Fast and Robust Recursive Algorithms for Separable Nonnegative Matrix Factorization*

Nicolas Gillis
Department of Mathematics and Operational Research
Faculté Polytechnique, Université de Mons
Rue de Houdain 9, 7000 Mons, Belgium
Email: nicolas.gillis@umons.ac.be

Stephen A. Vavasis
Department of Combinatorics and Optimization
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada
Email: vavasis@math.uwaterloo.ca

Abstract

In this paper, we study the nonnegative matrix factorization problem under the separability assumption (that is, there exists a cone spanned by a small subset of the columns of the input nonnegative data matrix containing all columns), which is equivalent to the hyperspectral unmixing problem under the linear mixing model and the pure-pixel assumption. We present a family of fast recursive algorithms, and prove they are robust under any small perturbations of the input data matrix. This family generalizes several existing hyperspectral unmixing algorithms hence provide for the first time a theoretical justification of their better practical performances.

Keywords. hyperspectral unmixing, linear mixing model, pure-pixel assumption, nonnegative matrix factorization, algorithms, separability, robustness.

1 Introduction

A hyperspectral image consists of a set of images taken at different wavelengths. It is acquired by measuring the spectral signature of each pixel present in the scene, that is, by measuring the reflectance (the fraction of the incident electromagnetic power that is reflected by a surface at a given wavelength) of each pixel at different wavelengths. One of the most important tasks in hyperspectral imaging is called unmixing. It requires the identification of the constitutive materials present in the image and estimation of their abundances in each pixel. The most widely used model is the *linear mixing model*: the spectral signature of each pixel results from the additive linear combination of the spectral signatures of the constitutive materials, called endmembers, where the weights of the linear combination correspond to the abundances of the different endmembers in that pixel.

More formally, let the m-by-n matrix M correspond to a hyperspectral image with m spectral bands and n pixels, and where each entry $M_{ij} \geq 0$ of matrix M is equal to the reflectance of the jth pixel at the ith wavelength. Hence, each column m_j of M corresponds to the spectral signature of a

*This work was supported in part by a grant from the U.S. Air Force Office of Scientific Research and a Discovery Grant from the Natural Science and Engineering Research Council (Canada).

given pixel. Assuming the image contains r constitutive materials whose spectral signatures are given by the vectors $w_k \in \mathbb{R}^m_+$ $1 \leq k \leq r$, we have, in the noiseless case,

$$m_j \;=\; \sum_{k=1}^{r} w_k \, h_{kj}, \qquad \text{for } j = 1, 2, \ldots, n,$$

where $h_{kj} \geq 0$ is the abundance of the kth endmember in the jth pixel, with $\sum_{k=1}^{r} h_{kj} = 1 \; \forall j$. Defining the m-by-r matrix $W = [w_1 \, w_2 \, \ldots \, w_k] \geq 0$ and the r-by-n matrix H with $H_{kj} = h_{kj} \; \forall j, k$, the equation above can be equivalently written as $M = WH$ where M, W and H are nonnegative matrices. Given the nonnegative matrix M, hyperspectral unmixing amounts to recovery of the endmember matrix W and the abundance matrix H. This inverse problem corresponds to the nonnegative matrix factorization problem (NMF), which is a difficult [27] and highly ill-posed problem [17].

However, if we assume that, for each constitutive material, there exists at least one pixel containing only that material (a 'pure' pixel), then the unmixing problem can be solved in polynomial time: it simply reduces to identifying the vertices of the convex hull of a set of points. This assumption, referred to as the *pure-pixel assumption* [13], is essentially equivalent to the separability assumption [14]: a nonnegative matrix M is called separable if it can be written as $M = WH$ where each column of W is equal, up to a scaling factor, to a column of M. In other words, there exists a cone spanned by a small subset of the columns of M containing all columns (see Section 2.1 for more details). It is worth noting that this assumption also makes sense for other applications. For example, in text mining, each entry M_{ij} of matrix M indicates the 'importance' of word i in document j (e.g., the number of appearances of word i in text j). The factors (W, H) can then be interpreted as follows: the columns of W represent the topics (i.e., bags of words) while the columns of H link the documents to these topics. Therefore,

- Separability of M (that is, each column of W appears as a column of M) requires that, for each topic, there exists at least one document discussing only that topic (a 'pure' document).
- Separability of M^T (that is, each row of H appears as a row of M) requires that, for each topic, there exists at least one word used only by that topic (a 'pure' word).

These assumptions often make sense in practice and are actually part of several existing document generative models, see [3, 4] and the references therein.

1.1 Previous Work

We focus in this paper on hyperspectral unmixing algorithms under the linear mixing model and the pure-pixel assumption, or, equivalently, to nonnegative matrix factorization algorithms under the separability assumption. Many algorithms handling this situation have been developed by the remote sensing community, see [5] for a comprehensive overview of recent hyperspectral unmixing algorithms. Essentially, these algorithms amount to identifying the vertices of the convex hull of the (normalized) columns of M, or, equivalently, the extreme rays of the convex cone generated by the columns of M. However, as far as we know, none of these algorithms have been proved to work when the input data matrix M is only approximately separable (that is, the original separable matrix is perturbed with some noise), and many algorithms are therefore not robust to noise. However, there exists a few recent notable exceptions:

- Arora et al. [3, Section 5] proposed a method which requires the resolution of n linear programs in $\mathcal{O}(n)$ variables (n is the number of columns of the input matrix), and is therefore not suited to dealing with large-scale real-world problems. In particular, in hyperspectral imaging, n corresponds to the number of pixels in the image and is of the order of 10^6. Moreover, it needs several parameters to be estimated a priori (the noise level, and a function of the columns of W; see Section 2.4).

- Esser et al. [16] proposed a convex model with n^2 variables (see also [15] where a similar approach is presented), which is computationally expensive. In order to deal with a large-scale real-world hyperspectral unmixing problem, the authors had to use a preprocessing, namely k-means, to select a subset of the columns in order to reduce the dimension n of the input matrix. Their technique also requires a parameter to be chosen in advance (either the noise level, or a penalty parameter balancing the importance between the approximation error and the number of endmembers to be extracted), only applies to a restricted noise model, and cannot deal with repeated columns of W in the data set (i.e., repeated endmembers).

- Bittorf et al. [6] proposed a method based on the resolution of a single convex optimization problem in n^2 variables (cf. Section 5.2). In order to deal with large-scale problems ($m \sim 10^6$, $n \sim 10^5$), a fast incremental gradient descent algorithm using a parallel architecture is implemented. However, the algorithm requires several parameters to be tuned, and the factorization rank has to be chosen a priori. Moreover, it would be impractical for huge-scale problems (for example for web-related applications where $n \sim 10^9$), and the speed of convergence could be an issue.

1.2 Contribution and Outline of the Paper

In this paper, we propose a new family of recursive algorithms for nonnegative matrix factorization under the separability assumption. They have the following features:

- They are robust to noise (Theorem 3).
- They are very fast, running in approximately $6mnr$ floating point operations, while the memory requirement is low, as only one m-by-n matrix has to be stored.
- They are extremely simple to implement and would be easily parallelized.
- They do not require any parameter to be chosen a priori, nor to be tuned.
- The solution does not need to be recomputed from scratch when the factorization rank is modified, as the algorithms are recursive.
- A simple post-processing strategy allows us to identify outliers (Section 3).
- Repeated endmembers are not an issue.
- Even if the input data matrix M is not approximately separable, they identify r columns of M whose convex hull has large volume (Section 4.1).

To the best of our knowledge, no other algorithms share all these desirable properties. The weak point of our approach is that the bound on the noise to guarantee recovery is weaker than in [3, 6]; see Section 2.4. Also, we will need to assume that the matrix W is full rank, which is not a necessary condition for the approaches above [3, 16, 6]. However, in practice, this condition is satisfied in most cases. At least, it is always assumed to hold in hyperspectral imaging and text mining applications, otherwise the abundance matrix H is typically not uniquely determined; see Section 2.1. Moreover, in Section 5.2, our approach will be shown to perform in average better than the one proposed in [6] on several synthetic data sets.

The paper is organized as follows. In Section 2, we introduce our approach and derive an a priori bound on the noise to guarantee the recovery of the pure pixels. In Section 3, we propose a simple way to handle outliers. In Section 4, we show that this family of algorithms generalizes several hyperspectral unmixing algorithms, including the successive projection algorithm (SPA) [2], the automatic target generation process (ATGP) [24], the successive volume maximization algorithm (SVMAX) [11],

and the p-norm based pure pixel algorithm (TRI-P) [1]. Therefore, our analysis gives the first theoretical justification of the better performances of this family of algorithms compared to algorithms based on locating pure pixels using linear functions (such as the widely used PPI [7] and VCA [23] algorithms) which are not robust to noise. This was, until now, only experimentally observed. Finally, we illustrate these theoretical results on several synthetic data sets in Section 5.

Notation. Given a matrix X, x_k, $X_{:k}$ or $X(:,k)$ denotes its kth column, and X_{ik}, x_{ik} or $x_k(i)$ the entry at position (i,k) (ith entry of column k). For a vector x, x_i or $x(i)$ denotes the ith entry of x. The unit simplex in dimension n is denoted $\Delta^n = \{x \in \mathbb{R}^n \mid x \geq 0, \sum_{i=1}^n x_i \leq 1\}$. We use the MATLAB notation $[A,\ B] = \begin{pmatrix} A & B \end{pmatrix}$ and $[A;B] = \binom{A}{B}$. Given a matrix $W \in \mathbb{R}^{m \times r}$, $m \geq r$, we denote $\sigma_i(W)$ the singular values of W in non-decreasing order, that is,

$$\sigma_1(W) \geq \sigma_2(W) \geq \cdots \geq \sigma_r(W) \geq 0.$$

The m-by-n all-zero matrix is denoted $0_{m \times n}$ while the n-by-n identity matrix is denoted I_n (the subscripts m and n might be discarded if the dimension is clear from the context).

2 Robust Recursive NMF Algorithm under Separability

In this section, we analyze a family of simple recursive algorithms for NMF under the separability assumption; see Algorithm 1. Given an input data matrix M and a function f, it works as follows:

Algorithm 1 Recursive algorithm for separable NMF

Require: Separable matrix $M = WH \in \mathbb{R}_+^{m \times n}$ (see Assumption 1), the number r of columns to be extracted, and a strongly convex function f satisfying Assumption 2.

Ensure: Set of indices J such that $M(:,J) = W$ up to permutation; cf. Theorems 1 and 3.

1: Let $R = M$, $J = \{\}$, $j = 1$.
2: **while** $R \neq 0$ and $j \leq r$ **do**
3: $\quad j^* = \operatorname{argmax}_j f(R_{:j})$. †
4: $\quad u_j = R_{:j^*}$.
5: $\quad R \leftarrow \left(I - \frac{u_j u_j^T}{\|u_j\|_2^2}\right) R$.
6: $\quad J = J \cup \{j^*\}$.
7: $\quad j = j + 1$.
8: **end while**

† In case of a tie, we pick an index j such that the corresponding column of the original matrix M maximizes f.

at each step, the column of M maximizing the function f is selected, and M is updated by projecting each column onto the orthogonal complement of the selected column.

Remark 1 (Stopping criterion for Algorithm 1)**.** *Instead of fixing a priori the number r of columns of the input matrix to be extracted, it is also possible to stop the algorithm whenever the norm of the residual (or of the last extracted column) is smaller than some specified threshold.*

In Section 2.1, we discuss the assumptions on the input separable matrix $M = WH$ and the function f that we will need in Section 2.2 to prove that Algorithm 1 is guaranteed to recover columns of M corresponding to columns of the matrix W. Then, we analyze Algorithm 1 in case some noise is added to the input separable matrix M, and show that, under these assumptions, it is robust under any small perturbations; see Section 2.3. Finally, we compare our results with the ones from [3, 6] in Section 2.4.

