Pattern Recognition 103 (2020) 107310

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Adaptive quantile low-rank matrix factorization

Shuang Xu, Chunxia Zhang*, Jiangshe Zhang

School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China

ARTICLE INFO

Article history:
Received 24 May 2019
Revised 8 February 2020
Accepted 24 February 2020
Available online 25 February 2020

Keywords:
Low-rank matrix factorization
Mixture of asymmetric Laplace distributions
Expectation maximization algorithm
Skew noise

ABSTRACT

Low-rank matrix factorization (LRMF) has received much popularity owing to its successful applications in both computer vision and data mining. By assuming noise to come from a Gaussian, Laplace or mixture of Gaussian distributions, significant efforts have been made on optimizing the (weighted) L_1 or L_2-norm loss between an observed matrix and its bilinear factorization. However, the type of noise distribution is generally unknown in real applications and inappropriate assumptions will inevitably deteriorate the behavior of LRMF. On the other hand, real data are often corrupted by skew rather than symmetric noise. To tackle this problem, this paper presents a novel LRMF model called AQ-LRMF by modeling noise with a mixture of asymmetric Laplace distributions. An efficient algorithm based on the expectation-maximization (EM) algorithm is also offered to estimate the parameters involved in AQ-LRMF. The AQ-LRMF model possesses the advantage that it can approximate noise well no matter whether the real noise is symmetric or skew. The core idea of AQ-LRMF lies in solving a weighted L_1 problem with weights being learned from data. The experiments conducted on synthetic and real data sets show that AQ-LRMF outperforms several state-of-the-art techniques. Furthermore, AQ-LRMF also has the superiority over the other algorithms in terms of capturing local structural information contained in real images.

1. Introduction

Researchers from machine learning [1], computer vision [2] and statistics [3] have paid increasing attention to low-rank matrix factorization (LRMF) [4]. Generally speaking, many real-world modeling tasks can be attributed as the problems of LRMF. The tasks include but are not limited to recommender systems [5], subspace learning [6–9], link prediction [10], computational biology [11,12] and image denoising [13,14].

The key idea of LRMF is to approximate a given matrix by the product of two low-rank matrices. Specifically, given an observed matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$, LRMF aims at solving the optimization problem

$$\min_{\mathbf{U},\mathbf{V}} ||\mathbf{\Omega} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^{\mathrm{T}})||, \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{m\times r}, \mathbf{V} \in \mathbb{R}^{n\times r}$ are two low-rank matrices (usually, $r \ll \min(m, n)$) and $\odot$ denotes the Hadamard product, that is, the element-wise product. The indicator matrix $\mathbf{\Omega} = (\omega_{ij})_{m\times n}$ implies whether some elements are missing, where $\omega_{ij} = 1$ if x_{ij} is non-missing and 0 otherwise. The symbol $||\cdot||$ indicates a certain norm of a matrix, in which the most prevalent one is L_2 norm. It is well-known that singular value decomposition provides a closed-form solution for L_2-norm LRMF without missing entries. With respect to the problems with $\mathbf{X}$ containing missing entries, researchers have presented many fast algorithms such as damped Newton algorithm [15], Chen's method [16], and Damped Wiberg (DW) [17] to solve Eq. (1). In the literature of LRMF, the most popular algorithm is DW proposed in [17]. The key idea of DW is to incorporate a damping factor into the Wiberg method to solve the corresponding problem. Although the L_2-norm LRMF greatly facilitates theoretical analysis, it provides the best solution in the sense of maximum likelihood principle only when noise is indeed sampled from a Gaussian distribution. If noise is from a heavy-tailed distribution or data are corrupted by outliers, however, L_2-norm LRMF may break down. Thereafter, L_1-norm LRMF begins to gain increasing interests of both theoretical researchers and practitioners due to its robustness [18]. In fact, L_1-norm LRMF hypothesizes that noise is from a Laplace distribution. As is often the case with L_2-norm LRMF, L_1-norm LRMF may provide unexpected results as well if its assumptions are violated.

Because the noise in real data generally deviates far away from a Gaussian or Laplace distribution, analysts are no longer satisfied with L_1- or L_2-norm LRMF. To further improve the robustness of LRMF, researchers attempt to directly model unknown noise via a mixture of Gaussians (MoG) due to its good property to universally approximate any continuous distribution [19,20]. Nevertheless, the technique cannot fit real noise precisely in some complex cases. For example, in theory, infinite Gaussian components are required to approximate a Laplace distribution. In practice, we

* Corresponding author.
E-mail address: cxzhang@mail.xjtu.edu.cn (C. Zhang).

https://doi.org/10.1016/j.patcog.2020.107310
0031-3203/

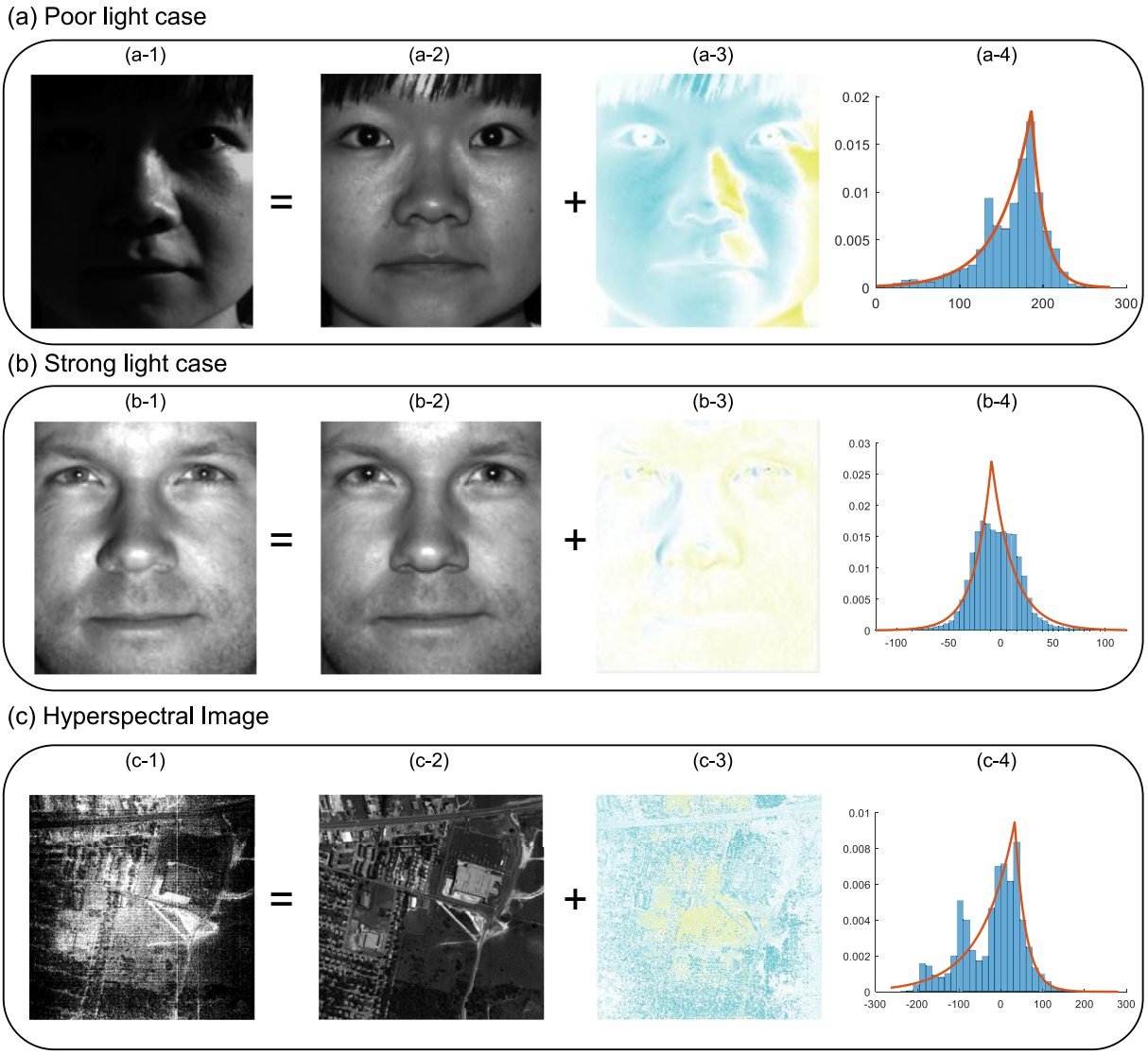


Fig. 1. (a) and (b) illustrate two face images corresponding to underexposure and overexposure cases, respectively. In particular, (a-1) and (b-1) are face images captured with improper light sources while (a-2) and (b-2) are face images obtained with proper light sources. (a-3) and (b-3) are residual images in which the yellow (blue) locations indicate positive (negative) values. (a-4) and (b-4) illustrate the histograms of the residual images as well as the PDF curves fitted by ALD with $\alpha_a = 115, \kappa_a = 0.71, \lambda_a = 0.05$ and $\alpha_b = -9, \kappa_b = 0.44, \lambda_b = 0.11$, respectively. The skewness of the residual face in (a-3) is -0.72 whilst that for (b-3) is 0.69. (c) shows a hyperspectral image. Similar to cases (a) and (b), the images from (c-1) to (c-4) are original, de-noised, noise images and the histogram of residuals, respectively. The skewness of the noise image (c-3) is -0.55. In (c-4), the fitted ALD is obtained with $\alpha = 33$, $\kappa = 0.75$ and $\lambda = 0.05$. Obviously, the distributions of noise shown here are all asymmetric.

only utilize finite Gaussian components due to the characteristics of MoG. On the other hand, Gaussian, Laplace and MoG distributions are all symmetric. In the situations with real noise being skew, it is obviously inappropriate to assume a symmetric noise distribution.

As a matter of fact, there are no strictly symmetric noise in real images. For instance, Fig. 1 illustrates several examples in which the real noise is either skewed to the left (e.g., (a-4) and (c-4)) or the right (e.g., (b-4)). In these situations, the symmetric distributions like Gaussian or Laplace are inadequate to approximate the noise. In statistics, scholars usually make use of quantile regression to deal with an asymmetric noise distribution [21]. Consider a simple case that there is only one covariate X, the quantile regression coefficient β can be obtained by

$$\hat{\beta}_\kappa = \arg\min_{\beta} \sum_{i=1}^{n} \rho_\kappa(y_i - x_i\beta), \tag{2}$$

where $\{(y_i, x_i)\}_{i=1}^n$ are n observations and κ is a pre-defined asymmetry parameter. Moreover, the quantile loss $\rho_\kappa(\cdot)$ is defined as

$$\begin{aligned}\rho_\kappa(\epsilon) &= \epsilon[\kappa - \mathbb{I}(\epsilon < 0)] \\ &= |\epsilon|[\kappa\mathbb{I}(\epsilon \geq 0) + (1-\kappa)\mathbb{I}(\epsilon < 0)]\end{aligned} \tag{3}$$

with $\mathbb{I}(\cdot)$ being the indicator function. Evidently, the quantile loss with $\kappa = 1/2$ corresponds to the L_1-norm loss. From the Bayesian viewpoint, the estimate obtained by minimizing the quantile loss in (2) coincides with the result by assuming noise coming from an asymmetric Laplace distribution (ALD) [22,23].

To overcome the shortcomings of existing LRMF methods that they assume a specific type of noise distribution, we present in this paper an adaptive quantile LRMF (AQ-LRMF) algorithm. The key idea of AQ-LRMF is to model noise via a mixture of asymmetric Laplace distributions (MoAL). Due to the existence of some latent variables, the expectation maximization (EM) algorithm is employed to estimate the parameters in AQ-LRMF under the maximum likelihood framework. The novelty of AQ-LRMF and our main contributions can be summarized as follows.

(1). The M-step of the EM algorithm corresponds to a weighted L_1-norm LRMF, where the weights encode the information about skewness and outliers.
(2). The weights are automatically learned from data under the framework of EM algorithm.
(3). Different from quantile regression, our method does not need to pre-define the asymmetry parameter of quantile loss, because it is adaptively determined by data.
(4). Our model can capture local structural information contained in some real images, although we do not encode it into our model.

Our conducted experiments show that AQ-LRMF can effectively approximate many different kinds of noise. If the noise has a strong tendency to take a particular sign, AQ-LRMF will produce better estimates than a method which assumes a symmetric noise distribution. In comparison with several state-of-the-art methods, the superiority of our method is demonstrated in both synthetic and real-data experiments such as image inpainting, face modeling, hyperspectral image (HSI) construction and so on. The code of this paper is available at https://xsxjtu.github.io/Projects/MoAL/main.html.

The rest of the paper is organized as follows. Section 2 presents related work of LRMF. In Section 3, we propose the AQ-LRMF model and also provide an efficient learning algorithm for it. Section 4 includes experimental studies. At last, some conclusions and future work are offered in Section 5.

2. Related work

The study of robust LRMF has a long history. Srebro and Jaakkola [24] suggested to use a weighted L_2 loss to improve LRMF's robustness to noise and missing data. The problem can be solved by a simple but efficient EM algorithm. However, its capability strongly relies on the chosen weights while it is not easy to automatically select proper weights. Since then, the research community began to replace L_2 loss with L_1 loss. One of the earliest explorations is made by Ke and Kanade [18]. They solved the L_1-norm LRMF by alternated linear or quadratic programming, but the speed is slow. Thereafter, many researchers attempted to develop some variants of L_1-norm LRMF to enhance its running speed as well as performance. Roughly speaking, the improved L_1-norm LRMF can be classified into two groups.

