Pattern Recognition 63 (2017) 15–29

Contents lists available at ScienceDirect

Pattern Recognition

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

Sequential dimensionality reduction for extracting localized features

Gabriella Casalino [a,*], Nicolas Gillis [b]

[a] Department of Informatics, University of Bari "A. Moro", Via E. Orabona 4, 70125 Bari, Italy
[b] Department of Mathematics and Operational Research, Université de Mons, Rue de Houdain 9, 7000 Mons, Belgium

ARTICLE INFO

Article history:
Received 1 June 2015
Received in revised form
4 July 2016
Accepted 10 September 2016
Available online 19 September 2016

Keywords:
Nonnegative matrix factorization
Underapproximation
Sparsity
Hyperspectral imaging
Dimensionality reduction
Spatial information

ABSTRACT

Linear dimensionality reduction techniques are powerful tools for image analysis as they allow the identification of important features in a data set. In particular, nonnegative matrix factorization (NMF) has become very popular as it is able to extract sparse, localized and easily interpretable features by imposing an additive combination of nonnegative basis elements. Nonnegative matrix underapproximation (NMU) is a closely related technique that has the advantage to identify features sequentially. In this paper, we propose a variant of NMU that is particularly well suited for image analysis as it incorporates the spatial information, that is, it takes into account the fact that neighboring pixels are more likely to be contained in the same features, and favors the extraction of localized features by looking for sparse basis elements. We show that our new approach competes favorably with comparable state-of-the-art techniques on synthetic, facial and hyperspectral image data sets.

1. Introduction

Linear dimensionality reduction (LDR) techniques are powerful tools for the representation and analysis of high dimensional data. The most well-known and widely used LDR is principal component analysis (PCA) [14]. When dealing with nonnegative data, it is sometimes crucial to take into account the nonnegativity in the decomposition to be able to interpret the LDR meaningfully. For this reason, nonnegative matrix factorization (NMF) was introduced and has been shown to be very useful in several applications such as document classification, air emission control and microarray data analysis; see, e.g., [7] and the references therein. Given a nonnegative input data matrix $M \in \mathbb{R}_+^{n\times m}$ and a factorization rank r, NMF looks for two matrices $U \in \mathbb{R}_+^{n\times r}$ and $V \in \mathbb{R}_+^{r\times m}$ such that $M \approx UV$. Hence each row $M(i,:)$ of the input matrix M is approximated via a linear combination of the rows of V: for $1 \le i \le n$,

$$M(i,:) \approx \sum_{k=1}^{r} U_{ik} V(k,:).$$

In other words, the rows of V form an approximate basis for the rows of M, and the weights needed to reconstruct each row of M are given by the entries of the corresponding row of U. The advantage of NMF over PCA (that does not impose nonnegativity constraints on the factors U and V) is that the basis elements V can be interpreted in the same way as the data (e.g., as vectors of pixel intensities; see Section 3 for some illustrations) while the nonnegativity of the weights in U make them easily interpretable as activation coefficients. In this paper, we focus on imaging applications and, in particular, on blind hyperspectral unmixing which we describe in the next section.

1.1. Nonnegative matrix factorization for hyperspectral images

A hyperspectral image (HSI) is a three dimensional data cube providing the electromagnetic reflectance of a scene at varying wavelengths measured by hyperspectral remote sensors. Reflectance varies with wavelength for most materials because energy at certain wavelengths is scattered or absorbed to different degrees, this is referred to as the spectral signature of a material; see, e.g., [21]. Some materials will reflect the light at certain wavelengths, others will absorb it at the same wavelengths. This property of hyperspectral images is used to uniquely identify the constitutive materials in a scene, referred to as endmembers, and classify pixels according to the endmembers they contain. A hyperspectral data cube can be represented by a two dimensional pixel-by-wavelength matrix $M \in \mathbb{R}_+^{n\times m}$. The columns $M(:,j) \in \mathbb{R}_+^n$ of M $(1 \le j \le m)$ are original images that have been converted into n-dimensional column vectors (stacking the columns of the image matrix into a single vector), while the rows $M(i,:) \in \mathbb{R}_+^m$ of M $(1 \le i \le n)$ are the spectral signatures of the pixels (see Fig. 1). Each

* Corresponding author.
E-mail addresses: gabriella.casalino@uniba.it (G. Casalino), nicolas.gillis@umons.ac.be (N. Gillis).

http://dx.doi.org/10.1016/j.patcog.2016.09.006
0031-3203/

$$\underbrace{M(i,:)}_{\text{spectral signature of } i\text{th pixel}} \approx \sum_{k=1}^{r} \underbrace{U_{ik}}_{\text{abundance of } k\text{th endmember in } i\text{th pixel}} \underbrace{V(k,:)}_{\text{spectral signature of } k\text{th endmember}}$$

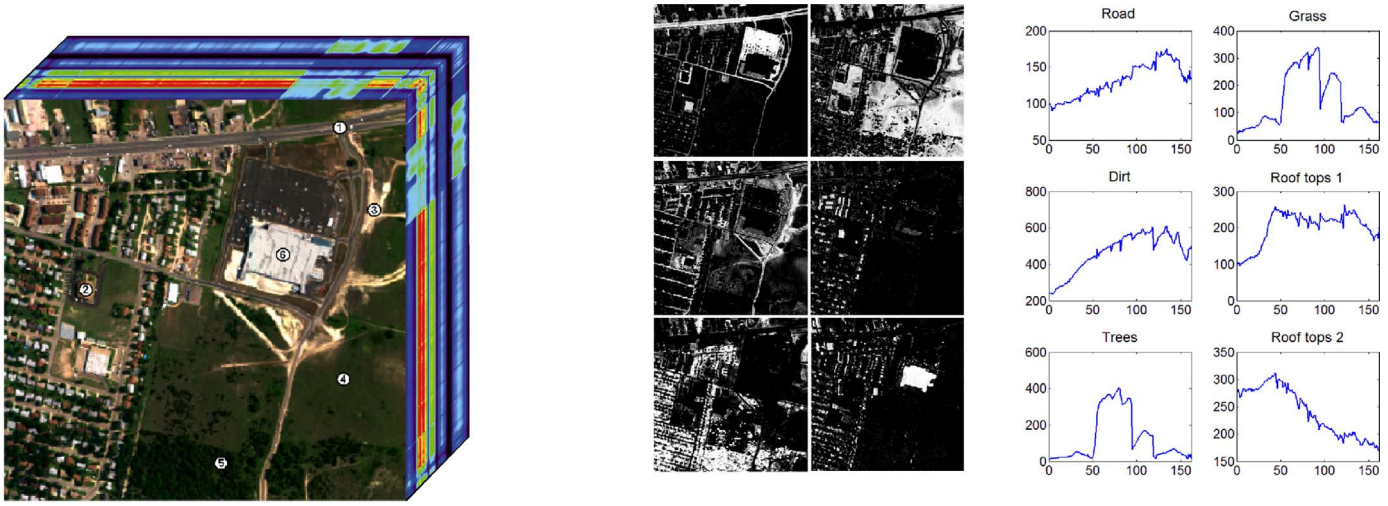


Fig. 1. Decomposition of the Urban hyperspectral image from http://www.agc.army.mil/, constituted mainly of six endmembers ($r=6$): road, grass, dirt, two kind of roof tops and trees. Each row of the matrix V is the spectral signature of an endmember, while each row of the matrix U is the abundance map of the corresponding endmember, that is, it contains the abundance of all pixels for that endmember.

entry M_{ij} represents the reflectance of the i-th pixel at the j-th wavelength. Under the linear mixing model, the spectral signature of each pixel results from the additive linear combination of the nonnegative spectral signatures of the endmembers it contains. In that case, NMF allows us to model hyperspectral images because of the nonnegativity of the spectral signatures and the abundances: Given a hyperspectral data cube represented by a two dimensional matrix $M \in \mathbb{R}^{n\times m}_+$, NMF approximates it with the product of two factor matrices $U \in \mathbb{R}^{n\times r}_+$ and $V \in \mathbb{R}^{r\times m}_+$ such that the spectral signature of each pixel (a row of matrix M) is approximated by the additive linear combination of the spectral signatures of the endmembers (rows of matrix V), weighted by coefficients U_{ik} representing the abundance of the k-th endmember in the i-th pixel. For all i, we have:

$$M(i,:) \approx \sum_{k=1}^{r} U_{ik} V(k,:), \tag{1.1}$$

where r is the number of endmembers in the image. The matrix U is called the abundance matrix while the matrix V is the endmember matrix. Fig. 1 illustrates this decomposition on the urban hyperspectral data cube.

Unfortunately, as opposed to PCA, NMF is a difficult problem (NP-hard) [22]. Moreover, the decomposition is in general non-unique and has to be recomputed from scratch when the factorization rank r is modified. For these reasons, a variant of NMF, referred to as nonnegative matrix underapproximation, was recently proposed that allows us to compute factors sequentially; it is presented in the next section.

1.2. Nonnegative matrix underapproximation

Nonnegative matrix underapproximation (NMU) [8] was introduced in order to solve NMF sequentially, that is, to compute one rank-one factor $U(:,k)V(k,:)$ at a time: first compute $U(:,1)V(1,:)$, then $U(:,2)V(2,:)$, etc. In other words, NMU tries to identify sparse and localized features sequentially. In order to keep the nonnegativity in the sequential decomposition, it is natural to use the following upper bound constraint for each rank-one factor of the decomposition: for all $1 \le k \le r$,

$$U(:,k)V(k,:) \le M - \sum_{p<k} U(:,p)V(p,:).$$

Hence, given a data matrix $M \in \mathbb{R}^{n\times m}_+$, NMU solves, at the first step, the following optimization problem:

$$\min_{u\in\mathbb{R}^n_+, v\in\mathbb{R}^m_+} \left\|M - uv^{\mathrm{T}}\right\|_F^2 \quad \text{such that} \quad uv^{\mathrm{T}} \le M,$$

referred to as rank-one NMU. Then, the nonnegative residual matrix $R = M - uv^{\mathrm{T}} \ge 0$ is computed, and the same procedure can be applied on the residual matrix R. After r steps, NMU provides a rank-r NMF of the data matrix M. Compared to NMF, NMU has the following advantages:

1. As PCA, the solution is unique (under some mild assumptions) [10].
2. As PCA, the solution can be computed sequentially, and hence the factorization rank does not need to be chosen a priori.
3. As NMF, it leads to a separation by parts. Moreover the additional underapproximation constraints enhance this property leading to better decompositions into parts [8] (see also Section 3 for some numerical experiments).

