of dimensionality reduction is required, which can be achieved through either feature selection or extraction techniques. This dimensionality reduction process is beneficial in terms of reducing the computational overhead associated with subsequent analyses. The most widely employed reduction algorithms [8] include Principal Component Analysis (PCA) and Minimum Noise Fraction (MNF).

Endmember estimation. The second step of HSI unmixing analysis concerns the identification of an appropriate vector basis to describe all the materials present in the image. The literature on this topic presents a variety of approaches [8], which can be broadly classified into two groups. The first group of approaches aims to find the most extreme spectra, which are typically the purest and best describe the vertices of the data simplex. The second group of approaches, on the other hand, looks for the spectra that are the most statistically distinct.

Abundance estimation. The final step of HSI unmixing analysis is the estimation of the proportions of materials, known as abundance, present in each image pixel through the process of model inversion. This step typically involves the use of linear or nonlinear regression techniques. A wide range of methods, including linear regression, neural networks and support vector regres-

sion, are commonly employed for this purpose. The inversion step is based on solving a constrained least squares problem, which aims to minimize the residual between the observed spectral vectors and the reconstructions based on the endmembers.

In the thesis, the focus is mainly on the second and third steps of HSI unmixing analysis. Dimensionality reduction is not explored in detail because it is a common data pre-processing step in all signal processing problems and is considered as a well-established step in the literature.

2.2 Mixing Models

Analytical models for the mixing of disparate materials serve as the foundation for the development of techniques to estimate the endmembers and corresponding abundances from mixed pixels. Mixture modelling is based on the assumption that within a given scene, the surface is dominated by a limited number of distinct materials that possess relatively constant spectral properties [9]. In light of this principle, various mixing models have been proposed in the literature, which can be broadly classified into two categories: 1) Linear mixing models (LMM); 2) Nonlinear mixing models (NLMM). In the following, we will provide a brief introduction to these two models.

2.2.1 Linear mixing model (LMM)

The linear model is often the first and simplest model employed in various research problems [8]. In this context, the reflectance model is represented as a checkerboard mixture, where any incident radiation interacts only with one component, as depicted in Fig. 2.4. If the surface of the corresponding pixel radiation reflectance can be viewed as a combination proportional to the abundance of each endmember, the

reflectance will convey the characteristics of each endmember present in that pixel with the same proportion. In this setup, there is a linear relationship between the fractional abundance of materials present in the pixel and the spectra in the reflected radiation [9]. In other words, the spectrum of a mixed pixel is a linear combination of the endmember spectra weighted by their corresponding fractional abundance in the pixel. Therefore, it is referred to as linear mixing model (LMM).

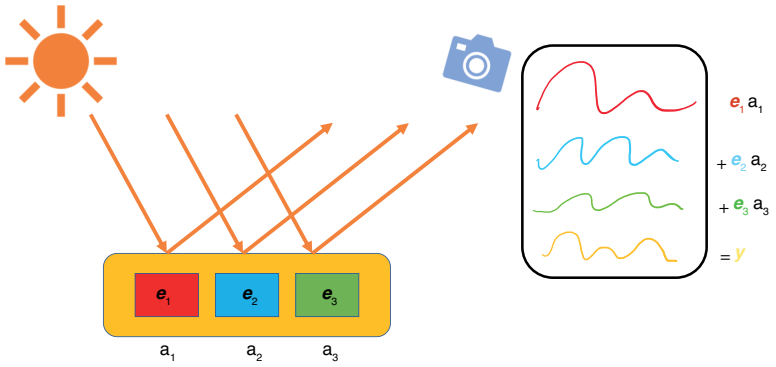


Figure 2.4: Illustration of linear mixing process

Mathematically, the LMM can be described as:

$$\mathbf{y} = \mathbf{E}\mathbf{a} + \mathbf{n} \tag{2.2.1}$$

where $\mathbf{y} \in \mathbb{R}^{p\times 1}$ is the reflectance for a specific pixel captured with p spectral bands. The endmember matrix, $\mathbf{E} = [\mathbf{e}_1, ..., \mathbf{e}_r] \in \mathbb{R}^{p\times r}$, contains the spectral signatures of r endmembers across p spectral bands, with each endmember's spectral signature represented by $\mathbf{e}_i \in \mathbb{R}^{p\times 1}$ where $i = 1, ..., r$. $\mathbf{a} \in \mathbb{R}^{r\times 1}$ is the corresponding abundance vector containing the abundances of r different endmembers present in the pixel. $\mathbf{n} \in \mathbb{R}^{p\times 1}$ models the additive white Gaussian noise.

Under matrix notation, Eq. (2.2.1) can be re-formulated as follows:

$$\mathbf{Y} = \mathbf{E}\mathbf{A} + \mathbf{N} \tag{2.2.2}$$

where $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_n] \in \mathbb{R}^{p\times n}$ is the HSI observations, which contains the reflectance spectra of n pixels across p spectral bands. The i^{th}

pixel's spectra is represented by $\mathbf{y}_i \in \mathbb{R}^{p\times 1}$. $\mathbf{A} = [\mathbf{a}_1, ..., \mathbf{a}_n] \in \mathbb{R}^{r\times n}$ is the corresponding fractional abundance matrix, i.e., $\mathbf{a}_i \in \mathbb{R}^{r\times 1}$ is the abundance vector containing the abundances of r different endmembers present in the i^{th} pixel; and $\mathbf{N} \in \mathbb{R}^{p\times n}$ is the additive white Gaussian noise. Typically, the value of r is in the order of tens, the value of p is in the order of hundreds, and the value of n is in the order of ten thousand. It is important to note that, unless otherwise specified, for the sake of simplicity in notation, the HSI data cube $\mathbf{Y}$, abundance map $\mathbf{A}$, and noise $\mathbf{N}$ discussed in this thesis are represented in their flattened forms, i.e. $\mathbf{Y} \in \mathbb{R}^{p\times n}$, $\mathbf{A} \in \mathbb{R}^{r\times n}$, and $\mathbf{N} \in \mathbb{R}^{p\times n}$. However, in reality, the HSI image being analyzed is of size $n_1 \times n_2$, i.e. $\mathbf{Y} \in \mathbb{R}^{p\times n_1\times n_2}$, $\mathbf{A} \in \mathbb{R}^{r\times n_1\times n_2}$, and $\mathbf{N} \in \mathbb{R}^{p\times n_1\times n_2}$. This distinction should be kept in mind when interpreting the results presented in this thesis.

Generally, the abundance is constrained by two properties: the non-negative constraint (ANC) and the sum-to-one constraint (ASC). The ANC requires that all elements of the abundance matrix, $\mathbf{A}$, are non-negative, i.e. $\mathbf{A} \geq \mathbf{0}$. The ASC requires that the abundance of each pixel sums up to one, i.e. $\mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n$, where $\mathbf{1}_r$ is an all-one vector with size $r \times 1$. The imposition of the ASC constraint is driven by the desire to provide a plausible description of the mixture components for each pixel in the image. It emphasizes that the observed reflectance spectrum is fully composed of endmember reflectance spectra. Similarly, the endmember matrix, $\mathbf{E}$, is also subject to the non-negative constraint (ENC) to ensure physical meaning, i.e. $\mathbf{E} \geq \mathbf{0}$. Although sharing similarities with blind source separation, the model (2.2.2) however possesses unique characteristics and difficulties due to physical constraints and endmember dependence [8]. Moreover, the unmixing problem is often ill-posed, lacking a unique solution, and the variability in spectral signatures caused by various factors complicates accurate endmember identification and modeling. Nonlinear spectral mix-

ing processes further contribute to the complexity of the problem.

The objective of blind unmixing is to estimate the appropriate endmember signature, $\mathbf{E}$, and fractional abundance matrix, $\mathbf{A}$, based on the observed reflectance vector, $\mathbf{Y}$, which can be formulated as optimisation problems as follows:

$$\min_{\mathbf{E},\mathbf{A}} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{A}\|_F^2 \quad s.t., \mathbf{E} \geq \mathbf{0}, \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \tag{2.2.3}$$

In some cases, the endmember signature, $\mathbf{E}$, is known in advance in the form of a spectral library and the estimation objective is restricted to determining the abundance, $\mathbf{A}$. The linear mixing process is illustrated in Fig. 2.4.

The simplicity of such model has led to the development of numerous algorithms [18, 29, 31, 32]. However, it is not suitable for handling more complex scenarios, where the reflectance acquired by sensors is the result of interactions with multiple materials at various depths or layers [8].

2.2.2 Nonlinear mixing model (NLMM)

As an alternative to the LMM, the nonlinear mixing model (NLMM) has been proposed as a model that accounts for nonlinear effects by incorporating additional nonlinear interaction terms into the LMM. Typically, an NLMM can be represented as follows:

$$\mathbf{Y} = \mathbf{E}\mathbf{A} + \mathbf{O} + \mathbf{N} \tag{2.2.4}$$

Where $\mathbf{Y}, \mathbf{E}, \mathbf{A}, \mathbf{N}$ are equivalent to the same quantities appearing in Eq. (2.2.2), and $\mathbf{O}$ denotes the additional term accounting for nonlinear mixing effects. Eq. (2.2.4) is also known as the robust Linear Mixing Model (rLMM), and the term $\mathbf{O}$ is used to denote the outlier terms in [13]. The goal of blind nonlinear unmixing is to estimate $\mathbf{E}$, $\mathbf{A}$,

and sometimes $\mathbf{O}$, given only $\mathbf{Y}$, which can be solved by minimizing the following objective function [13]:

$$\begin{aligned} &\min_{\mathbf{E},\mathbf{A},\mathbf{O}} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{A} - \mathbf{O}\|_F^2 + R(\mathbf{O}) \\ &s.t., \mathbf{E} \geq \mathbf{0}, \mathbf{O} \geq \mathbf{0}, \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \end{aligned} \tag{2.2.5}$$

where $R(\mathbf{O})$ is a regularizer imposed on the nonlinear component $\mathbf{O}$.

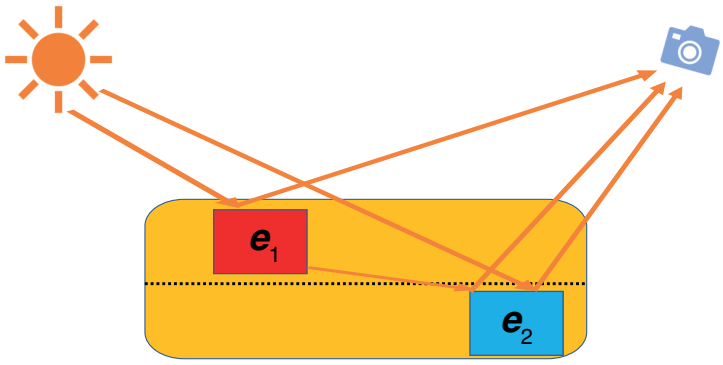


Figure 2.5: Bilinear model

A variety of NLMM has been proposed in the literature of hyperspectral imaging, which differs from each other based on how the term $\mathbf{O}$ is modelled. For instance, the Bilinear model is a variant of NLMM that aims to capture the nonlinear interactions that occur in multilayered scenes [38]. As shown in Fig. 2.5, the reflectance captured by the sensor comes not only from the interaction within a single component but also from the interaction between components. This model represents the observed spectra $\mathbf{y}_k$ for the k^{th} pixel as follows:

$$\mathbf{y}_k = \mathbf{E}\mathbf{a}_k + \sum_{i=1}^{r-1}\sum_{j=i+1}^{r} \beta_{i,j,k}\mathbf{e}_i \odot \mathbf{e}_j + \mathbf{n}_k \tag{2.2.6}$$

where, $\mathbf{y}_k, \mathbf{a}_k, \mathbf{n}_k$ correspond to the k^{th} column vector of $\mathbf{Y}, \mathbf{A}, \mathbf{N}$ in Eq. (2.2.4), respectively. The symbol $\odot$ represents the Hadamard (element-wise) product. The coefficient $\beta_{i,j,k}$ captures the degree of nonlinear interactions between the endmembers $\mathbf{e}_i$ and $\mathbf{e}_j$. Different constraints on the coefficients $\mathbf{a}_k, \beta_{i,j,k}$ have been studied in the literature. For example, the Fan Model (FM), which is the primary focus in this work, is

proposed in [39] by imposing the following constraints:

$$\begin{cases} \mathbf{a}_k \geq \mathbf{0} \\ \sum_{i=1}^{r} a_{i,k} = 1 \\ \beta_{i,j,k} = a_{i,k} a_{j,k} \end{cases} \tag{2.2.7}$$

A model known as the Nascimento Model (NM) [40] is proposed, which imposes the constraints as follows:

$$\begin{cases} \mathbf{a}_k \geq \mathbf{0}, \\ \beta_{i,j,k} \geq 0, \quad \forall i \neq j \\ \sum_{i=1}^{r} a_{i,k} + \sum_{i=1}^{r-1} \sum_{j=i+1}^{r} \beta_{i,j,k} = 1 \end{cases} \tag{2.2.8}$$

This model considers $\mathbf{e}_i \odot \mathbf{e}_j$ as additional pure endmembers. However, the model above does not generalize the Linear Mixture Model (LMM). To overcome this limitation, the Generalized Bilinear Model (GBM) [41] is proposed as a generalization of both the LMM and the Bilinear Model. Mathematically, in GBM, the observed spectrum of a pixel $\mathbf{y}_k$ can be written as follows:

$$\mathbf{y}_k = \mathbf{E}\mathbf{a}_k + \sum_{i=1}^{r-1} \sum_{j=i+1}^{r} \gamma_{i,j,k} a_{i,k} a_{j,k} \mathbf{e}_i \odot \mathbf{e}_j + \mathbf{n}_k \tag{2.2.9}$$

where the interaction coefficient $\gamma_{i,j,k} \in (0, 1)$ quantifies the nonlinear interaction between the spectral components $\mathbf{e}_i$ and $\mathbf{e}_j$. The abundance vector $\mathbf{a}_k$ is also subjected to ANC and ASC.

The Modified Generalized Bilinear Model (MGBM) [42, 43] is proposed to address the lack of consideration for within-endmember interactions $\mathbf{e}_i \odot \mathbf{e}_i$ in the above bilinear models. The MGBM describes the observed spectrum of a pixel $\mathbf{y}_k$ as follows:

$$\mathbf{y}_k = \mathbf{E}\mathbf{a}_k + \sum_{i=1}^{r} \sum_{j=i}^{r} \zeta_{i,j,k} \mathbf{e}_i \odot \mathbf{e}_j + \mathbf{n}_k \tag{2.2.10}$$

where, $\zeta_{i.j,k} \in [0, 1]$. The abundance vector $\mathbf{a}_k$ is also subject to ANC and ASC.

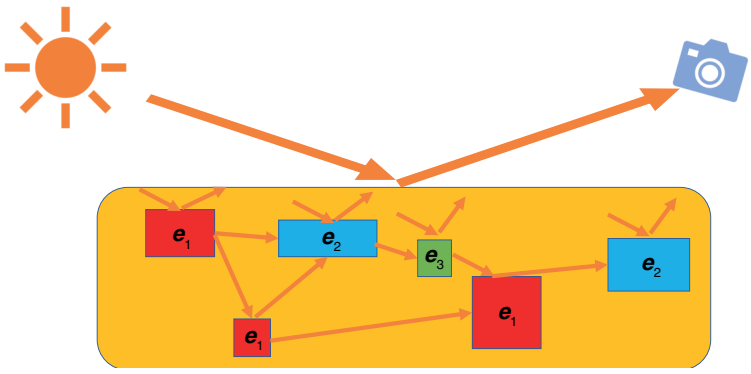


Figure 2.6: Intimate model

The intimate model, as illustrated in Fig. 2.6, is another widely adopted nonlinear mixing model used to model scenarios where endmembers are mixed at a scale smaller than the path of photons in the mixture. The most effective methods for solving this model are those proposed by Hapke in [44], due to their physically meaningful quantities. For further details on this model and its applications, please refer to [45, 46].

2.3 Model-based unmixing approaches

As previously discussed in Sec. 2.1, the process of hyperspectral image (HSI) unmixing typically involves the extraction of endmembers and the estimation of their corresponding abundances. This has led to the development of various model-based approaches in the literature. In this section, we conduct a literature review on these model-based approaches. We begin by examining approaches that focus on the estimation of endmembers [1, 11, 47–50]. We then present approaches that focus on the estimation of abundances [12, 51]. Finally, we conclude this section by introducing blind unmixing methods such as non-negative matrix factorization (NMF) [16] and Entropic Descent Archetypal Analysis (EDAA) [52], which simultaneously estimate both

endmembers and abundances.

2.3.1 Endmember estimation methods

The process of endmember estimation in HSI unmixing typically consists of two sub-steps: (1) determining the number of endmembers, r, present in the HSI data, and (2) estimating the r distinct endmember signatures from the HSI data. In literature, it is often assumed that the number of endmembers, r, is known. For further information on methods for determining r from HSI data, please refer to [8]. In terms of endmember estimation methods, these can be classified into two categories: geometry-based approaches and statistics-based approaches. The categorization of different types of endmember estimation algorithms is shown in Fig. 2.7.

Geometry-based endmember estimation algorithms are based on the principle that Hyperspectral Imaging (HSI) data is embedded within a simplex, with the vertices of the simplex representing the endmembers to be estimated. This concept of simplex is illustrated in Fig. 2.8. As such, the endmember estimation problem can be framed as identifying this simplex, which can be further divided into two categories: pure-pixel and minimum-volume approaches. The former assumes the presence of at least one pure pixel for each endmember in the HSI data, as illustrated in the left of Fig. 2.9. A plethora of pure-pixel based algorithms have been proposed in literature [8], which can be further classified into two types: projection-based approaches and maximum-volume based approaches. Projection-based approaches leverage the idea that endmembers should always be the extreme points when the HSI data is projected onto any subspace or direction. Representative projection-based algorithms include Pixel Purity Index (PPI) [47] and Vertex Component Analysis (VCA) [11]. Maximum-volume based approaches rely on the insight that the simplex with r HSI vectors as

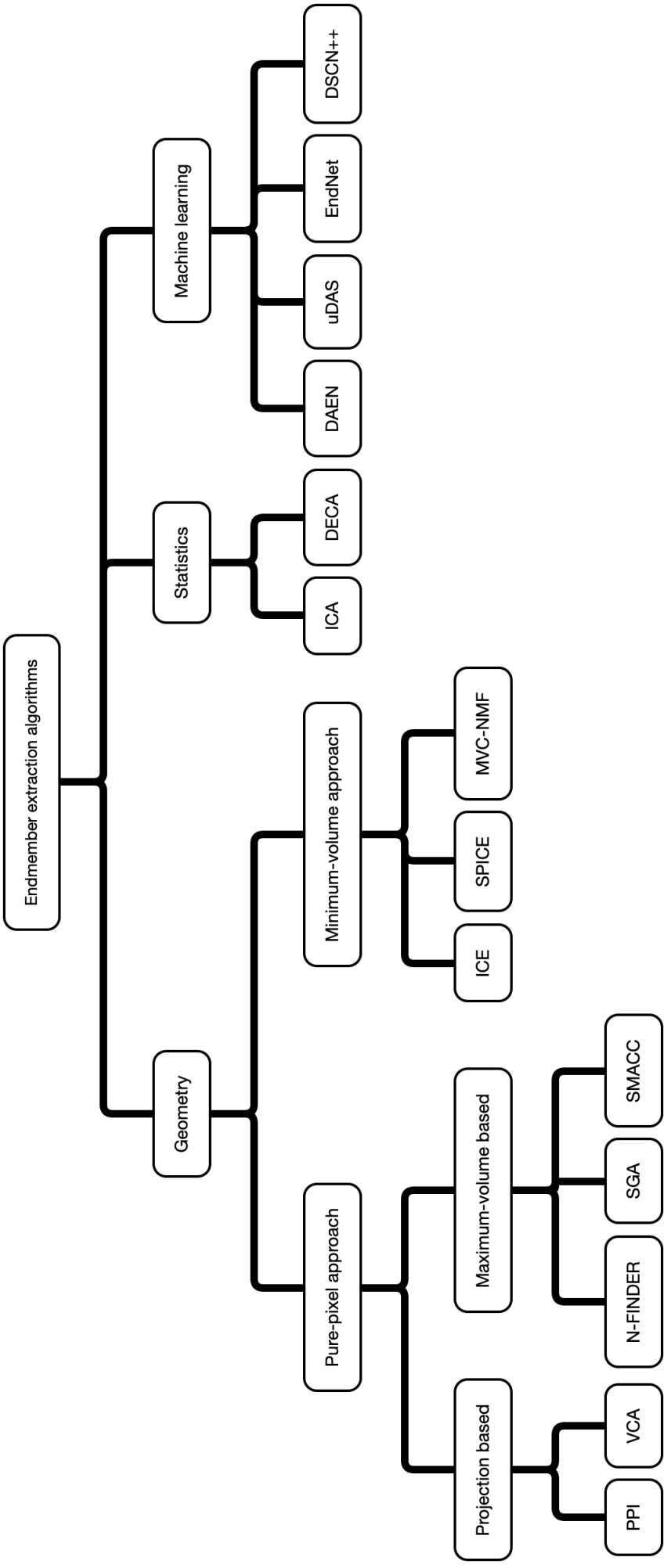


Figure 2.7: Categories of Endmember estimation algorithms

vertices should reach the maximum volume when those r HSI vectors are endmembers. Representative maximum-volume based algorithms include N-FINDER [48], simplex growing algorithm (SGA) [49], and sequential maximum angle convex cone (SMACC) [50].

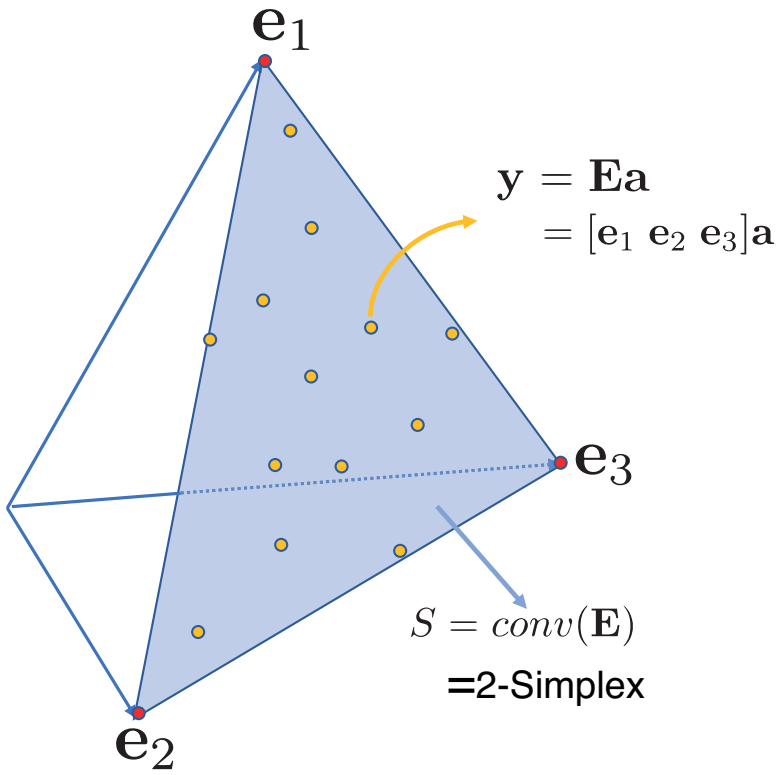


Figure 2.8: Simplex illustration. A 2-simplex S shown in shallow blue color is the convex hull of $\mathbf{E}$. The red circles represents the vertices of simplex and corresponding endmembers. Orange circles denote the HSI reflectance.

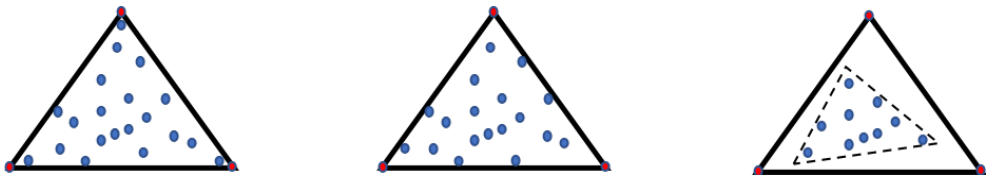


Figure 2.9: Illustration of (left) pure pixel approach; (middle) minimum-volume approach; (right)statistics approach. Blue points are HSI data and red point are endmembers.

The algorithms previously discussed are not suitable when there are no pure pixels present in the HSI dataset. However, it is still possible to extract endmembers by finding the minimum simplex that fits the HSI data, as long as there are at least $r - 1$ HSI spectral vectors on each facet of the simplex. This concept is illustrated in the mid-

dle of Fig. 2.9. The idea of searching for a minimum volume simplex, first introduced by [53], has inspired numerous geometrically-based extraction algorithms. The minimum-volume based algorithms can be formulated using the following formula:

$$\min_{\mathbf{E},\mathbf{A}} \frac{1}{2} \|\mathbf{Y} - \mathbf{E}\mathbf{A}\|_F^2 + \lambda V(\mathbf{E}), s.t., \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T \mathbf{1}_r = \mathbf{1}_n \tag{2.3.1}$$

Here, $V(\mathbf{E})$ represents the volume of the simplex formed by the endmember matrix. One possible definition of $V(\mathbf{E})$ is based on the assumption that the data has been projected onto a subspace of dimension r, and that the columns of $\mathbf{E}$ within this subspace are affinely independent [10]. This definition is given by

$$V(\mathbf{E}) = \frac{|det(\mathbf{E})|}{r!} \tag{2.3.2}$$

Alternatively, the volume can be defined using a different approach that involves shifting the data to the origin and then working in the subspace of dimension $r-1$. In this case, the volume can be calculated as follows:

$$V(\mathbf{E}) = \frac{1}{(r-1)!} \left| \det \begin{bmatrix} 1 & \cdots & 1 \\ \mathbf{e}_1 & \cdots & \mathbf{e}_r \end{bmatrix} \right| \tag{2.3.3}$$

Different definitions of $V(\mathbf{E})$ and optimization schemes can lead to various types of algorithms, such as the iterative constrained endmembers (ICE) algorithm [54] and the minimum volume transform-nonnegative matrix factorization (MVC-NMF) [15]. For more variants of these ideas, please refer to [10].

The right of Fig. 2.9 illustrates another type of HSI data, where the data is highly mixed and there are neither pure pixels nor spectral vectors near the facet present. In such cases, pure-pixel and minimum-volume based algorithms may not be effective, as the simplex found by these methods is smaller than the true one. To address this issue,

statistical-based algorithms have been proposed. These algorithms infer statistical information from the HSI data, providing valuable information about the endmembers. Bayesian methods are particularly popular in this field. One example of such an algorithm is the dependent component analysis method (DECA) [55], which assumes that the abundance vectors are drawn from a Dirichlet distribution and employs a generalized expectation maximization (GEM) algorithm to infer the endmember signatures. For a more in-depth understanding of statistical-based endmember estimation methods, please refer to [8].

2.3.2 Abundance estimation methods

After the estimation of endmember signatures, the HSI unmixing problem simplifies to an abundance estimation problem. This problem can be broadly classified into two categories: (1) linear approaches, which are based on the linear unmixing model, and (2) sparse approaches, which assume that the endmembers are known a priori through the use of a rich spectral library. We now provide a brief overview of these methods.

The linear approaches aim to estimate the abundance vector $\mathbf{a}$ for a mixed pixel in the linear model 2.2.1, given the observation $\mathbf{y}$ and the endmember signature matrix $\mathbf{E}$. This can be achieved by solving the following fully constrained least square (FCLS) problem:

$$\min_{\mathbf{a}} \frac{1}{2}\|\mathbf{y} - \mathbf{E}\mathbf{a}\|_2^2 \quad s.t., \mathbf{a} \geq \mathbf{0},\ \mathbf{a}^T\mathbf{1}_r = 1 \tag{2.3.4}$$

While the non-negativity constraint (ANC) is relatively easy to deal with, the sum-to-one constraint (ASC) is more challenging since it results in a set of inequalities that can only be solved by numerical methods. As a result, no closed-form solution can be derived for this problem. [2] proposed to solve the FCLS problem by resorting to an

iterative solver, which is commonly known as the FCLS solver.

Sparse-based approaches, which are inspired by popular sparse regression algorithms [56, 57], assume that the endmember matrix is known in the form of a rich spectral library. In this case, the abundance estimation problem becomes a constrained sparse regression (CSR) problem, as follows:

$$\min_{\mathbf{a}} \frac{1}{2}\|\mathbf{y}-\mathbf{E}\mathbf{a}\|_2^2+\lambda\|\mathbf{a}\|_1 \quad s.t., \mathbf{a} \geq \mathbf{0} \tag{2.3.5}$$

This problem has been solved by the renowned SunSAL algorithm [12] using the alternating direction method of multipliers (ADMM) solver [58] as follows:

$$\begin{cases} \mathbf{a}^{j+1} = (\mathbf{E}^T\mathbf{E}+\mu\mathbf{I})^{-1}(\mathbf{E}^T\mathbf{y}+\mu\left(\mathbf{u}^j+\mathbf{d}^j\right)) \\ \mathbf{u}^{j+1} = \max\left\{\mathbf{0}, \text{soft}\left(\mathbf{a}^{j+1}-\mathbf{d}^j, \lambda/\mu\right)\right\} \\ \mathbf{d}^{j+1} = \mathbf{d}^j-(\mathbf{a}^{j+1}-\mathbf{u}^{j+1}) \end{cases} \tag{2.3.6}$$

where μ is the ADMM free parameter. The operator $Soft$ is the soft-threshold operator given by, $soft(x,\theta) = sign(x)(|x|-\theta)_+$. Since its introduction, a plethora of sparsity-induced unmixing algorithms has been proposed, including CLSUnSAL [59], IRWSU [60], and TV-RSNMF [61]. These methods continue to be an active area of research in the field of HSI unmixing.

2.3.3 Blind Unmixing methods

There are also methods that perform endmember estimation and abundance estimation simultaneously, known as blind unmixing. This type of problem has a natural connection to nonnegative blind source separation (nBSS) methods [62], which are unsupervised learning techniques that extract nonnegative sources from mixed signals. One popular nBSS technique is nonnegative independent component analysis

(nICA) [63], which has been applied to the HSI unmixing problem in literature [64]. However, the success of nICA relies on the assumption that the sources (endmembers in HSI) are mutually independent, which is not always the case in HSI unmixing due to the sum-to-one constraint on the abundance vectors.

One of the most widely-used and successful nBSS methods for HSI unmixing is Nonnegative Matrix Factorization (NMF) [65]. This method aims to decompose a nonnegative observations matrix into two nonnegative matrices, which are interpreted as the endmember signature matrix and abundance matrix, respectively, in the context of HSI. One common approach [15] to solving the NMF problem is by formulating an optimization problem that minimizes the Euclidean distance between the observations matrix $\mathbf{Y}$ and the product of endmember and abundance matrices $\mathbf{EA}$, as shown below:

$$\min_{\mathbf{E},\mathbf{A}} \frac{1}{2}\|\mathbf{Y}-\mathbf{EA}\|_F^2, \quad s.t., \mathbf{E} \geq \mathbf{0}, \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \tag{2.3.7}$$

This type of method is demonstrated in Fig. 2.10. However, the objective function of NMF is non-convex, making the algorithm susceptible to finding local optima. To overcome this limitation, various regularizations have been proposed to enhance the performance of NMF based HSI unmixing methods.

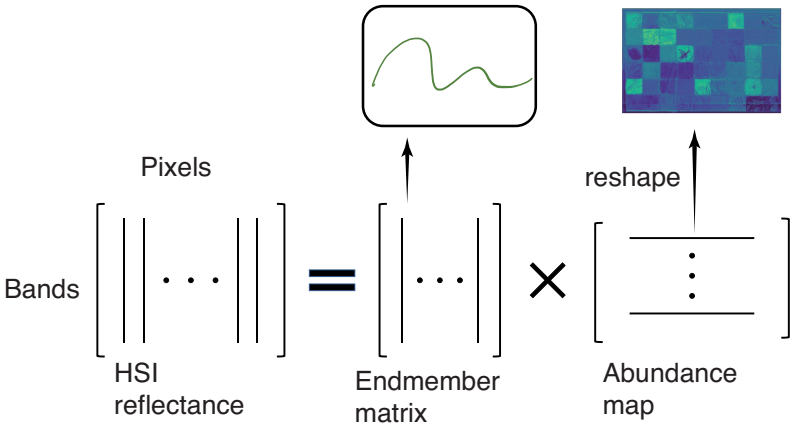


Figure 2.10: Illustration of NMF based unmixing framework.

A significant number of NMF based unmixing methods have sought to incorporate both spatial and spectral information through the implementation of additional constraints on endmembers and abundances. As previously discussed in Sec. 2.3.1, endmembers should take the form of a simplex that has the smallest volume among all simplices that encompass the HSI data. This concept has led to the development of the widely-used Minimum Volume Constrained NMF (MVC-NMF) approaches [15], which solves a constrained optimization problem, as shown below:

$$\begin{aligned} &\min_{\mathbf{E},\mathbf{A}} \frac{1}{2}\|\mathbf{Y}-\mathbf{E}\mathbf{A}\|_F^2 + \lambda J(\mathbf{E}) \\ &s.t., \mathbf{E} \geq \mathbf{0}, \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \end{aligned} \tag{2.3.8}$$

The penalty term $J(\mathbf{E})$ is used to constrain the volume of the simplex determined by the estimated endmembers. In MVC-NMF, the endmembers' dimensionality is first reduced by adopting the principle component analysis (PCA), which keeps the $r-1$ most significant principle components (PCs) out of p. The term $J(\mathbf{E})$ is then formulated as follows:

$$J(\mathbf{E}) = \frac{1}{2(r-1)!}\det^2\left(\begin{bmatrix} \mathbf{1}_r^T \\ \mathbf{E}_P \end{bmatrix}\right) \tag{2.3.9}$$

where $\mathbf{E}_P$ is the endmember matrix after PCA. To solve the optimization problem in Eq. (2.3.8), the alternating nonnegative least squares method is used [15]. Subsequently, various MVC-NMF algorithms have been extensively studied in literature [66–69] by incorporating additional constraints and regularizers, such as Total Variation (TV).

Another way to enhance the performance of NMF-based HSI unmixing methods is to incorporate more flexibility that takes into account more intricate structures and details. For instance, there can be a substantial disparity in the number of pixels associated with different endmembers. Weighted NMF methods [70] provide a way to account for this information by introducing a weight matrix generated from the

results of clustering. Other forms of generalizations of NMF-based methods, such as nonnegative tensor factorization (NTF) which factors a 3D structured tensor, or kernelized NMF that addresses nonlinearities, are also available. For a comprehensive survey of NMF-based HSI unmixing methods, it is recommended to refer to [71].

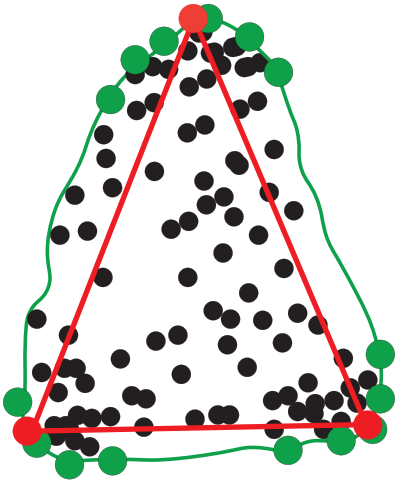


Figure 2.11: Visualisation of Archetypal Analysis, where observations are represented by black dots, extreme points by green dots, and the ground truth endmembers by red dots.

Recently, another widely used nBSS method, known as Archetypal Analysis (AA) [72], has gained popularity in the field of HSI analysis. This method is based on the concept that data is generated by a linear combination of a small number of archetypes, which are the extreme points of the data. In HSI unmixing, AA can be utilized to identify the endmember signatures and their corresponding abundances by assuming that the endmembers are the archetypes of the HSI data. In particular, AA formulates the problem by constraining the endmembers to be convex combinations of the pixels present in $\mathbf{Y}$, which can be expressed as $\mathbf{E} = \mathbf{YB}$, where $\mathbf{B} \in \mathbb{R}^{n \times r}$. The formulation of AA in HSI unmixing can be expressed as an optimization problem as follows:

$$\begin{aligned} &\min_{\mathbf{A},\mathbf{B}} \frac{1}{2}\|\mathbf{Y} - \mathbf{YBA}\|_F^2, \\ &s.t., \quad \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n, \mathbf{B} \geq \mathbf{0}, \mathbf{B}^T\mathbf{1}_n = \mathbf{1}_r \end{aligned} \tag{2.3.10}$$

This concept is illustrated in Fig. 2.11. The characteristic of AA, which represents each endmember as a convex combination of a limited number of pixels present in the HSI, offers a number of advantages. Firstly, it provides improved interpretability when compared to traditional NMF. Additionally, it has been demonstrated that AA can deliver estimations that are more robust to noise and spectral variability when compared to pure pixel methods [52].

One of the pioneering works that employed AA for solving the HSI unmixing problem is likely the study by [73]. They proposed a kernelized version of AA, which increases the model's flexibility but introduces the bandwidth of the kernel function as an additional parameter. In order to further mitigate the effects of noise and outliers, [74] proposed a robust kernel archetypal analysis (RKADA) method for blind hyperspectral unmixing. This method imposes a binary sparse constraint on the pixels' contributions in the standard AA formulation. Another approach, as proposed by [75], is to increase the sparsity of the abundances through an ℓ_1 sparsity-constrained AA algorithm. However, this formulation may result in abundances that do not add up to one, which negatively affects the physical interpretability of the unmixing results. A recent study by [52] introduced the Entropic Descent Archetypal Analysis (EDAA) for blind hyperspectral unmixing. EDAA solves Eq. (2.3.10) by performing the alternating updates as follows:

$$\begin{cases} \mathbf{A}^{j+1} = \operatorname{softmax}\left(\log\left(\mathbf{A}^{j}\right) + \eta_j \mathbf{B}^\top \mathbf{Y}^\top \left(\mathbf{Y} - \mathbf{Y}\mathbf{B}\mathbf{A}^{j}\right)\right) \\ \mathbf{B}^{j+1} = \operatorname{softmax}\left(\log\left(\mathbf{B}^{j}\right) + \eta_j \mathbf{Y}^\top \left(\mathbf{Y} - \mathbf{Y}\mathbf{B}\mathbf{B}^{j}\mathbf{A}\right)\mathbf{A}^\top\right) \end{cases} \tag{2.3.11}$$

Here, $\log\left(\mathbf{A}^{j}\right)$ applies the logarithm function to each element of $\mathbf{A}^{j}$, and the softmax function is applied column-wise on the matrix $\log\left(\mathbf{A}^{j}\right) + \eta_j \mathbf{B}^\top \mathbf{Y}^\top \left(\mathbf{Y} - \mathbf{Y}\mathbf{B}\mathbf{A}^{j}\right)$. This method employs an entropic gradient descent strategy and an ensembling mechanism to improve the unmixing performance. The study demonstrates that EDAA performs better

than traditional archetypal analysis algorithms and is robust to hyperparameter choices while maintaining a reasonable level of computational complexity.

In summary, model-based HSI unmixing methods have proven to be successful in identifying endmember signatures and abundances. However, these methods tend to be dependent on a range of assumptions and iterative optimization processes. Currently, there is no universal framework that can accommodate both linear and nonlinear unmixing scenarios.

2.4 Learning-based unmixing approaches

Recently, with the advancements in machine learning techniques, particularly deep neural networks (DNNs), a number of learning-based approaches have been proposed for HSI unmixing tasks [19–23, 76]. These methods can generally be classified into two categories: supervised and unsupervised. We now give a brief review of these methods.

2.4.1 Supervised learning approaches

Supervised methods [24, 25, 76] involve the use of a labeled dataset comprising N HSI reflectances and corresponding abundances, denoted as $\{\mathbf{y}_i, \mathbf{a}_i\}_{i=1}^{N}$, for training a neural network $f_{\boldsymbol{\theta}}$. The model parameters, denoted by $\boldsymbol{\theta}$, are learned by minimizing the objective function given by:

$$L = \frac{1}{2N} \sum_{i=1}^{N} \| f_{\boldsymbol{\theta}}(\mathbf{y}_i) - \mathbf{a}_i \|_2^2 \tag{2.4.1}$$

Once the model is trained, the model can be utilized for predicting the abundances of new, previously unseen HSI reflectances $\mathbf{y}$, which is achieved by calculating $\mathbf{a} = f_{\boldsymbol{\theta}}(\mathbf{y})$. Fig. 2.12 provides an illustration of this process using supervised unmixing networks. However, such methods require the availability of true abundance data, which may

not always be readily accessible.