2.1 Separability and Strong Convexity Assumptions

In the remainder of the paper, we will assume that the original data matrix $M = WH$ is separable, that is, each column of W appears as a column of M. Recall that this condition is implied by the pure-pixel assumption in hyperspectral imaging; see Introduction. We will also assume that *the matrix W is full rank*. This is often implicitly assumed in practice otherwise the problem is in general ill-posed, because the matrix H is then typically not uniquely determined; see, e.g., [3, 25].

Assumption 1. *The separable matrix $M \in \mathbb{R}^{m\times n}$ can be written as $M = WH = W[I_r, H']$, where $W \in \mathbb{R}^{m\times r}$ has rank r, $H \in \mathbb{R}^{r\times n}_+$, and the sum of the entries of each column of H' is at most one, that is, $\sum_{k=1}^{r} H'_{kj} \leq 1\ \forall j$, or, equivalently, $h'_j \in \Delta^r\ \forall j$.*

The assumption on matrix H is made without loss of generality by

(1) Permuting the columns of M so that the first r columns of M correspond to the columns of W (in the same order).

(2) Normalizing M so that the entries of each of its columns sum to one (except for its zero columns). In fact, we have that

$$M = WH \iff MD_M^{-1} = WD_W^{-1}(D_W H D_M^{-1}),$$

where

$$(D_X)_{ij} = \begin{cases} ||X_{:j}||_1 & \text{if } i = j \text{ and } X_{:j} \neq 0, \\ 1 & \text{if } i = j \text{ and } X_{:j} = 0, \\ 0 & \text{otherwise.} \end{cases}$$

By construction, the entries of each column of MD_M^{-1} and WD_W^{-1} sum to one (except for the zero columns of M), while the entries of each column of $(D_W H D_M^{-1})$ have to sum to one (except for ones corresponding to the zero columns of M which are equal to zero) since $M = WH$.

In the hyperspectral imaging literature, the entries of each column of matrix H are typically assumed to sum to one, hence Assumption 1 is slightly more general. This has several advantages:

- It allows the image to contain 'background' pixels with zero spectral signatures, which are present for example in hyperspectral images of objects in outer space (such as satellites).
- It allows us to take into account different intensities of light among the pixels in the image, e.g., if there are some shadow parts in the scene or if the angle between the camera and the scene varies. Hence, although some pixels contain the same material(s) with the same abundance(s), their spectral signature could differ by a scaling factor.
- In the noisy case, it allows us to take into account endmembers with very small spectral signature as noise, although it is not clear whether relaxing the sum-to-one constraint is the best approach [5].

Remark 2. *Our assumptions actually do not require the matrix M to be nonnegative, as W can be any full-rank matrix. In fact, after the first step of Algorithm 1, the residual matrix will typically contain negative entries.*

We will also need to assume that the function f in Algorithm 1 satisfies the following conditions.

Assumption 2. *The function $f : \mathbb{R}^m \to \mathbb{R}_+$ is strongly convex with parameter $\mu > 0$, its gradient is Lipschitz continuous with constant L, and its global minimizer is the all-zero vector with $f(0) = 0$.*

Notice that, for any strongly convex function g whose gradient is Lipschitz continuous and whose global minimizer is $\bar{x}$, one can construct the function $f(x) = g(\bar{x}+x) - g(\bar{x})$ satisfying Assumption 2. In fact, $f(0) = 0$ while $f(x) \geq 0$ for any x since $g(\bar{x}+x) \geq g(\bar{x})$ for any x. Recall that (see, e.g., [21]) a function is strongly convex with parameter μ if and only if it is convex and for any $x, y \in \mathrm{dom}(f)$

$$f(\delta x + (1-\delta)y) \;\leq\; \delta f(x) + (1-\delta)f(y) - \frac{\mu}{2}\delta(1-\delta)||x-y||_2^2,$$

for any $\delta \in [0,1]$. Moreover, its gradient is Lipschitz continuous with constant L if and only if for any $x, y \in \mathrm{dom}(f)$

$$||\nabla f(x) - \nabla f(y)||_2 \leq L||x-y||_2.$$

Convex analysis also tells us that if f satisfies Assumption 2 then, for any x, y,

$$f(x) + \nabla f(x)^T(y-x) + \frac{\mu}{2}||x-y||_2^2 \leq f(y)$$

and

$$f(y) \leq f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}||x-y||_2^2.$$

In particular, taking $x = 0$, we have, for any $y \in \mathbb{R}^m$,

$$\frac{\mu}{2}||y||_2^2 \quad\leq\quad f(y) \quad\leq\quad \frac{L}{2}||y||_2^2, \tag{1}$$

since $f(0) = 0$ and $\nabla f(0) = 0$ (because zero is the global minimizer of f).

The most obvious choice for f satisfying Assumption 2 is $f(x) = ||x||_2^2$; we return to this matter in Section 4.1.

2.2 Noiseless Case

We now prove that, under Assumption 1 and 2, Algorithm 1 recovers a set of indices corresponding to the columns of W.

Lemma 1. *Let $Y = [W, 0_{m\times(r-k)}] = [w_1\, w_2\, \ldots\, w_k\, 0_{m\times(r-k)}] \in \mathbb{R}^{m\times r}$ with $w_i \neq 0\ \forall i$ and $w_i \neq w_j$ $\forall i \neq j$, and let $f : \mathbb{R}^m \to \mathbb{R}_+$ be a strongly convex function with $f(0) = 0$. Then*

$$f(Yh) < \max_i f(w_i) \quad \text{for all } h \in \Delta^r \text{ such that } h \neq e_j \forall j,$$

where e_j is the jth column of the identity matrix.

Proof. By assumption on f, we have $f(w) > 0$ for any $w \neq 0$; see Equation (1). Hence, if $Yh = 0$, we have the result since $f(Yh) = 0 < f(w_i)$ for all i. Otherwise $Yh = \sum_{i=1}^k w_i h_i$ where $h_i \neq 0$ for at least one $1 \leq i \leq k$ so that

$$\begin{aligned} f(Yh) &= f\left(\sum_{i=1}^k h_i w_i + \left(1 - \sum_{i=1}^k h_i\right) 0\right) \\ &< \sum_{i=1}^k h_i f\left(w_i\right) \leq \max_i f(w_i). \end{aligned}$$

The first inequality is strict since $h \neq e_j \forall j$ and $h_i \neq 0$ for at least one $1 \leq i \leq k$, and the second follows from the fact that $\sum_{i=1}^k h_i \leq \sum_{i=1}^r h_i \leq 1$. □

Theorem 1. *Let the matrix $M = WH$ satisfy Assumption 1 and the function f satisfy Assumption 2. Then Algorithm 1 recovers a set of indices J such that $M(:, J) = W$ up to permutation.*

Proof. Let us prove the result by induction.

First step. Lemma 1 applies since f satisfies Assumption 2 while W is full rank. Therefore, the first step of Algorithm 1 extracts one of the columns of W. Assume without loss of generality the last column w_r of W is extracted, then the first residual has the form

$$R^{(1)} = \left(I - \frac{w_r w_r^T}{||w_r||_2^2}\right) WH = [W' \, 0_{m\times 1}]H = W^{(1)}H,$$

i.e., the matrix $R^{(1)}$ is obtained by projecting the columns of M onto the orthogonal complement of w_r. We observe that $W^{(1)}$ satisfies the conditions of Lemma 1 as well because W' is full rank since W is. This implies, by Lemma 1, that the second step of Algorithm 1 extracts one of the columns of W'.

Induction step. Assume that after k steps the residual has the form $R^{(k)} = [W^* \; \mathbf{0}_{m\times k}]H$ with W^* full rank. Then, by Lemma 1, the next extracted index will correspond to one of the columns of W^* (say, without loss of generality, the last one) and the next residual will have the form $R = [W^\dagger, \mathbf{0}_{m\times(k+1)}]H$ where $W^\dagger$ full rank since W^* is, and H is unchanged. By induction, after r steps, we have that the indices corresponding to the different columns of W have been extracted and that the residual is equal to zero ($R = \mathbf{0}_{m\times r}H$). □

2.3 Adding Noise

In this section, we analyze how perturbing the input data matrix affects the performances of Algorithm 1. We are going to assume that the input perturbed matrix M' can be written as $M' = M + N$ where M is the noiseless original separable matrix satisfying Assumption 1, and N is the noise with $||n_i||_2 \leq \epsilon$ for all i for some sufficiently small $\epsilon \geq 0$.

2.3.1 Analysis of a Single Step of Algorithm 1

Given a matrix W, we introduce the following notations: $\gamma(W) = \min_{i\neq j} ||w_i - w_j||_2$, $\nu(W) = \min_i ||w_i||_2$, $\omega(W) = \min\left\{\nu(W), \frac{1}{\sqrt{2}}\gamma(W)\right\}$, and $K(W) = \max_i ||w_i||_2$.

Lemma 2. *Let $Y = [W, Q]$ where $W \in \mathbb{R}^{m\times k}$ and $Q \in \mathbb{R}^{m\times(r-k)}$, and let f satisfy Assumption 2, with strong convexity parameter μ and its gradient having Lipschitz constant L. If*

$$\nu(W) > 2\sqrt{\frac{L}{\mu}}\, K(Q)\,,$$

then, for any $0 \leq \delta \leq \frac{1}{2}$,

$$f^* \quad = \quad \max_{x\in\Delta^r} f(Yx) \;\; \textit{such that } x_i \leq 1-\delta \;\; \textit{for } 1 \leq i \leq k, \tag{2}$$

satisfies $f^* \leq \max_i f(w_i) - \frac{1}{2}\,\mu\,(1-\delta)\,\delta\,\omega(W)^2$.

Proof. By strong convexity of f, the optimal solution x^* of (2) is attained at a vertex of the feasible domain $\{x \in \mathbb{R}^r \mid x_i \geq 0\,\forall i, \sum_{i=1}^r x_i \leq 1, x_i \leq 1-\delta\, 1 \leq i \leq k\}$, that is, either

(a) $x^* = 0$,

(b) $x^* = e_i$ for $k+1 \leq i \leq r$,

(c) $x^* = (1-\delta)e_j$ for $1 \leq j \leq k$,

(d) $x^* = \delta e_i + (1-\delta)e_j$ for $1 \le i, j \le k$, or

(e) $x^* = \delta e_i + (1-\delta)e_j$ for $k+1 \le i \le r$ and $1 \le j \le k$.

Before analyzing the different cases, let us provide a lower bound for f^*. Using Equation (1), we have

$$f((1-\delta)w_i) \ge \frac{1}{2}\mu(1-\delta)^2||w_i||_2^2.$$

Since $(1-\delta)w_i$ is a feasible solution and $0 \le \delta \le \frac{1}{2}$, this implies $f^* \ge \frac{\mu}{8}K(W)^2$. Recall that since f is strongly convex with parameter μ, we have

$$f\left(\delta y + (1-\delta)z\right) \le \delta f(y) + (1-\delta)f(z) - \frac{1}{2}\mu\delta(1-\delta)||y-z||_2^2.$$

Let us now analyze the different cases.

(a) Clearly, $x^* \neq 0$ since $f(0) = 0$ and $f(y) > 0$ for all $y \neq 0$, cf. Equation (1).