NMU was used successfully for example in hyperspectral [10] and medical imaging [16,17], and for document classification [3].

1.3. Outline and contribution of the paper

Modifications of the original NMU algorithm were made by adding prior information into the model, as it is also often done with NMF; see, e.g., [4] and the references therein. More precisely, two variants of NMU have been proposed:

1. One adding sparsity constraints on the abundance matrix, dubbed sparse NMU [11].

2. One adding spatial information about pixels, dubbed spatial NMU [12].

In this paper, we include both sparsity constraints and spatial information in NMU. This allows us to extract localized features in images more effectively. We present our algorithm in Section 2 and show in Section 3 that it competes favorably with comparable state-of-the-art techniques (NMF, sparse NMF and several variants of NMU) on synthetic, hyperspectral and facial image data sets.

2. Algorithm for NMU with priors

In this section, we describe our proposed technique that will incorporate both spatial and sparsity priors into the NMU model. This will allow us to extract more localized and more coherent features in images.

2.1. Reformulation of NMU and Lagrangian-based algorithm

First, let us briefly describe the original NMU algorithm from [8]. The original rank-one NMU problem is as follows:

$$\min_{u\in\mathbb{R}^n_+, v\in\mathbb{R}^m_+} \left\|M - uv^T\right\|_F^2 \quad \text{such that } uv^T \le M. \tag{2.1}$$

As described in the introduction, solving this problem allows us to compute NMF sequentially, that is, one rank-one factor at a time while preserving nonnegativity. In [8], approximate solutions for rank-one NMU (2.1) are obtained using the Lagrangian dual

$$\max_{\Lambda\ge 0}\min_{u\ge 0, v\ge 0} L(u,v,\Lambda) = \max_{\Lambda\ge 0}\min_{u\ge 0, v\ge 0} \left\|M - uv^T\right\|_F^2 + 2\sum_{i,j}\left(uv^T - M\right)_{ij}\Lambda_{ij}, \tag{2.2}$$

where $\Lambda \in \mathbb{R}^{n\times m}$ is the matrix containing the Lagrangian multipliers of the underapproximation constraints. The authors prove that for a fixed Λ, the problem $\min_{u\ge 0, v\ge 0} L(u,v,\Lambda)$, called Lagrangian relaxation of (2.1), is equivalent to $\min_{u\ge 0, v\ge 0} \left\|(M-\Lambda) - uv^T\right\|_F^2$ which is equivalent, up to a scaling of the variables, to

$$\max_{u\ge 0, v\ge 0} u^T(M-\Lambda)v \quad \text{such that } \|u\|_2 \le 1, \ \|v\|_2 \le 1. \tag{2.3}$$

In fact, one can show that any optimal solution of (2.2) is, up to a scaling factor, an optimal solution of (2.3), and vice versa. It has to be noted that, except for the trivial stationary point (that is, $u=0$ and $v=0$, which is optimal only for $M-\Lambda \le 0$), any stationary point of (2.3) will satisfy $\|u\|_2 = \|v\|_2 = 1$ (this can be shown by contradiction). Therefore, without loss of generality, one can assume $\|u\|_2 = 1$ and $\|v\|_2 = 1$.

Based on these observations, the original NMU algorithm optimizes alternatively over the variables u and v, and updates the Lagrangian multipliers accordingly. The advantage of this scheme is that it is relatively simple as the optimal solution of u given v can be written in closed form (and vice versa). The original NMU algorithm iterates between the following steps:

- $u \leftarrow \max(0, (M-\Lambda)v)$, $u \leftarrow u/\|u\|_2$,
- $v \leftarrow \max(0, (M-\Lambda)^T u)$, $v \leftarrow v/\|v\|_2$,
- $\Lambda \leftarrow \max(0, \Lambda + \mu(\sigma uv^T - M))$ where $\sigma = u^T(M-\Lambda)v$ and μ is a parameter.

Note that given u and v, $\sigma = u^T(M-\Lambda)v$ is the optimal rescaling of the rank-one factor uv^T to approximate $M-\Lambda$. The parameter μ can for example be equal to $\frac{1}{t}$ where t is the iteration index to guarantee convergence [8]. Note also that the original NMU algorithm shares similarities with the power method used to compute the best rank-one unconstrained approximation; see the discussion in [10].

2.2. Spatial information

The first information that we incorporate in NMU is that neighboring pixels are more likely to be contained in the same features. The addition of spatial information into NMU improves the decomposition of images by generating spatially coherent features (e.g., it is in general not desirable that features contain isolated pixels). In this paper, we use the anisotropic total variation (TV) regularization; see, e.g., [13]. We define the neighborhood of a pixel as its four adjacent pixels (above, below, left and right). To evaluate the spatial coherence, we use the following function:

$$\sum_{i=1}^{n}\sum_{j\in\mathcal{N}(i)} \left|u_i - u_j\right| = 2\,\|Nu\|_1, \tag{2.4}$$

where $\mathcal{N}(i)$ is the set of neighboring pixels of pixel i, and $N \in \mathbb{R}^{K\times n}$ is a neighbor matrix where each column corresponds to a pixel and each row indicates a pair (i,j) of neighboring pixels:

$$N(k,i) = 1 \quad \text{and} \quad N(k,j) = -1, \tag{2.5}$$

with $1 \le i < j \le m$ and $j \in \mathcal{N}(i)$, and K is the number of neighboring pairs ($K \le 4n$, because each pixel has at most 4 neighbors). The term $\|Nu\|_1$ therefore accounts for the distances between each pixel and its neighbors. Note that the ℓ_1 norm is used here because it is able to preserve the edges in the images, as opposed to the ℓ_2 norm which would smooth them out; see, e.g., [23,13]. The same penalty was used in [12] to incorporate spatial information into NMU.

2.3. Sparsity information

The second information incorporated in the proposed algorithm is that each feature should contain a relatively small number of pixels. In hyperspectral imaging, this translates into the fact that each pixel usually contains only a few endmembers: the row vectors of the abundance matrix U should have only few non-zero elements. In [11], the authors demonstrated that incorporating this sparsity prior into NMU leads to better decompositions. They added a regularization term to the objective function in (2.1) based on the ℓ_1-norm heuristic approach, in order to minimize the non-zero entries of u: $\|u\|_1$ where $\|u\|_2 = 1$. This is the most standard approach to incorporate sparsity and was also used to design sparse NMF algorithms [15].

2.4. Algorithm for NMU with spatial and sparsity constraints

In order to inject information in the factorization process, we add the regularization terms for the sparsity and spatial information in the NMU formulation (2.3):

$$\max_{\|u\|_2\le 1, \|v\|_2=1, u\ge 0, v\ge 0} \left(\underbrace{u^T\left(M-\Lambda\right)v - \varphi\,\|u\|_1 - \mu\,\|Nu\|_1}_{=f(u,v)} \right). \tag{2.6}$$

The objective function is composed of three terms: the first one relates to the classical least squares residual, the second enhances sparsity of the abundance vector u while the third improves its spatial coherence. The regularization parameters μ

and φ are used to balance the influence of the three terms. We will refer to (2.6) as prior NMU (PNMU). Note that we relax the constraint $\| u \|_2 = 1$ from NMU to $\| u \|_2 \leq 1$ to have a convex optimization problem in u; see below for more details.

Algorithm 1 formally describes the alternating scheme to solve the NMU problem with sparsity and spatial constraints (2.6).

Algorithm 1. Prior NMU: incorporating spatial information and sparsity into NMU.

Require: $M \in \mathbb{R}_+^{n\times m}$, $r \in \mathbb{N}_+$, $0 \leq \varphi' \leq 1$, $0 \leq \mu' \leq 1$, small parameter $\epsilon > 0$, maxiter, iter.

Ensure: $(U, V) \in \mathbb{R}_+^{n\times r} \times \mathbb{R}_+^{r\times m}$ s.t. $UV \leq M$ with U containing sparseness and locality information.

1: Generate the matrix N according to (2.5);
2: **for** $k = 1$: r **do**
3: $[x, y, \Lambda]$= rank-one underapproximation(M); % Initialization of (x, y) with an approximate solution to NMU (2.1)
4: $u_k \leftarrow x$; $v_k \leftarrow y$; $x \leftarrow \frac{x}{\| x \|_2}$; $y \leftarrow \frac{y}{\| y \|_2}$;
5: $w_i = \left(|Nx|_i + \epsilon\right)^{-0.5}$; $W = \mathrm{diag}(w)$; % Initialization of IRWLS weights
6: $\varphi = \varphi' \left\|(M - \Lambda)y\right\|_\infty$ % Setting the sparsity parameter φ
7: $x = \max\left(0, \left(x - \varphi\right)\right)$; $x = \frac{x}{\| x \|_2}$;
8: $z = \mathrm{rand}(m, 1)$; % Estimate of the eigenvector of B associated with the largest eigenvalue
9: **for** $t = 1$: maxiter **do**
10: $A = M - \Lambda$;
11: % Update of x
12: $B = \left(WN\right)^{\mathrm{T}}\left(WN\right)$;
13: for l=1: iter $z = Bz$; $z = \frac{z}{\| z \|_2}$; %Power method
14: **for** $l = 1$: iter **do**
15: $\mu = \mu' \frac{\| Ay \|_\infty}{\| Bx \|_\infty}$; % Setting of the spatial parameter μ
16: $L = \max\left(\varepsilon, \mu\left(z^{\mathrm{T}}Bz\right)\right)$ % Approximated Lipschitz constant
17: $\nabla f(x) = Ay - \mu Bx - \varphi$;
18: $x \leftarrow \mathcal{P}\left(Lx + \nabla f(x)\right)$;
19: **end for**
20: % Update of y
21: $y \leftarrow \max\left(0, A^{\mathrm{T}}x\right)$;
22: if $\|y\|_2 \neq 0$ then $y \leftarrow \frac{y}{\| y \|_2}$;
23: %Update of Λ and save (x, y)
24: **if** $x \neq 0$ and $y \neq 0$ **then**
25: $\sigma = x^{\mathrm{T}}Ay$; $u_k \leftarrow x$; $v_k \leftarrow \sigma y$;
26: $\Lambda \leftarrow \max\left(0, \Lambda - \frac{1}{t+1}\left(M - u_k v_k^{\mathrm{T}}\right)\right)$;
27: **else**
28: $\Lambda \leftarrow \frac{\Lambda}{2}$;
29: $x \leftarrow \frac{u_k}{\| u_k \|_2}$; $y \leftarrow \frac{v_k}{\| v_k \|_2}$;
30: **end if**
31: % Update of the weights
32: $w_i = \left(|Nx|_i + \epsilon\right)^{-0.5}$; $W = \mathrm{diag}(w)$;
33: **end for**
34: $M = \max\left(0, M - u_k v_k^{\mathrm{T}}\right)$;
35: **end for**