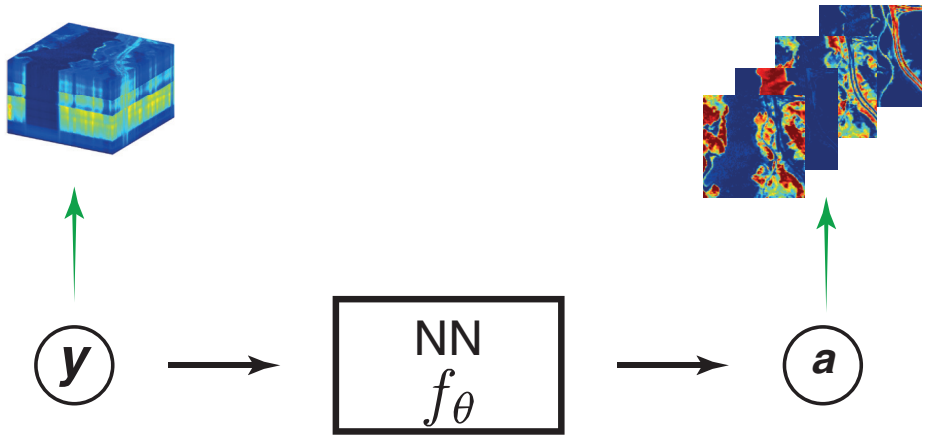


Figure 2.12: Supervised unmixing network.

2.4.2 Unsupervised learning approaches

In contrast to supervised methods, unsupervised methods for HSI unmixing do not require labeled data, and instead, the learning algorithm is designed to estimate both endmembers and abundances from the HSI reflectances alone. These blind unmixing methods [27–33] often employ an autoencoder network structure with a linear decoder, which takes the HSI spectra $\mathbf{y}_i$ as input and is trained to reconstruct the input spectra. Specifically, assuming that the encoder $f_{\boldsymbol{\theta}}$ has learnable parameters $\boldsymbol{\theta}$, and the decoder $g_{\boldsymbol{\phi}}$ also has learnable parameters $\boldsymbol{\phi}$, the autoencoder operates as follows:

$$\begin{cases} \text{Encoder} : \hat{\mathbf{a}}_i = f_{\boldsymbol{\theta}}(\mathbf{y}_i) \\ \text{Decoder} : \hat{\mathbf{y}}_i = g_{\boldsymbol{\phi}}(\hat{\mathbf{a}}_i) \end{cases} \tag{2.4.2}$$

Here, $\hat{\mathbf{a}}_i$ represents the bottleneck of the autoencoder, and $\hat{\mathbf{y}}_i$ represents the reconstructed HSI spectra. The decoder is typically a linear operation given by $g_{\boldsymbol{\phi}}(\hat{\mathbf{a}}_i) = \boldsymbol{\phi}\hat{\mathbf{a}}_i$. Usually, the autoencoder is trained on a dataset of N HSI observations $\{\mathbf{y}_i\}_{i=1}^N$ by minimizing the following

objective:

$$L_{\{\boldsymbol{\theta},\boldsymbol{\phi}\}} = \frac{1}{2N}\sum_{i=1}^{N}\|\hat{\mathbf{y}}_i - \mathbf{y}_i\|_2^2 \tag{2.4.3}$$

After training, the bottleneck of the autoencoder $\hat{\mathbf{a}}_i$ provides an estimation of the abundances, while the weights of the linear decoder ϕ give the endmember estimation. This is illustrated in Fig. 2.13. An example of an unsupervised method is the Deep Autoencoder Network (DAEN) [32]. This method utilizes the traditional Vertex Component Analysis (VCA) algorithm to generate a set of pure endmember candidates, which are then used to train a stack of autoencoders. This generates a good initialization for the parameters in the variational autoencoder, allowing for improved unmixing performance. In the work of uDAS [29], a sparsity constraint is applied on the encoder and a mDA denoising constraint is imposed on the neural network to enhance the unmixing performance. Additionally, EndNet [31] incorporates an additional layer using a projection metric in lieu of inner product in order to improve unmixing results. The loss function used in this method is composed of KullbackLeibler (KL) divergence, SAD similarity, and a sparsity penalty on the estimates. To further enhance the unmixing results, some methods [20, 21] have introduced the use of adversarial autoencoders, where the unmixing autoencoder is trained in an adversarial manner using an additional discriminator.

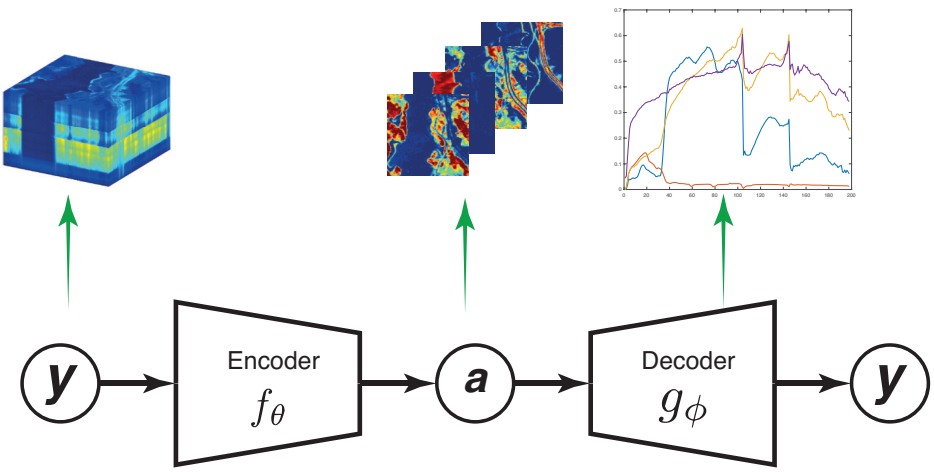


Figure 2.13: Unsupervised unmixing network.

2.4.3 Model-aware learning approaches

Despite the advancements in machine learning techniques for HSI unmixing, there are still several challenges to be addressed. One major issue is the lack of a systematic approach to designing network architectures, particularly the encoder network. Additionally, neural networks often lack interpretability, making it difficult to incorporate prior knowledge about the task into the design of the network.

Recently, algorithm unrolling or unfolding techniques [34, 35] have emerged as a potential solution to design interpretable network structures for unmixing tasks. A seminal work in this area is by [35], which aims to solve the sparse coding problem. Specifically, the problem is to minimize the following objective function:

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{W}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1 \tag{2.4.4}$$

where $\mathbf{y}$ is the observation vector, $\mathbf{W}$ is an over-complete dictionary, and $\mathbf{x}$ is a sparse code with as many coefficients as possible encouraged to be zero or small in magnitude. The Iterative Shrinkage and Thresholding Algorithms (ISTA) [77] is a popular method for solving this problem, which involves performing the following iterations:

$$\mathbf{x}^{j+1} = \mathrm{Soft}\left(\left(\mathbf{I} - \frac{1}{\mu}\mathbf{W}^T\mathbf{W}\right)\mathbf{x}^j + \frac{1}{\mu}\mathbf{W}^T\mathbf{y}, \lambda\right) \tag{2.4.5}$$

where Soft is the soft-thresholding operator, $\boldsymbol{I}$ is the identity matrix, and μ is a free parameter.

By defining $\boldsymbol{W}_t = \mathbf{I} - \frac{1}{\mu}\mathbf{W}^T\mathbf{W}$ and $\boldsymbol{W}_e = \frac{1}{\mu}\mathbf{W}^T$, the above iterations can be expressed as a single layer neural network:

$$\mathbf{x}^{j+1} = \mathrm{Soft}\left(\boldsymbol{W}_t\mathbf{x}^j + \boldsymbol{W}_e\mathbf{y}, \lambda\right) \tag{2.4.6}$$

Here $\boldsymbol{W}_t$, $\boldsymbol{W}_e$, and λ are learnable parameters and the layer involves

matrix-vector multiplication, summation, and soft-thresholding (nonlinear) operations, similar to a neural network. By cascading L such layers together, an L-layer deep network can be interpreted as executing L iterations of ISTA. This is illustrated in Fig. 2.14. The resulting network structure is interpretable and can leverage insights of the underlying model into network construction.

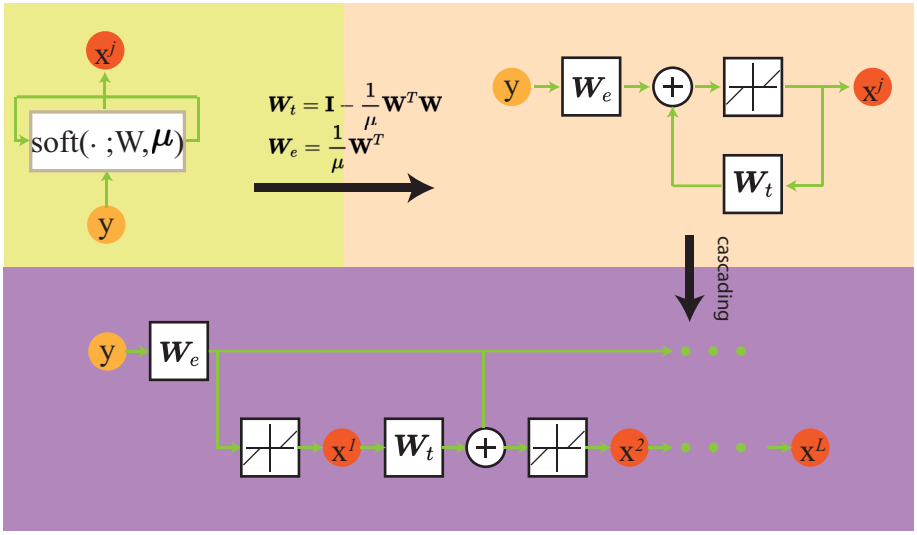


Figure 2.14: Illustration of algorithm unrolling/unfolding techniques.

Building on this approach, [18, 26] have proposed using the ISTA Algorithm to design unmixing networks, known as MNN-AE and MNN-BU. These networks are constructed by converting each step in the ISTA solver, which is used to solve a constrained sparsity linear regression unmixing problem, into a network operation. The network is then formed by concatenating multiple iterations (or layers in network language) of this operation. Similarly, [22] proposed an unmixing network by unrolling an alternating optimization algorithm of a sparsity constrained nonnegative matrix factorization model. It has been demonstrated in [78–80] that networks unfolded from iterative algorithms can deliver state-of-the-art performance, surpassing existing methods.

2.4.4 Challenges

Despite the recent advancements, it remains an active area of research to investigate whether the performance can be further improved by incorporating other advanced iterative solvers in algorithm unrolling techniques. Additionally, the methods mentioned above are limited to linear blind unmixing problems. To address nonlinear blind unmixing problems, [19] have proposed a deep autoencoder which involves an additive nonlinear mixture part. Likewise, a novel nonlinear autoencoder structure proposed by [23] combines a cross-product layer to account for nonlinear mixing mechanisms. However, these methods rely on an autoencoder architecture that assumes endmembers are embedded in the decoder's weights, which limits their flexibility to incorporate additional constraints or regularization on the endmembers and may not handle complex unmixing models. Furthermore, most of the current learning algorithms rely on autoencoder architectures for addressing either linear or nonlinear unmixing problems and fall short in effectively tackling both. Moreover, despite the success of deep learning-based methods in unmixing problems, these methods may not always produce physically meaningful results without proper guidance [4]. More specifically, these methods have the potential to generate arbitrary unmixing results wherein the retrieved reflectances do not align with any authentic endmembers. To address this issue, researchers have proposed methods that utilize existing algorithms to extract endmembers, which are then used to guide the training of an abundance estimation network. One such method is UnDIP [36], which utilizes the simplex volume maximization algorithm (SiVM) [1] to extract endmembers and uses them to guide the training of a deep image prior-based network, as shown in Fig. 2.15. Another approach is EGU-Net [4], which develops a two-stream deep network that learns from endmembers extracted from HSI data using existing methods,

and models nonlinear unmixing as general errors and spectral variability. Despite the advancements in the field, a major challenge in training unmixing networks with guidance is that the performance is often constrained by the quality of the guidance.

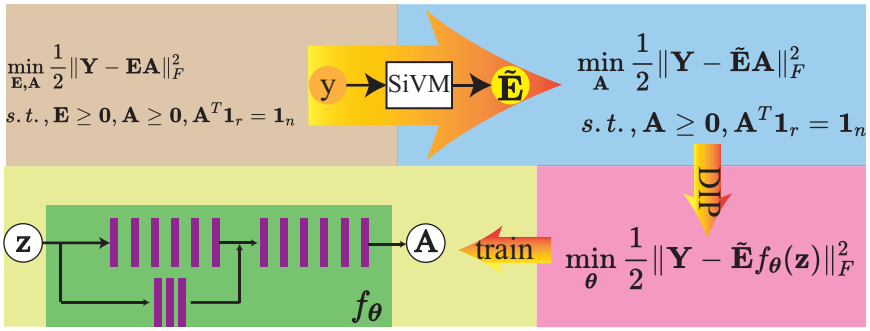


Figure 2.15: Illustration of UnDIP techniques.

2.5 Summary

In this Chapter, we provide a comprehensive overview of the hyperspectral image unmixing problem. We introduced the concept of unmixing and different types of mixing models, including linear and nonlinear models. We then reviewed traditional model-based unmixing approaches, including methods that focus on endmember estimation and abundance estimation. Additionally, we also discussed fully blind unmixing methods. Lastly, we introduced recent developments in learning-based unmixing approaches and highlighted the challenges that current methods face. Overall, this Chapter serves as a foundation for the subsequent chapters where we will delve deeper into the proposed unmixing methods.

Chapter 3

ADMM based unmixing network

In this chapter, we present a neural network approach for abundance estimation using a constrained sparse regression (CSR) formulation of the unmixing problem. By unrolling the Alternating Direction Method of Multipliers (ADMM) solver to the CSR objective, we construct a neural network that can be trained using supervised learning techniques with a new proposed composite loss function. Additionally, we propose another neural network structure for blind unmixing that can be trained using unsupervised learning techniques. In comparison with other unmixing architectures, our proposed neural networks have a structure that is both lightweight and rich, characterized by a reduced number of learnable parameters and an increased number of skip connections.

3.1 Problem formulation

In this chapter, we consider the HSI LMM model using the pixel-wise formulation, as given in Eq. (2.2.1). We begin by formulating the HSI unmixing problem using the CSR formulation. According to Eq. (2.3.5), when the endmember matrix is available in the form of a spectral library [81], the recovered abundance vector for a pixel's reflectance

tends to be sparse. To recover the abundances from the reflectance, we adopt the CSR-based optimization problem [18], which is given by:

$$\min_{\mathbf{a}} \frac{1}{2}||\mathbf{y} - \mathbf{E}\mathbf{a}||_2^2 + \lambda||\mathbf{a}||_1, s.t., \mathbf{a} \geq \mathbf{0} \tag{3.1.1}$$

where $\lambda \geq 0$ is a regularization parameter that controls the sparsity of solutions. It is worth mentioning that the ASC constraint is not imposed in Eq. (3.1.1), because otherwise, it would reduce to the FCLS problem as given in Eq. (2.3.4). Additionally, imposing ASC constraint along with ℓ_1 regularization would constrain the sparsity to be constant, which is unnecessary when formulating a CSR problem [12].

The solution to the CSR optimization problem in Eq. (3.1.1) can be achieved through various solvers. One commonly used method is the ADMM algorithm, which results in the well-known SunSAL algorithm [12]. By introducing an auxiliary variable $\mathbf{z}$ and a dual variable $\mathbf{d}$, with the constraint $\mathbf{a} = \mathbf{z}$, the ADMM algorithm provides an iterative scheme to compute the solution of the CSR problem in Eq. (3.1.1). The specific steps are as follows:

$$\mathbf{a}^{j+1} = (\mathbf{E}^T\mathbf{E} + \mu\mathbf{I})^{-1}(\mathbf{E}^T\mathbf{y} + \mu(\mathbf{z}^j + \mathbf{d}^j)) \tag{3.1.2}$$

$$\mathbf{z}^{j+1} = \max\left(soft\left(\mathbf{a}^{j+1} - \mathbf{d}^j, \frac{\lambda}{\mu}\right), 0\right) \tag{3.1.3}$$

$$\mathbf{d}^{j+1} = \mathbf{d}^j - \left(\mathbf{a}^{j+1} - \mathbf{z}^{j+1}\right) \tag{3.1.4}$$

where $\mathbf{a}^j$ is the value of variable $\mathbf{a}$ at j^{th} iteration (same for $\mathbf{z}^j, \mathbf{d}^j$) and $\mu \geq 0$ is a parameter that is usually chosen to be an upper bound to the largest eigenvalue of $\mathbf{E}^T\mathbf{E}$. The operator $Soft$ in Eq. (3.1.3) is the soft-threshold operator given by, $soft(x, \theta) = sign(x)(|x| - \theta)_+$.

3.2 Abundance Estimation Network

We present the construction of our ADMM-based network structure for abundance estimation, followed by the introduction of the initialisation and training techniques for this proposed network.

3.2.1 Unfolding ADMM into a Neural Network Layer

The proposed network's layer operations are derived from unrolling the three separate iterative operations of the ADMM solver as outlined in Eq. (3.1.2), Eq. (3.1.3), and Eq. (3.1.4), leading to three separate components.

A-Update Component

The *A-update component* at the $(j+1)^{th}$ layer of the neural network is derived by unfolding Eq. (3.1.2). Specifically, the $(j+1)^{th}$ iteration of $\mathbf{a}^{j+1}$ can be expressed as a function of the j^{th} estimates $\mathbf{z}^j$, $\mathbf{d}^j$, and $\mathbf{y}$ as follows:

$$\begin{aligned} \mathbf{a}^{j+1} &= f_A(\mathbf{z}^j, \mathbf{d}^j, \mathbf{y}; \mathbf{W}, \mathbf{B}) \\ &= \mathbf{W}\mathbf{y} + \mathbf{B}(\mathbf{z}^j + \mathbf{d}^j) \end{aligned} \tag{3.2.1}$$

where

$$\begin{aligned} \mathbf{W} &= (\mathbf{E}^T\mathbf{E} + \mu\mathbf{I})^{-1}\mathbf{E}^T \\ \mathbf{B} &= (\mathbf{E}^T\mathbf{E} + \mu\mathbf{I})^{-1}\mu \end{aligned} \tag{3.2.2}$$

To enhance flexibility, we will employ learnable parameters $\mathbf{W}^{j+1} \in \mathbb{R}^{r\times p}$ and $\mathbf{B}^{j+1} \in \mathbb{R}^{r\times r}$ in this $(j+1)^{th}$ layer, replacing the fixed parameters $\mathbf{W}$ and $\mathbf{B}$. This approach allows for the parameters $\mathbf{W}^{j+1}$ and $\mathbf{B}^{j+1}$ to deviate from the original $\mathbf{W}$ and $\mathbf{B}$ in Eq. (3.1.1) to better align with the data's characteristics.

It's worth mentioning that the *A-update component* can also be interpreted as a typical linear layer in a neural network [82]. The structure of the *A-update component* in a neural network layer is depicted

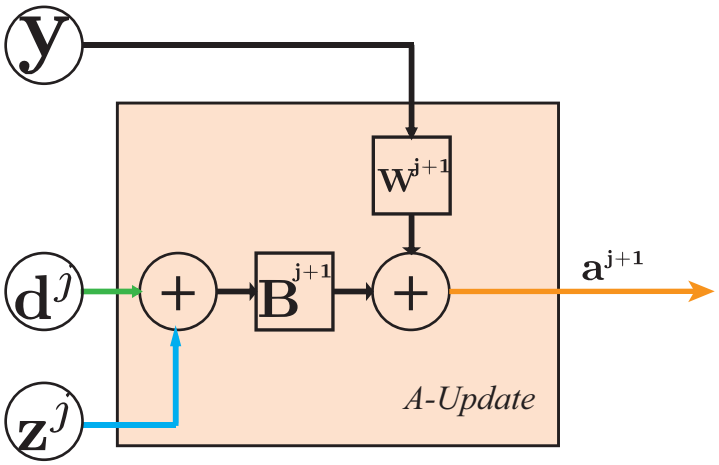


Figure 3.1: *A-update component* structure.

in Fig. 3.1.

Z-Update Component

The *Z-update component* of the $(j+1)^{th}$ neural network layer is obtained by unfolding Eq. (3.1.3). Specifically, the $(j+1)^{th}$ iteration $\mathbf{z}^{j+1}$ is calculated from the $(j+1)^{th}$ estimate $\mathbf{a}^{j+1}$ and the j^{th} estimate $\mathbf{d}^{j}$ using a soft-thresholding operation with parameter λ/μ followed by a max operation. We propose re-writing Eq. (3.1.3) as follows:

$$\begin{aligned} \mathbf{z}^{j+1} &= f_Z(\mathbf{a}^{j+1}, \mathbf{d}^j; \theta^{j+1}) \\ &= \max\left(soft\left(\mathbf{a}^{j+1} - \mathbf{d}^j, \theta^{j+1}\right), 0\right) \end{aligned} \tag{3.2.3}$$

Where $\theta^{j+1} \in \mathbb{R}$ is a learnable parameter that varies between layers. This parameter θ^{j+1} serves as a replacement for the parameter $\frac{\lambda}{\mu}$, offering the advantage of being able to be learned from data, allowing for better adaptation to the characteristics of a specific unmixing problem.

It is worth noting that the operation performed by the *Z-update component*, as illustrated in Fig. 3.2, can also be expressed as:

$$\mathbf{z}^{j+1} = ReLU(\mathbf{a}^{j+1} - \mathbf{d}^j - \theta^{j+1}\mathbf{I}) \tag{3.2.4}$$

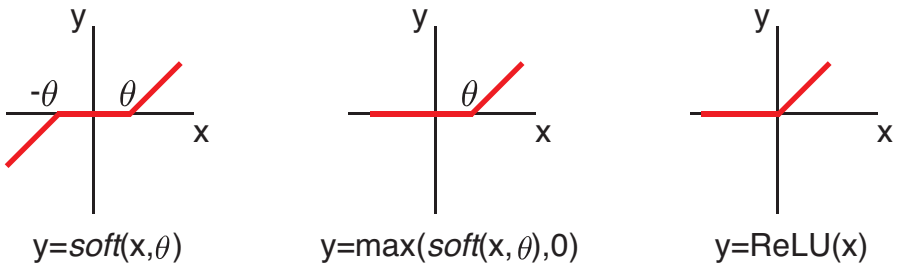


Figure 3.2: *soft* function *vs.* max*soft* function *vs.* ReLU function.

where $ReLU(\cdot)$ is a component-wise rectified linear unit operation [82]. This means that the *Z-update component* – which also ensures the satisfaction of the ANC constraint – acts as a ReLU operation in a standard neural network. The *Z-update component* of a neural network layer is illustrated in Fig. 3.3.

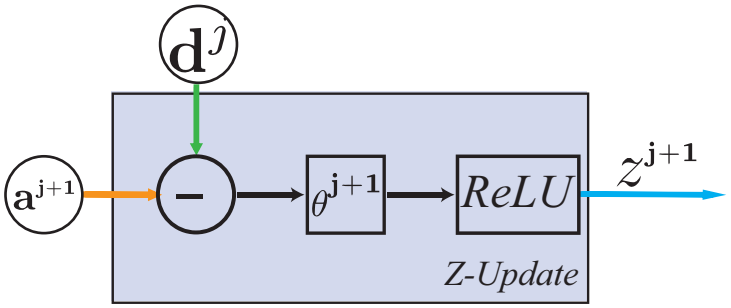


Figure 3.3: *Z-update component* structure.

D-Update Component

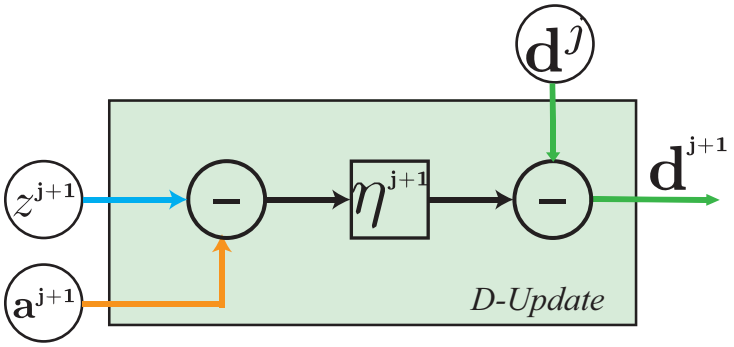


Figure 3.4: *D-update component* structure.

Finally, the *D-Update Component* of the $(j+1)^{th}$ neural network layer is derived from Eq. (3.1.4). Specifically, the $(j+1)^{th}$ iterate $\mathbf{d}^{j+1}$ is calculated as the difference between the j^{th} iterate $\mathbf{d}^{j}$ and the $(j+1)^{th}$

iterates $\mathbf{a}^{j+1}$ and $\mathbf{z}^{j+1}$. However, we suggest re-expressing Eq. (3.1.4) as:

$$\begin{aligned}\mathbf{d}^{j+1} &= f_D(\mathbf{a}^{j+1}, \mathbf{z}^{j+1}, \mathbf{d}^j; \eta^{j+1}) \\ &= \mathbf{d}^j - \eta^{j+1}(\mathbf{a}^{j+1} - \mathbf{z}^{j+1})\end{aligned} \tag{3.2.5}$$

where η^{j+1} is a learnable parameter, which functions as a step-size that can be adjusted for better adaptation to the unmixing problem. The *D-Update Component* of a neural network layer is illustrated in Fig. 3.4.

Overall Neural Network Layer

By combining the A-update, Z-update, and D-update components, the structure of a neural network layer is formed, as shown in Fig. 3.5.

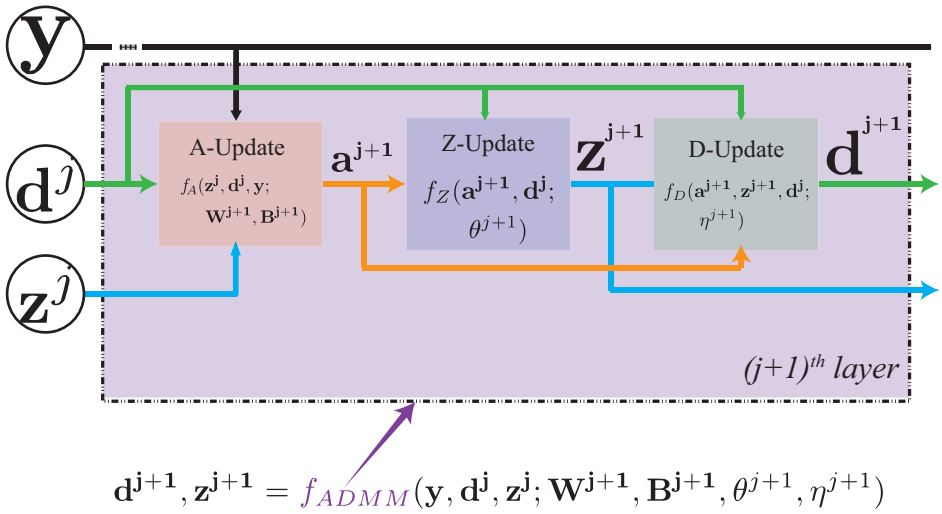


Figure 3.5: The neural network layer derived from the ADMM solver, replicating an iteration of the ADMM algorithm. Upon receiving inputs of $\mathbf{y}, \mathbf{d}^j, \mathbf{z}^j$, the layer executes the process defined in Eq. (3.2.6) and generates outputs of $\mathbf{d}^{j+1}, \mathbf{z}^{j+1}$. The layer encompasses learnable parameters, $\mathbf{\Theta}^{j+1} = \{\mathbf{W}^{j+1}, \mathbf{B}^{j+1}, \theta^{j+1}, \eta^{j+1}\}$.

Corresponding to one iteration of the ADMM algorithm, each network layer performs operations as follows:

$$\mathbf{d}^{j+1}, \mathbf{z}^{j+1} = f_{ADMM}(\mathbf{y}, \mathbf{d}^j, \mathbf{z}^j; \mathbf{W}^{j+1}, \mathbf{B}^{j+1}, \theta^{j+1}, \eta^{j+1}) \tag{3.2.6}$$

where, f_{ADMM} performs Eq. (3.2.1), Eq. (3.2.3) and Eq. (3.2.5) con-

secutively. However, the network layers have learnable parameters $\mathbf{\Theta}^{j+1} = \{\mathbf{W}^{j+1}, \mathbf{B}^{j+1}, \theta^{j+1}, \eta^{j+1}\}$, while each iteration of the ADMM algorithm has fixed parameters $\{\mathbf{A}, \mu, \lambda\}$. This learnable parameterization allows for a more compact network architecture, since fewer layers are needed to achieve the desired performance compared to the number of iterations in the ADMM algorithm.

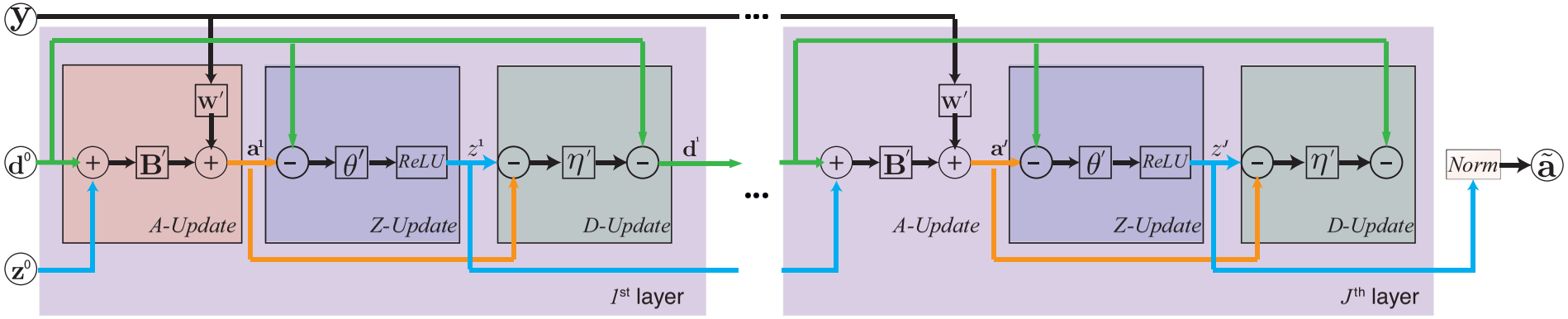
In Fig. 3.5, there are four shortcuts: one from $\mathbf{d}^j$ to both Z-update and D-update components, one from $\mathbf{a}^{j+1}$ to the D-update component, one from $\mathbf{z}^{j+1}$ to the A-update component of the next layer, and one from the input $\mathbf{y}$. This is in contrast to conventional neural networks, where the output of each component is only connected to the next component and the network input $\mathbf{y}$ is usually only connected to the first layer.

3.2.2 Abundance Estimation Network Structure

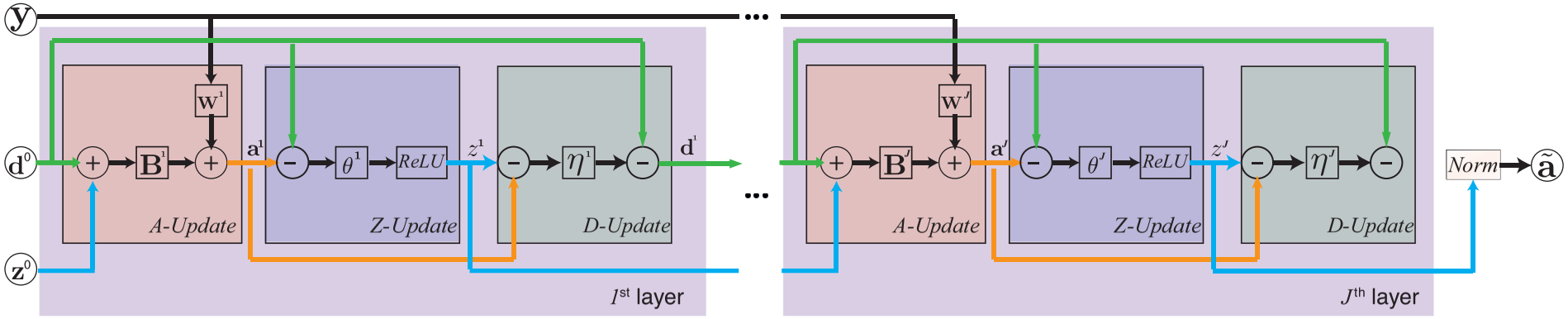
Previously, we proposed that each iteration of the ADMM algorithm can be viewed as a neural network layer and that the ADMM model parameters $\{\mathbf{A}, \mu, \lambda\}$ can be replaced with learnable ones in each layer. As a result, two different feed-forward neural networks can be constructed by concatenating J iteration blocks, imitating J iterations of the ADMM algorithm. While learned parameters have the potential to capture complex relationships and patterns in the data, this does not always lead to improved unmixing outcomes because it also introduces the risk of overfitting.

Our first network, referred to as UAENet-I, is formed by stacking J iteration blocks, each consisting of A-Update, Z-Update, and D-Update components. The learnable parameters are shared across all layers of the network, i.e., $\mathbf{\Theta}^j = \{\mathbf{W}', \mathbf{B}', \theta', \eta'\}$ for $j \in [1, J]$. A visualization of a J-layer UAENet-I can be found in Fig. 3.6a.

Our second network, designated as UAENet-II, is also constructed



(a) A J-layer UAENet-I.



(b) A J-layer UAENet-II.

Figure 3.6: The structures of UAENet. (a) a J-layer UAENet-I, of which the learnable parameters in each layer are same/tied/shared, $\mathbf{\Theta}^j = \{\mathbf{W}', \mathbf{B}', \theta', \eta'\}, \forall j \in [1, J]$. (b) a J-layer UAENet-II, of which the learnable parameters in each layer are different/untied/unshared, $\mathbf{\Theta}^j = \{\mathbf{W}^j, \mathbf{B}^j, \theta^j, \eta^j\}, \forall j \in [1, J]$.

by concatenating J iteration blocks. However, the learnable parameters are specific to each layer, rather than being shared across the network, i.e., $\mathbf{\Theta}^j = \{\mathbf{W}^j, \mathbf{B}^j, \theta^j, \eta^j\}, \forall j \in [1, J]$. UAENet-II exhibits increased capacity and flexibility, which may lead to improved unmixing results compared to UAENet-I. The architecture of a J-layer UAENet-II is illustrated in Fig. 3.6b.

In both of our networks (UAENet-I and UAENet-II), the input to the network is the HSI spectrum $\mathbf{y}$ and the accompanying pseudo inputs $\mathbf{z}^0$ and $\mathbf{d}^0$, both of which are set to 0 as per the standard approach in ADMM algorithms. These inputs are processed through J network layers with the set of learnable parameters $\mathbf{\Theta} = \{\mathbf{\Theta}^j\}_{j=1}^J$. The network output is derived from the *Z-Update* component in the last iteration block as this component ensures compliance with the ANC constraint. We then further normalize the output using l_1 normalization to satisfy the ASC constraint. Denote the unnormalized network output as $\mathbf{z}$, the component-wise calculation of the normalized output $\tilde{\mathbf{a}}$ is given by:

$$\tilde{a}_i = \frac{z_i}{\sum_j z_j} \tag{3.2.7}$$

Since $\mathbf{z}$ is the output of the ReLU operator, it is guaranteed to be non-negative. This normalization step is added as the ASC constraint is not directly incorporated into the optimization formulation.

3.2.3 Network Initialization and Training strategies

Initialization Approach

Given the interpretable parameterization, the neural networks are trained using warm initialization instead of random initialization to accelerate the training process. Specifically, the parameters of the network are initialized by utilizing the original parameters associated with the ADMM algorithms as follows:

- The parameters $\mathbf{W}^j$ and $\mathbf{B}^j$ for the *A-update component* of each layer are initialized according to Eq. (3.2.2). The endmember signature matrix $\mathbf{E}$ can be either selected from an appropriate spectral library or estimated using existing algorithms such as VCA [11]. Any relevant candidate signatures can then be concatenated with $\mathbf{E}$.

- The parameter θ^j for the *Z-update component* of each layer is initialised using λ/μ, where λ and μ are parameters associated with the ADMM algorithm.

- Finally, the parameter η^j associated with the *D-update component* of each layer has no direct equivalent in the ADMM algorithm. Nonetheless, it is set to one for initialisation purposes.

It is worth noting that with this initialization strategy, a J-layer UAENet is equivalent to J iterations of the ADMM algorithm.

Training Approach

The proposed neural networks are trained with a supervised learning approach using a training set, $D = \{\mathbf{y}_i, \mathbf{a}_i\}_{i=1}^N$, of N reflectance spectra, $\mathbf{y}_i$, and their corresponding abundances, $\mathbf{a}_i$. A new composite loss function, instead of the commonly used mean-squared error (MSE) loss function [18], is proposed to train the networks. This composite loss function is defined as:

$$L_{\mathbf{\Theta}} = \alpha_1 \cdot L_1 + \alpha_2 \cdot L_2 + \alpha_3 \cdot L_3 \tag{3.2.8}$$

where α_1, α_2, and α_3 are hyper-parameters that control the contribution of each component, L_1, L_2, and L_3, to the loss function.

The first component of the loss function incorporates the well-known

MSE as follows:

$$\begin{aligned} L_1 &= MSE(\{\mathbf{a}_i, \tilde{\mathbf{a}}_i\}_{i=1}^N) \\ &= \frac{1}{N} \sum_{\{\mathbf{y}_i, \mathbf{a}_i\} \in D} \|\mathbf{a}_i - \tilde{\mathbf{a}}_i(\mathbf{y}_i)\|_2^2 \end{aligned} \tag{3.2.9}$$

Here $\mathbf{y}_i$ is the i^{th} reflectance spectrum in the training set, $\mathbf{a}_i$ is the corresponding i^{th} abundance, and $\tilde{\mathbf{a}}_i(\mathbf{y}_i)$ represents the network's estimate of $\mathbf{a}_i$ given $\mathbf{y}_i$. This loss function is often utilised for training neural networks for unmixing purposes [26].

The second component of the loss function is derived from the abundance angle distance (AAD) [14, 15, 25, 29, 83], as follows:

$$\begin{aligned} L_2 &= AAD(\{\mathbf{a}_i, \tilde{\mathbf{a}}_i\}_{i=1}^N) \\ &= \frac{1}{N} \sum_{\{\mathbf{y}_i, \mathbf{a}_i\} \in D} \cos^{-1}\left(\frac{\mathbf{a}_i^T \tilde{\mathbf{a}}_i(\mathbf{y}_i)}{\|\mathbf{a}_i\|_2 \|\tilde{\mathbf{a}}_i(\mathbf{y}_i)\|_2}\right) \end{aligned} \tag{3.2.10}$$

The third component of the loss function originates from the abundance information divergence (AID) [15, 29] and is formulated as follows:

$$\begin{aligned} L_3 &= AID(\{\mathbf{a}_i, \tilde{\mathbf{a}}_i\}_{i=1}^N) \\ &= \frac{1}{N} \sum_{\{\mathbf{y}_i, \mathbf{a}_i\} \in D} KL\Big(\mathbf{a}_i | \tilde{\mathbf{a}}_i(\mathbf{y}_i)\Big) + KL\Big(\tilde{\mathbf{a}}_i(\mathbf{y}_i) | \mathbf{a}_i\Big) \end{aligned} \tag{3.2.11}$$

where $KL(\mathbf{x}|\tilde{\mathbf{x}})$ calculates the Kullback – Leibler divergence between two probabilities $\mathbf{x}$ and $\tilde{\mathbf{x}}$ as follows:

$$KL(\mathbf{x}|\tilde{\mathbf{x}}) = \sum_{m=1}^{r} \left(x_m \log\left(\frac{x_m}{\tilde{x}_m}\right)\right) \tag{3.2.12}$$

It's important to note that the second and third components of the loss function serve as additional metrics for measuring the discrepancy between the recovered abundance and the true abundance.

We introduce two additional components to the loss function for the

following reasons: (1) The MSE loss implicitly assumes [84] that the abundance estimates generated by the neural network follow a Gaussian distribution, which is not appropriate given that abundance vectors are subject to both ANC and ASC constraints. In contrast, the AAD metric, which measures the angle between vectors, does not make this assumption. (2) Given the ANC and ASC constraints, abundance vectors can be considered to have a probabilistic interpretation. The KL divergence is commonly used to measure the distance between two probability distributions, such as two abundance vectors. The AID metric is a symmetric version of the KL divergence that can also be used to quantify the difference between two abundance vectors. In prior research [27, 31], similar spectral information divergence (SID) and spectral angle distance (SAD) losses have been used to assess hyperspectral image reflectance reconstruction performance. However, we opt for a more comprehensive combination of loss functions to better capture abundance estimation performance. The results of our experiments demonstrate that this more robust combination leads to improved performance.