(b) $Yx^* = q_i$ for some i. Using Equation (1), we have

$$f^* = f(q_i) \le \frac{L}{2}K(Q)^2 < \frac{\mu}{8}\nu(W)^2 \le \frac{\mu}{8}K(W)^2 \le f^*,$$

since $\nu(W) > 2\sqrt{\frac{L}{\mu}}K(Q)$, a contradiction.

(c) $Yx^* = (1-\delta)w_i$ for some i :

$$f^* \le (1-\delta)f(w_i) - \frac{1}{2}\mu\delta(1-\delta)||w_i||_2^2. \tag{3}$$

By strong convexity, we also have $f(w_i) \ge \frac{\mu}{2}||w_i||_2^2 \ge \frac{1}{2}\mu(1-\delta)||w_i||_2^2$. Plugging it in (3) gives

$$\begin{aligned} f^* &\le f(w_i) - \delta f(w_i) - \frac{1}{2}\mu\delta(1-\delta)||w_i||_2^2 \\ &\le f(w_i) - \mu\delta(1-\delta)||w_i||_2^2 \\ &\le \max_i f(w_i) - \mu\delta(1-\delta)\nu(W)^2. \end{aligned}$$

(d) $Yx^* = \delta w_i + (1-\delta)w_j$ for some $i \neq j$:

$$\begin{aligned} f^* &\le \delta f(w_i) + (1-\delta)f(w_j) - \frac{1}{2}\mu\delta(1-\delta)||w_i - w_j||_2^2 \\ &\le \max_i f(w_i) - \mu\delta(1-\delta)\left(\frac{1}{\sqrt{2}}\gamma(W)\right)^2. \end{aligned}$$

(e) $Yx^* = \delta q_i + (1-\delta)w_j$ for some i, j. First, we have

$$\begin{aligned} f^* &\le \delta f(q_i) + (1-\delta)f(w_j) - \frac{1}{2}\mu\delta(1-\delta)||q_i - w_j||_2^2 \\ &\le f(w_j) + \delta f(q_i) - \delta f(w_j) \\ &\quad - \frac{1}{2}\mu\delta(1-\delta)\left(\nu(W) - K(Q)\right)^2. \end{aligned}$$

In fact, $||q_i - w_j||_2 \ge ||w_j||_2 - ||q_i||_2 \ge \nu(W) - K(Q)$. Then, using

$$f(q_i) \le \frac{L}{2}||q_i||_2^2 \le \frac{L}{2}K(Q)^2 < \frac{1}{4}\frac{\mu}{2}\nu(W)^2 \le \frac{1}{4}f(w_j),$$

$\nu(W) - K(Q) > \left(1 - \frac{1}{2}\sqrt{\frac{\mu}{L}}\right)\nu(W) \geq \frac{1}{2}\nu(W)$ and $f(w_j) \geq \frac{\mu}{2}(1-\delta)\nu(W)^2$, we obtain

$$\begin{aligned} f^* &< f(w_j) - \frac{3}{4}\delta f(w_j) - \frac{1}{8}\mu\delta(1-\delta)\nu(W)^2 \\ &\leq f(w_j) - \frac{1}{2}\mu\delta(1-\delta)\nu(W)^2. \end{aligned}$$

□

Lemma 3. *Let the function $f : \mathbb{R}^m \to \mathbb{R}_+$ satisfy Assumption 2, with strong convexity parameter μ and its gradient having Lipschitz constant L. Then, for any $x, n \in \mathbb{R}^m$ and any $\epsilon, K \geq 0$ satisfying $||x||_2 \leq K$ and $||n||_2 \leq \epsilon \leq K$, we have*

$$f(x+n) \leq f(x) + \left(\epsilon KL + \frac{L}{2}\epsilon^2\right) \leq f(x) + \frac{3}{2}\epsilon KL, \text{ and} \tag{4}$$

$$f(x+n) \geq f(x) - \left(\epsilon KL - \frac{\mu}{2}\epsilon^2\right) \geq f(x) - \epsilon KL. \tag{5}$$

Proof. For the upper bound (4), we use the fact that the gradient of f is Lipschitz continuous with constant L

$$\begin{aligned} f(x+n) &\leq f(x) + \nabla f(x)^T n + \frac{L}{2}||n||_2^2 \\ &\leq f(x) + \epsilon KL + \frac{L}{2}\epsilon^2 \leq f(x) + \frac{3}{2}\epsilon KL, \end{aligned}$$

for any $||x||_2 \leq K$, $||n||_2 \leq \epsilon \leq K$. The second inequality follows from the fact that $\nabla f(0) = 0$ and by Lipschitz continuity of the gradient: $||\nabla f(x) - 0||_2 \leq L||x - 0||_2 \leq LK$ for any $||x||_2 \leq K$.

For the lower bound (5), we use strong convexity

$$\begin{aligned} f(x+n) &\geq f(x) + \nabla f(x)^T n + \frac{\mu}{2}||n||_2^2 \\ &\geq f(x) - KL||n||_2 + \frac{\mu}{2}||n||_2^2 \\ &\geq f(x) - \left(\epsilon KL - \frac{\mu}{2}\epsilon^2\right), \end{aligned}$$

for any $||x||_2 \leq K$, $||n||_2 \leq \epsilon \leq K$. The third inequality follows from the fact that

$$\max_{0 \leq y \leq \epsilon} \left(yKL - \frac{\mu}{2}y^2\right) = \epsilon KL - \frac{\mu}{2}\epsilon^2,$$

since $K \geq \epsilon$ and $L \geq \mu$. □

We can now prove the theorem which will be used in the induction step to prove that Algorithm 1 works under small perturbations of the input separable matrix.

Theorem 2. *Let*

- *f satisfy Assumption 2, with strong convexity parameter μ, and its gradient have Lipschitz constant L.*
- *$M' = M + N$ with $M = YH = [W\, Q]H$, where $W \in \mathbb{R}^{m \times k}$, $K(N) \leq \epsilon$, $\nu(W) > 2\sqrt{\frac{L}{\mu}}K(Q)$, and $H = [I, H'] \in \mathbb{R}_+^{r \times n}$ where the sum of the entries of the columns of H' is at most one. We will denote $\omega(W)$ and $K(W)$, ω and K respectively.*

- *ϵ be sufficiently small so that $\epsilon \leq \frac{\mu\omega^2}{20KL}$.*

Then the index i corresponding to a column m_i' of M' that maximizes the function f satisfies

$$m_i = [W, Q]h_i, \quad \text{where } h_i(p) \geq 1 - \delta \text{ for some } 1 \leq p \leq k, \tag{6}$$

and $\delta = \frac{10\epsilon KL}{\mu\omega^2}$, which implies

$$||m_i' - w_p||_2 \leq \epsilon + 2K\delta = \epsilon\left(1 + 20\frac{K^2}{\omega^2}\frac{L}{\mu}\right). \tag{7}$$

Proof. First note that $\epsilon \leq \frac{\mu\omega^2}{20KL}$ implies $\delta = \frac{10\epsilon KL}{\mu\omega^2} \leq \frac{1}{2}$. Let us then prove Equation (6) by contradiction. Assume the extracted index, say i, for which $m_i' = m_i + n_i = Yh_i + n_i$ satisfies $h_i(l) < 1 - \delta$ for $1 \leq l \leq k$. We have

$$\begin{aligned}
f(m_i') &= f(m_i + n_i) \\
&\underset{(Lemma\ 3)}{\leq} f(Yh_i) + \frac{3}{2}\epsilon KL \\
&< \max_{x \in \Delta^r, x(l) < 1-\delta\ 1 \leq l \leq k} f(Yx) + \frac{3}{2}\epsilon KL \\
&\underset{(Lemma\ 2)}{\leq} \max_j f(w_j) - \frac{1}{2}\mu\delta(1-\delta)\omega^2 + \frac{3}{2}\epsilon KL \\
&\underset{(Lemma\ 3)}{\leq} \max_j f(w_j') - \frac{1}{2}\mu\delta(1-\delta)\omega^2 + \frac{5}{2}\epsilon KL,
\end{aligned} \tag{8}$$

where w_j' is the perturbed column of M corresponding to w_j (that is, the jth column of M'). The first inequality follows from Lemma 3. In fact, we have $\epsilon \leq K$ since $\mu \leq L$ and $\omega \leq K$, $||m_i||_2 = ||Wh_i||_2 \leq \max_i ||w_i||_2 = K$ (by convexity of $||.||_2$), and $||n_i||_2 \leq \epsilon\ \forall i$ so that $f(m_i') \leq f(m_i) + \frac{3}{2}\epsilon KL$. The second inequality is strict since the maximum is attained at a vertex with $x(l) = 1 - \delta$ for some $1 \leq l \leq k$ at optimality (see proof of Lemma 2). The third inequality follows from Lemma 2 while the fourth follows from the fact that $||w_j||_2 \leq K$ so that $f(w_j) - \epsilon KL \leq f(w_j')$ for all j by Lemma 3.

We notice that, since $\delta \leq \frac{1}{2}$,

$$\frac{1}{2}\mu\delta(1-\delta)\omega^2 \geq \frac{1}{4}\mu\omega^2\delta = \frac{1}{4}\mu\omega^2\frac{10\epsilon KL}{\mu\omega^2} = \frac{5}{2}\epsilon KL.$$

Combining this inequality with Equation (8), we obtain $f(m_i') < \max_j f(w_j')$, a contradiction since m_i' should maximize f among the columns of M' and the w_j''s are among the columns of M'.

To prove Equation (7), we use Equation (6) and observe that

$$m_i = (1 - \delta')w_p + \sum_{k \neq p} \alpha_k y_k \text{ for some } p \text{ and } 1 - \delta' \geq 1 - \delta,$$

so that $\sum_{k \neq p} \alpha_k \leq \delta' \leq \delta$. Therefore,

$$\begin{aligned}
\|m_i - w_p\|_2 &= \left\| -\delta' w_p + \sum_{k \neq p} \alpha_k w_k \right\|_2 \\
&\leq 2\delta' \max_j ||w_j||_2 \leq 2\delta' K \leq 2K\delta,
\end{aligned}$$

which gives

$$||m_i' - w_p||_2 \leq ||(m_i' - m_i) + (m_i - w_p)||_2 \leq \epsilon + 2K\delta,$$

for some $1 \leq p \leq k$. ∎

It is interesting to relate the ratio $\frac{K(W)}{\omega(W)}$ to the condition number of matrix W, given by the ratio of its largest and smallest singular values $\kappa(W) = \frac{\sigma_1(W)}{\sigma_r(W)}$.

Lemma 4. *Let $W = [w_1\, w_2\, \ldots\, w_r] \in \mathbb{R}^{m\times r}$. Then*

$$\omega(W) \geq \sigma_r(W)\,.$$

Proof. We have to show that $||w_i||_2 \geq \sigma_r(W)$ for all i, and $||w_i - w_j||_2 \geq \sqrt{2}\sigma_r(W)$ for all $i \neq j$. Let $(U, \Sigma, V) \in \mathbb{R}^{m\times r} \times \mathbb{R}^{r\times r} \times \mathbb{R}^{r\times r}$ be a compact singular value decomposition of $W = U\Sigma V^T$, where U and V are orthonormal and Σ is diagonal with the singular values of W on the diagonal. Then

$$||w_i||_2 = ||U\Sigma v_i||_2 = ||\Sigma v_i||_2 \geq \sigma_r(W)||v_i||_2 = \sigma_r(W),$$

while

$$\begin{aligned} ||w_i - w_j||_2 &= ||U\Sigma(v_i - v_j)||_2 = ||\Sigma(v_i - v_j)||_2 \\ &\geq \sigma_r(W)||v_i - v_j||_2 = \sqrt{2}\sigma_r(W). \end{aligned}$$

□

The ratio $\frac{K(W)}{\omega(W)}$ is then closely related to the conditioning of matrix $W \in \mathbb{R}^{m\times r}$. In fact, we have seen that $\omega(W) \geq \sigma_r(W)$ while, by definition, $\sigma_1(W) \geq K(W) \geq \nu(W) \geq \omega(W)$ so that

$$1 \leq \frac{K(W)}{\omega(W)} \leq \frac{\sigma_1(W)}{\sigma_r(W)} = \kappa(W).$$

In particular, this inequality implies that if $\kappa(W) = 1$ then $\frac{K(W)}{\omega(W)} = 1$.

