Integration of Physics-Based and Data-Driven Models for Hyperspectral Image Unmixing: A summary of current methods

Jie Chen, Min Zhao, Xiuheng Wang, Cédric Richard, Susanto Rahardja

Integration of Physics-Based and Data-Driven Models for Hyperspectral Image Unmixing

Jie Chen[⋆], *Senior Member, IEEE*, Min Zhao[⋆], *Student Member, IEEE*, Xiuheng Wang[†], *Student Member, IEEE*

Cédric Richard[†], *Senior Member, IEEE*, Susanto Rahardja[⋆‡], *Fellow, IEEE*

[⋆] School of Marine Science and Technology, Northwestern Polytechnical University, 710072, Xi'an, China

dr.jie.chen@ieee.org, minzhao@mail.nwpu.edu.cn, susantorahardja@ieee.org

[†] Laboratoire Lagrange, Université Côte d'Azur, 06000, Nice, France

xiuheng.wang@oca.eu, cedric.richard@unice.fr

[‡] Singapore Institute of Technology, 138683, Singapore

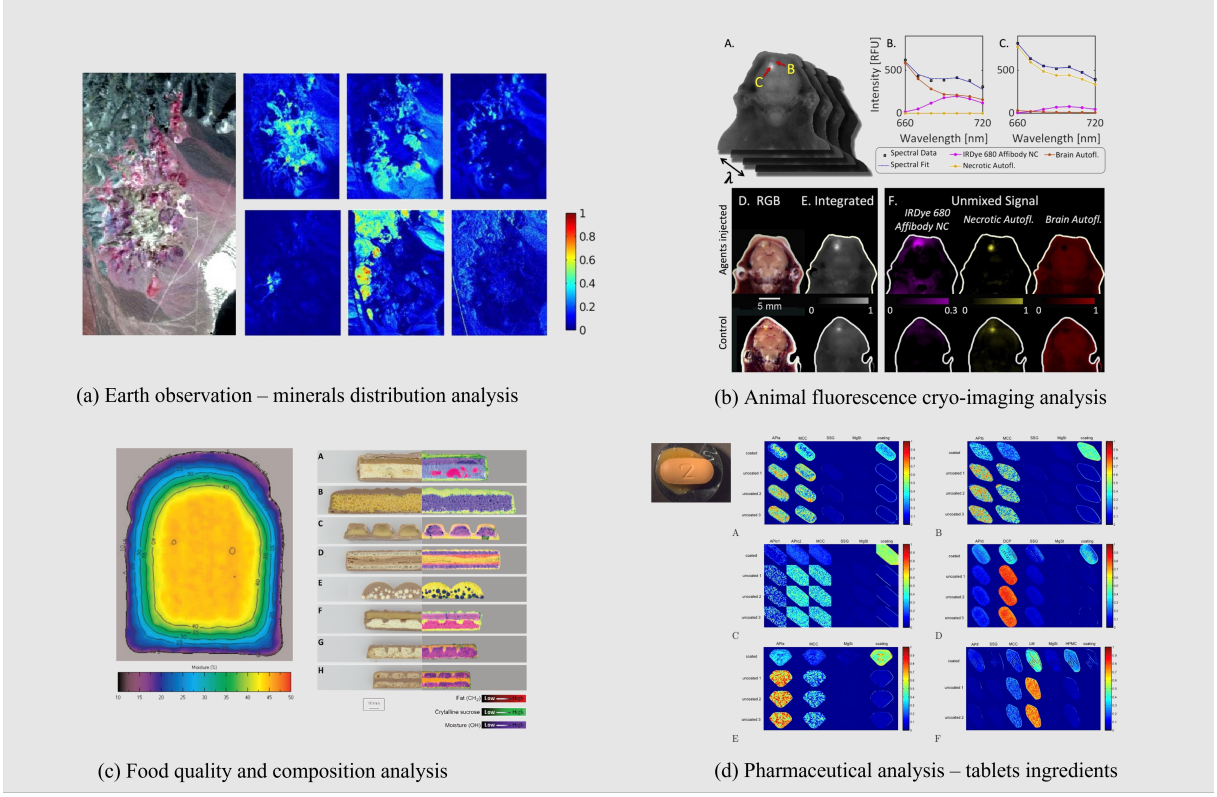
Abstract

Spectral unmixing is central when analyzing hyperspectral data. To accomplish this task, physics-based methods have become popular because, with their explicit mixing models, they can provide a clear interpretation. Nevertheless, due to their limited modeling capabilities, especially when analyzing real scenes with unknown complex physical properties, these methods may not be accurate. On the other hand, data-driven methods using deep learning in particular have developed rapidly in recent years thanks to their superior capability in modeling complex nonlinear systems. Simply transferring these methods as black boxes to perform unmixing may lead to low interpretability and poor generalization ability. To bring together the best of two worlds, recent research efforts have focused on combining the advantages of both physics-based models and data-driven methods. In this article, we present an overview of recent advances on this topic from various perspectives, including deep neural networks (DNN) design, prior capturing and loss selection. We summarise these methods within a common optimization framework and discuss ways for enhancing the understanding of these methods. The related source codes are made publicly available[1].

I. Introduction

Conventional machine vision applications widely use Red-Green-Blue (RGB) cameras. These cameras are suitable for identifying objects with shapes and colors consistent with the human visual system. As a significant advancement in imaging techniques, hyperspectral cameras measure objects or scenes by recording hundreds of narrow bands across a wide-range spectrum that can cover the non-visible range of light. In that way, hyperspectral images enable the identification of materials based on their unique spectral signatures, beyond their visible characteristics, coupled with spatial information [1]. Initial hyperspectral imaging applications started with remote sensing of the Earth's surface using satellite or airborne cameras, e.g., for mineral exploration, vegetation monitoring, and land analysis. Recently, the subsequent technical advancements and ongoing

[1] http://github.com/xiuheng-wang/awesome-hyperspectral-image-unmixing



(a) Earth observation – minerals distribution analysis

(b) Animal fluorescence cryo-imaging analysis

(c) Food quality and composition analysis

(d) Pharmaceutical analysis – tablets ingredients

Fig. 1. Applications of hyperspectral data unmixing. (a) Analysis of mineral distribution of remotely sensed data (CUPRITE) [4]; (b) Spectral unmixing for improving whole-body fluorescence cryo-imaging[1]; (c) Food analysis: moisture distribution in a fresh slice of bread and composition analysis of commercial chocolate bars[2]; (d) Pharmaceutical analysis: tablets ingredients distribution estimation[3].

cost reduction of hyperspectral cameras have made their deployment practical and possible in other emerging applications such as food safety inspection, medical diagnostics, industrial sorting, drug analysis, biometric forensics, archaeology, etc.

The spectral content of individual pixels in hyperspectral images is typically a mixture of the spectra of multiple materials [2], [3]. This phenomenon called spectral mixture is attributed to multiple factors, mainly including the low spatial resolution of hyperspectral imaging devices, the diversity and intimate interactions of materials in the imaged scenes, and multiple photon reflections from layered objects. Separating spectra of individual pixels into a set of spectral signatures (called endmembers), and determining the abundance fraction of each endmember, is an essential task in quantitative hyperspectral subpixel analysis. This process, denoted as spectral unmixing or mixed pixel decomposition, is currently used in many applications such as conventional mineral distribution analysis in remote sensing, biodistribution analysis in fluorescence images, food quality and composition control, and pharmaceutical inspection, as illustrated in Fig. 1.

Diverse solutions have been proposed for the spectral unmixing problem. They can be divided primarily into two categories: physics-based and data-driven methods. Physics-based methods exploit physical models to estimate light scattering and and interaction mechanisms among multiple materials in the imaged scene. The observed spectra are explicitly related to the endmembers and abundances, albeit with strong assumptions regarding photon-interacting processes. In practice, it is difficult

[1]https://opg.optica.org/boe/fulltext.cfm?uri=boe-12-1-395&id=444823

[2]https://www.specim.fi/better-control-of-food-quality-and-composition-with-hyperspectral-imaging

[3]http://www.lx.it.pt/~bioucas/files/PhD_Thesis_MBLopes.pdf

TABLE I

MAIN SYMBOLS USED IN THIS PAPER.

x, X	scalar	$\boldsymbol{y}$	a general observed pixel
$\boldsymbol{x}$	column vector	$\boldsymbol{y}_i$	i-th pixel of the image, with $1 \leq i \leq N$
$\boldsymbol{X}$	matrix	$\boldsymbol{m}_i = [m_{i,1}, \cdots, m_{i,L}]^\top$	i-th endmember, with $1 \leq i \leq R$
$\mathbb{X}$	3D data cube	$\boldsymbol{M} = [\boldsymbol{m}_1, \cdots, \boldsymbol{m}_R]$	endmembers matrix with R pure spectra
N	number of pixels	$\boldsymbol{a} = [a_1, \cdots, a_R]^\top$	a general abundance vector
L	number of spectral bands	$\boldsymbol{a}_i = [a_{i,1}, \cdots, a_{i,R}]^\top$	abundance vector of the i-th pixel
R	number of pure spectra (endmembers)	$\boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N]$	matrix by stacking $\mathbb{Y}$ with N pixels
$\mathbb{Y}$	image with N pixels and L spectral bands	$\boldsymbol{A} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N]$	abundance matrix of N pixels

to accurately model real scenes, and sophisticated models with complex mathematics are usually difficult to estimate. In contrast, data-driven unmixing methods can overcome some limitations of the aforementioned physics-based methods by directly deriving mixture models from the observed hyperspectral data. These techniques are effective in uncovering the underlying relationships in complicated scenes with abundant data. Nevertheless, explicit and useful knowledge of the physical mixing process is often overlooked in naive data-driven approaches. In order to reap the benefits of both physics-based and data-driven models, it is essential to properly integrate these techniques in order to achieve superior unmixing performance with clear interpretation.

The main purpose of this article is to provide an overview of hyperspectral unmixing models and techniques with special attention focused on the integration of physics-based models and data-driven methods. These methods are analyzed from multiple perspectives, including model structure design, prior capturing, and loss function design. In addition to review the materials dispersed in the literature, we establish a mathematical framework that can characterize and relate these methods. This article provides a set of methods that can serve as good examples for understanding the topics, and a selection of references that are sufficient to cover the main contributions. The symbols used in this article are provided in Table I.