On the one hand, researchers strived to propose fast numerical algorithms for L_1-norm LRMF. Under this framework, Eriksson and Hengel [25] developed the L_1-Wiberg algorithm for calculating the low-rank factorization of a matrix which minimizes the L_1 norm in the presence of missing data. Meng et al. [26] proposed a computationally efficient algorithm, cyclic weighted median (CWM) method, by solving a sequence of scalar minimization sub-problems to obtain the optimal solution. Recently, Kim et al. [27] used alternating rectified gradient method to solve a large-scale L_1-norm LRMF.

On the other hand, researchers tried to improve L_1-norm LRMF's performance by inserting a penalty into the objective function. Okutomi et al. [28] modified the objective function of L_1-Wiberg by adding the nuclear norm of $\mathbf{V}$ and the orthogonality constraint on $\mathbf{U}$. This method has been shown to be effective in addressing structure from motion issue. Inspired by majorization-minimization technique, Lin et al. [29] proposed LRMF-MM to solve an LRMF optimization task with L_1 loss plus the L_2-norm penalty that is placed on $\mathbf{U}$ and $\mathbf{V}$. In each step, they upper bound the original objective function by a strongly convex surrogate and then minimize the surrogate. Li et al. [30] considered a similar problem, but they replace the L_2-norm penalty imposed on $\mathbf{U}$ with $\mathbf{U}^{\mathrm{T}}\mathbf{U}=\mathbf{I}$. This model is solved by augmented Lagrange multiplier method. Furthermore, the authors of [30] designed a heuristic rank estimator for their model. Even though the above-mentioned approaches improved L_1-norm LRMF from a certain aspect, one has to notice that L_1 loss actually corresponds to the Laplace-distributed noise. Put in another way, these methods implicitly assume that the noise comes from a Laplace distribution. When the real distribution of noise deviates too far from Laplace, the robustness of L_1 LRMF will be suspectable.

Recently, the research community began to focus on probabilistic extensions of robust matrix factorizations. Generally speaking, it is assumed that $\mathbf{X}=\mathbf{U}\mathbf{V}^{\mathrm{T}}+\mathbf{E}$, where $\mathbf{E}$ is a noise matrix. Lakshminarayanan et al. [31] replaced Gaussian noise with Gaussian scale mixture noise. Nevertheless, it may be ineffective when processing heavy-tailed (such as Laplace-type) noise. Wang et al. [32] proposed a probabilistic L_1-norm LRMF, but they did not employ a fully Bayesian inference process. Beyond Laplace noise, Meng and Torre [19] presented a robust LRMF with unknown noise modeled by an MoG. In essence, the method iteratively optimizes $\min_{\mathbf{U},\mathbf{V},\boldsymbol{\theta}} ||\mathbf{W}(\boldsymbol{\theta})\odot(\mathbf{X}-\mathbf{U}\mathbf{V}^{\mathrm{T}})||_{L_2}$, where the vector $\boldsymbol{\theta}$ includes the MoG parameters which are automatically updated during optimization, and $\mathbf{W}(\boldsymbol{\theta})$ is the weight function of $\boldsymbol{\theta}$. Due to the benefit to adaptively assign small weights to corrupted entries, MoG-LRMF has been reported to be fairly effective. More recently, Cao et al. [33] presented a novel LRMF model by assuming noise as a mixture of exponential power (MoEP) distributions and offered both a generalized expectation maximization (GEM) algorithm and a variational GEM to infer all parameters involved in their proposed model.

In addition, it is worth mentioning that robust principle component analysis (robust PCA) [34] considers an issue similar to LRMF, that is,

$$\min_{\mathbf{A},\mathbf{E}} \operatorname{rank}(\mathbf{A})+\lambda||\mathbf{E}||_{L_0} \quad \text{s.t. } \mathbf{X}=\mathbf{A}+\mathbf{E}. \tag{4}$$

The underlying assumption of robust PCA is that the original data can be decomposed into the sum of a low-rank matrix and a sparse outlier matrix (i.e., the number of non-zero elements in $\mathbf{E}$ is small). Clearly, $\mathbf{A}$ plays the same role as the product of $\mathbf{U}$ and $\mathbf{V}^{\mathrm{T}}$. Since Eq. (4) involves a non-convex objective function, [34] consider a tractable convex alternative, called principal component pursuit, to handle the corresponding problem, namely,

$$\min_{\mathbf{A},\mathbf{E}} ||\mathbf{A}||_*+\lambda||\mathbf{E}||_{L_1} \quad \text{s.t. } \mathbf{X}=\mathbf{A}+\mathbf{E}, \tag{5}$$

where $\|\cdot\|_*$ denotes the nuclear norm. Nevertheless, principal component pursuit may sometimes fail to recover $\mathbf{E}$ when the real observation is also corrupted by a dense inlier matrix. To overcome this shortcoming, Zhou et al. [35] proposed the stable principal component pursuit (SPCP) by solving

$$\min_{\mathbf{A},\mathbf{E}} ||\mathbf{A}||_*+\lambda||\mathbf{E}||_{L_1} \quad \text{s.t. } ||\mathbf{X}-\mathbf{A}-\mathbf{E}||_{L_2}\le\varepsilon. \tag{6}$$

Actually, the underlying assumption of SPCP is $\mathbf{X}=\mathbf{A}+\mathbf{N}+\mathbf{E}$, where $\mathbf{A}$ is a low-rank component, $\mathbf{E}$ is a sparse matrix representing the gross sparse errors (i.e., outliers) in the observed data $\mathbf{X}$ and $\mathbf{N}$ is the small-magnitude noise that can be modeled by a Gaussian distribution. Both theoretical analysis and experiments have shown that SPCP guarantees the stable recovery of $\mathbf{E}$ [34,35].

Actually, our model AQ-LRMF (details are provided in Section 3) is a probabilistic extension of robust matrix factorization. The differences between AQ-LRMF and existing approaches can be summarized as follows. First, the noise in AQ-LRMF is assumed to be asymmetric (i.e., the noise is modeled with an MoAL), while the noise in existing methods is governed by a symmetric distribution, such as Gaussian and Laplacian. Second, the parameters in AQ-LRMF are inferred by an EM algorithm. In the M-step, the optimization with regard to $\mathbf{U}$ and $\mathbf{V}$ is cast into a weighted L_1-norm LRMF, where the weights are automatically learned from

data. Meanwhile, the weights embody the information about outliers and skewness. In contrast, the M-step in MoG-LRMF leads to a weighted L_2-norm LRMF. Because L_1 norm is more robust to noise and outliers, AQ-LRMF also inherits this good property to perform better than MoG-LRMF in handling various kinds of noise.

3. Adaptive quantile LRMF (AQ-LRMF)

3.1. Motivation

Generally speaking, researchers employ the L_2 or L_1 loss function when solving a low-rank matrix factorization problem. As argued in introduction, L_2 or L_1 loss implicitly hypothesizes that the noise distribution is symmetric. Nevertheless, the noise in real data is often asymmetric and Fig. 1 illustrates several examples.

In Fig. 1, there are two face images and a hyperspectral image. Fig. 1 (a) displays a face image that is captured with a poor light source. There are cast shadows in a large area, while there exists an overexposure phenomenon in a small area. As a result, the noise is negative skew. By contrast, Fig. 1 (b) illustrates a face image which is captured under a strong light source. Because of the camera range settings, there are saturated pixels, especially on the forehead. Under this circumstance, the noise is positive skew. Fig. 1 (c) shows a hyperspectral image that is mainly corrupted by stripe and Gaussian noise. Its residual image indicates that the signs of the noise are unbalanced, i.e., more pixels are corrupted by noise with negative values. Actually, the skewness values of three residual (noise) images are -0.72, 0.69 and -0.55, respectively. Note that a symmetric distribution has skewness 0, the noise contained in these real data sets is thus asymmetric.

As a matter of fact, the noise in real data can hardly be governed by a strictly symmetric probability distribution. Therefore, it is natural to utilize an asymmetric distribution to model realistic noise. In statistics, researchers usually make use of a quantile loss function defined in (3) to address this issue. It has been shown that quantile loss function corresponds to the situation that noise is from an asymmetric Laplace distribution [22,23]. In order to further improve the performance of LRMF, we attempt to use a mixture of asymmetric Laplacian distributions (MoAL) to approximate noise.

3.2. Asymmetric Laplace distribution

In what follows, we use $AL(\epsilon|\alpha, \lambda, \kappa)$ to denote an ALD with location, scale and asymmetric parameters α, $\lambda > 0$ and $0 < \kappa < 1$, respectively. Its probability distribution function (PDF) [23] is

$$\begin{aligned} p(x;\alpha,\lambda,\kappa) &= \lambda\kappa(1-\kappa)\begin{cases}\exp\left(\lambda(1-\kappa)(x-\alpha)\right), & \text{if} \quad x<\alpha;\\ \exp\left(-\lambda\kappa(x-\alpha)\right), & \text{if} \quad x\geq\alpha;\end{cases}\\ &= \lambda\kappa(1-\kappa)\exp(-|x-\alpha|\lambda[\kappa\mathbb{I}(x-\alpha\geq 0)\\ &\quad +(1-\kappa)\mathbb{I}(x-\alpha<0)]). \end{aligned} \tag{7}$$

Obviously, the location parameter α is exactly the mode of an ALD. In Fig. 2, we demonstrate the PDF curves for several ALDs with different parameters. In general, the skewness of an ALD, say, sk_{ALD}, takes value in the interval $(-2, 2)$ and it is controlled by the asymmetry parameter κ. An ALD is positive skew if $0 < \kappa < 0.5$, and is negative skew if $0.5 < \kappa < 1$. If $\kappa = 0.5$, the ALD becomes a Laplace distribution. The smaller the scale parameter λ is, the more heavy-tailed an ALD is.

It is worthwhile that skew Gaussian distributions [36] are also prevailing in both theory and applications. However, it is not ideal for the analysis of LRMF. On the one hand, the PDF of a skew Gaussian distribution is complex. On the other hand, its skewness lies in $(-1, 1)$ which is only a subset of the range of sk_{ALD}. Due to this fact, the fitting capability of an ALD is greater than that of a skew Gaussian distribution.

3.3. AQ-LRMF model

To enhance the robustness of LRMF in situations with skew and heavy-tailed noise, we propose an adaptive quantile LRMF (AQ-LRMF) by modeling unknown noise as an MoAL. In particular, we consider a generative model of the observed matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$. For each entry x_{ij}, suppose that there is

$$x_{ij} = \mathbf{u}_i\mathbf{v}_j^{\mathrm{T}} + \epsilon_{ij}, \tag{8}$$

where $\mathbf{u}_i$ is the ith row of $\mathbf{U}$, $\mathbf{v}_j$ is the jth row of $\mathbf{V}$, and ϵ_{ij} is the noise. In AQ-LRMF, we assume that ϵ_{ij} is distributed as an MoAL, namely,

$$p(\epsilon_{ij}) = \sum_{s=1}^{S}\pi_s AL_s(\epsilon_{ij}|0,\lambda_s,\kappa_s), \tag{9}$$

in which $AL_s(\epsilon_{ij}|0, \lambda_s, \kappa_s)$ stands for an asymmetric distribution with parameters $\alpha = 0, \lambda = \lambda_s$ and $\kappa = \kappa_s$. Meanwhile, π_s indicates the mixing proportion with $\pi_s \geq 0$ and $\sum_{s=1}^{S}\pi_s = 1$, and S means the number of mixture components.

To facilitate the estimation of unknown parameters, we introduce some latent binary variables $z_{ij1}, z_{ij2}, \cdots, z_{ijS}$ where $z_{ijs} \in \{0, 1\}$ and $\sum_{s=1}^{S} z_{ijs} = 1$. To ease presentation, let each noise ϵ_{ij} be equipped with an indicator vector $\mathbf{z}_{ij} = (z_{ij1}, z_{ij2}, \cdots, z_{ijS})^{\mathrm{T}}$. Here, $z_{ijs} = 1$ indicates that the noise ϵ_{ij} is drawn from the sth AL distribution. Evidently, $\mathbf{z}_{ij}$ follows a multinomial distribution, i.e., $\mathbf{z}_{ij} \sim \mathcal{M}(\pi_1, \cdots, \pi_S)$. Under these assumptions, we can have

$$p(\epsilon_{ij}) = \prod_{s=1}^{S}\left[\pi_s AL_s(\epsilon_{ij}|0,\lambda_s,\kappa_s)\right]^{z_{ijs}}. \tag{10}$$

Now, it is easy to obtain the probability of x_{ij} as

$$p(x_{ij}|\mathbf{u}_i,\mathbf{v}_j,\boldsymbol{\lambda},\mathbf{K},\boldsymbol{\pi}) = \prod_{s=1}^{S}\left[\pi_s AL_s(x_{ij}|\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}},\lambda_s,\kappa_s)\right]^{z_{ijs}}, \tag{11}$$

where $\boldsymbol{\lambda} = \{\lambda_1, \lambda_2, \cdots, \lambda_S\}$, $\mathbf{K} = \{\kappa_1, \kappa_2, \cdots, \kappa_S\}$ and $\boldsymbol{\pi} = \{\pi_1, \pi_2, \cdots, \pi_S\}$ are unknown parameters. To estimate $\mathbf{U}, \mathbf{V}$ as well as $\boldsymbol{\lambda}, \mathbf{K}, \boldsymbol{\pi}$, we employ the maximum likelihood principle. Consequently, the goal is to maximize the log-likelihood function of complete data shown below, namely,

$$\ell(\mathbf{U},\mathbf{V},\boldsymbol{\lambda},\mathbf{K},\boldsymbol{\pi}) = \sum_{(i,j)\in\Omega}\sum_{s=1}^{S} z_{ijs}\left[\log AL_s(x_{ij}|\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}},\lambda_s,\kappa_s) + \log\pi_s\right], \tag{12}$$

where Ω denotes the index set of the non-missing entries of data. Subsequently, we will discuss how to maximize the log-likelihood function $\ell(\mathbf{U}, \mathbf{V}, \boldsymbol{\lambda}, \mathbf{K}, \boldsymbol{\pi})$ to get our interested items.