2.3.2 Error Bound for Algorithm 1

We have shown that, if the input matrix M' has the form

$$M' = [W, Q][I_r, H'] + N,$$

where Q and N are sufficiently small and the sum of the entries of each column of $H' \geq 0$ is smaller than one, then Algorithm 1 extracts one column of M' which is close to a column of W; cf. Theorem 2. We now show that, at each step of Algorithm 1, the residual matrix satisfies these assumptions so that we can prove the result by induction.

We first give some useful lemmas; see [19] and the references therein.

Lemma 5 (Cauchy Interlacing Theorem)**.** *Let $W \in \mathbb{R}^{m\times r}$ and $P = \prod_{i=1}^{k}(I - u_i u_i^T)$ where $u_i \in \mathbb{R}^m$ with $||u_i||_2 = 1$ for all i, and $k \leq r - 1$. Then*

$$\sigma_1(W) \geq \sigma_1(PW) \geq \sigma_{r-k}(PW) \geq \sigma_r(W).$$

Lemma 6 (Singular Value Perturbation, Weyl)**.** *Let $M' = M + N \in \mathbb{R}^{r\times n}$ with $r \leq n$. Then, for all $1 \leq i \leq r$,*

$$\left|\sigma_i(M) - \sigma_i(M')\right| \;\leq\; \sigma_1(N) = ||N||_2\,.$$

Lemma 7. *Let $W = [W_1, W_2] \in \mathbb{R}^{m\times r}$ where $W_1 \in \mathbb{R}^{m\times r_1}$ and $W_2 \in \mathbb{R}^{m\times r_2}$. Let also $P = \prod_{i=1}^{r_2}\left(I - \frac{u_i u_i^T}{||u_i||_2^2}\right)$ be such that $\max_i ||PW_2(:, i)||_2 \leq \bar{\epsilon}$ for some $\bar{\epsilon} \geq 0$. Then,*

$$\sigma_{r_1}(PW_1) \geq \sigma_r(W) - \sqrt{r_2}\,\bar{\epsilon}.$$

Proof. We have

$$\begin{aligned}\sigma_{r_1}(PW_1) &= \sigma_{r_1}([PW_1, 0_{m\times r_2}]) \\ &\underset{(Lemma\ 6)}{\geq} \sigma_{r_1}([PW_1, PW_2]) - ||PW_2||_2 \\ &\geq \sigma_{r_1}(PW) - \sqrt{r_2}\max_i ||PW_2(:,i)||_2 \\ &\underset{(Lemma\ 5)}{\geq} \sigma_r(W) - \sqrt{r_2}\bar{\epsilon}\,.\end{aligned}$$

The second inequality follows from the fact that $||A||_2 \leq \sqrt{n}\max_i ||A(:,i)||_2$ for any matrix $A \in \mathbb{R}^{m\times n}$. □

We can now prove the main theorem of the paper which shows that, given a noisy separable matrix $M' = M + N = WH + N$ where M satisfies Assumption 1, Algorithm 1 is able to identify approximately the columns of W.

Theorem 3. *Let $M' = M + N = WH + N \in \mathbb{R}^{m\times n}$ where M satisfies Assumption 1 with $W \in \mathbb{R}^{m\times r}$, $r \geq 2$, and $H = [I_r\ H']$, and let f satisfy Assumption 2 with strong convexity parameter μ and its gradient has Lipschitz constant L. Let also $||n_i||_2 \leq \epsilon$ for all i with*

$$\epsilon < \sigma_r(W)\,\min\left(\frac{1}{2\sqrt{r-1}}, \frac{1}{4}\sqrt{\frac{\mu}{L}}\right)\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right)^{-1}, \tag{9}$$

and J be the index set of cardinality r extracted by Algorithm 1. Then there exists a permutation P of $\{1, 2, \ldots, r\}$ such that

$$\max_{1\leq j\leq r} ||m'_{J(j)} - w_{P(j)}||_2 \leq \bar{\epsilon} = \epsilon\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right).$$

Proof. Let us prove the result by induction. First, let us define the residual matrix $R^{(k)}$ obtained after k steps of Algorithm 1 as follows:

$$R^{(0)} = M', \qquad R^{(k+1)} = P^{(k)}R^{(k)} \text{ for } 1 \leq k \leq r-1,$$

with $P^{(k)} = \left(I - \frac{uu^T}{||u||_2^2}\right)$ is the orthogonal projection performed at step 5 of Algorithm 1 where u is the extracted column of $R^{(k)}$, that is, $u = r_i^{(k)}$ for some $1 \leq i \leq n$.

Then, let us assume that the residual $R^{(k)} \in \mathbb{R}^{m\times n}$ has the following form

$$R^{(k)} = [W^{(k)}, Q^{(k)}]H + N^{(k)},$$

where $W^{(k)} \in \mathbb{R}^{m\times(r-k)}$, $Q^{(k)} \in \mathbb{R}^{m\times k}$, $K(Q^{(k)}) \leq \bar{\epsilon}$, $\nu(W^{(k)}) > 2\sqrt{\frac{L}{\mu}}\bar{\epsilon}$, and $K(N^{(k)}) \leq \epsilon \leq \frac{\omega(W^{(k)})^2\mu}{20K(W^{(k)})L}$. Let us show that $R^{(k+1)}$ satisfies the same conditions as $R^{(k)}$. By assumption, $R^{(k)}$ satisfies the conditions of Theorem 2: the $(k+1)$th index extracted by Algorithm 1, say i, satisfies

$$||r_i^{(k)} - w_p^{(k)}||_2 \leq \epsilon\left(1 + 20\frac{K(W^{(k)})^2}{\omega(W^{(k)})^2}\frac{L}{\mu}\right),$$

for some $1 \leq p \leq r-k$. Let us assume without loss of generality that $p = r-k$. The next residual $R^{(k+1)}$ has the form

$$\begin{aligned}R^{(k+1)} = [&\underbrace{P^{(k)}W^{(k)}(:, 1{:}r-k-1)}_{W^{(k+1)}}, \underbrace{P^{(k)}w_p^{(k)}, P^{(k)}Q^{(k)}}_{Q^{(k+1)}}]H \\ &+ \underbrace{P^{(k)}N^{(k)}}_{N^{(k+1)}},\end{aligned} \tag{10}$$

where

- $K(N^{(k+1)}) \leq K(N^{(k)}) \leq \epsilon$ and $K(P^{(k)}Q^{(k)}) \leq K(Q^{(k)}) \leq \bar{\epsilon}$ because an orthogonal projection can only reduce the ℓ_2-norm of a vector.
- $||P^{(k)}w_p^{(k)}||_2 \leq \bar{\epsilon} = \epsilon\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right)$ since

$$\begin{aligned} ||P^{(k)}w_p^{(k)}||_2 &\leq ||r_i^{(k)} - w_p^{(k)}||_2 \\ &\underset{(Thm\ 2)}{\leq} \epsilon\left(1 + 20\frac{K(W^{(k)})^2}{\omega(W^{(k)})^2}\frac{L}{\mu}\right) \end{aligned}$$

where the first inequality follows from $P^{(k)}w_p^{(k)}$ being the projection of $w_p^{(k)}$ onto the orthogonal complement of $r_i^{(k)}$. Moreover,

$$\frac{K(W^{(k)})^2}{\omega(W^{(k)})^2} \leq 4\frac{K(W)^2}{\sigma_r^2(W)}. \tag{11}$$

In fact, $K(W^{(k)}) \leq K(W)$ because of the orthogonal projections, while $\omega(W^{(k)}) \geq \frac{1}{2}\sigma_r(W)$ follows from

$$\begin{aligned} \omega(W^{(k)}) &\underset{(Lemma\ 4)}{\geq} \sigma_{r-k}(W^{(k)}) \\ &\underset{(Lemma\ 7)}{\geq} \sigma_r(W) - \sqrt{k}\bar{\epsilon} \\ &\geq \sigma_r(W) - \sqrt{r-1}\bar{\epsilon} \\ &\geq \frac{1}{2}\sigma_r(W). \end{aligned}$$

Lemma 7 applies since $K(Q^{(k)}) \leq \bar{\epsilon}$. The last inequality follows from $\bar{\epsilon} \leq \frac{\sigma_r(W)}{2\sqrt{r-1}}$ since

$$\epsilon \leq \frac{\sigma_r(W)}{2\sqrt{r-1}}\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right)^{-1}.$$

For $R^{(k+1)}$ to satisfy the same conditions as $R^{(k)}$, it remains to show that $\nu(W^{(k+1)}) > 2\sqrt{\frac{L}{\mu}}\bar{\epsilon}$ and $\epsilon \leq \frac{\omega(W^{(k+1)})^2\mu}{20K(W^{(k+1)})L}$. Let us show that these hold for all $k = 0, 1, \ldots, r-1$:

- Since $\nu(W^{(k)}) \geq \omega(W^{(k)}) \geq \frac{1}{2}\sigma_r(W)$ (see above), $\nu(W^{(k)}) > 2\sqrt{\frac{L}{\mu}}\bar{\epsilon}$ is implied by $\frac{1}{2}\sigma_r(W) > 2\sqrt{\frac{L}{\mu}}\bar{\epsilon}$, that is,

$$\epsilon < \frac{1}{4}\sqrt{\frac{\mu}{L}}\sigma_r(W)\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right)^{-1}.$$

- Using Equation (11), we have

$$\begin{aligned} \epsilon &\leq \frac{\sigma_r(W)}{2\sqrt{r-1}}\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right)^{-1} \\ &\leq \sigma_r(W)\frac{\sigma_r(W)\mu}{80K(W)L} \leq \frac{\omega(W^{(k)})^2\mu}{20K(W^{(k)})L}. \end{aligned}$$

By assumption on the matrix M', these conditions are satisfied at the first step of the algorithm (we actually have that $Q^{(0)}$ is an empty matrix), so that, by induction, all the residual matrices satisfy

these conditions. Finally, Theorem 2 implies that the index i extracted by Algorithm 1 at the $(k+1)$th step satisfies

$$r_i^{(k)} = [W^{(k)}, Q^{(k)}]h_i + n_i, \quad \text{where } h_i(p) \geq 1 - \delta^{(k)},$$

$\delta^{(k)} = \frac{10\epsilon K(W^{(k)})L}{\omega(W^{(k)})^2\mu}$, and $p = r - k$ without loss of generality. Since the matrix H is unchanged between each step, this implies $m_i = Wh_i$ where $h_i(p) \geq 1 - \delta^{(k)}$, hence

$$\begin{aligned} ||m_i' - w_p||_2 &= ||m_i' - m_i + m_i - w_p||_2 \\ &\leq \epsilon + ||m_i - w_p||_2 \\ &\leq \epsilon + 2\delta^{(k)}K(W) \\ &= \epsilon + 20\frac{\epsilon K(W^{(k)})L}{\omega(W^{(k)})^2\mu}K(W) \\ &\underset{(Eq.\ 11)}{\leq} \epsilon\left(1 + 80\frac{K(W)^2}{\sigma_r^2(W)}\frac{L}{\mu}\right). \end{aligned}$$