II. A GENERAL PROBLEM FORMULATION OF THE UNMIXING PROBLEM

This section establishes a general modeling and optimization framework for spectral unmixing. The classical and recent physics-based and data-driven unmixing methods described subsequently are compatible with this framework.

Generally, hyperspectral unmixing consists of three essential steps, namely, the estimation of the number of endmembers present in the imaged scene, their extraction from available data, and the estimation of their respective abundances. Estimating the number of endmembers can be performed with signal dimension estimation techniques, or circumvented with sparse regression methods. Estimating the endmembers and their abundances can be performed sequentially or simultaneously. During unmixing process, endmembers and abundances are often constrained to preserve physical properties and interpretability. The endmembers represent the pure spectra existing in the imaged scene, implying that the so-called endmember nonnegativity constraint (ENC) is usually imposed. The abundances represent the contribution in percentage of each pure component, meaning that the abundance nonnegativity constraint (ANC) and the abundance sum-to-one constraint (ASC) are often considered. The

feasible region of endmembers and abundances is denoted by Ω_M and Ω_a, with:

$$\Omega_M = \big\{\boldsymbol{M} : \boldsymbol{M} \geq \mathbf{0}\big\} \tag{1}$$

$$\Omega_a = \big\{\boldsymbol{a} : \boldsymbol{a} \geq \mathbf{0};\ a_1 + a_2 + \cdots + a_R = 1\big\} \tag{2}$$

where $\cdot \geq \mathbf{0}$ denotes element-wise nonnegativity of its matrix or vector argument. Feasible region Ω_M is for the ENC, and feasible region Ω_a is for the ANC and ASC. Parameter R denotes the number of endmembers available in the imaged scene.

An observed pixel spectrum $\boldsymbol{y}$ can be described by the following general mixture mechanism:

$$\boldsymbol{y} = \mathcal{F}^{\star}(\boldsymbol{M}, \boldsymbol{a}) + \boldsymbol{\epsilon} \tag{3}$$

where $\mathcal{F}^{\star}$ defines the inherent photon interaction mechanism parameterized by the endmembers and their abundances. In a nutshell, spectral unmixing aims at determining the endmembers and their respective abundances by considering the inverse process of $(\mathcal{F}^{\star})^{-1} : \boldsymbol{y} \to \{\hat{\boldsymbol{M}}, \hat{\boldsymbol{a}}\}$ under the constraints defined by Ω_M and Ω_a, provided that this inverse is known or can be estimated. From the perspective of mathematical optimization, the general form of the unmixing problem can be formulated in different terms depending on known informations and priors.

Problem formulation I: Consider a scene with N pixels $\{\boldsymbol{y}_i\}_{i=1}^N$. Under the assumption that the mixture mechanism is defined by an explicit transformation $\mathcal{F}$, the optimization formulation of the unmixing problem is given by:

$$\begin{aligned} \big\{\hat{\boldsymbol{M}}, \{\hat{\boldsymbol{a}}_i\}_{i=1}^N\big\} = \underset{\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N}{\operatorname{argmin}} & \sum_{i=1}^{N} \mathcal{L}\big(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i\big) + \mathcal{R}\big(\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N\big) \\ \text{with} \quad & \hat{\boldsymbol{y}}_i = \mathcal{F}(\boldsymbol{M}, \boldsymbol{a}_i) \\ \text{s.t.} \quad & \boldsymbol{M} \in \Omega_M, \text{ and } \boldsymbol{a}_i \in \Omega_a \text{ for } \forall\, i \end{aligned} \tag{4}$$

where:

$\mathcal{L}(\cdot, \cdot)$ is a loss function defining the dissimilarity between its two arguments,

$\mathcal{R}$ is a regularizer that incorporates priors on endmembers $\boldsymbol{M}$ and abundances $\{\boldsymbol{a}_i\}_{i=1}^N$.

Extensive research has been conducted to derive explicit mathematical formulations for describing mixture mechanisms. These physics-based models are constructed with different assumptions regarding how photons interact with scene materials. The works in [2], [5] provide thorough reviews on physical mixture models and associated unmixing methods. Some typical models are presented as examples below:

LMM. In the linear mixture model (LMM), the macroscopically pure components are assumed to be homogeneously distributed in separate patches within the view field. Each photon only interacts with one pure material before reaching the observer, without being affected by any other material. The LMM is expressed as follows:

$$\begin{aligned} \boldsymbol{y} &= a_1\boldsymbol{m}_1 + a_2\boldsymbol{m}_2 + \cdots + a_R\boldsymbol{m}_R + \boldsymbol{\epsilon} \\ &= \boldsymbol{M}\boldsymbol{a} + \boldsymbol{\epsilon}, \end{aligned} \tag{5}$$

that is, the observed spectrum $\boldsymbol{y}$ is a linear combination of $a_j\boldsymbol{m}_j$, where $\boldsymbol{m}_j$ is the spectrum of the j-th pure material weighted by its abundance a_j. An additive noise is denoted by $\boldsymbol{\epsilon}$. A general form of the LMM considers different endmembers at each pixel due to, e.g., spectral variability. It is formulated as:

$$\boldsymbol{y}_i = \boldsymbol{M}_i\,\boldsymbol{a}_i + \boldsymbol{\epsilon}_i, \tag{6}$$

where $\boldsymbol{M}_i$, having the same structure as $\boldsymbol{M}$ defined in Table I, is the endmember matrix at the i-th pixel.

Bilinear model. Bilinear models [6] generalize the LMM by introducing second-order reflection signatures to capture, e.g., photon scatterings caused by multiple vegetation layers in agricultural scenarios. The mixture model is described by:

$$\boldsymbol{y} = \boldsymbol{M}\boldsymbol{a} + \sum_{i=1}^{R}\sum_{j=1}^{R}\gamma_{i,j}\boldsymbol{m}_i \odot \boldsymbol{m}_j + \boldsymbol{\epsilon}, \tag{7}$$

where $\odot$ denotes the Hadamard product, and $\gamma_{i,j}$ stands for the nonlinear contribution of the i-th and j-th endmembers. The expression of $\gamma_{i,j}$ allows us to define different bilinear models. For instance, $\gamma_{i,j} = \beta_{i,j}a_i a_j$ yields the generalized bilinear model in [6], where $\beta_{i,j}$ controls the interaction strength between the i-th and j-th endmember.

Hapke model. The Hapke model has been developed to characterize intimate mixtures with complex photon interactions [7]. The full Hapke model is complex and requires a number of parameters that are generally unavailable in real-world scenes. Under certain reasonable assumptions and simplifications such as spherical particles and isotropic scatters, the relationship between the bidirectional reflectance $\boldsymbol{y}$ and the single-scattering albedo (SSA) $\boldsymbol{w}$ becomes:

$$\boldsymbol{y} = \mathcal{S}(\boldsymbol{w}) = \frac{\boldsymbol{w}}{(1+2\mu\sqrt{1-\boldsymbol{w}})\left(1+2\mu_0\sqrt{1-\boldsymbol{w}}\right)} \tag{8}$$

where the division is the element-wise operator, μ_0 and μ are the cosines of the incoming and outgoing radiation angles, respectively. In spite of being nonlinear in terms of reflectance, the intimate mixture model is linear in the SSA domain and can be expressed as follows:

$$\boldsymbol{w} = \sum_{i=1}^{R} a_i \boldsymbol{w}_i. \tag{9}$$

Unmixing problems with physics-based models frequently fall under Problem formulation I when $\mathcal{F}$ is explicitly defined. Below are two examples of linear unmixing problems with known and unknown endmembers, respectively.

`Example 1 (The FCLS unmixing)`: Under the linear model assumption $\mathcal{F}(\boldsymbol{M},\boldsymbol{a}) = \boldsymbol{M}\boldsymbol{a}$ with known $\boldsymbol{M}$, considering independent pixels and using the squared ℓ_2-norm error $\mathcal{L}(\boldsymbol{y}_i,\hat{\boldsymbol{y}}_i) = \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|^2$, problem (4) reduces to the Fully Constrained Least Square (FCLS) problem [8]:

$$\begin{aligned} \hat{\boldsymbol{a}} = &\underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{M}\boldsymbol{a}\|^2 \\ &\text{s.t.} \quad \boldsymbol{a} \in \Omega_\alpha. \end{aligned} \tag{10}$$

`Example 2 (The regularized NMF unmixing)`: Under linear model assumption $\mathcal{F}(\boldsymbol{M},\boldsymbol{a}_i) = \boldsymbol{M}\boldsymbol{a}_i$ for $i = 1,\cdots,N$,

and using $\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|^2$ with a regularization term, problem (4) becomes:

$$\begin{aligned} \{\hat{\boldsymbol{M}}, \{\hat{\boldsymbol{a}}_i\}_{i=1}^N\} = \underset{\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N}{\operatorname{argmin}} \ & \|\boldsymbol{Y} - \boldsymbol{M}\boldsymbol{A}\|_F^2 + \mathcal{R}\big(\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N\big) \\ \text{s.t.} \quad & \boldsymbol{M} \in \Omega_M, \text{ and } \boldsymbol{a}_i \in \Omega_a, \end{aligned} \tag{11}$$

where we notice that $\|\boldsymbol{Y} - \boldsymbol{M}\boldsymbol{A}\|_F^2 = \sum_{i=1}^N \|\boldsymbol{y}_i - \boldsymbol{M}\boldsymbol{a}_i\|^2$. The design of regularizers $\mathcal{R}$ for enhancing the unmixing performance has been extensively studied in the literature. For example, the minimum volume-based term is usually introduced as a geometrical constraint on the endmember estimation [9]. Total variation, sparsity and low-rankness are usually used to regularize the abundance estimation [10].

Though the physics-based model and unmixing with Problem formulation I have been widely used, it is clear that the real mixture mechanism may not be explicitly known or accurately assumed, and the actual physics-based models have limited nonlinearity modeling capacity. Simultaneously learning the function $\mathcal{F}$ from data while determining the endmembers and abundances appears as a remedy to this problem.