Parameter Update Rule

The parameters of the proposed networks, UAENet-I and UAENet-II, are learned by minimizing the loss function $L_{\mathbf{\Theta}}$ defined in Eq. (3.2.8) using the stochastic gradient descent algorithm ADAM [85]. The hyperparameters are optimized via cross-validation techniques, resulting in $\alpha_1 = 1.0$, $\alpha_2 = 10^{-7}$, and $\alpha_3 = 10^{-5}$. The stochastic gradient algorithm updates the learnable parameters $\mathbf{\Theta}$ as follows:

$$\mathbf{\Theta}_{new} = \mathbf{\Theta}_{old} - l\frac{\partial L_{\mathbf{\Theta}_{old}}}{\partial \mathbf{\Theta}_{old}} \tag{3.2.13}$$

where l is the learning rate. The gradient of each learnable parameter in $\mathbf{\Theta}$ is computed using the back-propagation algorithm [82]. The for-

ward computation of the J-layer network progresses from the input to the output, following the path $\mathbf{y} \rightarrow \mathbf{\Theta}^1 \rightarrow \cdots \rightarrow \mathbf{\Theta}^j \rightarrow \cdots \rightarrow \mathbf{\Theta}^J \rightarrow \tilde{\mathbf{a}}$. The back-propagation algorithm calculates the gradient for the parameters in each layer in reverse order as follows:

$$\frac{\partial L_{\mathbf{\Theta}}}{\partial \mathbf{\Theta}^j} = \frac{\partial L_{\mathbf{\Theta}}}{\partial \tilde{\mathbf{a}}} \frac{\partial \tilde{\mathbf{a}}}{\partial \mathbf{\Theta}^J} \cdots \frac{\partial \mathbf{\Theta}^{j+1}}{\partial \mathbf{\Theta}^j}, \; j = 1, \cdots, J-1 \tag{3.2.14}$$

For example, the gradient for parameter θ^J in the last layer can be calculated as follows:

$$\frac{\partial L_{\mathbf{\Theta}}}{\partial \theta^J} = \frac{\partial L_{\mathbf{\Theta}}}{\partial \tilde{\mathbf{a}}} \frac{\partial \tilde{\mathbf{a}}}{\partial \theta^J} \tag{3.2.15}$$

where, according to Eq. (3.2.8) and Eq. (3.2.3),

$$\frac{\partial L_{\mathbf{\Theta}}}{\partial \tilde{\mathbf{a}}} = \alpha_1 \frac{\partial L_1}{\partial \tilde{\mathbf{a}}} + \alpha_2 \frac{\partial L_2}{\partial \tilde{\mathbf{a}}} + \alpha_3 \frac{\partial L_3}{\partial \tilde{\mathbf{a}}} \tag{3.2.16}$$

$$\frac{\partial \tilde{\mathbf{a}}}{\partial \theta^J} = \frac{\partial}{\partial \theta^J} f_Z(\mathbf{a}^J, \mathbf{d}^{J-1}; \theta^J) \tag{3.2.17}$$

It is noteworthy that the right-hand side of Eq. (3.2.17) is represented by f_Z instead of f_A as the network output $\tilde{\mathbf{a}}$ is derived from the last layer's *Z-Update* component. The gradient for the other parameters can be calculated in a similar fashion.

The stochastic gradient algorithm will continue until predetermined termination conditions are met. Upon completion of training, the network is capable of efficiently estimating the abundance vector for a new HSI spectrum with a single forward computation.

3.3 Blind unmixing Network

We can also construct a neural network that can estimate both endmembers and their abundances from HSI reflectance data through unsupervised training.

3.3.1 Blind Unmixing Network Structure

Our ADMM-based blind unmixing network draws inspiration from the previously proposed abundance estimation network. Specifically, in line with Eq. (2.2.1), the network architecture is an extension of the UAENet by incorporating an additional linear layer, resulting in the reconstruction of the original HSI spectrum through the estimation of the abundances, as outlined below:

$$\tilde{\mathbf{y}} = \tilde{\mathbf{E}}\tilde{\mathbf{a}} \tag{3.3.1}$$

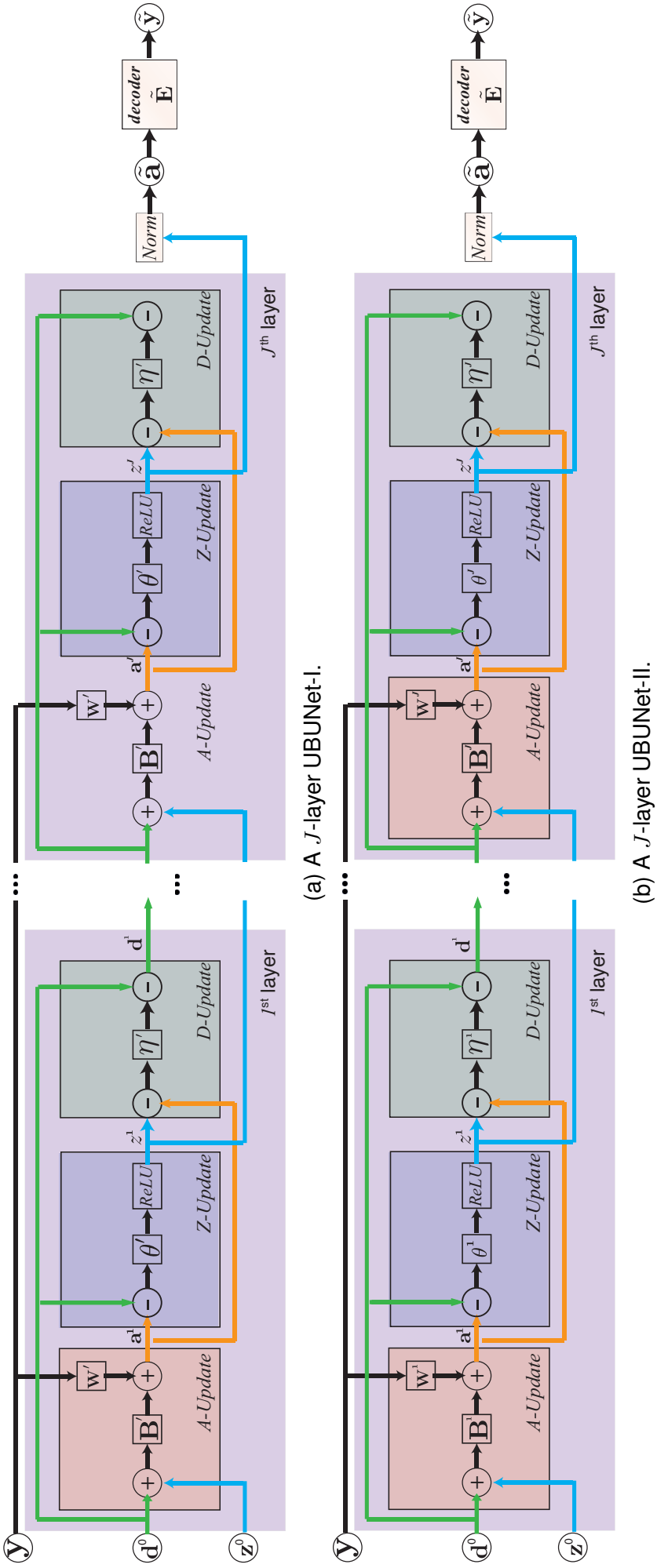
where $\tilde{\mathbf{a}}$ represents the estimated abundance by UAENet, and $\tilde{\mathbf{y}}$ represents the reconstructed reflectance spectrum. The matrix $\tilde{\mathbf{E}}$, which is the weight of the additional linear layer, models the endmember signatures and must be non-negative due to physical constraints. It is worth noting that this auto-encoder-like structure has also been utilized in methods such as uDAS [29] and MNN-BU [18]. In contrast to uDAS, which employs a traditional auto-encoder with a black-box encoder, and MNN-BU, which utilizes an auto-encoder with an ISTA-based encoder, our network originates from unfolding the ADMM algorithm.

Two separate blind unmixing networks can be constructed in correspondence with the prior abundance estimation networks. The network is referred to as UBUNet-I if the parameters are shared across its layers, and as UBUNet-II if the parameters are not shared where the acronym BU stands for blind unmixing. The structure of a J-layer UBUNet is illustrated in Fig. 3.7.

3.3.2 Network Initialization and Training strategies

Initialization Approach

The neural networks are initialised using warm initialization instead of random initialization for the parameters. The parameters related to



(a) A J-layer UBUNet-I. (b) A J-layer UBUNet-II.

Figure 3.7: The structures of UBUNet. (a) a J-layer UBUNet-I, which is constructed by appending UAENet-I with a linear decoder layer. (b) a J-layer UBUNet-II, which is constructed by appending UAENet-II with a linear decoder layer.

the *A-update*, *Z-update* and *D-update* component in each layer of the UBUNet are initialized using the same method employed for initializing the corresponding parameters in the UAENet. Conversely, the additional decoding layer in the UBUNet is initialized to correspond to an estimate of the endmember signature matrix, which can be acquired through the process described in Sec. 3.2.3.

Training Approach

The proposed networks, similar to well-known auto-encoders, can be trained in an unsupervised manner. Given access to a training set $D = \{\mathbf{y}_i\}_{i=1}^N$ consisting of N reflectance spectra $\mathbf{y}_i$, the network is trained by minimising the loss function as follows:

$$L_{\mathbf{\Theta}} = MSE(\{\mathbf{y}_i, \tilde{\mathbf{y}}_i\}_{i=1}^N) = \frac{1}{N} \sum_{\{\mathbf{y}_i\} \in D} \|\mathbf{y}_i - \tilde{\mathbf{y}}_i\|_2^2 \tag{3.3.2}$$

where $\tilde{\mathbf{y}}_i$ is defined in Eq. (3.3.1). Note that this loss function is simpler than the previous one because the lack of abundance in the training set prohibits the use of either AAD or AID.

After training, the UBUNet will output a reconstruction of reflectance spectra $\tilde{\mathbf{y}}_i$. In line with Eq. (3.3.1), the UBUNet is built by combining UAENet and a linear decoder layer with parameter $\tilde{\mathbf{E}}$, hence $\tilde{\mathbf{y}}_i$ can be represented as:

$$\tilde{\mathbf{y}}_i = \tilde{\mathbf{E}} f_{AE}(\mathbf{y}_i) \tag{3.3.3}$$

where $f_{AE}(\mathbf{y}_i)$ is the output of UAENet and represents the estimated abundance for the reflectance spectra $\mathbf{y}_i$, i.e. $\tilde{\mathbf{a}}_i = f_{AE}(\mathbf{y}_i)$. The parameter $\tilde{\mathbf{E}}$ is the estimated endmember matrix.

Parameter Update Rule

The parameters of the network are optimized using the stochastic gradient descent algorithm ADAM [85]. The gradients for each parameter

are computed through the back-propagation algorithm. Specifically, the gradient of the loss function with respect to the decoder parameters $\tilde{\mathbf{E}}$ and the encoder output $\tilde{\mathbf{a}}_i$ are expressed as follows:

$$\frac{\partial L_{\mathbf{\Theta}}}{\partial \tilde{\mathbf{E}}} = \frac{2}{N} \sum_{\{\mathbf{y}_i\} \in D} (\tilde{\mathbf{E}}\tilde{\mathbf{a}}_i - \mathbf{y}_i)\tilde{\mathbf{a}}_i^T \tag{3.3.4}$$

$$\frac{\partial L_{\mathbf{\Theta}}}{\partial \tilde{\mathbf{a}}_i} = \frac{2}{N} \tilde{\mathbf{E}}^T (\tilde{\mathbf{E}}\tilde{\mathbf{a}}_i - \mathbf{y}_i) \tag{3.3.5}$$

The remaining gradients are calculated in the same manner as for UAENet. It is important to note that during each update, the decoder parameter $\tilde{\mathbf{E}}$ is constrained to be non-negative, consistent with physical constraints.

3.4 Network Structure/Complexity Analysis

In this section, we will examine network complexity from both structural and parametric perspectives.

3.4.1 Network Structure

It is noteworthy to compare the structure of the ADMM-based layer with other layers used in networks that solve the unmixing problem. As shown in Fig. 3.8, we compare three network structures: (1) a typical ResNet [86] known for its improved performance through skip connections, (2) a network layer derived from solving the CSR problem using the ISTA solver [26], and (3) a network layer derived from solving the CSR problem using the proposed ADMM method. The ADMM-based layer has significantly more shortcuts and skip connections than the competing layers, as ResNet only skip-connect adjacent operations and the ISTA-based layer includes only shortcuts from the input. This advantage of the ADMM-based layer is expected to lead to better unmixing results, as previous research has shown that rich connections

improve neural network performance [87–90].

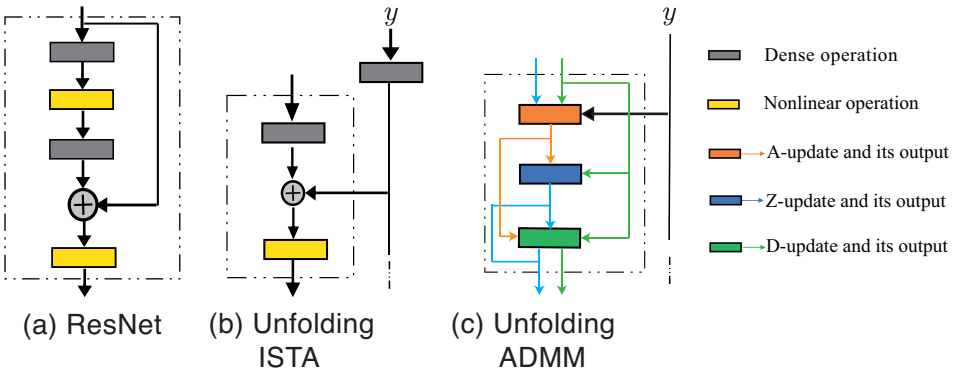


Figure 3.8: Comparison of layer structures of various networks. (a) ResNet layer structure, which introduces shortcuts between adjacent operations. (b) ISTA based layer structure, which includes skipping connections only from the input. (c) Our ADMM based layer structure, which contains both types of connections.

3.4.2 Network Complexity

It is also worthwhile to compare the number of learnable parameters in the proposed UAENet-I & II with existing state-of-the-art architectures. The number of parameters serves as an indicator of network complexity. As shown in [18], unfold networks generally have fewer parameters compared to conventional networks. Here, we briefly compare our proposed UAENet with other networks derived from unfolding ISTA [18].

Table 3.1: Number of learnable parameters: Abundance estimation

method	MNN-AE-1	MNN-AE-2	UAENet-I	UAENet-II
♯	1.3×10^3	2.7×10^3	1.3×10^3	2.7×10^3

Table 3.2: Number of learnable parameters: Blind Unmixing

method	MNN-BU-1	MNN-BU-2	UBUNet-I
♯	2.7×10^3	5.4×10^3	2.7×10^3
method	UBUNet-II	UnDIP	EGU-Net-pw
♯	5.4×10^3	1.3×10^6	1.9×10^5

In UAENet-I, where parameters are shared across layers, the total number of parameters is $(r^2 + rp + 2)$, where r^2 corresponds to $\mathbf{B}'$, rp corresponds to $\mathbf{W}'$, and the remaining parameters relate to the scalars θ' and η'. In UAENet-II, where parameters are specific to each layer, there are $(r^2 + rp + 2)J$ learnable parameters, with J being the number of iteration blocks/layers in the network. In comparison, in networks derived from unfolding ISTA such as MNN-AE-1 (parameters shared) and MNN-AE-2 (parameters specific), the number of parameters is $(r^2 + rp + 1)$ and $(r^2 + rp + 1)J$, respectively. The number of parameters for various abundance estimation and blind unmixing networks are reported in Table 3.1 and Table 3.2, respectively.

In conclusion, the proposed network has fewer parameters than most state-of-art unmixing networks such as UnDIP and EGU-Net. This lighter model is less likely to overfit [84] and is expected to outperform competing networks.

3.5 Experiments

We now assess the efficacy of our proposed methods UAENet and UBUNet by comparing them with some of the state-of-the-art unmixing techniques.

3.5.1 Performance Metrics

To evaluate the performance of different algorithms, we use commonly accepted metrics from the literature, such as the root mean square error (RMSE) and abundance angle distance (AAD) [15], between the true abundance vector $\mathbf{a}_k$ for the k^{th} pixel and its estimated value $\tilde{\mathbf{a}}_k$. These metrics are given by:

$$RMSE_k = \sqrt{\frac{1}{r}\sum_{i=1}^{r}(a_{i,k} - \tilde{a}_{i,k})^2} \tag{3.5.1}$$

$$AAD_k = \frac{180}{\pi} \cos^{-1}\left(\frac{\mathbf{a}_k^T \tilde{\mathbf{a}}_k}{\|\mathbf{a}_k\|_2 \|\tilde{\mathbf{a}}_k\|_2}\right) \tag{3.5.2}$$

These metrics are then averaged over all pixels to obtain the final scalar values.

For endmember estimation, we use the Spectral Angle Distance (SAD) between a true endmember signature $\mathbf{e}_i$ and its estimated value $\tilde{\mathbf{e}}_i$ to measure dissimilarity. SAD is defined as:

$$SAD_i = \frac{180}{\pi} \cos^{-1}\left(\frac{\mathbf{e}_i^T \tilde{\mathbf{e}}_i}{\|\mathbf{e}_i\|_2 \|\tilde{\mathbf{e}}_i\|_2}\right) \tag{3.5.3}$$

This metric is then averaged over all endmembers to obtain the final scalar value.

3.5.2 Data

To evaluate the performance of various unmixing algorithms, we employed the HSI dataset synthesis procedure as outlined in [18] for assessing the efficacy of linear unmixing algorithms. The procedure involves the following steps:

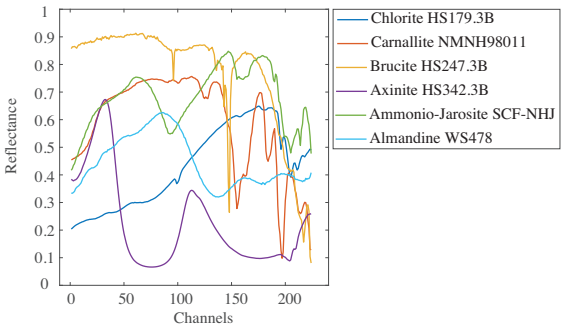


Figure 3.9: Endmember signatures for synthetic data.

- *Endmember generation.* Endmembers are selected from the USGS spectral library (splib06) [81], which contains spectral reflectance values for different minerals over 224 channels. Six spectral signatures are randomly selected and form a 224×6 endmember matrix, as shown in Fig. 3.9.

- *Abundance generation.* A synthetic image of size $a^2 \times a^2$ pixels is divided into a^2 disjoint patches of size $a \times a$ pixels. Two endmembers are randomly selected for all pixels of a patch and assigned with fractions γ and $1-\gamma$, while the remaining four endmembers are assigned with a value of 0. The abundance map is then convolved with a Gaussian filter of size $(a+1) \times (a+1)$ with variance 2, followed by a pixel-wise re-scaling to meet the ASC constraint. In this thesis, we set $a = 10$ and $\gamma = 0.8$.

- *Mixing process.* We generate synthetic data for linear mixing models by following the linear model in Eq. (2.2.2).

- *Noise contamination.* Finally, we add additive white Gaussian noise (AWGN) to the generated HSI data. The signal-to-noise ratio (SNR) is defined as $SNR = 10\log_{10}\left(E[\mathbf{x}^T\mathbf{x}]/E[\mathbf{n}^T\mathbf{n}]\right)$, where $\mathbf{x}$ represents the original, noise-free HSI data, and $\mathbf{n}$ is the added noise.

3.5.3 Experiments setup

We now present the efficacy of UAENet and UBUNet for linear abundance estimation and linear blind unmixing tasks, respectively. We compare our UAENet-I and UAENet-II approaches with the unfolding-based learning algorithms MNN-AE-1 and MNN-AE-2 [18] for the task of abundance estimation. The experiments, unless otherwise specified, use the default setup where the synthetic HSI dataset is contaminated with AWGN noise with an SNR of 15 dB. The network is trained using a training set comprised of 1000 randomly selected pixels from the synthetic HSI dataset, with the remaining pixels reserved for testing. The hyperparameters $\alpha_1, \alpha_2, \alpha_3$ in Equation (3.2.8) are determined through experimentation on the training dataset. The reported performance is based on the evaluation of the test dataset. The networks,

UAENet and MNN-AE, consist of two layers (iteration blocks), as we have found that additional layers do not lead to significant performance improvements [18]. We train the networks using the Adam optimizer with a learning rate of $1e-4$, a batch size of 64, and 500 epochs.

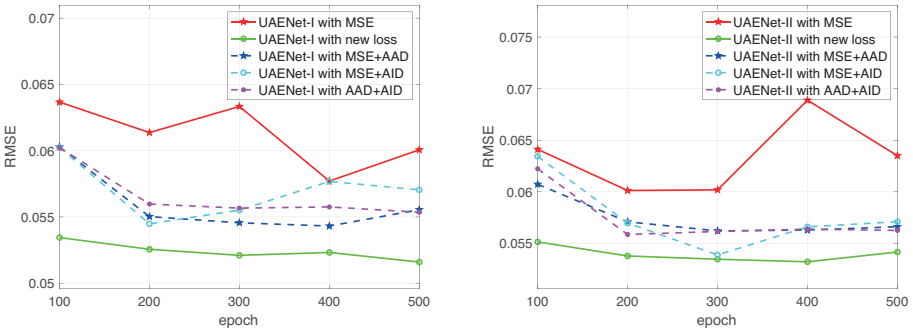
In the blind unmixing scenario, we evaluate our proposed unsupervised UBUNet against several state-of-the-art learning-based blind unmixing algorithms, including MNN-BU [18], UnDIP [36], and EGU-Net-pw [4]. In this experiment, by default, the synthetic HSI dataset is contaminated with AWGN noise of an SNR of 25 dB. The network is trained and evaluated on the entire dataset, as it is an unsupervised learning algorithm. The UBUNet and MNN-BU models consist of two layers (iteration blocks) followed by a linear (decoding) layer and are trained using the Adam optimizer with a learning rate of $1e-4$, batch size of 64, and 500 epochs.

In our proposed methods, we adopt the training strategies, initialization strategies, and hyper-parameter settings that have been previously reported. On the other hand, the competing methods employ the default hyper-parameter settings as outlined in their original papers.

3.5.4 Impact of the proposed composite loss

In this experiment, we assess the effect of the proposed composite loss function Eq. (3.2.8) on the abundance estimation task. The proposed composite loss function is compared with other loss function combinations, such as MSE+AAD, MSE+AID, and AAD+AID, to highlight the improvement it brings. We largely follow the default experimental settings, except that the number of training data is set to 256. The abundance RMSE metric as a function of training epoch is presented in Fig. 3.10. Although similar trends are observed for other metrics, such as the mean absolute error (MAE), we specifically report the results using RMSE to maintain consistency with the other

comparative analyses presented in this study. It can be observed that for both network structures, the new composite loss function delivers better and more stable results. This is due to the fact that the MSE loss function has an implicit assumption [84] of a Gaussian distribution for the estimated abundance, which is inconsistent with ASC and ANC constraints. In contrast, the proposed new loss function reduces the impact of such assumptions. It is worth noting that with sufficient training duration, alternative combinations of loss functions may also achieve comparable performance to the proposed approach.



(a) UAENet-I. (b) UAENet-II.

Figure 3.10: The impact of different composite loss on abundance estimation performance. (a) UAENet-I. (b) UAENet-II.

3.5.5 Impact of number of layers

In this experiment, we assess the effect of the number of layers on the abundance estimation performance of various approaches. We use the default experimental setup previously described, with the exception of varying the number of layers from 1 to 7. The performance of the approaches, as measured by the RMSE metric, is presented in Fig. 3.11 as a function of the number of layers. It is evident that for an unfolding-based unmixing network, the number of layers does not significantly affect the final performance, unlike a conventional neural network that usually benefits from a deeper structure [82]. This difference can be attributed to the strong inductive bias inherent in the

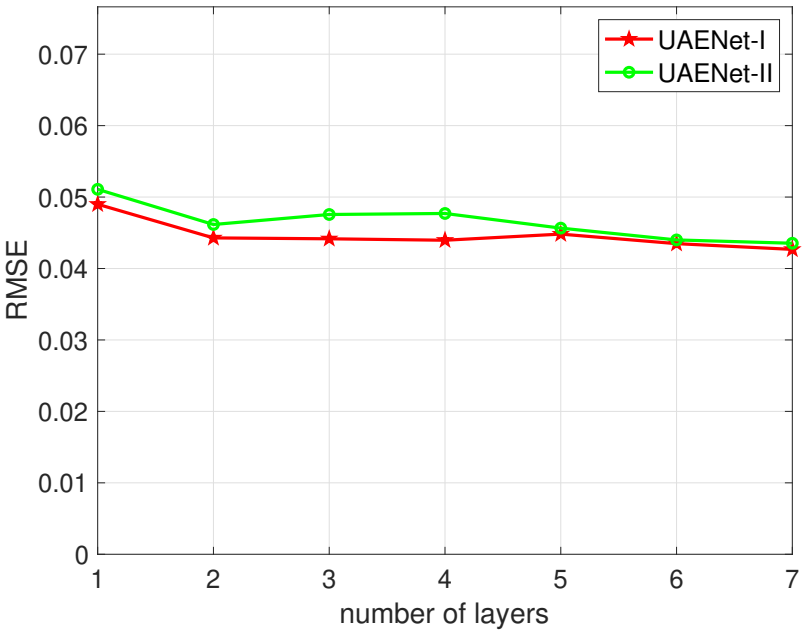


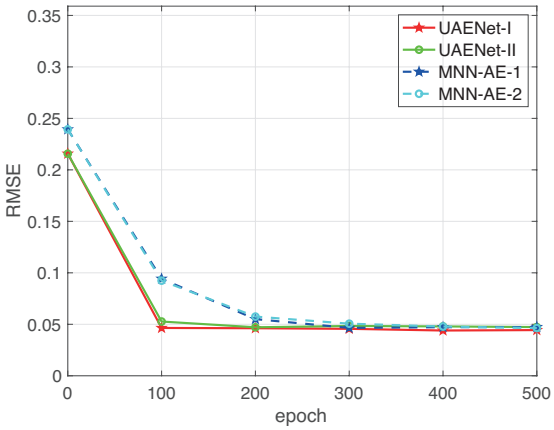
Figure 3.11: The impact of layers on abundance estimation performance.

unfolding network architecture, allowing for a strong performance to be achieved with only a small number of layers. Thus, we choose to use 2 layers in our subsequent experiments.

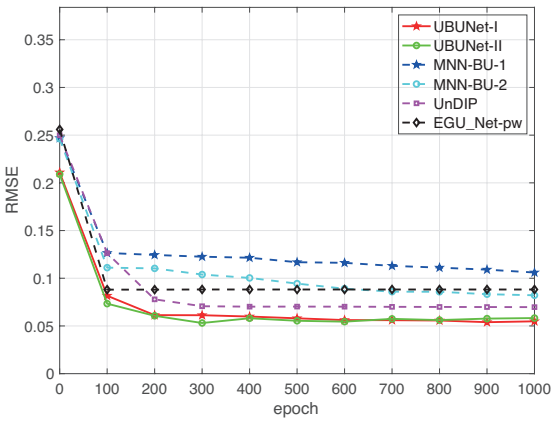
3.5.6 Impact of epochs

In this study, we evaluate the convergence performance of our proposed methods in comparison to other competing methods. The abundance estimation case is evaluated by varying the number of training epochs from 0 to 500, while the blind unmixing case is evaluated by setting the number of epochs to range from 0 to 1000. In Fig. 3.12a, the abundance estimation performance of UAENet and MNN-AE is presented as a function of epoch. Fig. 3.12b and Fig. 3.12c show the blind unmixing performance vs. the number of epochs.

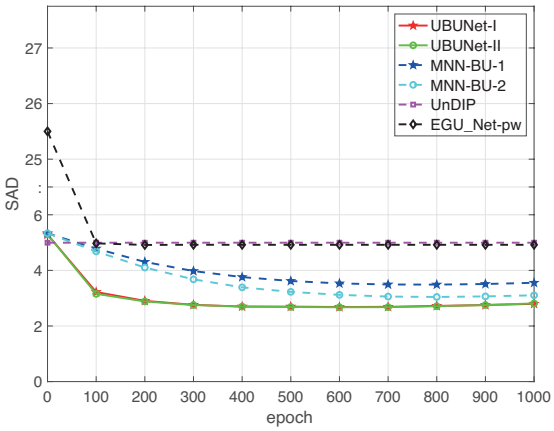
Our proposed methods show faster convergence compared to MNN-AE and MNN-BU. Specifically, in the abundance estimation case, UAENet converges in around 100 training epochs, while MNN-AE networks take 300 training epochs to converge. In the blind unmixing case,



(a) Abundance estimation: RMSE versus epochs.



(b) Blind unmixing: Abundance RMSE versus epochs.



(c) Blind unmixing: Endmember SAD versus epochs.

Figure 3.12: The impact of training epochs on the performance. (a) Abundance estimation case: RMSE versus epochs. (b) Blind unmixing case: Abundance RMSE versus epochs. (c) Blind unmixing case: Endmember SAD versus epochs.

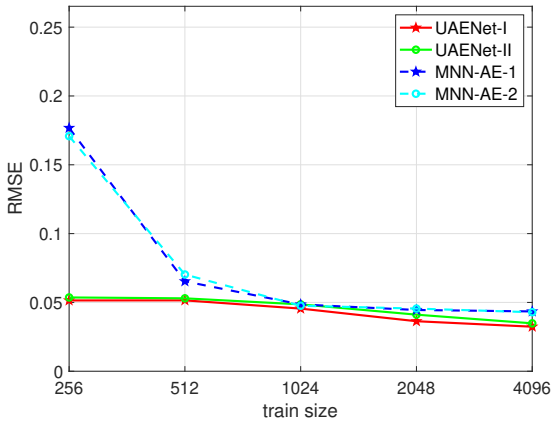
.

UBUNet achieves impressive performance at around 200 epochs, while MNN-BU needs around 600 epochs. We attribute this to the faster convergence of ADMM-based solvers compared to ISTA-based ones and the more complex weighted loss function adopted in our learning algorithms that promotes additional dissimilarity. It is worth noting that prior to training, at epoch 0, the proposed ADMM-based unmixing network has an RMSE of approximately 0.2, while the RMSE of the ISTA-based unmixing networks MNN-AE and MNN-BU is around 0.25, highlighting the superiority of the proposed ADMM-based unmixing network architecture. Although the state-of-the-art UnDIP and EGU-Net achieve a similar convergence speed, our proposed blind unmixing networks show better unmixing performance. Particularly, in terms of SAD, our network achieves 2.5 while both UnDIP and EGU-Net have a value of 5.0, resulting in 2X better performance than the state-of-the-art methods. This improvement in performance is a result of the fact that UnDIP and EGU-Net depend on endmember extraction algorithms for guidance, limiting their overall performance.

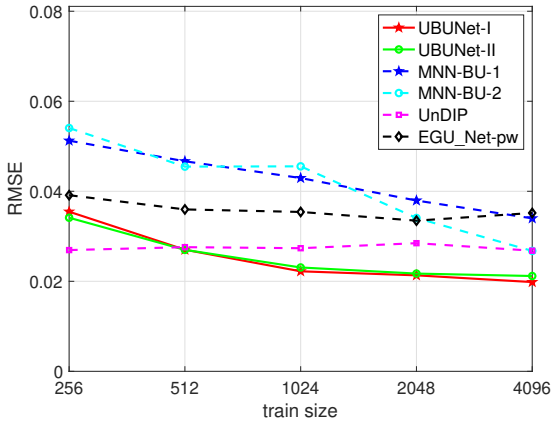
3.5.7 Impact of training size

We evaluate the performance of the various methods in relation to the number of training data points. For both abundance estimation and blind unmixing, we use a training set composed of randomly selected pixels (signatures) ranging from 256 to 4096, while maintaining the default settings for other experimental conditions.

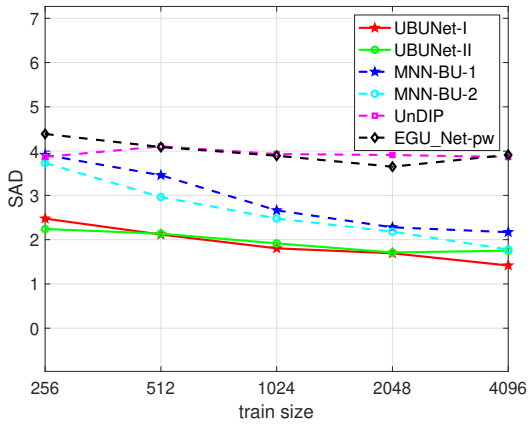
The performance of abundance estimation and endmember estimation *vs.* training size are presented in Fig. 3.13. Our proposed methods exhibit superior performance compared to competing methods when the size of the training dataset is small. Specifically, for the abundance estimation scenario, UAENet outperforms MNN-AE when the size of the training dataset is as low as 256, which may be attributed to



(a) Abundance estimation: RMSE versus Train size.



(b) Blind unmixing: abundance RMSE versus Train size.



(c) Blind unmixing: endmember SAD versus Train size.

Figure 3.13: The impact of train size on the performance. (a) Abundance estimation case: RMSE. (b)Blind unmixing case: RMSE. (c) Blind unmixing case: SAD in degree.

the more complex structure in the ADMM-based network compared to ISTA-based networks and the more complex loss function. In the blind

unmixing scenario with a small training dataset, UBUNet also demonstrates superior performance in terms of both abundance estimation (RMSE) and endmember estimation (SAD). As expected, the performance of most networks improves with an increase in the training set size.

Nevertheless, state-of-the-art algorithms such as UnDIP and EGU-Net generally have poorer unmixing performance. For instance, the SAD for UnDIP and EGU-Net is approximately 4.0, while the SAD for the proposed method is around 2.0. This is due to the fact that UnDIP and EGU-Net are reliant on the guidance provided by existing endmember extraction algorithms, thereby limiting their performance.

3.6 Summary

We have introduced new hyperspectral unmixing networks based on unfolding techniques. By taking the traditional constrained sparse regression approach to linear unmixing, we demonstrate how the ADMM solver can be converted to a neural network architecture with interpretable learning modules that have a similar counterpart in machine learning. Our approach unifies the benefits of both model-based and learning-based unmixing methods and can be trained in a supervised or unsupervised manner with newly proposed weighted loss functions. Additionally, the proposed neural network architecture has a rich structure, including skipping connections and residual blocks, and a small number of learnable parameters, leading to superior performance in image analysis and processing tasks. The effectiveness of the proposed methods is further supported by the experimental results. Although the technique may yield unmixing outcomes, there is no guarantee of their physical significance. Furthermore, this approach exhibits inadequacy in addressing complex scenarios involving nonlinear

mixing effects.

Chapter 4

Blind unmixing using double deep image prior

The unmixing network described in Chapter 3 has an efficient and interpretable design, but it cannot always ensure physically plausible unmixing results. Additionally, it lacks the capability to tackle nonlinear unmixing problems. To address these limitations, we present a general unsupervised framework that can handle both linear and nonlinear blind unmixing scenarios, based on the Deep-Image-Prior (DIP) technique. To obtain meaningful unmixing results, we also propose a composite loss function that can be used in both linear and nonlinear unmixing scenarios. We now present the proposed general framework for linear and nonlinear blind hyperspectral unmixing tasks using double deep image prior (BUDDIP).

4.1 Problem Formulation

In this chapter, we employ the matrix formulation for solving the hyperspectral image (HSI) unmixing problem as described in Eq. (2.2.2) and Eq. (2.2.4) for linear and nonlinear cases, respectively. Our discussion begins by focusing on the linear case, while the nonlinear case will be addressed naturally as we develop BUDDIP. The objective of blind

linear unmixing is to estimate the endmembers $\mathbf{E}$ and abundances $\mathbf{A}$ based solely on HSI reflectances $\mathbf{Y}$. A commonly used approach is to minimize the following equation [91]:

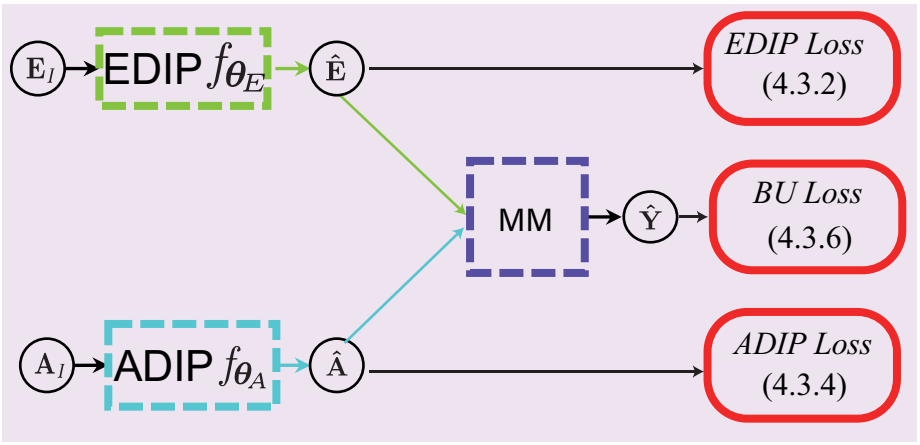
$$\begin{aligned} \hat{\mathbf{E}}, \hat{\mathbf{A}} = \textit{arg}\min_{\mathbf{E},\mathbf{A}} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{A}\|_F^2 + R(\mathbf{A}) \\ s.t., \mathbf{E} \geq \mathbf{0}, \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \end{aligned} \tag{4.1.1}$$

where $R(\mathbf{A})$ represents a regularization term that depends on the abundance matrix $\mathbf{A}$, such as total variation (TV) [91]. The selection of R is often influenced by prior knowledge of the task. The optimisation problem Eq. (4.1.1) is usually solved using the multiplicative update rule [14] or a two-stage cyclic descent method [91], alternating between optimizing $\mathbf{A}$ for fixed $\mathbf{E}$ and vice versa.

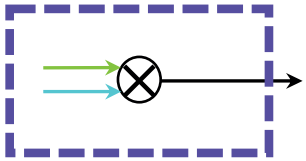
4.2 BUDDIP

The proposed BUDDIP network is a self-supervised end-to-end network comprised of three modules: EDIP, ADIP, and MM, as illustrated in Fig. 4.1(a). EDIP is dedicated to determining endmember estimates, while ADIP focuses on estimating abundances. Upon obtaining the endmember and abundance estimates, the outputs of both EDIP and ADIP, $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$, are integrated into the MM module to generate a reconstructed hyperspectral spectrum, $\hat{\mathbf{Y}}$.

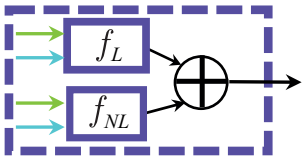
The proposed BUDDIP is a versatile unmixing technique that offers the option to choose a mixing module (MM). If MM chose to be LMM, it becomes linear BUDDIP (L-BUDDIP) and can effectively handle linear unmixing tasks. If NLMM is chosen, it transforms into non-linear BUDDIP (NL-BUDDIP) and can tackle non-linear unmixing problems. To achieve meaningful unmixing results, BUDDIP is trained with the guidance of endmembers and abundances, generated from any prevalent unmixing methods such as SiVM [1] + FCLS [2]. However, un-



(a) BUDDIP



(b) when MM=LMM



(c) when MM=NLMM

Figure 4.1: (a) The general architecture of the proposed BUDDIP. It consists of three modules: endmember estimation DIP (EDIP), abundance estimation DIP (ADIP) and mixing module (MM). The output of EDIP is denoted by the green arrow while that of ADIP is by the blue arrow. (b) when MM performs LMM as defined in Eq. (4.2.5), we coined the whole model in (a) as L-BUDDIP, which can solve the linear blind unmixing problem. (c) While MM performs NLMM as defined in Eq. (4.2.6), we coined the whole model in (a) as NL-BUDDIP, which can solve the nonlinear blind unmixing problem. f_L and f_{NL} are defined in Eq. (4.2.7) and Eq. (4.2.8), respectively.

like other guidance-based unmixing networks such as UnDIP [36] and EGU-Net [4], BUDDIP has the potential to outperform the guidance it was trained with.