3.4. Learning of AQ-LRMF

Since each x_{ij} associates with an indicator vector $\mathbf{z}_{ij} = (z_{ij1}, z_{ij2}, \cdots, z_{ijS})^{\mathrm{T}}$ in which z_{ijk}'s $(k = 1, \cdots, S)$ are latent variables, the EM algorithm [37] is utilized to train the AQ-LRMF model. Particularly, the algorithm needs to iteratively implement the following two steps (i.e., E-step and M-step) to maximize the likelihood of the corresponding problem until the algorithm converges. For ease of exposition, we let $e_{ij} = x_{ij} - \mathbf{u}_i\mathbf{v}_j^{\mathrm{T}}$ and abbreviate $AL_s(e_{ij}|0, \lambda_s, \kappa_s)$ as $AL_s(e_{ij})$ in the following discussions.

E-step: Compute the conditional expectation of the latent variable z_{ijs} as

$$\gamma_{ijs} = E(z_{ijs}|x_{ij}) = \frac{\pi_s AL_s(e_{ij})}{\sum_{a=1}^{S}\pi_a AL_a(e_{ij})}. \tag{13}$$

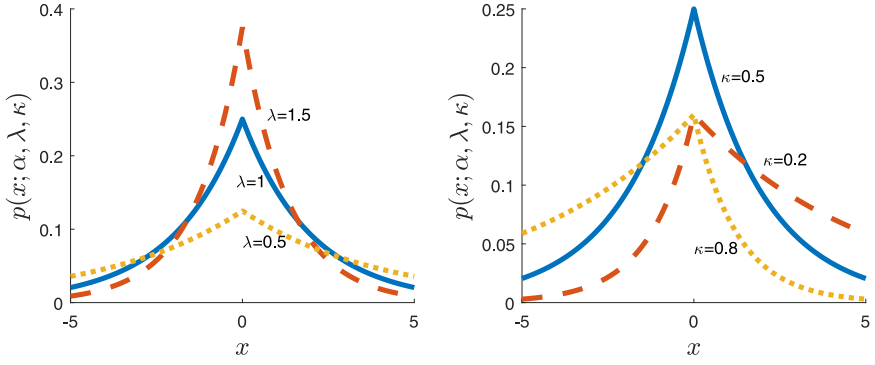


Fig. 2. The PDF curves of ALDs. The location parameter is $\alpha = 0$. Left: $\kappa = 0.5$; right: $\lambda = 1$.

In order to attain the updating rules of other parameters, we need to compute the Q-function. According to the working mechanism of EM algorithm, the Q-function can be obtained by taking expectation of the log-likelihood function shown in (12) with regard to the conditional distribution of the latent variables z_{ij1}, z_{ij2}, $\cdots$, z_{ijS}. Specifically, it can be derived as

$$\begin{aligned}
Q &= E_{\mathbf{Z}|\mathbf{X}}[\ell(\mathbf{U},\mathbf{V},\boldsymbol{\lambda},\mathbf{K},\boldsymbol{\pi})] \\
&= E_{\mathbf{Z}|\mathbf{X}}\{\sum_{(i,j)\in\Omega}\sum_{s=1}^{S} z_{ijs}\big[\log AL_s(e_{ij}|0,\lambda_s,\kappa_s)+\log\pi_s\big]\} \\
&= \sum_{(i,j)\in\Omega}\sum_{s=1}^{S}\gamma_{ijs}\big[\log AL_s(e_{ij}|0,\lambda_s,\kappa_s)+\log\pi_s\big] \\
&= \sum_{(i,j)\in\Omega}\sum_{s=1}^{S}\gamma_{ijs}\big\{\log\pi_s+\log\lambda_s\kappa_s(1-\kappa_s) \\
&\quad -|e_{ij}|\lambda_s\big[(1-\kappa_s)\mathbb{I}(e_{ij}<0)+\kappa_s\mathbb{I}(e_{ij}\geq 0)\big]\big\} \\
&\equiv \sum_{(i,j)\in\Omega}\sum_{s=1}^{S}\gamma_{ijs}\big[\log\kappa_s(1-\kappa_s)\lambda_s\pi_s-\lambda_s\rho_{ijs}|e_{ij}|\big],
\end{aligned} \tag{14}$$

where

$$\rho_{ijs} = \big[(1-\kappa_s)\mathbb{I}(e_{ij}<0)+\kappa_s\mathbb{I}(e_{ij}\geq 0)\big]. \tag{15}$$

M-step: Maximize the Q-function by iteratively updating its parameters as follows.

(1). **Update $\boldsymbol{\pi_s}$**: To attain the update for $\boldsymbol{\pi_s}$, we need to solve the following constrained optimization problem

$$\max_{\pi_s}\sum_{(i,j)\in\Omega}\sum_{s=1}^{S}\gamma_{ijs}\log\pi_s,\quad \text{s.t.}\quad \sum_{s=1}^{S}\pi_s = 1, \tag{16}$$

via the Lagrangian multiplier method. By some derivations, we have

$$\pi_s = \frac{N_s}{N},\quad \text{where}\quad N_s = \sum_{(i,j)\in\Omega}\gamma_{ijs}, \tag{17}$$

in which N stands for the cardinality of Ω.

(2). **Update $\boldsymbol{\lambda_s}$**: Compute the gradient $\frac{\partial Q}{\partial\lambda_s}$ and let it be zero. Consequently, the update of λ_s can be obtained as

$$\lambda_s = \frac{N_s}{\sum_{(i,j)\in\Omega}\rho_{ijs}\gamma_{ijs}|e_{ij}|}. \tag{18}$$

(3). **Update $\boldsymbol{\kappa_s}$**: Compute the gradient $\frac{\partial Q}{\partial\kappa_s}$ and let it be zero, we can have

$$\eta_s\kappa_s^2-(2N_s+\eta_s)\kappa_s+N_s = 0, \tag{19}$$

where the coefficients $\eta_s = \lambda_s\sum_{(i,j)\in\Omega}\gamma_{ijs}e_{ij}$. Evidently, Eq. (19) is a two-order equation with regard to κ_s and it has a unique root satisfying $0<\kappa_s<1$, that is,

$$\kappa_s = \frac{2N_s+\eta_s-\sqrt{4N_s^2+\eta_s^2}}{2\eta_s}. \tag{20}$$

(4). **Update U, V**: By omitting some constants, the objective function to optimize **U, V** can be rewritten as

$$\begin{aligned}
&\max -\sum_{(i,j)\in\Omega}\sum_{s=1}^{S}\lambda_s\gamma_{ijs}\rho_{ijs}|x_{ij}-\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}}| \\
&\Leftrightarrow \min\sum_{i=1}^{m}\sum_{j=1}^{n}w_{ij}|x_{ij}-\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}}| \\
&\Leftrightarrow \min||\mathbf{W}\odot(\mathbf{X}-\mathbf{U}\mathbf{V}^{\mathrm{T}})||_{L_1},
\end{aligned} \tag{21}$$

where the (i,j)th entry of $\mathbf{W}$ is

$$w_{ij} = \begin{cases}\sum_{s=1}^{S}\lambda_s\gamma_{ijs}\rho_{ijs}, & \text{if } (i,j)\in\Omega,\\ 0, & \text{if } (i,j)\notin\Omega.\end{cases} \tag{22}$$

Hence, the optimization problem in Eq. (21) is equivalent to the weighted L_1-LRMF, which can be solved by a fast off-the-shelf algorithm. In this paper, the cyclic weighted median filter (CWM) [26] is employed to solve Eq. (21) and the detailed derivations will be introduced in the next subsection.

Here, it is interesting that the M-step in AQ-LRMF is the same as that of MoG-LRMF [19], except that the latter one minimizes a weighted L_2 loss. Due to this feature, AQ-LRMF is more robust than MoG-LRMF. On the other hand, each weight of MoG-LRMF embodies the information about whether the corresponding entry is an outlier. For each weight of AQ-LRMF, it actually contains additional information about the sign of bias. In particular, λ_s is the scale parameter and the entries with smaller λ_s correspond to outliers. According to the definition of ρ_{ijs} in Eq. (15), we know that ρ_{ijs} is a function of the skewness parameter κ_s. If the residual $e_{ij}\geq 0$, $\rho_{ijs}=\kappa_s$ and $\rho_{ijs}=1-\kappa_s$ otherwise. Hence, the weights assigned to two different points still differ if two residuals with the same absolute value have different signs. In conclusion, AQ-LRMF has more capacity to process heavy-tailed skew data.

Based on the above analysis, we summarize the main steps to learn the parameters involved in AQ-LRMF as shown in Algorithm 1. We now discuss the computational complexity of Algorithm 1. The complexity of updating $\boldsymbol{\gamma}$ is $O(mnS)$ and that of updating $\boldsymbol{\pi}$ and $\boldsymbol{\lambda}$ is the same. As for the complexity to update $\boldsymbol{\kappa}$, it is $O(S)$. At last, the complexity to update **U**, **V** will be $O(mnS)$ if Eq. (21) is solved by CWM. Thus, the total time complexity of Algorithm 1 is $O(T(mnS+S))$, where T is the number of iterations for the algorithm to reach convergence. Note that Algorithm 1 is derived by EM algorithm, it can thus converge to a local optimum

Algorithm 1 Learning algorithm of AQ-LRMF.

Input:
The observed matrix $\mathbf{X}$ of order $m \times n$; the index set Ω of non-missing entries of $\mathbf{X}$; number of components S in MoAL.

Output:
$\mathbf{U}, \mathbf{V}$.

1: Initialize $\mathbf{U}, \mathbf{V}, \boldsymbol{\lambda}, \mathbf{K}, \boldsymbol{\pi}$.
2: (Initial E-step): Evaluate γ_{ijs} by Eq. (13), $i = 1, ..., m$; $j = 1, ..., n$; $s = 1, \cdots, S$.
3: **while** the convergence criterion does not satisfy **do**
4: (M-step 1): Update $\pi_s, \lambda_s, \kappa_s$ $(s = 1, \cdots, S)$ with Eq. (17), (18) and (20), respectively.
5: (E-step 1): Evaluate γ_{ijs} by Eq. (13), $i = 1, ..., m$; $j = 1, ..., n$; $s = 1, \cdots, S$.
6: (M-step 2): Update $\mathbf{U}, \mathbf{V}$ by solving Eq. (21) with the CWM method.
7: (E-step 2): Evaluate γ_{ijs} by Eq. (13), $i = 1, ..., m$; $j = 1, ..., n$; $s = 1, \cdots, S$.
8: (Tune S): For each pair $(i, j) \in \Omega$, compute its noise component index $\mathbf{C}(i, j) = \arg\max_s \gamma_{ijs}$. Remove any ALD components which are not in $\mathbf{C}$. Let S be the current number of ALD components.
9: **end while**

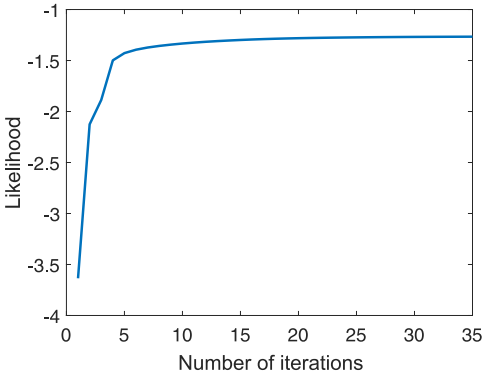


Fig. 3. In a synthetic experiment, how the likelihood value varies as the number of iterations increases.

within finite iterations since the likelihood does not decrease in each step. As an example, Fig. 3 depicts how the likelihood value varies as the number of iteration increases in a synthetic experiment (please see the detailed settings in Section 4.1). It is shown that the likelihood value increases quickly in the first few iterations, and then it gradually levels off. Finally, the algorithm converges at the 35th iteration.

3.5. Solution of the weighted L_1-LRMF

As stated in the last subsection, the learning of AQ-LRMF can be cast into a weighted L_1-LRMF problem. Now we will provide more details about how to solve it (i.e., how to update $\mathbf{U}, \mathbf{V}$ by Eq. (21)) with the CWM method [26].

