remote sensing

MDPI

Article

Remote Sensing Image Denoising via Low-Rank Tensor Approximation and Robust Noise Modeling

Tian-Hui Ma, Zongben Xu and Deyu Meng *

School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an 710049, China; tianhuima@xjtu.edu.cn (T.-H.M.); zbxu@mail.xjtu.edu.cn (Z.X.)

* Correspondence: dymeng@mail.xjtu.edu.cn; Tel.: +86-181-0924-1418

Received: 14 March 2020; Accepted: 15 April 2020; Published: 17 April 2020

Abstract: Noise removal is a fundamental problem in remote sensing image processing. Most existing methods, however, have not yet attained sufficient robustness in practice, due to more or less neglecting the intrinsic structures of remote sensing images and/or underestimating the complexity of realistic noise. In this paper, we propose a new remote sensing image denoising method by integrating intrinsic image characterization and robust noise modeling. Specifically, we use low-Tucker-rank tensor approximation to capture the global multi-factor correlation within the underlying image, and adopt a non-identical and non-independent distributed mixture of Gaussians (non-i.i.d. MoG) assumption to encode the statistical configurations of the embedded noise. Then, we incorporate the proposed image and noise priors into a full Bayesian generative model and design an efficient variational Bayesian algorithm to infer all involved variables by closed-form equations. Moreover, adaptive strategies for the selection of hyperparameters are further developed to make our algorithm free from burdensome hyperparameter-tuning. Extensive experiments on both simulated and real multispectral/hyperspectral images demonstrate the superiority of the proposed method over the compared state-of-the-art ones.

Keywords: remote sensing image denoising; low-rank tensor approximation; noise modeling; variational inference

1. Introduction

Remote sensing images, such as multispectral images (MSIs) and hyperspectral images (HSIs), provide abundant spatial and spectral information of real scenes and play a central role in many real-world applications, such as urban planning, surveillance, and environmental monitoring. Unfortunately, during the acquisition process, remote sensing images are often corrupted by various kinds of noise, such as Gaussian noise, speckle noise, and stripe. Image denoising aims to recover an underlying clean image from its noisy observation, which is a fundamental problem in remote sensing image processing. To obtain effective signal-noise separations, denoising methods usually rely on some prior assumptions imposed on the image and noise components.

One of the key issues in denoising methods is the rational design of an image prior, which encourages some expected properties of the denoised image. As a significant property of remote sensing images, low-rankness means that high-dimensional image data lie in a low-dimensional subspace, which can also be considered to be sparsity over a learned basis. Methods based on low-rankness along this line can be categorized into two classes: matrix-based and tensor-based ones. Matrix-based methods perform low-rank matrix approximation on the unfolding (tensor matricization) of the noisy image along the spectral mode. To obtain an efficient low-rank solution, low-rank matrix factorization methods factorize the objective matrix into a product of two flat ones [1–9]; rank minimization methods penalize some surrogates of the rank function, such as the convex

envelope nuclear norm [10–12] or tighter non-convex metrics, e.g., log-determinant penalty [13], Schatten p-norm [14,15], γ-norm (Laplace function) [16], and truncated/weighted nuclear norm [17,18]. These matrix-based methods, however, can capture only the spectral correlation but ignore the global multi-factor correlation in remote sensing images, which usually leads to suboptimal results under severe noise corruption. On the other hand, tensor-based methods explicitly model the underlying image as a low-rank tensor, by solving a tensor decomposition model or minimizing the corresponding induced tensor rank [19]. Representative works include CANDECOMP/PARAFAC (CP) decomposition with CP rank [20–22], Tucker decomposition with Tucker rank [23–27], tensor singular value decomposition (t-SVD) with tubal rank [28,29], and tensor train (TT) decomposition with TT rank [30–32]. Considering that each tensor decomposition represents a specific type of high-dimensional data structure, recent works attempt to combine the merits of different low-rank tensor models, such as the hybrid CP-Tucker model [33] and the Kronecker-basis-representation (KBR)-based tensor sparsity measure [34,35]. By characterizing the correlations across both spatial and spectral modes, the above tensor-based methods have the advantage of preserving the intrinsic multilinear structure of remote sensing images, achieving state-of-the-art denoising performance.

Another critical issue in current denoising methods is the choice of a noise prior, which characterizes the statistical properties of the data noise. This is generally realized by imposing certain assumptions on the noise distribution, leading to specific loss functions between the noisy image and the denoising result. Two traditional noise priors are the Gaussian prior [1,2,21,24] (L_2-norm loss) and the Laplacian prior [5,36,37] (L_1-norm loss), which are widely used for suppressing dense noise and sparse noise (outlier), respectively. A combination of Gaussian and Laplacian priors [12,27,29,38] ($L_1 + L_2$ loss) is commonly considered in mixed noise removal. However, these priors are generally not flexible enough to fit the noise in real applications, whose distributions are much more complicated than Gaussian/Laplacian or a simple mixture of them. To handle such complex noise, several works model the noise with a mixture of Gaussians (MoG) distribution [3,4,8] (weighted-L_2 loss), due to its universal approximation capability to any continuous probability density function [39]. Later, MoG has been generalized to a mixture of exponential power (MoEP) distribution [6] (weighted-L_p loss) for further flexibility and adaptivity. Despite the sophistication of the above priors, they all assume that the noise is independent and identically distributed (i.i.d.), which is still limited in handling realistic noise with non-i.i.d. statistical structures. In remote sensing images, the noise across different bands always exhibits evident distinctions in configuration and magnitude. To encode such noise characteristics, recent works impose non-i.i.d. assumptions on the noise distribution, such as non-i.i.d. MoG [7] and Dirichlet process Gaussian mixture [9,40], resulting in better noise fitting capability and higher denoising accuracy.

Some attempts have been presented to combine the advantages of recent developments in image characterization and noise modeling. To the best of our knowledge, only several studies are constructed as follows. Chen et al. [41] proposed a robust tensor factorization method based on CP decomposition and MoG noise assumption. However, their model does not consider uncertainty information of latent variables, such as the CP factor matrices and the MoG parameters, and thus is prone to overfitting due to point estimations of these variables by optimization-based approaches. To overcome this defect, Luo et al. [42] formulated the robust CP decomposition with MoG noise assumption as a full Bayesian model, in which all latent variables are given prior distributions and inferred under a variational Bayesian framework. Considering that CP decomposition cannot well capture the correlations along different tensor modes, Chen et al. [43] further integrated Tucker decomposition and MoG noise modeling into a generalized robust tensor factorization framework. However, this method also suffers from the overfitting problem and requires some critical hyperparameters to be manually specified, such as the tensor rank and the number of MoG components. Moreover, all the above methods impose an i.i.d. assumption on the data noise, which still under-estimates the complexity of realistic noise and thus leaves room for further improvement.

To overcome the aforementioned issues, in this paper we propose a new remote sensing image denoising method by taking into consideration the intrinsic properties of both remote sensing images and realistic noise. The main contribution of this work is summarized below.

- We formulate the image denoising problem as a full Bayesian generative model, in which a low-Tucker-rank image prior is exploited to characterize the intrinsic low-rank tensor structure of the underlying image, and a non-i.i.d. MoG noise prior is adopted to encode the complex and distinct statistical structures of the embedded noise.
- We design a variational Bayesian algorithm for an efficient solution to the proposed model, where each variable can be updated in closed-form. Moreover, we develop adaptive strategies for the selection of involved hyperparameters, to make our algorithm free from burdensome hyperparameter-tuning.
- We conduct extensive denoising experiments on both simulated and real MSIs/HSIs, and the results show the superiority of the proposed method over the compared state-of-the-art ones.

The rest of the paper is organized as follows. Section 2 introduces some notation used throughout the paper. Section 3 describes the proposed model and the corresponding variational inference algorithm. Section 4 presents experimental results and discussions. Section 5 concludes the paper.

2. Notation

We use boldface Euler script letters for tensors, e.g., $\boldsymbol{\mathcal{A}}$, boldface uppercase letters for matrices, e.g., $\mathbf{A}$, boldface lowercase letters for vectors, e.g., $\mathbf{a}$, and lowercase letters for scalars, e.g., a. In particular, we use $\mathbf{I}$, $\mathbf{0}$, and $\mathbf{1}$ for identity matrices, arrays of all zeros, and arrays of all ones, respectively. We use a pair of lowercase and uppercase letters for an index and its upper bound, e.g., $i = 1, \ldots, I$. We use Matlab expressions to denote elements and subarrays, e.g., $\mathbf{a}(i)$, $\mathbf{A}(i,:)$, and $\boldsymbol{\mathcal{A}}(i,:,:)$.

Given a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ ($\boldsymbol{\mathcal{A}}$ reduces to a matrix when $D = 2$ or a vector when $D = 1$). The Frobenius norm and the 1-norm of $\boldsymbol{\mathcal{A}}$ are, respectively, defined as

$$\|\boldsymbol{\mathcal{A}}\|_F := \sqrt{\sum_{i_1,\ldots,i_D=1}^{I_1,\ldots,I_D} \boldsymbol{\mathcal{A}}(i_1,\ldots,i_D)^2} \quad \text{and} \quad \|\boldsymbol{\mathcal{A}}\|_1 := \sum_{i_1,\ldots,i_D=1}^{I_1,\ldots,I_D} |\boldsymbol{\mathcal{A}}(i_1,\ldots,i_D)|.$$

For a given dimension $d \in \{1,\ldots,D\}$, the mode-d unfolding of $\boldsymbol{\mathcal{A}}$ is denoted as $\text{unfold}_d(\boldsymbol{\mathcal{A}})$ or, more compactly, as $\mathbf{A}_{(d)}$, whose size is $I_d \times (I_1 \ldots I_{d-1} I_{d+1} \ldots I_D)$. The inverse process is denoted as $\text{fold}_d(\mathbf{A}_{(d)}) := \boldsymbol{\mathcal{A}}$. More precisely, the tensor element $\boldsymbol{\mathcal{A}}(i_1,\ldots,i_D)$ maps to the matrix element $\mathbf{A}_{(d)}(i'_1, i'_2)$ satisfying

$$i'_1 = i_d \quad \text{and} \quad i'_2 = 1 + \sum_{k=1,k\neq d}^{D} (i_k - 1) \prod_{m=1,m\neq d}^{k-1} I_m,$$

see [19] for more details. The mapping between $(i_1,\ldots,i_D)$ and (i'_1, i'_2) is denoted as

$$\boldsymbol{\mathcal{A}}^{I_1\times\cdots\times I_D}(i_1,\ldots,i_D) = \mathbf{A}_{(d)}^{I_d \times (I_1\ldots I_{d-1} I_{d+1}\ldots I_D)}(i'_1, i'_2).$$

The Tucker rank of $\boldsymbol{\mathcal{A}}$ is defined as a vector consisting of the ranks of its unfoldings, i.e.,

$$\text{rank}(\boldsymbol{\mathcal{A}}) := (\text{rank}(\mathbf{A}_{(1)}), \text{rank}(\mathbf{A}_{(2)}), \ldots, \text{rank}(\mathbf{A}_{(D)})).$$

Additional notation is defined where it occurs.

3. Tucker Rank Minimization with Non-i.i.d. MoG Noise Modeling

This section is divided into three parts. Section 3.1 formulates the denoising problem as a full Bayesian model named Tucker rank minimization with non-i.i.d. MoG noise modeling (NMoG-Tucker).

Section 3.2 presents a variational inference algorithm for solving the proposed model. Section 3.3 discusses the selection of hyperparameters involved in our model.

3.1. Bayesian Model Formulation

Let $\mathcal{Y}, \mathcal{X} \in \mathbb{R}^{I\times J\times K}$ denote the noisy image and the underlying clean image, respectively. To characterize the low-Tucker-rank prior of remote sensing images, we consider the following low-rank matrix factorization of the mode-d unfoldings of $\mathcal{Y}$ ($d = 1, 2, 3$):

$$\mathbf{Y}_{(d)} = \mathbf{U}_d\mathbf{V}_d^T + \mathbf{N}_d, \tag{1}$$

where $\mathbf{U}_d \in \mathbb{R}^{I_d\times R_d}$ ($\{I_d\}_{d=1}^3 = \{I, J, K\}$) and $\mathbf{V}_d \in \mathbb{R}^{J_d\times R_d}$ ($\{J_d\}_{d=1}^3 = \{JK, IK, IJ\}$) are factor matrices with column number $R_d \leq \min(I_d, J_d)$, and $\mathbf{N}_d$ denotes the noise embedded in $\mathbf{Y}_{(d)}$. Below we formulate (1) as a full Bayesian model by imposing prior distributions on the involved variables.

Prior of the noise $\mathbf{N}_d$. We impose a non-i.i.d. MoG prior on $\mathbf{N}_d$ to characterize the complex structure of realistic noise. For simplicity of presentation, let us consider the tensor (We remark that $\mathcal{N}_d := \text{fold}_d(\mathbf{N}_d)$ depends on the dimension d because our model (1) does not enforce the equality between the low-rank components of $\mathcal{Y}$ along different modes, i.e., $\{\text{fold}_d(\mathbf{U}_d\mathbf{V}_d^T)\}_{d=1}^3$, considering that the low-rankness degrees along different modes are generally not the same) $\mathcal{N}_d := \text{fold}_d(\mathbf{N}_d) \in \mathbb{R}^{I\times J\times K}$. We assume that each element in the k-th band of $\mathcal{N}_d$ follows a MoG distribution

$$\mathcal{N}_d(i, j, k) \sim \textstyle\sum_{l=1}^{L_d} \mathbf{\Pi}_d(k, l)\mathcal{N}(\mathcal{N}_d(i, j, k)|0, \boldsymbol{\tau}_d(l)^{-1}), \tag{2}$$

where L_d is the number of Gaussian components, $\mathbf{\Pi}_d(k, :) \in \mathbb{R}^{L_d}$ is the mixing proportion satisfying $\mathbf{\Pi}_d(k, l) > 0$ and $\sum_{l=1}^{L_d} \mathbf{\Pi}_d(k, l) = 1$, and $\boldsymbol{\tau}_d \in \mathbb{R}^{L_d}$ contains precisions of the Gaussian components. By introducing an indicator variable $\mathcal{Z}_d \in \{0, 1\}^{I\times J\times K\times L_d}$, we rewrite (2) as the following two-level generative process [44]:

$$\begin{aligned} &\mathcal{N}_d(i, j, k) \sim \textstyle\prod_{l=1}^{L_d} \mathcal{N}(\mathcal{N}_d(i, j, k)|0, \boldsymbol{\tau}_d(l)^{-1})^{\mathcal{Z}_d(i,j,k,l)}, \\ &\mathcal{Z}_d(i, j, k, :) \sim \text{Multinomial}(\mathcal{Z}_d(i, j, k, :)|\mathbf{\Pi}_d(k, :)), \end{aligned} \tag{3}$$

where $\mathcal{Z}_d(i, j, k, :) \in \{0, 1\}^{L_d}$ with $\sum_{l=1}^{L_d} \mathcal{Z}_d(i, j, k, l) = 1$ follows a multinomial distribution parameterized by $\mathbf{\Pi}_d(k, :)$. Then, we impose conjugate priors on $\boldsymbol{\tau}_d$ and $\mathbf{\Pi}_d$ to obtain a complete Bayesian model,

$$\boldsymbol{\tau}_d(l) \sim \text{Gamma}(\boldsymbol{\tau}_d(l)|a_0, b_0), \tag{4}$$

$$\mathbf{\Pi}_d(k, :) \sim \text{Dirichlet}(\mathbf{\Pi}_d(k, :)|\alpha_0\mathbf{1}), \tag{5}$$

where $\text{Gamma}(\cdot|a_0, b_0)$ denotes the Gamma distribution with parameters a_0 and b_0, and $\text{Dirichlet}(\cdot|\alpha_0\mathbf{1})$ denotes the Dirichlet distribution parameterized by $\alpha_0\mathbf{1} \in \mathbb{R}^{L_d}$.