The second inequality is obtained using $m_i = Wh_i = h_i(p)w_p + \sum_{k\neq p} h_i(k)w_k$ and $\sum_{k\neq p} h_i(k) \leq 1 - h_i(p)$ so that

$$\begin{aligned} ||w_p - m_i||_2 &= ||(1 - h_i(p))w_p - \sum_{k\neq p} h_i(k)w_k||_2 \\ &\leq 2(1 - h_i(p)) \max_k ||w_k||_2 \leq 2\delta^{(k)}K(W). \end{aligned}$$

□

2.4 Bounds for Separable NMF and Comparison with the Algorithms of Arora et al. [3] and Bittorf et al. [6]

Given a noisy separable matrix $M' = M + N = WH + N$, we have shown that Algorithm 1 is able to approximately recover the columns of the matrix W. In order to compare the bound of Theorem 3 with the ones obtained in [3] and[1] [6], let us briefly recall the results obtained in both papers: Given a separable matrix $M = WH$, the parameter α is defined as the minimum among the ℓ_1 distances between each column of W and its projection onto the convex hull of the other columns of W. Without loss of generality, it is assumed that the ℓ_1 norm of the columns of W is equal to one (by normalizing the columns of M), hence $\alpha \leq 2$. Then, given that the noise N added to the separable matrix M satisfies

$$\epsilon_1 = \max_i ||n_i||_1 \leq \mathcal{O}(\alpha^2),$$

Arora et al. [3] identify a matrix U such that

$$\max_{1\leq k\leq r} \min_{1\leq j\leq r} ||u_j - w_k||_1 \leq \mathcal{O}\left(\frac{\epsilon_1}{\alpha}\right).$$

The algorithm of Bittorf et al. [6] identifies a matrix U satisfying

$$\max_{1\leq k\leq r} \min_{1\leq j\leq r} ||u_j - w_k||_1 \leq \mathcal{O}\left(\frac{r\,\epsilon_1}{\alpha}\right),$$

given that $\epsilon_1 \leq \mathcal{O}\left(\frac{\alpha\gamma_1}{r}\right)$ where $\gamma_1 = \min_{i\neq j} ||w_i - w_j||_1 \geq \alpha$; see [18]. Let us compare these bounds to ours. Since the ℓ_1 norm of the columns of W is equal to one, we have $\sigma_r(W) \leq K(W) \leq 1$. Moreover,

[1] The error bounds in [6] were only valid for separable matrices without duplicates of the columns of W. They were later improved and generalized to any separable matrix in [18].

denoting p_i the orthogonal projection of w_i onto the linear subspace generated by the columns of W but the ith, we have

$$\sigma_r(W) \leq \min_i ||w_i - p_i||_2 \leq \min_i ||w_i - p_i||_1 \leq \alpha.$$

By Theorem 3, Algorithm 1 therefore requires

$$\epsilon_2 = \max_i ||n_i||_2 \leq \mathcal{O}\left(\frac{\sigma_r^3(W)}{\sqrt{r}}\right) \leq \mathcal{O}\left(\frac{\alpha^3}{\sqrt{r}}\right),$$

to obtain a matrix U such that

$$\max_{1\leq k\leq r} \min_{1\leq j\leq r} ||u_j - w_k||_2 \leq \mathcal{O}\left(\frac{\epsilon_2}{\sigma_r^2(W)}\right).$$

This shows that the above bounds are tighter, as they only require the noise to be bounded above by a constant proportional to α^2 to guarantee an NMF with error proportional to $\frac{\epsilon_1}{\alpha}$. In particular, if W is not full rank, Algorithm 1 will fail to extract more than rank(W) columns of W, while the value of α can be much larger than zero implying that the algorithms from [3, 6] will still be robust to a relatively large noise.

To conclude, the techniques in [3, 6] based on linear programming lead to better error bounds. However, there are computationally much more expensive (at least quadratic in n, while Algorithm 1 is linear in n, cf. Section 1.1), and have the drawback that some parameters have to be estimated in advance: the noise level ϵ_1, and

- the parameter α for Arora et al. [3] (which is rather difficult to estimate as W is unknown),
- the factorization rank r for Bittorf et al. [6][2], hence the solution has to be recomputed from scratch when the value of r is changed (which often happens in practice as the number of columns to be extracted is typically estimated with a trial and error approach).

Moreover, the algorithms from [3, 6] heavily rely on the separability assumption while Algorithm 1 still makes sense even if the separability assumption is not satisfied; see Section 4.1. Table 1 summarizes these results. Note that we keep the analysis simple and only indicate the growth in terms of n. The reason is threefold: (1) in many applications (such as hyperspectral unmixing), n is much larger than m and r, (2) a more detailed comparison of the running times would be possible (that is, in terms of m, n, and r) but is not straightforward as it depends on the algorithm used to solve the linear programs (and possibly on the parameters α and ϵ_1), and (3) both algorithms [3, 6] are at least quadratic in n (for example, the computational cost of each iteration of the first-order method proposed in [6] is proportional to mn^2, so that the complexity is linear in m).

3 Outlier Detection

It is important to point out that Algorithm 1 is very sensitive to outliers, as are most algorithms aiming to detect the vertices of the convex hull of a set of points, e.g., the algorithms from [3, 6] discussed in the previous section. Therefore, one should ideally discard the outliers beforehand, or design variants of these methods robust to outliers. In this section, we briefly describe a simple way for dealing with (a few) outliers. This idea is inspired from the approach described in [15].

[2] In their incremental gradient descent algorithm, the parameter ϵ does not need to be estimated. However, other parameters need to be tuned, namely, primal and dual step sizes.

Table 1: Comparison of robust algorithms for separable NMF.

	Arora et al. [3]	Bittorf et al. [6]	Algorithm 1
Flops	$\Omega(n^2)$	$\Omega(n^2)$	$\mathcal{O}(n)$
Memory	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
Noise	$\epsilon_1 \leq \mathcal{O}(\alpha^2)$	$\epsilon_1 \leq \mathcal{O}\left(\frac{\alpha\gamma_1}{r}\right)$	$\epsilon_2 \leq \mathcal{O}\left(\frac{\sigma_r^3(W)}{\sqrt{r}}\right)$
Error	$\|.\|_1 \leq \mathcal{O}\left(\frac{\epsilon_1}{\alpha}\right)$	$\|.\|_1 \leq \mathcal{O}\left(r\frac{\epsilon_1}{\alpha}\right)$	$\|.\|_2 \leq \mathcal{O}\left(\frac{\epsilon_2}{\sigma_r^2(W)}\right)$
Parameters	ϵ_1, α	ϵ_1, r	r

Let us assume that the input matrix M is a separable matrix $WH = W[I, H']$ satisfying Assumption 1 to which was added t outliers gathered in the matrix $T \in \mathbb{R}^{m\times t}$:

$$\begin{aligned} M &= \big[W,\, T,\, WH'\big] \\ &= [W,\, T] \begin{bmatrix} I_r & 0_{r\times t} & H' \\ 0_{t\times r} & I_t & 0_{t\times r} \end{bmatrix} \\ &= [W, T][I_{r+t}, F'] = [W, T]F, \end{aligned} \tag{12}$$

where $h'_i \in \Delta^r$ for all i, hence $f'_i \in \Delta^r$ for all i. Assuming $[W,T]$ has rank $r+t$ (hence $t \leq m-r$), the matrix M above also satisfies Assumption 1. In the noiseless case, Algorithm 1 will then extract a set of indices corresponding to columns of W and T (Theorem 1). Therefore, assuming that the matrix H' has at least one non-zero element in each row, one way to identifying the outliers would be to

(1) Extract $r+t$ columns from the matrix M using Algorithm 1,

(2) Compute the corresponding optimal abundance matrix F, and

(3) Select the r columns corresponding to rows of F with the largest sum,

see Algorithm 2. (Note that Algorithm 2 requires to solve a convex quadratic program, hence it is computationally much more expensive than Algorithm 1.) It is easy to check that Algorithm 2 will

Algorithm 2 Recursive algorithm for separable NMF with outliers

Require: A separable matrix $M \in \mathbb{R}_+^{m\times n}$ containing at most $t \leq m-r$ outliers, the number r of columns to be extracted, and a strongly convex function f satisfying Assumption 2.

Ensure: Set of indices J' such that $M(:, J') = W$ up to permutation and scaling; cf. Theorem 4.

1: Compute J, the index set of cardinality $(r+t)$ extracted by Algorithm 1.

2: $G = \operatorname{argmin}_{x_i\in\Delta^{r+t}\,\forall i} ||M - M(:,J)X||_F^2$.

3: Compute $J' \subseteq J$, the set of r indices having the largest score $||G(j,:)||_1$ among the indices in J.

recover the r columns of W because the optimal solution G computed at the second step is unique and equal to F (since $[W,T]$ is full rank), hence $||G(j,:)||_1 > 1$ for the indices corresponding to the columns of W while $||G(j,:)||_1 = 1$ for the outliers; see Equation (12).

In the noisy case, a stronger condition is necessary: the sum of the entries of each row of H' must be larger than some bound depending on the noise level. In terms of hyperspectral imaging, it means that for an endmember to be distinguishable from an outlier, its abundance in the image should be sufficiently large, which is perfectly reasonable.

Theorem 4. *Let $M' = [W, T, WH'] + N \in \mathbb{R}^{m\times n}$ where $[W, WH']$ satisfies Assumption 1 with $W \in \mathbb{R}^{m\times r}$, $T \in \mathbb{R}^{m\times t}$, $r \geq 2$, and let f satisfy Assumption 2 with strong convexity parameter μ and its gradient has Lipschitz constant L. Let also denote $B = [W, T]$, $s = r+t$, and $||n_i||_2 \leq \epsilon$ for all i with*

$$\epsilon < \sigma_s(B)\ \min\left(\frac{1}{2\sqrt{s-1}}, \frac{1}{4}\sqrt{\frac{\mu}{L}}\right)\left(1 + 80\frac{K(B)^2}{\sigma_s^2(B)}\frac{L}{\mu}\right)^{-1},$$

and J be the index set of cardinality r extracted by Algorithm 2. If

$$2(2n - t - r)\frac{\bar{\epsilon} + \epsilon}{\sigma_s(B)} < ||H'(i,:)||_1 \quad \text{for all } i, \tag{13}$$

then there exists a permutation P of $\{1, 2, \ldots, r\}$ such that

$$\max_{1\leq j\leq r} ||m'_{J(j)} - w_{P(j)}||_2 \leq \bar{\epsilon} = \epsilon\left(1 + 80\frac{K(B)^2}{\sigma_s^2(B)}\frac{L}{\mu}\right).$$

Proof. By Theorem 3, the columns extracted at the first step of Algorithm 2 correspond to the columns of W and T up to error $\bar{\epsilon}$. Let then $W + N_W$ and $T + N_T$ be the columns extracted by Algorithm 1 with $K(N_W), K(N_T) \leq \bar{\epsilon}$.