Essentially, CWM minimizes the objective via solving a series of scalar minimization subproblems. Let $\mathbf{U} = (\tilde{\mathbf{u}}_1, \tilde{\mathbf{u}}_2, \cdots, \tilde{\mathbf{u}}_r) \in \mathbb{R}^{m \times r}$ and $\mathbf{V} = (\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \cdots, \tilde{\mathbf{v}}_r) \in \mathbb{R}^{n \times r}$, respectively. To update v_{ji} $(j = 1, \cdots, n; i = 1, \cdots, r)$, we assume that the other parameters have been estimated. As a result, the original problem can be rewritten as the optimization problem regarding v_{ji}, i.e.,

$$||\mathbf{W} \odot (\mathbf{X} - \mathbf{U}\mathbf{V}^T)||_{L_1} = ||\mathbf{W} \odot (\mathbf{X} - \sum_{j=1}^{r} \tilde{\mathbf{u}}_j \tilde{\mathbf{v}}_j^T)||_{L_1}$$
$$= ||\mathbf{W} \odot (\mathbf{E}_i - \tilde{\mathbf{u}}_i \tilde{\mathbf{v}}_i^T)||_{L_1}$$
$$= ||\tilde{\mathbf{w}}_j \odot (\tilde{\mathbf{e}}_j^i - \tilde{\mathbf{u}}_i v_{ji})||_{L_1} + \mathrm{c}, \quad (23)$$

where $\mathbf{E}_i = \mathbf{X} - \sum_{j \neq i} \tilde{\mathbf{u}}_j \tilde{\mathbf{v}}_j^T$, and $\tilde{\mathbf{w}}_j$ and $\tilde{\mathbf{e}}_j^i$ are jth column of $\mathbf{W}$ and $\mathbf{E}_i$, respectively. In Eq. (23), c denotes a constant term that does not depend on v_{ji}. In this way, the optimal v_{ji}, say v_{ji}^*, can be easily attained by the weighted median filter. Specifically, let $\mathbf{e} = \tilde{\mathbf{w}}_j \odot \tilde{\mathbf{e}}_j^i$ and $\mathbf{u} = \tilde{\mathbf{w}}_j \odot \tilde{\mathbf{u}}_i$, we can reformulate Eq. (23) as

$$||\tilde{\mathbf{w}}_j \odot (\tilde{\mathbf{e}}_j^i - \tilde{\mathbf{u}}_i v_{ji})||_{L_1} = ||\mathbf{e} - \mathbf{u} v_{ji}||_{L_1}$$
$$= \sum_{l=1}^{m} |e_l - u_l v_{ji}| = \sum_{l=1}^{m} |u_l| \cdot |v_{ji} - \frac{e_l}{u_l}|. \quad (24)$$

Hence, the optimal v_{ji}^* can be obtained as

$$v_{ji}^* = \mathrm{argmin}_{v_{ji}} ||\tilde{\mathbf{w}}_j \odot (\tilde{\mathbf{e}}_j^i - \tilde{\mathbf{u}}_i v_{ji})||_{L_1} = \mathrm{argmin}_{v_{ji}} \sum_{l=1}^{m} |u_l| \cdot |v_{ji} - \frac{e_l}{u_l}|. \quad (25)$$

From Eq. (25), it can be seen that v_{ji}^* coincides with the weighted median of the sequence $\{\frac{e_l}{u_l}\}_{l=1}^m$ under weights $\{|u_l|\}_{l=1}^m$. By adopting the similar derivation process, u_{ji}^* $(j = 1, \cdots, m, i = 1, \cdots, r)$, the optimal value for each element u_{ji} of $\mathbf{U}$, can be expressed as

$$u_{ji}^* = \mathrm{argmin}_{u_{ji}} ||\mathbf{w}_j \odot (\mathbf{e}_j^i - \tilde{\mathbf{v}}_i^{\mathrm{T}} u_{ji})||_{L_1}, \quad (26)$$

where $\mathbf{w}_j$ and $\mathbf{e}_j^i$ represent the jth row of $\mathbf{W}$ and $\mathbf{E}_i$, respectively. In short, the optimal $\mathbf{U}, \mathbf{V}$ can be obtained by employing CWM to repeatedly update $v_{ji} (j = 1, \cdots, n; i = 1, \cdots, r)$ and $u_{ji} (j = 1, \cdots, m; i = 1, \cdots, r)$ until the algorithm converges. To facilitate the understanding, the following Algorithm 2 lists the main steps to attain the optimal solution of Eq. (21). Note that Algorithm 2 corresponds to step 6 in Algorithm 1.

Algorithm 2 Solving Eq. (21) by the CWM method.

Input:
The observed matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$; the index set Ω; π_s, λ_s and γ_{ijs} $(i = 1, \cdots, m, j = 1, \cdots, n, s = 1, \cdots, S)$; initial value of $\mathbf{U}, \mathbf{V}$.

Output:
The optimal $\mathbf{U}, \mathbf{V}$.

1: Calculate each element w_{ij} of $\mathbf{W}$ by Eq. (22).
2: **while** the convergence criterion does not satisfy **do**
3: Cyclicly apply the weighted median filter to update each entry v_{ji} $(j = 1, \cdots, n, i = 1, \cdots, r)$ of $\mathbf{V}$ with all the other elements of $\mathbf{U}, \mathbf{V}$ fixed by solving Eq. (25).
4: Cyclicly apply the weighted median filter to update each entry u_{ji} $(j = 1, \cdots, m, i = 1, \cdots, r)$ of $\mathbf{U}$ with all the other elements of $\mathbf{U}, \mathbf{V}$ fixed by solving Eq. (26).
5: **end while**

3.6. Some details of Algorithm 1

Tuning the number of components S in MoAL: Too large S violates Occam Razor's principle, while too small S leads to poor performance. In consequence, as described in step 8 of Algorithm 1, we employ an effective method to tune S. To begin with, we initialize S to be a relatively small number such as 4, 5, ⋯, 8. After each iteration, we compute the cluster that x_{ij} belongs to, by $\mathbf{C}(i, j) = \arg\max_s \gamma_{ijs}$. If there is no entry belonging to cluster s, we remove the corresponding ALD component.

Initialization: In Algorithm 1, the entries in $\mathbf{U}$ and $\mathbf{V}$ can be initialized by using a procedure analogous to that used in [19]. Particularly, the (i, j)th entry u_{ij} of $\mathbf{U}$ was initialized in our experiments

Table 1
The basic information of the used real-world data sets.

Data set	Type of task	Size	subsection
CAVE	image denoising	262144×31	4.3
Extended Yale B	face modeling	32256×64	4.4
Urban	hyperspectral image reconstruction	94249×210	4.5
Terrain	hyperspectral image reconstruction	153500×210	4.5

as $2\xi_{ij}c - c$, where ξ_{ij} denotes a random number sampled from the standard Gaussian distribution $\mathcal{N}(0, 1)$. In addition, $c = \sqrt{\bar{x}/r}$ where $\bar{x}$ is the median of all entries in $\mathbf{X}$ and r indicates the rank of $\mathbf{U}$ and $\mathbf{V}$. Due to the characteristics of $\mathbf{U}$ and $\mathbf{V}$, each entry of $\mathbf{V}$ was initialized similarly. Moreover, the elements in $\boldsymbol{\lambda}, \mathbf{K}$ and $\boldsymbol{\pi}$ was randomly sampled from the uniform distribution on [0,1]. After initializing $\pi_1, \cdots, \pi_S$, they were normalized so that their sum equals to 1.

Convergence condition: By following the common practice of EM algorithm, we terminate the iteration if the change of $||\mathbf{U}||$ is smaller than a pre-defined value or the maximum iteration number is reached.

4. Experimental studies

We carried out experiments in this section to examine the performance of AQ-LRMF model. Several state-of-the-art methods were considered, including four robust LRMF methods (namely, MoG [19],[1] CWM [26], Damped Wiberg (DW)[2] [17], RegL1ALM[3] [28]) and a robust PCA method (SPCP solved by quasi Newton method)[4] [38]. We wrote the programming code for CWM with Matlab. For the other compared algorithms, the codes provided by the corresponding authors were available. Since SPCP does not work in presence of missing entries, it was thus excluded from some experiments which involve missing data. Notice that DW is only considered in Section 4.1 because it meets the "out of memory" problem for large-scale datasets. In the meantime, we assigned the same rank to all the considered algorithms except for SPCP since it can automatically determine the rank. To make the comparison more fair, all algorithms were initialized with the same values. Each algorithm was terminated when either 100 iterative steps are reached or the change of $||\mathbf{U}||$ is less than 1×10^{-50}. In order to simplify notations, our proposed method AQ-LRMF was denoted as AQ in later discussions. All the experiments were conducted with Matlab R2015b and run on a computer with Intel Core CPU 2.30 GHz, 4.00 GB RAM and Windows 7(64-bit) system.

The remainder of this section has the following structure. Section 4.1 studies the performance of each algorithm on synthetic data in the presence of various kinds of noise as well as missing values. Because LRMF has been applied in many fields, we also examined the performance of the compared algorithms on several real-world tasks. Sections 4.2 and 4.3 employ some inpainted and multispectral images to investigate how the compared algorithms behave on real images which contain missing values and various kinds of noise, respectively. Finally, Sections 4.4 and 4.5 examine the performance of all algorithms on face modeling and hyperspectral image processing tasks. Table 1 summarizes the basic information of real-world data sets.

[1] http://www.gr.xjtu.edu.cn/c/document_library/get_file?folderId=1816179&name=DLFE-32163.rar.
[2] http://www.vision.is.tohoku.ac.jp/us/download/.
[3] https://sites.google.com/site/yinqiangzheng/.
[4] https://github.com/stephenbeckr/fastRPCA.

4.1. Synthetic experiments

First, we compared the behavior of each method with synthetic data containing different kinds of noise. Similar to [33], we randomly generated 30 low rank matrices $\mathbf{X} = \mathbf{U}\mathbf{V}^{\mathrm{T}}$ of size 40×20 for each case, where $\mathbf{U} \in \mathbb{R}^{40\times r}$ and $\mathbf{V} \in \mathbb{R}^{20\times r}$ were sampled from the standard Gaussian distribution $\mathcal{N}(0, 1)$. In particular, we considered the situations with $r = 4$ and $r = 8$. In the experiment, we stochastically set 20% entries of $\mathbf{X}$ as missing data and corrupted the non-missing entries with the following three groups of noise, respectively. (i) The first group include 4 kinds of heavy-tailed noise, i.e., $Lap(0,\ 1.5)$ (Laplace noise with scale parameter $b = 1.5$ and location parameter $\mu = 0$), Gaussian noise with $\mu = 0, \sigma = 5$ and Student's t noise with degrees of freedom 1 and 2, respectively. (ii) Two kinds of skew noise are included in the second group, i.e., asymmetric Laplace noise with $\lambda = 1, \kappa = 0.7$ and skew normal noise with $\sigma = 3, \kappa = 0.7$. (iii) Two kinds of mixture noise are included in the last group. The first one is $0.5\mathcal{N}(0, 1) + 0.3Lap(0, 1) + 0.2Lap(0, 2)$ and another one is $0.5\mathcal{N}(0, 1) + 0.3Lap(0, 1) + 0.2AL(0, 1, 0.8)$. It is worthwhile to mention that the two mixture noises simulate the noise contained in real data, where most entries are corrupted by standard Gaussian noise and the rest entries are corrupted by heavy-tailed or skew noise. To evaluate the performance of each method, we employed the average L_1 and L_2 errors which are defined as $\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}|x_{ij} - \mathbf{u}_i\mathbf{v}_j^{\mathrm{T}}|$ and $\sqrt{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}(x_{ij} - \mathbf{u}_i\mathbf{v}_j^{\mathrm{T}})^2}$, respectively.

In our experiments, the value for the parameter r in all algorithms but SPCP was set as the true rank r that was used to generate synthetic data. For each compared algorithm, Tables 2 and 3 summarize the L_1 and L_2 errors averaged over 30 randomly generated matrices when $r = 4$ and $r = 8$, respectively. In the last two rows of Tables 2 and 3, we list the mean and median of the L_1 errors as well as the L_2 errors of all cases. In the situation with $r = 4$, it is quite obvious that our method reaches the minimum L_1 and L_2 errors for each type of noise, while MoG and CWM almost take the second place. And the approaches RegL1ALM and DW can hardly deal with the heavy-tailed and skew noise well. Note that two critical techniques are employed in AQ, that is, asymmetric noise modeling by an MoAL and solving the weighted L_1-norm LRMF by CWM. Based on the superiority of AQ over CWM as demonstrated in Tables 2 and 3, we can conclude that asymmetric noise modeling indeed plays an important role in AQ for it achieving better performance. From the results corresponding to $r = 8$, similar conclusions can be drawn. However, CWM evidently outperforms MoG under this circumstance, which indicates that MoG may be instable when the real rank in observed data is high. In addition, the running speed of AQ is fairly competitive, as shown in Table 4. To compare the algorithms in a clearer manner, we also demonstrate two scattergrams of the L_1 errors versus the running times of each algorithm in Fig. 4. It can be seen that AQ strikes a quite good balance between the reconstruction accuracy and time complexity. Although the L_1 errors of MoG are comparable with those of AQ and CWM, it costs more time. Moreover, CWM is observed to have almost the same time complexity with AQ, but it is outperformed by AQ in terms of reconstruction.

Table 2
The average L_1 and L_2 errors for each algorithm on synthetic data with rank 4. The best and second best results are highlighted in bold and italic typeface, respectively.