The proposed prior can characterize the following intrinsic properties of realistic noise.

- First, noise in each band exhibits complex statistical properties, which cannot be well captured by simple distributions such as Gaussian or Laplacian. We model the noise in each band by an i.i.d. MoG distribution, which is a universal approximator to any continuous distribution.
- Second, noise across different bands is non-identical in terms of structure and extent, due to sensor malfunctions and atmospheric conditions. This band-noise-distinctness nature is encoded by the band-dependent mixing proportion in MoG, leading to a non-i.i.d. noise distribution.
- Third, there is a strong correlation among the noise distributions in all bands, since real-life noise corruption is generally attributed to only a few main factors. In the proposed prior, the noise correlation is reflected by the fact that the MoG distributions of different bands share the same set of Gaussian components.

Prior of the factor matrices $\mathbf{U}_d$ and $\mathbf{V}_d$. Inspired by the sparse Bayesian learning principle [45], we assume that the columns of $\mathbf{U}_d$ and $\mathbf{V}_d$ are generated from the following Gaussian distributions:

$$\mathbf{U}_d(:,r) \sim \mathcal{N}(\mathbf{U}_d(:,r)|\mathbf{0}, \gamma_d(r)^{-1}\mathbf{I}), \tag{6}$$
$$\mathbf{V}_d(:,r) \sim \mathcal{N}(\mathbf{V}_d(:,r)|\mathbf{0}, \gamma_d(r)^{-1}\mathbf{I}), \tag{7}$$

where $\gamma_d \in \mathbb{R}^{R_d}$ denotes precisions following the conjugate prior

$$\gamma_d(r) \sim \text{Gamma}(\gamma_d(r)|c_0, d_0). \tag{8}$$

Prior of the solution $\mathcal{X}$. Given the learned low-rank components along three modes, we assume that each element in $\mathcal{X}$ is generated from the following weighted multiplication of Gaussian distributions:

$$p(\mathcal{X}(i,j,k)) = c\prod_{d=1}^{3}\mathcal{N}(\mathcal{X}(i,j,k)|\mathbf{U}_d(i_d,:)\mathbf{V}_d(j_d,:)^T, \xi^{-1})^{\mathbf{w}(d)}, \tag{9}$$

where (i_d, j_d) maps to (i,j,k) such that $\mathbf{A}_{(d)}^{I_d\times J_d}(i_d,j_d) = \mathcal{A}^{I\times J\times K}(i,j,k)$, ξ denotes the precision of the Gaussian distributions, $\mathbf{w} \in \mathbb{R}^3$ contains weights of the three modes satisfying $\mathbf{w}(d) > 0$ and $\sum_{d=1}^{3}\mathbf{w}(d) = 1$, and c is a normalization constant.

Full Bayesian model and posterior. We can construct a full Bayesian model by combining (1)–(9); the corresponding graphical model is shown in Figure 1. Then, the goal is to infer the posterior of all involved variables, which can be expressed as

$$\begin{aligned} &p(\mathcal{X}, \{\mathbf{U}_d, \mathbf{V}_d, \gamma_d, \tau_d, \mathcal{Z}_d, \mathbf{\Pi}_d\}_{d=1}^3|\mathcal{Y}) \\ &\propto p(\mathcal{X}, \{\mathbf{U}_d, \mathbf{V}_d, \gamma_d, \tau_d, \mathcal{Z}_d, \mathbf{\Pi}_d\}_{d=1}^3, \mathcal{Y}) \\ &= p(\mathcal{X}|\{\mathbf{U}_d, \mathbf{V}_d\}_{d=1}^3)\prod_{d=1}^{3}\left\{p(\mathbf{Y}_{(d)}|\mathbf{U}_d, \mathbf{V}_d, \tau_d, \mathcal{Z}_d)p(\mathbf{U}_d|\gamma_d)p(\mathbf{V}_d|\gamma_d)p(\gamma_d)p(\tau_d)p(\mathcal{Z}_d|\mathbf{\Pi}_d)p(\mathbf{\Pi}_d)\right\}. \end{aligned} \tag{10}$$

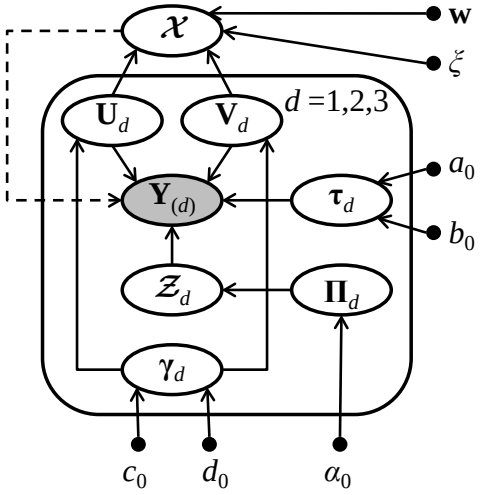

Figure 1. Graphical model of NMoG-Tucker. Hollow nodes, shadowed nodes, and small solid nodes denote unobserved variables, observed data, and hyperparameters, respectively; a solid arrow from node a to node b indicates the explicit conditional distribution $p(b|a)$; a dashed arrow from node a to node b implies that b is implicitly conditioned on a; the box is a compact representation indicating that there are three sets of variables corresponding to the three tensor modes.

Optimization-based interpretation. From an optimization perspective, maximizing the posterior (10) is equivalent to minimizing its negative logarithm, i.e.,

$$
\begin{aligned}
&-\ln p(\boldsymbol{\mathcal{X}}, \{\mathbf{U}_d, \mathbf{V}_d, \boldsymbol{\gamma}_d, \boldsymbol{\tau}_d, \boldsymbol{\mathcal{Z}}_d, \boldsymbol{\Pi}_d\}_{d=1}^3 | \boldsymbol{\mathcal{Y}}) \\
=& -\sum_{d=1}^3 \Big\{ \mathbf{w}(d) \sum_{i,j,k} \ln \mathcal{N}(\boldsymbol{\mathcal{X}}(i,j,k) | \mathbf{U}_d(i_d,:)\mathbf{V}_d(j_d,:)^T, \xi^{-1}) \\
&\quad + \sum_{i,j,k,l} \boldsymbol{\mathcal{Z}}_d(i,j,k,l) \ln \mathcal{N}(\boldsymbol{\mathcal{Y}}(i,j,k) | \mathbf{U}_d(i_d,:)\mathbf{V}_d(j_d,:)^T, \boldsymbol{\tau}_d(l)^{-1}) \\
&\quad + \sum_r \ln \mathcal{N}(\mathbf{U}_d(:,r) | \mathbf{0}, \boldsymbol{\gamma}_d(r)^{-1}\mathbf{I}) + \sum_r \ln \mathcal{N}(\mathbf{V}_d(:,r) | \mathbf{0}, \boldsymbol{\gamma}_d(r)^{-1}\mathbf{I}) \\
&\quad + \sum_r \ln \text{Gamma}(\boldsymbol{\gamma}_d(r) | c_0, d_0) + \sum_l \ln \text{Gamma}(\boldsymbol{\tau}_d(l) | a_0, b_0) \\
&\quad + \sum_{i,j,k} \ln \text{Multinomial}(\boldsymbol{\mathcal{Z}}_d(i,j,k,:) | \boldsymbol{\Pi}_d(k,:)) + \sum_k \ln \text{Dirichlet}(\boldsymbol{\Pi}_d(k,:) | \alpha_0 \mathbf{1}) \Big\}
\end{aligned} \tag{11}
$$

$$
\begin{aligned}
=& \sum_{d=1}^3 \Big\{ \frac{\mathbf{w}(d)\xi}{2} \|\mathbf{X}_{(d)} - \mathbf{U}_d\mathbf{V}_d^T\|_F^2 + \frac{1}{2}\|\text{unfold}_d(\boldsymbol{\mathcal{H}}_d) \odot (\mathbf{Y}_{(d)} - \mathbf{U}_d\mathbf{V}_d^T)\|_F^2 + \frac{1}{2}\|\mathbf{U}_d \text{diag}(\boldsymbol{\gamma}_d)\|_F^2 \\
&+ \frac{1}{2}\|\mathbf{V}_d \text{diag}(\boldsymbol{\gamma}_d)\|_F^2 + \sum_r \left(d_0 \boldsymbol{\gamma}_d(r) - \left(\frac{I_d + J_d}{2} + c_0 - 1 \right) \ln \boldsymbol{\gamma}_d(r) \right) \\
&+ \sum_l \left(b_0 \boldsymbol{\tau}_d(l) - \left(\frac{1}{2} \sum_{i,j,k} \boldsymbol{\mathcal{Z}}_d(i,j,k,l) + a_0 - 1 \right) \ln \boldsymbol{\tau}_d(l) \right) \\
&+ \sum_{i,j,k,l} \left(\frac{1}{2} \ln(2\pi) - \ln \boldsymbol{\Pi}_d(k,l) \right) \boldsymbol{\mathcal{Z}}_d(i,j,k,l) + \sum_{k,l} (1 - \alpha_0) \ln \boldsymbol{\Pi}_d(k,l) \Big\},
\end{aligned} \tag{12}
$$

where $\boldsymbol{\mathcal{H}}_d \in \mathbb{R}^{I \times J \times K}$ contains the noise level estimations with $\boldsymbol{\mathcal{H}}_d(i,j,k) = \sqrt{\sum_{l=1}^{L_d} \boldsymbol{\mathcal{Z}}_d(i,j,k,l)\boldsymbol{\tau}_d(l)}$, $\odot$ denotes the element-wise multiplication, $\text{diag}(\boldsymbol{\gamma}_d)$ denotes the diagonal matrix with the elements of $\boldsymbol{\gamma}_d$ on its main diagonal. Below we illustrate the origin and the effect of each term in (12).

- The first ℓ_2-norm term is derived from the weighted multiplication of Gaussians prior on the solution $\boldsymbol{\mathcal{X}}$ (9). It forms $\boldsymbol{\mathcal{X}}$ by penalizing the Euclidean distances between the unfoldings $\{\mathbf{X}_{(d)}\}_{d=1}^3$ and the low-rank components $\{\mathbf{U}_d\mathbf{V}_d^T\}_{d=1}^3$.
- The second weighted-ℓ_2-norm term is derived from the non-i.i.d. MoG prior on the noise $\mathbf{N}_d$ (3). It serves as a spatially varying loss function that suppresses the noise according to the local noise level estimations embedded in the weight matrix $\text{unfold}_d(\boldsymbol{\mathcal{H}}_d)$.
- The third and the fourth weighted-ℓ_2-norm terms are derived from the Gaussian priors on the factor matrices $\mathbf{U}_d$ and $\mathbf{V}_d$ (6,7). They promote the joint group sparsity of $\{\mathbf{U}_d, \mathbf{V}_d\}$ in the unit of column pair $\{\mathbf{U}_d(:,r), \mathbf{V}_d(:,r)\}$, which implies the sparsity of $\mathbf{U}_d\mathbf{V}_d^T$ under rank-one bases $\{\mathbf{U}_d(:,r)\mathbf{V}_d(:,r)^T\}_{r=1}^{R_d}$, i.e., the low-rankness of $\mathbf{U}_d\mathbf{V}_d^T$.
- The remainder terms are derived from the priors on the variables $\{\boldsymbol{\gamma}_d, \boldsymbol{\tau}_d, \boldsymbol{\mathcal{Z}}_d, \boldsymbol{\Pi}_d\}$ and provide them with suitable regularization.

3.2. Approximate Variational Inference

We use the variational Bayesian (VB) method [44] to obtain an approximate inference of the posterior (10), since the exact solution is computationally intractable. Below we briefly introduce the general framework of VB, and then present the inference results for our model.

General framework of VB. Denoting by $\boldsymbol{\theta}$ unobserved variables and by $\mathbf{D}$ observed data, VB aims to seek a variational distribution $q(\boldsymbol{\theta})$ to approximate the true posterior $p(\boldsymbol{\theta}|\mathbf{D})$, by minimizing the Kullback–Leibler (KL) divergence between q and p, i.e.,

$$
\min_{q \in \mathcal{C}} \text{KL}(q \| p) := -\int q(\boldsymbol{\theta}) \ln \left\{ \frac{p(\boldsymbol{\theta}|\mathbf{D})}{q(\boldsymbol{\theta})} \right\} \text{d}\boldsymbol{\theta}, \tag{13}
$$

where $\mathcal{C}$ imposes certain restrictions on q to make the minimization tractable. In general, q is restricted to have the factorization $q(\boldsymbol{\theta}) = \prod_i q_i(\boldsymbol{\theta}_i)$, where $\{\boldsymbol{\theta}_i\}$ are disjoint groups of the variables in $\boldsymbol{\theta}$. Under this assumption, one can approach the solution to (13) in an iterative way, by alternatively

minimizing $\mathrm{KL}(q\|p)$ with respect to each $q_i(\boldsymbol{\theta}_i)$ while keeping the others fixed. More precisely, q_i can be calculated by the following closed-form solution:

$$q_i(\boldsymbol{\theta}_i) = \frac{\exp(\langle \ln p(\boldsymbol{\theta}, \mathbf{D})\rangle_{\boldsymbol{\theta}\setminus\boldsymbol{\theta}_i})}{\int \exp(\langle \ln p(\boldsymbol{\theta}, \mathbf{D})\rangle_{\boldsymbol{\theta}\setminus\boldsymbol{\theta}_i}) \mathrm{d}\boldsymbol{\theta}_i}, \tag{14}$$

where $\langle\cdot\rangle_{\boldsymbol{\theta}\setminus\boldsymbol{\theta}_i}$ denotes the expectation with respect to q over all variables except $\boldsymbol{\theta}_i$.