Problem formulation II: Under the assumption that the mixture mechanism $\mathcal{F}$ is unknown or partially unknown, the optimization formulation that learns $\mathcal{F}$ from the data can be formulated as:

$$\begin{aligned} \{\hat{\boldsymbol{M}}, \{\hat{\boldsymbol{a}}_i\}_{i=1}^N, \hat{\mathcal{F}}\} = \underset{\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N, \mathcal{F} \in \mathcal{H}}{\operatorname{argmin}} \ & \sum_{i=1}^N \mathcal{L}\big(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i\big) + \mathcal{R}\big(\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N\big) \\ \text{with} \quad & \hat{\boldsymbol{y}}_i = \mathcal{F}\big(\boldsymbol{M}, \boldsymbol{a}_i\big) \\ \text{s.t.} \quad & \boldsymbol{M} \in \Omega_M, \text{ and } \boldsymbol{a}_i \in \Omega_a \end{aligned} \tag{12}$$

where $\mathcal{H}$ denotes a functional space where $\mathcal{F}$ lies. Typical nonlinear unmixing problems with conventional machine learning techniques often fall into this category [4], [11].

Problem formulations I and II imply that the three steps outlined below are essential for achieving good unmixing performance:

- Determining a mixture model $\mathcal{F}$ that matches the photon interaction mechanism;
- Defining a loss function that characterizes the dissimilarity between the reconstructed and the observed spectra;
- Constructing a regularizer that captures the prior information on the variables.

Learning $\mathcal{F}$ from data opens the possibility of determining mixture mechanisms that are difficult to formulate using explicit descriptions and expressions. However, this strategy may fail if $\mathcal{F}$ is not sufficiently parameterized and jointly constrained with physics-based models. Some researchers have specified several forms for $\mathcal{F}$ to make it more interpretable. In addition to the trivial linear form, the additive nonlinear model and post-nonlinear model are two major categories that have been extensively investigated.

Additive nonlinear models. Models in this category consider that the mixture consists of a linear mixture with an additive nonlinear component involving the endmembers and their abundances, that is,

$$\boldsymbol{y} = \boldsymbol{M}\boldsymbol{a} + \mathcal{F}_{\text{add}}(\boldsymbol{M}, \boldsymbol{a}) + \boldsymbol{\epsilon}. \tag{13}$$

Several subcategories, depending on specific forms for $\mathcal{F}_{\text{add}}$, have been investigated to make the problem tractable, including:

- The linear-mixture/nonlinear-fluctuation model: This model assumes that the mixture consists of two components, namely, a linear mixture component and a nonlinear fluctuation depending on the interaction between the endmembers, that is,

$$\boldsymbol{y} = \boldsymbol{M}\boldsymbol{a} + \mathcal{F}_{\text{add}}(\boldsymbol{m}_1, \cdots, \boldsymbol{m}_R) + \boldsymbol{\epsilon}. \tag{14}$$

  Though being useful and attractive, this model has some limitations: the additive nonlinear fluctuation function of this model does not depend on the abundances, and all endmembers contribute uniformly to the nonlinearity of the model.
- The generalized linear-mixture/nonlinear-fluctuation model: This form suggests that the contribution of each material to the nonlinear interaction is proportional to its abundance, that is,

$$\boldsymbol{y} = \boldsymbol{M}\boldsymbol{a} + \mathcal{F}_{\text{add}}(a_1\boldsymbol{m}_1, \cdots, a_R\boldsymbol{m}_R) + \boldsymbol{\epsilon}. \tag{15}$$

  This assumption appears reasonable given that a material with insignificant abundance may contribute minimally to both the linear and nonlinear mixing components in $\boldsymbol{y}$, and vice versa.

Post-nonlinear models. Models in this category consider that a spectrum is produced by distorting the LMM with a nonlinear function, that is,

$$\boldsymbol{y} = \mathcal{F}_{\text{post}}(\boldsymbol{M}\boldsymbol{a}) + \boldsymbol{\epsilon}. \tag{16}$$

A specific form can be constructed by setting $\mathcal{F}_{\text{post}} = I + \mathcal{F}_{\text{add}}$, leading to the additive post-nonlinear model:

$$\boldsymbol{y} = \boldsymbol{M}\boldsymbol{a} + \mathcal{F}_{\text{add}}(\boldsymbol{M}\boldsymbol{a}) + \boldsymbol{\epsilon}, \tag{17}$$

which can also be viewed as a special case of (15) by setting $\mathcal{F}_{\text{add}}(a_1\boldsymbol{m}_1, \cdots, a_R\boldsymbol{m}_R) = \mathcal{F}_{\text{add}}(a_1\boldsymbol{m}_1 + \cdots + a_R\boldsymbol{m}_R)$.

Figure 2 shows the hierarchical structure of the above categories and how they cover typical physics-based mixture models. The examples below illustrate the relationship between Problem formulation II and two typical nonlinear unmixing techniques.

`Example 3 (K-Hype and regularized K-Hype unmixing)`**:** The derivation of the K-Hype algorithm is based on a linear-mixture/nonlinear-fluctuation model of the form (14). The nonlinear fluctuation function $\mathcal{F}_{\text{add}}$ is constrained to be a member of a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_{\text{RKHS}}$ and expressed with kernels. Considering pixels independently

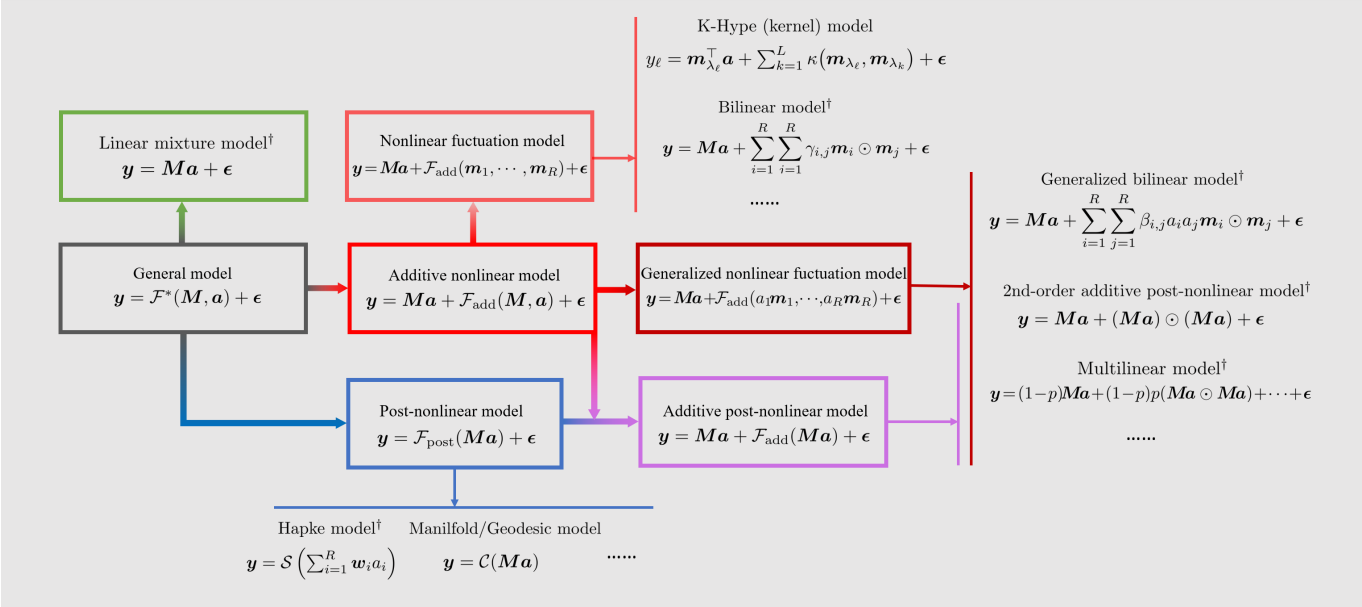


Fig. 2. Hierarchical structure of general mixture models and physics-based models. The arrows go from general models to more specific ones. Physics-based models are marked with †.

and assuming that $\boldsymbol{M}$ is known, the optimization problem is given by:

$$\begin{aligned} \{\hat{\boldsymbol{a}}, \hat{\mathcal{F}}_{\text{add}}\} = \underset{\boldsymbol{a}, \mathcal{F}_{\text{add}} \in \mathcal{H}_{\text{RKHS}}}{\text{argmin}} \; & \frac{1}{2\mu} \sum_{\ell=1}^{L} e_\ell^2 + \frac{1}{2}\left(\|\mathcal{F}_{\text{add}}\|^2 + \|\boldsymbol{a}\|^2\right) \\ \text{with} \quad & e_\ell = y_\ell - \boldsymbol{m}_{\lambda_\ell}^\top \boldsymbol{a} - \mathcal{F}_{\text{add}}(\boldsymbol{m}_{\lambda_\ell}) \\ \text{s.t.} \quad & \boldsymbol{a} \in \Omega_a \end{aligned} \tag{18}$$

where $\boldsymbol{m}_{\lambda_\ell}$ denotes the ℓ-th row of $\boldsymbol{M}$, i.e., all reflectance at band λ_ℓ. This problem can be solved using the Lagrangian method, which yields an explicit expression relating $\hat{\boldsymbol{a}}$ and the dual variables, and expands $\hat{\mathcal{F}}_{\text{add}}$ as a sequence of kernels parameterized by $\{\boldsymbol{m}_{\lambda_\ell}\}_{\ell=1}^{L}$. The K-Hype formulation, along with its sparse variant [11], and its spatially-regularized variant in [4], clearly falls into Problem formulation II.

`Example 4 (Unmixing with manifold learning/geodesic distance)`**:** The geodesic distance can be calculated to conduct data unmixing on a manifold rather than in a linear space. This process considers a nonlinear but continuous bijective mapping $\mathcal{C}$ between the linear space of abundance coefficients and a manifold in the spectral space:

$$\boldsymbol{y} = \mathcal{C}(\boldsymbol{M}\boldsymbol{a}) + \boldsymbol{\epsilon}, \tag{19}$$

which is of the form (16) as shown in [12], [13]. More specifically, defining a K-Nearest Neighbor (KNN) graph with the data set, the geodesic distance can be approximated as the length of the shortest path between two points. The endmembers $\{\boldsymbol{m}_i\}_{i=1}^{R}$ span the simplex with the largest volume on the data manifold, and the abundance coefficients of each data to be unmixed can be expressed as volume ratios in this manifold. The manifold formulation also falls into Problem formulation II as $\mathcal{C}$ is learnt from data.