4.2.1 Endmember Estimation using DIP

We first present how to construct EDIP using DIP techniques. Similar to the two-stage cyclic descent method [91], we assume that, during the endmember estimation stage, we have access to an estimation of the abundance matrix $\mathbf{A}_G$. This can be obtained through the use of any existing abundance estimation algorithms such as FCLS [2]. As a

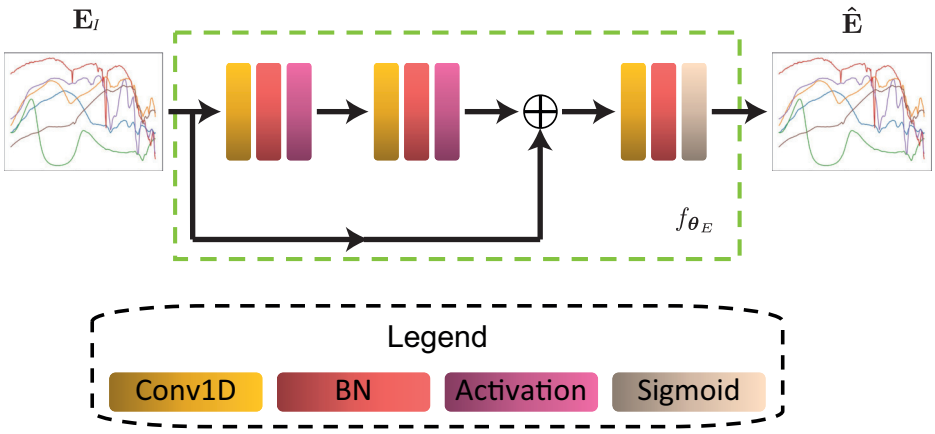


Figure 4.2: The architecture of the proposed EDIP. We propose to give a meaningful input $\mathbf{T}_E = \mathbf{E}_I$, where $\mathbf{E}_I$ is an estimation of endmember generated by existing methods, such as SiVM [1]. The outputs of the main branch and the skipped input are added and forwarded to the final output block.

result, the optimization problem in Eq. (4.1.1) can be reduced to:

$$\hat{\mathbf{E}} = \textit{arg}\min_{\mathbf{E}} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{A}_G\|_F^2 \quad s.t., \mathbf{E} \geq \mathbf{0} \tag{4.2.1}$$

In this work, based on the concept of DIP, we propose to estimate the endmembers through a DIP network (EDIP) $f_{\boldsymbol{\theta}_E}$ with learnable parameters $\boldsymbol{\theta}_E$ and an input $\mathbf{T}_E$. This results in the following optimization problem:

$$\boldsymbol{\theta}_E^* = \textit{arg}\min_{\boldsymbol{\theta}_E} \frac{1}{2}\|\mathbf{Y} - f_{\boldsymbol{\theta}_E}(\mathbf{T}_E)\mathbf{A}_G\|_F^2 \tag{4.2.2}$$

In the original DIP [37] and the unmixing work using DIP UnDIP [36], the network is fed with a random input, i.e., $\mathbf{T}_E = \boldsymbol{z}_E$, where $\boldsymbol{z}_E$ is either Gaussian noise [36] or uniform noise between 0 and 0.1 [37]. However, this random input strategy has a major drawback: it does not contain any relevant information about the task or data at hand. Additionally, as noted in [92], when the network is given both noisy observations and random noise, it tends to ignore the noise. In this work, we aim to provide the DIP network with a more meaningful input rather than random noise. Given the numerous impressive unmixing works in the hyperspectral literature, we propose to use existing unmixing algo-

rithms, such as SiVM [1], to generate an estimate of the endmembers, $\mathbf{E}_I$, which is then used as the input to the proposed EDIP network, i.e., $\mathbf{T}_E = \mathbf{E}_I$. This can be seen as a noisy estimation of the true endmembers.

With the proposed input strategy, the network will have a good starting point and only need to learn the difference between the desired output and the input. This further drives the design of the EDIP network $f_{\boldsymbol{\theta}_E}$. In particular, we adopt a ResNet-like architecture [86] as the skip-connections in ResNet can facilitate the network in learning the difference between input and output. Unlike the traditional ResNet structures used in DIP and UnDIP, we use a simpler network structure as depicted in Fig. 4.2.

The proposed EDIP network $f_{\boldsymbol{\theta}_E}$ is based on a block, which is comprised of a Convolutional layer, a Batch normalization layer, and an Activation layer. This type of block is widely used in neural network architectures such as ResNet [37, 93]. However, unlike the conventional 2D Convolutional layer in literature, we use a 1D Convolutional layer in the proposed EDIP. This is because 2D Convolutional networks are typically applied to image-related problems represented as 3D tensors, whereas the input and output in the proposed EDIP network are endmember signatures represented as 2D matrices. The 1D Convolution is performed over the spectral band dimension p to capture the spectral information, as there is no spatial information in the endmember matrix. The block is repeated twice. The skip connection shown in Fig. 4.2 is used to force the network to learn the difference between input and desired output. The output from the main branch and the skip connection are added and then fed into the final output block. This block is similar to the previous one, except that the activation layer is replaced with the Sigmoid activation layer to satisfy the ENC constraint. After the parameters $\boldsymbol{\theta}_E^\star$ are learned, the estimated endmember is given by

$\hat{\mathbf{E}} = f_{\theta_E^\star}(\mathbf{E}_I)$.

4.2.2 Abundance Estimation using DIP

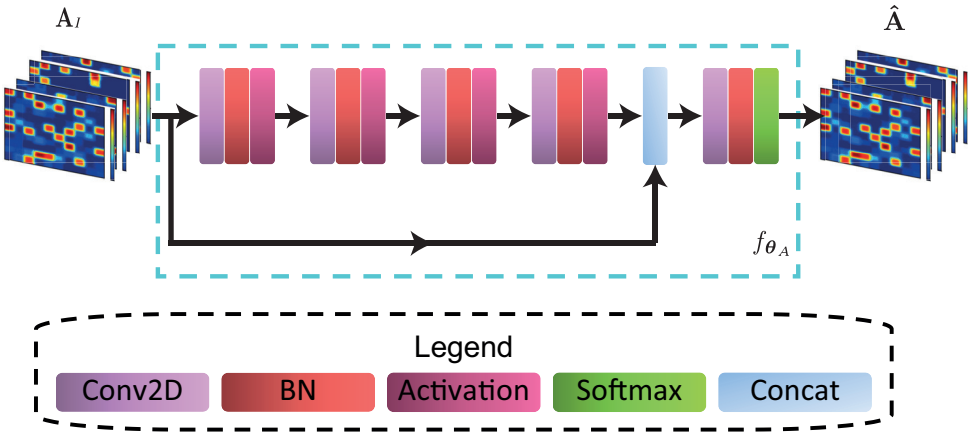


Figure 4.3: The architecture of the proposed ADIP. We propose to give a meaningful input $\mathbf{T}_A = \mathbf{A}_I$, where $\mathbf{A}_I$ is an estimation of abundances generated by existing methods, such as FCLS [2]. The outputs of the skip connection and the main branch are concatenated and forwarded to the final output block.

In this section, we present the derivation of the DIP network for abundance estimation, known as the Abundance DIP (ADIP) network. We start with the assumption that, at the abundance estimation stage, an estimate of the endmembers, $\mathbf{E}_G$, is available through a method such as SiVM [1]. The optimization problem Eq. (4.1.1) then is reduced to the abundance estimation problem formulated as:

$$\begin{aligned} \hat{\mathbf{A}} = \textit{arg}\min_{\mathbf{A}} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}_G\mathbf{A}\|_F^2 + R(\mathbf{A}) \\ s.t., \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \end{aligned} \tag{4.2.3}$$

We propose estimating the abundances using the ADIP network, f_{θ_A}, with learnable parameters $\boldsymbol{\theta}_A$ and input $\mathbf{T}_A$. The resulting optimization problem is given by:

$$\boldsymbol{\theta}_A^* = \textit{arg}\min_{\boldsymbol{\theta}_A} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}_G f_{\boldsymbol{\theta}_A}(\mathbf{T}_A)\|_F^2 \tag{4.2.4}$$

The regulariser $R(\cdot)$ is dropped according to the DIP techniques. As

with the EDIP network, we aim to provide the ADIP network with a meaningful input, rather than random noise. To this end, we use an existing unmixing algorithm to generate an estimate of the abundance, $\mathbf{A}_I$, which serves as the input for the ADIP network: $\mathbf{T}_A = \mathbf{A}_I$.

Similar to the EDIP network, the architecture of the ADIP network is designed to learn the difference between the input and desired output, The ADIP network, shown in Fig. 4.3, consists of four blocks to capture the rich information in the abundance maps, which are represented as a 3D tensor (see Eq. (2.2.2)). Unlike the EDIP network, the ADIP network uses a 2D convolutional layer to capture spatial information. Additionally, the output layer of the ADIP network uses softmax to satisfy ASC and ANC constraints. After learning the parameters, $\boldsymbol{\theta}_A^\star$, the estimated abundance is given by $\hat{\mathbf{A}} = f_{\boldsymbol{\theta}_A^\star}(\mathbf{A}_I)$.

4.2.3 Blind Unmixing using Double DIP

Finally, we present the proposed general framework for blind unmixing using double DIP (BUDDIP). Upon obtaining the estimated endmember and abundance matrices, $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$, through EDIP and ADIP, respectively, they can be fed into a Mixture Module (MM) to immediately generate a reconstruction of the observed HSI image. The MM, which will be detailed later, has a flexible structure that depends on the selected physical model. After integrating MM with EDIP and ADIP, the resulting architecture of the proposed general BUDDIP framework is depicted in Fig. 4.1. This framework can be utilized to tackle both linear and nonlinear unmixing problems by applying different MM models.

Linear BUDDIP (L-BUDDIP)

In the case of LMM, the MM performs the linear mixing process as specified in Eq. (2.2.2). Specifically, the MM computes the reconstruction of the observed HSI image, $\hat{\mathbf{Y}}$, using the estimated endmember

and abundance, $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$, by simply multiplying them as follows:

$$\hat{\mathbf{Y}} = \hat{\mathbf{E}}\hat{\mathbf{A}} \tag{4.2.5}$$

This process is visualized in Fig. 4.1(a) and (b). We refer to this configuration of the proposed general framework as the L-BUDDIP.

Nonlinear BUDDIP (NL-BUDDIP)

In the case of NLMM, the observed HSI image can be reconstructed by feeding $\hat{\mathbf{E}}, \hat{\mathbf{A}}$ into the MM, which performs the nonlinear mixing process as described in Eq. (2.2.4). Specifically, the HSI reconstruction $\hat{\mathbf{Y}}$ is obtained by:

$$\hat{\mathbf{Y}} = f_L(\hat{\mathbf{E}}, \hat{\mathbf{A}}) + f_{NL}(\hat{\mathbf{E}}, \hat{\mathbf{A}}) \tag{4.2.6}$$

where $f_L(\hat{\mathbf{E}}, \hat{\mathbf{A}})$ represents the linear part of the mixing process, given by:

$$f_L(\hat{\mathbf{E}}, \hat{\mathbf{A}}) = \hat{\mathbf{E}}\hat{\mathbf{A}} \tag{4.2.7}$$

On the other hand, the nonlinear part, $f_{NL}(\hat{\mathbf{E}}, \hat{\mathbf{A}})$ given by

$$f_{NL}(\hat{\mathbf{E}}, \hat{\mathbf{A}}) = \hat{\mathbf{O}} \tag{4.2.8}$$

is responsible for the nonlinear effects in NLMM, where $\hat{\mathbf{O}} = [\hat{\mathbf{o}}_1, ..., \hat{\mathbf{o}}_n] \in R^{p \times n}$. It can be adjusted depending on the selected nonlinear model. For example, in the case of the FM model, as described in Eq. (2.2.6) and Eq. (2.2.7), f_{NL} outputs $\hat{\mathbf{O}}$ as follows:

$$\hat{\mathbf{o}}_k = \sum_{i=1}^{r-1} \sum_{j=i+1}^{r} a_{i,k} a_{j,k} \mathbf{e}_i \odot \mathbf{e}_j \tag{4.2.9}$$

Fig. 4.1(a)&(c) depicts the architecture of the proposed NL-BUDDIP. In other nonlinear models with additional parameters, such as the GBM model, these parameters can be learned through another DIP network

according to [93].

4.3 Training Details

In this section, we delve into the optimization of parameters for BUDDIP.

Loss function

As previously discussed, BUDDIP generates three outputs: $\hat{\mathbf{E}}$ for endmembers, $\hat{\mathbf{A}}$ for abundances, and $\hat{\mathbf{Y}}$ for reconstructed HSI observation as illustrated in Fig.4.1. To train the network effectively, we employ six loss terms, two for each output. To clarify the notation used later, a summary of relevant notations is provided in Table 4.1.

Table 4.1: Summary of some notations.

$\mathbf{E}_I$, $\mathbf{A}_I$	meaningful input for EDIP and ADIP respectively
$\mathbf{E}_G$, $\mathbf{A}_G$	training guidance for ADIP and EDIP respectively
$\hat{\mathbf{E}}$, $\hat{\mathbf{A}}$	estimation of endmembers and abundances by the proposed network
$\mathbf{Y}$, $\hat{\mathbf{Y}}$, $\hat{\mathbf{Y}}^E$,$\hat{\mathbf{Y}}^A$	HSI observation and its reconstruction by the MM, EDIP, ADIP module, respectively
$\boldsymbol{\theta}_E$, $\boldsymbol{\theta}_A$	learnable parameters of proposed EDIP and ADIP network

To estimate endmembers $\hat{\mathbf{E}}$ by EDIP, in accordance with optimization problem Eq. (4.2.2), we can immediately propose the use of the following loss function

$$L_{EDIP}(\boldsymbol{\theta}_E) = \frac{1}{2}\|\mathbf{Y} - f_{\boldsymbol{\theta}_E}(\mathbf{E}_I)\mathbf{A}_G\|_F^2 \tag{4.3.1}$$

As previously discussed, the EDIP network $f_{\boldsymbol{\theta}_E}$ receives a meaningful input $\mathbf{E}_I$, which is an estimate of endmembers obtained from existing unmixing algorithms, and produces an estimate of endmembers

$\hat{\mathbf{E}} = f_{\boldsymbol{\theta}_E}(\mathbf{E}_I)$. The abundance matrix $\mathbf{A}_G$, also obtained from existing unmixing algorithms, serves as a guide for training the EDIP network, ensuring that it generates a meaningful estimate of endmembers. In this work, $\mathbf{E}_I$ and $\mathbf{A}_G$ are obtained using unmixing methods such as SiVM [1] and FCLS, respectively.

However, as discussed in Eq. (3.2.8), the MSE loss term has certain limitations and the unmixing performance can be further improved by incorporating additional loss terms based on other measurements, such as geometrical or information measurements. Therefore, we are motivated to propose an extension to the loss function Eq. (4.3.1) by including additional angle distance loss terms. Specifically, we suggest utilizing the following loss function:

$$L_{EDIP}(\boldsymbol{\theta}_E) = \alpha_1 \cdot L_{EMSE} + \alpha_2 \cdot L_{EAng} \tag{4.3.2}$$

where,

$$\begin{aligned} L_{EMSE} &= \frac{1}{2}\|\mathbf{Y} - \hat{\mathbf{Y}}^E\|_F^2 \\ L_{EAng} &= \frac{1}{n}\sum_{k=1}^{n} \frac{180}{\pi} \cos^{-1}\left(\frac{\mathbf{y}_k^T \hat{\mathbf{y}}_k^E}{\|\mathbf{y}_k\|_2 \|\hat{\mathbf{y}}_k^E\|_2}\right) \\ \hat{\mathbf{Y}}^E &= \hat{\mathbf{E}}\mathbf{A}_G \\ \hat{\mathbf{E}} &= f_{\boldsymbol{\theta}_E}(\mathbf{E}_I) \end{aligned} \tag{4.3.3}$$

The hyperparameters α_1 and α_2 in Eq. (4.3.2) control the relative importance of the corresponding loss term. Here, $\hat{\mathbf{Y}}^E$ represents the HSI reconstruction by EDIP $f_{\boldsymbol{\theta}_E}$ with the guidance of $\mathbf{A}_G$. Additionally, $\mathbf{y}_k$ and $\hat{\mathbf{y}}_k^E$ are the k^{th} pixel of HSI observation $\mathbf{Y}$ and HSI reconstruction $\hat{\mathbf{Y}}^E$, respectively. The proposed loss function in Eq. (4.3.2) extends the one in Eq. (4.3.1) by including an additional angle distance loss. The first loss term, L_{EMSE}, in Eq. (4.3.2) minimizes the discrepancy between $\mathbf{Y}$ and $\hat{\mathbf{Y}}^E$ in Euclidean distance. On the other hand, the second loss term, L_{EAng}, provides a measure of the disparity from a geometric perspective. As previously demonstrated in [94], the inclusion of this

additional angle loss term can enhance the performance of the unmixing network. It is possible to introduce an information divergence term similar to Eq. (3.2.8). However, doing so would require adding another loss weight, which could be computationally demanding to fine-tune.

We propose using a similar approach for abundance estimation $\hat{\mathbf{A}}$. Specifically, we suggest employing the loss function given by:

$$L_{ADIP}(\boldsymbol{\theta}_A) = \alpha_3 \cdot L_{AMSE} + \alpha_4 \cdot L_{AAng} \tag{4.3.4}$$

where,

$$\begin{aligned} L_{AMSE} &= \frac{1}{2}\|\mathbf{Y} - \hat{\mathbf{Y}}^A\|_F^2 \\ L_{AAng} &= \frac{1}{n}\sum_{k=1}^{n} \frac{180}{\pi} \cos^{-1}\left(\frac{\mathbf{y}_k^T \hat{\mathbf{y}}_k^A}{\|\mathbf{y}_k\|_2 \|\hat{\mathbf{y}}_k^A\|_2}\right) \\ \hat{\mathbf{Y}}^A &= \mathbf{E}_G \hat{\mathbf{A}} \\ \hat{\mathbf{A}} &= f_{\boldsymbol{\theta}_A}(\mathbf{A}_I) \end{aligned} \tag{4.3.5}$$

and α_3 and α_4 are loss weights of L_{AMSE} and L_{AAng}. Similar to the EDIP, $\hat{\mathbf{Y}}^A$ is the HSI reconstruction by ADIP $f_{\boldsymbol{\theta}_A}$, given the guidance $\mathbf{E}_G$. $\mathbf{y}_k$ and $\hat{\mathbf{y}}_k^A$ are the k^{th} pixel of HSI observation $\mathbf{Y}$ and HSI reconstruction $\hat{\mathbf{Y}}^A$, respectively. Moreover, the ADIP network also requires meaningful input and guidance during training, which is denoted by $\mathbf{A}_I$ and $\mathbf{E}_G$, respectively. Like before, these estimations are generated by existing unmixing methods, such as SiVM [1] and FCLS.

Additionally, for the reconstructed HSI observation $\hat{\mathbf{Y}}$ defined in Eq. (4.2.5) and Eq. (4.2.6), we impose an extra loss function called the blind unmixing (BU) loss, given by:

$$L_{BU}(\boldsymbol{\theta}_E, \boldsymbol{\theta}_A) = \alpha_5 \cdot L_{BUMSE} + \alpha_6 \cdot L_{BUAng} \tag{4.3.6}$$

where,

$$\begin{aligned} L_{BUMSE} &= \frac{1}{2}\|\mathbf{Y} - \hat{\mathbf{Y}}\|_F^2 \\ L_{BUAng} &= \frac{1}{n}\sum_{k=1}^{n} \frac{180}{\pi} \cos^{-1}\left(\frac{\mathbf{y}_k^T \hat{\mathbf{y}}_k}{\|\mathbf{y}_k\|_2 \|\hat{\mathbf{y}}_k\|_2}\right) \end{aligned} \tag{4.3.7}$$

and α_5 and α_6 are the loss weights. $\mathbf{y}_k, \hat{\mathbf{y}}_k$ are the k^{th} pixel of HSI observation $\mathbf{Y}$ and HSI reconstruction $\hat{\mathbf{Y}}$, respectively. The inclusion of this additional BU loss is crucial as, without it, EDIP and ADIP would produce endmember and abundance estimates that closely align with the guidance $\mathbf{E}_G$ and $\mathbf{A}_G$.

The final loss function is a combination of the above losses, as follows:

$$L(\boldsymbol{\theta}_E, \boldsymbol{\theta}_A) = L_{EDIP} + L_{ADIP} + L_{BU} \tag{4.3.8}$$

where, $L_{EDIP}, L_{ADIP}, L_{BU}$ are defined in Eq. (4.3.2), Eq. (4.3.4) and Eq. (4.3.6). It is worth mentioning that the final loss function deviates from the one described in the publication [95] by incorporating an additional trigonometric discrepancy measurement. From the perspective of unmixing, the terms L_{EDIP} and L_{ADIP} in the composite loss function ensure that the network produces meaningful endmembers and abundances, in the sense that $\hat{\mathbf{E}}, \hat{\mathbf{A}}$ cannot deviate too much from $\mathbf{E}_G, \mathbf{A}_G$. At the same time, L_{BU} allows the network to have the freedom to search for better estimations than the guidance $\mathbf{E}_G, \mathbf{A}_G$. From the perspective of network training, L_{EDIP} and L_{ADIP} can be interpreted as regularizations on the outputs of the BUDDIP network, $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$, with L_{BU} serving as the data fidelity term. This composite loss function can therefore alleviate the need for regularisation techniques in other DIP methods such as early stopping [37] and exponentially averaging over different runs [36]. By properly choosing the hyperparameters $\alpha_{1\sim6}$, the network outputs will eventually reach an equilibrium state between inducing small fitting error (L_{BU}), and regularisation penalty, (L_{EDIP} and L_{ADIP}).

In this work, the proposed loss function in Eq. (4.3.8) is utilized in both linear and nonlinear blind unmixing scenarios. Although in nonlinear unmixing, the form of L_{EDIP} and L_{ADIP} could potentially be changed to accommodate nonlinear reconstruction, for the sake

of simplicity and consistency, the same linear form is adopted for both scenarios. However, the purpose of L_{EDIP} and L_{ADIP} is to ensure that the network produces meaningful output, which is still valid under nonlinear conditions without the need for modification. This is because the nonlinear model Eq. (2.2.4) still contains a linear component. For the same reason, the input $\mathbf{E}_I$ and $\mathbf{A}_I$ and training guidance $\mathbf{E}_G$ and $\mathbf{A}_G$ are generated via the same unmixing algorithms, SiVM [1] and FCLS, for both linear and nonlinear cases.

In contrast to the two-stage-cyclic descent method [91], the proposed network is trained in an end-to-end manner using the gradient descent optimizer, ADAM [85]. Given only the HSI image $\mathbf{Y}$, the learnable parameters $\boldsymbol{\theta}_E, \boldsymbol{\theta}_A$ are learned by minimizing the composite loss in Eq. (4.3.8).

Adaptive loss weight strategy for NL-BUDDIP

We also propose an adaptive loss weight strategy to enhance the performance of nonlinear unmixing. The conventional approach of setting fixed weights for all the loss terms, i.e., L_{EDIP}, L_{ADIP}, and L_{BU}, can lead to suboptimal results. If $\alpha_{1\sim4}$ are given greater weight, the network might simply follow the guidance $\mathbf{E}_G$ and $\mathbf{A}_G$, while if $\alpha_{5\sim6}$ is given more weight, the network may output meaningless endmembers and abundances. To overcome this challenge, our proposed strategy adjusts the weights adaptively throughout the training process. Initially, the network is given more weight to converge quickly to the guidance, with $\alpha_{1\sim4}$ being larger. Once the network has approached the guidance, the weights are adjusted to give the network more freedom to search for a better solution by reducing $\alpha_{1\sim4}$ and increasing $\alpha_{5\sim6}$. To ensure stability, a threshold is set for the loss weights. The steps involved in this strategy are outlined in Algorithm 1.

Algorithm 1: Adaptive Loss Weight Strategy for enhanced loss function Eq. (4.3.8).

Input:

- $\alpha_{1\sim6}^{init}$ - the initial value of loss weights
- γ_1, γ_2 - the rate of updating loss weights
- $\alpha_{min}, \alpha_{max}$ - the boundary of loss weights
- g - update gap
- K - the number of training epochs.

Train:

Initialise $\alpha_{1\sim6} \leftarrow \alpha_{1\sim6}^{init}$;
for $k \leftarrow 1$ **to** K **do**
    train the proposed network with $\alpha_{1\sim6}$;
    if $k \bmod g == 0$ **then**
        $[\alpha_{1\sim4}, \alpha_{5\sim6}] \leftarrow [\alpha_{1\sim4} * \gamma_1, \alpha_{5\sim6}/\gamma_2]$;
        $\alpha_{1\sim6} \leftarrow$ Clip $(\alpha_{1\sim6}, \alpha_{min}, \alpha_{max})$;

4.4 Experiment

To evaluate the effectiveness of our proposed approach, BUDDIP, we will compare it with several state-of-the-art unmixing methods. We will use the same performance metrics as discussed in Sec. 3.5.1, namely, RMSE and AAD for abundance estimation and SAD for endmember estimation.

4.4.1 Data

We evaluate the performance of different unmixing algorithms using various datasets.

Synthetic Dataset 1

To evaluate the effectiveness of both linear and nonlinear unmixing algorithms, we utilized the HSI dataset synthesis approach proposed in [18]. To adapt the procedure for assessing the performance of nonlinear unmixing algorithms, we modified the linear mixing process to its nonlinear counterpart. The procedure consists of the following steps:

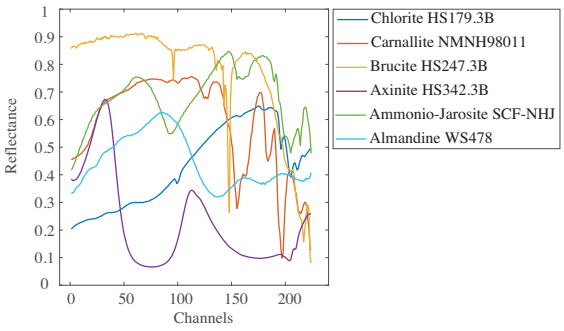


Figure 4.4: Endmember signatures for synthetic data.

- *Endmember generation.* We selected six spectral signatures from the USGS spectral library (splib06) [81], which provides spectral reflectance values for various minerals across 224 channels. These signatures formed a 224×6 endmember matrix, as illustrated in Fig. 4.4.

- *Abundance generation.* We divided a synthetic HSI of size $a^2 \times a^2$ pixels into a^2 non-overlapping patches of size $a \times a$ pixels. For each patch, we randomly assigned two endmembers and their corresponding abundances γ and $1-\gamma$, while the other four endmembers were set to zero. We then convolved the abundance map with a Gaussian filter of size $(a+1) \times (a+1)$ and variance 2, and scaled the abundance values to satisfy the abundance sum-to-one constraint (ASC). Throughout this experiment, we set $a = 10$ and $\gamma = 0.8$.

- *Mixing process.* We synthesized HSI data for both linear and nonlinear mixing models, by applying either the linear model in Eq. (2.2.2) or the nonlinear model in Eq. (2.2.4) to the generated endmembers and abundances.

- *Noise contamination.* We added additive white Gaussian noise (AWGN) to the synthetic HSI data. The signal-to-noise ratio (SNR) was defined as $SNR = 10\log_{10}\left(E[\mathbf{x}^T\mathbf{x}]/E[\mathbf{n}^T\mathbf{n}]\right)$, where $\mathbf{x}$ represents the original, noise-free HSI data, and $\mathbf{n}$ is the added

noise.

Synthetic Dataset 2

In order to evaluate the effectiveness of HSI unmixing methods in the absence of pure pixels in the dataset, we adopted the synthetic procedure described in [96–99]. In this procedure, a purity measure for an observed pixel $\mathbf{y}_k$ is defined as $\rho_k = \|\mathbf{a}_k\|_2 \in [1/\sqrt{r}, 1]$, where $\mathbf{a}_k \in \mathbb{R}^{r\times 1}$ represents the corresponding true abundance for $\mathbf{y}_k$, and the range $[1/\sqrt{r}, 1]$ is a result of the ANC and ASC constraints. A higher value of ρ_k indicates greater purity of the pixel $\mathbf{y}_k$. The purity level of a dataset with n observed pixels $\{\mathbf{y}_k\}_{k=1}^n$ is denoted by ρ, where $\rho - 0.1 \leq \rho_k \leq \rho, \forall k \in [1, n]$. The synthetic generation procedure is similar to that of synthetic dataset 1, with the exception of the abundance generation process, which is outlined as follows:

- Randomly sampling $K = 10n$ abundance vectors from a Dirichlet distribution $D(\boldsymbol{\mu})$ where $\boldsymbol{\mu} = (1/r)\mathbf{1}_r$, that is,

$$\Omega = \{\mathbf{a}_k | \mathbf{a}_k \sim D(\boldsymbol{\mu}), \forall k = 1, ..., K\} \tag{4.4.1}$$

  and calculate the corresponding purity ρ_k for all k.

- Construct a set of n abundance vectors with purity level ρ by randomly choosing n samples from Ω subject to $\rho_k \in [\rho - 0.1, \rho]$.

4.4.2 Effectiveness of BUDDIP

We demonstrate the effectiveness of our proposed BUDDIP unmixing framework using the synthetic HSI dataset 1 with dimensions 100x100 pixels that are contaminated with AWGN noise to achieve an SNR of 30 dB. The structure of BUDDIP is summarized in Table 4.2. To evaluate the performance of L-BUDDIP in linear unmixing, we use the Adam optimizer with a learning rate of $5e-3$ and train for 6000 epochs. We

compare the results of L-BUDDIP with state-of-the-art learning-based methods such as UnDIP [36] and EGU-Net-ss [4] that are trained with guidance, using the synthetic dataset 1 under the linear mixing model. We also include the traditional method SiVM [1] + FCLS [2] for reference as it is used to generate the guidance. In the case of nonlinear unmixing, we evaluate the performance of NL-BUDDIP using the synthetic dataset 1 under the FM model. The Mixing Module in the proposed network uses the FM model defined in Eq. (2.2.6) and Eq. (2.2.7), while the structure of BUDDIP remains the same as outlined in Table 4.2. By default, we train NL-BUDDIP for 12000 epochs, as the nonlinear unmixing case is more challenging than the linear case.

Table 4.2: Hyperparameters of BUDDIP structure.

EDIP					
	In Channel	Out Channel	kernel size	stride	pad
Conv1D	p	256	3	1	same
	256	p	3	1	same
	p	p	1	1	same
ADIP					
	In Channel	Out Channel	kernel size	stride	pad
Conv2D	r	32	3	1	same
	32	64	3	1	same
	64	64	3	1	same
	64	r	3	1	same
	$2r$	r	1	1	same
Activation		LeakyReLU		negative_slope=0.1	

Linear Unmixing

- **Hyperparameter Study:** In this study, we explore the effect of the hyperparameters $\alpha_{1\sim6}$ in the proposed composite loss function Eq. (4.3.8) by varying them within $\{0.0, 0.001, 0.01, 0.1, 1.0\}$. It is worth noting that when $\alpha_2 = \alpha_4 = \alpha_6 = 0$, the loss function Eq. (4.3.8) deactivates all the additional angle loss terms. The unmixing performance in terms of abundance AAD and endmember

SAD are presented in Fig. 4.5. The figure illustrates that deactivating L_{EDIP} and L_{ADIP} by setting $\alpha_{1\sim4} = 0.0$ leads to meaningless unmixing results. On the other hand, when $\alpha_5 = \alpha_6 = 0.0$ and any of $\alpha_{1\sim4}$ is set to 1.0, the network outputs unmixing results that are similar to those obtained by SiVM+FCLS guidance.

Additionally, when deactivating L_{EAng}, L_{AAng}, L_{BUAng} and activating L_{EMSE}, L_{AMSE}, L_{BUMSE} with $\alpha_2 = \alpha_4 = \alpha_6 = 0$ and $\alpha_1 = 0.1, \alpha_3 = 0.01, \alpha_5 = 1$, the proposed method achieves improved unmixing performance with an AAD of 5.67 and SAD of 1.88. Moreover, the composite loss function Eq. (4.3.8) with all six loss terms activated via $\alpha_2 = 0.001$, $\alpha_4 = 0.01$, $\alpha_6 = 0.1$, and $\alpha_1 = \alpha_3 = \alpha_5 = 1$ yields the best unmixing performance with an AAD of 4.65 and SAD of 1.68.

Henceforth, we adopt the default settings of the composite loss function Eq. (4.3.8), where $\alpha_2 = 0.001$, $\alpha_4 = 0.01$, $\alpha_6 = 0.1$, and $\alpha_1 = \alpha_3 = \alpha_5 = 1,$ for the subsequent experiments. This experiment also suggests that the proposed method is highly sensitive to hyperparameters $\alpha_{1\sim6}$ as they determine the relative importance of each loss term.

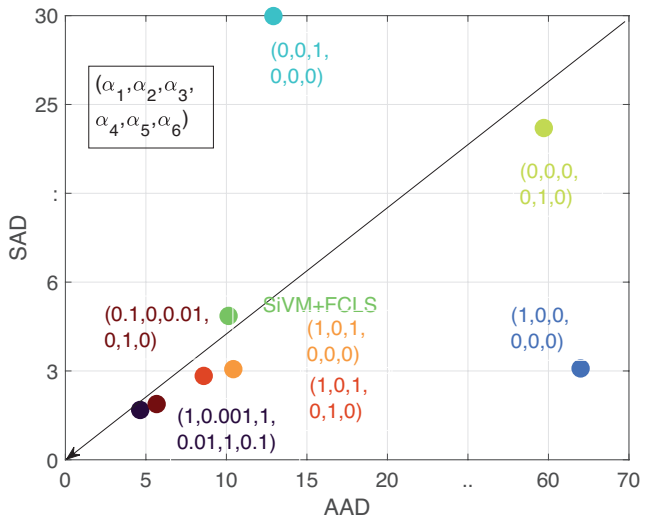
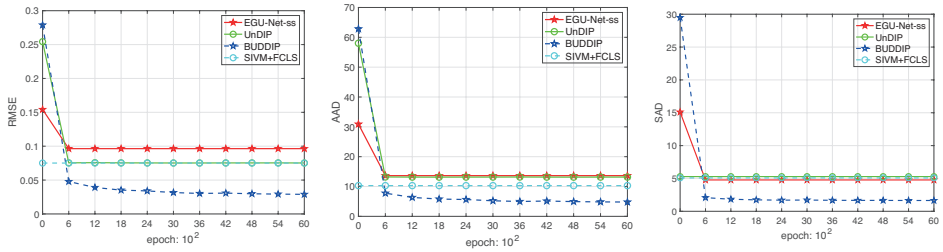


Figure 4.5: The impact of hyperparameter $\alpha_{1\sim6}$ on linear unmixing performance.

- **Performance Limitation of Guidance:** We now assess the impact of the guidance on various guidance-based unmixing networks. Our proposed methods use the default training settings. On the other hand, competitors such as EGU-Net [4] and UnDIP [36] utilize different traditional unmixing methods, VCA+FCLS and SiVM, respectively, to generate their training guidance. To ensure a fair comparison, we use the training guidance generated by SiVM+FCLS for all methods in this study. The results of unmixing performance by different methods are illustrated in Fig. 4.6. It can be observed that the proposed methods outperform the guidance SiVM+FCLS, while the performance of the competitors is limited by the guidance. This is because the proposed methods are capable of finding better unmixing solutions while the competitors simply aim to imitate the guidance.

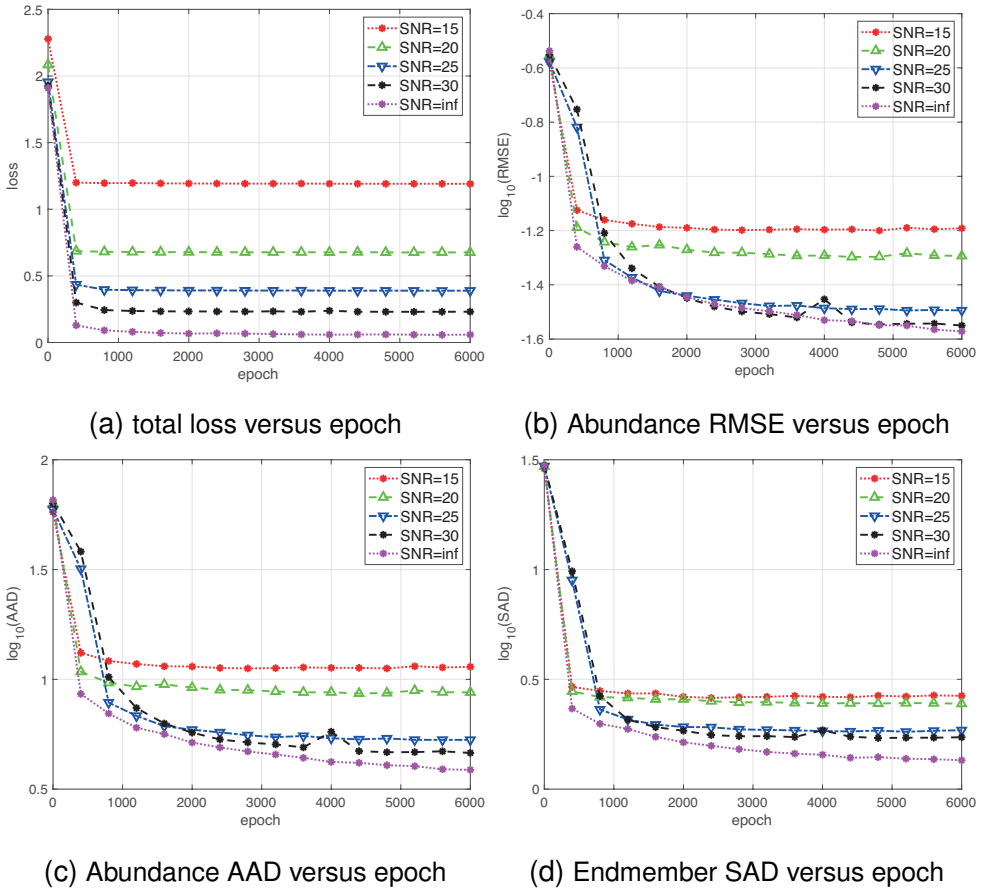


(a) Abundance RMSE versus epoch (b) Abundance AAD versus epoch (c) Endmember SAD versus epoch

Figure 4.6: Linear unmixing performance of various guidance-based networks. (a) Abundance RMSE versus epoch. (b) Abundance AAD versus epoch. (c) Endmember SAD versus epoch.

- **Impact of Noise:** We aim to assess the impact of noise on the network training process by varying the SNR in the range of $[15, 20, 25, 30, inf]$ dB while maintaining the default settings. To this end, we monitor the progress of the training process, including the total loss, abundance RMSE, abundance AAD, and endmember SAD versus epochs, as shown in Fig. 4.7. It is evident that the loss and various metrics decrease as the SNR increases. Moreover, the network gradually requires slightly more epochs to

achieve convergence as the SNR increases. This observation aligns with the findings presented in [37], where the network requires more epochs to tackle image reconstruction tasks in low-noise conditions. To strike a balance between training efficiency and unmixing quality, we set the epoch number to 6000.



(a) total loss versus epoch

(b) Abundance RMSE versus epoch

(c) Abundance AAD versus epoch

(d) Endmember SAD versus epoch

Figure 4.7: Impact of noise on network training for linear unmixing. (a) total loss versus epoch. (b) Abundance RMSE versus epoch. (c) Abundance AAD versus epoch. (d) Endmember SAD versus epoch.