As in the original NMU model, the optimal solution for v given u can still be written in closed form. However, because of the spatial information (the term $\| Nu \|_1$), there is no closed form for the optimal solution of u given v, although the problem is convex in u. In order to find an approximate solution to that subproblem, we combine iterative reweighted least squares with a standard projected gradient scheme from [20], as proposed in [12]; see below for more details. Finally, a simple block-coordinate descent scheme is used to find good solutions to problem (2.6). This is achieved by applying the following alternating scheme that optimizes one block of variables while keeping the other fixed:

1. $v \leftarrow \max\left(\left(M - \Lambda\right)^{\mathrm{T}}u, 0\right)$, $v \leftarrow v/\| v \|_2$ (lines 21 and 22).
2. $u \leftarrow \mathcal{P}\left(u - \frac{1}{L}\nabla f_w(u)\right)$ (from line 12 to line 19) where L is the Lipschitz constant of $\nabla f_w\left(u\right)$, where f_w is a smooth approximation of f; see the paragraph below and Eq. (2.8). The projection is defined as

$$\mathcal{P}(s) = \begin{cases} \dfrac{\max(0, s)}{\left\|\max(0, s)\right\|_2} & \text{if } \left\|\max(0, s)\right\|_2 \geq 1, \\ \max(0, s) & \text{otherwise.} \end{cases}$$

3. $\Lambda \leftarrow \max\left(0, \Lambda + \mu\left(\sigma uv^{\mathrm{T}} - M\right)\right)$ with $\sigma = u^{\mathrm{T}}Mv$ (from line 24 to line 30).

Variables (u, v, Λ) are initialized with a solution of the original NMU algorithm from [11] (line 3).

Smooth approximation of f. The variables v and Λ are updated as in the original NMU algorithm, whilst the update of u is modified in order to take into account the penalty terms. The update of u involves the non-differentiable ℓ_1-norm term $\| Nu \|_1$ in the objective function f. Note that, since $u \geq 0$, $\| u \|_1 = \sum_{i=1}^m u_i$ hence it is differentiable on the feasible set. The authors in [12] suggested to use iteratively re-weighted least squares (IRWLS) to approximate the ℓ_1-norm. At the t-th iteration, the term $\|Nu\|_1$ is replaced with

$$\|Nu\|_1 \approx u^{\mathrm{T}}\left(\underbrace{N^{\mathrm{T}}W^{(t)\mathrm{T}}W^{(t)}N}_{=B}\right)u, \tag{2.7}$$

where $W^{(t)} = \mathrm{diag}\left(w^{(t)}\right)$, $\mathrm{diag}(x)$ takes a vector x as an input and returns a diagonal matrix with the entries of x on the diagonal, and $w_i^{(t)} = \left(\left|Nu^{(t-1)}\right|_i + \varepsilon\right)^{-\frac{1}{2}}$ (lines 12 and 32). The idea is to approximate the ℓ_1-norm of a vector $z \in \mathbb{R}^n$ as a weighted ℓ_2-norm:

$$\| z \|_1 \approx \sum_{i=1}^n w_i z_i^2, \quad \text{with } w_i = \frac{1}{|z_i| + \epsilon},$$

where ϵ is a small constant (we used 10^{-3}). In our case, the weights are estimated using the value of u at the previous iteration. We denote f_w the modified objective function that approximates f at the current iterate. Hence the non-differentiable term $\| Nu \|_1$ is approximated with a convex and quadratic differentiable term $u^T Bu$, and we can use a standard gradient descent scheme from smooth convex optimization [20] (line 17); the gradient being

$$\nabla f_w\left(u\right) = \left(M - \Lambda\right)v - \varphi e - \mu\left(Bu\right), \tag{2.8}$$

where e is the vector of all ones of appropriate dimension. The Lipschitz constant L of ∇f_w is equal to the largest singular value of B. Since the matrix B is rather large (but sparse), it is computational costly to compute exactly its largest eigenvalue, hence we use several steps of the power method (line 13) to estimate the Lipschitz constant L.

Choice of the penalty parameters φ and μ. It is in general difficult to choose a priori 'optimal' values for the penalty parameters φ (sparsity parameter) and μ (local information). In fact, it is difficult to know the sparsity and spatial coherence of the unknown localized features. Moreover, it is difficult to relate them to the parameters φ and μ. Note that this choice also depends on the scaling of the input matrix: if M is multiplied by a constant, the first term in (2.6) is increased by the same constant.

In this paper, we use scaling-independent parameters: we replace φ and μ with (lines 15 and 6):

$$\mu = \mu' \frac{\|(M-\Lambda)v\|_\infty}{\|Bu\|_\infty} \quad \text{and} \quad \varphi = \varphi' \left\|(M-\Lambda)v\right\|_\infty,$$

for some $\mu' \in [0, 1]$ and $\varphi' \in [0, 1]$. These choices were proposed respectively in [12,11]. They have the advantage to give a range of possible values for the penalty parameters:

- $\varphi' = 1$ is the highest possible sparsity level: it would imply that the largest value of x is set to zero by the thresholding operator in the projection $\mathcal{P}$.
- $\mu' = 1$ would give more importance to the spatial coherence and will generate extremely smooth features (see next section for some numerical experiments).

Computational cost. The computational cost of Algorithm 1 is the same as the other NMU variants: it requires $O(mn)$ operations to compute each rank-one factor, for a total of $O(mnr)$ operations. Most of the cost resides in the matrix–vector products (Av and A^Tu for the updates of u and v) and the update of the Lagrangian multipliers Λ. Hence, the cost is linear in m, n and r, as for most NMF algorithms (in particular the one described in the next section).

Convergence. The convergence analysis of Algorithm 1 is non-trivial, because it combines several strategies that are themselves difficult to analyze. In fact, our algorithm is based on a Lagrangian relaxation (whose subproblems are solved using a two-block coordinate descent method) where the Lagrangian variables Λ are updated to guarantee the convergence of the scheme [2] (see also the discussion in [8]). This is combined with a reweighted least squares approach to approximate the non-differentiable ℓ_1 norm $\| Nu \|_1$, which is also guaranteed to converge [5]. In practice, we have observed that Algorithm 1 converges usually within 500 iterations (but this depends in particular on the size of the data set); see Section 3.1 for some numerical experiments.

3. Experimental results

In this section, we conduct several experiments to show the effectiveness of PNMU.

In the first part, we use synthetic data sets for which we know the ground truth hence allowing us to quantify very precisely the quality of the solutions obtained by the different algorithms.

In the second part, we validate the good performance of PNMU on real data sets, and compare the algorithms on two widely used data sets, namely, the Cuprite hyperspectral image and the CBCL face data set.

The following algorithms will be compared:

- *Nonnegative matrix factorization (NMF).* We use the accelerated HALS algorithm (A-HALS) [9] (with parameters $\alpha = 0.5$ and $\varepsilon = 0.1$, as suggested by the authors).
- *Sparse NMF (SNMF).* It is a sparse version of A-HALS [6] that adds an ℓ_1-norm sparsity-enhancing term for the abundance matrix U. We run SNMF with target sparsity for the matrix U given by the sparsity of the abundance matrix U obtained by PNMU. Hence we can compare PNMU with SNMF meaningfully as they will have comparable sparsity levels, as it was done in [8] to compare NMU with SNMF.
- *Nonnegative matrix underapproximation (NMU).* This is PNMU with $\varphi' = 0$ and $\mu' = 0$.
- *Nonnegative matrix underapproximation with local information (LNMU).* This is PNMU with $\varphi' = 0$.
- *Sparse nonnegative matrix underapproximation (SNMU).* This is PNMU with $\mu' = 0$.
- *Nonnegative matrix underapproximation with prior information (PNMU).* This is Algorithm 1.

All the numerical results are obtained by implementing the algorithms in Matlab 7.8 codes and running them on a machine equipped with an Intel® *Xeron*® CPU E5420 Dual Core 250 GHz, RAM 8.00 GB. The code is available online from https://sites.google.com/site/nicolasgillis/code. We use maxiter=500, iter=10 for all experiments in the paper.

3.1. Synthetic data sets

Let us construct synthetic hyperspectral images. We use $r=4$ materials perfectly separated. Materials are adjacent rectangles of size $10 \times (k+1)$ pixels with $k = 1, 2, 3, 4$, so that the image at each wavelength contains 10×14 pixels ($m=140$); see the top row of Fig. 5. Mathematically, for $1 \le i \le 140$ and $1 \le k \le r$,

$$U(i,k) = \begin{cases} 1 & \text{if } 5(k-1)(k+2)+1 \le i \le 5(k-1)(k+2)+10(k+1), \\ 0 & \text{otherwise.} \end{cases}$$

We use $n=20$ wavelengths. The spectral signatures are sinusoids with different phases plus a constant to make them nonnegative:

$$V(j,k) = 1.1 + \sin\big(x(j) + \phi(k)\big) \ge 0,$$

where $x(j) = j\frac{2\pi}{n}$ with $j = 1, 2, \ldots, n$, and $\phi(k) = (k-1)\frac{2\pi}{r}$ with $k = 1, 2, \ldots, r$. We also permute the rows of V so that adjacent materials have less similar spectral signatures (we used the permutation [1 3 2 4]). Fig. 2 shows the spectral signatures.

For the noise, we combine Gaussian and salt-and-pepper noise. Let us denote $\bar{M} = 1.1$ the average of the entries of UV (which is the noiseless synthetic HSI):

- Gaussian. We use

  $$G(i,j) \sim g\bar{M}\,\mathcal{N}(0,1), \quad 1 \le i \le n, \quad 1 \le j \le m,$$

  where g is the intensity of the noise. We use the function `randn(n,m)` of Matlab.
- Salt–pepper. For the salt-and-pepper noise, we use a sparse matrix P of density p whose non-zero entries are distributed

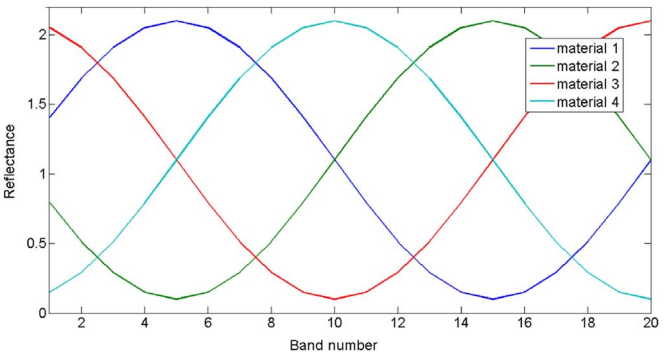


Fig. 2. Synthetic spectral signatures.