Problem formulations I and II cover physics-based methods or conventional machine learning techniques with handcrafted regularizers. This article focuses more on the integration of physics-based methods with recent machine learning techniques

powered by DNNs and other data-driven techniques. Problem formulation III encompasses these new methods by introducing additional hierarchical structures on endmembers and abundances.

Problem formulation III: Under the assumption that the mixture mechanism $\mathcal{F}$ is unknown or partially unknown, the optimization formulation that i) jointly learns $\mathcal{F}$ from data, and that ii) considers hierarchical reparameterization of the endmembers and abundances, can be formulated as:

$$\begin{aligned}\big\{\hat{\boldsymbol{M}}, \boldsymbol{\Theta}_{\boldsymbol{M}}^{\star}, \{\hat{\boldsymbol{a}}_i\}_{i=1}^N, \boldsymbol{\Theta}_{\boldsymbol{a}}^{\star}, \hat{\mathcal{F}}\big\} = & \underset{\boldsymbol{M}, \boldsymbol{\Theta}_{\boldsymbol{M}}, \{\boldsymbol{a}_i\}_{i=1}^N, \boldsymbol{\Theta}_{\boldsymbol{a}}, \mathcal{F}\in\mathcal{H}}{\operatorname{argmin}} \quad \sum_{i=1}^{N} \mathcal{L}\big(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i\big) + \mathcal{R}\big(\boldsymbol{M}, \{\boldsymbol{a}_i\}_{i=1}^N\big) \\ & \text{with} \quad \hat{\boldsymbol{y}}_i = \mathcal{F}\big(\boldsymbol{M}(\boldsymbol{\Theta}_{\boldsymbol{M}}), \boldsymbol{a}_i(\boldsymbol{\Theta}_{\boldsymbol{a}})\big) \\ & \text{s.t.} \quad \boldsymbol{M} \in \Omega_M, \text{ and } \boldsymbol{a}_i \in \Omega_a\end{aligned} \tag{20}$$

where $\boldsymbol{\Theta}_{\boldsymbol{M}}$ and $\boldsymbol{\Theta}_{\boldsymbol{a}}$ are parameters that allow us to determine the endmembers matrix $\boldsymbol{M}$ and the abundances $\{\boldsymbol{a}_i\}_{i=1}^N$, respectively, and $\mathcal{H}$ denotes a given function space. For example, in recent DNN structures, $\boldsymbol{\Theta}_{\boldsymbol{a}}$ can represent the encoder in an autoencoder unmixing framework, and $\boldsymbol{\Theta}_{\boldsymbol{M}}$ can represent the decoder part that generates the endmembers. Recent physically inspired data-driven methods will be reviewed in the following three sections. Their connections to Problem formulation III will be also established. In contrast to conventional machine learning based unmixing methods that mostly focus on learning mixture models from data, recent data-driven methods have shown their superiority in improving a variety of aspects including model design, prior capturing, and loss function design.

III. Integrating of physics-based models in DNN design

In practice, it appears natural to model $\mathcal{F}^{\star}$ in (3) using DNNs since they offer enhanced modeling capacity compared to conventional nonlinear modeling techniques. Instead of using DNNs as black boxes, elegant designs of network structures for modeling (15)–(17) are indispensable for preserving physical interpretation and then to facilitate the extraction of abundances and endmembers. Deep autoencoder networks are considered as the most appropriate ones and are intensively studied for this purpose. Essentially, an autoencoder consists of an encoder network and a decoder network. The encoder part seeks for a low dimensional representation (e.g., the abundances) to reconstruct the input image. Given an input $\boldsymbol{y}$,

$$\boldsymbol{\varrho} = f_{\boldsymbol{\Theta}_{\text{enc}}}(\boldsymbol{y}) \tag{21}$$

where $f_{\boldsymbol{\Theta}_{\text{enc}}}$ is the function representing the encoder with $\boldsymbol{\Theta}_{\text{enc}}$ denoting all network parameters. The decoder part reverses the functionality of the encoder by decompressing the hidden representation vector to reconstruct the original input data, namely:

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\Theta}_{\text{dec}}}(\boldsymbol{\varrho}) \tag{22}$$

where $f_{\boldsymbol{\Theta}_{\text{dec}}}$ is the function representing the decoder with $\boldsymbol{\Theta}_{\text{dec}}$ denoting all network parameters. For instance, an autoencoder can be designed to provide an estimate $\boldsymbol{\varrho}$ of the abundances $\boldsymbol{a}$, and to estimate the endmembers and the mixture mechanism with the decoder network. Then, identifications $\boldsymbol{\Theta}_{\boldsymbol{a}} \leftarrow \boldsymbol{\Theta}_{\text{enc}}$ and $\mathcal{F} \leftarrow f_{\boldsymbol{\Theta}_{\text{dec}}}$ can be considered to fit these structures into

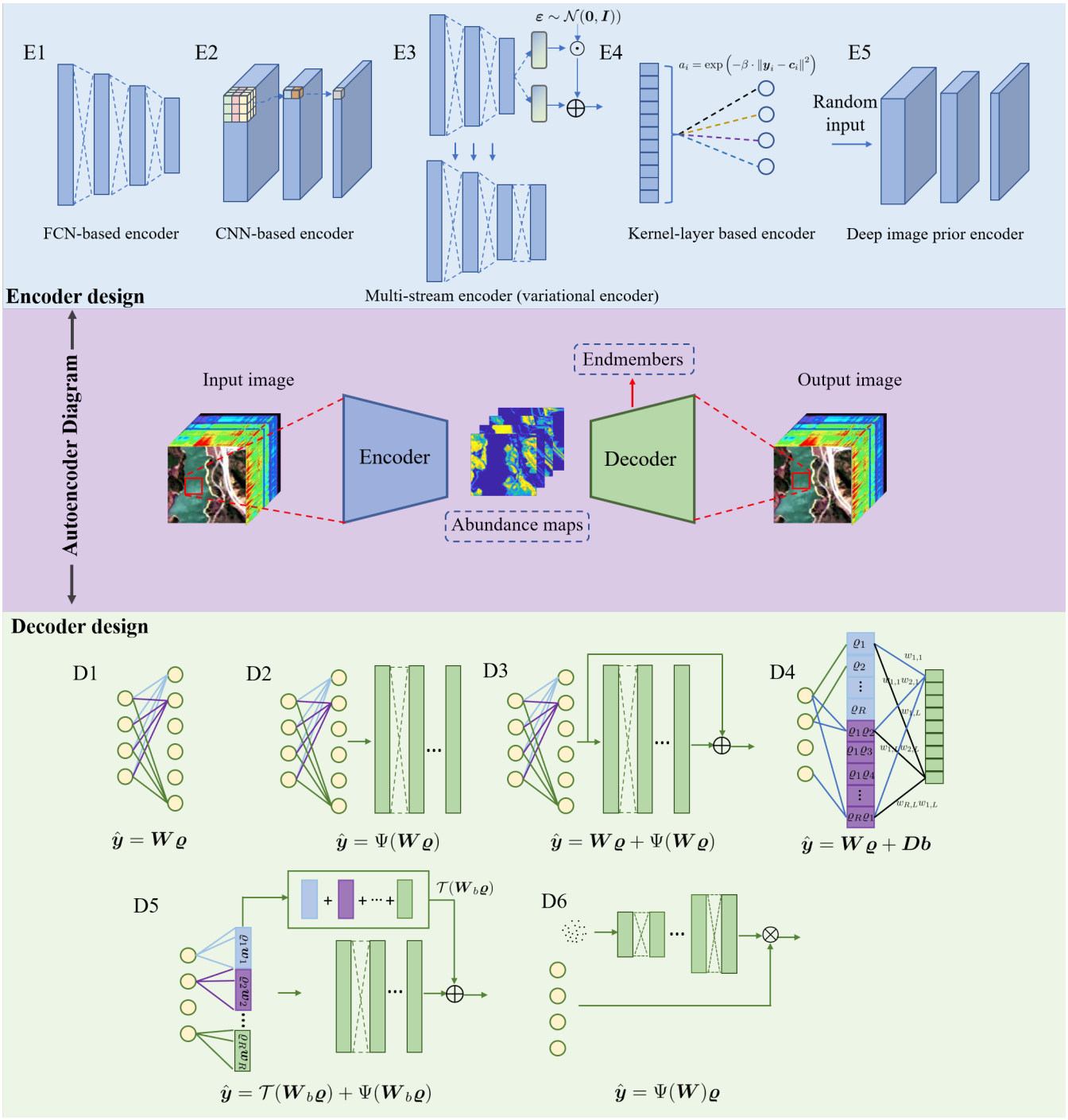


Fig. 3. This figure illustrates the structure of an autoencoder for hyperspectral unmixing, and a variety of selections of encoder and decoder designs.

Problem formulation III. In the following, rather than simply presenting the original autoencoders introduced in existing works as a whole, we explore the designs of encoders and decoders separately in order to demonstrate their extra flexibility and the potential of combining different designs. The basic structure of an autoencoder and typical encoder and decoder designs for unmixing are summarized in Fig. 3. We start with decoders that are in fact associated with mixture models. Then, we proceed with the encoders that implicitly benefit from data priors.

A. Design of structured decoders to learn models from data

Starting with the decoder of a single linear layer, we progressively demonstrate that the mixture models learned by an autoencoder are actually dependent on the decoder.

1) Linear decoder: The relationship between the input and output of a fully connected linear layer decoder is as follows:

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\Theta}_{\rm dec}}(\boldsymbol{\varrho}) = \boldsymbol{W}\boldsymbol{\varrho}. \tag{23}$$

As shown in D1 of Fig. 3, this form is consistent with the LMM defined in (5). Consequently, the learning process of the autoencoder, which minimizes the gap between $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$, leads to $\boldsymbol{\varrho}$ as an estimate of $\boldsymbol{a}$, and $\boldsymbol{W}$ as an estimate of the endmembers matrix $\boldsymbol{M}$. Despite the fact that the linear unmixing problem has been extensively studied with various classes of methods [14]–[19], this structure motivates the other sophisticated designs that learn structured models from the data.