Factorized form of the approximate posterior q. We assume that the approximation of the posterior (10) has the following factorization (the subscripts of q are omitted without confusion):

$$\begin{aligned} & q(\boldsymbol{\mathcal{X}}, \{\mathbf{U}_d, \mathbf{V}_d, \boldsymbol{\gamma}_d, \boldsymbol{\tau}_d, \boldsymbol{\mathcal{Z}}_d, \boldsymbol{\Pi}_d\}_{d=1}^3) \\ = & \prod_{i,j,k=1}^{I,J,K} q(\boldsymbol{\mathcal{X}}(i,j,k)) \prod_{d=1}^3 \Big\{ \prod_{i=1}^{I_d} q(\mathbf{U}_d(i,:)) \prod_{j=1}^{J_d} q(\mathbf{V}_d(j,:)) \prod_{r=1}^{R_d} q(\boldsymbol{\gamma}_d(r)) \\ & \prod_{l=1}^{L_d} q(\boldsymbol{\tau}_d(l)) \prod_{i,j,k,l=1}^{I,J,K,L_d} q(\boldsymbol{\mathcal{Z}}_d(i,j,k,l)) \prod_{k=1}^{K} q(\boldsymbol{\Pi}_d(k,:)) \Big\}. \end{aligned} \tag{15}$$

According to (14), we give the analytical inference of each component in (15) as below.

Estimation of the low-rank component. Variables involved in the low-rank component are the factor matrices $\{\mathbf{U}_d \in \mathbb{R}^{I_d \times R_d}\}_{d=1}^3$ and $\{\mathbf{V}_d \in \mathbb{R}^{J_d \times R_d}\}_{d=1}^3$ with column-wise precisions $\{\boldsymbol{\gamma}_d \in \mathbb{R}^{R_d}\}_{d=1}^3$, and the solution $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$. For each row of $\mathbf{U}_d$, we have that

$$q(\mathbf{U}_d(i,:)) = \mathcal{N}(\mathbf{U}_d(i,:) | \boldsymbol{\mu}_{\mathbf{U}_d(i,:)}, \boldsymbol{\Sigma}_{\mathbf{U}_d(i,:)}), \tag{16}$$

with covariance $\boldsymbol{\Sigma}_{\mathbf{U}_d(i,:)} \in \mathbb{R}^{R_d \times R_d}$ and mean $\boldsymbol{\mu}_{\mathbf{U}_d(i,:)} \in \mathbb{R}^{R_d}$ given by

$$\begin{aligned} \boldsymbol{\Sigma}_{\mathbf{U}_d(i,:)} &= \Big(\sum_{j=1}^{J_d} \Big(\mathbf{w}(d)\xi + \sum_{l=1}^{L_d} \langle \boldsymbol{\mathcal{Z}}_d(i',j',k',l)\rangle \langle \boldsymbol{\tau}(l)\rangle \Big) \langle \mathbf{V}_d(j,:)^T \mathbf{V}_d(j,:)\rangle + \mathrm{diag}(\langle \boldsymbol{\gamma}_d \rangle) \Big)^{-1}, \\ \boldsymbol{\mu}_{\mathbf{U}_d(i,:)} &= \sum_{j=1}^{J_d} \Big(\mathbf{w}(d)\xi \langle \boldsymbol{\mathcal{X}}(i',j',k')\rangle + \sum_{l=1}^{L_d} \langle \boldsymbol{\mathcal{Z}}_d(i',j',k',l)\rangle \langle \boldsymbol{\tau}(l)\rangle \boldsymbol{\mathcal{Y}}(i',j',k') \Big) \langle \mathbf{V}_d(j,:)\rangle \boldsymbol{\Sigma}_{\mathbf{U}_d(i,:)}, \end{aligned}$$

where (i',j',k') maps to (i,j) such that $\boldsymbol{\mathcal{A}}^{I \times J \times K}(i',j',k') = \mathbf{A}_{(d)}^{I_d \times J_d}(i,j)$. Similarly, for each row of $\mathbf{V}_d$, we have that

$$q(\mathbf{V}_d(j,:)) = \mathcal{N}(\mathbf{V}_d(j,:) | \boldsymbol{\mu}_{\mathbf{V}_d(j,:)}, \boldsymbol{\Sigma}_{\mathbf{V}_d(j,:)}), \tag{17}$$

where

$$\begin{aligned} \boldsymbol{\Sigma}_{\mathbf{V}_d(j,:)} &= \Big(\sum_{i=1}^{I_d} \Big(\mathbf{w}(d)\xi + \sum_{l=1}^{L_d} \langle \boldsymbol{\mathcal{Z}}_d(i',j',k',l)\rangle \langle \boldsymbol{\tau}(l)\rangle \Big) \langle \mathbf{U}_d(i,:)^T \mathbf{U}_d(i,:)\rangle + \mathrm{diag}(\langle \boldsymbol{\gamma}_d \rangle) \Big)^{-1}, \\ \boldsymbol{\mu}_{\mathbf{V}_d(j,:)} &= \sum_{i=1}^{I_d} \Big(\mathbf{w}(d)\xi \langle \boldsymbol{\mathcal{X}}(i',j',k')\rangle + \sum_{l=1}^{L_d} \langle \boldsymbol{\mathcal{Z}}_d(i',j',k',l)\rangle \langle \boldsymbol{\tau}(l)\rangle \boldsymbol{\mathcal{Y}}(i',j',k') \Big) \langle \mathbf{U}_d(i,:)\rangle \boldsymbol{\Sigma}_{\mathbf{V}_d(j,:)}. \end{aligned}$$

For each element in $\boldsymbol{\gamma}_d$, we have that

$$q(\boldsymbol{\gamma}_d(r)) = \mathrm{Gamma}(\boldsymbol{\gamma}_d(r) | c_{\boldsymbol{\gamma}_d}, d_{\boldsymbol{\gamma}_d(r)}), \tag{18}$$

with parameters $c_{\boldsymbol{\gamma}_d} \in \mathbb{R}$ and $d_{\boldsymbol{\gamma}_d(r)} \in \mathbb{R}$ given by

$$\begin{aligned} c_{\boldsymbol{\gamma}_d} &= c_0 + \frac{1}{2}(I_d + J_d), \\ d_{\boldsymbol{\gamma}_d(r)} &= d_0 + \frac{1}{2}(\langle \mathbf{U}_d(:,r)^T \mathbf{U}_d(:,r)\rangle + \langle \mathbf{V}_d(:,r)^T \mathbf{V}_d(:,r)\rangle). \end{aligned}$$

For each element in $\boldsymbol{\mathcal{X}}$, we have that

$$q(\boldsymbol{\mathcal{X}}(i,j,k)) = \mathcal{N}(\boldsymbol{\mathcal{X}}(i,j,k) | \mu_{\boldsymbol{\mathcal{X}}(i,j,k)}, \xi^{-1}), \tag{19}$$

with mean $\mu_{\boldsymbol{\mathcal{X}}(i,j,k)} \in \mathbb{R}$ given by

$$\mu_{\boldsymbol{\mathcal{X}}(i,j,k)} = \sum_{d=1}^{3} \mathbf{w}(d) \langle \mathbf{U}_d(i_d,:) \rangle \langle \mathbf{V}_d(j_d,:)^T \rangle,$$

where (i_d, j_d) maps to (i,j,k) such that $\mathbf{A}_{(d)}^{I_d \times J_d}(i_d, j_d) = \boldsymbol{\mathcal{A}}^{I \times J \times K}(i,j,k)$.

Estimation of the noise component. Variables involved in the noise component are the precisions $\{\boldsymbol{\tau}_d \in \mathbb{R}^{L_d}\}_{d=1}^3$, the mixing proportions $\{\boldsymbol{\Pi}_d \in \mathbb{R}^{K \times L_d}\}_{d=1}^3$, and the indicator variables $\{\boldsymbol{\mathcal{Z}}_d \in \mathbb{R}^{I \times J \times K \times L_d}\}_{d=1}^3$. For each element in $\boldsymbol{\tau}_d$, we have that

$$q(\boldsymbol{\tau}_d(l)) = \text{Gamma}(\boldsymbol{\tau}_d(l) | a_{\boldsymbol{\tau}_d(l)}, b_{\boldsymbol{\tau}_d(l)}), \tag{20}$$

with parameters $a_{\boldsymbol{\tau}_d(l)} \in \mathbb{R}$ and $b_{\boldsymbol{\tau}_d(l)} \in \mathbb{R}$ given by

$$\begin{aligned} a_{\boldsymbol{\tau}_d(l)} &= a_0 + \frac{1}{2} \sum_{i,j,k=1}^{I,J,K} \langle \boldsymbol{\mathcal{Z}}_d(i,j,k,l) \rangle, \\ b_{\boldsymbol{\tau}_d(l)} &= b_0 + \frac{1}{2} \sum_{i,j,k=1}^{I,J,K} \langle \boldsymbol{\mathcal{Z}}_d(i,j,k,l) \rangle \langle (\boldsymbol{\mathcal{Y}}(i,j,k) - \mathbf{U}_d(i_d,:) \mathbf{V}_d(j_d,:)^T)^2 \rangle, \end{aligned}$$

where (i_d, j_d) maps to (i,j,k) such that $\mathbf{A}_{(d)}^{I_d \times J_d}(i_d, j_d) = \boldsymbol{\mathcal{A}}^{I \times J \times K}(i,j,k)$. For each row of $\boldsymbol{\Pi}_d$, we have that

$$q(\boldsymbol{\Pi}_d(k,:)) = \text{Dirichlet}(\boldsymbol{\Pi}_d(k,:) | \boldsymbol{\alpha}_{\boldsymbol{\Pi}_d(k,:)}), \tag{21}$$

with a parameter $\boldsymbol{\alpha}_{\boldsymbol{\Pi}_d(k,:)} \in \mathbb{R}^{L_d}$ given by

$$\boldsymbol{\alpha}_{\boldsymbol{\Pi}_d(k,:)}(l) = \alpha_0 + \sum_{i,j=1}^{I,J} \langle \boldsymbol{\mathcal{Z}}_d(i,j,k,l) \rangle.$$

For each mode-4 fiber of $\boldsymbol{\mathcal{Z}}_d$, we have that

$$q(\boldsymbol{\mathcal{Z}}_d(i,j,k,:)) = \text{Multinomial}(\boldsymbol{\mathcal{Z}}_d(i,j,k,:) | \boldsymbol{\rho}_{\boldsymbol{\mathcal{Z}}_d(i,j,k,:)}), \tag{22}$$

with a parameter $\boldsymbol{\rho}_{\boldsymbol{\mathcal{Z}}_d(i,j,k,:)} \in \mathbb{R}^{L_d}$ given by

$$\begin{aligned} \boldsymbol{\rho}_{\boldsymbol{\mathcal{Z}}_d(i,j,k,:)}(l) = c \exp \Bigg(&-\frac{1}{2} \ln(2\pi) + \frac{1}{2} \langle \ln \boldsymbol{\tau}_d(l) \rangle - \frac{1}{2} \langle \ln \boldsymbol{\tau}_d(l) \rangle \langle (\boldsymbol{\mathcal{Y}}(i,j,k) - \mathbf{U}_d(i_d,:) \mathbf{V}_d(j_d,:)^T)^2 \rangle \\ &+ \langle \ln \boldsymbol{\Pi}_d(k,l) \rangle \Bigg), \end{aligned}$$

where (i_d, j_d) maps to (i,j,k) such that $\mathbf{A}_{(d)}^{I_d \times J_d}(i_d, j_d) = \boldsymbol{\mathcal{A}}^{I \times J \times K}(i,j,k)$ and c is a normalization constant to ensure that $\sum_{l=1}^{L_d} \boldsymbol{\rho}_{\boldsymbol{\mathcal{Z}}_d(i,j,k,:)}(l) = 1$.

Pseudo-code and complexity analysis. The pseudo-code of the overall algorithm is summarized in Algorithm 1. The total complexity per iteration is approximately

$$\mathcal{O}\left(\sum_{d=1}^{3} (I_d + J_d) R_d^3 + I_d J_d R_d^2 L_d \right), \tag{23}$$

where the first term is due to calculating the covariance matrices of $\{\mathbf{U}_d, \mathbf{V}_d\}_{d=1}^3$ (16,17), and the second term is due to calculating the parameters of $\{\mathbf{Z}_d\}_{d=1}^3$ (22). Since, in general, it holds $R_d \ll \min(I_d, J_d)$, the complexity of our algorithm depends linearly on the size of the input data.

Algorithm 1. Variational Bayesian algorithm for NMoG-Tucker.

Input: Observed image $\mathcal{Y}$.
Initialization:
1. Set the iteration index $t := 0$.
2. Initialize the low-rank component $\{\mathbf{U}_d^{(t)}, \mathbf{V}_d^{(t)}, \gamma_d^{(t)}\}_{d=1}^3$.
3. Initialize the noise component $\{\boldsymbol{\tau}_d^{(t)}, \mathcal{Z}_d^{(t)}\}_{d=1}^3$.

Iteration: while not converged **do**
4. Given $\{\boldsymbol{\tau}_d^{(t)}, \mathcal{Z}_d^{(t)}\}_{d=1}^3$, update the low-rank component $\{\mathbf{U}_d^{(t+1)}, \mathbf{V}_d^{(t+1)}, \gamma_d^{(t+1)}\}_{d=1}^3$ and $\mathcal{X}^{(t+1)}$ by (16)–(19).
5. Given $\{\mathbf{U}_d^{(t+1)}, \mathbf{V}_d^{(t+1)}, \gamma_d^{(t+1)}\}_{d=1}^3$, update the noise component $\{\boldsymbol{\tau}_d^{(t+1)}, \boldsymbol{\Pi}_d^{(t+1)}, \mathcal{Z}_d^{(t+1)}\}_{d=1}^3$ by (20)–(22).
6. Set $t := t + 1$.

End while and **output** $\mathcal{X} = \mathcal{X}^{(t)}$.

3.3. Selection of Hyperparameters

This section is devoted to the selection of hyperparameters involved in our model. We develop adaptive strategies to learn their values using the results of the current iteration, which makes our algorithm free from burdensome hyperparameter-tuning.