$r = 4$	L_1 error					L_2 error				
	AQ	MoG	CWM	RegL1ALM	DW	AQ	MoG	CWM	RegL1ALM	DW
Laplace Noise (b=1.5)	**1.22**	*1.24*	1.38	1.63	1.51	**1.82**	*1.88*	1.92	4.60	4.61
Gaussian Noise ($\sigma = 5$)	**2.97**	3.31	*3.15*	4.62	4.03	**4.66**	5.94	*4.14*	13.29	13.94
Student's t Noise ($df = 1$)	**1.52**	2.53	*1.99*	22.66	598.22	**3.41**	13.51	*4.79*	239.27	11952.61
Student's t Noise ($df = 2$)	**0.98**	1.32	*1.14*	2.18	8.24	**1.56**	3.59	*1.63*	11.38	153.04
AL Noise ($\lambda = 1, \kappa = 0.7$)	**1.93**	2.68	*2.40*	3.76	4.83	**2.90**	6.93	*3.36*	13.18	42.23
SN Noise ($\sigma = 3, \kappa = 0.7$)	**1.89**	*2.02*	2.08	2.62	2.04	**2.65**	3.16	*2.75*	6.41	2.99
Mixture Noise 1	**0.85**	*0.91*	1.00	1.10	1.08	**1.23**	1.47	*1.43*	3.25	3.45
Mixture Noise 2	**0.98**	1.40	*1.24*	3.15	21.23	**1.62**	3.95	*1.94*	18.70	483.78
mean	**1.54**	1.93	*1.80*	5.22	80.15	**2.48**	5.05	*2.74*	38.76	1582.08
median	**1.37**	1.71	*1.68*	2.88	4.43	**2.24**	3.77	*2.35*	12.28	28.08

Table 3
The average L_1 and L_2 errors for each algorithm on synthetic data with rank 8. The best and second best results are highlighted in bold and italic typeface, respectively.

$r = 8$	L_1 error					L_2 error				
	AQ	MoG	CWM	RegL1ALM	DW	AQ	MoG	CWM	RegL1ALM	DW
Laplace Noise (b=1.5)	**1.82**	2.18	*1.92*	3.03	4.98	*2.86*	4.49	**2.81**	8.88	42.57
Gaussian Noise ($\sigma = 5$)	*4.17*	4.86	**4.04**	7.60	10.74	*6.18*	8.26	**5.39**	20.64	82.87
Student's t Noise ($df = 1$)	**2.87**	4.36	*3.41*	15.33	450.20	**8.79**	18.56	*11.87*	77.32	8917.25
Student's t Noise ($df = 2$)	**1.60**	2.24	*1.80*	3.44	22.36	**2.61**	5.93	*2.82*	13.08	397.99
AL Noise ($\lambda = 1, \kappa = 0.7$)	**2.88**	3.88	*3.04*	5.87	16.09	*4.48*	9.28	**4.35**	17.70	227.67
SN Noise ($\sigma = 3, \kappa = 0.7$)	**2.59**	3.00	*2.70*	4.44	5.37	*3.67*	5.00	**3.66**	11.92	35.97
Mixture Noise 1	**1.34**	*1.59*	1.59	2.15	2.98	**2.09**	3.34	*2.40*	6.32	23.29
Mixture Noise 2	**1.69**	2.43	*1.92*	4.30	18.33	**2.88**	5.94	*3.09*	16.19	203.50
mean	**2.37**	3.07	*2.55*	5.77	66.38	**4.19**	7.60	*4.55*	21.51	1241.39
median	**2.21**	2.72	*2.31*	4.37	13.41	**3.27**	5.94	*3.38*	14.64	143.19

Table 4
The running time (in seconds) of each algorithm on synthetic data.

	$r = 4$					$r = 8$				
	AQ	MoG	CWM	RegL1ALM	DW	AQ	MoG	CWM	RegL1ALM	DW
Laplace Noise (b=1.5)	**0.02501**	0.04044	*0.02502*	0.81899	0.07907	**0.07898**	0.13455	*0.07900*	1.33627	0.26134
Gaussian Noise ($\sigma = 5$)	*0.02490*	**0.01330**	0.02490	0.80570	0.12140	*0.07972*	**0.06740**	0.07974	1.35875	0.42626
Student's t Noise ($df = 1$)	**0.03080**	0.12480	*0.03080*	0.86030	0.17470	**0.08268**	0.21673	*0.08268*	1.26710	0.28241
Student's t Noise ($df = 2$)	**0.02560**	0.13690	*0.02560*	0.81970	0.10020	**0.06157**	0.22075	*0.06157*	1.25522	0.27744
AL Noise ($\lambda = 1, \kappa = 0.7$)	*0.01885*	**0.01619**	0.01887	0.84787	0.10406	**0.07640**	0.12882	*0.07640*	1.30657	0.29866
SN Noise ($\sigma = 3, \kappa = 0.7$)	**0.02399**	0.04659	0.02399	0.75145	0.08879	*0.07675*	**0.07408**	0.07675	1.23501	0.25752
Mixture Noise 1	**0.02189**	0.05013	*0.02189*	0.76148	0.07082	**0.07168**	0.15263	*0.07169*	1.21608	0.24865
Mixture Noise 2	**0.02133**	0.07331	*0.02134*	0.81955	0.10586	**0.07214**	0.17796	*0.07214*	1.31874	0.31770
mean	**0.02404**	0.06271	*0.02405*	0.81063	0.10561	**0.07499**	0.14661	*0.07500*	1.28672	0.29625
median	**0.02444**	0.04836	*0.02444*	0.81927	0.10213	**0.07657**	0.14359	*0.07658*	1.28684	0.27993

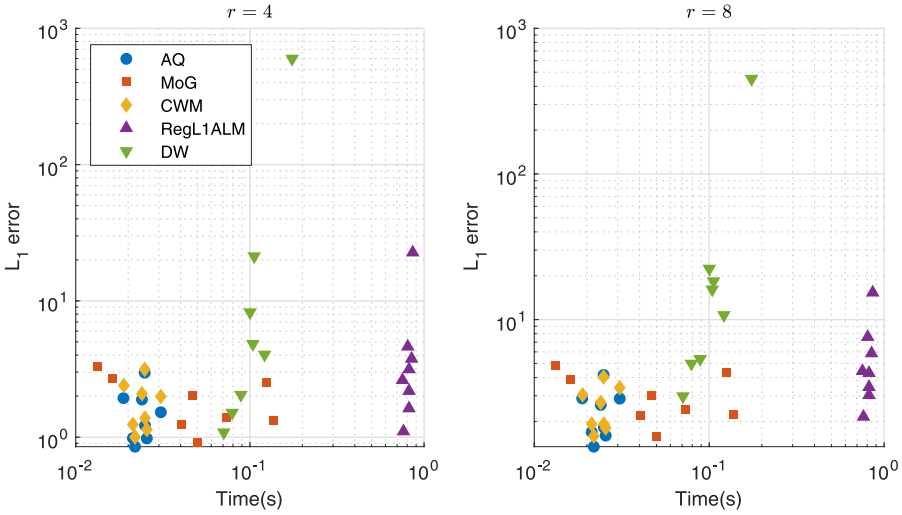


Fig. 4. The scatter gram of the L_1 errors versus the running times of each algorithm on synthetic data.

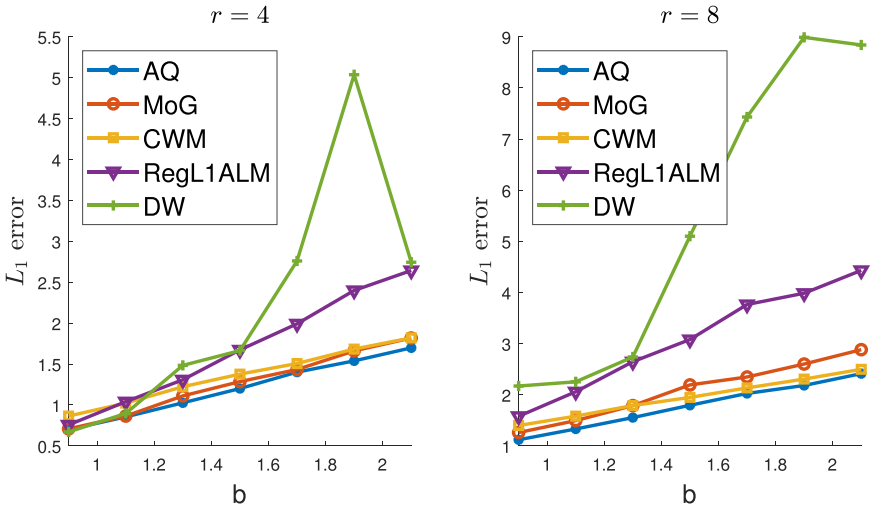


Fig. 5. The performance of each algorithm on synthetic data with different levels of noise.

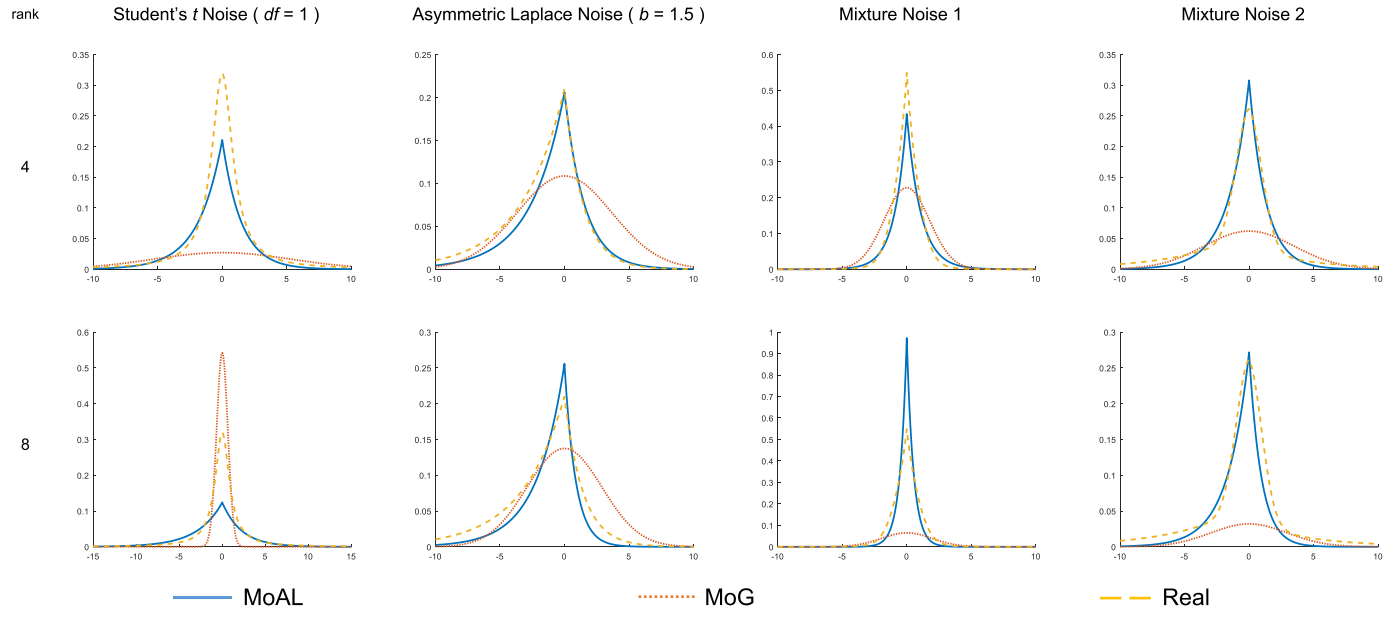


Fig. 6. The comparison of the PDFs for real noise and the ones fitted by AQ and MoG in the synthetic experiments.

Aiming at investigating the behavior of each algorithm more extensively, we also did experiments by varying noise level under a specific type of noise. As an example, we used Laplace noise to generate data with different levels of noise. The scale parameter b was varied from 0.9 to 2.1 with increment 0.2. Under each situation, the experiment was carried out similarly to the previous synthetic experiments. In Fig. 5, the L_1 errors of each algorithm are plotted as a function of the parameter b. It can be observed from Fig. 5 that AQ almost always outperforms the other counterparts at each noise level. In summary, the simulation results presented in this subsection strongly indicate that AQ is a very competitive LRMF tool to cope with the tasks involving different kinds of noise.

To delve into the difference between AQ and MoG, we further compared the distributions of the residuals corresponding to AQ and MoG. Here, the PDFs for real noise were utilized as the baseline. Specifically, two symmetric and two asymmetric cases are illustrated in Fig. 6. Here, the shown PDFs fitted by AQ and MoG correspond to those reach the maximum likelihood over 30 random experiments. It is obvious that AQ does a much better job to approximate the real noise than MoG. Particularly, AQ almost provides a duplicate of real noise. In contrast, MoG is able to fit the tails, while, at the same time, it results in bad approximation to peaks. Hence, AQ has stronger power in fitting complex noise than MoG.

4.2. Image inpainting experiments

Image inpainting is a typical image processing task. In real applications, some parts of an image may be deteriorated so that the corresponding information is lost. To facilitate the understanding of the image, some sophisticated technique need to be adopted to recover its corrupted parts. This is exactly the objective of image inpainting. There is evidence that many images are low-rank matrices so that the single image inpainting can be done by matrix completion [39]. In image inpainting, the corrupted pixels are viewed as missing values and then the image can be recovered by an LRMF algorithm. In this paper, three typical RGB images[5] of

[5] https://sites.google.com/site/zjuyaohu/.