2) Post-nonlinear decoder: With the linear decoder at hand, one can connect it to additional nonlinear layers in order to create the post-nonlinear structure:

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\Theta}_{\rm dec}}(\boldsymbol{\varrho}) = \Psi(\boldsymbol{W}\boldsymbol{\varrho}) \tag{24}$$

where Ψ denotes an implicit mapping function represented by the subsequent network layers with nonlinear activation functions. Parameters $\boldsymbol{\Theta}_{\rm dec}$ can be determined by training the autoencoder so that the nonlinear function is learnt from the data. Compared with the post-nonlinear mixture model defined in (16), Ψ is learnt to mimic the generative model $\mathcal{F}_{\rm post}$, and the endmembers are obtained from the weights of the first layer of the decoder [20]. The structure that forms this process is shown by D2 in Fig. 3.

3) Additive nonlinear decoders: Instead of adding subsequential nonlinear layers, a network branch can be added in parallel to the linear layer for constructing additive nonlinear structures. However, the parameters coupled in the linear and the nonlinear components should be carefully addressed.

- Additive post-nonlinear decoder: Passing the linear component through a nonlinear network and adding the output to the linear component, leads to the following result:

  $$\hat{\boldsymbol{y}} = \boldsymbol{W}\boldsymbol{\varrho} + \Psi(\boldsymbol{W}\boldsymbol{\varrho}). \tag{25}$$

  The structure that conducts this process is depicted in D3 of Fig. 3. Comparing the output of this decoder to the model in (17), the endmember can be estimated from the network weights of the linear component, and Ψ mimics $\mathcal{F}_{\rm add}$ in (17) once the network is learnt [21].
- Generalized additive nonlinear decoder: Constructing a decoder that mimics the generalized linear-mixture/nonlinear-fluctuation model (15) requires more specific designs. First, a block diagonal matrix $\boldsymbol{W}_b = \text{bdiag}\{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_R\}$ is considered for the weights of the first linear layer, which is then sparsely connected. Consequently, the output of this layer is a collection of $\boldsymbol{w}_i$ weighted by ϱ_i, that is:

  $$\boldsymbol{W}_b\boldsymbol{\varrho} = [\varrho_1\boldsymbol{w}_1^\top, \varrho_2\boldsymbol{w}_2^\top, \cdots, \varrho_R\boldsymbol{w}_R^\top]^\top. \tag{26}$$

On one hand, the linearly-mixed component of the model can be obtained by adding all R-spaced entries of $\boldsymbol{W}_b\boldsymbol{\varrho}$ such that:

$$\mathcal{T}(\boldsymbol{W}_b\boldsymbol{\varrho}) = \boldsymbol{W}\boldsymbol{\varrho} \tag{27}$$

with $\mathcal{T}: \mathbb{R}^{LR} \mapsto \mathbb{R}^L$ defined as $\mathcal{T}(\boldsymbol{x}) = \left[\sum_{i=1}^{R} x_{R(i-1)+1}, \sum_{i=1}^{R} x_{R(i-1)+2}, \cdots, \sum_{i=1}^{R} x_{R(i-1)+L}\right]^\top$. On the other hand, the weighted nonlinear fluctuation component is obtained by feeding $\boldsymbol{W}_b\boldsymbol{\varrho}$ to a nonlinear network representing Ψ. Finally, the sum of these two components:

$$\hat{\boldsymbol{y}} = \mathcal{T}(\boldsymbol{W}_b\boldsymbol{\varrho}) + \Psi(\boldsymbol{W}_b\boldsymbol{\varrho}) \tag{28}$$

is consistent with the model in (15). The endmembers are estimated by $\boldsymbol{W}$ and the nonlinear mixing function $\mathcal{F}_{\text{add}}$ is estimated by Ψ from training data [22]. The network structure that achieves this design is depicted in D5 of Fig. 3.

- Bilinear decoder: Designing a decoder to mimic the bilinear model defined in (7) leads to an autoencoder formulated as:

$$\hat{\boldsymbol{y}} = \boldsymbol{W}\boldsymbol{\varrho} + \boldsymbol{D}\boldsymbol{b}, \tag{29}$$

where $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \cdots, \boldsymbol{d}_c] \in \mathbb{R}^{L\times c}$ is a cross-product matrix involving all pairs of endmembers, i.e., $\boldsymbol{d}_{i,j} = \boldsymbol{w}_i \odot \boldsymbol{w}_j$ with $c = \frac{R(R-1)}{2}$, and $\boldsymbol{b} = [b_1, b_2, \cdots, b_c] \in \mathbb{R}^c$ denotes the interaction abundance matrix. The structure of this model can be seen from D4 in Fig. 3. Referring to (7), the weights of the decoder provide estimates of the endmembers and their cross-products. Vector $\boldsymbol{b}$ provides estimates of the cross-products of all pairs of abundances $\boldsymbol{b} = [\varrho_1\varrho_2, \varrho_2\varrho_3, \cdots, \varrho_R\varrho_1]$ or the latent features learnt from the data [23], [24].

4) Endmember generating decoder: Instead of being extracted from the weights of a decoder, the endmembers can also be estimated from a sub-network, that is,

$$\hat{\boldsymbol{y}} = f_{\Theta_{\text{end}}}(\boldsymbol{z})\boldsymbol{\varrho} = \Psi(\boldsymbol{z})\boldsymbol{\varrho} \tag{30}$$

where $f_{\Theta_{\text{end}}}$ represents the endmember generative model, with $\boldsymbol{\Theta}_{\text{end}}$ denoting the network parameters. This sub-network can be a regular or variational autoencoder that maps pixels to endmembers [25], [26] or a pretrained decoder to generate the endmembers from a latent vector [27]. Variable $\boldsymbol{z}$ denotes the input of this sub-network, which can be either pixels, candidate endmembers or latent representations. This decoder is also a powerful tool to cope with spectral variability [26], [27]. Compared with Problem formulation III, we identify $\boldsymbol{\Theta}_{\boldsymbol{M}} \leftarrow \boldsymbol{\Theta}_{\text{end}}$, leading to a hierarchical structure $\boldsymbol{M}(\boldsymbol{\Theta}_{\boldsymbol{M}})$.

B. Design of encoders to integrate priors from data

The encoder part is designed to map the input spectra into a latent representation space, which is often considered as the abundance space. The encoder implicitly acts as a regularizer relating abundances and spectra, and imposes a hierarchical structure on $\boldsymbol{a}$ with parameters $\boldsymbol{\Theta}_{\boldsymbol{a}}$, i.e., of the form $\boldsymbol{a}(\boldsymbol{\Theta}_{\boldsymbol{a}})$. The encoder part has been designed in the following ways in the literature.

1) FCN-based encoders: Early works [14]–[17] employ fully connected layers to construct the encoder. The latent representation of the k-th layer is given by

$$\begin{aligned} \boldsymbol{h}_k &= \sigma_k(\boldsymbol{U}_k \boldsymbol{h}_{k-1} + \boldsymbol{b}_k) \\ \text{with} \qquad \boldsymbol{h}_0 &= \boldsymbol{y} \end{aligned} \tag{31}$$

where σ_k represents the nonlinear activation function, and $\boldsymbol{U}_k$ and $\boldsymbol{b}_k$ stand for the weights and bias of the k-th layer, respectively. The output of the final layer is denoted by $\boldsymbol{\varrho}$ and is connected to the decoder. Some variants have also been designed based on this structure. For example, in [14], [15], a sparse encoder is used to enforce abundance sparsity. The work [15] applies a denoising encoder to address the problem of noisy data and outliers in hyperspectral data.

2) CNN-based encoders: Convolutional neural networks (CNNs) have also been used for designing the encoder [18], [19], [22]. Encoder inputs can be the entire image or overlapping patches. The 2D or 3D convolution operator operates on 3D image cubes $\mathbb{Y}_n$ as follows:

$$\begin{aligned} \mathbb{H}_k &= \sigma_k(\mathbb{H}_{k-1} \circledast \mathbb{U}_k + \mathbb{B}_k) \\ \text{with} \qquad \mathbb{H}_0 &= \mathbb{Y}_n, \end{aligned} \tag{32}$$

where $\mathbb{U}_k$ denotes a convolution kernel, and $\mathbb{B}_k$ represents the bias parameters. As 2D convolutional kernels may cause local spectral distortion, 3D-CNN encoders are used in [19], [22] to jointly learn spatial structures and local spectral features of 3D input images.

3) RNN encoders: Recurrent neural networks (RNNs), such as long short term memory (LSTM) networks, benefit from memory cells and are better suited for processing sequences. With their recurrent hidden states depending on previous steps, they can take full advantage of the sequential characteristics in hyperspectral data. The abundances are estimated with an LSTM to encode spectral correlation and band-to-band variability information [21].

4) Kernel-layer based encoder: Another class of approaches uses kernel layers to process hyperspectral data. Such layers can map the input pixels into a manifold and estimate the abundances according to a Riemannian metric. For example, in [24], the kernel layer is defined as:

$$a_i = \exp\left(-\beta \cdot \|\boldsymbol{y}_i - \boldsymbol{c}_i\|^2\right) \tag{33}$$

where β is the kernel parameter, and $\boldsymbol{c}_i$ the centroid related to the i-th endmember. The smaller the distance between the mixed pixel and the endmember centroid is, the larger the corresponding abundance is.

5) Variational encoders: A variational encoder encodes each input pixel with a distribution in the latent space as opposed to a single vector. In practice, it is common to consider Gaussian distributions parameterized by their means and covariance matrices. The work [16] uses a variational encoder to infer abundances. In [26], the probabilistic generative model is used to address endmembers variability and to provide more accurate abundance and endmember estimates.

6) Multi-branch encoders: Some works use multi-branch encoders to capture different features by sharing weights or concatenating features. In [28], an endmember guided encoder is used to transfer the endmember information into a parallel

abundance estimate encoder. The work [29] uses a dual branch encoder, where one branch consists of fully connected layers to extract spectral information, and the other branch consists of 2D convolution layers to capture spatial information.