Selection of $\{R_d\}_{d=1}^3$. The hyperparameter $\{R_d\}_{d=1}^3$ controls the column numbers of the factor matrices $\{\mathbf{U}_d, \mathbf{V}_d\}_{d=1}^3$, which is an estimation of the Tucker rank of the solution. Since the true rank is often unknown in practice, we design an adaptive rank estimation strategy to improve the applicability of our method. The main idea consists of initializing R_d with a large value and then decreasing it gradually by dropping singular values smaller than an adaptive threshold. More precisely, denoting by t the iteration index, we choose $R_d^{(t)}$ as

$$R_d^{(t)} := \max\left\{r \mid \boldsymbol{\sigma}_d^{(t)}(r) \geq s_d^{(t)}\right\}, \tag{24}$$

where $\boldsymbol{\sigma}_d^{(t)} \subset [0,1]$ is a vector composed of the singular values of $\mathbf{U}_d^{(t)}(\mathbf{V}_d^{(t)})^T / \|\mathbf{U}_d^{(t)}(\mathbf{V}_d^{(t)})^T\|_2$ in a decreasing order ($\|\cdot\|_2$ denotes the spectral norm, i.e., the largest singular value), $s_d^{(t)} \in [0,1]$ is a threshold given by

$$s_d^{(t)} := \max\left(\min\left(\boldsymbol{\sigma}_d^{(t)}\left(\min\left\{i \mid \textstyle\sum_{r>i}\boldsymbol{\sigma}_d^{(t)}(r) < e^{\text{upper}}\|\boldsymbol{\sigma}_d^{(t)}\|_1\right\}\right), s_d^{(t-1)}\right), e^{\text{lower}}\boldsymbol{\sigma}_d^{(t)}(\text{end})\right), \tag{25}$$

where $e^{\text{upper}} \in (0,1)$ imposes an upper bound of the sum of the dropping singular values $\{\boldsymbol{\sigma}_d^{(t)}(r)\}_{r>i}$, $e^{\text{lower}} \in (0,1)$ imposes a lower bound of the threshold $s_d^{(t)}$, and $\boldsymbol{\sigma}_d^{(t)}(\text{end})$ denotes the last element of $\boldsymbol{\sigma}_d^{(t)}$. Our experiments use the default settings $e^{\text{upper}} = 10^{-2}$ and $e^{\text{lower}} = 2/3$; the effects of these two hyperparameters on the denoising performance will be discussed in Section 4.4.

We make some comments on the proposed rank estimation strategy. First, the dropping singular values in each iteration carry at most 1% energy of $\mathbf{U}_d^{(t)}(\mathbf{V}_d^{(t)})^T$, leading to a robust rank decreasing process. Second, the threshold tends to decrease if a rank reduction occurs, i.e., $s_d^{(t)} > \boldsymbol{\sigma}_d^{(t)}(\text{end})$, which avoids underestimating the true rank. Third, the threshold tends to increase if it is too small to trigger a rank reduction, i.e., $s_d^{(t)} < \frac{2}{3}\boldsymbol{\sigma}_d^{(t)}(\text{end})$, which avoids overestimating the true rank.

Selection of w. The hyperparameter $\mathbf{w}$ assigns relative weights of the three modes in the prior (9) and the posterior (19) of $\mathcal{X}$. We assume a positive correlation between $\mathbf{w}(d)$ and the low-rankness degree of $\mathbf{X}_{(d)}$, i.e., the more sparse the singular values of $\mathbf{X}_{(d)}$ are, the larger $\mathbf{w}(d)$ is. To measure the sparsity of singular values, we use the Gini index [46] (Here we take $G := 1 - G'$, where G' is the definition of the Gini index in [46]) defined by

$$G(\mathbf{a}) := \sum_{i=1}^{I} \left(\frac{2i-1}{I} \right) \frac{\mathbf{a}(i)}{\|\mathbf{a}\|_1},$$

where $\mathbf{a} \in \mathbb{R}^I$ is a non-zero vector composed of nonnegative elements in a decreasing order. The Gini index takes positive values, and smaller values indicate better sparsity. Then, at the t-th iteration, we choose $\mathbf{w}^{(t)}(d)$ as

$$\mathbf{w}^{(t)}(d) := c \exp\left(- \frac{G(\boldsymbol{\sigma}^{(t)}_{\mathbf{X}_{(d)}})}{\min_d G(\boldsymbol{\sigma}^{(t)}_{\mathbf{X}_{(d)}})} \right), \tag{26}$$

where $\boldsymbol{\sigma}^{(t)}_{\mathbf{X}_{(d)}}$ contains the singular values of $\mathbf{X}_{(d)}$ in a decreasing order and c is a normalization constant to ensure that $\sum_{d=1}^{3} \mathbf{w}^{(t)}(d) = 1$. Here we divide by $\min_d G(\boldsymbol{\sigma}^{(t)}_{\mathbf{X}_{(d)}})$ to measure the relative, rather than absolute, low-rankness degree.

Selection of ξ. The hyperparameter ξ is the precision of the Gaussian distributions in the prior (9) and the posterior (19) of $\boldsymbol{\mathcal{X}}$, which controls the contribution of $\boldsymbol{\mathcal{X}}$ to the inference results of $\{\mathbf{U}_d\}_{d=1}^3$ (16) and $\{\mathbf{V}_d\}_{d=1}^3$ (17), or, equivalently, penalizes the distances between $\boldsymbol{\mathcal{X}}$ and the low-rank components $\{\text{fold}_d(\mathbf{U}_d\mathbf{V}_d^T)\}_{d=1}^3$. For stability purpose, we initialize ξ with a small value and increase it gradually until the convergence of $\boldsymbol{\mathcal{X}}$. More precisely, at the t-th iteration, we set $\xi^{(t)}$ as

$$\xi^{(t)} := \xi_0^{(t)} \frac{IJK}{\|\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}^{(t)}\|_F^2}, \tag{27}$$

where ξ_0 is an auxiliary hyperparameter updated as

$$\xi_0^{(t)} := \begin{cases} 1.5\xi_0^{(t-1)}, & \text{if } \frac{\|\boldsymbol{\mathcal{X}}^{(t)} - \boldsymbol{\mathcal{X}}^{(t-1)}\|_F^2}{\|\boldsymbol{\mathcal{X}}^{(t-1)}\|_F^2} < \frac{\|\boldsymbol{\mathcal{X}}^{(t-1)} - \boldsymbol{\mathcal{X}}^{(t-2)}\|_F^2}{\|\boldsymbol{\mathcal{X}}^{(t-2)}\|_F^2}, \\ \xi_0^{(t-1)}, & \text{otherwise.} \end{cases}$$

Selection of $\{L_d\}_{d=1}^3$. The hyperparameter L_d is the number of Gaussian components in the MoG prior of the noise $\mathbf{N}_d$ (2). To adaptively fit the noise distribution, we initialize L_d with a relatively large value and iteratively decrease L_d to $L_d - 1$ if there exist two analogous Gaussian components satisfying

$$\frac{|\boldsymbol{\tau}_d(l_1) - \boldsymbol{\tau}_d(l_2)|}{|\boldsymbol{\tau}_d(l_1) + \boldsymbol{\tau}_d(l_2)|} \leq 0.05.$$

For an initialization of L_d, our experiments use the default setting $(L_1^{(0)}, L_2^{(0)}, L_3^{(0)}) = (8, 8, 8)$; its effects on the denoising performance will be discussed in Section 4.4.

Selection of other hyperparameters. The rest hyperparameters are a_0 and b_0 in the Gamma prior of $\{\boldsymbol{\tau}_d\}_{d=1}^3$, c_0 and d_0 in the Gamma prior of $\{\gamma_d\}_{d=1}^3$, and α_0 in the Dirichlet prior of $\{\boldsymbol{\Pi}_d\}_{d=1}^3$. We simply fix them to 10^{-6} in a non-informative manner, to minimize their impacts on the inference process [44]. Our method performs stably well in all experiments under these simple settings.

4. Numerical Experiments

We evaluate the denoising performance of the proposed NMoG-Tucker method on synthetic data, MSIs, and HSIs. Table 1 lists six state-of-the-art competing methods on low-rank matrix/tensor approximation: matrix-based methods LRMR [38], MoG-RPCA [4], and NMoG-LRMF [7]; tensor-based methods LRTA [24], PARAFAC [21], and KBR-RPCA [35]. Parameters involved in all competing methods are set to default values or manually tuned for the best possible denoising performance. All experiments are conducted under Windows 10 and Matlab R2016a (Version 9.0.0.341360) running on a desktop with an Intel(R) Core(TM) i7-8700K CPU at 3.70 GHz and 32 GB memory.

Table 1. Summary of competing methods. Here $\times_d$ denotes the d-mode product [19], and $\circ$ denotes the vector outer product.

Competing Method	**Data Prior**	**Noise Prior**
LRMR [38]	Matrix rank constraint $\text{rank}(\mathbf{X}_{(3)}) \leq r$	Gaussian + sparse
MoG-RPCA [4]	Low-rank matrix factorization $\mathbf{X}_{(3)} = \mathbf{U}\mathbf{V}^T$	MoG
NMoG-LRMF [7]	Low-rank matrix factorization $\mathbf{X}_{(3)} = \mathbf{U}\mathbf{V}^T$	Non-i.i.d. MoG
LRTA [24]	Tucker decomposition $\mathcal{X} = \mathcal{C} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3$	Gaussian
PARAFAC [21]	CP decomposition $\mathcal{X} = \sum_r \lambda_r \mathbf{U}_1(:,r) \circ \mathbf{U}_2(:,r) \circ \mathbf{U}_3(:,r)$	Gaussian
KBR-RPCA [35]	Kronecker-basis-representation $S(\mathcal{X}) = t\|\mathcal{C}\|_0 + (1-t)\prod_{d=1}^3 \text{rank}(\mathbf{X}_{(d)})$, where $\mathcal{X} = \mathcal{C} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3$	Gaussian + sparse

We conduct both simulated and real denoising experiments. In simulated experiments, the noisy data are generated by adding synthetic noises to the original ones, and the denoising performance is evaluated by both quantitative measures and visual quality. In real experiments, the goal is to recover real-world data without knowing the ground-truths, and the denoising results are mainly judged by visual quality.

In simulated experiments, we use the following four quantitative measures: relative error (ReErr), erreur relative globale adimensionnelle de synthèse (ERGAS) [47], mean of peak signal-to-noise ratio (MPSNR), and mean of structural similarity (MSSIM) [48]. Denoting by $\mathcal{X}_{\text{res}} \in \mathbb{R}^{I\times J\times K}$ an estimation to the original data $\mathcal{X}_{\text{ori}} \in \mathbb{R}^{I\times J\times K}$, the four measures of $\mathcal{X}_{\text{res}}$ with respect to $\mathcal{X}_{\text{ori}}$ are defined as follows:

$$\text{ReErr}(\mathcal{X}_{\text{res}}, \mathcal{X}_{\text{ori}}) := \frac{\|\mathcal{X}_{\text{res}} - \mathcal{X}_{\text{ori}}\|_F}{\|\mathcal{X}_{\text{ori}}\|_F},$$

$$\text{ERGAS}(\mathcal{X}_{\text{res}}, \mathcal{X}_{\text{ori}}) := 100\sqrt{\frac{1}{K}\sum_{k=1}^K \frac{\|\mathcal{X}_{\text{res}}(:,:,k) - \mathcal{X}_{\text{ori}}(:,:,k)\|_F^2}{\sum_{i,j=1}^{I,J} \mathcal{X}_{\text{ori}}(i,j,k)}},$$

$$\text{MPSNR}(\mathcal{X}_{\text{res}}, \mathcal{X}_{\text{ori}}) := \frac{1}{K}\sum_{k=1}^K 10\log_{10}\left(\frac{255^2 IJ}{\|\mathcal{X}_{\text{res}}(:,:,k) - \mathcal{X}_{\text{ori}}(:,:,k)\|_F^2}\right),$$

$$\text{MSSIM}(\mathcal{X}_{\text{res}}, \mathcal{X}_{\text{ori}}) := \frac{1}{K}\sum_{k=1}^K \text{SSIM}(\mathcal{X}_{\text{res}}(:,:,k), \mathcal{X}_{\text{ori}}(:,:,k)),$$

where the details of SSIM can be found in [48]. In general, better denoising results have smaller ReErr and ERGAS values and larger MPSNR and MSSIM values.

4.1. Synthetic Data Denoising

This section presents simulated experiments on synthetic data denoising. The original data are random low-rank tensors generated by the Tucker model with size $50 \times 50 \times 50$ and rank (R_1, R_2, R_3), i.e., $\mathcal{X}_{\text{ori}} := \mathcal{C} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3$, where the core tensor $\mathcal{C} \in \mathbb{R}^{R_1\times R_2\times R_3}$ and each factor matrix $\mathbf{U}_d \in \mathbb{R}^{50\times R_d}$ $(d = 1, 2, 3)$ are drawn from standard Gaussian distribution. We consider two rank settings $(10, 10, 10)$ and $(20, 15, 10)$. The original data are normalized to have unit mean absolute value, i.e., $\|\mathcal{X}_{\text{ori}}\|_1/50^3 = 1$. We test the following three kinds of synthetic noises.

- Gaussian noise: all entries mixed with Gaussian noise $\mathcal{N}(0, 0.1^2)$.

- Gaussian + sparse noise: 80% entries mixed with Gaussian noise $\mathcal{N}(0, 0.1^2)$ and 20% with additive uniform noise between $[-5, 5]$.
- Mixture noise: 40% entries mixed with Gaussian noise $\mathcal{N}(0, 0.01^2)$, 20% with Gaussian noise $\mathcal{N}(0, 0.2^2)$, 20% with additive uniform noise between $[-5, 5]$, and 20% missing (the locations of missing entries are not given as prior knowledge).

Table 2 reports the ReErr values and execution time of different methods on synthetic data denoising, where every result is an average over ten trials with different realizations of both data and noise. Regarding the denoising accuracy, our method consistently attains comparable or lower ReErr values than the competing methods, and its superiority becomes more significant as the noise complexity increases. Regarding the computational speed, LRTA is generally the fastest method, while our method is the slowest in all cases. The relatively high cost of our algorithm is mainly due to two facts: computing variables corresponding to all three modes requires three times more calculations than those in matrix-based methods; updating the factor matrices row by row is much slower than updating them as a whole in other tensor-based methods. An acceleration of our implementation will be left to future research.

Table 2. Quantitative performance and execution time (in seconds) of different methods on synthetic data denoising. Every result is an average over ten trials with different realizations of both data and noise. The best results are highlighted in bold.