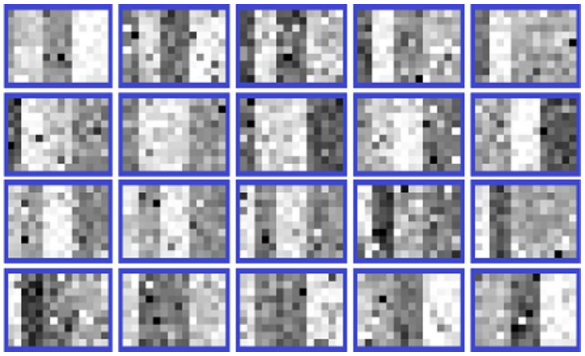


Fig. 3. Example of a synthetic HSI with $g = 30\%$ and $p = 15\%$. Each image corresponds to a wavelength.

following $\bar{M}\mathcal{N}(0, 1)$. We use the function `sprandn(n,m,p)` of Matlab.

Finally, we generate the noisy hyperspectral image $M = UV + G + P$; see Fig. 3 for an example. Given a solution $(\tilde{U}, \tilde{V})$ generated by an algorithm using matrix M, we will compare the quality of the r extracted materials as follows. First, we normalize the columns of $\tilde{U}$ so that the maximal entries are equal to one, that is, $\tilde{U}(:, k) \leftarrow \frac{\tilde{U}(:,k)}{\max_i \tilde{U}(i,k)}$ for all k. Then, we permute the columns of $\tilde{U}$ in order to minimize

$$\text{match}(U, \tilde{U}) = \frac{1}{mr} \sum_{k=1}^{r} \left\| U(:, k) - \tilde{U}(:, k) \right\|_2^2 \in [0, 1]. \tag{3.1}$$

The match will be equal to zero if $\tilde{U}$ coincides with U (up to a permutation of the columns), and equal to 1 when $U(i, k) = 0 \iff \tilde{U}(i, k) = 1$ for all i, k (perfect mismatch).

Note that for our particular U, the match value of the zero matrix is equal to $\text{match}(U, 0) = 25\%$ (which is the density of U) hence we should not expect values higher than that in our numerical experiments. As we will see, match values higher than 2% correspond to a relatively poor recovery of the matrix U; see Fig. 5.

3.1.1. Choice of ϕ' and μ'

First, let us observe the influence of ϕ' and μ' on PNMU and choose reasonable values for these parameters for these synthetic data sets. We fix the noise level to $g=0.2$ and $p=0.05$, and vary ϕ' and μ'. Fig. 4 shows the average match of PNMU over 20 randomly generated synthetic data sets for different values of ϕ' and μ'.

We observe that values of $\phi' \in [0.6, 0.9]$ and $\mu' \in [0.1, 0.5]$ provide good values for the match (below 1%, which is better than

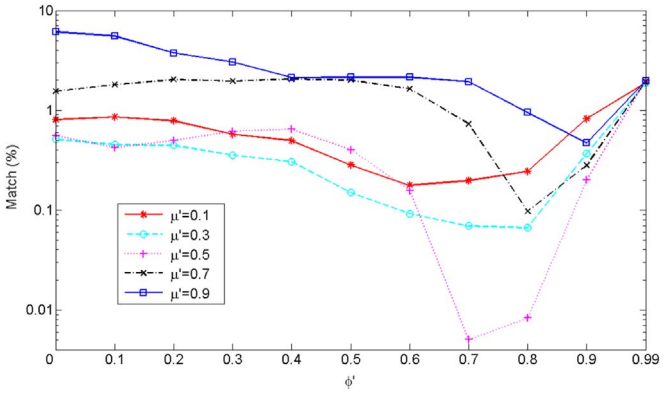


Fig. 4. Average value of the match (3.1) of PNMU over 20 randomly generated synthetic HSI with $g=0.2$ and $p=0.05$ for PNMU for different values of ϕ' and μ'.

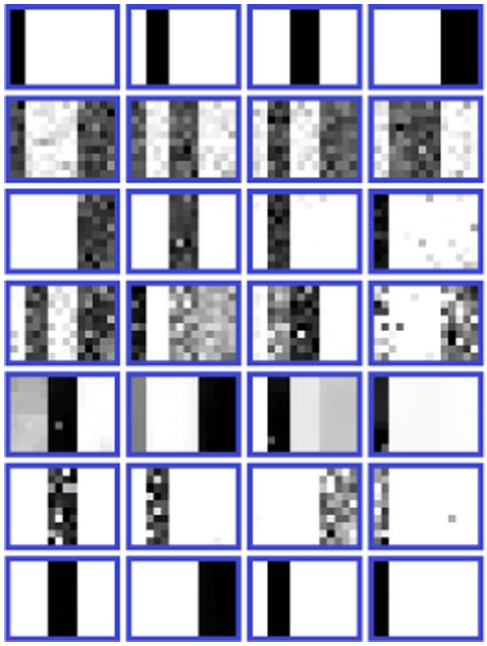


Fig. 5. Basis elements extracted by the different algorithms for the synthetic HSI with $g=0.3$ and $p=0.15$ of Fig. 3. From top to bottom: True materials, NMF, SNMF with sparsity 0.75, NMU, LNMU, SNMU and PNMU with $\phi' = 0.8$ and $\mu' = 0.5$. The match values are: (NMF) 14.34%, (SNMF) 1.64%, (NMU) 13.17%, (LNMU) 2.12%, (SNMU) 7.26%, and (PNMU) 0.003%.

the other NMU and NMF algorithms; see Fig. 5). For the comprehensive comparison of the different algorithms in the next section, we will use $\phi' = 0.7$ and $\mu' = 0.5$ for LNMU, SNMU and PNMU.

Fig. 5 shows the materials extracted by NMU, NMF, SNMF (with sparsity 0.75), LNMU, SNMU and PNMU with $\phi' = 0.7$ and $\mu' = 0.5$ for the noise level $g=0.3$ and $p=0.15$. We observe that PNMU identifies perfectly the four materials. SNMU performs relatively well but returns the basis elements with salt-and-pepper noise. LNMU returns spatially more coherent basis elements but they are mixture of several materials (hence not sparse enough). NMF and SNMF both return spatially less coherent basis elements, and SNMF returns more localized (sparser) ones identifying relatively well the materials (except for the smallest one).

Remark 1. The code to generate these synthetic HSI, and compare the different algorithms is available from https://sites.google.com/site/nicolasgillis/. Moreover, one can easily change the noise level, the number and the size of the materials, and the number of wavelengths. The values $g=0.3$ and $p=0.15$ were chosen because they correspond to a high noise level where PNMU still performs perfectly (see Fig. 7c). As we will see below, PNMU outperforms the other approaches for most reasonable values of the noise level.

Regarding the convergence of PNMU, although it is difficult to analyze rigorously (cf. the discussion in the previous section), Fig. 6 displays the difference between iterates for this particular numerical experiment, showing the relative fast convergence in this case.

3.1.2. Comparison

Let us vary the noise level in three different ways:

- We fix $p = 5\%$ and we vary $g = 0, 0.05, 0.1, \ldots, 1$. Fig. 7a shows the average value of the match in percent for the different algorithms. We observe that for all values of g smaller than 55%, PNMU outperforms the other approaches, being able to

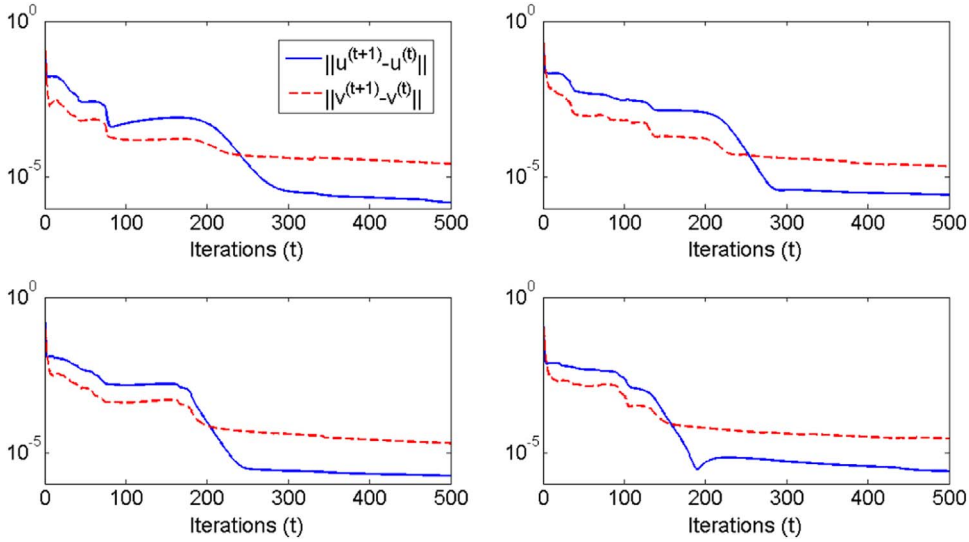


Fig. 6. Convergence for each rank-one factor generated by PNMU on the synthetic HSI with $g=0.3$ and $p=0.15$ from Fig. 3. Each plot displays the evolution of $\| u^{(t+1)} - u^{(t)} \|_2$ and $\| v^{(t+1)} - v^{(t)} \|_2$ where $(u^{(t)}, v^{(t)})$ is the tth iterate generated by PNMU for each rank-one factor uv^T computed sequentially (from left to right, top to bottom).

generate basis elements with match much smaller than 0.5% (in average, 0.12%). For higher noise levels, the performance of PNMU degrades rapidly. SNMF performs relatively well although the match is always higher than 0.7%. For high noise levels, it performs better than PNMU although for these values of the noise, all basis elements generated are relatively poor. The other approaches perform rather poorly, with most of the match values higher than 5%.

- We fix $g = 10\%$ and we use $p = 0, 0.01, \ldots, 1$. Fig. 7b shows the average value of the match in percent for the different algorithms. PNMU performs best up to $p = 35\%$ (which is rather high as 35% of the entries of M are highly perturbed) with match smaller than 0.15% for all $p \leq 0.1$, and average match of 0.22% for $0.1 \leq p \leq 0.2$. It is followed by SNMF while the other approaches perform relatively poorly.
- We use $g = 0.02q$ and $p = 0.01q$ with $q = 0, 1, 2, \ldots, 50$. Fig. 7c shows the average value of the match in percent for the different algorithms. Up to $q=20$, PNMU performs best with average match values smaller than 1%. For $q \geq 23$ which is rather high ($g = 46\%$ and $p = 23\%$), PNMU deteriorates rapidly but, although SNMF has the lowest match values, they are relatively high and correspond to poor basis elements (and we see that LNMU has similar match values for these high noise levels).