At the second step of Algorithm 2, the matrix G is equal to

$$G = \text{argmin}_{x_i\in\Delta^{r+t}\,\forall i}\, ||M' - [W + N_W, T + N_T]X||_F^2,$$

up to the permutation of its rows. It remains to show that

$$\min_{1\leq i\leq r} ||G(i,:)||_1 \ > \ \max_{r+1\leq i\leq r+t} ||G(i,:)||_1, \tag{14}$$

so that the last step of Algorithm 2 will identify correctly the columns of W among the ones extracted at the first step. We are going to show that

$$G \approx F = \begin{bmatrix} I_r & 0_{r\times t} & H' \\ 0_{t\times r} & I_t & 0_{t\times r} \end{bmatrix}.$$

More precisely, we are going to prove the following lower (resp. upper) bounds for the entries of the first r (resp. last t) rows of G:

(a) For $1 \leq i \leq r$, $G_{ij} \geq \max\left(0, F_{ij} - 2\frac{\bar{\epsilon}+\epsilon}{\sigma_s(B)}\right)$ for all j.

(b) For $r + 1 \leq i \leq r + t$, $G_{ij} \leq \min\left(1, F_{ij} + 2\frac{\bar{\epsilon}+\epsilon}{\sigma_s(B)}\right)$ for all j.

Therefore, assuming (a) and (b) hold, we obtain

$$\begin{aligned}
\min_{1\leq i\leq r} ||G(i,:)||_1 &\underset{(a)}{\geq} \left(1 - 2\frac{\bar{\epsilon} + \epsilon}{\sigma_s(B)}\right) + ||H'(i,:)||_1 \\
&\quad - 2(n - t - r)\frac{\bar{\epsilon} + \epsilon}{\sigma_s(B)} \\
&\underset{Eq.(13)}{>} 1 + 2(n-1)\frac{\bar{\epsilon} + \epsilon}{\sigma_s(B)} \\
&\underset{(b)}{\geq} \max_{r+1\leq i\leq r+t} ||G(i,:)||_1,
\end{aligned}$$

which proves the result. It remains to prove (a) and (b). First, we have that

$$||m'_j - [W + N_W, T + N_T]G_{:j}||_2 \leq \epsilon + \bar{\epsilon} \quad \text{for all } j. \tag{15}$$

In fact, for all j,

$$\begin{aligned}||m'_j - [W + N_W, T + N_T]G_{:j}||_2 \\ &\leq ||m_j + n_j - [W,T]F_{:j} - [N_W, N_T]F_{:j}||_2 \\ &\leq ||m_j - [W,T]F_{:j}||_2 + \epsilon + \bar{\epsilon} = \epsilon + \bar{\epsilon},\end{aligned}$$

since $G(:,j)$ leads to the best approximation of m'_j over Δ (see step 2 of Algorithm 2) and $f_j \in \Delta$.

Then, let us prove the upper bound for the block of matrix G at position $(2,1)$, that is, let us prove that

$$G_{ij} \leq 2\frac{\bar{\epsilon} + \epsilon}{\sigma_s(B)} \quad \text{for all } r+1 \leq i \leq r+t \text{ and } 1 \leq j \leq r.$$

(Note that $0 \leq G \leq 1$ by construction, hence some of the bounds are trivial, e.g., for the block (1,2).) The derivations necessary to obtain the bounds for the other (non-trivial) blocks are exactly the same and are then omitted here. Let $r+1 \leq i \leq t$ and $1 \leq j \leq r$ and denote $G_{ij} = \delta$, and let also $I = \{1, 2, \ldots, r+t\} \backslash \{i\}$. We have

$$\begin{aligned}||m'_j - [W + N_W, T + N_T]G_{:j}||_2 \\ &= ||(w_j + n_j) + BG_{:j} + [N_W, N_T]G_{:j}||_2 \\ &\geq ||w_j + B(:,I)G(I,j) + B(:,i)\delta||_2 - \epsilon - \bar{\epsilon} \\ &\geq \min_{y \in \mathbb{R}^{r+t-1}} \delta||B(:,I)y - B(:,i)||_2 - \epsilon - \bar{\epsilon} \\ &\geq \delta\sigma_s(B) - \epsilon - \bar{\epsilon}. \qquad (16)\end{aligned}$$

The first inequality follows from $K(N) \leq \epsilon$, $K([N_W, N_T]) \leq \bar{\epsilon}$ and $G_{:j} \in \Delta^{r+t}$, while the second inequality follows from the fact that w_j is a column of $B(:,I)$. The last inequality follows from the fact that the projection of any column of B onto the subspace spanned by the other columns is at least $\sigma_s(B)$. Finally, using Equations (15) and (16), we have $\bar{\epsilon} + \epsilon \geq \delta\sigma_s(B) - \bar{\epsilon} - \epsilon$, hence $G_{ij} = \delta \leq 2\frac{\bar{\epsilon}+\epsilon}{\sigma_s(B)}$. □

4 Choices for f and Related Methods

In this section, we discuss several choices for the function f in Algorithm 1, and relate them to existing methods.

4.1 Best Choice with $\mu = L$: $f(x) = ||x||_2^2$

According to our derivations (see Theorem 3), using functions f whose strong convexity parameter μ is equal to the Lipschitz constant L of its gradient is the best possible choice (since it minimizes the error bounds). The only function satisfying Assumption 2 along with this condition is, up to a scaling factor, $f(x) = ||x||_2^2 = \sum_{i=1}^m x_i^2$. In fact, Assumption 2 implies $\frac{\mu}{2}||x||_2^2 \leq f(x) \leq \frac{L}{2}||x||_2^2$; see Equation (1). However, depending on the problem at hand, other choices could be more judicious (see Sections 4.2 and 4.3). It is worth noting that Algorithm 1 with $f(x) = ||x||_2^2$ has been introduced and analyzed by several other authors:

- **Choice of the Reflections for QR Factorizations.** Golub and Businger [8] construct QR factorizations of matrices by performing, at each step of the algorithm, the Householder reflection with respect to the column of M whose projection onto the orthogonal complement of the previously extracted columns has maximum ℓ_2-norm.

- **Successive Projection Algorithm.** Araújo et al. [2] proposed the successive projection algorithm (SPA), which is equivalent to Algorithm 1 with $f(x) = ||x||_2^2$. They used it for variable selection in spectroscopic multicomponent analysis, and showed it works better than other standard techniques. In particular, they mention 'SPA seems to be more robust than genetic algorithms' but were not able to provide a rigorous justification for that fact (which our analysis does). Ren and Chang [24] rediscovered the same algorithm, which was referred to as the automatic target generation process (ATGP). It was empirically observed in [29] to perform better than other hyperspectral unmixing techniques (namely, PPI [7] and VCA [23]). However, no rigorous explanation of their observations was provided. In Section 5, we describe these techniques and explain why they are not robust to noise, which theoretically justifies the better performances of Algorithm 1. Chan et al. [11] analyzed the same algorithm (with the difference that the data is preprocessed using a linear dimensionality reduction technique). The algorithm is referred to as the successive volume maximization algorithm (SVMAX). They also successfully use Algorithm 1 as an initialization for a more sophisticated approach which does not take into account the pure-pixel assumption.

- **Greedy Heuristic for Volume Maximization.** Çivril and Magdon-Ismail [9, 10] showed that Algorithm 1 with $f(x) = ||x||_2^2$ is a very good greedy heuristic for the following problem: given a matrix M and an integer r, find a subset of r columns of M whose convex hull has maximum volume. More precisely, unless $\mathcal{P} = \mathcal{NP}$, they proved that the approximation ratio guaranteed by the greedy heuristic is within a logarithmic factor of the best possible achievable ratio by any polynomial-time algorithm. However, the special case of separable matrices was not considered. This is another advantage of Algorithm 1: even if the input data matrix M is not approximately separable, it identifies r columns of M whose convex hull has large volume. For the robust algorithms from [3, 6] discussed in Section 2.4, it is not clear whether they will be able to produce a meaningful output in that case; see also Section 5.2 for some numerical experiments.

4.2 Generalization: $f(x) = \sum_{i=1}^m h(x_i)$

Given a one-dimensional function $h : \mathbb{R} \to \mathbb{R}_+$ satisfying Assumption 2, it is easy to see that the separable function $f(x) = \sum_{i=1}^m h(x_i)$ also satisfies Assumption 2 (notice that $h(y) = y^2$ gives $f(x) = ||x||_2^2$.). For example, we could take

$$h(y) = \frac{y^2}{\alpha + |y|}, \quad \alpha > 0, \quad \text{hence } f(x) = \sum_{i=1}^m \frac{x_i^2}{\alpha + |x_i|}.$$

This choice limits the impact of large entries in x, hence would potentially be more robust to outliers. In particular, as α goes to zero, $f(x)$ converges to $||x||_1$ while, when α goes to infinity, it converges to $\frac{||x||_2^2}{\alpha}$ (in any bounded set).

Lemma 8. *In the ball $\{y \in \mathbb{R} \mid |y| \leq K\}$, the function*

$$h(y) = \frac{y^2}{\alpha + |y|}, \quad \alpha > 0,$$

is strongly convex with parameter $\mu = \frac{2\alpha^2}{(\alpha+K)^3}$ and its gradient is Lipschitz continuous with constant $L = \frac{2}{\alpha}$.

Proof. On can check that

$$\frac{2\alpha^2}{(\alpha + K)^3} \leq h''(y) = \frac{2\alpha^2}{(\alpha + |y|)^3} \leq \frac{2}{\alpha}, \quad \text{for any } |y| \leq K \ ,$$

hence $\mu = \frac{2\alpha^2}{(\alpha+K)^3}$, and $L = \frac{2}{\alpha}$. □

For example, one can choose $\alpha = K$ for which we have $\frac{L}{\mu} = 2$, which is slightly larger than one but is less sensitive to large, potentially outlying, entries of M. Let us illustrate this on a simple example:

$$M' = \begin{pmatrix} 2 & 2 \\ 0 & 1 \\ 2 & 2 \\ 1 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & 0.5 \end{pmatrix} + \begin{pmatrix} 0 & 0 & \epsilon \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{17}$$

One can check that, for any $\epsilon \leq 0.69$, Algorithm 1 with $f(x) = ||x||_2^2$ recovers the first two columns of M, that is, the columns of W. However, using $f(x) = \sum_i \frac{x_i^2}{1+|x_i|}$, Algorithm 1 recovers the columns of W for any $\epsilon \leq 1.15$. Choosing appropriate function $f(x)$ depending on the input data matrix and the noise model is a topic for further research.

Remark 3 (Relaxing Assumption 2)**.** *The condition that the gradient of f must be Lipschitz continuous in Assumption 2 can be relaxed to the condition that the gradient of f is continuously differentiable. In fact, in all our derivations, we have always assumed that f was applied on a bounded set (more precisely, the ball $\{x \mid ||x||_2 \leq K(W)\}$). Since $g \in \mathcal{C}^1$ implies that g is locally Lipschitz continuous, the condition $f \in \mathcal{C}^2$ is sufficient for our analysis to hold. Similarly, the strong convexity condition can be relaxed to local strong convexity.*

It would be interesting to investigate more general classes of functions for which our derivations hold. For example, for any increasing function $g : \mathbb{R}_+ \to \mathbb{R}$, the output of Algorithm 1 using f or using $(g \circ f)$ will be the same, since $f(x) \geq f(y) \iff g(f(x)) \geq g(f(y))$. Therefore, for our analysis to hold, it suffices that $(g \circ f)$ satisfies Assumption 2 for some increasing function g. For example, $||x||_2^4$ is not strongly convex although it will output the same result as $||x||_2^2$ hence will satisfy the same error bounds.