`Remark:` The ANC and the ASC are usually addressed at the last layer of the encoder either by using specific activation functions (e.g. ReLU, Sigmoid, absolute operator) to enforce ASC followed by a normalization layer to enforce ANC, or by using SoftMax to enforce ANC and ASC simultaneously. They can also be addressed approximatively via a penalty term in the loss function.

`Example 5 (The 3D-CNN based autoencoder network for additive nonlinearity unmixing):` Considering the 3D-CNN based encoder (E2 in Fig. 3) and the generalized additive nonlinear decoder defined in (28) (D5 in Fig. 3), leads to the algorithm proposed in [22].

`Example 6 (A denoising autoencoder with sparsity for spectral unmixing):` Considering the denoising and sparse encoder and the linear decoder defined in (23) (E1 and D1 in Fig. 3), leads to the method proposed in [15].

`Example 7 (The kernelization and cross product layer based autoencoder for bilinear and post nonlinearity unmixing):` Considering the kernel layer based encoder (E4 in Fig. 3), the bilinear nonlinear decoder defined in (29) (D4 in Fig. 3) and the post nonlinear decoder defined in (25) (D3 in Fig. 3), yields the method proposed in [24].

`Example 8 (The probabilistic generative model based autoencoder for spectral unmixing with spectral variability):` A two-stream autoencoder framework with a probabilistic generative model (E3 and D6 in Fig. 3) that copes with spectral variability, leads to the method proposed in [26].

`Example 9 (Multitask based autoencoder network for bilinear spectral unmixing):` The combination of two deep autoencoders with a multitask learning framework leads to the method in [23]. One of the autoencoders models the linear components and estimates the endmembers as well as their abundances, while the other one models the bilinear components defined in (29) and estimates interaction abundances.

In addition to the regular autoencoders that have been described so far, there are also some variants of autoencoder frameworks that have been proposed for spectral unmixing. For instance, in [16], a group of autoencoders is first applied in order to address issues regarding outliers and generate a good initialization. Then, an autoencoder with a single-layer linear decoder is used to perform the process of unmixing. To improve the unmixing performance, two cascaded autoencoders are utilized in an end-to-end fashion to address cycle-consistency constraints in [30]. In [31], the endmembers are extracted using a geometric endmember extraction method, and the abundances are estimated using a deep image prior based on a CNN (depicted by E5 in Fig. 3). In [32], a two-stream network, with one stream for spectral information and the other one for exploiting spatial information, is collaboratively learnt to make full use of the spectral and spatial information. The work [33] constructs an attention network to fuse the information of light detection and ranging (LiDAR) data into the unmixing process.

Remarks on unmixing with deep autoencoder networks:

- The underlying unmixing model used in an autoencoder network is determined by the structure of the decoder, independently of the encoder.
- The encoder implicitly incorporates the inference relation from the observed spectra to the abundances, and can thus be considered as a prior for unmixing.
- Unmixing with an autoencoder falls into Problem formulation III with $\mathcal{F}$ and $\boldsymbol{\Theta}_M$ associated with the corresponding parts of the decoder and $\boldsymbol{\Theta}_{\boldsymbol{a}}$ learnt by the encoder network.

IV. Prior information learning with data-driven approaches

In addition to modeling the spectral mixture mechanism, DNNs are superior to conventional machine learning techniques in extracting information from data priors. It is intriguing to use DNNs to explicitly learn prior knowledge from data and use it with physical model-based inversion algorithms. Instead of designing sophisticated regularizers, the plug-and-play paradigm uses a denoiser with deep architectures to learn image prior and then incorporates the denoiser into an iterative optimization of the problem in (4). On the other hand, deep unrolling algorithms aim to unfold an iterative optimization of the problem in (4) with a specific regularizer into a trainable end-to-end deep network.

A. Incorporating data-driven priors with denoisers

Benefiting from the variable splitting technique, the plug-and-play framework makes it possible to exploit the priors learnt from training data to solve various image restoration problems. State-of-the-art denoising algorithms, especially fast-speed and powerful CNN-based denoisers are usually plugged into this framework as the proximity operator that captures the intrinsic spatial and spectral structures of hyperspectral images. By leveraging the flexibility of model-based optimization and the powerful learning ability of CNN, this framework has shown its great potential to perform unmixing tasks. Among three problem formulations, the form based on Problem formulation-I is considered here for illustration due to its simplicity. In the plug-and-play framework, an auxiliary variable $\boldsymbol{Z}$ is introduced in (4) to replace $\boldsymbol{A}$ in the regularization term, with an extra constraint $\boldsymbol{Z} = \boldsymbol{A}$ to ensure the equivalence of the optimization problems. Using the ADMM method results in the following iterative updates:

$$\left\{\boldsymbol{M}^{(k+1)}, \boldsymbol{A}^{(k+1)}\right\} = \underset{\boldsymbol{M},\{\boldsymbol{a}_i\}_{i=1}^N}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}\left(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i\right) + \frac{\rho}{2}\left\|\boldsymbol{A} - \boldsymbol{Z}^{(k)} + \boldsymbol{V}^{(k)}\right\|_F^2 \qquad \text{with} \quad \hat{\boldsymbol{y}}_i = \mathcal{F}(\boldsymbol{M}, \boldsymbol{a}_i) \tag{34a}$$

$$\text{s.t.} \quad \boldsymbol{M} \in \Omega_M, \text{ and } \boldsymbol{a}_i \in \Omega_a$$

$$\boldsymbol{Z}^{(k+1)} = \underset{\boldsymbol{Z}}{\operatorname{argmin}} \frac{\rho}{2}\left\|\boldsymbol{Z} - (\boldsymbol{A}^{(k+1)} + \boldsymbol{V}^{(k)})\right\|_F^2 + \mathcal{R}(\boldsymbol{Z}) \tag{34b}$$

$$\boldsymbol{V}^{(k+1)} = \boldsymbol{V}^{(k)} + \boldsymbol{A}^{(k+1)} - \boldsymbol{Z}^{(k+1)} \tag{34c}$$

where superscript $^{(k)}$ refers to the iteration index, ρ is the penalty parameter and $\boldsymbol{V}^{(k)}$ is the dual variable to be updated at iteration k. In this formulation, the regularizer $\mathcal{R}\big(\boldsymbol{A}\big)$ can be implicitly defined by a plugged denoising operator `Denoiser`, and (34b) can be seen as the denoising procedure of $\big(\boldsymbol{A}^{(k+1)}+\boldsymbol{V}^{(k)}\big)$ and replaced by:

$$\boxed{\boldsymbol{Z}^{(k+1)}=\texttt{Denoiser}\big(\boldsymbol{A}^{(k+1)}+\boldsymbol{V}^{(k)}\big)} \tag{35}$$

Generally, `Denoiser` can be any off-the-shelf denoising operator. This offers the opportunity of incorporating CNN-based denoisers with powerful prior learning ability into the physical model-based iterative optimization. In particular, the work [34] considers the LMM assumption in (5) with known $\boldsymbol{M}$ and incorporates conventional or deep learning based denoisers with two strategies, namely, a direct regularizer on $\boldsymbol{A}$, and a regularizer on reconstructed spectra, i.e., $\boldsymbol{MA}$. Considering the case of an unknown $\boldsymbol{M}$, the work in [35] jointly estimates $\boldsymbol{M}$ and exploits a handcrafted sparsity prior with the regularizer $||\boldsymbol{A}||_{2,1}$, and a denoiser prior with implicit regularizer $\mathcal{R}\big(\boldsymbol{A}\big)$ in the plug-and-play framework to produce more stable unmixing results. The work in [36] proposes a nonlinear plug-and-play unmixing method based on the general bilinear model in (7), and both traditional and CNN-based denoisers are plugged into the iterative optimization.

B. Unrolling iterative optimization into learnable deep architectures

Different from plug-and-play algorithms, the deep unrolling paradigm unfolds and truncates iterative optimization algorithms into trainable deep architectures. Considering the LMM in (5) with known $\boldsymbol{M}$, deep unrolling methods solve the unmixing problem in (10) with the $||\boldsymbol{A}||_1$ sparsity regularizer using iterative algorithms such as the iterative soft thresholding algorithm (ISTA) and ADMM algorithm. In [37], by introducing an auxiliary variable $\boldsymbol{z}$ such that $\boldsymbol{z}=\boldsymbol{a}$ along with a dual variable $\boldsymbol{v}$, the optimization problem is iteratively solved by the ADMM with the following steps:

$$\boldsymbol{a}^{(k+1)} = \boldsymbol{W}\boldsymbol{y}+\boldsymbol{B}(\boldsymbol{z}^{(k)}+\boldsymbol{v}^{(k)}) \tag{36a}$$

$$\boldsymbol{z}^{(k+1)} = \max(\text{soft}\big(\boldsymbol{a}^{(k+1)}-\boldsymbol{v}^{(k)},\lambda/\rho\big),0) \tag{36b}$$

$$\boldsymbol{v}^{(k+1)} = \boldsymbol{v}^{(k)}-\eta(\boldsymbol{a}^{(k+1)}-\boldsymbol{z}^{(k+1)}) \tag{36c}$$

where λ is the regularization parameter, ρ denotes the penalty parameter, η is a parameter offering additional flexibility, and $\boldsymbol{W}=(\boldsymbol{M}^\top\boldsymbol{M}+\mu\boldsymbol{I})^{-1}\boldsymbol{M}^\top$ and $\boldsymbol{B}=(\boldsymbol{M}^\top\boldsymbol{M}+\mu\boldsymbol{I})^{-1}\rho$. Operator soft$(\cdot)$ in (36b) is the soft-threshold operator given by $\text{soft}(\boldsymbol{x},\lambda/\rho)=\text{sign}(\boldsymbol{x})(|\boldsymbol{x}|-\lambda/\rho)_+$. The key feature of deep unrolling unmixing algorithms is to leverage a reparameterization of these iterative update steps through the consecutive DNN layers (denoted as $f_{\boldsymbol{a}}$, $f_{\boldsymbol{z}}$ and $f_{\boldsymbol{v}}$) with learnable parameters $\{\boldsymbol{W}^{(k+1)},\boldsymbol{B}^{(k+1)},\theta^{(k+1)},\eta^{(k+1)}\}$, respectively:

$$\boxed{\begin{aligned}
\boldsymbol{a}^{(k+1)} &= f_{\boldsymbol{a}}(\boldsymbol{z}^{(k)},\boldsymbol{v}^{(k)},\boldsymbol{y};\boldsymbol{W}^{(k+1)},\boldsymbol{B}^{(k+1)}) = \boldsymbol{W}^{(k+1)}\boldsymbol{y}+\boldsymbol{B}^{(k+1)}(\boldsymbol{z}^{(k)}+\boldsymbol{v}^{(k)})\\
\boldsymbol{z}^{(k+1)} &= f_{\boldsymbol{z}}(\boldsymbol{a}^{(k+1)},\boldsymbol{v}^{(k)};\theta^{(k+1)}) = \text{ReLU}(\boldsymbol{a}^{(k+1)}-\boldsymbol{v}^{(k)}-\theta^{(k+1)}\boldsymbol{I})\\
\boldsymbol{v}^{(k+1)} &= f_{\boldsymbol{v}}(\boldsymbol{a}^{(k+1)},\boldsymbol{z}^{(k+1)},\boldsymbol{v}^{(k)};\eta^{(k+1)}) = \boldsymbol{v}^{(k)}-\eta^{(k+1)}(\boldsymbol{a}^{(k+1)}-\boldsymbol{z}^{(k+1)})
\end{aligned}} \tag{37}$$

where $\theta^{(k+1)}$ is a learnable parameter which plays the role of λ/ρ and $\mathrm{ReLU}(\cdot)$ is a component-wise rectified linear unit operation. These specific-designed layers are then cascaded into an entire DNN architecture for unmixing, which can be trained with supervised loss functions in an end-to-end manner. For the blind unmixing problem in (11), the work in [38] builds a non-convex sparse NMF model using $||\boldsymbol{A}||_p$ with $0 < p < 1$ and designs an interpretable sparsity constrained NMF network based on the ISTA to jointly estimate $\boldsymbol{M}$ and $\boldsymbol{a}$.

V. Integrating loss learnt from data into physics-based inverse problems

Loss functions $\mathcal{L}$ in (4), (12) and (20) also play a crucial role in determining the unmixing performance. In addition to the data-driven model, data-learned loss functions have been proposed recently to improve the unmixing performance, particularly for Problem formulation III. The subsequent review focuses on autoencoder-based works as they mostly refer to Problem formulation III.

A. General geometric distances as cost functions

Geometric distances which are usually used as the essential loss in autoencoder based unmixing methods enforce the network to reconstruct the input image. Works [14]–[16] use the mean-squared error (MSE) to measure the similarity between the input and reconstructed pixels, that is,

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{N}\sum_{i=1}^{N} \left\|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\right\|^2. \tag{38}$$

In contrast to the MSE, which is sensitive to the scale of spectra, the spectral angle distance (SAD) and spectral information divergence (SID) are scale-invariant. The SAD considers the angle between two spectra as a spectral similarity measure:

$$\mathcal{L}_{\mathrm{SAD}} = \frac{1}{N}\sum_{i=1}^{N} \arccos\left(\frac{\langle \boldsymbol{y}_i, \hat{\boldsymbol{y}}_i\rangle}{\|\boldsymbol{y}_i\|\,\|\hat{\boldsymbol{y}}_i\|}\right). \tag{39}$$

The SID models the spectra as a probability distribution and considers the band-to-band spectral variability as the uncertainty existing in random variables to lower the gap between input and reconstructed pixels,

$$\mathcal{L}_{\mathrm{SID}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{p}_i \log\left(\frac{\boldsymbol{p}_i}{\hat{\boldsymbol{p}}_i}\right), \tag{40}$$

where $\boldsymbol{p} = \left(\boldsymbol{y}/\mathbf{1}^\top\boldsymbol{y}\right)$ and $\hat{\boldsymbol{p}} = \left(\hat{\boldsymbol{y}}/\mathbf{1}^\top\hat{\boldsymbol{y}}\right)$ represent the probability distribution vector of the input and estimated pixels respectively. Nevertheless, these conventional losses are defined at the pixel level based on basic statistical assumptions that disregard inherent image structures.

B. Deep learning metrics as cost functions

Generative adversarial networks (GANs) have recently been used to address the limitations of geometric distance loss functions. A GAN consists of two components, a generator and a discriminator, and models the distribution of data using deep neural networks. The generator $\mathcal{G}$ is trained to generate samples that are similar to real data in order to confuse the discriminator.

The discriminator $\mathcal{D}$ receives samples from the real data distribution (positive data) and samples from the generator (negative data) and attempts to distinguish between the real and generated samples. These two networks engage in a min-max adversarial game during the training process, that can be expressed as follows:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathrm{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{y})}[\log \mathcal{D}(\boldsymbol{z})] + \mathrm{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[\log(1-\mathcal{D}(\mathcal{G}(\boldsymbol{z}))]. \tag{41}$$

The architecture of unmixing methods using adversarial loss usually contains two terms, namely, an autoencoder and a discriminator. The former, considered to be the generator, aims to estimate the abundances and extract the endmembers. The latter is a deep binary classifier and is used to distinguish between generated and real samples. Thus, the discriminator may be able to transfer intrusive properties of real data that are useful for unmixing. The loss function utilized for generator training is always composed of two terms:

$$\mathcal{L}_{\mathcal{G}} = \mathcal{L}_{\mathrm{AE}} + \mathcal{L}_{\mathrm{adv}} \tag{42}$$

where $\mathcal{L}_{\mathrm{adv}} = \frac{1}{N}\sum_{i=1}^{N} \log \mathcal{D}\left(\boldsymbol{z}_i\right) + \frac{1}{N}\sum_{i=1}^{N} \log\left(1-\mathcal{D}\left(\hat{\boldsymbol{z}}_i\right)\right)$ is the loss function that is used to generate samples $\hat{\boldsymbol{z}}$ that confuse the discriminator, and $\mathcal{L}_{\mathrm{AE}}$ is the reconstruction loss function, e.g., the geometric cost function. The loss function for training the discriminator is defined as $\mathcal{L}_{\mathrm{D}} = -\mathcal{L}_{\mathrm{adv}}$. The work [39] introduces the Wasserstein distance calculated from the discriminator as a regularization term to characterize the distribution similarity between the input and reconstructed spectra. In [40], the adversarial network is utilized to model the abundance distribution.

Recent works have introduced the perception mechanism to discover the discrepancy between reconstructed pixels and their corresponding ground-truth (input pixels). This strategy allows us to relax pixel-level reconstruction and enhance the unmixing performance using an end-to-end process. The perceptual similarity can be viewed as a feature matching operator that compares the perceptual features of the reconstructed output pixels to those of the input pixels. The expression for the perceptual loss defined in a feature domain is:

$$\mathcal{L}_{\mathrm{Perceptual}} = \frac{1}{N}\sum_{i=1}^{N} \left\|\mathcal{P}(\boldsymbol{y}_i) - \mathcal{P}(\hat{\boldsymbol{y}}_i)\right\|^2, \tag{43}$$

where $\mathcal{P}$ stands for a feature extractor. For example, in [30], a cycle-consistency strategy is used to further refine the detailed information in the unmixing process. In [39], the features extracted from the hidden layers of a discriminator are used to ensure the consistency of high-level representations.

VI. Discussions and perspectives

This article reviewed hyperspectral unmixing methods with special attention focused on the integration of physics-based and data-driven methods. A mathematical framework was established to introduce a variety of existing works with a unified formalism. As summarized in this paper, notable advances have been recently achieved in this topic. Table II lists some features of the surveyed methods and summarizes some pros and cons of these methods. However, there are still several important aspects that merit further discussions and investigations.

TABLE II

FEATURES OF THE SURVEYED METHODS.

	Features						General pros and cons
Reference	**Architecture**	**Model**	**Loss**	**Blind**	**Initialization**	**Prior**	
[14]	shallow layer	LMM	MSE+SAD	√	VCA	ℓ_1 on abundances, ℓ_2 on endmembers	**Pro 1.** Integrating physics-based models in decoder design enables the methods to model complex mixture mechanism with physical characteristics. **Pro 2.** Convolutional filters are able to capture the spatial correlation. **Pro 3.** Integrating geometric distance with deep learning metric loss excavates inherent image structures. **Pro 4.** Proper initialization can speed up the convergence of the parameters and decrease the possibility of parameter overshooting. **Con 1.** Current methods require to pre-selecting the model type in advance and cannot involve multiple mixing models in complex scenes. **Con 2.** For spatial discontinuity regions, convolutional filters may blur the abundance maps. **Con 3.** Additional feature extraction networks of calculating the deep metric loss increase the number of network parameters.
[15]	untied denoising autoencoder		MSE		VCA+FCLS	$\ell_{2,1}$ on encoder weights	
[16]	stacked autoencoder, VAE		MSE		VCA+FCLS	minimum volume	
[17]	deep encoder		SAD		VCA+FCLS	n/a	
[18]	CNN encoder		SAD	×	randomly	n/a	
[19]	3D CNN encoder		SID		randomly	n/a	
[20]	post-nonlinear decoder		MSE		VCA	n/a	
[21]	LSTM encoder, additive post-nonlinear decoder		MSE		VCA	TV on abundances	
[22]	3D CNN encoder, generalized additive nonlinear decoder		MSE		VCA	TV on endmembers, ℓ_2 weight decay	
[23]	multitask learning, bilinear decoder	NLMM	MSE		VCA+FCLS	graph regularizer on abundances	
[24]	kernel-layer based encoder, bilinear decoder, post-nonlinear decoder		MSE		K-means centers	n/a	
[25]	two-stream autoencoder	LMM	SID-SAD		region-based VCA	TV on pixel	
[26]	VAE	spectral variability	MSE+KL+SAD		VCA	minimum volume	
[27]			MSE+KL		FCLS	TV on abundances	
[28]	multi-branch encoder	NLMM	MSE		scaled constrained least squares (SCLS)	ℓ_2 weight decay	
[29]	dual branch encoder	LMM	squared sine distance	√	VCA	$\ell_{\frac{1}{2}}$ on abundances	
[30]	cascaded autoencoder		cycle-consistency loss		VCA	ASC constraint	
[31]	deep image prior		MSE		a geometrical endmembers estimation	n/a	
[32]	two-stream autoencoder, collaborative learning		SAD		VCA+FCLS	$\ell_{\frac{1}{2}}$ on abundances	
[33]	attention networks, multimodal feature encoder		SAD		VCA	$\ell_{\frac{1}{2}}$ on abundances	
[34]	plug-and-play		MSE	×	FCLS	denoiser prior	
[35]			MSE	√	VCA+FCLS	denoiser prior, $\ell_{2,1}$ on abundance	
[36]		NLMM	MSE	×	FCLS	denoiser prior	
[37]	unroll ADMM	LMM	MSE+SAD+SID	√	VCA	ℓ_1 on abundances	
[38]	unroll NMF		MSE		VCA+FCLS	ℓ_p on abundances	
[39]	deep learning metric loss		SAD+ Wasserstein distance+ perceptual loss		randomly	n/a	
[40]	adversarial network		SAD+adversarial loss		VCA+FCLS	n/a	