In all cases, we observe that PNMU outperforms the other NMF and NMU variants where the noise regime is reasonable. In particular, it performs very well (match $<1\%$) for values up to $g = 0.1$, $p = 0.3$ (Fig. 7b), $g = 0.55$, $p = 0.05$ (Fig. 7c) and $g = 0.4$, $p = 0.2$ (Fig. 7a), which is not the case for the other algorithms.

3.2. Real-world data sets

In this section, we first illustrate the sensitivity of PNMU with respect to the parameters μ' and φ' on the Cuprite hyperspectral image. Not surprisingly, as we have already observed in the previous section, these parameters play a critical role.

However, as opposed to classical NMU that is completely unsupervised, NMU with prior information may require human supervision for tuning its parameters and choosing a good trade-off between the reconstruction accuracy, and the spatial coherence and sparsity of the features.

Then, a quantitative analysis is conducted. Because the ground truth is not known for these data sets, we use three measures: the relative error in percent

$$\text{Relative Error} = 100\frac{\|M - UV^T\|_F}{\|M\|_F},$$

the sparsity (percentage of nonzero entries) of the factors U

$$s(U) = 100\frac{\#\text{zeros}(U)}{mr} \in [0, 100] \quad \text{for} \quad U \in \mathbb{R}^{m\times r},$$

and the spatial coherence of the basis elements

$$\ell(U) = \sum_{k=1}^{r} \frac{\| NU(:,k)\|_1}{\| U(:,k)\|_2}.$$

Recall that $\| NU(:,k)\|_1$ amounts for the spatial coherence of the kth column of the abundance matrix U, while we normalize the columns of U to have a fair comparison (since the algorithms do not normalize the columns of U in the same way). The aim of our experiments is to compare the algorithms in terms of the trade-off between these three measures.

$\| NU(:,k)\|_1$

Adding constraint to the factorization process leads to an increase of the approximation error, but the constrained variants return sparser and/or spatially more coherent solutions. As already noted in [8], it is not fair to compare directly the approximation error of NMU with NMF because NMU computes a solution sequentially. In order to compare the quality of the generated sparsity patterns, one can postprocess the solutions obtained by all algorithms by optimizing over the non-zero entries of U and V (A-HALS can be easily adapted to handle this situation). We will refer to *"Improved"* as the relative approximation after this post-processing step.

Two sets of experimentation are conducted. The aim of the first experiments is to show the behavior of the proposed method, and to test its effectiveness in correctly detecting the endmembers in a widely used hyperspectral image, the Cuprite image, and reducing noise in the abundance maps. The second experiment shows the effectiveness of PNMU in correctly detecting the parts of facial images on one of the most widely used data set in the NMF literature, namely the MIT-CBCL face data set.

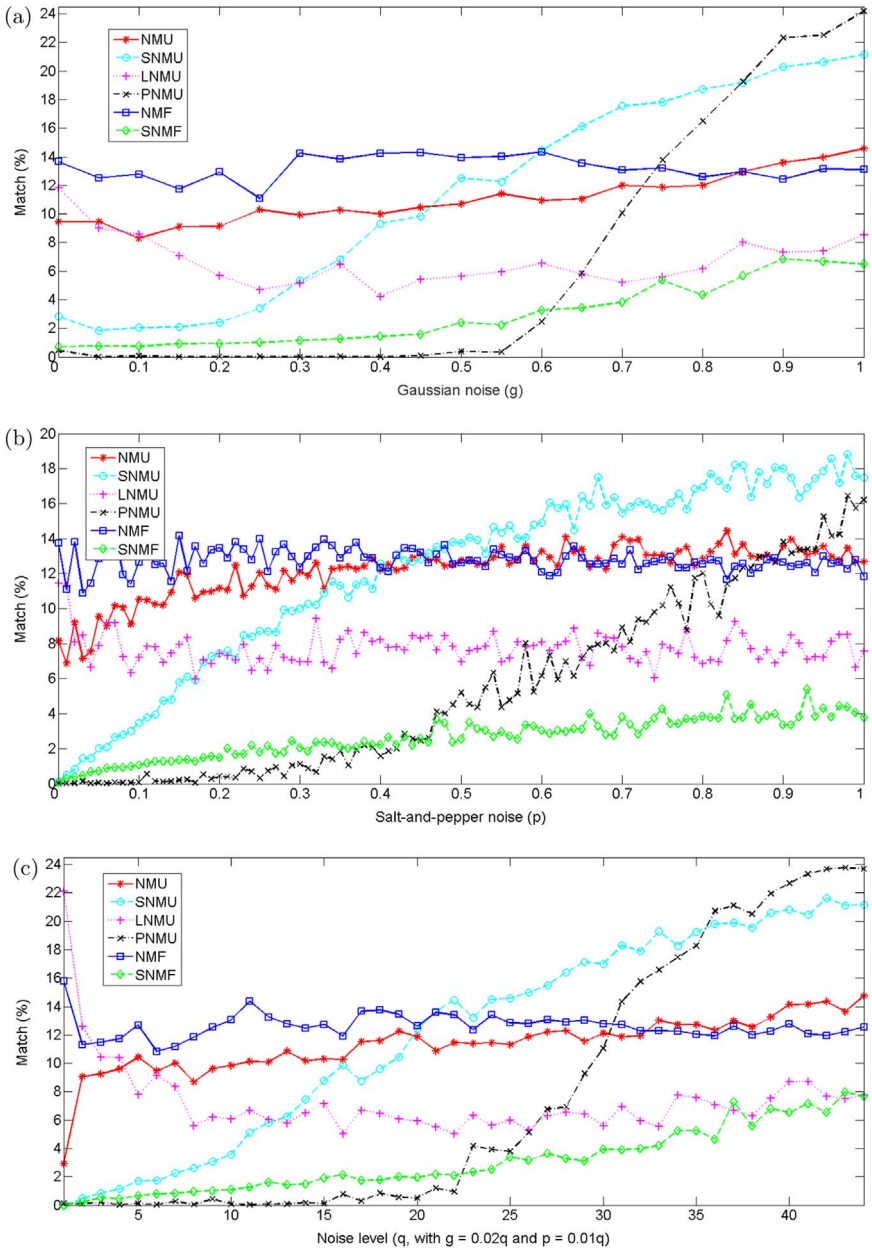


Fig. 7. (a) Evolution of the average match value (3.1) over 20 different synthetic data sets for different values of the Gaussian noise level (g) and for a fixed salt-and-pepper noise level ($p = 5\%$). (b) Evolution of the average match value (3.1) over 20 different synthetic data sets for different values of the salt-and-pepper noise level (p) and for a fixed the Gaussian noise level ($g = 10\%$). (c) Evolution of the average match value (3.1) over 20 different synthetic data sets for different values of the Gaussian noise level ($g = 0.02q$) and the salt-and-pepper noise level ($p = 0.01q$).

3.2.1. Cuprite

The Cuprite data set[1] is widely used to assess the performance of blind hyperspectral unmixing techniques. It represents spectral data collected over a mining area in southern Nevada, Cuprite. It consists of 188 images with 250×191 pixels containing about 20 different endmembers (minerals), although it is unclear how many minerals are present and where they are located precisely; see, e.g., [19,1] for more information. First we discuss the influence of the parameters μ' and φ' on the abundances obtained with PNMU. Then a quantitative comparison is performed to show the effectiveness of PNMU compared to the tested NMU and NMF variants.

Sensitivity of PNMU to the parameters μ' and φ'. In this section, we conduct some experiments to show the sensitivity of PNMU to the parameters μ' and φ'. To show a wide range of abundance elements for different values of μ' and φ', we only extract 7 abundance columns ($r=7$, light tones in the figures indicate a high degree of membership). Fig. 8 shows the first seven abundance images extracted by PNMU, when the penalty for sparsity is fixed ($\varphi' = 0.2$) and the penalty for spatial coherence varies ($\mu' = 0.1, 0.3, \ldots, 0.9$). We observe that increasing μ' improves the spatial coherence of the abundance images. However, high values of μ' (starting for $\mu' = 0.5$) lead to blurred images and loss of contours (Figs. 8c, d and e).

Adding sparsity constraints into NMU allows us to detect endmembers more effectively because sparsity prevents the mixture of different endmembers in one abundance element [11]. Fig. 9 shows the first seven bases obtained with PNMU for μ' fixed to 0.1 and where φ' varies ($\varphi' = 0.1, 0.3, \ldots, 0.9$). It can be observed that, as for the spatial information, small values of the sparsity

[1] Available at http://speclab.cr.usgs.gov/PAPERS.imspec.evol/aviris.evolution.html.