Table 5
The average L_1 and L_2 errors of each method on image inpainting experiments. The best and second best results are highlighted in bold and italic typeface, respectively.

		Image A				Image B				Image C			
		AQ	MoG	CWM	RegL1	AQ	MoG	CWM	RegL1	AQ	MoG	CWM	RegL1
L_1 error	Small	2.59	**2.32**	2.91	*2.33*	**5.60**	7.43	*6.90*	8.08	**5.13**	*5.83*	6.30	6.97
	Large	**5.59**	9.25	*7.84*	8.77	**6.88**	20.16	*8.78*	19.11	**6.84**	*7.15*	7.67	17.70
	Random	*2.81*	**2.18**	3.06	2.45	*5.91*	**5.75**	6.65	6.79	*5.61*	*6.74*	6.77	**5.07**
L_2 error	Small	**5.05**	10.65	*6.62*	17.91	**10.11**	31.13	*14.85*	43.26	**8.67**	*9.64*	10.50	28.29
	Large	**17.95**	49.06	*29.23*	47.04	**13.98**	79.97	*17.95*	75.74	*12.77*	**12.65**	13.93	63.09
	Random	**5.33**	10.97	*6.80*	26.52	**9.86**	16.90	*11.47*	34.06	**9.18**	*10.50*	11.07	11.60

size $300 \times 300 \times 3$ were employed. In our experiments, each image was reshaped to 300×900. By following the common practice in the research of image inpainting, we artificially corrupted the given images by putting some masks onto them. In doing so, it is convenient to examine how well each method performs to restore the original images. Here, three kinds of masks were considered, namely, random mask where 20% pixels were stochastically removed, text masks with big and small fonts, respectively. Some evidence [39] has shown that the information of a single image will be lost if the rank is set to a relatively low value. Thus, the rank was set to 80 in this experiment for all algorithms.

Fig. 7 displays the original, masked and reconstructed images, and Table 5 reports the average L_1 and L_2 errors of each algorithm. It is obvious that removing a random mask is the easiest task. In this situation, there is no significantly visible difference among the reconstructed images. AQ and MoG are the best performers. In contrast, the results shown in Fig. 7 and Table 5 indicate that text mask removal is more difficult, especially when the images are corrupted with big fonts. The main reason lies in that the text mask is spatially correlated while it is difficult for any LRMF algorithm to effectively utilize this type of information. Under these circumstances, it can be observed in Fig. 7 and Table 5 that AQ outperforms the other methods to remove the text masks in terms of both reconstruction error and visualization. RegL1ALM and MoG perform badly and the clear text can often be seen in their reconstructed images. Although CWM produces slightly better results, its average L_1 error is still higher than that of AQ. In a word, AQ possesses the superiority over the other algorithms in our investigated image inpainting tasks. In particular, AQ achieves the smallest average L_1 error in 5 cases and the second smallest one in 3 cases. When evaluating all algorithms with L_2 error, the superiority of AQ is more significant.

4.3. Multispectral image experiments

In this subsection, we study the behavior of all algorithms in image denoising tasks. The Columbia Multispectral Image database, CAVE,[6] was employed, where every scene contains 31 bands with size 512×512. To achieve our purpose, seven scenes out of them (i.e., Balloon, Clay, Feathers, Flowers, Hairs, Paints and Pompoms) were utilized to test the effectiveness of our methods. The used images were resized by half and the pixels were rescaled to [0,1]. Analogous to the strategy used in image inpainting experiments, some noise was artificially added to the original images. Then, each LRMF algorithm was applied to remove the noise so that the corrupted images can be restored as accurate as possible. In the experiments, three different kinds of noise were considered, that is, Laplace noise with scale parameter $b = 10$, asymmetric Laplace noise with $\lambda = 10, \kappa = 0.7$ and mixture noise, i.e., $0.5\mathcal{N}(0, 0.5) + 0.3AL(0, 8, 0.9) + 0.2AL(0, 8, 0.7)$. The rank was set to 4 for all algorithms.

[6] http://www1.cs.columbia.edu/CAVE/databases/multispectral.

Table 6 reports the average L_1 and L_2 errors of each method. Evidently, AQ behaves best in most cases. For Laplace noise, RegL1ALM sometimes outperforms AQ to attain the best results. The success of RegL1ALM can be attributed to the special format of its objective function, namely, L_1 norm loss plus two penalties on **U** and **V**. On the one hand, the L_1 norm loss is exactly compatible with Laplace noise. On the other hand, there is empirical evidence showing that its used penalties can lead to better performance on image datasets. For asymmetric Laplace and mixture noise, it is not surprising that AQ outperforms all the other methods. Under some circumstances, SPCP reaches the second lowest reconstruction error, while the other ones perform badly. The reason for the good behavior of SPCP may be that it does not rely on the assumption of noise distribution, while the other approaches implicitly assume that the noise distribution is not skew.

4.4. Face modeling experiments

Here, we applied the LRMF techniques to address the face modeling task. The Extended Yale B database[7] consisting of 64 images with size 192×168 of each subject was considered. Therefore, it leads to a 32256×64 matrix for each subject. Particularly, we used the face images of the third and fifth subjects. The first column of Fig. 8 demonstrates some typical faces for illustration. We set the rank to 4 for all methods except for SPCP which determines the rank automatically. The second to sixth columns of Fig. 8 display the faces reconstructed by the compared LRMF algorithms.

From Fig. 8, we can observe that that all methods are able to remove the cast shadows, saturations and camera noise. However, the performance of SPCP seems to be worse in comparison with other algorithms. Evidently, AQ always outperforms the other methods due to its pretty reconstruction. As shown in Fig. 1, there is an asymmetric distribution in the face with a large dark region. Because of this, the techniques MoG, CWM, RegL1ALM and SPCP which utilize the symmetric loss function lead to bad results, while AQ with the quantile loss function produces the best reconstructed images.

4.5. Hyperspectral image experiments

In this subsection, we employed two HSI datasets, Urban and Terrain,[8] to investigate the behavior of all algorithms. There are 210 bands, each of which is of size 307×307 for Urban and 500×307 for Terrain. Thus, the data matrix is of size 94249×210 for Urban and 153500×210 for Terrain. Here, we utilized the same experimental settings as those used in Section 4.4. DW was still unavailable in this experiment due to the computational problem. As show in the first column of Fig. 9, some parts of bands are seriously polluted by the atmosphere and water absorption.

[7] http://cvc.yale.edu/projects/yalefaces/yalefaces.html.

[8] http://www.erdc.usace.army.mil/Media/Fact-Sheets/Fact-Sheet-Article-View/Article/610433/hypercube/.

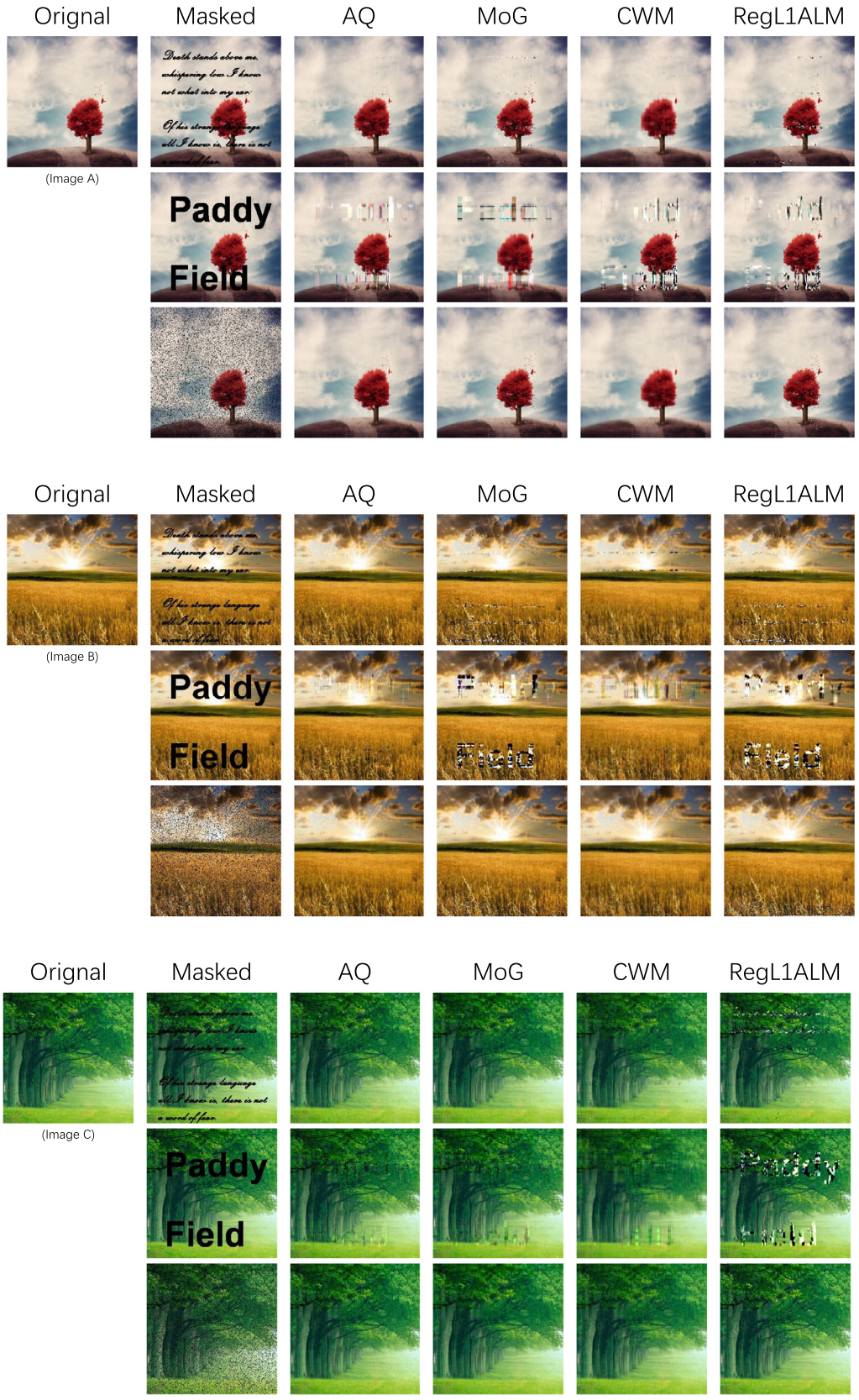


Fig. 7. The original, masked and inpainting images.

Table 6
The average L_1 and L_2 errors of each method on multispectral image experiments. The best and second best results are highlighted in bold and italic typeface, respectively. AL refers to asymmetric Lapalce.

Scene	Type of Noise	L_1 error					L_2 error				
		AQ	MoG	CWM	RegL1ALM	SPCP	AQ	MoG	CWM	RegL1ALM	SPCP
Balloon	Laplace	**0.0074**	0.0348	0.0398	*0.0343*	0.0487	**0.0140**	0.0494	0.0553	*0.0487*	0.0693
	AL	**0.0804**	0.1964	0.1501	0.1379	*0.1280*	**0.1053**	0.2310	0.1882	0.1836	*0.1510*
	Mixture	**0.1974**	0.2514	0.2453	0.2422	*0.2026*	*0.2555*	0.3620	0.3221	0.3879	**0.2374**
Clay	Laplace	**0.0335**	0.0402	0.0362	*0.0344*	0.0596	**0.0454**	0.0518	0.0522	*0.0492*	0.1131
	AL	**0.0778**	0.1787	0.1385	*0.1342*	0.1380	**0.1266**	0.2110	0.1828	0.1830	*0.1789*
	Mixture	**0.1569**	0.2169	0.2403	0.2453	*0.2097*	**0.2491**	0.3372	0.3362	0.3714	*0.2548*
Feathers	Laplace	0.0392	*0.0390*	0.0417	**0.0373**	0.0470	**0.0520**	0.0543	0.0588	*0.0522*	0.0805
	AL	**0.0911**	0.1526	0.1487	0.1396	*0.1303*	**0.1217**	0.1918	0.1876	0.1832	*0.1566*
	Mixture	**0.1946**	0.2575	0.2455	0.2425	*0.2046*	*0.2506*	0.3955	0.3272	0.3911	**0.2414**
Flowers	Laplace	*0.0362*	0.0395	0.0371	**0.0343**	0.0437	**0.0486**	*0.0513*	0.0530	0.0526	0.0794
	AL	**0.0761**	0.1708	0.1450	0.1374	*0.1289*	**0.1113**	0.2054	0.1861	0.1826	*0.1569*
	Mixture	**0.1709**	0.2339	0.2393	0.2437	*0.2025*	*0.2718*	0.3799	0.3307	0.3679	**0.2404**
Hairs	Laplace	0.0321	0.0380	0.0358	*0.0292*	**0.0253**	*0.0426*	0.0514	0.0511	0.0517	**0.0377**
	AL	**0.0681**	0.1969	0.1373	0.1346	*0.1201*	**0.1112**	0.2288	0.1736	0.1825	*0.1370*
	Mixture	**0.1412**	0.2172	0.2305	0.2387	*0.1979*	*0.2595*	0.3684	0.3288	0.3847	**0.2291**
Paints	Laplace	0.0424	*0.0370*	0.0431	**0.0354**	0.0396	0.0554	**0.0497**	0.0648	*0.0501*	0.0636
	AL	**0.1009**	0.1910	0.1432	0.1402	*0.1259*	**0.1329**	0.2244	0.1815	0.1831	*0.1474*
	Mixture	*0.2119*	0.2311	0.2460	0.2399	**0.2018**	*0.3070*	0.3777	0.3433	0.3861	**0.2356**
Pompoms	Laplace	0.0493	0.0420	*0.0414*	**0.0379**	0.0824	0.0641	*0.0542*	0.0548	**0.0527**	0.1211
	AL	**0.1002**	0.1937	0.1508	*0.1422*	0.1472	**0.1311**	0.2268	0.1928	*0.1847*	0.1850
	Mixture	**0.1630**	0.4121	0.2489	0.2323	*0.2153*	**0.2210**	0.5119	0.3281	0.3887	*0.2595*
mean	Laplace	**0.0300**	0.0338	0.0344	*0.0304*	0.0433	**0.0403**	0.0453	0.0488	*0.0447*	0.0706
	AL	**0.0743**	0.1600	0.1267	0.1208	*0.1148*	**0.1050**	0.1899	0.1616	0.1603	*0.1391*
	Mixture	**0.1545**	0.2275	0.2120	0.2106	*0.1793*	**0.2268**	0.3416	0.2895	0.3347	*0.2123*