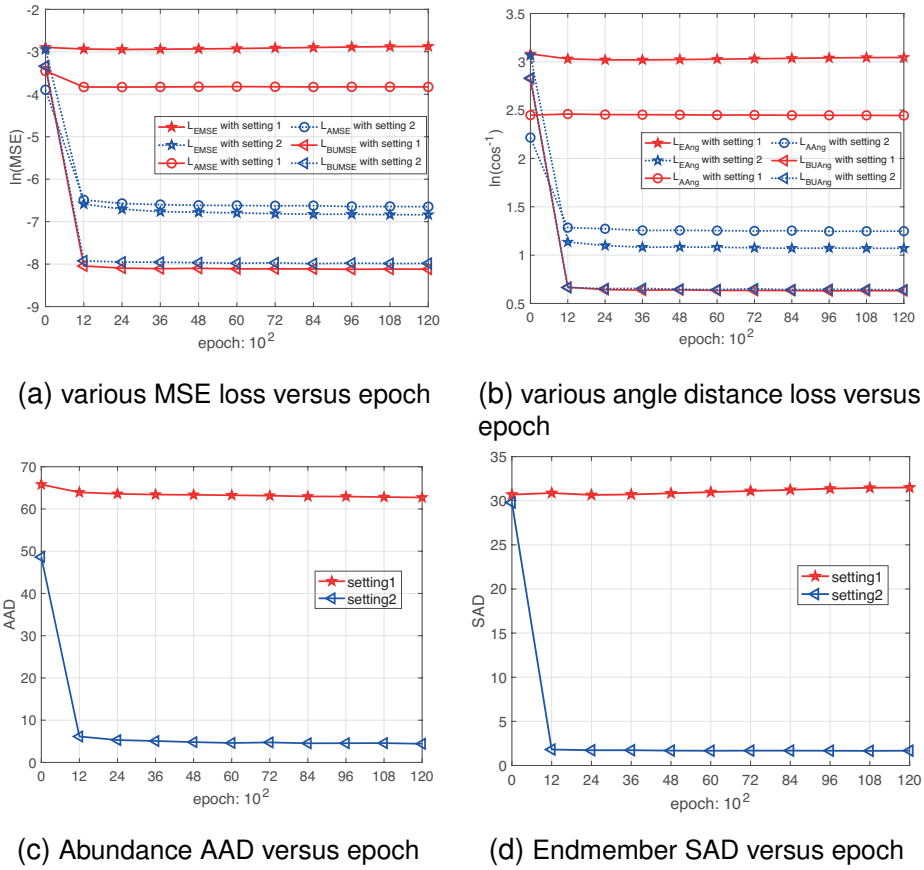
- **Convergence Analysis:** In this section, we aim to analyze the convergence of the proposed method by comparing the training process under two different settings. In Setting 1, we train the network without guidance by deactivating L_{EDIP} and L_{ADIP} using $\alpha_{1\sim4} = 0, \alpha_5 = 1$, and $\alpha_6 = 0.1$. In Setting 2, we activate the guidance terms by using $\alpha_1 = \alpha_3 = \alpha_5 = 1.0, \alpha_2 = 0.001, \alpha_4 = 0.01, \alpha_6 = 0.1$. The rest of the settings remain the same as the default settings except for the number of epochs, which we set

to 12000 for analysis purposes. We evaluate the training process and corresponding unmixing performance as a function of the number of epochs, which we present in Fig. 4.8.

As shown in Fig. 4.8, Setting 1 results in the smallest fitting error L_{BUMSE} and converges at around 1200 epochs. However, it also delivers the worst unmixing performance with an AAD of 62 (see Fig. 4.8c) and a SAD of 31 (see Fig. 4.8d). Moreover, the endmember SAD values exhibit an increasing trend with more epochs, suggesting the occurrence of overfitting. Correspondingly, the loss/regularisation terms $L_{EMSE}, L_{AMSE}, L_{EAng}, L_{AAng}$ in Fig. 4.8a and Fig. 4.8b are large.

On the other hand, in Setting 2, where all the loss terms are activated, the network reaches an equilibrium state at around epoch 6000, where it minimizes all six loss terms. This setting also results in improved unmixing performance, with an abundance AAD of approximately 4.4 and a SAD of 1.67. Furthermore, the proposed method's unmixing performance, measured by AAD and SAD, remains stable even with additional training epochs, as the L_{EDIP} and L_{ADIP} terms serve as effective regularizations. Hence, the proposed approach eliminates the need for conventional techniques such as early stopping [37] or exponentially weighted averaging of outputs from multiple runs [36], which are commonly used in DIP-based methods. A trade-off between training efficiency and unmixing quality leads us to choose 6000 epochs for training.

- **Comparison of Different Input:** In this section, we assess the efficacy of the proposed meaningful input strategy. We maintain the default BUDDIP settings, except for the input type, where we compare the network's performance using two input types: Gaussian input as suggested by UnDIP [36], and the proposed



(a) various MSE loss versus epoch

(b) various angle distance loss versus epoch

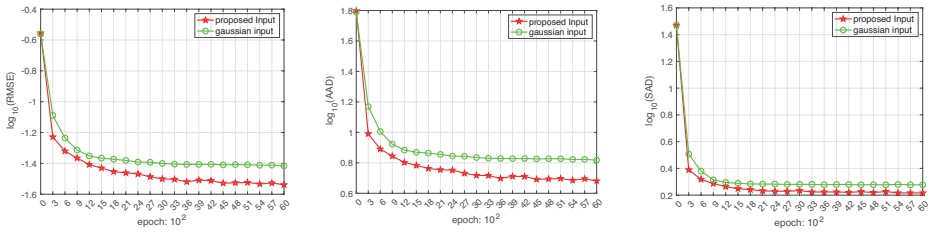
(c) Abundance AAD versus epoch

(d) Endmember SAD versus epoch

Figure 4.8: Convergence analysis for linear unmixing case. In setting 1, we train the network with $\alpha_5 = 1, \alpha_6 = 0.1, \alpha_{1\sim4} = 0$, that is, the guidance L_{EDIP} and L_{ADIP} are deactivated. In setting 2, we activate the guidance L_{EDIP} and L_{ADIP} by using $\alpha_1 = \alpha_3 = \alpha_5 = 1.0, \alpha_2 = 0.001, \alpha_4 = 0.01, \alpha_6 = 0.1$. (a) various MSE loss versus epochs, (b) various angle distance loss versus epochs, (c) Abundance AAD versus epochs, (d) Endmember SAD versus epochs.

input. The unmixing performance of the network as a function of training epochs is depicted in Fig. 4.9. It can be observed that with the proposed input strategy, the network achieves better unmixing performance when the number of epochs is small, i.e., 300. This improvement can be attributed to the proposed input, which can be viewed as a noisy observation of the underlying ground truth. Therefore, the network's task is to generate more refined unmixing results given the noisy input. In comparison, when using Gaussian input, the network needs to generate unmixing results given non-informative input. Additionally, it is worth noting that BUDDIP with the proposed input delivers better

unmixing results with the increase of training epochs.



(a) Abundance RMSE versus epoch (b) Abundance AAD versus epoch (c) Endmember SAD versus epoch

Figure 4.9: Linear unmixing performance of BUDDIP with different input strategies: gaussian input and the proposed input. (a) Abundance RMSE versus epoch. (b) Abundance AAD versus epoch. (c) Endmember SAD versus epoch.

Nonlinear Unmixing

- **Fixed vs. Adaptive Loss Weight Strategy:** We now evaluate the impact of the proposed adaptive loss weight strategy in Algorithm 1 under the nonlinear unmixing case. Specifically, we vary the hyperparameters $\alpha_2^{init}, \alpha_4^{init}, \gamma_1, \gamma_2$, while the remaining hyperparameters in Algorithm 1 are set to $\alpha_1^{init} = 100, \alpha_3^{init} = 10, \alpha_5^{init} = 1, \alpha_6^{init} = 0.1, \alpha_{min} = 1e-3, \alpha_{max} = 1e+2, g = 300$ by grid search techniques. The unmixing performance is reported in Fig. 4.10. The data points where $\gamma_1 = \gamma_2 = 1$ represent the outcomes of the fixed loss weight strategy, while the remaining data points correspond to the adaptive loss weight strategy. It is evident that utilizing the fixed strategy enables the network to generate unmixing results that surpass the guidance. However, the proposed adaptive loss weight strategy yields better unmixing performance in terms of both endmember SAD and abundance AAD when $\alpha_2^{init} = 1, \alpha_4^{init} = 10, \gamma_1 = 0.8, \gamma_2 = 0.9$. Consequently, we will utilize these hyperparameter settings as our default configuration in the subsequent experiments.

- **Impact of Noise:** In this study, we assess the impact of noise on

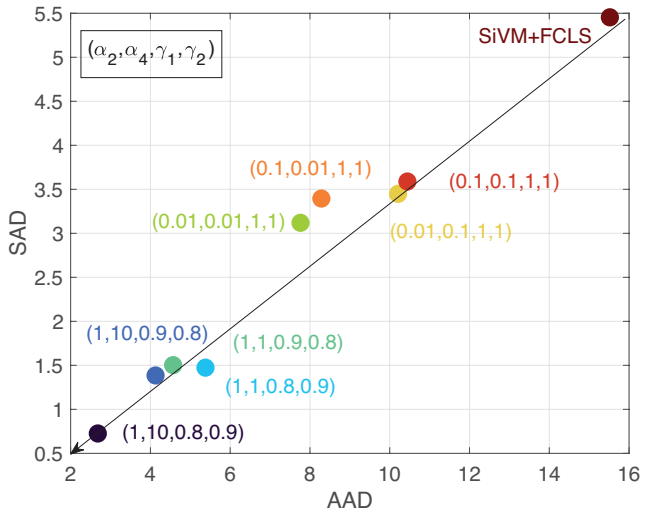
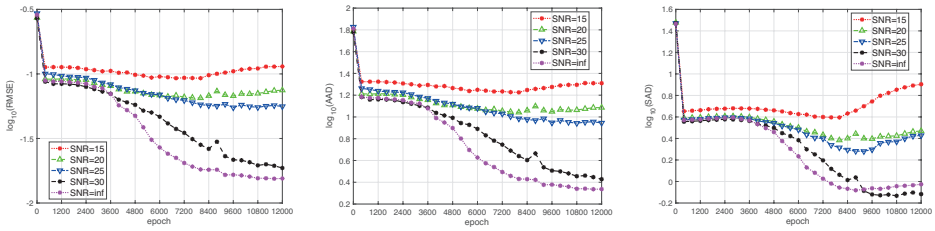


Figure 4.10: Fixed versus adaptive loss weight strategy under the nonlinear unmixing case. The X axis is abundance AAD and Y axis is endmember SAD. The dots with $\gamma_1 = \gamma_2 = 1$ are the fixed loss weight strategy, while the remaining are adaptive loss weight strategy.

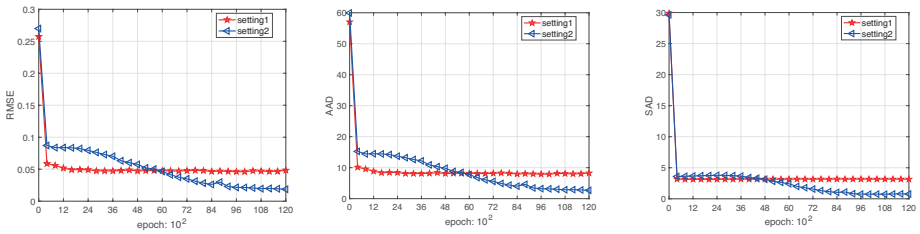
the nonlinear unmixing network training process by altering SNR in the range of $[15, 20, 25, 30, inf]$ dB, while retaining the default settings. We then monitor the training progress, including the abundance RMSE, abundance AAD, and endmember SAD over epochs, as illustrated in Fig. 4.11. The results indicate that the network necessitates more epochs to achieve convergence as the SNR increases, similar to the linear unmixing case. However, for low SNR scenarios such as 15 dB, the network may overfit the data after approximately 8000 epochs. It should be highlighted that the hyperparameters employed in this experiment are optimized for an SNR of 30 dB. However, by fine-tuning the hyperparameters correspondingly, it is possible to mitigate the issue of overfitting and improve the performance for varying SNR levels. Ultimately, for SNRs larger than 30 dB, the network can still enhance the unmixing quality even after 8000 epochs. These experiments illustrate that while the adaptive loss weight strategy can enhance performance, improper setting of hyperparameters can result in overfitting.



(a) Abundance RMSE versus epoch (b) Abundance AAD versus epoch (c) Endmember SAD versus epoch

Figure 4.11: Impact of noise on network training for nonlinear unmixing. (a) Abundance RMSE versus epoch. (b) Abundance AAD versus epoch. (c) Endmember SAD versus epoch.

- **Convergence Analysis:** In this section, we analyze the convergence of the proposed method for nonlinear unmixing by comparing the training process under two different settings. In Setting 1, we train the network utilizing the fixed loss weight strategy with $\alpha_1 = 100$, $\alpha_3 = 10$, $\alpha_5 = \gamma_1 = \gamma_2 = 1.0$, $\alpha_6 = 0.1$, and $\alpha_2 = \alpha_4 = 0.01$. In Setting 2, we train the network employing the adaptive loss weight strategy with $\alpha_1 = 100$, $\alpha_3 = \alpha_4 = 10$, $\alpha_2 = \alpha_5 = 1.0$, $\alpha_6 = 0.1$, $\gamma_1 = 0.8$, and $\gamma_2 = 0.9$. The remaining settings remain unaltered, adhering to the default configuration. We evaluate the corresponding unmixing performance as a function of the number of epochs, which we present in Fig. 4.12. As demonstrated in Fig.4.10, the SiVM+FCLS guidance method achieved an abundance AAD of 15.5 and an endmember SAD of 5.45. In contrast, the fixed strategy of NL-BUDDIP obtained superior unmixing performance, with an abundance AAD of 7.76 and an endmember SAD of 3.12, reaching faster convergence around epoch 2400. Nevertheless, by implementing the adaptive strategy in Setting 2, we further improved the unmixing performance, achieving an AAD of 2.6810 and a SAD of 0.7257, despite a slower convergence rate at around 12000 epochs.



(a) Abundance RMSE versus epoch (b) Abundance AAD versus epoch (c) Endmember SAD versus epoch

Figure 4.12: Convergence analysis for the nonlinear unmixing case. In setting 1, we train the network with the fixed loss weight strategy. In setting 2, we use the adaptive loss weight strategy. (a) Abundance RMSE versus epochs, (b) Abundance AAD versus epochs, (c) Endmember SAD versus epochs.

4.4.3 Comparing with state-of-the-arts methods

In this section, we compare our proposed methods BUDDIP against various state-of-the-art unmixing techniques, including network-based methods UnDIP [36], EGU-Net-ss [4], MNN-BU-2 [18], CNNAEU [100], and MiSiCNet [101], as well as traditional methods SiVM [1]+FCLS [2], rNMF [13], HyperCSI [96], HiSun [62], and EDAA [52]. We also compare our method with UBUNet-II, which was proposed in previous work. Unless stated otherwise, we conduct experiments under the default setting, where the structure of our proposed network is summarized in Table 4.2. SiVM [1]+FCLS [2] is used to generate the training guide for those methods that need guidance or initialisation.

For the linear unmixing case, we train the proposed L-BUDDIP network using the Adam optimizer to minimize the loss function Eq. (4.3.8) with $\alpha_2 = 0.001$, $\alpha_4 = 0.01$, $\alpha_6 = 0.1$, and $\alpha_1 = \alpha_3 = \alpha_5 = 1$. We set the learning rate to $5e-3$ and the number of epochs to 6000. The synthetic HSI image dataset for both synthetic datasets 1&2 is constructed under the linear mixing model, consisting of 100×100 pixels. We also contaminate these HSI reflectances with AWGN, resulting in $SNR = 30$ dB.

For the nonlinear unmixing case, we train the proposed NL-BUDDIP network using the Adam optimizer to minimize the loss function Eq.

(4.3.8) using adaptive loss Algorithm 1 with a learning rate set to $5e-3$ and the number of epochs set to 12000. The hyperparameters for Algorithm 1 are set to $\alpha_1^{init} = 100$, $\alpha_2^{init} = \alpha_5^{init} = 1$, $\alpha_3^{init} = \alpha_4^{init} = 10$, $\alpha_6^{init} = 0.1$, $\gamma_1 = 0.8$, $\gamma_2 = 0.9$, $\alpha_{min} = 1e-3$, $\alpha_{max} = 1e+2$, and $g = 300$. The synthetic HSI image dataset for both synthetic datasets 1&2 is constructed under the FM model, consisting of 100×100 pixels. We also contaminate these HSI reflectances with AWGN, resulting in $SNR = 30$ dB. In our proposed network, the Mixing Module (MM) correspondingly uses the FM model defined in Eq. (2.2.6) and Eq. (2.2.7).

Performance vs. HSI data purity

In this section, we explore the impact of varying purity level ρ on the unmixing performance of various methods using synthetic dataset 2. To this end, similar to [96], we set the purity ρ to vary in the range of $[0.8, 0.9, 1.0]$, where $\rho = 0.8$ corresponds to a highly mixed HSI data scenario and $\rho = 1.0$ represents highly pure pixels. Furthermore, we evaluate the proposed L-BUDDIP method with guidance generated from two methods, SiVM+FCLS and HyperCSI, respectively. Table 4.3 and Table 4.4 present the results of both linear and nonlinear unmixing, respectively.

Regarding linear unmixing, the results indicate that HyperCSI outperforms other competing methods when the data is highly mixed, i.e., $\rho = 0.8$ and $\rho = 0.9$. However, when the proposed method is trained with guidance from HyperCSI, it significantly improves the performance by almost 1.6 times better than HyperCSI. Specifically, at $\rho = 0.8$, the proposed method improves the abundance AAD from 2.19 to 1.97 and the endmember SAD from 0.80 to 0.49 compared to HyperCSI. Additionally, although SiVM+FCLS fails to deliver satisfactory unmixing performance, the proposed method can utilize the unmixing results from SiVM+FCLS as training guidance and improve the per-

formance by three times, from an abundance AAD of 11.4 to 3.82. At a purity level of $\rho = 1.0$, the proposed BUDDIP guided with SiVM achieves the lowest RMSE and AAD, as well as the second-best SAD. These results suggest that the quality of the guidance used during training significantly impacts the proposed method's performance, with higher quality guidance leading to better performance.

For nonlinear unmixing, the results indicate that at the purity level of $\rho = 0.8$, HyperCSI outperforms other competing methods. However, although SiVM+FCLS only achieves moderate performance, the proposed NL-BUDDIP can utilize those unmixing results as guidance and deliver seven times better performance with an AAD of 3.9959 and SAD of 0.8752, even better than HyperCSI with an AAD of 4.6 and SAD of 1.5. When ρ increases to 0.9 and 1.0, the proposed NL-BUDDIP achieves the best performance among all competitors.

Overall, the proposed BUDDIP demonstrates excellent performance across different purity levels for both linear and nonlinear unmixing cases. Moreover, the proposed method can utilize the unmixing results of state-of-the-art methods as training guidance ($\mathbf{E}_G$ and $\mathbf{A}_G$) to further improve its performance.

Processing Time Comparison

In this section, we present a comparison of the processing time of different unmixing methods. The experiments were conducted on a test platform comprising Intel Xeon Gold 6248 CPU 2.50GHz, Tesla V100 GPU, and 503GB RAM. The average processing times for the linear unmixing case and the nonlinear unmixing case are reported in Table 4.3 and Table 4.4, respectively. The results are obtained by averaging over ten independent runs. Notably, the network-based unmixing methods generally outperform traditional unmixing methods in terms of processing time, except for HyperCSI. Among the network-based

Table 4.3: Linear Unmixing Performance Comparison of Various Methods for Different Purity Level ρ. The Best Results are highlighted in Bold. The second best are denoted with a star.

method	$\rho = 0.8$			$\rho = 0.9$			$\rho = 1.0$			time (:s)
	RMSE	AAD	SAD	RMSE	AAD	SAD	RMSE	AAD	SAD	
SiVM+FCLS	0.0692	11.4335	5.0056	0.0328	4.4675	2.4792	0.0106	1.1287⋆	0.4726	11.9374
HyperCSI	0.0140⋆	2.1875⋆	0.7998⋆	0.0120⋆	1.6817⋆	0.5989⋆	0.0114	1.4366	0.5826	0.1181
HiSun	0.0439	5.6432	4.7341	0.0237	2.7296	2.0948	0.0251	2.3893	0.8225	8.7195
EDAA	0.0633	10.4134	3.5987	0.0701	10.3568	2.0083	0.0316	3.3124	**0.2614**	12.2705
MNN-BU2	0.0614	10.2151	3.3293	0.0831	12.6412	3.2022	0.0256	2.3661	1.1599	0.1926
UBUNet-II	0.0348	5.7705	1.7182	0.0461	6.7399	1.5596	0.0227	2.8388	0.6132	0.1490
EGU-Net-ss	0.0959	13.6120	4.6811	0.0538	6.0256	2.7246	0.0177	1.9359	0.7267	0.3659
UnDIP	0.0720	11.9410	5.0056	0.0345	4.7341	2.4792	0.0139	1.6776	0.4726	0.0086
MiSiCNet	0.0953	14.3151	6.0990	0.0657	8.0372	4.6247	0.0292	3.3278	2.8937	0.0023⋆
CNNAEU	0.3667	65.1646	16.4183	0.4160	69.2101	16.8180	0.4234	70.9834	14.7135	0.0054
L-BUDDIP (SiVM+FCLS)	0.0236	3.8200	1.5430	0.0161	2.2130	0.9460	**0.0104**	**1.0941**	0.2920⋆	**0.0018**
L-BUDDIP (HyperCSI)	**0.0126**	**1.9681**	**0.4875**	**0.0119**	**1.5704**	**0.4587**	0.0105⋆	1.1408	0.4383	

Table 4.4: Nonlinear Unmixing Performance Comparison of Various Methods for Different Purity Level ρ. The Best Results are in Bold. The second best are denoted with a star.

method	$\rho = 0.8$			$\rho = 0.9$			$\rho = 1.0$			time (:s)
	RMSE	AAD	SAD	RMSE	AAD	SAD	RMSE	AAD	SAD	
rNMF	0.3042	50.732	23.3305	0.3353	52.6879	22.4824	0.3665	54.2828	22.4233	3.1484
SiVM+FCLS	0.1002	17.9685	6.3412	0.0465	6.7308	3.4658	0.0192	2.4020	0.6095	10.9354
HyperCSI	0.0302⋆	4.6126⋆	1.5008⋆	0.0237⋆	3.2769	1.1230⋆	0.0176	2.1633	0.7855	0.1551
HiSun	0.0405	5.6666	5.4575	0.0212	2.5992⋆	2.6349	0.0188	2.0754	0.6695	9.5697
EDAA	0.0698	11.7178	3.8387	0.0406	5.4092	1.6947	0.0390	4.3166	**0.2894**	12.3635
MNN-BU2	0.1451	29.0092	12.0890	0.0736	9.6026	5.5778	0.1391	22.4132	3.8198	0.1872
UBUNet-II	0.0543	9.1804	2.6652	0.0550	7.9753	1.7887	0.0312	3.8843	0.9381	0.2729
EGU-Net-ss	0.1022	15.3104	4.4069	0.0499	5.9326	2.4708	0.0164⋆	1.8787⋆	0.625	0.3789
UnDIP	0.1496	28.7185	7.1064	0.0479	6.9371	3.4658	0.0238	3.1231	0.6095	0.0099
MiSiCNet	0.0988	15.7423	5.7003	0.0658	8.5409	4.4252	0.0312	3.6481	2.8217	**0.0026**
CNNAEU	0.3600	66.6197	16.2333	0.3972	69.6882	16.9117	0.4428	72.7636	16.5118	0.0077
NL-BUDDIP	**0.026**	**3.9959**	**0.8752**	**0.0182**	**2.2613**	**0.5897**	**0.0121**	**1.1652**	0.4202⋆	0.0039⋆

methods, MiSiCNet stands out as the fastest nonlinear unmixing approach, while our proposed methods demonstrate the fastest processing time to state-of-the-art methods in the linear unmixing case.

Network Complexity Comparison

In this section, we analyze the complexity of various unmixing networks by comparing the number of learnable parameters, which are summarized in Table 4.5. It is evident that the proposed BUDDIP network has fewer learnable parameters than other guidance-based unmixing networks, such as UnDIP and EGU-Net-ss. Although unfolding-based networks, such as MNN-BU and UBUNet, have fewer learnable parameters, the proposed method delivers better unmixing performance. The reduction in network complexity primarily arises from the fact that, with the proposed input strategy, we can design a more efficient network structure.

Table 4.5: Number of learnable parameters

method	CNNAEU	UnDIP	BUDDIP	MiSiCNet
♯	3.2×10^5	1.3×10^6	4.6×10^5	1.7×10^6
method	MNN -BU-2	UBUNet -II	EGU-Net -ss	-
♯	5.4×10^3	5.4×10^3	3.56×10^6	-

Significance Test

In this experiment, we conducted significance tests to assess the performance of the proposed BUDDIP method and compare it with various other methods, including the proposed UBUNet, in terms of abundance estimation mean squared error (MSE) using the F-test and adjusted R-squared metrics with synthetic dataset 1. The F-test between competitors and BUDDIP was employed to determine if the complex model BUDDIP significantly outperformed the simpler com-

petitor model, with a larger F statistic and smaller p-value indicating a notable improvement in data fit. A larger adjusted R-squared indicated a better fit of the model to the data.

To ensure fairness in the comparison, we employed existing methods SiVM+FCLS, for generating initialization in the unmixing networks MNN-BU, UBUNet, MiSiCNet, UnDIP and EGU-Net. The results are summarized in Table 4.6. It is noteworthy that MiSiCNet and UnDIP lacked F-statistic and p-values due to having more free parameters than the proposed BUDDIP, yet the proposed BUDDIP demonstrated superior performance. Conversely, EGU-Net-ss lacked adjusted R-squared, F-statistic, and p-value due to an excess of free parameters compared to the number of data points.

The table clearly demonstrates that the proposed BUDDIP method achieved the largest adjusted R-squared value of 0.984, while the proposed UBUNet obtained the second-largest value of 0.977. Moreover, when comparing BUDDIP with methods that have fewer free parameters, the F-test reveals an extremely small p-value of 1.11×10^{-16}, indicating that BUDDIP's improvement is statistically significant with a 99.99% confidence level based on the evaluation data.

Table 4.6: Significane Test

method	adjusted R-squared	F-statistic	p-value
CNNAEU	-1.963	2700.85	1.11×10^{-16}
UnDIP	0.832	-	-
BUDDIP	0.984	-	-
MiSiCNet	0.279	-	-
MNN-BU-2	0.945	13.32	1.11×10^{-16}
UBUNet-II	0.977	3.19	1.11×10^{-16}
EGU-Net-ss	-	-	-

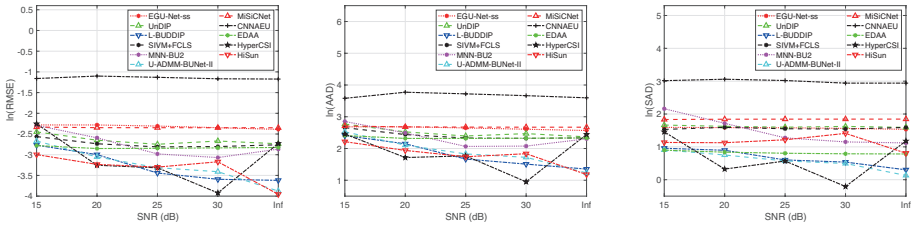
Performance vs. SNR

We conducted an evaluation of various methods for their robustness against noise contamination using synthetic dataset 1 for both linear and nonlinear unmixing cases. The default experimental settings were retained except for varying the signal-to-noise ratio (SNR) in the range of $[15, 20, 25, 30, \text{inf}]$ dB. To ensure a fair comparison, existing methods were used to generate an initialization for the unmixing networks MNN-BU and UBUNet, MiSiCNet, like UnDIP and EGU-Net, using SiVM+FCLS. The linear and nonlinear unmixing performances are presented in Fig. 4.13 and Fig. 4.14, respectively.

For the linear case, we observed that HiSun delivers better abundance estimation than other methods when the SNR is low, such as at $SNR = 15$ dB. As the SNR increases, the proposed UBUNet-II and L-BUDDIP benefit from the reduction in noise and achieve state-of-the-art performance when the SNR reaches 25 dB. However, when the SNR is 30 dB, HyperCSI achieves the best performance among the compared methods, but its performance deteriorates significantly when there is no noise.

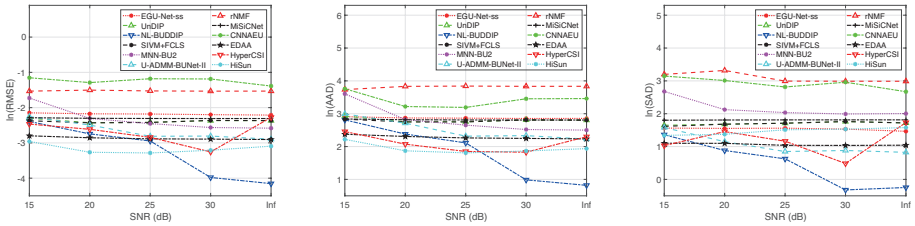
Regarding the nonlinear case, UnDIP and EGU-Net-ss performed similarly to SiVM+FCLS, while the proposed method showed better unmixing results when the SNR was relatively large. When the SNR is low, such as at 20 dB, HiSun performs the best in terms of abundance estimation. On the other hand, the proposed method, NL-BUDDIP, demonstrated the best performance among all the compared methods when the SNR was greater than 25 dB.

These results suggest that the proposed method achieves state-of-the-art robustness for both linear and nonlinear unmixing cases.



(a) Abundance RMSE versus SNR (b) Abundance AAD versus SNR (c) Endmember SAD versus SNR

Figure 4.13: Linear unmixing performance of various unmixing methods versus SNR (dB). (a) Abundance RMSE versus SNR. (b) Abundance AAD versus SNR. (c) Endmember SAD versus SNR.



(a) Abundance RMSE versus SNR (b) Abundance AAD versus SNR (c) Endmember SAD versus SNR

Figure 4.14: Nonlinear unmixing performance of various unmixing methods versus SNR (dB). (a) Abundance RMSE versus SNR. (b) Abundance AAD versus SNR. (c) Endmember SAD versus SNR.

4.5 Summary

In this chapter, a general neural network approach has been presented for solving the hyperspectral blind unmixing problem. Unlike the popular autoencoder structure, the proposed method leverages the Deep Image Prior (DIP) techniques and is comprised of three modules: an Endmember Estimation module (EDIP), an Abundance Estimation module (ADIP), and a Mixing Module (MM). The EDIP and ADIP modules produce estimates for the endmembers and abundances, respectively, while the MM, based on the unmixing model, creates a reconstruction of the observed HSI reflectances. Instead of using the general noise input employed in DIP techniques, the proposed method uses the estimations from existing unmixing methods as input, resulting in a more efficient DIP network structure with fewer learnable parameters. A new composite loss function has been proposed for both linear and

nonlinear unmixing cases, to ensure meaningful unmixing results and enhance performance. In the nonlinear case, an adaptive loss weight strategy has also been proposed to further improve performance. The effectiveness of the proposed methods has also been demonstrated through experiments. Although experiments have demonstrated the effectiveness of the proposed method, there exist certain limitations that should be taken into consideration. Firstly, the method's sensitivity to the hyperparameters used in the composite loss functions can significantly impact its performance. Secondly, the quality of the guidance employed during training also plays a crucial role in achieving better performance, with higher quality guidance leading to superior results. Lastly, the proposed adaptive loss weight strategy, while capable of delivering improved performance, can result in overfitting without meticulous tuning of the hyperparameters.

Chapter 5

Variations of BUDDIP

Despite the accomplishments of the BUDDIP framework mentioned previously, there are still a number of research questions that remain unanswered. Firstly, it continues to depend on the conventional CNN architecture, and it is unclear how to construct the DIP framework for the purpose of unmixing. Although unfolding techniques have emerged as a promising solution, existing unfolding based unmixing methods are based on a linear mixing model (LMM) and do not utilize convolution based techniques which are recognized as powerful tools in image processing tasks [82]. Secondly, it is unclear whether the performance can be further enhanced through the use of explicit regularizers.

In this chapter, we propose solutions to address these issues. To address the structural issue, we propose a new Network for Blind unmixing using ADMM unfolding (NBA). Specifically, we introduce a novel MatrixConv Unmixing (MCU) Model for both endmember estimation (EE) and abundance estimation (AE) tasks. This model has the ability to capture both global and local features through multiplication and convolution operations. To solve each task, we propose to use the alternating direction method of multipliers (ADMM) solver. We then apply algorithm unrolling techniques to each solver to construct two new unrolling-based DIP architectures, UEDIP and UADIP, for endmember and abundance estimation, respectively. Our final network, NBA, is

constructed by combining these two DIP networks. In addition, we also present a BUDDIP framework, which is unrolled from LMM-based ADMM solvers, for comparison purposes.

To tackle the second issue, we incorporate an explicit regularizer by denoising (RED) [102, 103] for endmember and abundance estimations. These additional regularizers can further enhance the network's performance.

5.1 MatrixConv Unmixing (MCU) Model

Our first method is the introduction of NBA, a novel unfolding-based network structure. Before we delve into the construction of UEDIP and UADIP, we will first introduce the MCU model that we propose for both endmember and abundance estimation tasks. This model serves as the basis for our network constructions. This chapter is devoted to the linear unmixing problem Eq. (4.1.1), since the nonlinear case, as discussed in NL-BUDDIP, can be viewed as a natural extension within the BUDDIP framework.

5.1.1 MCU-based EE

The blind unmixing problem represented by Eq. (4.1.1), is reduced to the EE problem when the abundance $\mathbf{A}$ is known a priori, which can be either given as ground truth or estimated by existing methods. The goal of the EE problem is to solve the following optimization:

$$\min_{\mathbf{E}} \frac{1}{2}\|\mathbf{Y}^T - \mathbf{A}^T\mathbf{E}^T\|_F^2, \quad s.t., \mathbf{E} \geq \mathbf{0} \tag{5.1.1}$$

where the transpose form of Eq. (4.1.1) has been used for the sake of simplicity in the following derivation. Suppose that $\mathbf{E}^T$ is obtained as

the result of convolutional sparse coding (CSC):

$$\mathbf{E}^T = \sum_{i=1}^{m_E} \mathbf{d}_i^E * \boldsymbol{\alpha}_i^E \tag{5.1.2}$$

where, $\mathbf{d}_i^E$ and $\boldsymbol{\alpha}_i^E$ are the i^{th} 1D convolutional kernel of size k_E and the corresponding sparse code, respectively. m_E is the number of convolutional kernels. As convolution is a linear operation, it can be represented as a linear operator according to [104]. Specifically, the convolution operation in Eq. (5.1.2) can be expressed as:

$$\sum_{i=1}^{m_E} \mathbf{d}_i^E * \boldsymbol{\alpha}_i^E = \mathbf{D}_E \boldsymbol{\Gamma}_E \tag{5.1.3}$$

Here, $\mathbf{D}_E$ and $\boldsymbol{\Gamma}_E$ are the convolutional dictionary and corresponding sparse code that encompass the sets of $\{\mathbf{d}_i^E\}_{i=1}^{m_E}$ and $\{\boldsymbol{\alpha}_i^E\}_{i=1}^{m_E}$, respectively.

Assuming $\mathbf{D}_E$ is known, which will subsequently be substituted with learnable parameters through unfolding techniques, the EE problem Eq. (5.1.1) is equivalent to an MCU problem, which can be formulated as:

$$\min_{\boldsymbol{\Gamma}_E} \frac{1}{2}\|\mathbf{Y}^T - \mathbf{A}^T\mathbf{D}_E\boldsymbol{\Gamma}_E\|_F^2 + \lambda\|\boldsymbol{\Gamma}_E\|_1 \tag{5.1.4}$$

Iterative update algorithms like ISTA and ADMM can be used to solve the above problem. We opt for the ADMM solver as recent studies have shown that unfolding ADMM-based networks perform better than unfolding ISTA-based networks [94]. The scaled form of augmented Lagrangian of Eq. (5.1.4) is given as:

$$\frac{1}{2}\|\mathbf{Y}^T - \mathbf{A}^T\mathbf{D}_E\boldsymbol{\Omega}_E\|_F^2 + \lambda\|\boldsymbol{\Gamma}_E\|_1 + \frac{\rho_E}{2}\|\boldsymbol{\Gamma}_E - \boldsymbol{\Omega}_E + \mathbf{u}_E\|_F^2 \tag{5.1.5}$$

where $\boldsymbol{\Omega}_E, \mathbf{u}_E, \rho_E$ are variables introduced by the ADMM algorithm.

And ADMM updates become:

$$\begin{cases} \mathbf{\Omega}_E^{j+1} = arg\min_{\mathbf{\Omega}_E} \frac{1}{2}\|\mathbf{Y}^T - \mathbf{A}^T\mathbf{D}_E\mathbf{\Omega}_E\|_F^2 + \frac{\rho_E}{2}\|\mathbf{\Gamma}_E^j - \mathbf{\Omega}_E + \mathbf{u}_E^j\|_F^2 \\ \mathbf{\Gamma}_E^{j+1} = arg\min_{\mathbf{\Gamma}_E} \lambda\|\mathbf{\Gamma}_E\|_1 + \frac{\rho_E}{2}\|\mathbf{\Gamma}_E - \mathbf{\Omega}_E^{j+1} + \mathbf{u}_E^j\|_F^2 \\ \mathbf{u}_E^{j+1} = \mathbf{u}_E^j + (\mathbf{\Gamma}_E^{j+1} - \mathbf{\Omega}_E^{j+1}) \end{cases} \tag{5.1.6}$$

The ADMM solver for the problem above is expressed as follows:

$$\begin{cases} \mathbf{\Omega}_E^{j+1} = ((\mathbf{A}^T\mathbf{D}_E)^T(\mathbf{A}^T\mathbf{D}_E) + \rho_E\mathbf{I})^{-1} \\ \qquad\qquad ((\mathbf{A}^T\mathbf{D}_E)^T\mathbf{Y}^T + \rho_E(\mathbf{\Gamma}_E^j + \mathbf{u}_E^j)) \\ \mathbf{\Gamma}_E^{j+1} = \textit{Soft}_{\frac{\lambda}{L}}((1 - \frac{\rho_E}{L})\mathbf{\Gamma}_E^j + \frac{\rho_E}{L}(\mathbf{\Omega}_E^{j+1} - \mathbf{u}_E^j)) \\ \mathbf{u}_E^{j+1} = \mathbf{u}_E^j + (\mathbf{\Gamma}_E^{j+1} - \mathbf{\Omega}_E^{j+1}) \end{cases} \tag{5.1.7}$$

where $\textit{Soft}_z(x) = sign(x) \cdot (|x| - z)_+$ is the soft-threshold operator.

5.1.2 MCU-based AE

Similarly, when the endmember $\mathbf{E}$ is known in advance, the blind unmixing problem is reduced to the abundance estimation (AE) problem. The latter involves minimizing

$$\min_{\mathbf{A}} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{A}\|_F^2, \quad s.t., \mathbf{A} \geq \mathbf{0}, \mathbf{A}^T\mathbf{1}_r = \mathbf{1}_n \tag{5.1.8}$$

Based on DIP techniques [37], we can remove the regularization term in Eq. (4.1.1). In contrast to model Eq. (5.1.2), we assume that the abundance is a result of nonnegative convolutional sparse coding (NCSC), expressed as follows:

$$\mathbf{A} = \sum_{i=1}^{m_A} \mathbf{d}_i^A * \boldsymbol{\alpha}_i^A, \quad where \quad \boldsymbol{\alpha}_i^A \geq 0, \ \forall i \tag{5.1.9}$$

Here, $\mathbf{d}_i^A$ and $\boldsymbol{\alpha}_i^A$ denote the i^{th} 2D convolutional kernel and its corresponding sparse code, respectively. m_A is the number of convolutional

kernels. We employ nonnegative sparse code since we experimentally find that it achieves better performance in the abundance estimation problem. This could be attributed to the inherent non-negativity of abundance. Furthermore, since convolution is a linear operation, it can be transformed into the following form:

$$\sum_{i=1}^{m_A} \mathbf{d}_i^A * \boldsymbol{\alpha}_i^A = \mathbf{D}_A \boldsymbol{\Gamma}_A \tag{5.1.10}$$

The matrix $\mathbf{D}_A$ is formed by concatenating the Toeplitz matrices that unroll the set $\{\mathbf{d}_i^A\}$, while $\boldsymbol{\Gamma}_A$ is the corresponding sparse code that unrolls $\{\boldsymbol{\alpha}_i^A\}$. Assuming $\mathbf{D}_A$ is known, the abundance estimation problem in Eq. (5.1.8) can be transformed into an MCU problem as follows:

$$\min_{\boldsymbol{\Gamma}_A} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{D}_A\boldsymbol{\Gamma}_A\|_F^2 + \lambda\|\boldsymbol{\Gamma}_A\|_1, \quad s.t., \boldsymbol{\Gamma}_A \geq \mathbf{0} \tag{5.1.11}$$

Similarly, we introduce the auxiliary variable $\boldsymbol{\Omega}_A, \mathbf{u}_A, \rho_A$ to obtain the augmented Lagrangian of Eq. (5.1.11) as follows:

$$\frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{D}_A\boldsymbol{\Omega}_A\|_F^2 + \lambda\|\boldsymbol{\Gamma}_A\|_1 + R_+(\boldsymbol{\Gamma}_A) + \frac{\rho_A}{2}\|\boldsymbol{\Gamma}_A - \boldsymbol{\Omega}_A + \mathbf{u}_A\|_F^2 \tag{5.1.12}$$

where $R_+(\boldsymbol{\Gamma}_A)$ represents the non-negative constraint. To solve this problem, we can use the ADMM solver, which involves three steps as follows:

$$\begin{cases} \boldsymbol{\Omega}_A^{j+1} = arg\min_{\boldsymbol{\Omega}_A} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}\mathbf{D}_A\boldsymbol{\Omega}_A\|_F^2 + \frac{\rho_A}{2}\|\boldsymbol{\Gamma}_A^j - \boldsymbol{\Omega}_A + \mathbf{u}_A^j\|_F^2 \\ \boldsymbol{\Gamma}_A^{j+1} = arg\min_{\boldsymbol{\Gamma}_A} \lambda\|\boldsymbol{\Gamma}_A\|_1 + R_+(\boldsymbol{\Gamma}_A) + \frac{\rho_A}{2}\|\boldsymbol{\Gamma}_A - \boldsymbol{\Omega}_A^{j+1} + \mathbf{u}_A^j\|_F^2 \\ \mathbf{u}_A^{j+1} = \mathbf{u}_A^j + (\boldsymbol{\Gamma}_A^{j+1} - \boldsymbol{\Omega}_A^{j+1}) \end{cases} \tag{5.1.13}$$

and ADMM solver proceeds as follows:

$$\begin{cases} \mathbf{\Omega}_A^{j+1} = ((\mathbf{E}\mathbf{D}_A)^T(\mathbf{E}\mathbf{D}_A) + \rho_A\mathbf{I})^{-1} \\ \qquad\qquad ((\mathbf{E}\mathbf{D}_A)^T\mathbf{Y} + \rho_A(\mathbf{\Gamma}_A^j + \mathbf{u}_A^j)) \\ \mathbf{\Gamma}_A^{j+1} = \max\left(\textit{Soft}_{\frac{\lambda}{L}}((1-\frac{\rho_A}{L})\mathbf{\Gamma}_A^j + \frac{\rho_A}{L}(\mathbf{\Omega}_A^{j+1} - \mathbf{u}_A^j)), 0\right) \\ \mathbf{u}_A^{j+1} = \mathbf{u}_A^j + (\mathbf{\Gamma}_A^{j+1} - \mathbf{\Omega}_A^{j+1}) \end{cases} \tag{5.1.14}$$

5.2 Unfolding based EDIP

In this section, we outline the process of constructing the UEDIP structure through the unfolding of the ADMM solver presented in Eq. (5.1.7).