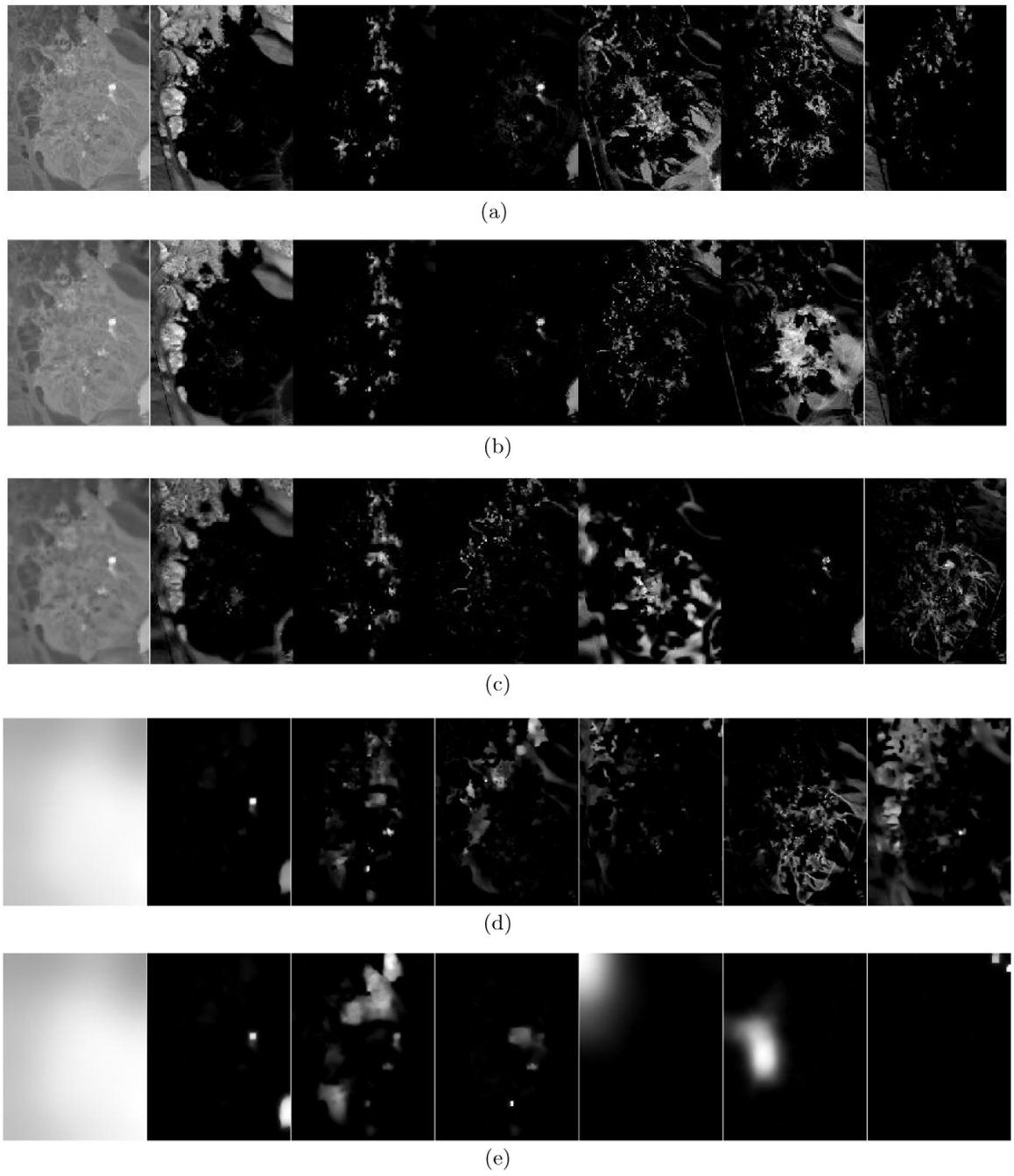


Fig. 8. Influence of the locality term on the first seven abundance images for the Cuprite data set: (a) mu = 0.1, (b) mu = 0.3, (c) mu = 0.5, (d) mu = 0.7, and (e) mu = 0.9.

parameter φ' (0.1 or 0.3) provide good results; in fact, for higher values of this parameter, the abundances are too sparse, and PNMU is not able to detect good features; see Figs. 8c, d and e.

Furthermore, it is necessary to point out that the sparsity and the locality constraints influence each other. Indeed, higher sparsity implies that less pixels are present in each abundance element which improves the spatial coherence (at the limit, all pixels take the value zero and the spatial coherence is perfect). For example, we observe in Fig. 8 that increasing μ' also strongly influences sparsity. A human supervision is necessary to adequately tune these parameters. For the Cuprite data set, we have observed that PNMU with $\mu' = 0.1$ and $\varphi' = 0.2$ returns basis elements with a good tradeoff between sparsity and spatial coherence.

Quantitative comparison. We now compare PNMU with the algorithms listed in the introduction of Section 3 to show its ability to extract features with a good trade-off between reconstruction error, sparsity and spatial coherence. Figs. 10 and 11 display the abundance images obtained by the different algorithms for $r=21$ on the Cuprite data set. Table 1 provides a quantitative comparison of the different algorithms, with the values of the relative error, sparsity and spatial coherence.

We observe the following:

- NMF is not able to detect endmembers, and generates mostly dense hence not localized abundance images, most of them containing salt-and-pepper-like noise; see Fig. Fig. 10a.
  This qualitative observation is confirmed by the quantitative measurements from Table 1: although NMF has the lowest

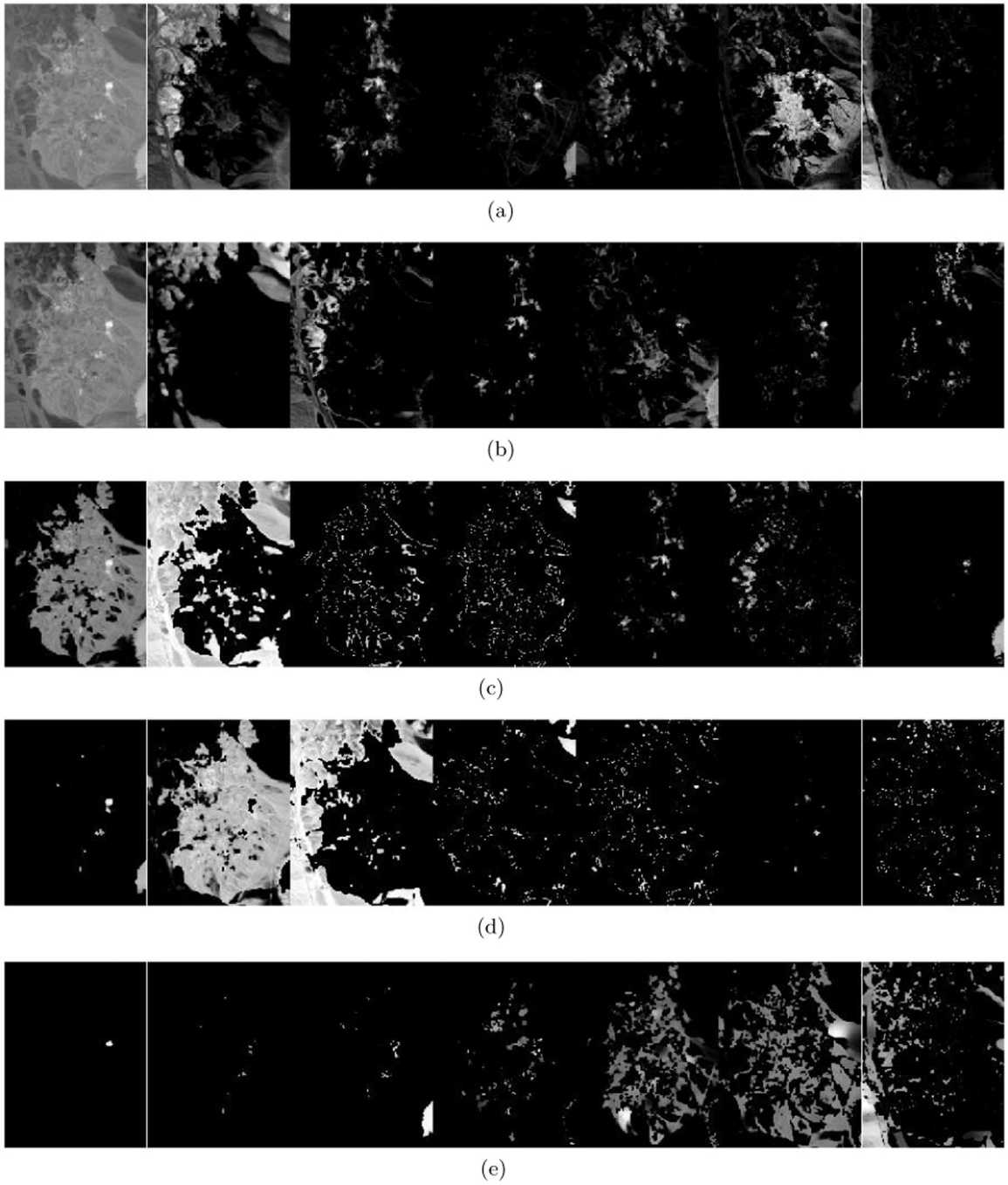


Fig. 9. Influence of the sparsity term on the first seven abundance images for the Cuprite data set. : (a) phi = 0.1, (b) phi = 0.3, (c) phi = 0.5, (d) phi=0.7, and (e) phi = 0.9.

Table 1
Comparison of the relative approximation error, the sparsity and the spatial coherence for the Cuprite data set.

	NMF	SNMF	NMU	LNMU	SNMU	PNMU
Error	0.62	2.59	1.37	1.44	1.87	1.85
Improved	**0.61**	2.02	0.71	0.66	1.13	1.09
$s(U)$	3.76	**77.96**	50.41	40.90	76.33	75.29
$\ell(U)$	3606	3829	2585	2039	2292	**1381**

Bold values indicate the best performance among the six algorithms.

reconstruction error (since it only focuses on minimizing it), it has the lowest value for the sparsity of the abundances, and the second highest value for their spatial coherence (higher values mean worse spatial coherence).

- Despite the fact that SNMF gives sparser solutions than NMF, the abundance images are surprisingly very poor in terms of local coherence (highest value); see Fig. 10b.
- NMU returns sparse abundances, and is able to localize different endmembers (see the description of the PNMU abundance maps below), even if some images are rather noisy and the edges defining the materials are not always well identified; see, e.g., 11th and 16th abundance elements in Fig. 10c. It generates sparser and spatially more coherent features than NMF, while its relative error is comparable (0.61% vs. 0.71%).
- Not surprisingly, LNMU provides spatially more coherent (from 2585 to 2039) but denser (from 50.41 to 40.90) bases than NMU.