Performance evaluation: Evaluating the performance of hyperspectral unmixing algorithms has always been nontrivial since few research works have been devoted to producing ground-truth data [41]. Performance evaluation becomes even more difficult when considering data-driven methods. Preliminary comparison results of several autoencoder-based unmixing methods with synthetic and real datasets have been recently provided in [42]. This study demonstrates the significance of using appropriate loss functions, nonlinear methods, spatial information, and proper initialization. However, thorough and objective evaluation studies are still missing.

Model selection: Real data usually involve multiple mixing models in complex scenes. It is anticipated that data-driven models, such as the generalized linear-mixture/nonlinear-fluctuation model, will exhibit superior scene-specific adaptability. Nevertheless, it is challenging, if not impossible, to pre-select an optimal model in advance. Using a mixing model with parallel multiple branches and an attention module to balance the contribution of each mixing model is a possible strategy to address this issue. Utilizing superpixel-based strategies and adopting distinct models within each superpixel and at its borders is an alternative solution.

A fully integrated framework: This article reviews the use of autoencoders to learn mixture models from data, the plug-and-play method to learn prior from data, and feature level losses from data. Existing works include one or two of above. Investigating an elegant strategy combining physics-based and data-driven methods that benefits from all of these factors is still missing. This requires deeper understanding and original methods in both aspects of optimization and neural network design.

Fast implementation: Compared to classical optimization methods, data-driven approaches using DNNs require significantly higher computational resources especially when multiple data-driven modules collaborate to perform unmixing. Fast implementation must be considered for practical use of unmixing algorithms. The formulation of the optimization problems, the design of the network structures, and the way of integrating them are crucial for enhancing the convergence speed of the learning process. A trained encoder can be used for a supervised or semi-supervised unmixing when the computation resource is limited. Further, a compute unified device architecture (CUDA) implementation can also be useful for executing parallel optimization-solving modules.

REFERENCES

[1] P. Ghamisi, N. Yokoya, J. Li, W. Liao, S. Liu, J. Plaza, B. Rasti, and A. Plaza, "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 37–78, 2017.

[2] N. Dobigeon, J. Tourneret, C. Richard, J. Bermudez, S. McLaughlin, and A. Hero, "Nonlinear unmixing of hyperspectral images: Models and algorithms," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 82–94, 2013.

[3] R. Borsoi, T. Imbiriba, J. Bermudez, C. Richard, J. Chanussot, L. Drumetz, J. Tourneret, A. Zare, and C. Jutten, "Spectral variability in hyperspectral data unmixing: A comprehensive review," *IEEE Geosci. Remote Sens. Mag.*, vol. 9, no. 4, pp. 223–270, 2021.

[4] J. Chen, C. Richard, and P. Honeine, "Nonlinear estimation of material abundances in hyperspectral images with ℓ_1-norm spatial regularization," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2654–2665, 2013.

[5] R. Heylen, M. Parente, and P. Gader, "A review of nonlinear hyperspectral unmixing methods," *IEEE J. Sel. Top. Appl. Earth Observat. Remote Sens.*, vol. 7, no. 6, pp. 1844–1868, 2014.

[6] A. Halimi, Y. Altman, 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.

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

[8] D. Heinz and C. 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.

[9] 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.

[10] J. Peng, W. Sun, H. Li, W. Li, X. Meng, C. Ge, and Q. Du, "Low-rank and sparse representation for hyperspectral image processing: A review," *IEEE Geosci. Remote Sens. Mag.*, vol. 10, no. 1, pp. 10–43, 2021.

[11] Z. Li, J. Chen, and S. Rahardja, "Kernel-based nonlinear spectral unmixing with dictionary pruning," *Remote Sensing*, vol. 11, no. 5, pp. 529, 2019.

[12] R. Heylen, D. Burazerovic, and P. Scheunders, "Non-linear spectral unmixing by geodesic simplex volume maximization," *IEEE J. Sel. Top. Sig. Process.*, vol. 5, no. 3, pp. 534–542, 2010.

[13] R. Heylen and P. Scheunders, "A distance geometric framework for nonlinear hyperspectral unmixing," *IEEE J. Sel. Top. Appl. Earth Observat. Remote Sens.*, vol. 7, no. 6, pp. 1879–1888, 2014.

[14] 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, 2018.

[15] 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, 2018.

[16] 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.

[17] B. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, "Hyperspectral unmixing using a neural network autoencoder," *IEEE Access*, vol. 6, pp. 25646–25656, 2018.

[18] 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.

[19] F. Khajehrayeni and H. Ghassemian, "Hyperspectral unmixing using deep convolutional autoencoders in a supervised scenario," *IEEE J. Sel. Top. Appl. Earth Observat. Remote Sens.*, vol. 13, pp. 567–576, 2020.

[20] M. Wang, M. Zhao, J. Chen, and S. Rahardja, "Nonlinear unmixing of hyperspectral data via deep autoencoder networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 9, pp. 1467–1471, 2019.

[21] M. Zhao, L. Yan, and J. Chen, "LSTM-DNN based autoencoder network for nonlinear hyperspectral image unmixing," *IEEE J. Sel. Top. Sig. Process.*, vol. 15, no. 2, pp. 295–309, 2021.

[22] 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.*, vol. 60, pp. 1–15, 2022.

[23] Y. Su, X. Xu, J. Li, H. Qi, P. Gamba, and A. Plaza, "Deep autoencoders with multitask learning for bilinear hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 10, pp. 8615–8629, 2020.

[24] K. T. Shahid and I. D. Schizas, "Unsupervised hyperspectral unmixing via nonlinear autoencoders," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2021.

[25] Q. Jin, Y. Ma, X. Mei, and J. Ma, "TANet: An unsupervised two-stream autoencoder network for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, 2021.

[26] S. Shi, M. Zhao, L. Zhang, Y. Altmann, and J. Chen, "Probabilistic generative model for hyperspectral unmixing accounting for endmember variability," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, 2022.

[27] R. Borsoi, T. Imbiriba, and J. Bermudez, "Deep generative endmember modeling: An application to unsupervised spectral unmixing," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 374–384, 2019.

[28] Z. Han, D. Hong, L. Gao, B. Zhang, and J. Chanussot, "Deep half-siamese networks for hyperspectral unmixing," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 11, pp. 1996–2000, 2020.

[29] Z. Hua, X. Li, Y. Feng, and L. Zhao, "Dual branch autoencoder network for spectral-spatial hyperspectral unmixing," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2021.

[30] L. Gao, Z. Han, D. Hong, B. Zhang, and J. Chanussot, "Cycu-net: Cycle-consistency unmixing network by learning cascaded autoencoders," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–14, 2021.

[31] B. Rasti, B. Koirala, P. Scheunders, and P. Ghamisi, "UnDIP: Hyperspectral unmixing using deep image prior," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, 2021.

[32] L. Qi, F. Gao, J. Dong, X. Gao, and Q. Du, "SSCU-Net: Spatial–spectral collaborative unmixing network for hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, 2022.

[33] Z. Han, D. Hong, L. Gao, J. Yao, B. Zhang, and J. Chanussot, "Multimodal hyperspectral unmixing: Insights from attention networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2022.

[34] M. Zhao, X. Wang, J. Chen, and W. Chen, "A plug-and-play priors framework for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2021.

[35] M. Zhao, T. Gao, J. Chen, and W. Chen, "Hyperspectral unmixing via nonnegative matrix factorization with handcrafted and learned priors," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2021.

[36] Z. Wang, L. Zhuang, L. Gao, A. Marinoni, B. Zhang, and M. K. Ng, "Hyperspectral nonlinear unmixing by using plug-and-play prior for abundance maps," *Remote Sensing*, vol. 12, no. 24, pp. 4117, 2020.

[37] C. Zhou and M. R. Rodrigues, "ADMM-based hyperspectral unmixing networks for abundance and endmember estimation," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–18, 2021.

[38] 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.*, vol. 60, pp. 1–16, 2021.

[39] A. Min, Z. Guo, H. Li, and J. Peng, "JMnet: Joint metric neural network for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–12, 2021.

[40] Q. Jin, Y. Ma, F. Fan, J. Huang, X. Mei, and J. Ma, "Adversarial autoencoder network for hyperspectral unmixing," *IEEE Trans. Neu. Net. Learn. Sys.*, pp. 1–15, 2021.

[41] M. Zhao, J. Chen, and Z. He, "A laboratory-created dataset with ground truth for hyperspectral unmixing evaluation," *IEEE J. Sel. Top. Appl. Earth Observat. Remote Sens.*, vol. 12, no. 7, pp. 2170–2183, 2019.

[42] B. Palsson, J. Sveinsson, and M. Ulfarsson, "Blind hyperspectral unmixing using autoencoders: A critical comparison," *IEEE J. Sel. Top. Appl. Earth Observat. Remote Sens.*, vol. 15, pp. 1340–1372, 2022.