	Gaussian noise		**Gaussian + sparse noise**		**Mixture noise**	
Rank (10,10,10)	**ReErr**	**Time**	**ReErr**	**Time**	**ReErr**	**Time**
Noisy data	7.41e-02	-	6.92e-01	-	8.08e-01	-
LRMR	4.09e-02	**0.11**	1.22e-01	2.27	4.40e-01	2.30
MoG-RPCA	3.35e-02	1.22	4.54e-02	6.22	3.21e-01	19.16
NMoG-LRMF	3.35e-02	4.78	4.15e-02	4.54	3.30e-01	15.29
LRTA	1.06e-02	0.12	1.43e-01	**0.18**	3.43e-01	**0.42**
PARAFAC	1.97e-02	5.19	2.67e-01	4.26	4.53e-01	4.04
KBR-RPCA	**9.91e-03**	2.93	1.44e-02	2.86	5.00e-02	2.91
NMoG-Tucker	1.00e-02	14.84	**1.17e-02**	31.14	**3.25e-03**	45.84
	Gaussian noise		**Gaussian + sparse noise**		**Mixture noise**	
Rank (20,15,10)	**ReErr**	**Time**	**ReErr**	**Time**	**ReErr**	**Time**
Noisy data	7.56e-02	-	7.00e-01	-	8.12e-01	-
LRMR	4.27e-02	0.18	1.27e-01	2.09	4.58e-01	2.09
MoG-RPCA	3.42e-02	1.34	4.58e-02	5.01	2.84e-01	17.58
NMoG-LRMF	3.42e-02	3.72	4.22e-02	4.30	3.12e-01	15.17
LRTA	1.55e-02	**0.14**	2.04e-01	**0.18**	3.85e-01	**0.36**
PARAFAC	1.87e-01	5.10	3.17e-01	4.61	4.98e-01	4.18
KBR-RPCA	**1.45e-02**	2.45	2.16e-02	2.23	9.06e-02	3.12
NMoG-Tucker	1.47e-02	16.63	**1.72e-02**	34.62	**1.97e-02**	62.04

4.2. MSI Denoising

This section presents simulated experiments on MSI denoising. The original data are six MSIs (*Beads*, *Cloth*, *Hairs*, *Jelly Beans*, *Oil Painting*, *Watercolors*) from the Columbia MSI Database (http://www1.cs.columbia.edu/CAVE/databases/multispectral) [49] containing scenes of a variety of real-world objects. Each MSI is of size $512 \times 512 \times 31$ with intensity range scaled to $[0, 1]$. We test the following two kinds of synthetic noises.

- Gaussian noise: all entries mixed with Gaussian noise $\mathcal{N}(0, 0.05^2)$. The signal-to-noise-ratio (SNR) value averaged over all 31 bands and all six MSIs is 13.88 dB.
- Mixture noise: 60% entries mixed with Gaussian noise $\mathcal{N}(0, 0.01^2)$, 20% with Gaussian noise $\mathcal{N}(0, 0.2^2)$, and 20% with additive uniform noise between $[-5, 5]$. The SNR value averaged over all 31 bands and all six MSIs is -14.38 dB.

Table 3 reports the quantitative performance of different methods on MSI denoising, where every result is an average over six testing MSIs. For Gaussian noise, our method achieves comparable denoising performance to LRMR, LRTA, and KBR-RPCA. For mixture noise, our method performs better than the competing methods in terms of all three quantitative measures, and KBR-RPCA is the second best.

Table 3. Quantitative performance of different methods on MSI denoising. Every result is an average over six testing MSIs. The best results are highlighted in bold.

	Gaussian Noise			Mixture Noise		
	MPSNR	**MSSIM**	**ERGAS**	**MPSNR**	**MSSIM**	**ERGAS**
Noisy image	26.02	0.8088	204.24	-2.24	0.0233	5287.19
LRMR	35.30	0.9631	72.70	20.38	0.6893	418.92
MoG-RPCA	31.31	0.8131	123.98	31.34	0.9475	125.40
NMoG-LRMF	32.88	0.9453	106.49	32.39	0.9531	146.44
LRTA	**35.40**	0.9575	71.53	20.61	0.4120	386.96
PARAFAC	26.82	0.7349	211.89	17.10	0.2496	569.36
KBR-RPCA	35.19	0.9637	73.95	33.43	0.9548	96.54
NMoG-Tucker	35.37	**0.9703**	**71.49**	**36.02**	**0.9787**	**85.33**

Figure 2 shows the average PSNR and SSIM values across all bands of the denoising results by different methods. For easy observation of the details, we plot the differences between our results and the competing ones at larger scales. It can be observed that our method achieves comparable or better performance for most bands, while KBR-RPCA exhibits the best robustness over all bands.

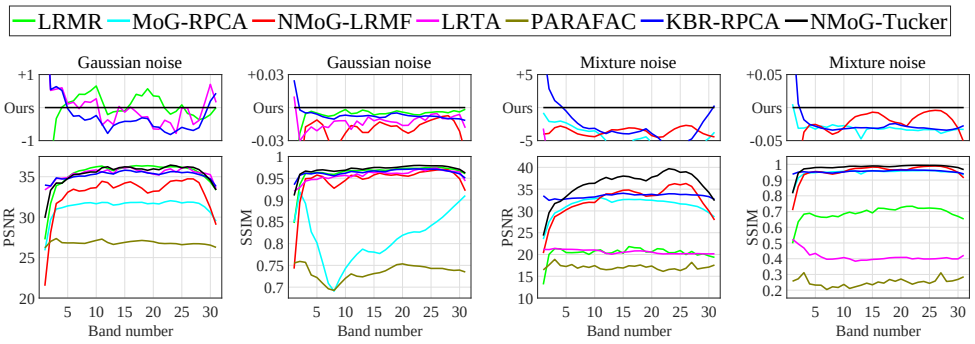

Figure 2. PSNR and SSIM values of each band in MSI denoising, averaged over six testing MSIs. Differences between our results and the competing ones are plotted at larger scales.

Figure 3 shows two examples on MSI denoising under Gaussian noise and mixture noise. These figures suggest that the results by the competing methods generally maintain some noise or alter image details, whereas our results exhibit higher visual quality in both noise removal and detail preservation. For better visualization, we enlarge a certain patch and show the corresponding error map, which highlights the difference between the denoised patch and the original one. A close inspection reveals that our error maps contain less color information than the competing ones, indicating that our method better recovers the spatial-spectral structures of the original MSIs.

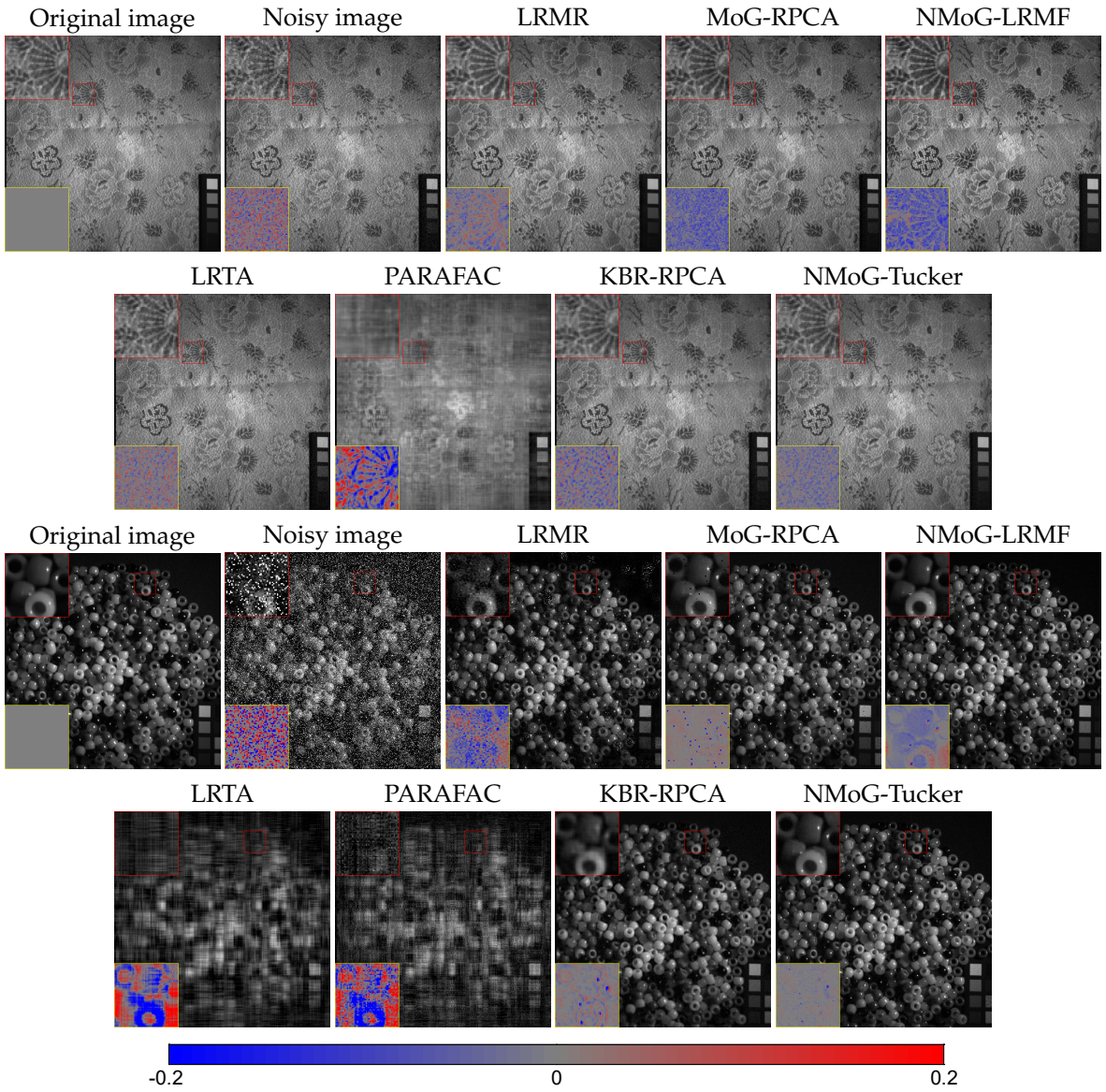

Figure 3. MSI denoising examples. Top two rows: band 31 in *Cloth* under Gaussian noise. Bottom two rows: band 31 in *Beads* under mixture noise. For better visualization, we show enlargements of two demarcated patches and the corresponding error maps (difference between the currently displayed patch and the original one). Error maps with less color information indicate better denoising performance.

4.3. HSI Denoising

We conduct both simulated and real experiments on HSI denoising.

Simulated HSI denoising. We adopt two original HSIs considered in NMoG-LRMF [7], i.e., a $200 \times 200 \times 160$ sub-image of *Washington DC Mall* (http://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html) (*DCmall* for short) and a $200 \times 200 \times 89$ sub-image of *Cuprite* (http://peterwonka.net/Publications/code/LRTC_Package_Ji.zip) [25]. The intensity range is scaled to $[0, 1]$. To simulate the degradation scenarios in real-world HSIs, we test the following three kinds of synthetic noises.

- Gaussian noise: all entries mixed with Gaussian noise $\mathcal{N}(0, 0.05^2)$. For *DCmall*, the SNR value of each band varies from 6 to 20 dB, and the mean SNR value of all 160 bands is 13.79 dB. For *Cuprite*, the SNR value of each band varies from 16 to 20 dB, and the mean SNR value of all 89 bands is 18.69 dB.

- Speckle noise: all bands are corrupted by non-i.i.d. speckle noise with signal-dependent intensity, which is simulated by multiplicative uniform noise with mean 1 and variance randomly sampled from $[0.001, 0.5]$ for each band. For both *DCmall* and *Cuprite*, the SNR value of each band varies from 3 to 30 dB. The mean SNR value of all 160 bands in *DCmall* is 19.52 dB, and that of all 89 bands in *Cuprite* is 20.03 dB.
- Mixture noise: all bands are corrupted by non-i.i.d. Gaussian noise with zero-mean and band-dependent variances, and the SNR value of each band is uniformly sampled from 10 to 20 dB. Then, we randomly choose 90/50 bands in *DCmall*/*Cuprite* to add complex noises: the first 40/20 bands are corrupted by stripe noise with stripe number between $[20, 40]$ and stripe intensity between $[-0.25, 0.25]$; the middle 40/20 bands are corrupted by deadline with line number between $[5, 15]$; 50% to 70% entries in the last 40/20 bands are corrupted by speckle noise with mean 1 and variance 0.3. Thus, each band is randomly corrupted by one to three types of noises. For both *DCmall* and *Cuprite*, the SNR value of each band varies from 4 to 20 dB. The mean SNR value of all 160 bands in *DCmall* is 11.62 dB, and that of all 89 bands in *Cuprite* is 12.04 dB.

Table 4 presents the quantitative performance of different methods on simulated HSI denoising, where every result is an average over five trials with different noise realizations. Compared with the competing methods, our method consistently yields better performance in terms of MPSNR, MSSIM, and ERGAS in all cases.

Table 4. Quantitative performance of different methods on simulated HSI denoising. Every result is an average over five trials with different noise realizations. The best results are highlighted in bold.

	Gaussian noise			Speckle noise			Mixture noise		
DCmall	**MPSNR**	**MSSIM**	**ERGAS**	**MPSNR**	**MSSIM**	**ERGAS**	**MPSNR**	**MSSIM**	**ERGAS**
Noisy data	26.02	0.7627	187.93	31.65	0.8697	226.53	23.94	0.6988	316.59
LRMR	38.54	0.9848	43.35	38.44	0.9789	58.46	37.10	0.9785	58.66
MoG-RPCA	38.97	0.9865	41.59	33.85	0.9520	144.52	34.81	0.9597	110.30
NMoG-LRMF	39.47	0.9876	38.91	39.66	0.9847	59.58	38.68	0.9838	56.39
LRTA	36.86	0.9731	52.07	31.66	0.8698	226.17	24.05	0.7021	314.09
PARAFAC	32.02	0.9360	90.77	32.75	0.9410	89.04	28.81	0.8722	164.61
KBR-RPCA	37.31	0.9819	49.94	38.20	0.9844	48.77	36.82	0.9797	54.89
NMoG-Tucker	**39.52**	**0.9877**	**38.68**	**40.23**	**0.9876**	**42.66**	**39.23**	**0.9865**	**44.51**
	Gaussian noise			**Speckle noise**			**Mixture noise**		
Cuprite	**MPSNR**	**MSSIM**	**ERGAS**	**MPSNR**	**MSSIM**	**ERGAS**	**MPSNR**	**MSSIM**	**ERGAS**
Noisy data	26.02	0.6953	124.07	27.23	0.7052	225.22	19.42	0.4071	327.75
LRMR	36.69	0.9668	38.03	35.59	0.9511	59.71	32.93	0.9300	60.28
MoG-RPCA	35.51	0.9697	43.46	31.62	0.9415	133.78	28.67	0.9101	119.08
NMoG-LRMF	36.67	0.9696	39.08	36.74	0.9737	59.43	33.77	0.9446	58.81
LRTA	34.69	0.9324	48.11	27.29	0.7070	222.79	21.01	0.4687	272.44
PARAFAC	29.41	0.8223	86.12	29.82	0.8395	85.87	25.81	0.7150	158.41
KBR-RPCA	35.49	0.9564	44.01	34.54	0.9611	51.75	32.70	0.9208	59.98
NMoG-Tucker	**37.41**	**0.9706**	**35.59**	**38.72**	**0.9805**	**32.43**	**34.04**	**0.9462**	**51.58**

Figure 4 plots the average PSNR and SSIM values across all bands by different methods, as well as the differences between our results and the competing ones at larger scales. These results suggest that our method achieves leading quantitative performance for most bands. We also observe that the matrix-based competing methods LRMR, MoG-RPCA, and NMoG-LRMF suffer from sharp PSNR and SSIM drops at certain bands in the cases of speckle noise and mixture noise, e.g., bands 40–80 in *DCmall* and bands 40–60 in *Cuprite*. In comparison, our method does not exhibit such phenomenon, which demonstrates its robustness over entire HSI bands.