Using unfolding techniques, we can introduce learnable parameters $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{F}_1$, $\mathbf{F}_2$, $\mathbf{F}_3$, s_1, s_2, and s_3 that are designed to replace the parameters in Eq. (5.1.7). Specifically, the operations $\mathbf{F}_1 * \mathbf{A}_1$ and $\mathbf{F}_2 * \mathbf{A}_2 \times \mathbf{F}_3$ serve as replacements for $(\mathbf{A}^T\mathbf{D}_E)^T$ and $((\mathbf{A}^T\mathbf{D}_E)^T(\mathbf{A}^T\mathbf{D}_E) + \rho_E\mathbf{I})^{-1}$, respectively. The parameters s_1, s_2, and s_3 are used to substitute for ρ_E, λ/L, and ρ_E/L in Eq. (5.1.7). It is worth noting that the linear matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ are employed alongside the convolutional dictionaries $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$, which are obtained by reversing Eq. (5.1.3). Thus, we define the operation of a network layer, as illustrated in Fig. 5.1, given by

$$\begin{cases} \mathbf{\Omega}_E^{j+1} = \mathbf{F}_2 * \mathbf{A}_2 \times \mathbf{F}_3 * (\mathbf{F}_1 * \mathbf{A}_1 \times \mathbf{Y}^T + s_1(\mathbf{\Gamma}_E^j + \mathbf{u}_E^j)) \\ \mathbf{\Gamma}_E^{j+1} = \textit{Soft}_{s_2}((1-s_3)\mathbf{\Gamma}_E^j + s_3(\mathbf{\Omega}_E^{j+1} - \mathbf{u}_E^j)) \\ \mathbf{u}_E^{j+1} = \mathbf{u}_E^j + (\mathbf{\Gamma}_E^{j+1} - \mathbf{\Omega}_E^{j+1}) \end{cases} \tag{5.2.1}$$

As depicted in Fig. 5.1, it is important to note that we exclude $\mathbf{F}_1 * \mathbf{A}_1 \times \mathbf{Y}^T$ from the layer-wise operations, as this value is shared among different layers to reduce the number of learnable parameters. Each layer takes as input $\mathbf{Y}, \mathbf{\Gamma}_E^{j-1}, \mathbf{u}_E^{j-1}$ and outputs updated estimations of $\mathbf{\Gamma}_E^j$ and $\mathbf{u}_E^j$. It is important to emphasize that this UEDIP structure is different

from that of a general-purpose deep neural network (DNN) or convolutional neural network (CNN), as it combines matrix multiplication (i.e., $\mathbf{A}_1$, $\mathbf{A}_2$) with convolutions (i.e., $\mathbf{F}_1$, $\mathbf{F}_2$, $\mathbf{F}_3$). In contrast, a general DNN does not involve convolutions, and in a general CNN, the linear matrix multiplication is usually added after a series of convolution layers, which serves as the output layer.

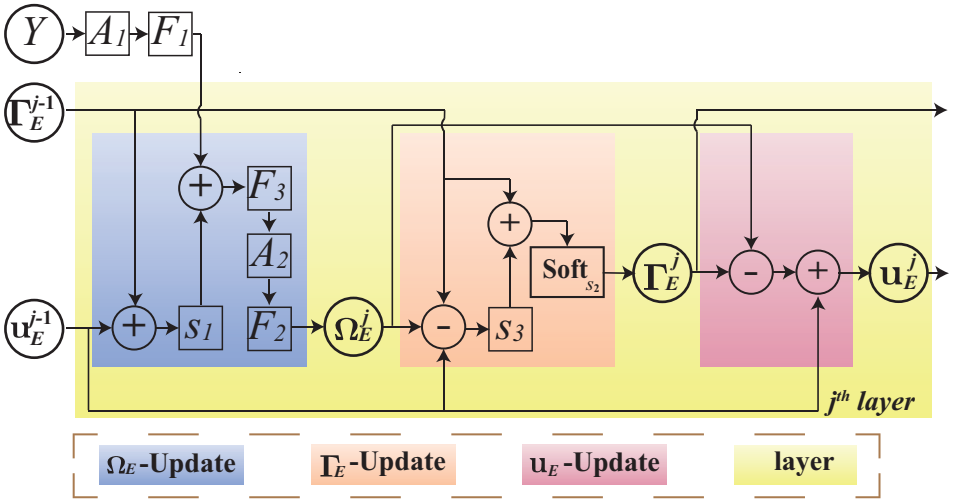


Figure 5.1: UEDIP layer-wise operations.

Fig. 5.2 illustrates the $J_E = 3$-layer unfolding ADMM-based EDIP (UEDIP) structure, which takes $\mathbf{Y}$ as input and outputs the endmember estimation $\hat{\mathbf{E}}$. In the first layer, we initialize $\mathbf{\Gamma}_E^0$ and $\mathbf{u}_E^0$ as zeros. In the last layer, we obtain the endmember estimation by passing the sparse code $\mathbf{\Gamma}_E^3$ through a convolutional operator $\mathbf{F}_3$, as shown in Fig. 5.2, and then applying a Sigmoid activation to satisfy the ENC constraint. The learnable parameters in each layer are untied. The UEDIP network is denoted as $f\boldsymbol{\theta}_E(\mathbf{Y})$, where $\boldsymbol{\theta}_E$ represents the learnable parameters in UEDIP.

Building upon the concept of DIP [36], we can assume that the abundance estimation $\mathbf{A}_G$ is available through existing algorithms like FCLS [2]. Once we obtain the endmember estimation $\hat{\mathbf{E}}$ using the

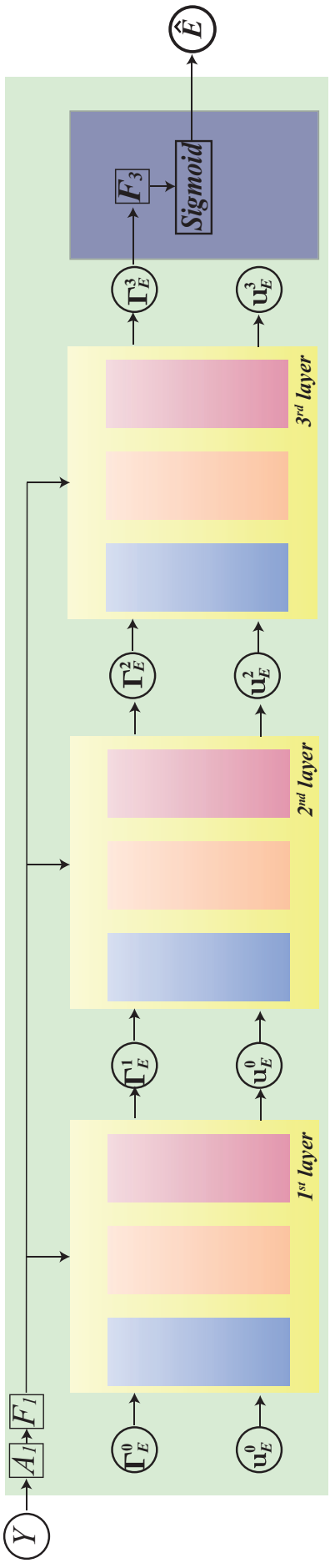


Figure 5.2: An illustration of 3-layer UEDIP structure. It takes $\mathbf{Y}$ as input and outputs endmember estimation $\hat{\mathbf{E}}$. The operation in each layer is shown in Fig. 5.1.

UEDIP network $f_{\boldsymbol{\theta}_E}$, the learning problem can be formulated as follows:

$$\boldsymbol{\theta}_E^* = \textit{arg}\min_{\boldsymbol{\theta}_E} \frac{1}{2}\|\mathbf{Y} - f_{\boldsymbol{\theta}_E}(\mathbf{Y})\mathbf{A}_G\|_F^2 \tag{5.2.2}$$

Once the parameters $\boldsymbol{\theta}_E^*$ have been learned, UEDIP is capable of estimating the endmember by computing $\hat{\mathbf{E}} = f_{\boldsymbol{\theta}_E^*}(\mathbf{Y})$.

5.3 Unfolding based ADIP

To construct a UADIP structure, we will employ unfolding techniques to the ADMM solver presented in Eq. (5.1.14). Specifically, we will define learnable parameters, namely $\mathbf{E}_1, \mathbf{E}_2, \mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3$, and v_1, v_2, v_3, such that $\mathbf{U}_1 * \mathbf{E}_1$ and $\mathbf{U}_2 * \mathbf{E}_2 \times \mathbf{U}_3$ can replace the role of $(\mathbf{E}\mathbf{D}_A)^T$ and $((\mathbf{E}\mathbf{D}_A)^T(\mathbf{E}\mathbf{D}_A) + \rho_A\mathbf{I})^{-1}$ in Eq. (5.1.14), while v_1, v_2, v_3 can play the role of $\rho_A, \lambda/L, \rho_A/L$ in the same equation. It should be noted that since $\max(\textit{Soft}(\cdot))$ is equivalent to a shifted *ReLU*, we can define a layer of network operation as follows:

$$\begin{cases} \boldsymbol{\Omega}_A^{j+1} = \mathbf{U}_2 * \mathbf{E}_2 \times \mathbf{U}_3 * (\mathbf{U}_1 * \mathbf{E}_1 \times \mathbf{Y} + v_1(\boldsymbol{\Gamma}_A^j + \mathbf{u}_A^j)) \\ \boldsymbol{\Gamma}_A^{j+1} = \textit{ReLU}((1 - v_3)\boldsymbol{\Gamma}_A^j + v_3(\boldsymbol{\Omega}_A^{j+1} - \mathbf{u}_A^j) - v_2) \\ \mathbf{u}_A^{j+1} = \mathbf{u}_A^j + (\boldsymbol{\Gamma}_A^{j+1} - \boldsymbol{\Omega}_A^{j+1}) \end{cases} \tag{5.3.1}$$

Fig. 5.3 illustrates the layer-wise operation, where we exclude $\mathbf{U}_1 * \mathbf{E}_1 \times \mathbf{Y}$, since it is shared among different layers. Each layer produces updated estimations of $\boldsymbol{\Gamma}_A$ and $\mathbf{u}_A$.

An unfolding ADMM based ADIP (UADIP) structure with three layers ($J_A = 3$) is illustrated in Fig. 5.4. It takes $\mathbf{Y}$ as input and outputs abundance estimation $\hat{\mathbf{A}}$. We use zero initialization for $\boldsymbol{\Gamma}_A^0$ and $\mathbf{u}_A^0$ in the first layer, and in the last layer, the sparse code $\boldsymbol{\Gamma}_A^3$ is generated and passed through a convolutional operator $\mathbf{U}_3$ to estimate the abundance, following our model in Eq. (5.1.9). Finally, a Softmax activation

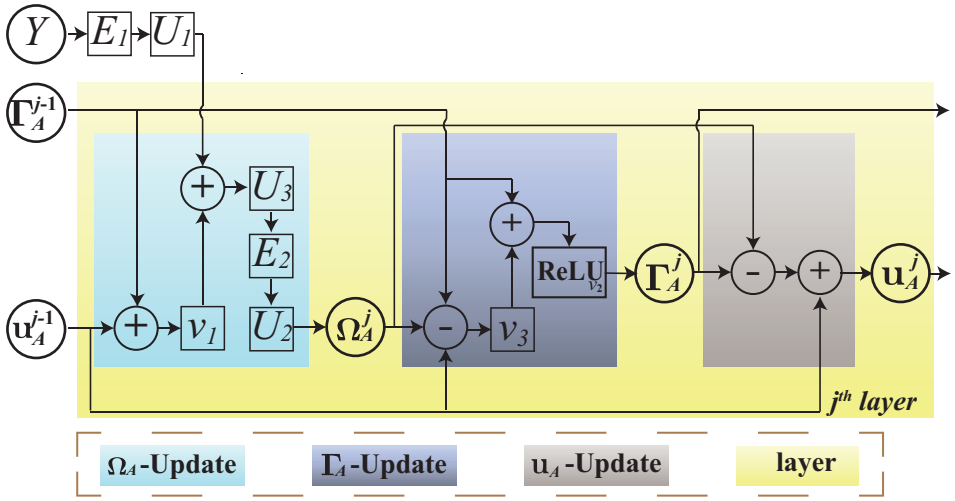


Figure 5.3: UADIP layer-wise operations.

is used to impose the ASC and ANC constraints. Similar to UEDIP, we adopt the untied parameterisation strategy for different layers in UADIP. The UADIP network is denoted as $f_{\boldsymbol{\theta}_A}(\mathbf{Y})$, where $\boldsymbol{\theta}_A$ represents the learnable parameters in UADIP.

Assuming that we have the endmember estimation $\mathbf{E}_G$ from existing algorithms such as SiVM [1], after generating the abundance estimation $\hat{\mathbf{A}}$ using UADIP network $f_{\boldsymbol{\theta}_A}$, we arrive at the learning problem given by:

$$\boldsymbol{\theta}_A^* = \textit{arg}\min_{\boldsymbol{\theta}_A} \frac{1}{2}\|\mathbf{Y} - \mathbf{E}_G f_{\boldsymbol{\theta}_A}(\mathbf{Y})\|_F^2 \tag{5.3.2}$$

Once the parameters $\boldsymbol{\theta}_A^*$ are learned, UADIP can estimate the abundance as $\hat{\mathbf{A}} = f_{\boldsymbol{\theta}_A^*}(\mathbf{Y})$.

5.4 Network for Blind-unmixing using ADMM unfolding (NBA)

In this section, we propose the Double DIP Network for Blind-Unmixing using ADMM unfolding (NBA), which combines the UEDIP and UADIP models. Given the linear model Eq. (2.2.2), we first estimate the endmember and abundance using UEDIP and UADIP, respectively, resulting in $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$. We then obtain the reconstructed HSI observations by

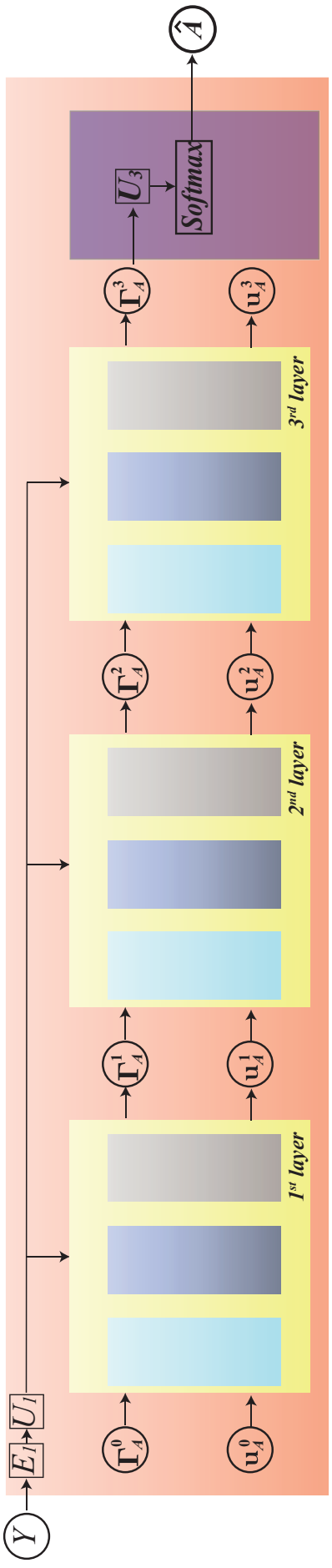


Figure 5.4: An illustration of 3-layer UADIP structure. It takes $\mathbf{Y}$ as input and outputs abundance estimation $\hat{\mathbf{A}}$. The operation in each layer is shown in Fig. 5.3.

computing:

$$\hat{\mathbf{Y}} = \hat{\mathbf{E}}\hat{\mathbf{A}} \tag{5.4.1}$$

The overall structure of the proposed NBA network is illustrated in Fig. 5.5, where we denote the learnable parameters in the NBA network as $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_E, \boldsymbol{\theta}_A\}$, where $\boldsymbol{\theta}_E, \boldsymbol{\theta}_A$ correspond to the learnable parameters in UEDIP and UADIP, respectively. It is important to note that while our focus is on linear unmixing, the NBA framework can also be readily applied to nonlinear unmixing problems, similar to the BUDDIP method.

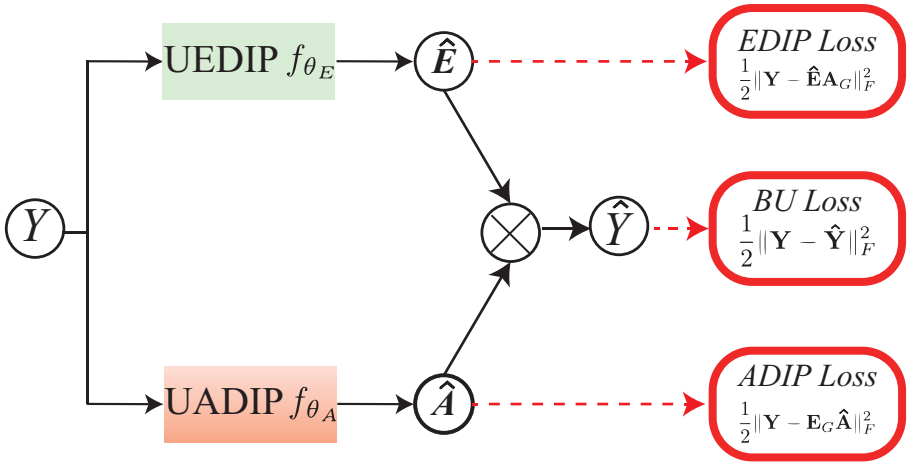


Figure 5.5: The proposed NBA framework.

Similar to BUDDIP, the NBA network provides three outputs: endmember estimation $\hat{\mathbf{E}}$, abundance estimation $\hat{\mathbf{A}}$ and HSI reconstructions $\hat{\mathbf{Y}}$ when given an HSI image $\mathbf{Y}$. To train the NBA network in an end-to-end manner, we propose using the composite loss function, similar to Eq. (4.3.8), which is defined as follows:

$$\begin{aligned} L(\boldsymbol{\Theta}) &= \alpha_1 \cdot L_{EDIP} + \alpha_2 \cdot L_{ADIP} + \alpha_3 \cdot L_{BU} \\ \text{where, } L_{EDIP}(\boldsymbol{\theta}_E) &= \frac{1}{2}\|\mathbf{Y} - f_{\boldsymbol{\theta}_E}(\mathbf{Y})\mathbf{A}_G\|_F^2 \\ L_{ADIP}(\boldsymbol{\theta}_A) &= \frac{1}{2}\|\mathbf{Y} - \mathbf{E}_G f_{\boldsymbol{\theta}_A}(\mathbf{Y})\|_F^2 \\ L_{BU}(\boldsymbol{\Theta}) &= \frac{1}{2}\|\mathbf{Y} - \hat{\mathbf{Y}}\|_F^2 \end{aligned} \tag{5.4.2}$$

The loss weights, $\alpha_1, \alpha_2, \alpha_3$, play a crucial role in controlling the im-

portance of each loss term in the overall loss function. In addition, To guide the training process, we generate the endmember and abundance guidance terms, denoted by $\mathbf{E}_G$ and $\mathbf{A}_G$, respectively, using established unmixing techniques such as SiVM [1] and FCLS [2]. By incorporating these guidance terms into the loss function, the network produces meaningful estimates of endmembers and abundances, while still allowing for the exploration of better solutions beyond the guidance provided by $\mathbf{E}_G$ and $\mathbf{A}_G$. It should be noted that in Eq. (5.4.2), we have excluded the angle loss terms in Eq. (4.3.8) to simplify the hyperparameter tuning and facilitate faster experimental validation.

5.5 BUDDIP unrolled from LMM

To demonstrate the efficacy of the MCU model, we also introduce a variant of the BUDDIP network, denoted as LA-BUDDIP, that is unrolled from LMM-based ADMM solvers instead of MCU-based ones. To begin with, we consider the linear endmember estimation (EE) model Eq. (5.1.1) and linear abundance estimation (AE) model Eq. (5.1.8), and develop ADMM solvers for them. Following the unfolding technique, we can construct the EDIP and ADIP modules for the LA-BUDDIP network by unrolling the corresponding ADMM solvers. Therefore, we will focus on the construction of the EDIP and ADIP modules in the following, while the remaining part remains unchanged from the BUDDIP.

5.5.1 Unrolling for EDIP

We begin by introducing the auxiliary variable $\mathbf{X}_E$ and reformulating the linear EE model Eq. (5.1.1) as follows:

$$\min_{\mathbf{E},\mathbf{X}_E} \frac{1}{2}\|\mathbf{Y}^T - \mathbf{A}^T\mathbf{X}_E^T\|_F^2 + \lambda R_+(\mathbf{E}^T),\ \text{s.t. } \mathbf{X}_E^T = \mathbf{E}^T \tag{5.5.1}$$

where, $R_+(\mathbf{E})$ denotes the constraint $\{\mathbf{E} : \mathbf{E} \geq \mathbf{0}\}$. For the sake of simplicity, the transform notation T is omitted and the objective is reformulated in augmented Lagrangian form as follows:

$$\min_{\mathbf{E},\mathbf{X}_E} \frac{1}{2}\|\mathbf{Y} - \mathbf{A}\mathbf{X}_E\|_F^2 + \lambda R_+(\mathbf{E}) + \frac{\rho}{2}\|\mathbf{u}_E + \mathbf{E} - \mathbf{X}_E\|_2^2 \tag{5.5.2}$$

The problem can be tackled using the ADMM Solver, which yields the following iterative updates:

$$\begin{cases} \mathbf{X}_E^{j+1} = (\mathbf{A}^T\mathbf{A} + \rho\mathbf{I})^{-1}(\mathbf{A}^T\mathbf{Y} + \rho(\mathbf{E}^j + \mathbf{u}_E^j)) \\ \mathbf{E}^{j+1} = \mathsf{max}(\mathbf{E}^j - \rho(\mathbf{E}^j + \mathbf{u}_E^j - \mathbf{X}_E^{j+1}), 0) \\ \mathbf{u}_E^{j+1} = \mathbf{u}_E^j + (\mathbf{E}^{j+1} - \mathbf{X}_E^{j+1}) \end{cases} \tag{5.5.3}$$

Utilizing unfolding techniques, we can define the learnable parameters $\boldsymbol{W}_E, \boldsymbol{B}_E, \theta_E$ and η_E, and subsequently rephrase the equation above as follows:

$$\begin{cases} \mathbf{X}_E^{j+1} = \boldsymbol{W}_E\mathbf{Y} + \boldsymbol{B}_E(\mathbf{E}^j + \mathbf{u}_E^j) \\ \mathbf{E}^{j+1} = \mathsf{max}((1-\theta_E)\mathbf{E}^j + \theta_E(\mathbf{X}_E^{j+1} - \mathbf{u}_E^j), 0) \\ \mathbf{u}_E^{j+1} = \mathbf{u}_E^j + \eta_E(\mathbf{E}^{j+1} - \mathbf{X}_E^{j+1}) \end{cases} \tag{5.5.4}$$

Here, $\boldsymbol{W}_E$ takes the place of $(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I})^{-1}\mathbf{A}^T$, $\boldsymbol{B}_E$ acts as $(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I})^{-1}\rho$, and θ_E corresponds to ρ. Additionally, η_E is used as a step-size in a manner similar to Eq. (3.2.5) to enhance adaptability to the specific problem at hand. Analogous to UEDIP, the operation of a network layer is defined by Eq. (5.5.4), and the EDIP with J_{LE} layers in LA-BUDDIP is created by concatenating J_{LE} consecutive operations described in Eq. (5.5.4). The framework with $J_{LE} = 3$ is illustrated in Fig. 5.6. As with UEDIP, the parameters in each layer are untied.

Figure 5.6: An illustration of 3-layer EDIP structure for LA-BUDDIP. It takes $\mathbf{Y}$ as input and outputs endmember estimation $\hat{\mathbf{E}}$. The operation in each layer is defined in Eq. (5.5.4).

5.5.2 Unrolling for ADIP

We will now explain in detail the ADIP construction for LA-BUDDIP. To achieve this, we will introduce an auxiliary variable $\mathbf{X}_A$ and reformulate the linear AE model Eq. (5.1.8) as follows:

$$\min_{\mathbf{A},\mathbf{X}_A} \frac{1}{2}\|\mathbf{Y}-\mathbf{E}\mathbf{X}_A\|_F^2+\lambda R_A(\mathbf{A}), \quad s.t., \mathbf{X}_A=\mathbf{A} \tag{5.5.5}$$

where $R_A(\mathbf{A})$ represents the constraint in the linear AE model Eq. (5.1.8), $\{\mathbf{A}:\mathbf{A}\geq\mathbf{0},\mathbf{A}^T\mathbf{1}_r=\mathbf{1}_n\}$. We re-write the model in augmented Lagrangian form as follows:

$$\min_{\mathbf{A},\mathbf{X}_A} \frac{1}{2}\|\mathbf{Y}-\mathbf{E}\mathbf{X}_A\|_F^2+\lambda R_A(\mathbf{A})+\frac{\rho}{2}\|\mathbf{u}_A+\mathbf{A}-\mathbf{X}_A\|_2^2 \tag{5.5.6}$$

An effective approach to solving this problem is by using the ADMM Solver, which results in the following iterative updates:

$$\begin{cases} \mathbf{X}_A^{j+1}=(\mathbf{E}^T\mathbf{E}+\rho\mathbf{I})^{-1}(\mathbf{E}^T\mathbf{Y}+\rho(\mathbf{A}^j+\mathbf{u}_A^j)) \\ \mathbf{A}^{j+1}=\text{Softmax}(\mathbf{A}^j-\rho(\mathbf{A}^j+\mathbf{u}_A^j-\mathbf{X}_A^{j+1})) \\ \mathbf{u}_A^{j+1}=\mathbf{u}_A^j+(\mathbf{A}^{j+1}-\mathbf{X}_A^{j+1}) \end{cases} \tag{5.5.7}$$

where $Softmax$ is used to impose the constraint $R_A(\mathbf{A})$. We can define the learnable parameters $\boldsymbol{W}_A$, $\boldsymbol{B}_A$, θ_A, and η_A using unfolding techniques. Then, we can rewrite the aforementioned equation as follows:

$$\begin{cases} \mathbf{X}_A^{j+1}=\boldsymbol{W}_A\mathbf{Y}+\boldsymbol{B}_A(\mathbf{A}^j+\mathbf{u}_A^j) \\ \mathbf{A}^{j+1}=\text{Softmax}((1-\theta_A)\mathbf{A}^j-\theta_A(\mathbf{u}_A^j-\mathbf{X}_A^{j+1})) \\ \mathbf{u}_A^{j+1}=\mathbf{u}_A^j+\eta_A(\mathbf{A}^{j+1}-\mathbf{X}_A^{j+1}) \end{cases} \tag{5.5.8}$$

Specifically, $\boldsymbol{W}_A$ replaces $(\mathbf{E}^T\mathbf{E}+\rho\mathbf{I})^{-1}\mathbf{E}^T$, $\boldsymbol{B}_A$ replaces $(\mathbf{E}^T\mathbf{E}+\rho\mathbf{I})^{-1}\rho$, and θ_A corresponds to ρ. Additionally, η_A acts as a step-size, similar

to Eq. (3.2.5), to enhance adaptability to the specific problem at hand.

Similar to UADIP, the operation of a network layer is defined by Eq. (5.5.8). By concatenating J_{LA} consecutive operations described in Eq. (5.5.8), we can construct the ADIP with J_{LA} layers in LA-BUDDIP, which is depicted in Fig. 5.7 for the case where $J_{LA} = 3$. Like UADIP, the parameters in each layer are untied.

5.6 Explicit Regularisations

According to the DIP technique described in [37], the explicit regularizer $R(\cdot)$ in a recovery problem, such as the one in Eq. (4.1.1), can be dropped. The neural network parameterization instead, can implicitly capture the prior. Later work by [103] found that the performance of DIP networks can be further improved by adding an explicit RED regularizer [102]. This is because RED provides an additional explicit regularization in addition to the implicit one. In this study, we take this one step further by introducing two RED regularizers into double DIP network structures to further enhance the network's performance. Specifically, we propose adding explicit RED regularization to both $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$ in Eq. (5.4.2). This results in the following loss function:

$$\mathcal{L}_{\mathcal{R}} = L + \alpha_4 \cdot \rho(\hat{\mathbf{E}}) + \alpha_5 \cdot \rho(\hat{\mathbf{A}}) \tag{5.6.1}$$

For the sake of simplicity, we have omitted the network parameters $\boldsymbol{\Theta}$ in the loss function L. The RED regularizer $\rho(\boldsymbol{x})$ in Eq. (5.6.1) is defined as follows:

$$\rho(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T(\boldsymbol{x} - f_D(\boldsymbol{x})) \tag{5.6.2}$$

where $f_D(\cdot)$ is the denoiser function, which can be any well-established method such as Non-Local Means (NLM) [105] or BM3D [106]. The RED regularizer minimizes the inner product between the image $\boldsymbol{x}$ and

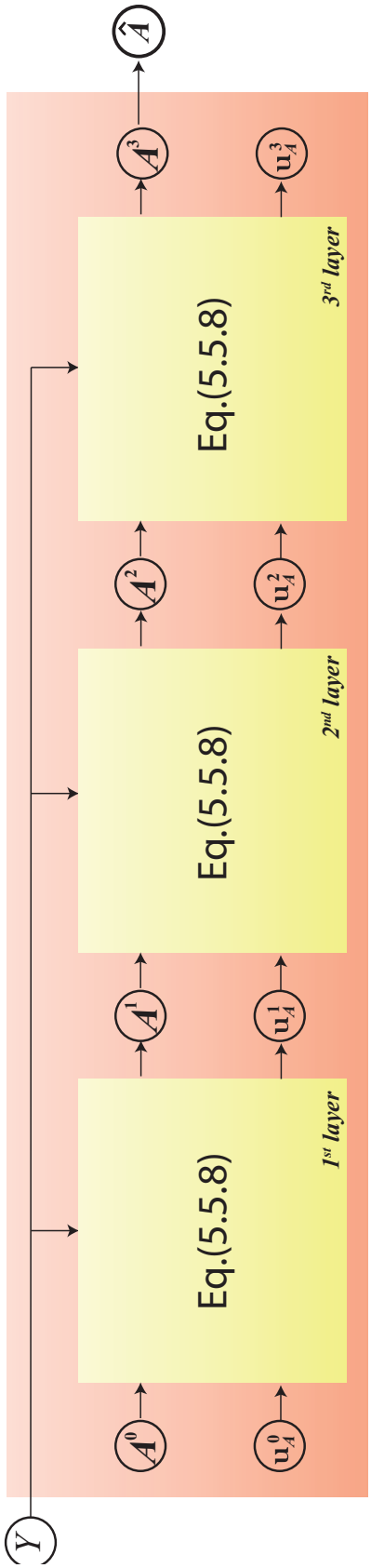


Figure 5.7: An illustration of 3-layer ADIP structure for LA-BUDDIP. It takes $\mathbf{Y}$ as input and outputs endmember estimation $\hat{\mathbf{A}}$. The operation in each layer is defined in Eq. (5.5.8).

its denoising residual $\boldsymbol{x} - f_D(\boldsymbol{x})$. RED can represent a variety of regularizations by incorporating different image denoisers f_D. In this work, we use NLM as our default denoiser. We note that RED is intended for 3D image data, while the endmembers are embedded in 2D space, $\mathbf{E} \in R^{P \times R}$. To address this issue, we expand the endmembers to 3D space, $R^{1 \times P \times R}$, and treat them as grey-image-like data.

It is worth noting that the loss function defined in Eq. (5.6.1) can improve not only the performance of the NBA network, but also that of the BUDDIP network. As depicted in Fig. 4.1 and Fig. 5.5, these two networks differ only in their structures and inputs. This observation is further illustrated in Fig. 5.8.

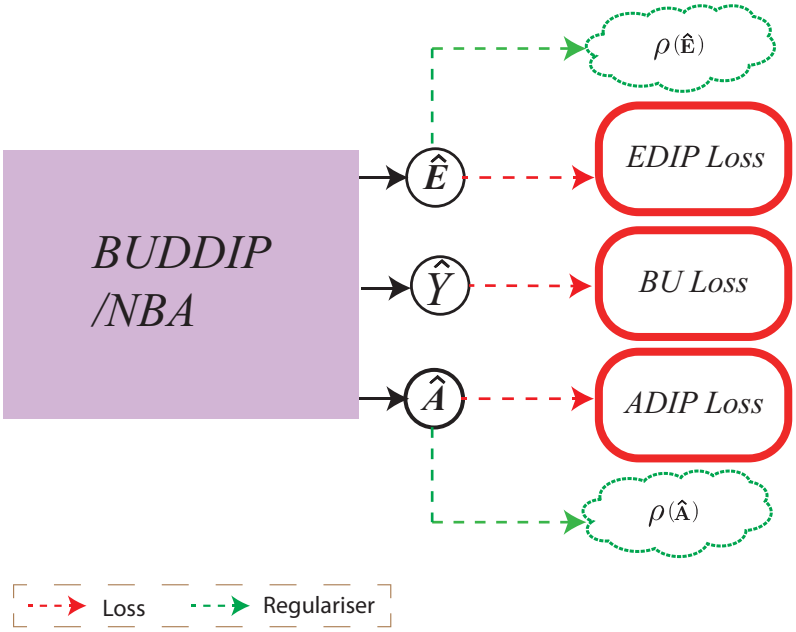


Figure 5.8: The proposed explicit regularisations to enhance BUDDIP/NBA network.

After adding the RED regularizers, it is not straightforward to train the network with the objective in Eq. (5.6.1). Therefore, in this work, we propose using the ADMM solver [102, 103] to solve Eq. (5.6.1). We first rewrite the objective in the form of a scaled Augmented Lagrangian (AL):

$$\begin{aligned} L + \alpha_4 \cdot \rho(\mathbf{X}_E) + \frac{\mu_E}{2} \|\mathbf{X}_E - \hat{\mathbf{E}} - \mathbf{d}_E\|_F^2 \\ + \alpha_5 \cdot \rho(\mathbf{X}_A) + \frac{\mu_A}{2} \|\mathbf{X}_A - \hat{\mathbf{A}} - \mathbf{d}_A\|_F^2 \end{aligned} \tag{5.6.3}$$

where, $\{\mathbf{X}_E, \mathbf{X}_A, \mathbf{d}_E, \mathbf{d}_A\}$ are auxiliary variables introduced via AL, and μ_E, μ_A are also the hyperparameters of AL. The ADMM solver to Eq. (5.6.3) gives the following iterative update rules:

$$\begin{cases} \boldsymbol{\Theta}^{t+1} = \textit{arg}\min_{\boldsymbol{\Theta}} L + \frac{\mu_E}{2}\|\mathbf{X}_E^t - \hat{\mathbf{E}} - \mathbf{d}_E^t\|_F^2 \\ \qquad\qquad + \frac{\mu_A}{2}\|\mathbf{X}_A^t - \hat{\mathbf{A}} - \mathbf{d}_A^t\|_F^2 \end{cases} \tag{5.6.4}$$

$$\begin{cases} \mathbf{X}_E^{t+1}, \mathbf{X}_A^{t+1} = \textit{arg}\min_{\mathbf{X}_E, \mathbf{X}_A} \alpha_4 \cdot \rho(\mathbf{X}_E) + \alpha_5 \cdot \rho(\mathbf{X}_A) \\ \qquad\qquad + \frac{\mu_E}{2}\|\mathbf{X}_E - \hat{\mathbf{E}}^{t+1} - \mathbf{d}_E^t\|_F^2 \\ \qquad\qquad + \frac{\mu_A}{2}\|\mathbf{X}_A^t - \hat{\mathbf{A}}^{t+1} - \mathbf{d}_A^t\|_F^2 \end{cases} \tag{5.6.5}$$

$$\begin{cases} \{\mathbf{d}_E^{t+1}, \mathbf{d}_A^{t+1}\} = \{\mathbf{d}_E^t + (\hat{\mathbf{E}}^{t+1} - \mathbf{X}_E^{t+1}), \\ \qquad\qquad \mathbf{d}_A^t + (\hat{\mathbf{A}}^{t+1} - \mathbf{X}_A^{t+1})\} \end{cases} \tag{5.6.6}$$

The optimization of the first objective Eq. (5.6.4) is performed through

Algorithm 2: ADMM solver of Objective Eq. (5.6.3).

Parameters:

- $\alpha_1 \sim \alpha_5$ - loss weights
- μ_E, μ_A - ADMM hyper-parameters
- T - the maximum number of ADMM iterations.

Init: $\mathbf{d}_E^0, \mathbf{d}_A^0, \mathbf{X}_E^0, \mathbf{X}_A^0 \leftarrow 0, 0, 0, 0$, $t \leftarrow 0$, and initialise network randomly.

for $t \leftarrow 1$ **to** T **do**
  Update $\boldsymbol{\Theta}$: solving Eq. (5.6.4) via training network.
  Update $\mathbf{X}_E, \mathbf{X}_A$: using fixed point strategy Eq. (5.6.7).
  Update $\mathbf{d}_E, \mathbf{d}_A$: using Eq. (5.6.6).

network training, where the outputs $\hat{\mathbf{E}}$ and $\hat{\mathbf{A}}$ depend on $\boldsymbol{\Theta}$. For the second objective Eq. (5.6.5), there are two possible solutions: fixed-point based or gradient-descent based [102, 103]. In this study, we employ the fixed-point based solution, which is formulated as follows:

$$\begin{cases} \mathbf{X}_E^{t+1} = \frac{1}{\alpha_4 + \mu_E}(\alpha_4 f_D(\mathbf{X}_E^t) + \mu_E(\hat{\mathbf{E}}^{t+1} + \mathbf{d}_E^t)) \\ \mathbf{X}_A^{t+1} = \frac{1}{\alpha_5 + \mu_A}(\alpha_5 f_D(\mathbf{X}_A^t) + \mu_A(\hat{\mathbf{A}}^{t+1} + \mathbf{d}_A^t)) \end{cases} \tag{5.6.7}$$

Finally, the last update of Eq. (5.6.6) is a simple update of the Lagrangian multiplier. The overall steps of our proposed approach are summarized in Algorithm 2.

5.7 Experiment

In this section, we will evaluate the efficacy of the proposed methods using the same performance metrics as discussed in Sec. 3.5.1. These metrics include RMSE and AAD for abundance estimation and SAD for endmember estimation.