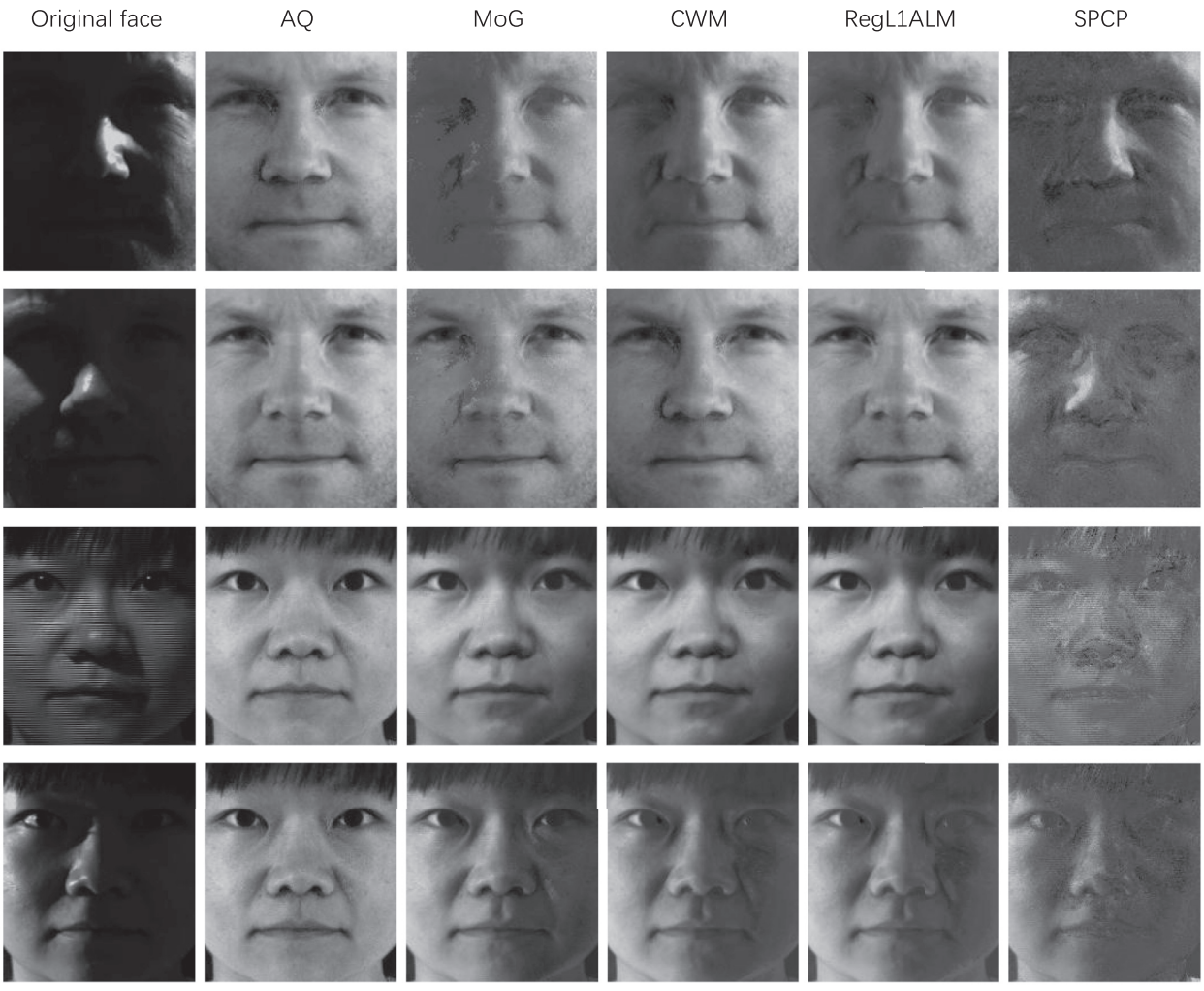


Fig. 8. The original faces and the reconstructed ones.

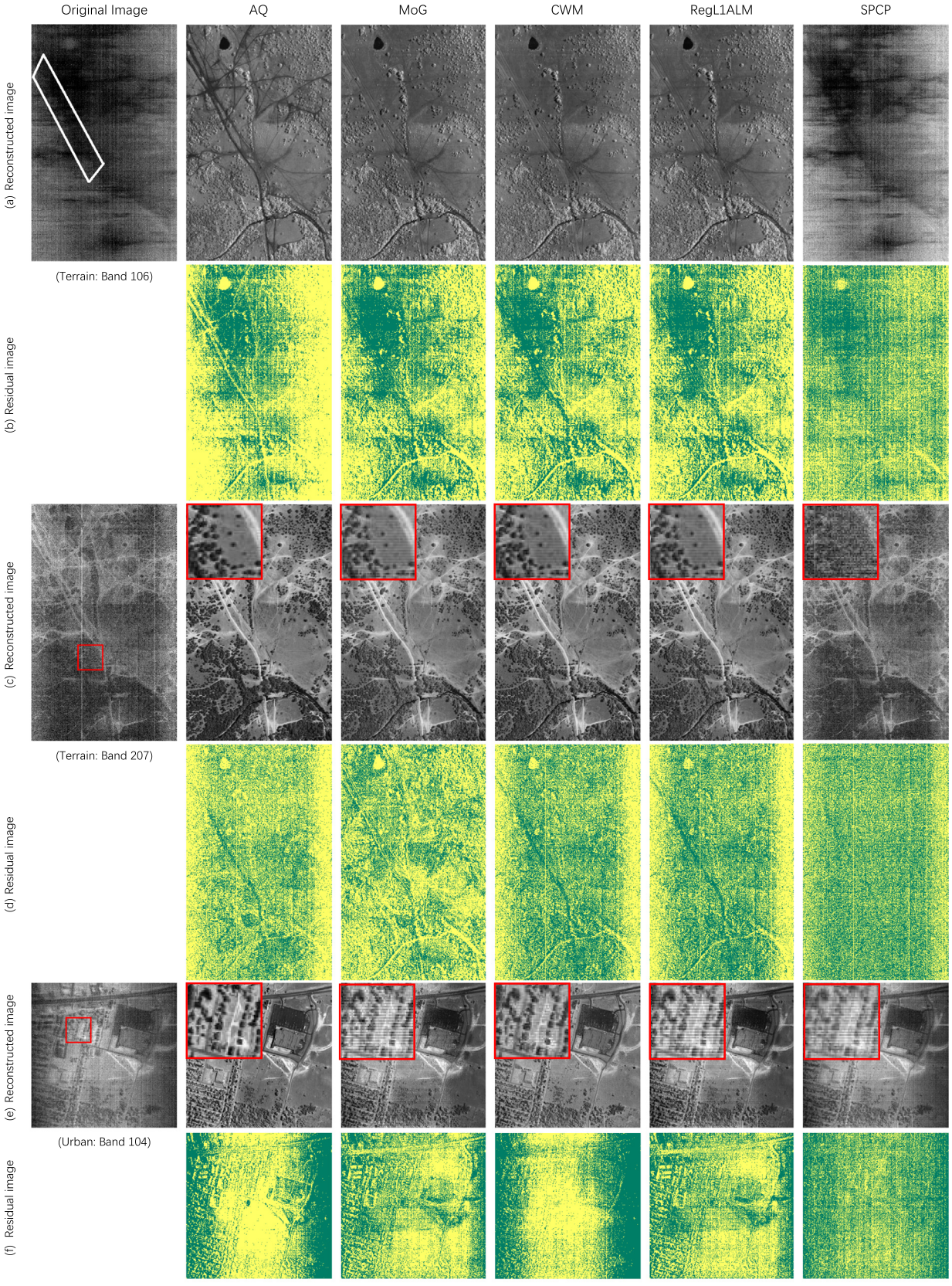


Fig. 9. The reconstructed and residual images. Note that the white frame and red box are markers used to emphasize the local patch.

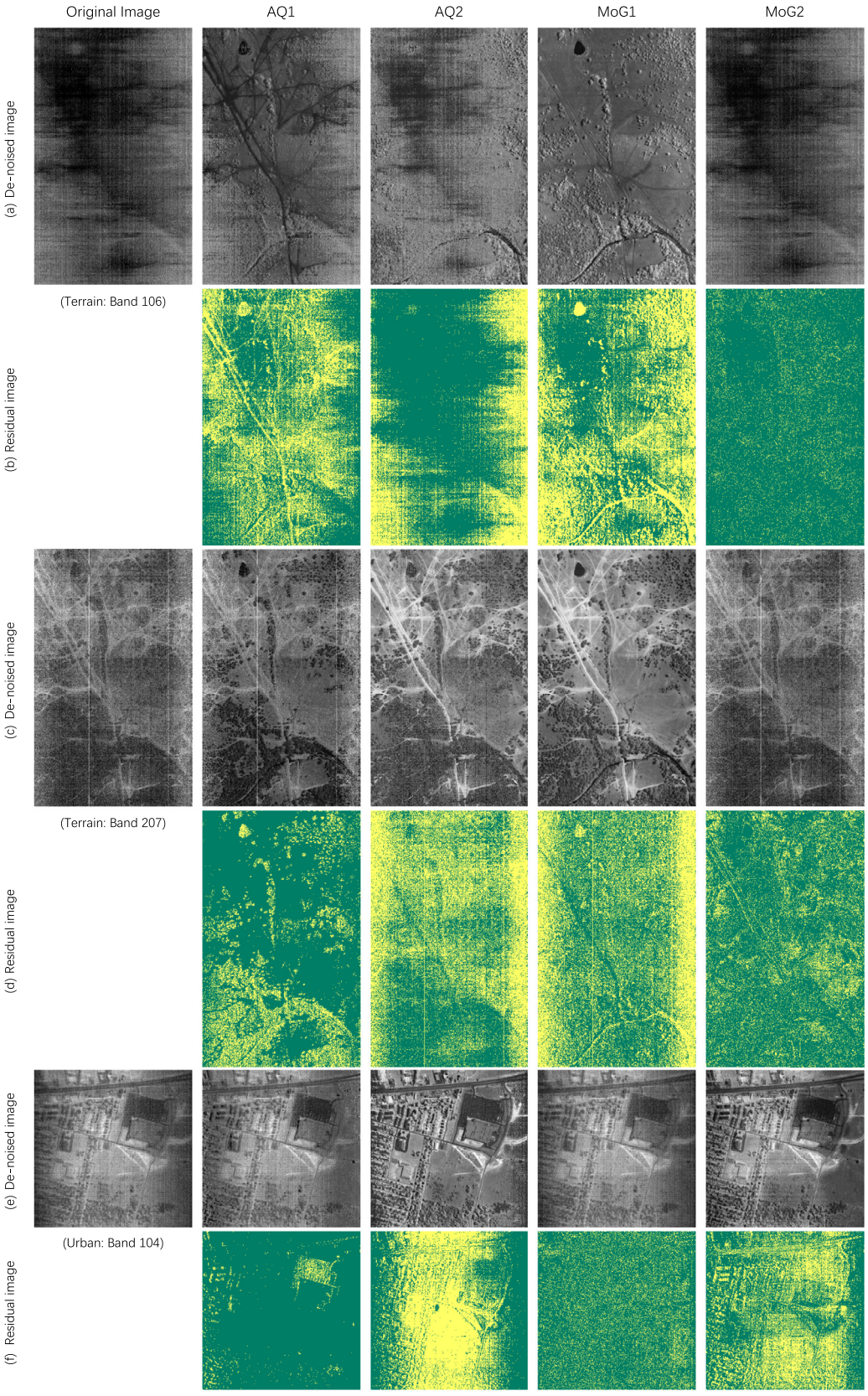


Fig. 10. The de-noised and residual images produced by the two components of AQ (i.e., columns marked with AQ1 and AQ2) and MoG (i.e., columns marked with MoG1 and MoG2). For example, the image lies in the first row and second column is the de-noised image which is obtained by removing the first AQ component from the original image.