4.3 Using ℓ_p-norms

Another possible choice is

$$f(x) \;=\; ||x||_p^2 \;=\; \left(\sum_{i=1}^{m} |x_i|^p \right)^{2/p}.$$

For $1 < p \leq 2$, $f(x)$ is strongly convex with parameter $2(p-1)$ with respect to the norm $||.||_p$ [22, Section 4.1.1], while its gradient is locally Lipschitz continuous (see Remark 3). For $2 \leq p < +\infty$, the gradient of $f(x)$ is Lipschitz continuous with respect to the norm $||.||_p$ with constant $2(p-1)$ (by duality), while it is locally strongly convex. Therefore, f satisfies Assumption 2 for any $1 < p < +\infty$ in any bounded set, hence our analysis applies. Note that, for $p = 1$ and $p = +\infty$, the algorithm is not guaranteed to work, even in the noiseless case (when points are on the boundary of the convex hull of the columns of W): consider for example the following separable matrices

$$M = \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & 0.5 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & 0.5 \end{pmatrix},$$

for which the ℓ_1-norm may fail (selecting the last column of M), and

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0.5 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & 0.5 \end{pmatrix},$$

for which the ℓ_∞-norm may fail. Similarly as in the previous section, using ℓ_p-norms with $p \neq 2$ might be rewarding in some cases. For example, for $1 < p < 2$, the ℓ_p-norm is less sensitive to large entries

of M. Consider the matrix from Equation (17): for $p = 1.5$, Algorithm 1 extracts the columns of W for any $\epsilon \leq 0.96$, while for $p = 4$, it only works for $\epsilon \leq 0.31$ (recall for $p = 2$ we had $\epsilon \leq 0.69$).

Algorithm 1 with $f(x) = ||x||_p$ has been previously introduced as the ℓ_p-norm based pure pixel algorithm (TRI-P) [1], and shown to perform, in average, better than other existing techniques (namely, N-FINDR [28], VCA [23], and SGA [12]). The authors actually only performed numerical experiments for $p = 2$, but did not justify this choice (the reason possibly is that it gave the best numerical results, as our analysis suggests), and could not explain why Algorithm 1 performs better than other approaches in the noisy case.

5 Numerical Experiments

In the first part of this section, we compare Algorithm 1 with several fast hyperspectral unmixing algorithms under the linear mixing model and the pure-pixel assumption. We first briefly describe them (computational cost and main properties) and then perform a series of experiments on synthetic data sets in order to highlight their properties. For comparisons of Algorithm 1 with other algorithms on other synthetic and real-world hyperspectral data sets, we refer the reader to [2, 24, 29, 30, 11, 1] since Algorithm 1 is a generalization of the algorithms proposed in [2, 24, 11, 1]; see Section 4.

In the second part of the section, we compare Algorithm 1 with the Algorithm of Bittorf et al. [6].

5.1 Comparison with Fast Hyperspectral Unmixing Algorithms

We compare the following algorithms:

1. **Algorithm 1 with** $f(x) = ||x||_2^2$. We will only test this variant because, according to our analysis, it is the most robust. (Comparing different variants of Algorithm 1 is a topic for further research.) The computational cost is rather low: steps 3 and 5 are the only steps requiring computation, and have to be performed r times. We have

   - Step 3. Compute the squared norm of the columns of R, which requires n times $2m$ operations (squaring and summing the elements of each column), and extract the maximum, which requires n comparisons, for a total of approximately $2mn$ operations.
   - Step 5. It can be compute in the following way

   $$R \leftarrow \left(I - \frac{u_j u_j^T}{||u_j||_2^2} \right) R = R - \frac{u_j}{||u_j||_2^2} (u_j^T R),$$

   where computing $x^T = u_j^T R$ requires $2mn$ operations, $y = \frac{u_j}{||u_j||_2^2}$ m operations, and $R - yx^T$ $2mn$ operations, for a total of approximately $4mn$ operations.

   The total computational cost of Algorithm 1 is then about $6mnr$ operations, plus some negligible terms.

   Remark 4 (Sparse Matrices)**.** *If the matrix M is sparse, R will eventually become dense which is often impractical. Therefore, R should be kept in memory as the original matrix M minus the rank-one updates.*

   Remark 5 (Reducing Running Time)**.** *Using recursively the formula*

   $$||(I - uu^T)v||_2^2 = ||v||_2^2 - (u^T v)^2,$$

   for any $u, v \in \mathbb{R}^m$ with $||u||_2 = 1$, we can reduce the computational cost to $2mnr + \mathcal{O}(mr^2)$ operations. Although this formula is unstable when u is almost parallel to v, this is negligible as

we are only interested in the column with the largest norm (on all the tested data sets, including the 40000 randomly generated matrices below, we have always obtained the same results with both implementations). This is the version we have used for our numerical experiments as it turns out to be much faster (for example, on 200-by-200 matrices with $r = 20$, it is about seven times faster, and on a real-world 188-by-47750 hyperspectral image with $r = 15$, about 20 times faster –taking less than half a second), and handles sparse matrices as it does not compute the residual matrix explicitly (for example, it takes about half a second for the 19949-by-43586 20-newsgroup data set for $r = 20$). Note that all experiments have been performed with MATLAB R2011b on a laptop with 2GHz Intel Core i7-2630QM. The code is available at `https://sites.google.com/site/nicolasgillis/code`.

2. **Pure Pixel Index (PPI)** [7]. PPI uses the fact that the maxima (and minima) of randomly generated linear functions over a polytope are attained on its vertices with probability one. Hence, PPI randomly generates a large number of linear functions (that is, functions $f(x) = c^T x$ where $c \in \mathbb{R}^m$ is uniformly distribution over the sphere), and identifies the columns of M maximizing and minimizing these functions. Under the separability assumption, these columns must be, with probability one, vertices of the convex hull of the columns of M. Then, a score is attributed to each column of M: it is equal to the number of times the corresponding column is identified as a minimizer or maximizer of one of the randomly generated linear functions. Finally, the r columns of M with the largest score are identified as the columns of W. Letting K be the number of generated linear functions, we have to evaluate K times linear functions overs n vertices in dimension m for a total computational cost of $\mathcal{O}(Kmn)$. For our experiments, we will use $K = 1000$. There are several pitfalls in using PPI:

   (a) It is not robust to noise. In fact, linear functions can be maximized at any vertex of the convex hull of a set of points. Therefore, in the noisy case, as soon as a column of the perturbed matrix M' is not contained in the convex hull of the columns of W, it can be identified as a vertex. This can occur for arbitrarily small perturbation, as will be confirmed by the experiments below.

   (b) If not enough linear functions are generated, the algorithm might not be able to identify all the vertices (even in the noiseless case). This is particularly critical in case of ill-conditioning because the probability that some vertices maximize a randomly generated linear function can be arbitrarily low.

   (c) If the input noisy data matrix contains many columns close to a given column of matrix W, the score of these columns will be typically small (they essentially share the score of the original column of matrix W), while an isolated column which does not correspond to a column of W could potentially have a higher score than these columns, hence be extracted. This can for example be rather critical for hyperspectral images where there typically are many pixels close to pure pixels (i.e., columns of M corresponding to the same column of W). Moreover, for the same reasons, PPI might extract columns of M corresponding to the same column of W.

3. **Vertex Component Analysis (VCA)** [23]. The first step of VCA is to preprocess the data using principal component analysis which requires $O(nm^2 + m^3)$ [23]. Then, the core of the algorithm requires $O(rm^2)$ operations (see below), for a total of $O(nm^2 + m^3)$ operations[3]. Notice that the preprocessing is particularly well suited for data sets where $m \ll n$, such as hyperspectral images, where m is the number of hyperspectral images with $m \sim 100$, while n is the number of pixels per image with $n \sim 10^6$. The core of VCA is very similar to Algorithm 1: at each step, it projects the data onto the orthogonal complement of the extracted column.

[3]The code is available at `http://www.lx.it.pt/~bioucas/code.htm`.

However, instead of using a strictly convex function to identify a vertex of the convex hull of M (as in Algorithm 1), it uses a randomly generated linear function (that is, it selects the column maximizing the function $f(x) = c^T x$ where c is randomly generated, as PPI does). Therefore, for the same reasons as for PPI, the algorithm is not robust. However, it solves the second and third pitfalls of PPI (see point (b) and (c) above), that is, it will always be able to identify enough vertices, and a cluster of points around a vertex are more likely to be extracted than an isolated point. Note that, in the VCA implementation, only one linear function is generated at each step which makes it rather sensitive to this choice. Finally, the two main differences between VCA and Algorithm 1 are that: (1) VCA uses a pre-processing (although we could implement a version of Algorithm 1 with the same pre-processing), and (2) VCA uses randomly generated linear functions to pick vertices of the convex hull of the columns of M (which makes it non-robust to noise, and also non-deterministic).

4. **Simplex Volume Maximization (SiVM)** [26]. SiVM recursively extracts columns of the matrix M while trying to maximize the volume of the convex hull of the corresponding columns. Because evaluating the volumes induced by adding a column not yet extracted to the previously selected ones is computationally expensive, the function is approximated via a heuristic, for a total computational cost of $O(mnr)$ operations (as for Algorithm 1). The heuristic assumes that the columns of W are located at the same distance, that is, $||w_i - w_j||_2 = ||w_k - w_l||_2$ for all $i \neq j$ and $k \neq l$. Therefore, there is not guarantee that the algorithm will work, even in the noiseless case; this will be particularly critical for ill-conditioned problems, which will be confirmed by the experiments below.

We now generate several synthetic data sets allowing to highlight the properties of the different algorithms, and in particular of Algorithm 1. We are going to consider noisy separable matrices generated as follows.

1. The matrix W will be generated in two different ways :

   (a) *Uniform Distribution.* The entries of matrix $W \in \mathbb{R}^{200\times 20}$ are randomly and independently generated following an uniform distribution between 0 and 1 (using the *rand()* function of MATLAB).

   (b) *Ill-conditioned.* First, we generate a matrix $W' \in \mathbb{R}^{200\times 20}$ as above (that is, using the *rand()* function of MATLAB). Then, we compute the compact SVD ($U \in \mathbb{R}^{200\times 20}, \Sigma \in \mathbb{R}^{20\times 20}, V \in \mathbb{R}^{20\times 20}$) of $W' = U\Sigma V^T$, and finally generate $W = USV^T$ where S is a diagonal matrix whose diagonal entries are equal to α^{i-1} for $i = 1, 2, \ldots, 20$ where $\alpha^{19} = 10^{-3}$ (that is, $\alpha = 0.695$) in such a way that $\sigma_1(W) = 1$ and $\sigma_{20}(W) = 10^{-3}$, hence $\kappa(W) = 1000$. (Note that W might not be nonnegative, a situation which is handled by the different algorithms.)

2. The matrices H and N will be generated in two different ways as well :

   (c) *Middle Points.* We set $H = [I_{20}, H'] \in \mathbb{R}^{20\times 210}$, where the columns of H' contain all possible combinations of two non-zero entries equal to 0.5 at different positions (hence H' has $\binom{20}{2} = 190$ columns). This means that the first 20 columns of M are equal to the columns of W while the 190 remaining ones are equal to the middle points of the columns of W. We do not perturb the first 20 columns of M (that is, $n_i = 0$ for $1 \leq i \leq 20$), while, for the 190 remaining ones, we use

   $$n_i = \delta\,(m_i - \bar{w}) \text{ for } 21 \leq i \leq 210, \quad \delta \geq 0,$$

   where $\bar{w}$ is the average of the columns of W (geometrically, this is the vertex centroid of the convex hull of the columns of W). This means that we move the columns of M toward

the outside of the convex hull of the columns of W. Hence, for any $\delta > 0$, the columns of M' are not contained in the convex hull of the columns of W (although the rank of M' remains equal to 20).