5.7.1 Data

We evaluated the effectiveness of our proposed methods using a synthesized HSI dataset generation procedure, based on the method outlined in [18]. The process consists of the following steps:

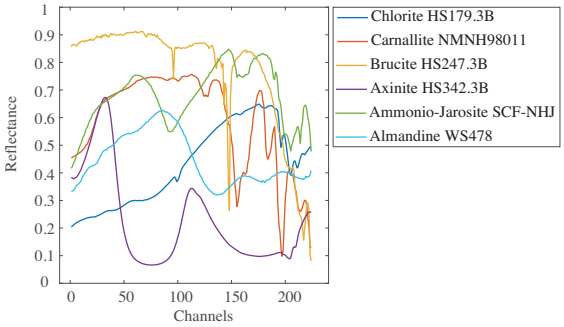


Figure 5.9: Endmember signatures for synthetic data.

- *Endmember generation.* Firstly, we generated endmembers by randomly selecting six spectral signatures from the USGS spectral library (splib06) [81], which contains spectral reflectance values for various minerals over 224 channels. These six signatures formed a 224×6 endmember matrix, as shown in Fig. 5.9.

- *Abundance generation.* we generated abundances for a synthetic image of size $a^2 \times a^2$ pixels. We divided the image into a^2

disjoint patches of size $a \times a$ pixels and assigned two endmembers randomly to all pixels of a patch with fractions γ and $1-\gamma$, while the remaining four endmembers were assigned a value of 0. The abundance map was then convolved with a Gaussian filter of size $(a+1)\times(a+1)$ with variance 2, followed by a pixel-wise re-scaling to meet the abundance sum-to-one (ASC) constraint. In this experiment, we set $a = 10$ and $\gamma = 0.8$.

- *Mixing process.* We employ the linear mixing model described in Eq. (2.2.2) to generate synthetic data for linear unmixing problems.

- *Noise contamination.* Finally, we added additive white Gaussian noise (AWGN) to the generated HSI data. The signal-to-noise ratio (SNR) was defined as $SNR = 10\log_{10}\left(E[\mathbf{x}^T\mathbf{x}]/E[\mathbf{n}^T\mathbf{n}]\right)$, where $\mathbf{x}$ represents the original, noise-free HSI data, and $\mathbf{n}$ is the added noise.

5.7.2 Effectiveness of NBA

We investigated the effectiveness of the proposed NBA for linear unmixing on the synthetic dataset contaminated with additive white Gaussian noise (AWGN) at an SNR of 30 dB. The dataset comprised 100 by 100 pixels. The UADIP and UEDIP networks used in our experiments had $J_A = 3$ layers, $m_A = 128$ kernels, and a kernel size of $k_A = 5$, and $J_E = 1$ layer, $m_E = 128$ kernels, and a kernel size of $k_E = 5$, respectively. Our networks were trained to minimize the composite loss function Eq. (5.4.2) using the Adam optimizer with a learning rate of $1e-3$ for 5000 epochs. As default hyperparameters, we set $\alpha_1 = 0.1$, $\alpha_2 = 0.01$, and $\alpha_3 = 1.0$ in the composite loss function Eq. (5.4.2). The training guidance is generated via SiVM+FCLS.

Network Depth

The network performance is usually heavily dependent on the network depth. In this study, the impact of network depth on the unmixing performance of the NBA network is evaluated and reported in Table 5.2. The default settings are used except for the depth of the UEDIP and UADIP networks, which are varied between 1 and 5. The corresponding numbers of learnable parameters for various network depths are shown in Table 5.1. The results show that the best SAD is achieved with $J_E = 2, J_A = 5$, and the best RMSE/AAD is achieved with $J_E = 4, J_A = 2$. However, both settings have over 11.5×10^5 learnable parameters. Therefore, for a balance between network performance and complexity, $J_E = 1, J_A = 3$ is chosen as the default setting for subsequent experiments.

Table 5.1: The number of learnable parameters with different network depth.

# learnable parameters: $\times 10^5$		UADIP depth J_A				
		1	2	3	4	5
UEDIP depth J_E	1	7.12	7.50	7.89	8.27	8.65
	2	9.98	10.37	10.75	11.14	11.52
	3	12.85	13.24	13.62	14.00	14.39
	4	15.72	16.10	16.49	16.87	17.26
	5	18.59	18.97	19.36	19.74	20.12

Kernel Size

In this experiment, we investigate the impact of the convolutional kernel size on the performance of the NBA network. The default settings are retained while varying the kernel size in the range of $[1, 3, 5, 7, 9]$. The results are reported in Table 5.3. It is clear that when $k_E = k_A = 5$, the proposed NBA network delivers the best AE performance, i.e., the RMSE and AAD are 0.027 and 4.719, respectively. In the meantime,

Table 5.2: Impact of network depth.

RMSE/AAD/SAD		UADIP depth J_A				
		1	2	3	4	5
UEDIP depth J_E	1	0.0354/6.1334/1.841	0.0327/5.7093/1.851	0.0271/4.7193/1.783	0.0274/4.7498/1.837	0.0282/4.8547/1.815
	2	0.0330/5.7389/1.810	0.0393/6.8544/2.214	0.0293/5.0439/1.803	0.0328/5.6541/1.800	0.0280/4.7900/**1.765**
	3	0.0323/5.5989/1.831	0.0414/7.3061/2.242	0.0412/7.2690/2.243	0.0398/6.8344/2.227	0.0290/4.9993/1.816
	4	0.0362/6.3297/1.826	**0.0262/4.6140**/1.80	0.0280/4.8669/1.822	0.0335/5.7946/1.929	0.0336/5.8215/1.904
	5	0.0342/5.9275/1.841	0.0325/5.6432/1.776	0.0320/5.5348/1.844	0.0316/5.4428/1.820	0.0302/5.2054/1.789

Table 5.3: Impact of convolutional kernel size.

RMSE/AAD/SAD		UADIP kernel size k_A				
		1	3	5	7	9
UEDIP kernel size k_E	1	0.200/41.621/20.132	0.181/37.500/20.335	0.039/6.838/2.253	0.032/5.521/1.804	0.245/55.133/4.542
	3	0.127/26.250/2.948	0.030/5.174/**1.705**	0.237/52.544/24.390	0.032/5.504/1.803	0.245/55.110/4.354
	5	0.054/9.765/1.945	0.235/51.832/19.333	**0.027/4.719**/1.783	0.030/5.256/1.769	0.246/55.047/4.317
	7	0.230/50.113/20.156	0.203/42.483/20.237	0.199/41.363/21.036	0.032/5.444/1.829	0.029/4.937/1.775
	9	0.191/41.687/9.037	0.207/43.638/18.853	0.198/41.747/18.236	0.209/44.345/19.116	0.244/54.948/6.076

the best SAD of EE is 1.705 when $k_E = k_A = 3$. In this work, we set the default kernel size to be $k_E = k_A = 5$ for a good trade-off between the accuracy of AE and EE.

Number of kernels

In this experiment, we investigate the effect of the number of kernels on the final unmixing performance of the NBA network. We keep the default settings except for varying the number of kernels in the range of $[32, 64, 128, 256]$. The results are presented in Table 5.4. The results demonstrate that the proposed NBA network achieves the best abundance estimation performance with an RMSE of 0.027 and AAD of 4.705 when $m_E = 32$ and $m_A = 128$. Meanwhile, the best SAD of 1.740 is obtained with $m_E = 32, m_A = 64$ and $m_E = 128, m_A = 256$. We choose $m_E = m_A = 128$ as our default settings because it provides a balance between the accuracy of AE and EE.

Impact of NCSC

We now assess the impact of NCSC Eq. (5.1.9) by comparing the performance of two blind linear unmixing networks: NBA and NBA with CSC. The ADIP network of NBA is unfolded from the NCSC model, while the ADIP of NBA with CSC is unfolded from the CSC model. We used SiVM+FCLS to generate training guidance for both networks denoted as $\mathbf{E}_G, \mathbf{A}_G$ and set identical hyperparameters for both networks. To ensure a fair comparison, we trained both networks using the composite loss function Eq. (5.4.2) with weights $\alpha_1 = 0.1$, $\alpha_2 = 0.001$, and $\alpha_3 = 1.0$. Fig. 5.10 shows various metrics versus epochs during the training process, and we report all performance measurements by averaging 10 independent runs. The results indicate that NBA with NCSC outperforms NBA with CSC in estimating abundance due to the ANC and ASC constraints.

Table 5.4: Impact of the number of kernels.

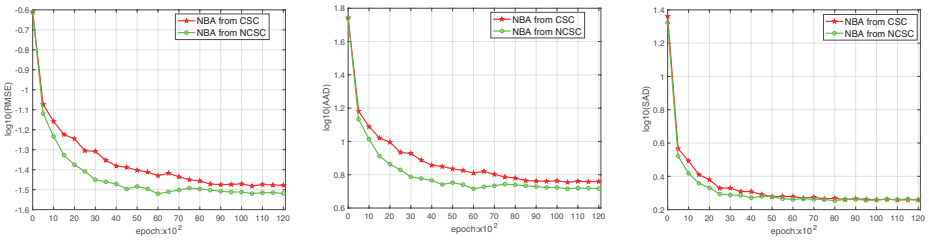
RMSE/AAD/SAD		UADIP number of kernels m_A			
		32	64	128	256
UEDIP number of kernels m_E	32	0.0406/7.1662/1.868	0.0293/5.1246/**1.740**	**0.0270**/**4.7051**/1.795	0.0295/5.0537/1.777
	64	0.0301/5.3076/1.782	0.0292/5.0791/1.780	0.0286/4.9127/1.811	0.0281/4.8197/1.755
	128	0.0331/5.7963/1.816	0.0436/7.7979/2.269	0.0271/4.7193/1.783	0.0296/5.1193/**1.740**
	256	0.2403/53.5809/21.983	0.2309/50.7145/22.613	0.2320/50.9685/19.264	0.2368/52.4924/22.325

Table 5.5: Comparison between normal DIP networks and unfolding based DIP networks.

unfolding model	network	#learnable parameters	Abundance estimation metric: RMSE	Abundance estimation metric: AAD	Endmember estimation metric: SAD
-	BUDDIP	4.6×10^5	0.0323	5.5347	1.8611
LMM	LA-BUDDIP	7.40×10^5	0.1279	26.0893	3.2869
MCU	NBA	7.89×10^5	**0.0296**	**5.0919**	**1.7911**

Table 5.6: Comparison between DIP networks with and without RED regulariser.

network		Abundance estimation metric: RMSE	Abundance estimation metric: AAD	Endmember estimation metric: SAD
BU task	BUDDIP	0.0323	5.5347	1.8611
	BUDDIP+RED	0.0306	5.2572	1.8045
	NBA	0.0296	5.0919	1.7911
	NBA+RED	**0.0266**	**4.6294**	**1.6023**



(a) RMSE versus epoch (b) AAD versus epoch (c) SAD versus epoch

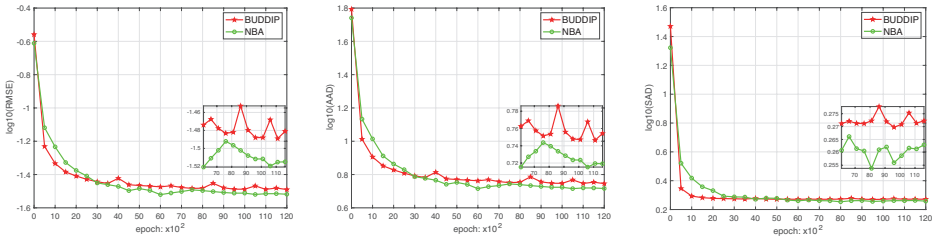
Figure 5.10: The impact of NCSC Eq. (5.1.9) on blind unmixing performance. The metrics are drawn with the corresponding log value. (a) Abundance RMSE versus epoch. (b) Abundance AAD versus epoch. (c) Endmember SAD versus epoch.

Comparison between BUDDIP and NBA

We aim to evaluate the effectiveness of the proposed unfolding structure by comparing the performance of two blind linear unmixing networks: BUDDIP, a classical Resnet-like network, and NBA, an unfolding-based network. We generated the training guidance for both networks, denoted as $\mathbf{E}_G, \mathbf{A}_G$, using SiVM+FCLS, and set the hyperparameters of NBA and BUDDIP to their default values. To ensure a fair comparison and reduce the computational burden of hyperparameter tuning, we train both BUDDIP and NBA using only the MSE-type loss terms. Specifically, BUDDIP is trained with the composite loss function Eq. (4.3.8) where the weights are set as $\alpha_1 = 0.1$, $\alpha_3 = 0.01$, and $\alpha_2 = \alpha_4 = \alpha_6 = 0$, $\alpha_5 = 1.0$. Meanwhile, NBA is trained with the composite loss function Eq. (5.4.2) where the weights are set as $\alpha_1 = 0.1$, $\alpha_2 = 0.001$, and $\alpha_3 = 1.0$. The training process, including various metrics versus epochs, is depicted in Fig. 5.11, and the final performance is summarized in Table 5.5. All the reported performances are reported by averaging 10 independent runs.

Our results indicate that although BUDDIP converges faster, NBA outperforms BUDDIP in terms of final estimation quality. Specifically, the RMSE of abundance estimation drops from 0.0323 to 0.0296, and the SAD of endmember estimation decreases from 1.8611 to 1.7911. Additionally, NBA has 7.89×10^5 learnable parameters, whereas BUD-

DIP has 4.6×10^5 learnable parameters. In conclusion, the proposed unfolding structure achieves better performance with a similar number of learnable parameters compared to the general Resnet-like structure.



(a) RMSE versus epoch (b) AAD versus epoch (c) SAD versus epoch

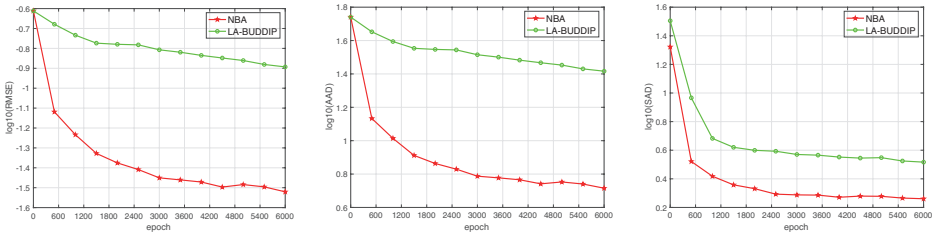
Figure 5.11: The impact of unfolding on blind unmixing performance. The metrics are drawn with the corresponding log value. (a) Abundance RMSE versus epoch. (b) Abundance AAD versus epoch. (c) Endmember SAD versus epoch.

The Effect of MCU model

In this experiment, we evaluate the effectiveness of the proposed MCU unmixing model by comparing NBA with LA-BUDDIP, where the former is derived from the MCU model and the latter from the LMM model. Our evaluation involves comparing the performance of blind linear unmixing between LA-BUDDIP and NBA, with both BU networks' training guidance, represented by $\mathbf{E}_G, \mathbf{A}_G$, generated through SiVM+FCLS. We set the hyper-parameters of NBA to their default values, while LA-BUDDIP is constructed such that the number of learnable parameters is at the same level as NBA. We also train both networks using the ADAM optimiser to minimise the loss function Eq. (5.4.2). The results are depicted in Fig. 5.12, and the final performance is summarised in Table 5.5.

Our results indicate that NBA has better performance and faster convergence than LA-BUDDIP. Specifically, the SAD of endmember estimation for NBA is around 1.79, while that for LA-BUDDIP is 3.287. The RMSE of abundance estimation for NBA and LA-BUDDIP is around

0.0296 and 0.128, respectively. This improvement can be attributed to the network architecture derived from the MCU unmixing model.



(a) RMSE versus epoch (b) AAD versus epoch (c) SAD versus epoch

Figure 5.12: Blind unmixing performance comparison between unfolding MCU model based network (NBA) and unfolding LMM based network (LA-BUDDIP). The metrics are drawn with the corresponding log value. (a) Abundance RMSE versus epoch. (b) Abundance AAD versus epoch. (c) Endmember SAD versus epoch.

5.7.3 Effectiveness of Explicit Regularisations

Hyperparameter Tuning

In this experiment, we sought to investigate the impact of hyperparameters α_1 and α_2 on the performance of NBA+RED, which determine the relative importance of L_{EDIP} and L_{ADIP} loss terms in the loss function Eq. (5.6.3). By default, we use $\mu_E = \mu_A = 0.1, \alpha_4 = \alpha_5 = 0.01$. To this end, we kept α_3 fixed at 1 and varied α_1 and α_2 in the range of $[0.1, 0.01, 0.001]$, while keeping all other settings at their default values. The results are summarised in Table 5.7. It indicates that as α_1 increases, the network's endmember estimation improves overall in terms of SAD. Conversely, when α_1 is less than 0.1, an increase in α_2 leads to a gradual improvement in abundance estimation. However, when α_1 reaches 0.1, the opposite trend is observed. This may be attributed to the fact that both disregarding and exaggerating guidance can result in poor unmixing. The optimal unmixing performance can be achieved by appropriately adjusting the loss weight, such as with $\alpha_1 = 0.1$ and $\alpha_2 = 0.001$. These findings also suggest that the proposed method is sensitive to hyperparameters.

Table 5.7: Hyperparameter tuning.

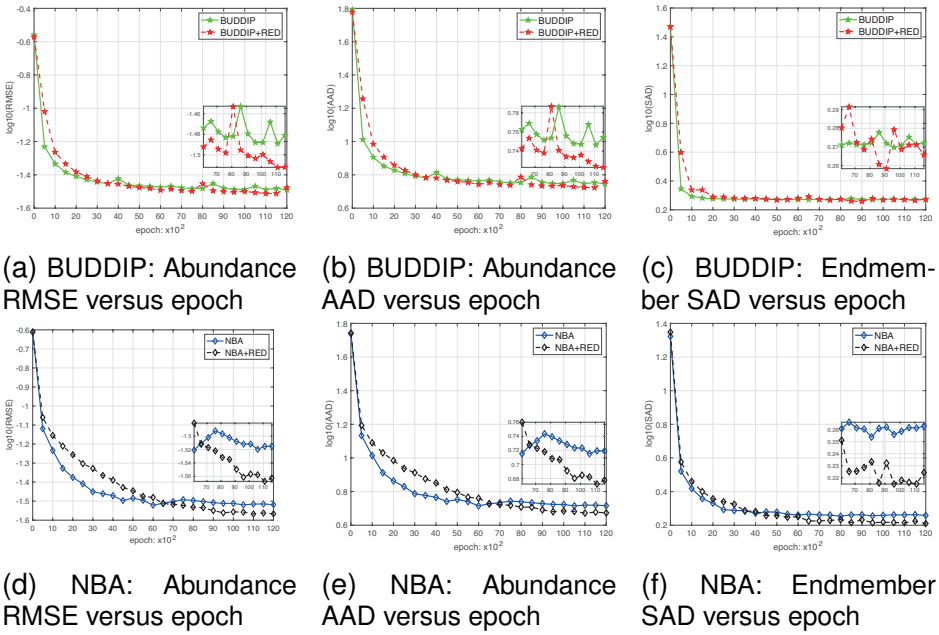
α_1	α_2	α_3	RMSE	AAD	SAD
0.001	0.001	1	0.2442	54.9031	21.1657
0.001	0.01	1	0.2387	53.3415	22.3389
0.001	0.1	1	0.23	50.4799	23.9072
0.01	0.001	1	0.2382	52.9666	22.7712
0.01	0.01	1	0.2285	50.1863	20.7052
0.01	0.1	1	0.2256	48.9509	22.4393
0.1	0.001	1	$\mathbf{0.0236}$	$\mathbf{4.1299}$	$\mathbf{1.4854}$
0.1	0.01	1	0.0268	4.5942	1.7046
0.1	0.1	1	0.2075	43.7296	19.7209

BUDDIP/NBA with and without Explicit Regularisations

We aim to assess the effectiveness of explicit regularizers by comparing the performance of BUDDIP with and without RED, as well as that of NBA with and without RED, using the same synthetic data as presented in Sec. 5.7.2. The following hyperparameters were set: BUDDIP was trained with $\alpha = 0.1, \alpha_2 = 0.01, \alpha_3 = 1.0$ in loss Eq. (5.4.2) and ADAM optimizer with a learning rate of $5e-3$; for BUDDIP with RED, we used $\alpha_1 = 0.1, \alpha_2 = 0.01, \alpha_3 = 1.0, \alpha_4 = 0.56, \alpha_5 = 0.33, \mu_E = 0.318, \mu_A = 0.9$ in objective Eq. (5.6.3) and ADAM optimizer with a learning rate of $5e-3$; for NBA, we used $\alpha = 0.1, \alpha_2 = 0.001, \alpha_3 = 1.0$ in objective Eq. (5.4.2) and ADAM optimizer with a learning rate of $1e-3$; for NBA with RED, we used $\alpha_1 = \mu_E = \mu_A = 0.1, \alpha_2 = 0.001, \alpha_4 = \alpha_5 = 0.01, \alpha_3 = 1.0$ in objective Eq. (5.6.3) and ADAM optimizer with a learning rate of $1e-3$. All networks were trained with 12000 epochs. We summarize the results in Table 5.6 and present the performance metrics versus training epochs in log values in Fig. 5.13.

The results demonstrate that the addition of RED regularizers significantly improves the performance of both BUDDIP and NBA networks. For instance, in the NBA, the AAD of abundance estimation improves from 5.0919 to 4.6294, and the SAD of endmember estima-

tion improves from 1.79 to 1.60. We attribute this improvement to the explicit regularizers added to the networks. Similar performance improvements are also observed in BUDDIP+RED when compared with BUDDIP.



(a) BUDDIP: Abundance RMSE versus epoch

(b) BUDDIP: Abundance AAD versus epoch

(c) BUDDIP: Endmember SAD versus epoch

(d) NBA: Abundance RMSE versus epoch

(e) NBA: Abundance AAD versus epoch

(f) NBA: Endmember SAD versus epoch

Figure 5.13: The impact of RED on blind unmixing performance. The metrics are drawn with the corresponding log value. (a) BUDDIP: Abundance RMSE versus epoch. (b) BUDDIP: Abundance AAD versus epoch. (c) BUDDIP: Endmember SAD versus epoch. (d) NBA: Abundance RMSE versus epoch. (e) NBA: Abundance AAD versus epoch. (f) NBA: Endmember SAD versus epoch.

5.8 Summary

This chapter presents several variations of BUDDIP to further enhance the performance. Firstly, we propose a novel neural network architecture for hyperspectral unmixing problems. Unlike the general linear mixture model (LMM), we propose a MatrixConv Unmixing (MCU) model and corresponding ADMM solvers. By employing the algorithm unrolling technique for each solver, we construct unfolding-based deep image prior networks for endmember estimation (UEDIP) and abundance estimation (UADIP). These two networks are combined based on LMM to create our final network for blind unmixing using ADMM un-

folding, which we refer to as NBA. Additionally, we propose the explicit inclusion of two regularizers for endmember and abundance estimation using regularization by denoising (RED) techniques. Experimental results have further showcased the efficacy of the proposed methods. However, in contrast to vanilla BUDDIP, all of these variations necessitate a greater number of training epochs.

Chapter 6

Comparison of Methods on Real Data

In this chapter, we aim to evaluate the effectiveness of our proposed methods by comparing them against some of the current state-of-the-art unmixing techniques using real hyperspectral imaging datasets. To perform this comparison, we will utilize the performance metrics discussed in Sec. 3.5.1, which include RMSE and AAD for abundance estimation, as well as SAD for endmember estimation.

6.1 Data

To evaluate the performance of various unmixing algorithms, We use three commonly used real HSI datasets.

1. JASPER RIDGE. The first real dataset is the well-known Jasper Ridge [107]. This dataset features 512×614 pixels with a range of 224 spectral bands from 380 nm to 2500 nm, and a spectral resolution of 9.46 nm. The four endmembers present in the scene are Road, Soil, Water, and Tree. To reduce the computational burden and facilitate faster experimentation, we consider a 100×100 pixels sub-image of the original image. To account for the effects of dense water vapor and atmosphere, we eliminate

26 spectral bands, leaving only 198 out of the 224 for unmixing. A representative image of the 80th spectral band is depicted in Fig. 6.1a.

2. URBAN. The Urban dataset [16, 108] is a widely used hyperspectral dataset in studies of hyperspectral unmixing. The image comprises 307×307 pixels, with each pixel representing an area of 2×2 square meters. The dataset includes 210 wavelengths, spanning from 400 nm to 2500 nm, resulting in a spectral resolution of 10 nm. However, due to the presence of dense water vapor and atmospheric effects, several channels (1-4, 76, 87, 101-111, 136-153, and 198-210) were removed, resulting in a total of 162 channels. This is a standard preprocessing step for hyperspectral unmixing analyses. The ground truth for the Urban dataset is available in three different versions, each with a different number of endmembers. For our study, we used the version with six endmembers, which were identified as Asphalt, Grass, Tree, Roof, Metal, and Dirt, respectively. The Urban HSI image associated with the 80th channel is depicted in Fig. 6.1b.

3. SAMSON. The Samson dataset is another real dataset that we utilized in this study. This dataset comprises of an image with a resolution of 952×952 pixels, each pixel captured at 156 different channels covering a range of wavelengths from 401 nm to 889 nm. The spectral resolution is high, with a value of 3.13 nm. Since the original image is quite large, computations would have been expensive. Therefore, a region of 95×95 pixels has been selected, starting at the (252,332) pixel of the original image. The image consists of three distinct endmembers, namely Soil, Tree, and Water. Fig. 6.1c demonstrates the Samson dataset at the 80th channel.

(a) Jasper Ridge (b) Urban (c) Samson

Figure 6.1: HSI image at 80th channel. (a) Jasper Ridge. (b) Urban. (c) Samson.

6.2 Evaluation on real data

We now evaluate the unmixing performance of various methods on various real datasets. The unmixing methods used in this comparison include the traditional unmixing methods SiVM [1]+FCLS [2], rNMF [13], HyperCSI [96], HiSun [62], EDAA [52] and learning based unmixing networks MNN-BU-2 [18], EGU-Net-ss [4], UnDIP [36], CNNAEU [100], MiSiCNet [101] and the proposed UBUNet-II, L-BUDDIP, NL-BUDDIP and NBARED. In this study, we utilized SiVM+FCLS as the default generator for techniques like MNN-BU-2, EGU-Net-ss, UnDIP, and MiSiCNet, which require initialization or guidance. To demonstrate the effectiveness of our proposed methods, we initially trained L-BUDDIP with loss function Eq. (4.3.8) and NBARED with Algorithm 2, using guidance generated from SiVM+FCLS. Following this, we further trained both L-BUDDIP and NL-BUDDIP to minimize the loss function Eq. (4.3.8) by utilizing guidance from EDAA, resulting in state-of-the-art performance. It's important to note that the MM module utilized in the NL-BUDDIP was employed as the FM model.

JasperRidge

We first evaluate the performance on Jasper Ridge. Experiment settings are summarised in Table 6.1. The qualitative results of estimated endmembers and abundances are shown in Fig. 6.2 and Fig. 6.3, respectively. The corresponding quantitative results are shown in Ta-

ble 6.4. Fig. 6.2 presents the results of various unmixing methods on the Jasper Ridge dataset using guidances/initialization generated from SiVM+FCLS. Methods such as SiVM+FCLS, CNNAEU, UnDIP, EGU-Net, MNN-BU, and UBUNet failed to accurately estimate the signatures of the road. Additionally, HyperCSI and HiSun were unsuccessful in unmixing the signature of Water, while MiSiCNet did not yield meaningful unmixing results in this dataset.

In contrast, Table 6.4 shows that the proposed UBU-Net, L-BUDDIP, and NBARED methods surpassed the guidance SiVM+FCLS in terms of abundance AAD and endmember SAD. Notably, L-BUDDIP achieved an abundance AAD of 16.24 and an endmember SAD of 8.15, while UBUNet-II achieved an abundance AAD of 17.60 and an endmember SAD of 8.0. Similarly, NBARED produced an abundance AAD of 14.4 and an endmember SAD of 5.36. Conversely, SiVM+FCLS produced an abundance AAD of 20.72 and an endmember SAD of 11.35, demonstrating the effectiveness of the proposed methods.

Furthermore, when using guidance generated from EDAA, the proposed L-BUDDIP and NL-BUDDIP produced the most visually appealing unmixing results. The quantitative results in Table 6.4 also show that the proposed L-BUDDIP with guidance from EDAA had the best SAD for the road endmember and the overall average SAD among all endmembers. With the improved endmember estimation, the proposed L-BUDDIP achieved state-of-the-art abundance estimation in terms of RMSE and AAD. The proposed NL-BUDDIP also performed slightly better than L-BUDDIP in terms of abundance estimation.

While EDAA delivered very good unmixing results with an AAD of 7.66, the proposed BUDDIP method further improved the AAD to 5.20. The qualitative results illustrated in Fig. 6.3 indicate that the proposed BUDDIP with guidance from EDAA generated the best abundance estimation compared to the competitors.

Table 6.1: Experiment settings for Jasper Ridge.

method	guidance	training strategy	hyperparameter
L-BUDIP	SiVM+FCLS	loss function Eq. (4.3.8)	$\alpha_1 = 45.25, \alpha_2 = 100, \alpha_3 = 16.60,$ $\alpha_4 = 47.16, \alpha_5 = 1.0, \alpha_6 = 0.08$
NBARED	SiVM+FCLS	Algorithm 2	$\alpha_1 = 0.49, \alpha_2 = 0.49, \alpha_3 = 1.0, \alpha_4 = 0.018,$ $\alpha_5 = 0.536, \mu_E = 0.688, \mu_A = 0.228$
L-BUDIP	EDAA	loss function Eq. (4.3.8)	$\alpha_1 = \alpha_3 = \alpha_6 = 0.01,$ $\alpha_2 = \alpha_4 = 100, \alpha_5 = 1.0$
NL-BUDDIP	EDAA	algorithm 1	$\alpha_1^{init} = \alpha_3^{init} = \alpha_4^{init} = 10,$ $\alpha_2^{init} = 100, \alpha_5^{init} = 0.1, \alpha_6^{init} = 0.01,$ $\gamma_1 = \gamma_2 = 0.9, \alpha_{min} = 0.01, \alpha_{max} = 100, g = 700$

method	net optimiser	learning rate	epoch
L-BUDIP	Adam	0.005	6000
NBARED	Adam	0.0005	6000
L-BUDIP	Adam	0.005	6000
NL-BUDDIP	Adam	0.005	6000

Table 6.2: Endmembers estimated by different methods on the Jasper Ridge dataset. Blue solid lines indicate the true value, while red dot lines indicate the scaled estimated value. From top to bottom: Tree, Water, Soil, and Road. (a) SiVM+FCLS. (b) MNN-BU-2. (c) UBUNet-II. (d) EGU-Net-ss. (e) UnDIP. (f) L-BUDDIP with EDAA. (g) rNMF. (h) NL-BUDDIP with EDAA. (i) HyperCSI. (j) HiSun. (k) EDAA. (l) MiSICNet. (m) CNNAEU. (n) L-BUDDIP with SiVM+FCLS. (o) NBARED with SiVM+FCLS.

Table 6.3: Abundance maps estimated by different methods on the Jasper Ridge dataset. From top to bottom: Tree, Water, Soil, and Road. (a) SiVM+FCLS. (b) MNN-BU-2. (c) UBUNet-II. (d) EGU-Net-ss. (e) UnDIP. (f) L-BUDDIP with EDAA. (g) rNMF. (h) NL-BUDDIP with EDAA. (i) HyperCSI. (j) HiSun. (k) EDAA. (l) MiSICNet. (m) CNNAEU. (n) L-BUDDIP with SiVM+FCLS. (o) NBARED with SiVM+FCLS. (p) Reference.

Table 6.4: Mean and Standard Deviation of abundance RMSE, AAD (In Degrees), endmember SAD (In Degrees) by Different methods on the Jasper Ridge. The Best Results are in Bold. The second best are denoted with a star.

Methods	RMSE	AAD	SAD of Tree	SAD of Water	SAD of Soil	SAD of Road	averaged SAD
HyperCSI	0.2005±0.0000	20.7649±0.0000	5.4103±0.0000	44.3726±0.0000	7.5442±0.0000	6.6429±0.0000	15.9925±0.0000
HiSun	0.3152±0.0000	46.0728±0.0000	14.4662±0.0000	54.6580±0.0000	10.3029±0.0000	5.8153±0.0000	21.3106±0.0000
SiVM+FCLS	0.1480±0.0000	20.7198±0.0000	8.5545±0.0000	14.4877±0.0000	6.5558±0.0000	15.7991±0.0000	11.3493±0.0000
MNN-BU-2	0.2154±0.0007	33.1936±0.0927	7.9771±0.0046	14.4324±0.0117	5.3219±0.0367	8.6187±0.0020	9.0875±0.0117
UBUNet-II	0.1332±0.0022	17.6087±0.3170	8.6635±0.0160	4.0919±0.1313	5.6216±0.0113	13.6586±0.2183	8.0089±0.0417
EGU-Net-ss	0.2110±0.0019	30.7221±0.2117	**2.7626±0.1517**	4.5969±0.5298	5.6690±0.0646	22.0731±0.3154	8.7754±0.0242
UnDIP	0.1748±0.0252	25.3249±4.4570	8.5545±0.0000	14.4877±0.0000	6.5558±0.0000	15.7991±0.0000	11.3493±0.0000
L-BUDDIP (EDAA)	0.0445±0.0005*	5.2348±0.1152*	4.2622±0.0335	2.9074±0.0209	2.7430±0.0206*	**2.8273±0.0463**	**3.1850±0.0045**
rNMF	0.1750±0.0002	25.2385±0.0233	8.7639±0.0098	16.1319±0.0122	25.8640±0.0630	32.0590±0.0210	20.7047±0.0148
NL-BUDDIP (EDAA)	**0.0434±0.0005**	**5.1956±0.0722**	4.3011±0.0489	2.8836±0.0352*	2.7488±0.0228	2.8532±0.0681*	3.1966±0.0078*
MiSiCNet	0.1894±0.0000	27.4191±0.0015	**2.4860±0.0010**	16.5820±0.0069	10.0646±0.0007	9.3605±0.0018	9.6233±0.0015
CNNAEU	0.2726±0.0860	36.6164±12.9452	3.8993±0.5724*	4.3835±1.2212	5.0614±1.3909	31.3367±16.9217	11.1702±4.0669
EDAA	0.0583±0.0000	7.6572±0.0000	4.2205±0.0000	**2.8038±0.0000**	**2.7262±0.0000**	3.1506±0.0000	3.2253±0.0000
L-BUDDIP (SiVM+FCLS)	0.1238±0.0003	16.2442±0.0447	8.7884±0.0139	13.2621±0.0064	6.4180±0.0212	4.1317±0.0013	8.1501±0.0036
NBARED (SiVM+FCLS)	0.1111±0.0017	14.3978±0.2366	7.9002±1.9212	5.8873±0.4233	4.4598±1.7742	3.1890±0.8697	5.3591±0.3571

Urban

We then evaluate the performance using the Urban dataset with 6 endmembers. The hyperparameters for various methods are summarised in Table 6.5. The qualitative results of estimated endmembers and abundances are shown in Fig. 6.6 and Fig. 6.7, respectively. The corresponding quantitative results are summarised in Table 6.8. The results of EGU-Net-ss algorithm are not reported as it is not compatible with this dataset. The quantitative results indicate that the EDAA algorithm outperforms the competing methods, with an abundance AAD of 21.44 and an endmember SAD of 7.914. In contrast, HyperCSI and HiSun achieve higher AADs of 60.48 and 34.779, respectively, with HiSun achieving a similar endmember SAD of 8.01 to EDAA. The guidance generator SiVM+FCLS produces an abundance AAD of 52 and endmember SAD of 39, while UBUNet-II performs even worse with an abundance AAD of 65.7 and endmember SAD of 39.65 due to the lack of training guidance, which can lead to physically meaningless unmixing results. However, the proposed L-BUDDIP and NBARED methods outperform SiVM+FCLS, with improved abundance AADs of 40.96 and 47.67, respectively, and improved endmember SADs of 14.48 and 27.95, respectively.

Furthermore, when using guidance generated from EDAA, the proposed NL-BUDDIP and L-BUDDIP methods achieve the best and second-best abundance AADs of 17.89 and 17.905, respectively. They also produce the second-best and best average endmember SADs of 7.549 and 6.993, respectively. The qualitative results in Fig. 6.6 and Fig. 6.7 demonstrate that the proposed BUDDIP method with guidance from EDAA delivers state-of-the-art unmixing results.

Table 6.5: Experiment settings for Urban.

method	guidance	training strategy	hyperparameter
L-BUDIP	SiVM+FCLS	loss function Eq. (4.3.8)	$\alpha_1 = 0.01, \alpha_2 = 0.1, \alpha_3 = 1.0,$ $\alpha_4 = \alpha_5 = \alpha_6 = 1e-5$
NBARED	SiVM+FCLS	Algorithm 2	$\alpha_1 = 1e-4, \alpha_2 = 1e-2, \alpha_3 = 1.0, \alpha_4 = 1e-5,$ $\alpha_5 = 1e-4, \mu_E = \mu_A = 1e-4$
L-BUDIP	EDAA	loss function Eq. (4.3.8)	$\alpha_1 = \alpha_6 = 0.1, \alpha_2 = 100,$ $\alpha_3 = 0.01, \alpha_4 = 10, \alpha_5 = 1.0$
NL-BUDDIP	EDAA	algorithm 1	$\alpha_1^{init} = \alpha_3^{init} = \alpha_4^{init} = 100, \alpha_2^{init} = 10,$ $\alpha_5^{init} = 1.0, \alpha_6^{init} = 0.01, \gamma_1 = \gamma_2 = 0.8,$ $\alpha_{min} = 0.001, \alpha_{max} = 100, g = 100$

method	net optimiser	learning rate	epoch
L-BUDIP	Adam	0.005	6000
NBARED	Adam	0.0001	6000
L-BUDIP	Adam	0.005	6000
NL-BUDDIP	Adam	0.005	6000

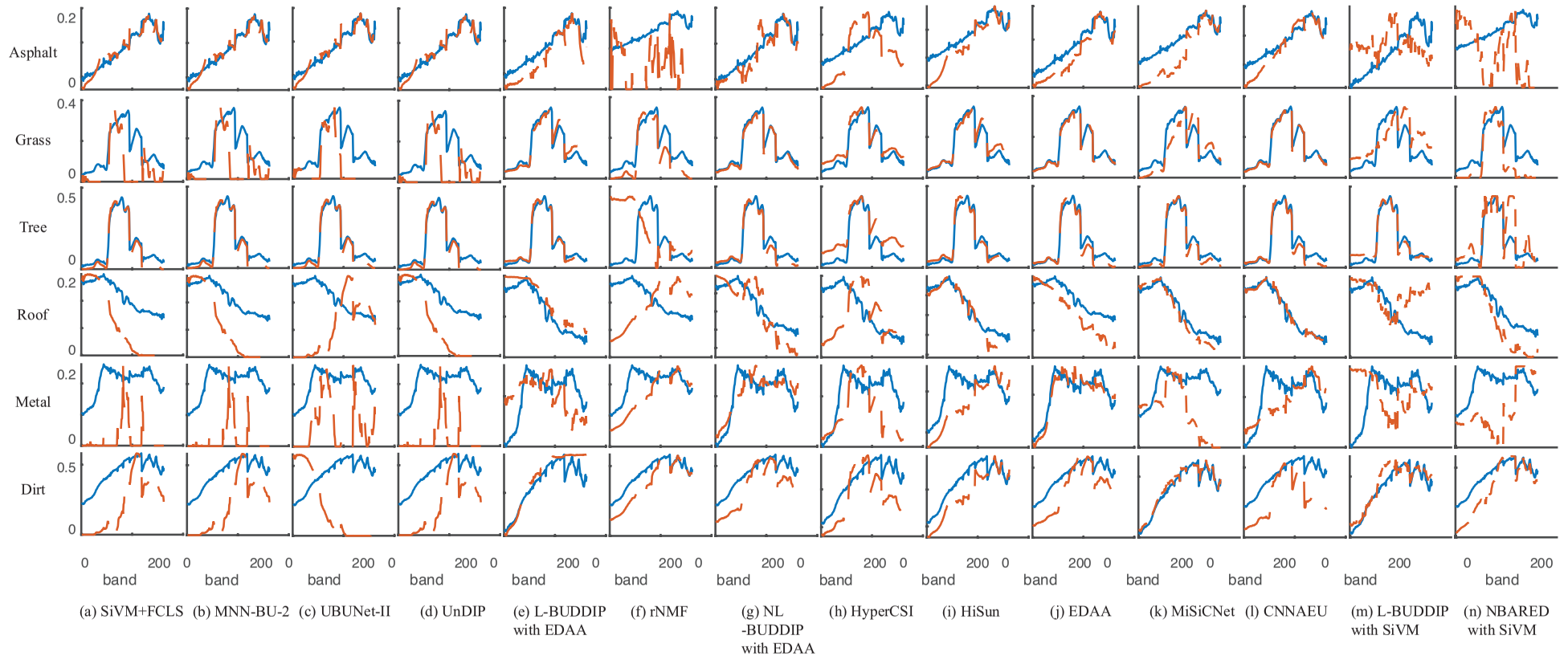


Table 6.6: Endmembers estimated by different methods on Urban dataset. Blue solid lines indicate the true value, while red dot lines indicate the scaled estimated value. From top to bottom: Asphalt, Grass, Tree, Roof, Metal, and Dirt. (a) SiVM+FCLS. (b) MNN-BU-2. (c) UBUNet-II. (d) UnDIP. (e) L-BUDDIP with EDAA. (f) rNMF. (g) NL-BUDDIP with EDAA. (h) HyperCSI. (i) HiSun. (j) EDAA. (k) MiSiCNet. (l) CNNAEU. (m) L-BUDDIP with SiVM+FCLS. (n) NBARED with SiVM+FCLS.