The reconstructed images of bands 106 and 207 in the Terrain data set and the band 104 in the Urban data set are shown in Fig. 9 (a), (c) and (e), respectively. Their residual images (i.e., $\mathbf{X} - \hat{\mathbf{U}}\hat{\mathbf{V}}^{\mathrm{T}}$) are also demonstrated below the reconstructed ones. Obviously, the band 106 in Terrain is seriously polluted. Nevertheless, our proposed AQ method still effectively reconstructs a clean and smooth one. Although MoG, CWM and RegL1ALM remove most parts of noise, they miss a part of local information, that is, the line from upper left corner to bottom right hand side (i.e., the white parallelogram marked in the original image). As for SPCP, it only removes few parts of noise. The residual images also reveal that AQ behaves better to deal with the detailed information. Note that the band 207 in Terrain and the band 104 in Urban are mainly corrupted by the stripe and Guassian-like noise. Under these circumstances, AQ still outperforms the others because the latter fails to remove the stripe noise. In particular, for the interested areas that are marked by rectangles and amplified areas, the bands reconstructed by MoG, CWM, RegL1ALM and SPCP contain evident stripes. As far as the reconstructed images produced by AQ are concerned, however, this phenomenon does not exist.

We conjectured that the main reason for the different behavior of these algorithms lies in their used loss function. For CWM, RegL1ALM and SPCP, too simple loss function lead them to work not well when encountering complicated noise. In contrast, AQ and MoG perform better because they use multiple distribution components to model noise. It is very interesting to study the difference between AQ and MoG. For these two algorithms, we found that they both approximate the noise in our considered three bands with two components. For AQ (MoG), we denoted them as AQ1 and AQ2 (MoG1 and MoG2), respectively. In Fig. 10, we presented de-noised images and residual images produced by each component. Take the de-noised image in the column AQ1 as an example, it corresponds to $\hat{\mathbf{U}}\hat{\mathbf{V}}^{\mathrm{T}} + \mathrm{AQ2}$ and the residual image shown below it corresponds to AQ1 (i.e., $\mathbf{X} - \hat{\mathbf{U}}\hat{\mathbf{V}}^{\mathrm{T}} - \mathrm{AQ2}$). The other images can be understood similarly. In doing so, we can further figure out the role that each component in AQ or MoG plays. When dealing with the band 106 in Terrain, the first AQ component is seen to de-noise the center parts, while the second one targets at the left and right edges. For the band 207 in Terrain, two AQ components de-noise the bottom and the rest parts, respectively. Regarding the band 104 in Urban, they focus on the right upper and center parts, respectively. By inspecting the results generated by MoG, however, we cannot discover some regular patterns for the role that two components play. Therefore, it can be concluded that AQ can capture the local structural information of real images, although we do not encode it into our model. The reason may be that the pixels with the same skewness in real images tend to cluster. In this aspect, AQ also possesses superiority over MoG.

5. Conclusions and future work

Aiming at enhancing the performance of existing LRMF methods to cope with complicated noise in real applications, we propose in this work a new low-rank matrix factorization method AQ-LRMF to recover subspaces. The core idea of AQ-LRMF is to directly model unknown noise by a mixture of asymmetric Laplace distributions. We also present an efficient procedure based on the EM algorithm to estimate the parameters in AQ-LRMF. Actually, the objective function of AQ-LRMF corresponds to the adaptive quantile loss like those used in quantile regression. Nevertheless, AQ-LRMF does not need to pre-define the asymmetry parameter of quantile loss whereas quantile regression needs a user to specify its corresponding parameter in advance. Thus, AQ-LRMF has an advantage over quantile regression in this aspect. Based on the experimental results on synthetic and real data, the novel AQ-LRMF model is seen to always outperform several other state-of-the-art counterparts. In addition, AQ-LRMF also has the superiority to capture local structural information in real images. Therefore, AQ-LRMF can be deemed as a competitive tool to cope with complex real problems.

In the future, we believe that the present work can be extended in the following two aspects. The current paper mainly investigates skew noise for images. It will be very interesting to study whether skew noise is also valid for other tasks such as recommender system and link prediction. On the other hand, the idea of skew noise modeling can be extended to related problems, including non-negative LRMF [40] and low-rank tensor factorization [41] by considering the intrinsically high-dimensional multilinear structures of real data.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

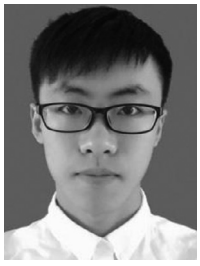
Acknowledgements

The authors would like to thank the editor and the reviewers for their useful suggestions which have helped to improve the paper greatly. This research was supported by the National Key Research and Development Program of China [grant number 2018YFC0809001].

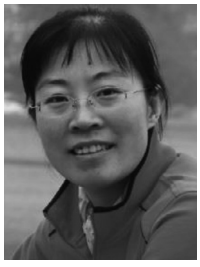
References

[1] J. Ye, Generalized low rank approximations of matrices, Mach. Learn. 61 (1-3) (2005) 167–191.
[2] M. Udell, C. Horn, R. Zadeh, S. Boyd, Generalized low rank models, Found. Trends Mach. Learn. 9 (1) (2016) 1–118.
[3] V. Koltchinskii, K. Lounici, A.B. Tsybakov, Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion, Ann. Stat. 39 (5) (2011) 2302–2329.
[4] D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788–791.
[5] H. Wu, Z. Zhang, K. Yue, B. Zhang, J. He, L. Sun, Dual-regularized matrix factorization with deep neural networks for recommender systems, Knowl.-Based Syst. 145 (2018) 46–58.
[6] D. Tolić, N. Antulov-Fantulin, I. Kopriva, A nonlinear orthogonal non-negative matrix factorization approach to subspace clustering, Pattern Recognit. 82 (2018) 40–55.
[7] H. Xiong, D. Kong, Elastic nonnegative matrix factorization, Pattern Recognit. 90 (2019) 464–475.
[8] C. Deng, Z. Lv, W. Liu, J. Huang, D. Tao, X. Gao, Multi-view matrix decomposition: a new scheme for exploring discriminative information, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, 2015, pp. 3438–3444.
[9] M. Yang, C. Deng, F. Nie, Adaptive-weighting discriminative regression for multi-view classification, Pattern Recognit. 88 (2019) 236–245, doi:10.1016/j.patcog.2018.11.015.
[10] W. Wang, Y. Feng, P. Jiao, W. Yu, Kernel framework based on non-negative matrix factorization for networks reconstruction and link prediction, Knowl.-Based Syst. 137 (2017) 104–114.
[11] J. Xu, C. Deng, X. Gao, D. Shen, H. Huang, Predicting alzheimer's disease cognitive assessment via robust low-rank structured sparse model, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, pp. 3880–3886.
[12] P. Wang, C. Yang, H. Chen, C. Song, X. Zhang, D. Wang, Transcriptomic basis for drought-resistance in brassica napus l., Sci. Rep. 7 (1) (2017) 40532.
[13] X. Fei, Y. Chen, P. Chong, Y. Wang, X. Liu, G. He, Denoising of hyperspectral image using low-rank matrix factorization, IEEE Geosci. Remote Sens. Lett. 14 (7) (2017) 1141–1145.
[14] C. Deng, J. Xu, K. Zhang, D. Tao, X. Gao, X. Li, Similarity constraints-based structured output regression machine: an approach to image super-resolution, IEEE Trans. Neural Netw. Learn. Syst. 27 (12) (2016) 2472–2485, doi:10.1109/TNNLS.2015.2468069.
[15] A.M. Buchanan, A.W. Fitzgibbon, Damped newton algorithms for matrix factorization with missing data, in: IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 2005, pp. 316–322.
[16] P. Chen, Optimization algorithms on subspaces: Revisiting missing data problem in low-rank matrix, Int. J. Comput. Vis. 80 (1) (2008) 125–142, doi:10.1007/s11263-008-0135-7.

[17] T. Okatani, T. Yoshida, K. Deguchi, Efficient algorithm for low-rank matrix factorization with missing components and performance comparison of latest algorithms, in: IEEE International Conference on Computer Vision (ICCV), 2011, pp. 842–849.
[18] Q. Ke, T. Kanade, Robust l_1 norm factorization in the presence of outliers and missing data by alternative convex programming, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005, pp. 739–746.
[19] D. Meng, F.D.L. Torre, Robust matrix factorization with unknown noise, in: IEEE International Conference on Computer Vision (ICCV), 2014, pp. 1337–1344.
[20] V. Maz'Ya, G. Schmidt, On approximate approximations using Gaussian kernels, IMA J. Numer. Anal. 16 (1) (1996) 13–29.
[21] C. Davino, M. Furno, D. Vistocco, Quantile Regression: Theory and Applications, Hoboken: John Wiley & Sons, 2014.
[22] H. Kozumi, G. Kobayashi, Gibbs sampling methods for bayesian quantile regression, J. Stat. Comput. Simul. 81 (11) (2011) 1565–1578.
[23] K. Yu, J. Zhang, A three-parameter asymmetric laplace distribution and its extension, Commun. Stat. - Theory Methods 34 (9-10) (2005) 1867–1879.
[24] N. Srebro, T. Jaakkola, Weighted low-rank approximations, in: International Conference on Machine Learning (ICML), 2003, pp. 720–727.
[25] A. Eriksson, A. van den Hengel, Efficient computation of robust low-rank matrix approximations in the presence of missing data using the l_1 norm, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 771–778.
[26] D. Meng, Z. Xu, L. Zhang, J. Zhao, 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, 2013, pp. 704–710.
[27] E. Kim, M. Lee, C.H. Choi, N. Kwak, S. Oh, Efficient l_1 -norm-based low-rank matrix approximations for large-scale problems using alternating rectified gradient method, IEEE Trans. Neural Netw. Learn. Syst. 26 (2) (2015) 237–251.
[28] Y. Zheng, G. Liu, S. Sugimoto, S. Yan, M. Okutomi, Practical low-rank matrix approximation under robust l1-norm, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 1410–1417.
[29] Z. Lin, C. Xu, H. Zha, Robust matrix factorization by majorization minimization, IEEE Trans. Pattern Anal. Mach. Intell. 40 (1) (2018) 208–220.
[30] S. Li, J. Zhang, X. Guo, Efficient low rank matrix approximation via orthogonality pursuit and l_2 regularization, in: IEEE International Conference on Multimedia and Expo, 2017, pp. 871–876.
[31] B. Lakshminarayanan, G. Bouchard, C. Archambeau, Robust bayesian matrix factorisation, J. Mach. Learn. Res. 15 (2011) 425–433.
[32] N. Wang, T. Yao, J. Wang, D.-Y. Yeung, A probabilistic approach to robust matrix factorization, in: European Conference on Computer Vision (ECCV), Springer, 2012, pp. 126–139.
[33] X. Cao, Y. Chen, Q. Zhao, D. Meng, Low-rank matrix factorization under general mixture noise distributions, IEEE Trans. Image Process. 25 (10) (2016) 4677–4690.
[34] E. Candès, X. Li, Y. Ma, J. Wright, Robust principal component analysis? J. ACM 58 (3) (2011) 11.
[35] Z. Zhou, X. Li, J. Wright, E. Candès, Stable principal component pursuit, in: IEEE International Symposium on Information Theory Proceedings, Austin, TX, USA, 2010, pp. 1518–1522.
[36] A. Azzalini, A.D. Valle, The multivariate skew-normal distribution, Biometrika 83 (4) (1996) 715–726.
[37] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm, J. Royal Stat. Soc. Ser. B (Methodol.) 39 (1) (1977) 1–38.
[38] A. Aravkin, S. Becker, V. Cevher, P. Olsen, A variational approach to stable principal component pursuit, in: Conference on Uncertainty in Artificial Intelligence (UAI), Arlington, Virginia, United States, 2914, pp. 32–41.
[39] Y. Hu, D. Zhang, J. Ye, X. Li, X. He, Fast and accurate matrix completion via truncated nuclear norm regularization, IEEE Trans. Pattern Anal. Mach. Intell. 35 (9) (2013) 2117–2130.
[40] P. He, X. Xu, J. Ding, B. Fan, Low-rank nonnegative matrix factorization on stiefel manifold, Inf. Sci. 514 (2020) 131–148, doi:10.1016/j.ins.2019.12.004.
[41] X. Chen, Z. Han, Y. Wang, Q. Zhao, D. Meng, L. Lin, Y. Tang, A generalized model for robust tensor factorization with noise modeling by mixture of gaussians, IEEE Trans. Neural Netw. Learn.Syst. 29 (11) (2018) 5380–5393, doi:10.1109/TNNLS.2018.2796606.



Shuang Xu is currently pursuing the Ph.D. degree in statistics with the School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, China. His current research interests include Bayesian statistics, deep learning and complex network.



Chunxia Zhang is a professor in School of Mathematics and Statistics at Xi'an Jiaotong University. She has authored and coauthored about 40 journal papers on ensemble learning techniques, high-dimensional data analysis and etc. Her main interests are in the area of ensemble learning, variable selection and deep learning.

Jiangshe Zhang is a Professor in School of Mathematics and Statistics at Xi'an Jiaotong University. He has authored and co-authored one monograph and over 80 conference and journal papers on robust clustering, dimension reduction, deep learning and etc. His current research interests include Bayesian statistics, global optimization and deep learning.