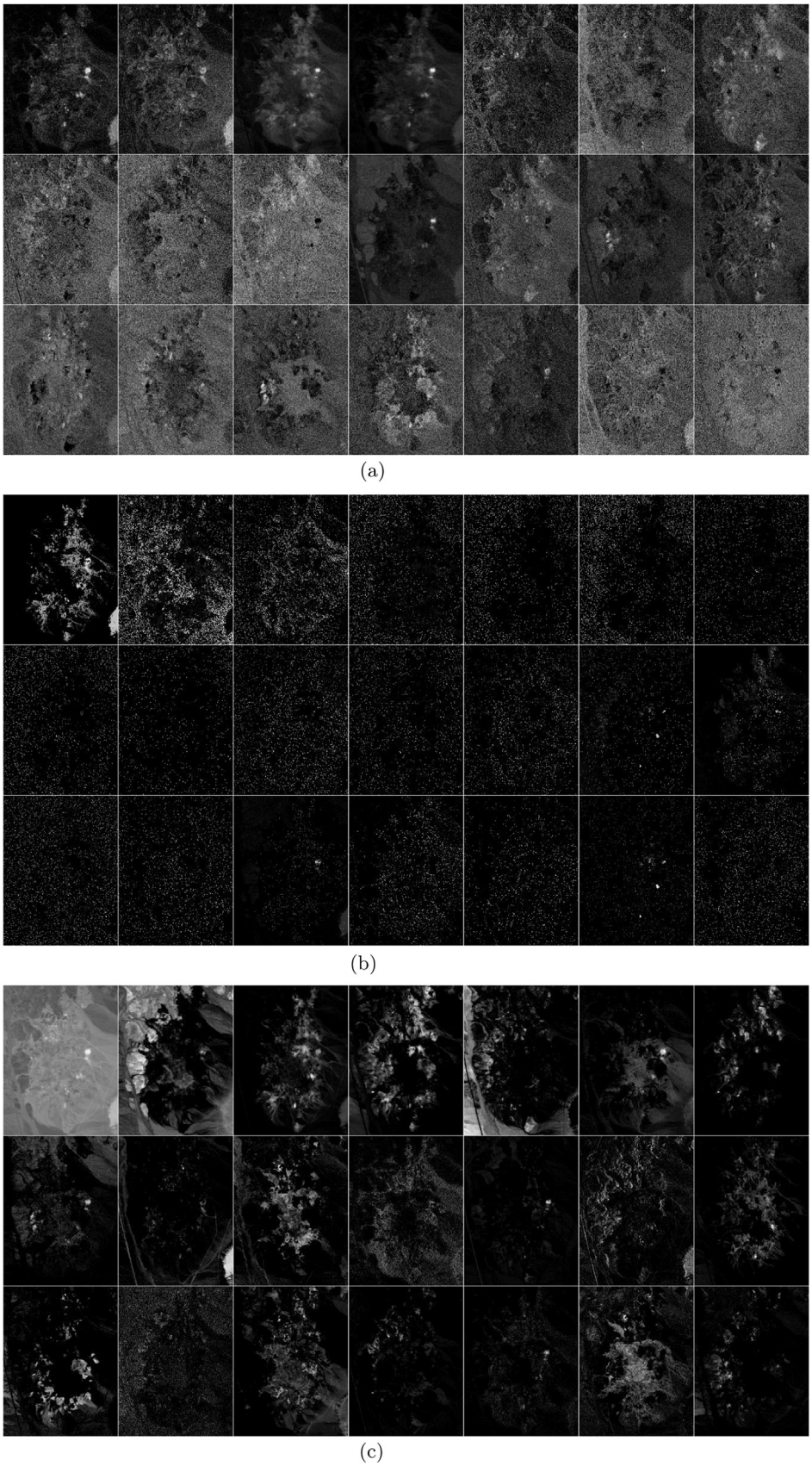
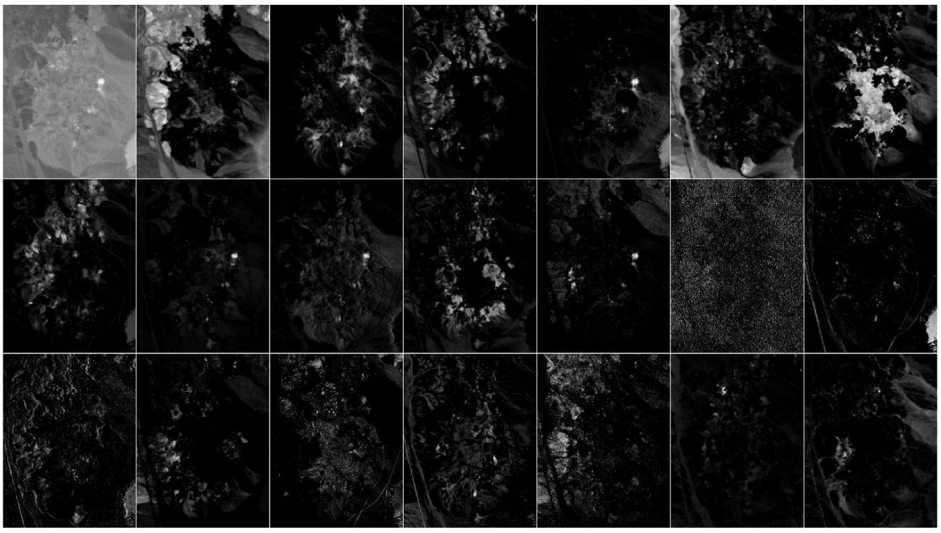
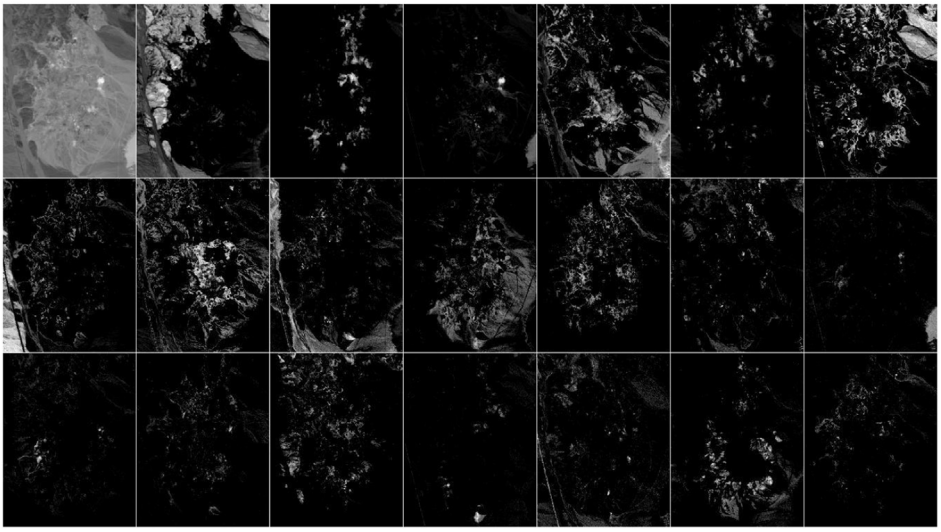


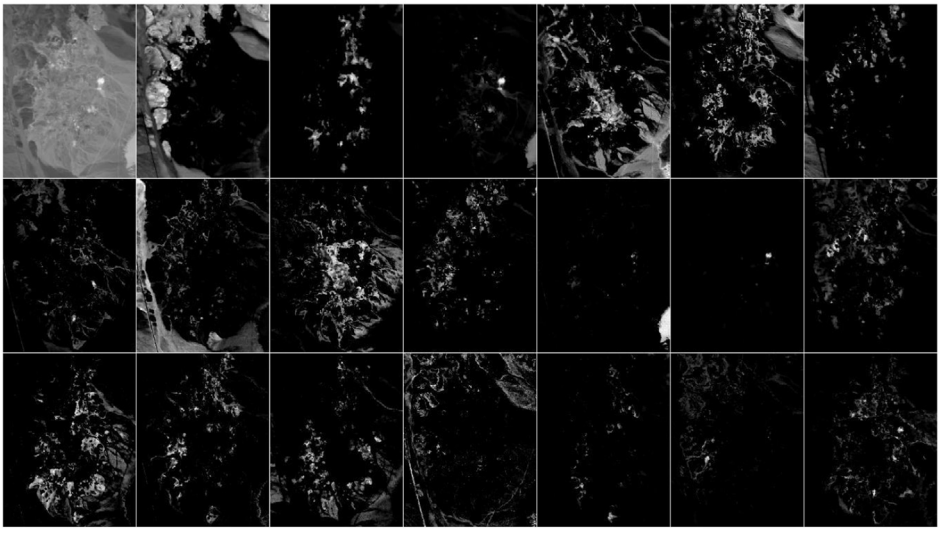
Fig. 10. Abundance elements obtained for the Cuprite data set with (a) NMF, (b) SNMF and (c) NMU algorithms



(a)



(b)



(c)

Fig. 11. Abundance elements obtained for the Cuprite data set with (a) LNMU, (b) SNMU and (c) PNMU.

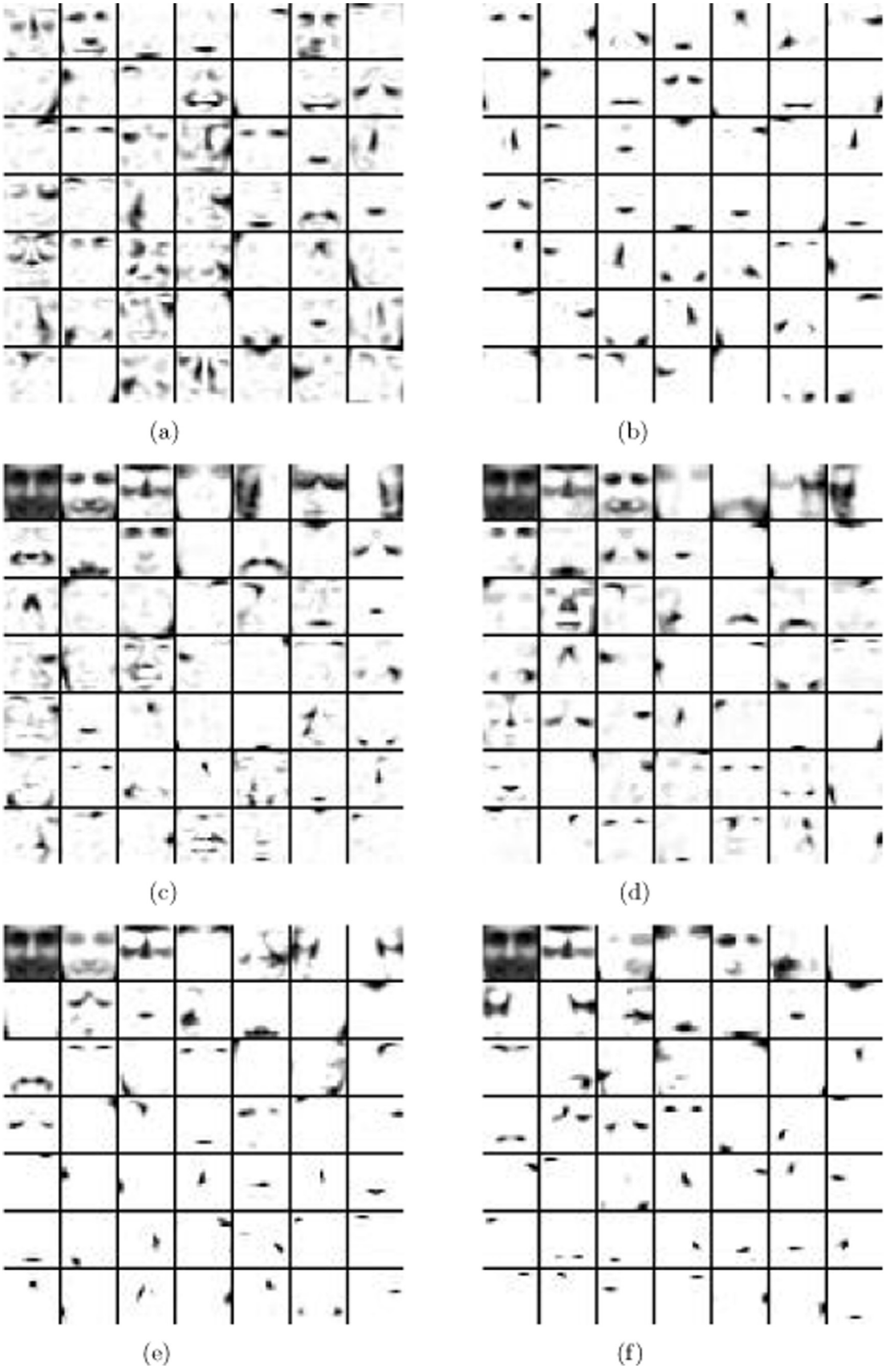


Fig. 12. Abundance elements obtained for the MIT-CBCL Face data set with (a) NMF, (b) SNMF, (c) NMU, (d) LNMU, (e) SNMU, and (f) PNMU.