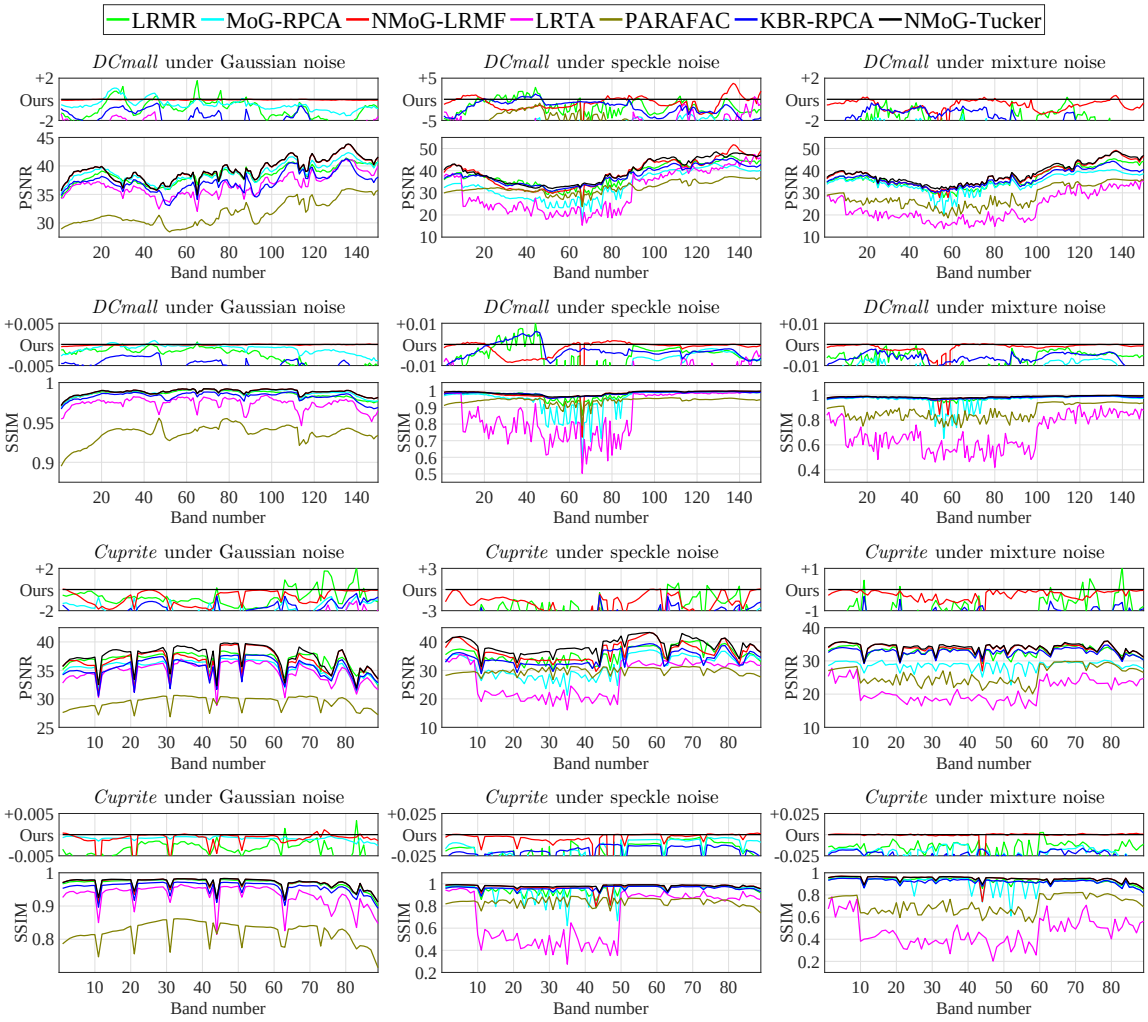

Figure 4. PSNR and SSIM values of each band for simulated HSI denoising. Differences between our results and the competing ones are plotted at larger scales.

Figure 5 shows two denoising examples on typical bands in *DCmall* and *Cuprite*. The noisy band in *DCmall* is severely contaminated by a mixture of Gaussian noise, deadline, and speckle noise; the noisy band in in *Cuprite* is overwhelmed by heavy speckle noise. We observe that the matrix-based methods LRMR, MoG-RPCA, and NMoG-LRMF, although adopting flexible noise priors, cannot adequately separate the original HSIs from such severe degradations, especially for *Cuprite* with fewer spectral bands. As for tensor-based methods, LRTA can hardly reduce the noise, while PARAFAC leaves all the deadlines. Their poor performance is due to the Gaussian noise assumption, which is not able to fit complex noise. In comparison, adopting more intrinsic data and noise priors, KBR-RPCA and our method yield satisfactory denoising results in both cases. Compared with KBR-RPCA, our method preserves finer HSI structures with less residual noise, which can be seen from the demarcated patches and the corresponding error maps. Our better performance is mainly attributed to the non-i.i.d. MoG noise prior, which has a better fitting capability than the Gaussian + sparse assumption in the RPCA framework.

Real HSI denoising. Our experiment uses two real HSI datasets: *Indian Pines* (https://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html) of size $145 \times 145 \times 220$ and *Urban* (http://www.tec.army.mil/hypercube) of size $307 \times 307 \times 210$. In both datasets, some bands are polluted by atmosphere and water absorption with little useful information. We do not remove them, to test the robustness of different methods under severe degradation. The intensity range of the input HSI is scaled to $[0, 1]$.

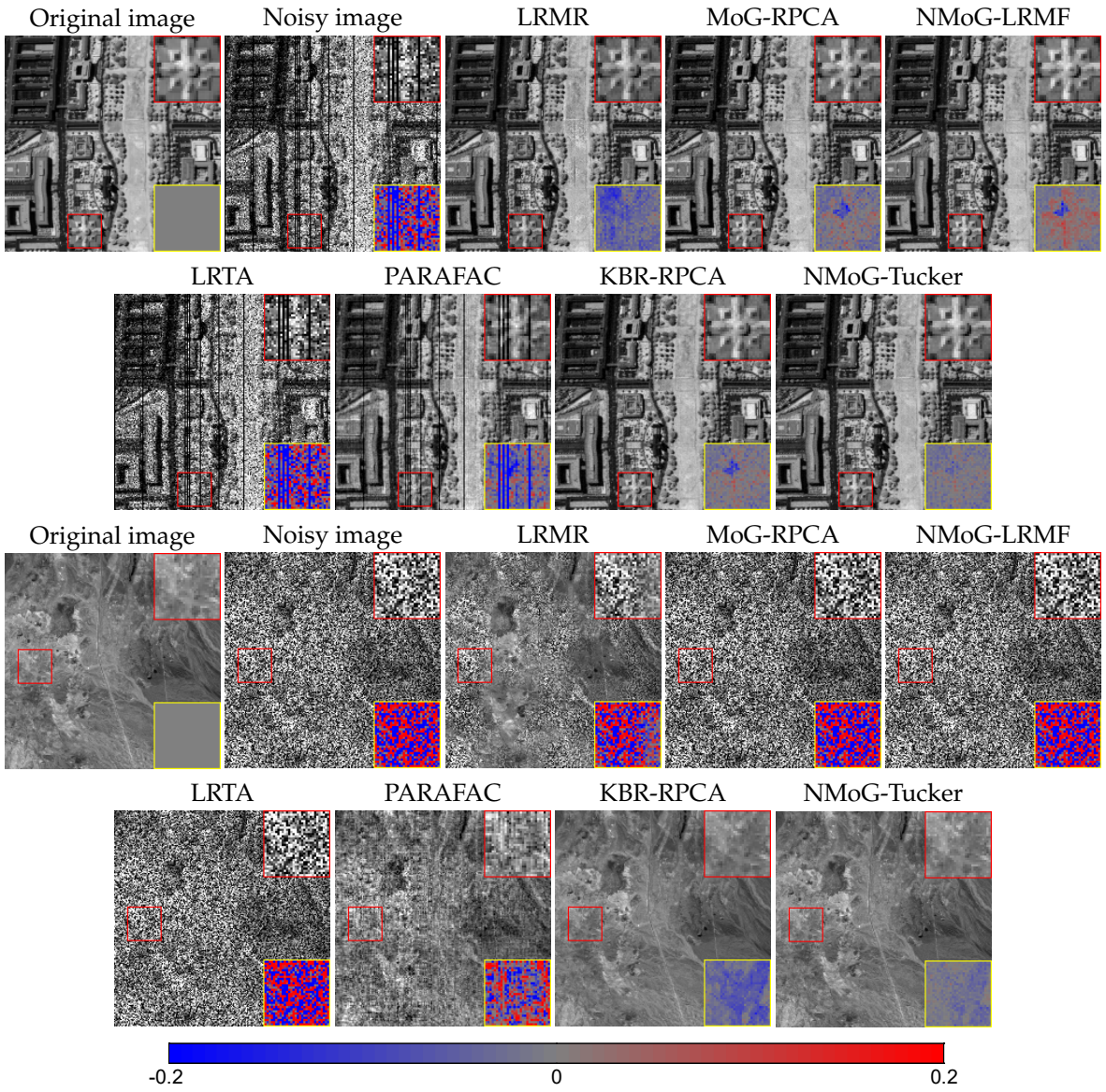

Figure 5. Simulated HSI denoising examples. Top two rows: band 58 in *DCmall* under mixture noise. Bottom two rows: band 43 in *Cuprite* under speckle noise. For better visualization, we show enlargements of two demarcated patches and the corresponding error maps, similarly to Figure 3.

Figure 6 shows a denoising example on band 220 in *Indian Pines*. One can see that the original band is overwhelmed by noise with almost no useful information. From the denoising results by the competing methods, we observe that LRTA fails to handle such severe degradation; MoG-RPCA still leaves much noise in the whole image; LRMR, PARAFAC, and KBR remove more noise but simultaneously lose tiny image details; NMoG-LRMF yields a visually satisfactory result, but seems to produce false edges in the demarcated patches. On the other hand, the proposed method outperforms the competing methods in terms of both noise removal and detail preservation.

Figure 7 presents a classification example on *Indian Pines*. This test aims to provide a task-oriented evaluation of the denoising performance of different methods, from the perspective of the influence on the classification accuracy. In the ground-truth classification result, a total of 10249 samples are divided into 16 classes, and the number of samples in each class ranges from 20 to 2455. To conduct a supervised classification, we randomly choose ten samples for each class as training data, and use the remaining samples in each class as testing data. Then, the support vector machine (SVM) classification [50] is performed on the noisy image and its denoised versions by different methods, and the classification

results are quantitatively evaluated by overall accuracy (OA). It can be seen that noise corruption significantly limits the classification accuracy, and the classification results of the denoised HSIs are more or less improved since the noise is suppressed. Among all denoising methods, our method leads to the highest OA value, demonstrating its superiority in benefiting the SVM classification.

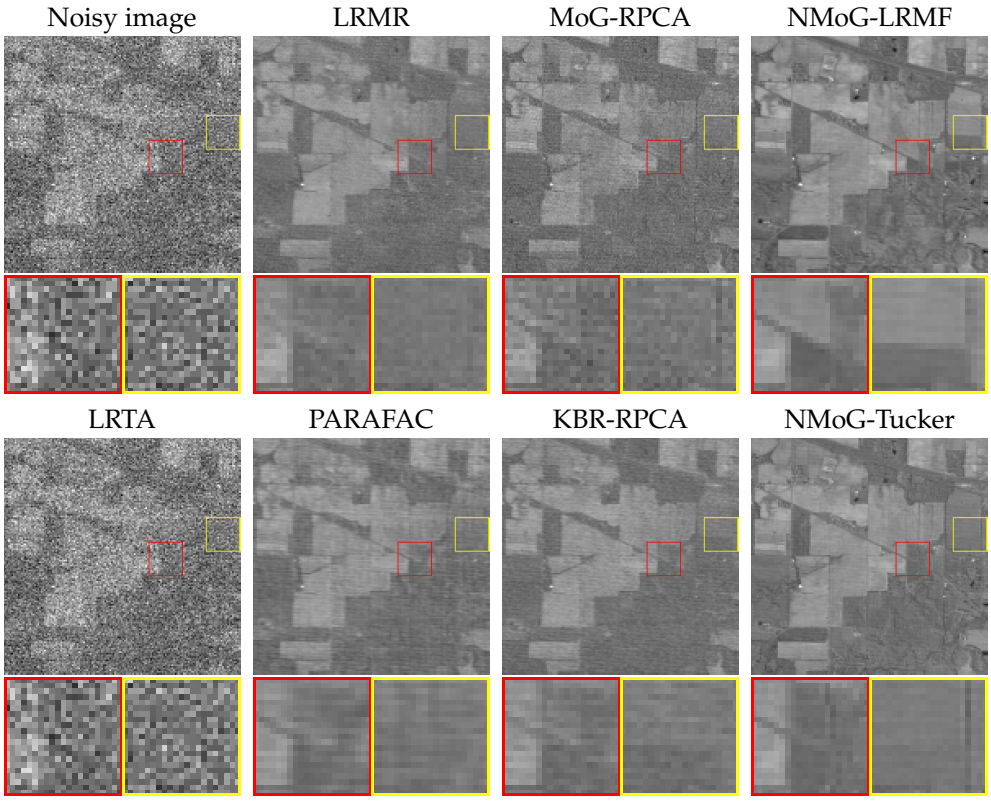

Figure 6. Real HSI denoising example on band 220 in *Indian Pines*. For better visualization, we show enlargements of two demarcated patches.

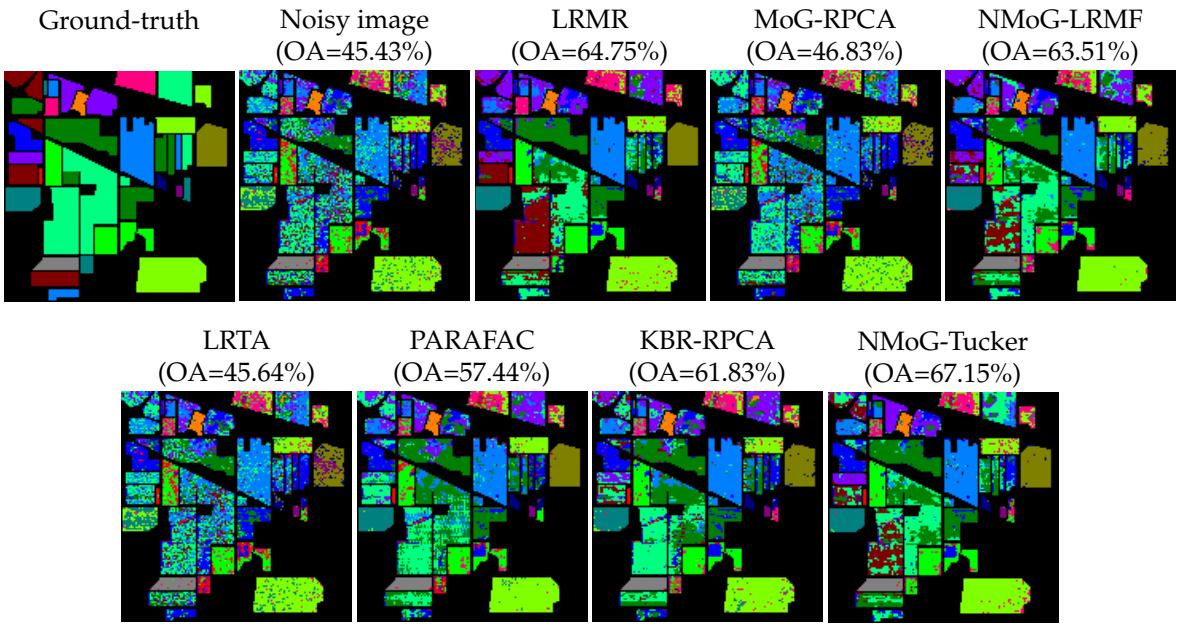

Figure 7. Real HSI classification example on *Indian Pines*. The classification results are obtained by performing SVM on the noisy and the denoised HSIs, and the corresponding OA values are reported in parentheses.