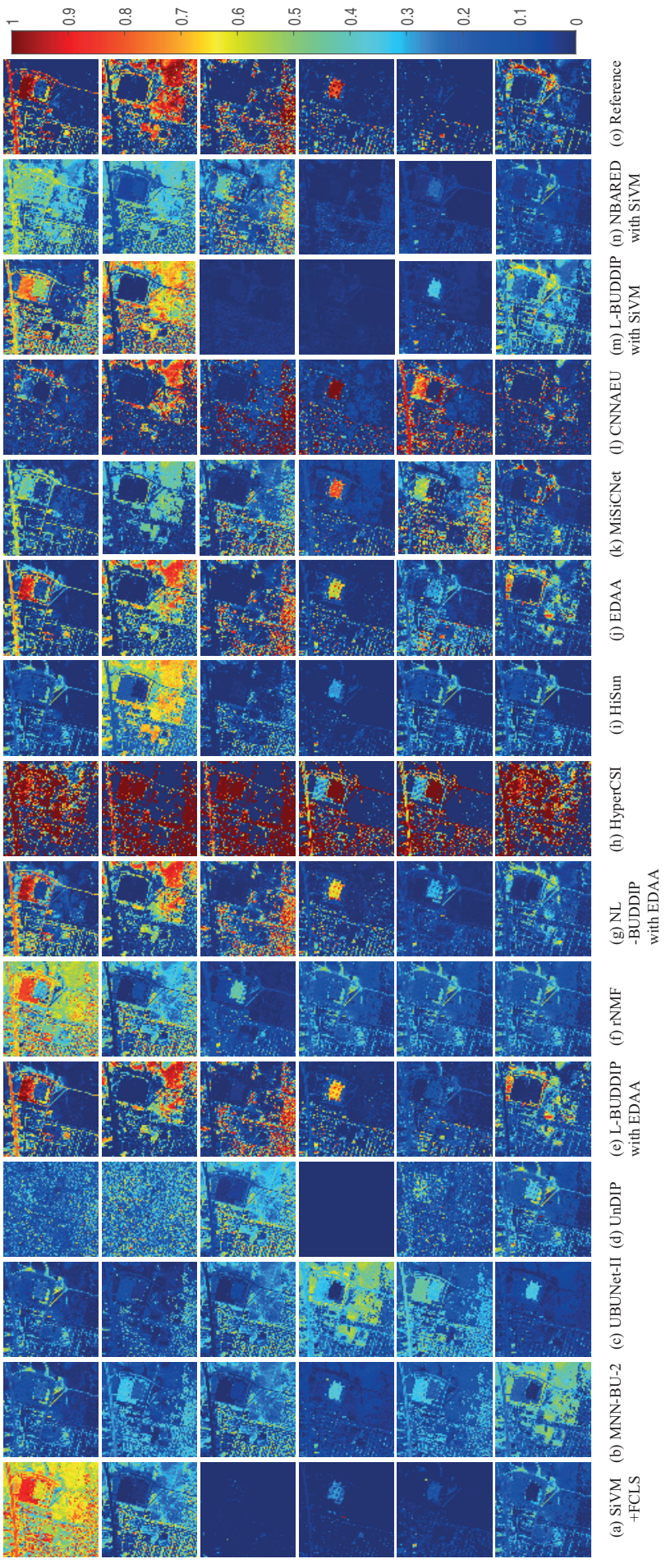


Table 6.7: Abundance maps estimated by different methods on Urban Ridge dataset. From top to bottom: Asphalt, Grass, Tree, Roof, Metal, and Dirt. (a) SiVM+FCLS. (b) MNN-BU-2. (c) UBUNet-II. (d) UnDIP. (e) L-BUDDIP with EDAA. (f) rNMF. (g) NL-BUDDIP with EDAA. (h) HyperCSI. (i) HiSun. (j) EDAA. (k) MiSiCNet. (l) CNNAEU. (m) L-BUDDIP with SiVM+FCLS. (n) NBARED with SiVM+FCLS. (o)Reference.

Table 6.8: Mean and Standard Deviation of abundance RMSE, AAD (In Degrees), endmember SAD (In Degrees) by Different methods on Urban. The Best Results are in Bold. The second best are denoted with a star.

Methods	RMSE	AAD	SAD of Asphalt	SAD of Grass	SAD of Tree	SAD of Roof	SAD of Metal	SAD of Dirt	averaged SAD
HyperCSI	1.143±0.000	60.480±0.000	18.545±0.000	7.449±0.000	16.886±0.000	20.102±0.000	14.695±0.000	13.142±0.000	15.137±0.000
HiSun	0.200±0.000	34.779±0.000	7.900±0.000	7.146±0.000	**6.145±0.000**	5.924±0.000	12.810±0.000	8.135±0.000	8.010±0.000
SiVM+FCLS	0.261±0.000	52.148±0.000	52.359±0.000	61.106±0.000	6.225±0.000*	45.861±0.000	50.271±0.000	20.044±0.000	39.311±0.000
MNN-BU-2	0.282±0.000	56.894±0.166	**3.725±0.007**	47.075±0.784	8.764±0.079	33.290±0.076	67.682±2.401	32.844±0.065	32.230±0.281
UBUNet-II	0.312±0.008	65.723±1.551	4.073±0.196*	44.272±3.625	8.779±0.353	56.795±0.938	63.475±12.079	60.558±0.857	39.658±2.492
UnDIP	0.276±0.005	55.130±1.472	52.359±0.000	61.106±0.000	6.225±0.000	45.861±0.000	50.271±0.000	20.044±0.000	39.311±0.000
L-BUDDIP (EDAA)	0.103±0.002*	17.905±0.311*	6.322±0.806	3.645±1.679	7.624±1.746	6.829±2.484	7.234±4.164	10.305±3.389	**6.993±0.426**
rNMF	0.303±0.000	57.584±0.032	48.095±0.155	21.276±0.022	63.091±0.118	35.506±0.040	17.851±0.040	10.957±0.047	32.796±0.027
NL-BUDDIP (EDAA)	**0.102±0.001**	**17.893±0.097**	5.143±0.229	3.501±0.512	8.316±0.127	10.327±0.319	6.251±0.078*	11.756±0.586	7.549±0.026*
MiSiCNet	0.2050±0.0001	36.2517±0.0106	12.077±0.021	16.474±0.017	7.031±0.012	3.804±0.011*	35.807±0.101	**1.570±0.005**	12.794±0.018
CNNAEU	0.234±0.036	41.703±6.342	8.143±6.689	2.191±0.436*	8.067±0.223	**2.388±0.276**	16.700±6.246	9.694±5.359	7.864±0.500
EDAA	0.119±0.000	21.440±0.000	4.848±0.000	**2.174±0.000**	8.917±0.000	13.709±0.000	**4.524±0.000**	13.313±0.000	7.914±0.000
L-BUDDIP (SiVM+FLCS)	0.2234±0.0016	40.9626±0.2465	11.9344±2.7176	19.9144±7.5690	22.6421±21.6879	11.2254±3.3931	17.0729±2.7203	3.8077±0.8625*	14.4818±1.2602
NBARED (SiVM+FLCS)	0.2476±0.0130	47.6669±3.0343	35.1673±12.8577	23.9474±8.5359	32.6751±17.1940	26.7588±17.2913	37.5044±7.0130	11.6671±4.4366	27.9533±3.3695

Samson

We then evaluate the performance using the Samson dataset. The hyperparameters of various methods are summarised in Table 6.9. The qualitative results of estimated endmembers and abundances are shown in Fig. 6.10 and Fig. 6.11, respectively. The corresponding quantitative results are reported in Table 6.12. The results of EGU-Net-ss algorithm are not reported as it is not compatible with this dataset. It is clear that the EDAA algorithm can readily generate the best unmix among the competitors, with an abundance RMSE of 0.0232, AAD of 2.6365 and endmember SAD of 1.5091. In comparison, Hyper-CSI, HiSun, MiSiCNet and CNNAEU show an abundance AAD over 15 and endmember SAD over 8.9. However, when the unmixing results of EDAA are used as guidance for the proposed methods, the proposed L-BUDDIP can deliver further improved abundance RMSE and AAD, which are 0.0145 and 1.4957, respectively. Similarly, the proposed NL-BUDDIP produces the best endmember SAD of 1.4287. On the other hand, when using guidance from SiVM+FCLS, the proposed L-BUDDIP and NBARED can also produce improved performance. Specifically, L-BUDDIP achieves an abundance AAD of 13.68 and endmember SAD of 2.45 and NBARED achieves an abundance AAD of 26.30 and endmember SAD of 2.87. In comparison, the guidance SiVM+FCLS achieves an abundance AAD of 34.67 and an endmember SAD of 3.47. The qualitative results shown in Fig. 6.10 and Fig. 6.11 also illustrate the proposed methods with guidance from EDAA achieve the most visually appealing results.

6.3 Summary

In this chapter, we evaluated the performance of our proposed interpretable unmixing networks UBUNet, as well as the general unmix-

Table 6.9: Experiment settings for Samson.

method	guidance	training strategy	hyperparameter
L-BUDIP	SiVM+FCLS	loss function Eq. (4.3.8)	$\alpha_1 = 93.44, \alpha_2 = 52.18, \alpha_3 = 21.66,$ $\alpha_4 = 11.09, \alpha_5 = 1.0, \alpha_6 = 9.88$
NBARED	SiVM+FCLS	Algorithm 2	$\alpha_1 = 0.11, \alpha_2 = 0.31, \alpha_3 = 1.0, \alpha_4 = 0.086,$ $\alpha_5 = 0.483, \mu_E = 0.138, \mu_A = 0.408$
L-BUDIP	EDAA	loss function Eq. (4.3.8)	$\alpha_1 = 0.01, \alpha_2 = \alpha_3 = \alpha_5 = 1.0, \alpha_4 = 100, \alpha_6 = 0.1$
NL-BUDDIP	EDAA	algorithm 1	$\alpha_1^{init} = \alpha_2^{init} = \alpha_4^{init} = 10, \alpha_3^{init} = 100,$ $\alpha_5^{init} = 1.0, \alpha_6^{init} = 0.01, \gamma_1 = 0.9, \gamma_2 = 0.8,$ $\alpha_{min} = 0.001, \alpha_{max} = 100, g = 500$

method	net optimiser	learning rate	epoch
L-BUDIP	Adam	0.005	6000
NBARED	Adam	0.0005	6000
L-BUDIP	Adam	0.005	6000
NL-BUDDIP	Adam	0.005	6000

Table 6.10: Endmembers estimated by different methods on Samson dataset. Blue solid lines indicate the true value, while red dot lines indicate the scaled estimated value. From top to bottom: Soil, Tree and Water. (a) SiVM+FCLS. (b) MNN-BU-2. (c) UBUNet-II. (d) UnDIP. (e) L-BUDDIP with EDAA. (f) rNMF. (g) NL-BUDDIP with EDAA. (h) HyperCSI. (i) HiSun. (j) EDAA. (k) MiSiCNet. (l) CNNAEU. (m) L-BUDDIP with SiVM+FCLS. (n) NBARED with SiVM+FCLS.

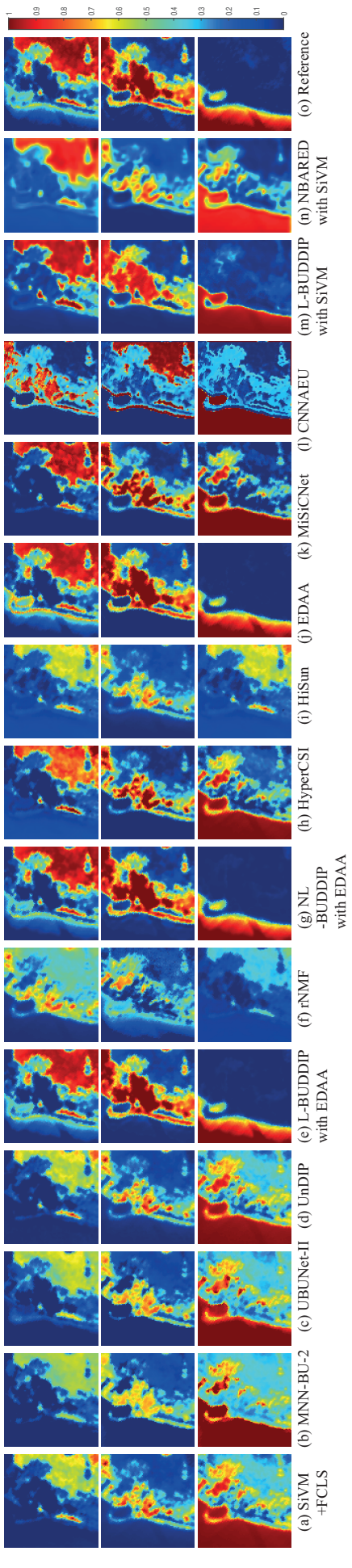


Table 6.11: Abundance maps estimated by different methods on Samson Ridge dataset. From top to bottom: Soil, Tree and Water. (a) SiVM+FCLS. (b) MNN-BU-2. (c) UBUNet-II. (d) UnDIP. (e) L-BUDDIP with EDAA. (f) rNMF. (g) NL-BUDDIP with EDAA. (h) HyperCSI. (i) HiSun. (j) EDAA. (k) MiSiCNet. (l) CNNAEU. (m) L-BUDDIP with SiVM+FCLS. (n) NBARED with SiVM+FCLS. (o)Reference.

Table 6.12: Mean and Standard Deviation of abundance RMSE, AAD (In Degrees), endmember SAD (In Degrees) by Different methods on Samson. The Best Results are in Bold. The second best are denoted with a star.

Methods	RMSE	AAD	SAD of Soil	SAD of Tree	SAD of Water	averaged SAD
HyperCSI	0.2096±0.0000	24.9273±0.0000	0.6043±0.0000⋆	1.7542±0.0000	24.4953±0.0000	8.9513±0.0000
HiSun	0.345±0.0000	31.0935±0.0000	2.8093±0.0000	4.3793±0.0000	44.1736±0.0000	17.1207±0.0000
SiVM+FCLS	0.2827±0.0000	34.6659±0.0000	1.1866±0.0000	1.7745±0.0000	7.4436±0.0000	3.4683±0.0000
MNN-BU-2	0.2910±0.0005	35.1605±0.0721	1.6588±0.0002	2.5186±0.0006	6.8634±0.0069	3.6803±0.0023
UBUNet-II	0.2846±0.0023	34.7073±0.3347	1.3765±0.0007	2.1909±0.0018	8.2532±0.0084	3.9402±0.0027
UnDIP	0.2819±0.0005	34.6133±0.0719	1.1866±0.0000	1.7745±0.0000	7.4436±0.0000	3.4683±0.0000
L-BUDDIP (EDAA)	**0.0145±0.0000**	**1.4957±0.0096**	0.8822±0.0175	**1.7022±0.0044**	1.7180±0.0142	1.4341±0.0054⋆
rNMF	0.3755±0.0000	41.9549±0.0283	25.9228±0.0078	45.5596±0.2066	27.7314±0.0326	33.0712±0.0615
NL-BUDDIP (EDAA)	0.0146±0.0002⋆	1.5064±0.0176⋆	0.8743±0.0127	1.7104±0.0045⋆	1.7013±0.0085⋆	**1.4287±0.0027**
MiSiCNet	0.1615±0.0000	19.5886±0.0010	**0.5918±0.0005**	2.0280±0.0006	22.9854±0.0321	8.5350±0.0107
CNNAEU	0.3525±0.1003	17.0234±2.1128	23.1191±13.1524	25.1235±20.5612	26.9984±29.0400	25.0803±17.5999
EDAA	0.0232±0.0000	2.6365±0.0000	1.3221±0.0000	1.9065±0.0000	**1.2988±0.0000**	1.5091±0.0000
L-BUDDIP (SiVM+FCLS)	0.1224±0.0060	13.6797±0.7344	1.1899±0.0925	2.9678±0.2761	3.1881±0.1546	2.4486±0.0298
NBARED (SiVM+FCLS)	0.2230±0.0197	26.3007±2.0631	1.5716±0.2559	2.5716±0.2494	4.4712±0.2280	2.8715±0.0974

ing framework BUDDIP and NBARED, by conducting experiments on three real datasets. We compared the results with several state-of-the-art unmixing methods, including MNN-AE, MNN-BU, UnDIP, EGU-Net, SiVM+FCLS, rNMF, HyperCSI, HiSun, EDAA, MiSiCNet and CN-NAEU. Our findings demonstrate the effectiveness of the proposed methods in comparison to existing unmixing techniques.

Chapter 7

Conclusion and Future Directions

7.1 Conclusion

Despite the significant advancements in learning-based unmixing techniques, most of these methods are dependent on an auto-encoder structure where the encoder uses a black-box deep neural network architecture, lacking a systematic approach for designing unmixing network structures. Another drawback of current methods is that, without proper guidance, deep learning-based techniques cannot ensure generating physically meaningful unmixing results. Although there is a growing number of unmixing networks that are trained with guidance, their unmixing performance is often restricted by the quality of the guidance. Additionally, the reliance on the auto-encoder structure in most learning algorithms restricts their ability to generalize from linear to nonlinear unmixing problems.

Inspired by algorithm unfolding techniques, we introduce a new HSI unmixing algorithm that combines both model- and learning-based approaches. The algorithm is based on the unrolling of the Alternating Direction Method of Multipliers (ADMM) solver in a linear mixture model represented by a constrained sparse regression problem. We intro-

duce two neural network structures: a supervised network for abundance estimation and an unsupervised network for blind unmixing. Our extensive experiments reveal that the proposed methods achieve faster convergence and superior performance even with a limited training dataset size, surpassing other unmixing methods such as MNN-AE&BU, UnDIP, and EGU-Net.

In response to the limitations of guidance, we present a general unsupervised framework, motivated by Deep Image Prior techniques, for both linear and nonlinear blind unmixing models. Our framework includes three modules: an Endmember Estimation module using DIP (EDIP), an Abundance Estimation module using DIP (ADIP), and a Mixing module (MM). The EDIP and ADIP modules generate endmembers and abundances respectively, while the MM, built based on the postulated unmixing model, produces a reconstruction of the HSI observations. To produce meaningful unmixing results, we propose a composite loss function that is applicable to both linear and nonlinear unmixing models. An adaptive loss weight strategy is also proposed to enhance unmixing results in nonlinear mixing scenarios. Experiments on both synthetic and real datasets demonstrate the superiority of the proposed methods over current state-of-the-art unmixing algorithms.

Finally, we introduce a novel blind unmixing network, called NBA, which combines the concepts of unfolding techniques and deep image priors techniques. Our proposed method utilizes a novel MatrixConv Unmixing (MCU) Model, which is solved using ADMM iterative solvers. We then unfold these solvers to construct the network structure for EDIP and ADIP. To enhance performance, we further apply a denoising regularizer (RED) to the endmember and abundance estimations. Experimental results on both synthetic and real datasets demonstrate the effectiveness of our proposed method.

7.2 Future Directions

There are several promising directions for future research on learning-based hyperspectral image (HSI) unmixing techniques.

7.2.1 Incorporate Advanced Generative Models

Given DIP techniques as a generative model, one possible direction is incorporating advanced generative models, such as variational autoencoder (VAE) [109], Generative Adversarial Networks (GAN) [110], diffusion model [111], and Transformers [112], which have shown impressive performance in corresponding generative tasks. Although there are some preliminary works [113–116] that explore this area, it remains a hot research topic.

7.2.2 Combine with other HSI techniques

In addition, there have been efforts to solve other HSI tasks such as HSI classification problems by utilizing spectral unmixing as a data augmentation technique [117]. Despite significant progress in HSI classification with recent advancements in neural networks, overfitting due to complex model structures and small training sets remains a major issue. To address this, reducing the complexity of neural networks can prevent overfitting but it may also limit their ability to capture abstract features. In [117], they propose an abundance-based multi-HSI classification method [117]. Conversely, it is worth exploring whether the performance of HSI unmixing can be further enhanced by utilizing other HSI techniques, such as HSI classification.

7.2.3 Other Applications

HSI techniques can also be combined with other imaging modalities such as fluorescence microscopy in other applications. For exam-

ple, in mineral exploration, HSI unmixing can identify mineral signatures and determine the type and abundance of minerals in a geological area [118]. In disease detection, HSI unmixing can identify cancerous tissue, monitor disease progression, and develop new diagnostic tools [119]. HSI unmixing can also be used in environmental monitoring [6] to monitor water quality, detect oil spills, and track the spread of pollutants. In agriculture [120], HSI unmixing can monitor crop health, estimate crop yields, identify invasive species, and monitor forest health and biomass. In art investigation, HSI unmixing [121] can identify the materials and techniques used in art objects, monitor their condition, and guide conservation efforts.

Bibliography

[1] R. Heylen, D. Burazerovic, and P. Scheunders, “Fully constrained least squares spectral unmixing by simplex projection,” *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4112–4122, 2011.

[2] D. C. Heinz and Chein-I-Chang, “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.

[3] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, “Hyperspectral remote sensing data analysis and future challenges,” *IEEE Geosci. Remote Sens. Mag.*, vol. 1, no. 2, pp. 6–36, 2013.

[4] D. Hong, L. Gao, J. Yao, N. Yokoya, J. Chanussot, U. Heiden, and B. Zhang, “Endmember-guided unmixing network (egu-net): A general deep learning framework for self-supervised hyperspectral unmixing,” *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–14, 2021.

[5] S. Zhao, Q. Yuan, J. Li, Y. Hu, X. Liu, and L. Zhang, “A fast and effective irregular stripe removal method for moon mineralogy mapper (m 3),” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–19, 2021.

[6] D. Zeng, S. Zhang, F. Chen, and Y. Wang, “Multi-scale cnn based garbage detection of airborne hyperspectral data,” *IEEE Access*, vol. 7, pp. 104 514–104 527, 2019.

[7] N. Rohani, E. Pouyet, M. Walton, O. Cossairt, and A. K. Katsaggelos, “Pigment unmixing of hyperspectral images of paintings using deep neural networks,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3217–3221.

[8] G. Camps and G. Camps-Valls, *Remote Sensing Image Processing*, ser. Synthesis lectures on image, video, and multimedia processing. Morgan & Claypool Publishers, 2011. [Online]. Available: https://books.google.co.uk/books?id=kjMWGbygJIQC

[9] N. Keshava and J. F. Mustard, “Spectral unmixing,” *IEEE Geosci. Remote Sens. Mag.*, vol. 19, no. 1, pp. 44–57, 2002.

[10] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, “Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 5, no. 2, pp. 354–379, 2012.

[11] J. M. P. Nascimento and J. M. B. Dias, “Vertex component analysis: a fast algorithm to unmix hyperspectral data,” *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 4, pp. 898–910, Apr. 2005.

[12] J. M. Bioucas-Dias and M. A. T. Figueiredo, “Alternating direction algorithms for constrained sparse regression: Application to hyperspectral unmixing,” in *2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, 2010, pp. 1–4.

[13] C. Févotte and N. Dobigeon, “Nonlinear hyperspectral unmixing with robust nonnegative matrix factorization,” *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4810–4819, 2015.

[14] X.-R. Feng, H.-C. Li, J. Li, Q. Du, A. Plaza, and W. J. Emery, “Hyperspectral unmixing using sparsity-constrained deep nonnegative matrix factorization with total variation,” *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 10, pp. 6245–6257, 2018.

[15] L. Miao and H. Qi, “Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization,” *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 3, pp. 765–777, 2007.

[16] Y. Qian, S. Jia, J. Zhou, and A. Robles-Kelly, “Hyperspectral unmixing via $l_{1/2}$ sparsity-constrained nonnegative matrix factorization,” *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4282–4297, 2011.

[17] Y. Qian, F. Xiong, S. Zeng, J. Zhou, and Y. Y. Tang, “Matrix-vector nonnegative tensor factorization for blind unmixing of hyperspectral imagery,” *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 3, pp. 1776–1792, 2016.

[18] Y. Qian, F. Xiong, Q. Qian, and J. Zhou, “Spectral mixture model inspired network architectures for hyperspectral unmixing,” *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 10, pp. 7418–7434, 2020.

[19] M. Zhao, M. Wang, J. Chen, and S. Rahardja, “Hyperspectral unmixing for additive nonlinear models with a 3-d-cnn autoencoder network,” *IEEE Trans. Geosci. Remote Sens.*, pp. 1–15, 2021.

[20] Q. Jin, Y. Ma, F. Fan, J. Huang, X. Mei, and J. Ma, “Adversarial autoencoder network for hyperspectral unmixing,” *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, 2021.

[21] A. Min, Z. Guo, H. Li, and J. Peng, “Jmnet: Joint metric neural network for hyperspectral unmixing,” *IEEE Trans. Geosci. Remote Sens.*, pp. 1–12, 2021.

[22] F. Xiong, J. Zhou, S. Tao, J. Lu, and Y. Qian, “Snmf-net: Learning a deep alternating neural network for hyperspectral unmixing,” *IEEE Trans. Geosci. Remote Sens.*, pp. 1–16, 2021.

[23] K. T. Shahid and I. D. Schizas, “Unsupervised hyperspectral unmixing via nonlinear autoencoders,” *IEEE Trans. Geosci. Remote Sens.*, pp. 1–13, 2021.

[24] G. A. Licciardi and F. Del Frate, “Pixel unmixing in hyperspectral data by means of neural networks,” *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4163–4172, 2011.

[25] X. Zhang, Y. Sun, J. Zhang, P. Wu, and L. Jiao, “Hyperspectral unmixing via deep convolutional neural networks,” *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 11, pp. 1755–1759, 2018.

[26] Q. Qian, F. Xiong, and J. Zhou, “Deep unfolded iterative shrinkage-thresholding model for hyperspectral unmixing,” in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Yokohama, Japan, Jul. 2019, pp. 2151–2154.

[27] F. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, “Neural network hyperspectral unmixing with spectral information divergence objective,” in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Fort Worth, TX, USA, Jul. 2017, pp. 755–758.

[28] B. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, "Hyperspectral unmixing using a neural network autoencoder," *IEEE Access*, vol. 6, pp. 25 646–25 656, 2018.

[29] Y. Qu and H. Qi, "udas: An untied denoising autoencoder with sparsity for spectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 3, pp. 1698–1712, 2019.

[30] S. Ozkan and G. B. Akar, "Improved deep spectral convolution network for hyperspectral unmixing with multinomial mixture kernel and endmember uncertainty," *arXiv preprint arXiv:1808.01104*, 2018.

[31] S. Ozkan, B. Kaya, and G. B. Akar, "Endnet: Sparse autoencoder network for endmember extraction and hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 1, pp. 482–496, 2019.

[32] Y. Su, J. Li, A. Plaza, A. Marinoni, P. Gamba, and S. Chakravortty, "Daen: Deep autoencoder networks for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4309–4321, 2019.

[33] 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, 2020.

[34] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, 2021.

[35] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, Haifa, Israel, Jun. 2010, pp. 399–406.

[36] B. Rasti, B. Koirala, P. Scheunders, and P. Ghamisi, “Undip: Hyperspectral unmixing using deep image prior,” *IEEE Trans. Geosci. Remote Sens.*, pp. 1–15, 2021.

[37] V. Lempitsky, A. Vedaldi, and D. Ulyanov, “Deep image prior,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[38] N. Dobigeon, J.-Y. Tourneret, C. Richard, J. C. M. Bermudez, S. McLaughlin, and A. O. Hero, “Nonlinear unmixing of hyperspectral images: Models and algorithms,” *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 82–94, 2014.

[39] W. Fan, B. Hu, J. Miller, and M. Li, “Comparative study between a new nonlinear model and common linear model for analysing laboratory simulated-forest hyperspectral data,” *International Journal of Remote Sensing*, vol. 30, no. 11, pp. 2951–2962, 2009.

[40] J. M. Nascimento and J. M. Bioucas-Dias, “Nonlinear mixture model for hyperspectral unmixing,” in *Image and Signal Processing for Remote Sensing XV*, vol. 7477. International Society for Optics and Photonics, 2009, p. 74770I.

[41] A. Halimi, Y. Altmann, N. Dobigeon, and J.-Y. Tourneret, “Nonlinear unmixing of hyperspectral images using a generalized bilinear model,” *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4153–4162, 2011.

[42] Q. Qu, N. M. Nasrabadi, and T. D. Tran, “Abundance estimation for bilinear mixture models via joint sparse and low-rank representation,” *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 7, pp. 4404–4423, 2014.

[43] I. Meganem, P. Déliot, X. Briottet, Y. Deville, and S. Hosseini, "Linearquadratic mixing model for reflectances in urban environments," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 1, pp. 544–558, 2014.

[44] B. Hapke, *Theory of reflectance and emittance spectroscopy*. Cambridge university press, 2012.

[45] J. M. Nascimento and J. M. Bioucas-Dias, "Unmixing hyperspectral intimate mixtures," in *Image and signal processing for remote sensing XVI*, vol. 7830. SPIE, 2010, pp. 102–109.

[46] R. Close, P. Gader, A. Zare, J. Wilson, and D. Dranishnikov, "Endmember extraction using the physics-based multi-mixture pixel model," in *Imaging Spectrometry XVII*, vol. 8515. SPIE, 2012, pp. 176–189.

[47] J. W. Boardman, "Automating spectral unmixing of aviris data using convex geometry concepts," in *JPL, Summaries of the 4th Annual JPL Airborne Geoscience Workshop. Volume 1: AVIRIS Workshop*, 1993.

[48] M. E. Winter, "N-findr: An algorithm for fast autonomous spectral end-member determination in hyperspectral data," in *Imaging Spectrometry V*, vol. 3753. SPIE, 1999, pp. 266–275.

[49] C.-I. Chang, C.-C. Wu, W. Liu, and Y.-C. Ouyang, "A new growing method for simplex-based endmember extraction algorithm," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 10, pp. 2804–2819, 2006.

[50] T.-H. Chan, W.-K. Ma, A. Ambikapathi, and C.-Y. Chi, "A simplex volume maximization framework for hyperspectral endmember extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4177–4193, 2011.

[51] M.-D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Collaborative sparse regression for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 1, pp. 341–354, 2013.

[52] A. Zouaoui, G. Muhawenayo, B. Rasti, J. Chanussot, and J. Mairal, "Entropic descent archetypal analysis for blind hyperspectral unmixing," *arXiv preprint arXiv:2209.11002*, 2022.

[53] M. D. Craig, "Minimum-volume transforms for remotely sensed data," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 3, pp. 542–552, 1994.

[54] M. Berman, H. Kiiveri, R. Lagerstrom, A. Ernst, R. Dunne, and J. F. Huntington, "Ice: A statistical approach to identifying endmembers in hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 10, pp. 2085–2095, 2004.

[55] J. M. P. Nascimento and J. M. Bioucas-Dias, "Hyperspectral unmixing algorithm via dependent component analysis," in *2007 IEEE International Geoscience and Remote Sensing Symposium*, 2007, pp. 4033–4036.

[56] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[57] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of 27th Asilomar conference on signals, systems and computers*. IEEE, 1993, pp. 40–44.

[58] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating

direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[59] M. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Collaborative sparse regression for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 1, pp. 341–354, 2014.

[60] C. Y. Zheng, H. Li, Q. Wang, and C. P. Chen, "Reweighted sparse regression for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 1, pp. 479–488, 2015.

[61] W. He, H. Zhang, and L. Zhang, "Total variation regularized reweighted sparse nonnegative matrix factorization for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3909–3921, 2017.

[62] C.-H. Lin and J. M. Bioucas-Dias, "Nonnegative blind source separation for ill-conditioned mixtures via john ellipsoid," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 2209–2223, 2020.

[63] M. D. Plumbley, "Algorithms for nonnegative independent component analysis," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 534–543, 2003.

[64] J. M. Nascimento and J. M. Dias, "Does independent component analysis play a role in unmixing hyperspectral data?" *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 1, pp. 175–187, 2005.

[65] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[66] K. Qu and W. Bao, "Multiple-priors ensemble constrained nonnegative matrix factorization for spectral unmixing," *IEEE J. Sel.*

Topics Appl. Earth Observ. Remote Sens., vol. 13, pp. 963–975, 2020.

[67] A. M. S. Ang and N. Gillis, "Algorithms and comparisons of nonnegative matrix factorizations with volume regularization for hyperspectral unmixing," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 12, pp. 4843–4853, 2019.

[68] J. Li, J. M. Bioucas-Dias, A. Plaza, and L. Liu, "Robust collaborative nonnegative matrix factorization for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6076–6090, 2016.

[69] Y. Yuan, Z. Zhang, and Q. Wang, "Improved collaborative nonnegative matrix factorization and total variation for hyperspectral unmixing," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 998–1010, 2020.

[70] X. Lv, W. Wang, and H. Liu, "Cluster-wise weighted nmf for hyperspectral images unmixing with imbalanced data," *Remote Sensing*, vol. 13, no. 2, p. 268, 2021.

[71] X.-R. Feng, H.-C. Li, R. Wang, Q. Du, X. Jia, and A. J. Plaza, "Hyperspectral unmixing based on nonnegative matrix factorization: A comprehensive review," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, 2022.

[72] A. Cutler and L. Breiman, "Archetypal analysis," *Technometrics*, vol. 36, no. 4, pp. 338–347, 1994.

[73] G. Zhao, X. Jia, and C. Zhao, "Multiple endmembers based unmixing using archetypal analysis," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 5039–5042.

[74] W. Sun, G. Yang, K. Wu, W. Li, and D. Zhang, “Pure endmember extraction using robust kernel archetypoid analysis for hyperspectral imagery,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 131, pp. 147–159, 2017.

[75] M. Xu, Z. Yang, G. Ren, H. Sheng, S. Liu, W. Liu, and C. Ye, “ℓ_1 sparsity-constrained archetypal analysis algorithm for hyperspectral unmixing,” *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.

[76] C. Zhou and M. R. Rodrigues, “An admm based network for hyperspectral unmixing tasks,” in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 1870–1874.

[77] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[78] X. Chen, J. Liu, Z. Wang, and W. Yin, “Theoretical linear convergence of unfolded ista and its practical weights and thresholds,” *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[79] R. Giryes, Y. C. Eldar, A. M. Bronstein, and G. Sapiro, “Tradeoffs between convergence speed and reconstruction accuracy in inverse problems,” *IEEE Trans. Signal Process.*, vol. 66, no. 7, pp. 1676–1690, 2018.

[80] T. Moreau and J. Bruna, “Understanding trainable sparse coding via matrix factorization,” *arXiv preprint arXiv:1609.00285*, 2016.

[81] “Usgs library.” [Online]. Available: https://www.usgs.gov/labs/spec-lab

[82] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[83] R. Rajabi and H. Ghassemian, "Spectral unmixing of hyperspectral imagery using multilayer nmf," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 1, pp. 38–42, 2014.

[84] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[85] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[86] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[87] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, Honolulu, HI, USA, Jul. 2017, pp. 1132–1140.

[88] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, 2017.

[89] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 1646–1654.

[90] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 5835–5843.

[91] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, “Blind hyperspectral unmixing using total variation and ℓ_q sparse regularization,” *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 11, pp. 6371–6384, 2016.

[92] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5967–5976.

[93] Y. Gandelsman, A. Shocher, and M. Irani, “" double-dip": Unsupervised image decomposition via coupled deep-image-priors,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 026–11 035.

[94] C. Zhou and M. R. Rodrigues, “Admm-based hyperspectral unmixing networks for abundance and endmember estimation,” *IEEE Trans. Geosci. Remote Sens.*, 2021, early Access.

[95] ——, “Blind unmixing using a double deep image prior,” in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 1665–1669.

[96] C.-H. Lin, C.-Y. Chi, Y.-H. Wang, and T.-H. Chan, “A fast hyperplane-based minimum-volume enclosing simplex algorithm for blind hyperspectral unmixing,” *IEEE Trans. Signal Process.*, vol. 64, no. 8, pp. 1946–1961, 2015.

[97] T.-H. Chan, C.-Y. Chi, Y.-M. Huang, and W.-K. Ma, “A convex analysis-based minimum-volume enclosing simplex algorithm for hyperspectral unmixing,” *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4418–4432, 2009.

[98] A. Ambikapathi, T.-H. Chan, W.-K. Ma, and C.-Y. Chi, "Chance-constrained robust minimum-volume enclosing simplex algorithm for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4194–4209, 2011.

[99] C.-H. Lin, W.-K. Ma, W.-C. Li, C.-Y. Chi, and A. Ambikapathi, "Identifiability of the simplex volume minimization criterion for blind hyperspectral unmixing: The no-pure-pixel case," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 10, pp. 5530–5546, 2015.

[100] B. Palsson, M. O. Ulfarsson, and J. R. Sveinsson, "Convolutional autoencoder for spectral–spatial hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 1, pp. 535–549, 2020.

[101] B. Rasti, B. Koirala, P. Scheunders, and J. Chanussot, "Misicnet: Minimum simplex convolutional network for deep hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, 2022.

[102] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (red)," *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1804–1844, 2017.

[103] G. Mataev, P. Milanfar, and M. Elad, "Deepred: Deep image prior powered by red," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[104] X. Deng and P. L. Dragotti, "Deep convolutional neural network for multi-modal image restoration and fusion," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3333–3348, 2020.

[105] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference*

on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, 2005, pp. 60–65 vol. 2.

[106] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[107] F. Zhu, Y. Wang, S. Xiang, B. Fan, and C. Pan, "Structured sparse method for hyperspectral unmixing," *ISPRS J. Photogramm. Remote Sens.*, vol. 88, pp. 101–118, 2014.

[108] F. Zhu, Y. Wang, B. Fan, S. Xiang, G. Meng, and C. Pan, "Spectral unmixing via data-guided sparsity," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5412–5427, 2014.

[109] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[110] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[111] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference on Machine Learning*. PMLR, 2015, pp. 2256–2265.

[112] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[113] S. Shi, M. Zhao, L. Zhang, and J. Chen, "Variational autoencoders for hyperspectral unmixing with endmember variability,"

in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 1875–1879.

[114] M. Tang, Y. Qu, and H. Qi, "Hyperspectral nonlinear unmixing via generative adversarial network," in *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2020, pp. 2404–2407.

[115] S. L. Polk, K. Cui, A. H. Chan, D. A. Coomes, R. J. Plemmons, and J. M. Murphy, "Unsupervised diffusion and volume maximization-based clustering of hyperspectral images," *Remote Sensing*, vol. 15, no. 4, p. 1053, 2023.

[116] P. Ghosh, S. K. Roy, B. Koirala, B. Rasti, and P. Scheunders, "Deep hyperspectral unmixing using transformer network," *arXiv preprint arXiv:2203.17076*, 2022.

[117] A. J. Guo and F. Zhu, "Improving deep hyperspectral image classification performance with spectral unmixing," *Signal Processing*, vol. 183, p. 107949, 2021.

[118] S. Sudharsan, R. Hemalatha, and S. Radha, "A survey on hyperspectral imaging for mineral exploration using machine learning algorithms," in *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*. IEEE, 2019, pp. 206–212.

[119] I. H. Aboughaleb, M. H. Aref, and Y. H. El-Sharkawy, "Hyperspectral imaging for diagnosis and detection of ex-vivo breast cancer," *Photodiagnosis and Photodynamic Therapy*, vol. 31, p. 101922, 2020.

[120] A. Gómez-Sánchez, M. Marro, M. Marsal, S. Zacchetti, R. Rocha de Oliveira, P. Loza-Alvarez, and A. de Juan, "Lin-

ear unmixing protocol for hyperspectral image fusion analysis applied to a case study of vegetal tissues," *Scientific Reports*, vol. 11, no. 1, p. 18665, 2021.

[121] D. Bai, D. W. Messinger, and D. Howell, "A hyperspectral imaging spectral unmixing and classification approach to pigment mapping in the gough & selden maps," *Journal of the American Institute for Conservation*, vol. 58, no. 1-2, pp. 69–89, 2019.