Table 2
Comparison of the relative approximation error, the sparsity and the spatial coherence for the MIT-CBCL data set.

	NMF	SNMF	NMU	LNMU	SNMU	PNMU
Error	8.18	8.67	15.29	15.90	15.50	14.63
Improved	**8.16**	8.46	8.76	8.46	9.80	9.52
$s(U)$	54.89	84.80	70.05	58.65	**87.69**	85.77
$\ell(U)$	457	310	419	**55**	334	271

Bold values indicate the best performance among the six algorithms.

In particular, the edge delimitation of several materials is better defined in the LNMU abundance maps, although some are still rather noisy (e.g., 13th, 17th, and 19th) and relatively dense (e.g., 6th and 8th); see Fig. 11a.

- Adding the penalty term inducing sparsity on NMU allows us to remove some of the noise, though the edges of some materials are still not well defined (Fig. 11b). In fact, compared to NMU, SNMU quite naturally increases the sparsity of NMU abundances (from 50.41 to 76.33) and improves the spatial coherence (from 2585 to 2292), although the relative error is slightly increased (from 0.71% to 1.13%).
- PNMU achieves the best trade-off between relative error (1.1% vs. 0.6% for NMF hence an increase of only 0.5%, while NMF clearly generates poor basis elements because they are mostly dense and noisy – see the discussion above), sparsity (close to that of SNMF, 77.96% vs. 75.29%), and spatial coherence (PNMU achieves the lowest value). Fig. 11c shows the abundance images obtained with PNMU for the Cuprite data set with parameters $\mu' = 0.1$ and $\varphi' = 0.2$. It can be observed that the images are less noisy (as for SNMU), and the edges of materials are well defined (as for LNMU). Hence, PNMU combines the advantages of all methods on this data set. In fact, it is able to identify many endmembers (see [19,1] where these materials are identified) among which, from left to right, top to bottom (Fig. 11c): (2) Desert Varnish, (3) Alunite 1, (5) Chalcedony and Hematite, (6) Alunite 2, (7) Kaolinite 1 and Goethite, (9) Amorphous iron oxydes and Goethite, (10) Chalcedony, (12) Montmorillonite, (13) Muscovite, (14) Kaolinite 1, (15) K-Alunite, (18) Kaolonite 2, and (20) Buddingtonite.

3.2.2. MIT-CBCL face database

The previous experiments show the effectiveness of the proposed method in identifying materials that compose a hyperspectral image. We now apply PNMU on MIT-CBCL Face database[2] with 2429 faces, 19×19 pixels each. This is one of the most widely used data set in the NMF literature and was used in the original paper of Lee and Seung [18]. This is a database of faces and non-faces images, used at the Center for Biological and Computational Learning at MIT and we use the faces subset of the training set.

We use this data set to show that PNMU can also be used for other types of images, and provides meaningful and useful results compared to NMF and NMU variants. As for the Cuprite data set, we will use $\mu' = 0.1$ and $\varphi' = 0.2$.

Fig. 12 displays the abundance images obtained with the different algorithms, while Table 2 reports the numerical results. As for hyperspectral images, PNMU provides a good trade-off between reconstruction error, sparsity and spatial coherence.

NMF, NMU and LNMU generate mixed abundances where different parts of the faces are represented (for example in Fig. 12c, the third abundance image in the fourth row, includes the nose, the mouth, the eyes and the eyebrows all together). The sparse methods (SNMU, PNMU, and SNMF) are able to detect unmixed parts of the faces, but among these, the parts returned by PNMU are more localized. In fact, PNMU has a sparsity level comparable with the other methods enhancing sparsity, but has a better spatial coherence.

[2] http://cbcl.mit.edu/software-datasets/FaceData2.html

4. Conclusions

In this paper a variant of NMU was proposed, namely PNMU, taking into account both sparsity and spatial coherence of the abundance elements. Numerical experiments have shown the effectiveness of PNMU in correctly generating sparse and localized features in images (in particular, synthetic, hyperspectral and facial images), with a better trade-off between sparsity, spatial coherence and reconstruction error.

Acknowledgment

We would like to thank the reviewers for their insightful feedback that helped us improve the paper significantly. NG acknowledges the support by the F.R.S.-FNRS (incentive grant for scientific research no. F.4501.16) and by the ERC (starting grant no. 679515)

References

[1] A. Ambikapathi, T.-H. Chan, W.-K. Ma, C.-Y. Chi, Chance-constrained robust minimum-volume enclosing simplex algorithm for hyperspectral unmixing, IEEE Trans. Geosci. Rem. Sens. 49 (2011) 4194–4209.
[2] K.M. Anstreicher, L.A. Wolsey, Two well-known properties of subgradient optimization, Math. Program. 120 (1) (2009) 213–220.
[3] M.W. Berry, N. Gillis, F. Glineur, Document classification using nonnegative matrix factorization and underapproximation, in: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), 2009, pp. 2782–2785.
[4] A. Cichocki S. Amari R. Zdunek A.H. Phan Non-Negative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation Wiley–Blackwell, Chichester 2009.
[5] I. Daubechies, R. DeVore, M. Fornasier, C.S. Güntürk, Iteratively reweighted least squares minimization for sparse recovery, Commun. Pure Appl. Math. 63 (2010) 1–38.
[6] N. Gillis, Sparse and unique nonnegative matrix factorization through data preprocessing, J. Mach. Learn. Res. 13 (2012) 3349–3386.
[7] N. Gillis The why and how of nonnegative matrix factorization, in regularization, optimization, kernels, and support vector machines J.A.K. Suykens M. Signoretto A. Argyriou Machine Learning and Pattern Recognition Series 2014 Chapman & Hall/CRC Boca Raton, 257-291.
[8] N. Gillis, F. Glineur, Using underapproximations for sparse nonnegative matrix factorization, Pattern Recognit. 43 (2010) 1676–1687.
[9] N. Gillis, F. Glineur, Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization, Neural Comput. 24 (2012) 1085–1105.
[10] N. Gillis, R.J. Plemmons, Dimensionality reduction, classification, and spectral mixture analysis using nonnegative underapproximation, Opt. Eng. 50 (2011) 027001.
[11] N. Gillis, R.J. Plemmons, Sparse nonnegative matrix underapproximation and its application to hyperspectral image analysis, Linear Algebra Appl. 438 (2013) 3991–4007.
[12] N. Gillis, R.J. Plemmons, Q. Zhang, Priors in sparse recursive decompositions of hyperspectral images, in: Proceedings of SPIE, Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVIII, vol. 8390, 2012, pp. 83901M.
[13] M.-D. Iordache, J.M. Bioucas-Dias, A. Plaza, Total variation regularization in sparse hyperspectral unmixing, in: Third Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), Lisbon, 2011.
[14] I.T. Jolliffe Principal Component Analysis Springer-Verlag, New York, 1986.
[15] H. Kim, H. Park, Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis, Bioinformatics 23 (2007) 1495–1502.
[16] I. Kopriva, X. Chen, Y. Jao, Nonlinear band expansion and nonnegative matrix underapproximation for unsupervised segmentation of a liver from a multi-phase CT image, in: SPIE Medical Imaging-Image Processing, vol. 7962, Orlando, 2011.
[17] I. Kopriva, M. Hadzija, M.P. Hadzija, M. Korolija, A. Cichocki, Rational variety mapping for contrast-enhanced nonlinear unsupervised segmentation of multispectral images of unstained specimen, Am. J. Pathol. 179 (2011) 547–554.
[18] D.D. Lee, H.S. Seung, Learning the parts of objects by nonnegative matrix factorization, Nature 401 (1999) 788–791.
[19] J.M.P. Nascimento, J.M. Bioucas Dias, Vertex component analysis: a fast algorithm to unmix hyperspectral data, IEEE Trans. Geosci. Remote Sens. 43 (2005) 898–910.

[20] Yu. Nesterov Introductory Lectures on Convex Optimization: A Basic Course Kluwer Academic Publishers, New York, 2004.
[21] R. Smith, Introduction to hyperspectral imaging, Microimages (2006) ⟨http://www.microimages.com/documentation/Tutorials/hyprspec.pdf⟩.
[22] S.A. Vavasis, On the complexity of nonnegative matrix factorization, SIAM J. Optim. 20 (2009) 1364–1377.
[23] A. Zymnis, S.-J. Kim, J. Skaf, M. Parente, S. Boyd, Hyperspectral image unmixing via alternating projected subgradients, in: Signals, Systems and Computers, 2007, pp. 1164–1168.

Gabriella Casalino received her Ph.D. degree in Computer Science in 2015, from University of Bari (Italy), the topic of her thesis being "Non-negative factorization methods for extracting semantically relevant features in Intelligent Data Analysis". From University of Bari she also received the M.Sc. degree and the B.Sc. degree in Computer Science in 2013 and 2008, respectively. She is currently a postdoctoral research fellow at the Department of Pharmaceutical Sciences, University of Bari, working on microarray data analysis. Her research interests include computational intelligence, intelligent data analysis, and non-negative matrix factorization.

Nicolas Gillis received a master degree and a Ph.D. degree in Applied Mathematics from Université Catholique de Louvain (Belgium) in 2007 and 2011, respectively. He is currently an associate professor at the Department of Mathematics and Operational Research, Faculté Polytechnique, Université de Mons, Belgium. His research interests lie in optimization, numerical linear algebra, machine learning, data mining and hyperspectral imaging.