Figure 8 shows a denoising example on band 99 in *Urban* under slight noise. In this example, the original band is mainly corrupted by several vertical stripes with intensity 0.01∼0.02. To visually evaluate the denoising performance, we show color maps of the noise components estimated by different methods, which should highlight the underlying noise with as few image structures as possible. For better visualization, we also plot the corresponding vertical mean profiles. From these results, we observe that LRTA fails to recognize the stripes, while the other competing methods can detect the stripes but simultaneously remove structural information of the original image. In comparison, our method extracts clearly the stripes with very few image features, indicating a more accurate signal-noise separation.

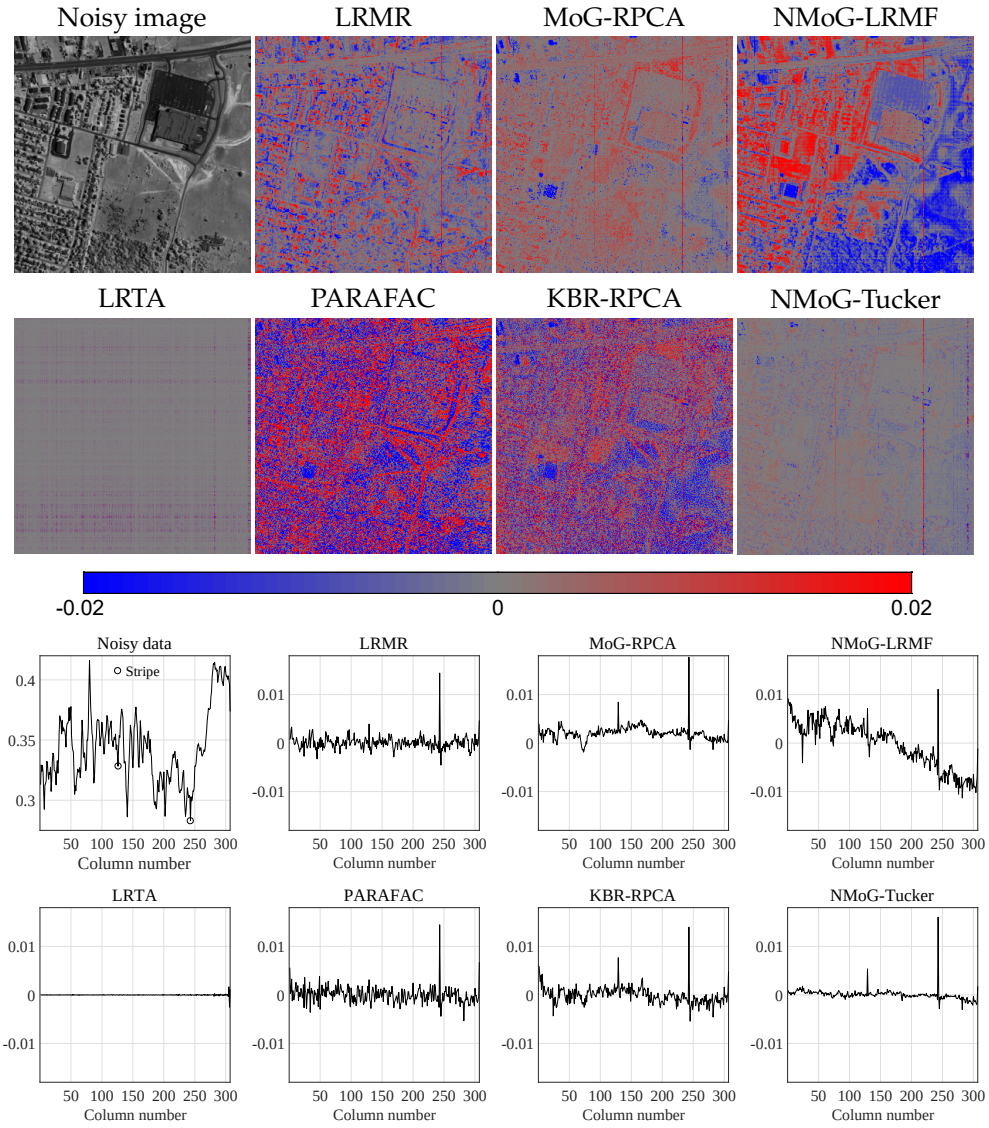

Figure 8. Real HSI denoising example on band 99 in *Urban* under slight noise. Top two rows: the noisy image and color maps of the noise components estimated by different methods (difference between the noisy image and its denoised version). Results highlighting more noise and fewer image structures indicate better denoising performance. Bottom two rows: the corresponding vertical mean profiles, where we mark the locations of stripes by circles in the noisy data.

Figure 9 displays a denoising example on band 206 in *Urban* under severe noise, including the noisy/denoised bands and the corresponding horizontal mean profiles. One can see that the original band is contaminated by a mixture of stripes, deadlines, and other complex noise, leading to rapid fluctuations in the horizontal mean profile. Regarding the denoising results by different methods,

LRTA can hardly suppress the noise; LRMR, MoG-RPCA, PARAFAC, and KBR-RPCA still leave some horizontal stripes, and the corresponding curves show evident fluctuations; NMoG-LRMF removes the noise and produces a smooth curve, but it also introduces some spectral distortions in certain regions, such as the red demarcated patch. Comparatively, our method effectively attenuates the noise and simultaneously reveals the original spatial-spectral information, providing a better trade-off between noise removal and detail preservation.

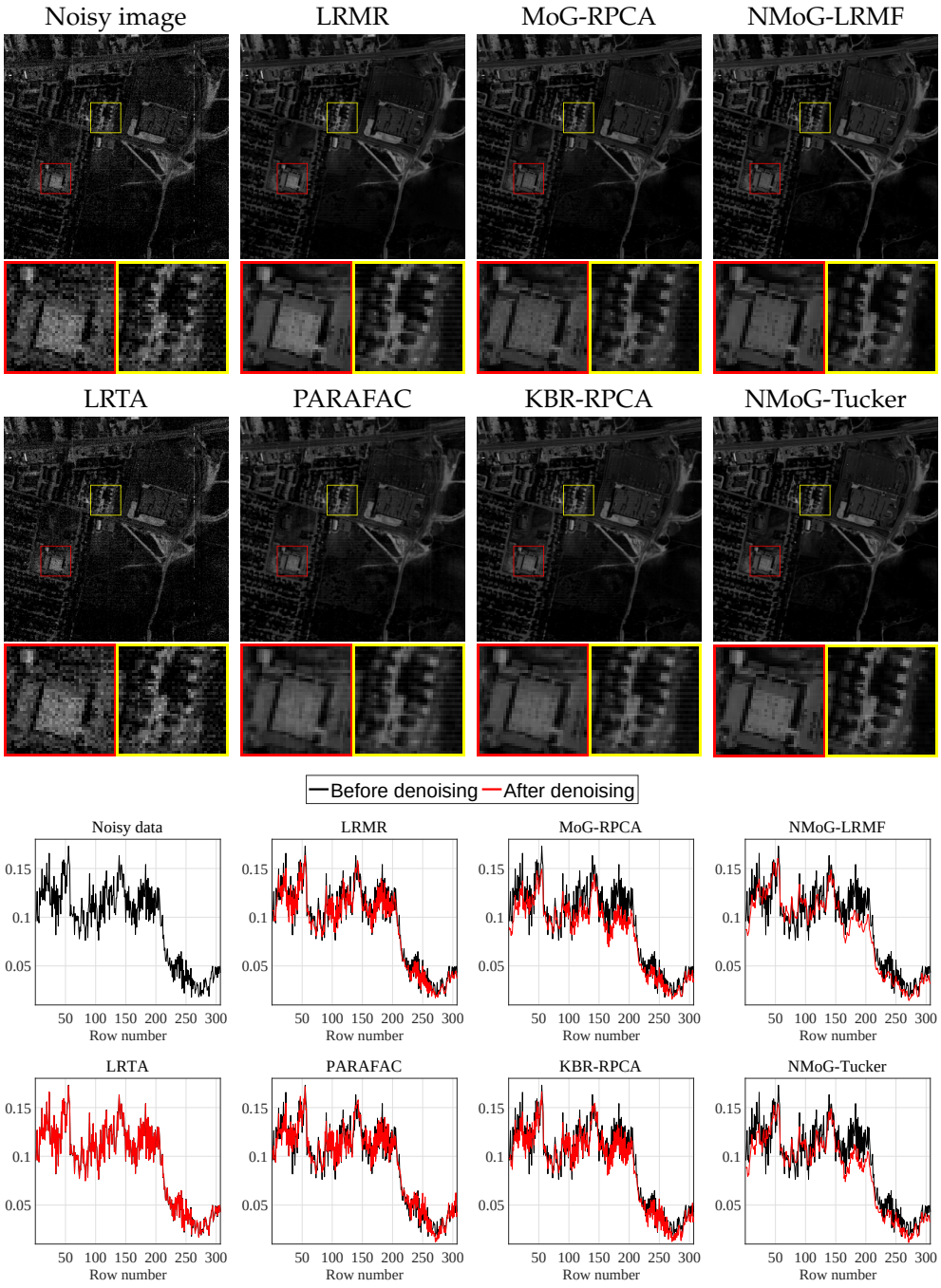

Figure 9. Real HSI denoising example on band 206 in *Urban* under severe noise. Top two rows: the noisy image and the denoising results by different methods, where show enlargements of two demarcated patches for better visualization. Bottom two rows: the corresponding horizontal mean profiles.

4.4. Discussion

In Section 3.3, we have developed adaptive strategies for the selection of hyperparameters involved in our model. These strategies themselves also introduce additional hyperparameters, which are fixed as default settings in our experiments. This section discusses the selection of those hyperparameters and tests their effects on the denoising performance.

The selection of e^{upper} and e^{lower}. The hyperparameters e^{upper} and e^{lower} are introduced in the update formula of the threshold $s_d^{(t)}$ (25), in order to determine the Tucker rank estimation $\{R_d^{(t)}\}_{d=1}^3$. In (25), e^{upper} controls the upper bound of the sum of the dropping singular values in each iteration. In general, a small e^{upper} leads to a slow but stable rank decreasing process; a large e^{upper} makes this process fast but aggressive, increasing the risk of underestimating the true rank. On the other hand, e^{lower} in (25) controls the lower bound of the threshold $s_d^{(t)}$, which provides a mechanism for avoiding overestimating the true rank. Roughly speaking, larger values of e^{lower} make our algorithm easier to reduce the rank. Please note that a too large e^{lower} tends to underestimate the true rank, e.g., if one sets $e^{\text{lower}} = 1$, then the rank decreasing process cannot stop until the rank reduces to zero.

Table 5 investigates the effects of e^{upper} and e^{lower} on the denoising performance of our method. This test is based on synthetic data denoising, and the original data are with size $50 \times 50 \times 50$ and Tucker rank $(20, 15, 10)$. We observe that our method yields rather stable ReErr values with exact estimations of the true rank, under a wide range of settings of e^{upper} and e^{lower}. One exception is the mixture noise case with $e^{\text{upper}} = 10^{-1}$, where the true rank is underestimated, resulting in an evident increase in ReErr. Since our method is robust with a reasonable range of e^{upper} and e^{lower}, we choose $e^{\text{upper}} = 10^{-2}$ and $e^{\text{lower}} = 2/3$ as their default settings in all experiments.

Table 5. ReErr values and estimated ranks of our method under different settings of e^{upper} and e^{lower}. This test is based on synthetic data denoising, and the original data are with size $50 \times 50 \times 50$ and Tucker rank $(20, 15, 10)$. The best results are highlighted in bold.

		Gaussian Noise		Gaussian + Sparse Noise		Mixture Noise	
e^{upper}	e^{lower}	ReErr	$\{R_d^{(\text{end})}\}_{d=1}^3$	ReErr	$\{R_d^{(\text{end})}\}_{d=1}^3$	ReErr	$\{R_d^{(\text{end})}\}_{d=1}^3$
10^{-3}	1/2	1.45e-02	$(20, 15, 10)$	1.69e-02	$(20, 15, 10)$	2.35e-2	$(20, 15, 10)$
	2/3	1.45e-02	$(20, 15, 10)$	1.69e-02	$(20, 15, 10)$	2.35e-2	$(20, 15, 10)$
	3/4	1.45e-02	$(20, 15, 10)$	1.69e-02	$(20, 15, 10)$	2.35e-2	$(20, 15, 10)$
10^{-2}	1/2	1.44e-02	$(20, 15, 10)$	1.68e-02	$(20, 15, 10)$	**2.33e-2**	$(20, 15, 10)$
	2/3	1.44e-02	$(20, 15, 10)$	1.68e-02	$(20, 15, 10)$	**2.33e-2**	$(20, 15, 10)$
	3/4	1.44e-02	$(20, 15, 10)$	1.68e-02	$(20, 15, 10)$	**2.33e-2**	$(20, 15, 10)$
10^{-1}	1/2	**1.44e-02**	$(20, 15, 10)$	**1.68e-02**	$(20, 15, 10)$	8.06e-2	$(19, 15, 10)$
	2/3	**1.44e-02**	$(20, 15, 10)$	**1.68e-02**	$(20, 15, 10)$	8.06e-2	$(19, 15, 10)$
	3/4	**1.44e-02**	$(20, 15, 10)$	**1.68e-02**	$(20, 15, 10)$	8.06e-2	$(19, 15, 10)$

The initialization of $\{L_d\}_{d=1}^3$. The hyperparameter L_d controls the number of Gaussian components in the MoG noise prior (2). In Section 3.3, we have developed an adaptive strategy to reduce L_d from a large starting point to the value matching the noise complexity. However, it remains a problem to choose an appropriate initialization $L_d^{(0)}$.

Table 6 studies the effects of $\{L_d^{(0)}\}_{d=1}^3$ on the denoising performance of our method. This test is based on synthetic data denoising, and the original data are with size $50 \times 50 \times 50$ and Tucker rank $(10, 10, 10)$. From these results, we have the following two observations. First, as expected, the developed selection strategy can find suitable values of $\{L_d^{(\text{end})}\}_{d=1}^3$ fitting the noise distribution. Second, our method performs poorly when $\{L_d^{(0)}\}_{d=1}^3$ is too small to provide sufficient noise fitting capability, while its performance tends to be stable after each $L_d^{(0)}$ is larger than a reasonable value, e.g., 8. Therefore, we choose the default setting of $\{L_d^{(0)}\}_{d=1}^3$ as $(8, 8, 8)$ in all experiments, since it is robust enough to most realistic noise.

Table 6. ReErr values and estimated numbers of Gaussian components of our method using different initializations of $\{L_d\}_{d=1}^3$. This test is based on synthetic data denoising, and the original data are with size $50 \times 50 \times 50$ and Tucker rank $(10, 10, 10)$. The best results are highlighted in bold.

	Gaussian Noise		Gaussian + Sparse Noise		Mixture Noise	
$\{L_d^{(0)}\}_{d=1}^3$	ReErr	$\{L_d^{(\mathrm{end})}\}_{d=1}^3$	ReErr	$\{L_d^{(\mathrm{end})}\}_{d=1}^3$	ReErr	$\{L_d^{(\mathrm{end})}\}_{d=1}^3$
$(1,1,1)$	9.86e-03	$(1,1,1)$	3.54e-01	$(1,1,1)$	7.11e-01	$(1,1,1)$
$(2,2,2)$	9.86e-03	$(1,1,1)$	**1.18e-02**	$(2,2,2)$	9.03e-03	$(2,2,2)$
$(3,3,3)$	9.83e-03	$(1,1,1)$	1.19e-02	$(2,3,2)$	4.15e-03	$(2,3,3)$
$(4,4,4)$	9.83e-03	$(1,1,1)$	1.19e-02	$(2,2,2)$	4.94e-03	$(3,3,3)$
$(5,5,5)$	9.82e-03	$(1,1,1)$	1.19e-02	$(2,2,2)$	3.90e-03	$(3,4,4)$
$(6,6,6)$	9.83e-03	$(1,1,1)$	1.19e-02	$(3,2,2)$	3.30e-03	$(3,4,4)$
$(8,8,8)$	9.83e-03	$(1,1,1)$	1.19e-02	$(2,2,2)$	3.37e-03	$(3,5,4)$
$(10,10,10)$	9.82e-03	$(1,1,1)$	1.18e-02	$(2,2,2)$	3.50e-03	$(5,4,6)$
$(15,15,15)$	9.81e-03	$(1,1,1)$	1.18e-02	$(3,2,2)$	3.24e-03	$(7,8,7)$
$(20,20,20)$	**9.80e-03**	$(1,1,1)$	1.18e-02	$(3,2,2)$	**2.97e-03**	$(7,8,8)$

5. Conclusions

We have proposed a new remote sensing image denoising method under the Bayesian framework. To achieve an effective and robust signal-noise separation, we have formulated the denoising problem as a full Bayesian generative model integrated with a low-Tucker-rank image prior and a non-i.i.d. MoG noise prior. The proposed model has the advantage of preserving the intrinsic low-rank tensor structure of remote sensing images, while exhibiting flexible fitting capability to realistic noise. For an efficient solution to the proposed model, we have designed a variational Bayesian algorithm to infer all involved variables by closed-form equations, as well as adaptive strategies for the selection of hyperparameters. Experimental results have shown that the proposed method is highly effective and superior over the competing methods on synthetic data, MSI, and HSI denoising. Future works include accelerating the numerical implementation and incorporating more advanced image priors to enhance the denoising performance, such as nonlocal self-similarity and deep neural networks [51].

Author Contributions: All authors contribute to methodology design and experimental validation; original draft preparation, T.-H.M.; review and editing, D.M. and Z.X. All authors have read and agree to the published version of the manuscript.

Funding: The research is supported by National Key R&D Program of China (2018YFB1004300), National Natural Science Foundation of China (U1811461, 11690011, 61721002, 11971373, 11901450), MoE-CMCC "Artificial Intelligence" Project (MCM20190701), National Postdoctoral Program for Innovative Talents (BX20180252), and Project funded by China Postdoctoral Science Foundation (2018M643611).

Acknowledgments: The authors would like to thank the editor and the anonymous referees for their valuable suggestions and comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mitra, K.; Sheorey, S.; Chellappa, R. Large-scale matrix factorization with missing data under additional constraints. In *Advances in Neural Information Processing Systems*; 2010; pp. 1651–1659. Available online: http://papers.nips.cc/paper/4111-large-scale-matrix-factorization-with-missing-data-under-additional-constraints (accessed on 17 April 2020).
2. Okatani, T.; Yoshida, T.; Deguchi, K. Efficient algorithm for low-rank matrix factorization with missing components and performance comparison of latest algorithms. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 842–849.
3. Meng, D.; De la Torre, F. Robust matrix factorization with unknown noise. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, NSW, Australia, 1–8 December 2013; pp. 1337–1344.

4. Zhao, Q.; Meng, D.; Xu, Z.; Zuo, W.; Zhang, L. Robust principal component analysis with complex noise. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 22–24 June 2014; pp. 55–63.
5. Zhao, Q.; Meng, D.; Xu, Z.; Zuo, W.; Yan, Y. L_1-norm low-rank matrix factorization by variational Bayesian method. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 825–839. [CrossRef] [PubMed]
6. Cao, X.; Zhao, Q.; Meng, D.; Chen, Y.; Xu, Z. Robust low-rank matrix factorization under general mixture noise distributions. *IEEE Trans. Image Process.* **2016**, *25*, 4677–4690. [CrossRef]
7. Chen, Y.; Cao, X.; Zhao, Q.; Meng, D.; Xu, Z. Denoising hyperspectral image with non-i.i.d. noise structure. *IEEE Trans. Cybern.* **2018**, *48*, 1054–1066. [CrossRef] [PubMed]
8. Yong, H.; Meng, D.; Zuo, W.; Zhang, L. Robust online matrix factorization for dynamic background subtraction. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1726–1740. [CrossRef] [PubMed]
9. Yue, Z.; Meng, D.; Sun, Y.; Zhao, Q. Hyperspectral image restoration under complex multi-band noises. *Remote Sens.* **2018**, *10*, 1631. [CrossRef]
10. Fazel, M.; Hindi, H.; Boyd, S.P. A rank minimization heuristic with application to minimum order system approximation. In Proceedings of the 2001 American Control Conference (Cat. No.01CH37148), Arlington, VA, USA, 25–27 June 2001; pp. 4734–4739.
11. Recht, B.; Fazel, M.; Parrilo, P.A. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.* **2010**, *52*, 471–501. [CrossRef]
12. He, W.; Zhang, H.; Shen, H.; Zhang, L. Hyperspectral image denoising using local low-rank matrix recovery and global spatial–spectral total variation. *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **2018**, *11*, 713–729. [CrossRef]
13. Fazel, M.; Hindi, H.; Boyd, S.P. Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. In Proceedings of the 2003 American Control Conference, Denver, CO, USA, 4–6 June 2003; pp. 2156–2162.
14. Xie, Y.; Qu, Y.; Tao, D.; Wu, W.; Yuan, Q.; Zhang, W. Hyperspectral image restoration via iteratively regularized weighted Schatten *p*-norm minimization. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 4642–4659. [CrossRef]
15. Yang, J.H.; Zhao, X.L.; Ma, T.H.; Chen, Y.; Huang, T.Z.; Ding, M. Remote sensing images destriping using unidirectional hybrid total variation and nonconvex low-rank regularization. *J. Comput. Appl. Math.* **2020**, *363*, 124–144. [CrossRef]
16. Chen, Y.; Guo, Y.; Wang, Y.; Wang, D.; Peng, C.; He, G. Denoising of hyperspectral images using nonconvex low rank matrix approximation. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 5366–5380. [CrossRef]
17. Oh, T.H.; Tai, Y.W.; Bazin, J.C.; Kim, H.; Kweon, I.S. Partial sum minimization of singular values in robust PCA: Algorithm and applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 744–758. [CrossRef] [PubMed]
18. Gu, S.; Xie, Q.; Meng, D.; Zuo, W.; Feng, X.; Zhang, L. Weighted nuclear norm minimization and its applications to low level vision. *Int. J. Comput. Vis.* **2017**, *121*, 183–208. [CrossRef]
19. Kolda, T.G.; Bader, B.W. Tensor decompositions and applications. *SIAM Rev.* **2009**, *51*, 455–500. [CrossRef]
20. Carroll, J.D.; Chang, J.J. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika* **1970**, *35*, 283–319. [CrossRef]
21. Liu, X.; Bourennane, S.; Fossati, C. Denoising of hyperspectral images using the PARAFAC model and statistical performance analysis. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 3717–3724. [CrossRef]
22. Zhao, Q.; Zhang, L.; Cichocki, A. Bayesian CP factorization of incomplete tensors with automatic rank determination. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1751–1763. [CrossRef]
23. Tucker, L.R. Some mathematical notes on three-mode factor analysis. *Psychometrika* **1966**, *31*, 279–311. [CrossRef]
24. Renard, N.; Bourennane, S.; Blanc-Talon, J. Denoising and dimensionality reduction using multilinear tools for hyperspectral images. *IEEE Geosci. Remote Sens. Lett.* **2008**, *5*, 138–142. [CrossRef]
25. Liu, J.; Musialski, P.; Wonka, P.; Ye, J. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 208–220. [CrossRef]

26. Peng, Y.; Meng, D.; Xu, Z.; Gao, C.; Yang, Y.; Zhang, B. Decomposable nonlocal tensor dictionary learning for multispectral image denoising. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 2949–2956.
27. Wang, Y.; Peng, J.; Zhao, Q.; Leung, Y.; Zhao, X.L.; Meng, D. Hyperspectral image restoration via total variation regularized low-rank tensor decomposition. *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **2018**, *11*, 1227–1243. [CrossRef]
28. Kilmer, M.E.; Braman, K.; Hao, N.; Hoover, R.C. Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. *SIAM J. Matrix Anal. Appl.* **2013**, *34*, 148–172. [CrossRef]
29. Fan, H.; Chen, Y.; Guo, Y.; Zhang, H.; Kuang, G. Hyperspectral image restoration using low-rank tensor recovery. *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.* **2017**, *10*, 4589–4604. [CrossRef]
30. Bengua, J.A.; Phien, H.N.; Tuan, H.D.; Do, M.N. Efficient tensor completion for color image and video recovery: Low-rank tensor train. *IEEE Trans. Image Process.* **2017**, *26*, 2466–2479. [CrossRef] [PubMed]
31. Oseledets, I.V. Tensor-train decomposition. *SIAM J. Sci. Comput.* **2011**, *33*, 2295–2317. [CrossRef]
32. Yang, J.H.; Zhao, X.L.; Ji, T.Y.; Ma, T.H.; Huang, T.Z. Low-rank tensor train for tensor robust principal component analysis. *Appl. Math. Comput.* **2020**, *367*, 124783. [CrossRef]
33. Liu, Y.; Long, Z.; Huang, H.; Zhu, C. Low CP rank and Tucker rank tensor completion for estimating missing components in image data. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, to be published. [CrossRef]
34. Zhao, Q.; Meng, D.; Kong, X.; Xie, Q.; Cao, W.; Wang, Y.; Xu, Z. A novel sparsity measure for tensor recovery. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 271–279.
35. Xie, Q.; Zhao, Q.; Meng, D.; Xu, Z. Kronecker-basis-representation based tensor sparsity and its applications to tensor recovery. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1888–1902. [CrossRef]
36. Candès, E.J.; Li, X.; Ma, Y.; Wright, J. Robust principal component analysis? *J. ACM* **2011**, *58*, 11. [CrossRef]
37. Meng, D.; Xu, Z.; Zhang, L.; Zhao, J. A cyclic weighted median method for L_1 low-rank matrix factorization with missing entries. In Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, Bellevue, WA, USA, 14–18 July 2013; pp. 704–710.
38. Zhang, H.; He, W.; Zhang, L.; Shen, H.; Yuan, Q. Hyperspectral image restoration using low-rank matrix recovery. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 4729–4743. [CrossRef]
39. Maz'ya, V.; Schmidt, G. On approximate approximations using Gaussian kernels. *IMA J. Numer. Anal.* **1996**, *16*, 13–29. [CrossRef]
40. Yue, Z.; Yong, H.; Meng, D.; Zhao, Q.; Leung, Y.; Zhang, L. Robust multiview subspace learning with nonindependently and nonidentically distributed complex noise. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, to be published. [CrossRef] [PubMed]
41. Chen, X.; Han, Z.; Wang, Y.; Zhao, Q.; Meng, D.; Tang, Y. Robust tensor factorization with unknown noise. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 5213–5221.
42. Luo, Q.; Han, Z.; Chen, X.; Wang, Y.; Meng, D.; Liang, D.; Tang, Y. Tensor RPCA by Bayesian CP factorization with complex noise. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 5029–5038.
43. Chen, X.; Han, Z.; Wang, Y.; Zhao, Q.; Meng, D.; Lin, L.; Tang, Y. A generalized model for robust tensor factorization with noise modeling by mixture of Gaussians. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 5380–5393. [CrossRef] [PubMed]
44. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: Berlin, Germany, 2006.
45. Babacan, S.D.; Luessi, M.; Molina, R.; Katsaggelos, A.K. Sparse Bayesian methods for low-rank matrix estimation. *IEEE Trans. Signal Process.* **2012**, *60*, 3964–3977. [CrossRef]
46. Hurley, N.; Rickard, S. Comparing measures of sparsity. *IEEE Trans. Inf. Theory* **2009**, *55*, 4723–4741. [CrossRef]
47. Wald, L. *Data Fusion: Definitions and Architectures: Fusion of Images of Different Spatial Resolutions*; Presses des l'Ecole MINES: Paris, France, 2002.
48. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef]

49. Yasuma, F.; Mitsunaga, T.; Iso, D.; Nayar, S.K. Generalized assorted pixel camera: Postcapture control of resolution, dynamic range, and spectrum. *IEEE Trans. Image Process.* **2010**, *19*, 2241–2253. [CrossRef]
50. Melgani, F.; Bruzzone, L. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Trans. Geosci. Remote Sens.* **2004**, *42*, 1778–1790. [CrossRef]
51. Zhao, X.L.; Xu, W.H.; Jiang, T.X.; Wang, Y.; Ng, M.K. Deep plug-and-play prior for low-rank tensor completion. *Neurocomputing* to be published. [CrossRef]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).