(d) *Dirichlet and Gaussian Distributions.* We set $H = [I_{20}, I_{20}, H'] \in \mathbb{R}^{20\times 240}$, where the columns of H' are generated following a Dirichlet distribution whose r parameters are chosen uniformly in $[0, 1]$ (the Dirichlet distribution generates vectors h'_i on the boundary of Δ^r, that is, $\sum_k h'_i(k) = 1\ \forall i$). We perturb each entry of M independently using the normal distribution:

$$n_i(k) \sim \delta\,\mathcal{N}(0,1) \text{ for } 1 \le i \le 240, 1 \le k \le 200.$$

(The expected value of the ℓ_2-norm of the columns of N is $\sqrt{m}\delta$, the square root of the expected value of a Chi-squared distribution.) Notice that each column of W is present twice as a column of M (in terms of hyperspectral unmixing, this means that there are two pure pixels per endmember).

Finally, we construct the noisy separable matrix $M' = WH + N$ in four different ways, see Table 2, where W, H and N are generated as described above for a total of four experiments. For each

Table 2: Generation of the noisy separable matrices for the different experiments and average value of $\kappa(W)$, $K(W)$, and $\sigma_r(W)$.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
W	(a)	(a)	(b)	(b)
N and H	(c)	(d)	(c)	(d)
$\kappa(W)$	10.84	10.84	1000	1000
$K(W)$	8.64	8.64	0.41	0.41
$\sigma_r(W)$	2.95	2.95	10^{-3}	10^{-3}
Average $(\max_i \|\|n_i\|\|_2\ \delta^{-1})$	3.05	16.15	0.29	16.15

experiment, we generate 100 matrices for 100 different values of δ and compute the percentage of columns of W that the algorithms were able to identify (hence the higher the curve, the better); see Figure 1. (Note that we then have 10000 matrices generated for each experiment.) We observe the following

- Algorithm 1 is the most robust algorithm as it is able to identify all the columns of W for the largest values of the perturbation δ for all experiments; see Table 3 and Figure 1.

Table 3: Robustness: maximum values of δ for perfect recovery.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
Algorithm 1	0.252	0.238	0.011	1.74^*10^{-4}
PPI	0.175	0	0	0
VCA	0	0.210	0	0
SiVM	0.126	0.224	/	/

(The sign / means that the algorithm failed even in the noiseless case.)

- In Exp. 1, PPI and SiVM perform relatively well, the reason being that the matrix W is well-conditioned (see Table 2) while VCA is not robust to any noise. In fact, as explained in Section 5.1, VCA only uses one randomly generated linear function to identify a column of W at

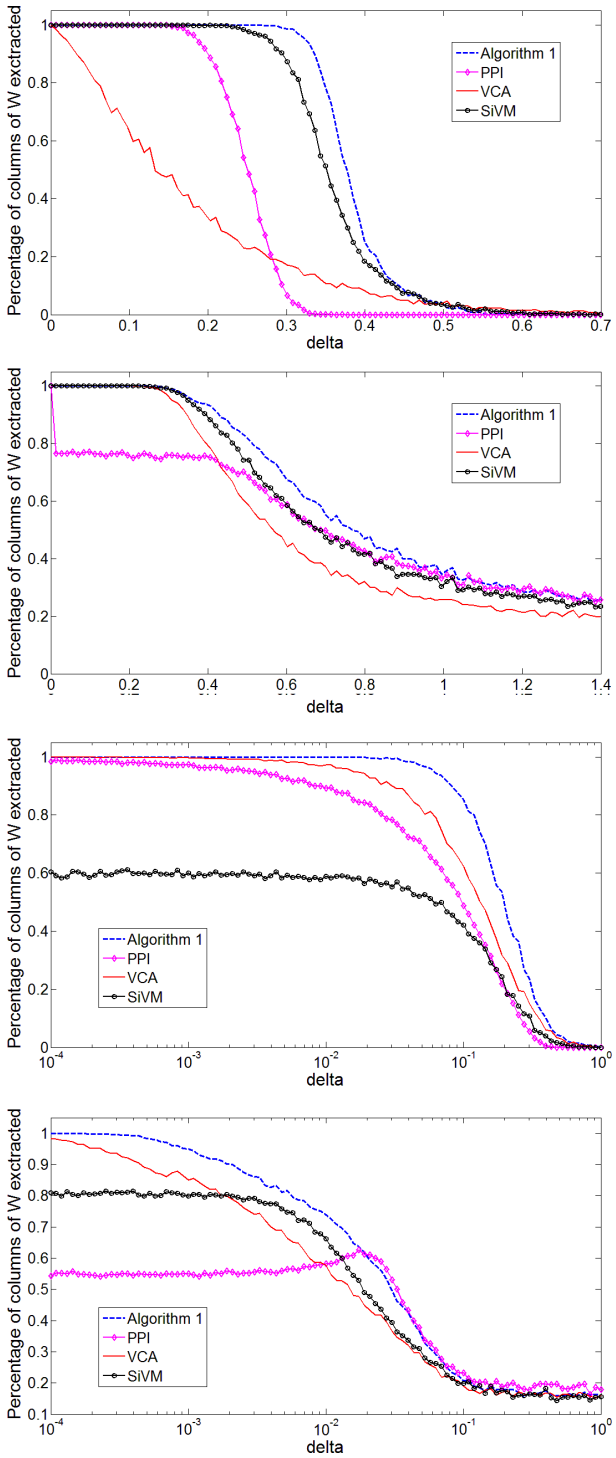

Figure 1: Comparison of Algorithm 1, PPI, VCA and SiVM. From top to bottom: Exp. 1, Exp. 2, Exp. 3, and Exp. 4.

each step, hence can potentially extract any column of M since they all are vertices of the convex hull of the columns of M (the last columns of M are the middle points of the columns of W and are perturbed toward the outside of the convex hull of the columns of W).

- In Exp. 2, the matrix W is well-conditioned so that SiVM still performs well. PPI is now unable

to identify all columns of M, because of the repetition in the data set (each column of W is present twice as a column of M)[4]. VCA now performs much better because the columns of M are strictly contained in the interior of the convex hull of the columns of W.

- In Exp. 3 and 4, SiVM performs very poorly because of the ill-conditioning of matrix W.
- In Exp. 3, as opposed to Exp. 1, PPI is no longer robust because of ill-conditioning, although more than 97% of the columns of W are perfectly extracted for all $\delta \leq 10^{-3}$. VCA is not robust but performs better than PPI, and extracts more than 97% of the columns of W for all $\delta \leq 10^{-2}$ (note that Algorithm 1 does for $\delta \leq 0.05$).
- In Exp. 4, PPI is not able to identify all the columns of W because of the repetition, while, as opposed to Exp. 2, VCA is not robust to any noise because of ill-conditioning.
- Algorithm 1 is the fastest algorithm although PPI and SiVM have roughly the same computational time. VCA is slower as it uses PCA as a preprocessing; see Table 4.

Table 4: Average computational time (s.) for the different algorithms.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
Algorithm 1	0.0080	0.0087	0.0086	0.0086
PPI	0.052	0.051	0.049	0.054
VCA	2.69	0.24	2.71	1.41
SiVM	0.025	0.027	0.028	0.027

These experiments also show that the error bound derived in Theorem 3 is rather loose, which can be partly explained by the fact that our analysis considers the worst-case scenario (while our experiments use either a structured noise or Gaussian noise). Recall that the value of ϵ in Theorem 3 is the smallest value such that $||n_i||_2 \leq \epsilon$ for all i; see Equation (9). Table 2 gives the average value of the maximum norm of the columns of N for each experiment. Based on these values, the first row of Table 5 shows the average upper bound for δ to guarantee recovery; see Theorem 3.

Table 5: Comparison of the average value of δ predicted by Theorem 3 to guarantee recovery compared to the observed values.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
Th. 3	$3.7^{*}10^{-5}$	$7^{*}10^{-6}$	$6.7^{*}10^{-12}$	$1.2^{*}10^{-13}$
Observed	0.259	0.238	0.011	$1.74^{*}10^{-4}$

5.2 Comparison with the Algorithm of Bittorf et al. [6]

In this section, we compare the Algorithm of Bittorf et al. (BRRT) (Algorithm 3 in [6]; see also Algorithm 2 in [18])[5] with Algorithm 1. BRRT has to solve a linear program with $\mathcal{O}(n^2)$ variables which we solve using CVX [20]. Therefore we are only able to solve small-scale problems (in fact, CVX

[4]For $\delta = 0$, we observe that our implementation of the PPI algorithm actually recovers all columns of W. The reason is that the first columns of M are exactly equal to each other and that the MATLAB *max(.)* function only outputs the smallest index corresponding to a maximum value. This is why PPI works in the noiseless case even when there are duplicates.

[5]We do not perform a comparison with the algorithm of Arora et al. [3] as it is not very practical (the value of α has to be estimated, see Section 2.4) and has already been shown to perform similarly as BRRT in [6].

uses an interior-point method): we perform exactly the same experiments as in the previous section but for $m = 10$ and $r = 5$ for all experiments, so that

- $n = 5 + \binom{5}{2} = 15$ for the first and third experiments (the last ten columns of M are the middle points of the five columns of W).
- $n = 5 + 5 + 10 = 20$ for the second and fourth experiments (we repeat twice each endmember, and add 10 points in the convex hull of the columns of W).

The average running time for BRRT on these data sets using CVX [20] is about two seconds while, for Algorithm 1, it is less than 10^{-3} seconds. (Bittorf et al. [6] propose a more efficient solver than CVX for their LP instances. As mentioned in Section 2.4, even with their more efficient solver, Algorithm 1 is much faster for large n.) Figure 2 shows the percentage of correctly extracted columns with respect to δ, while Table 6 shows the robustness of both algorithms.

Table 6: Robustness: maximum values of δ for perfect recovery.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
Algorithm 1	**0.070**	$1.4^{*}10^{-6}$	$7.9^{*}10^{-4}$	$\mathbf{2.4^{*}10^{-6}}$
BRRT [6]	0.042	$\mathbf{5.5^{*}10^{-6}}$	$\mathbf{1.8^{*}10^{-3}}$	10^{-6}

Quite surprisingly, Algorithm 1 performs in average better than BRRT. Although BRRT is more robust in two of the four experiments (that is, it extracts correctly all columns of W for a larger value of δ), the percentage of columns it is able to correctly extract decreases much faster as the noise level increases. For example, Table 7 shows the maximum value of δ for which 99% percent of the columns of W are correctly extracted. In that case, Algorithm 1 always performs better. A possible

Table 7: Maximum values of δ for recovering 99% of the columns of W.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
Algorithm 1	0.126	$2.8^{*}10^{-3}$	$1.2^{*}10^{-2}$	10^{-4}
BRRT [6]	0.05	$4.0^{*}10^{-4}$	$2.2^{*}10^{-3}$	$1.8^{*}10^{-5}$

explanation for this behavior is that, when the noise is too large, the condition for recovery are not satisfied as the input matrix is far from being separable. However, using Algorithm 1 still makes sense as it extracts columns whose convex hull has large volume [9, 10] while it is not clear what BRRT does in that situation (as it heavily relies on the separability assumption). Therefore, although BRRT guarantees perfect recovery for higher noise levels, it appears that, in practice, when the noise level is high, Algorithm 1 is preferable.

6 Conclusion and Further Work

In this paper, we have introduced and analyzed a new family of fast and robust recursive algorithms for separable NMF problems which are equivalent to hyperspectral unmixing problems under the linear mixing model and the pure-pixel assumption. This family generalizes several existing hyperspectral unmixing algorithms, and our analysis provides a theoretical framework to explain the better performances of these approaches. In particular, our analysis explains why algorithms like PPI and VCA are less robust against noise compared to Algorithm 1.

Many questions remain open, and would be interesting directions for further research:

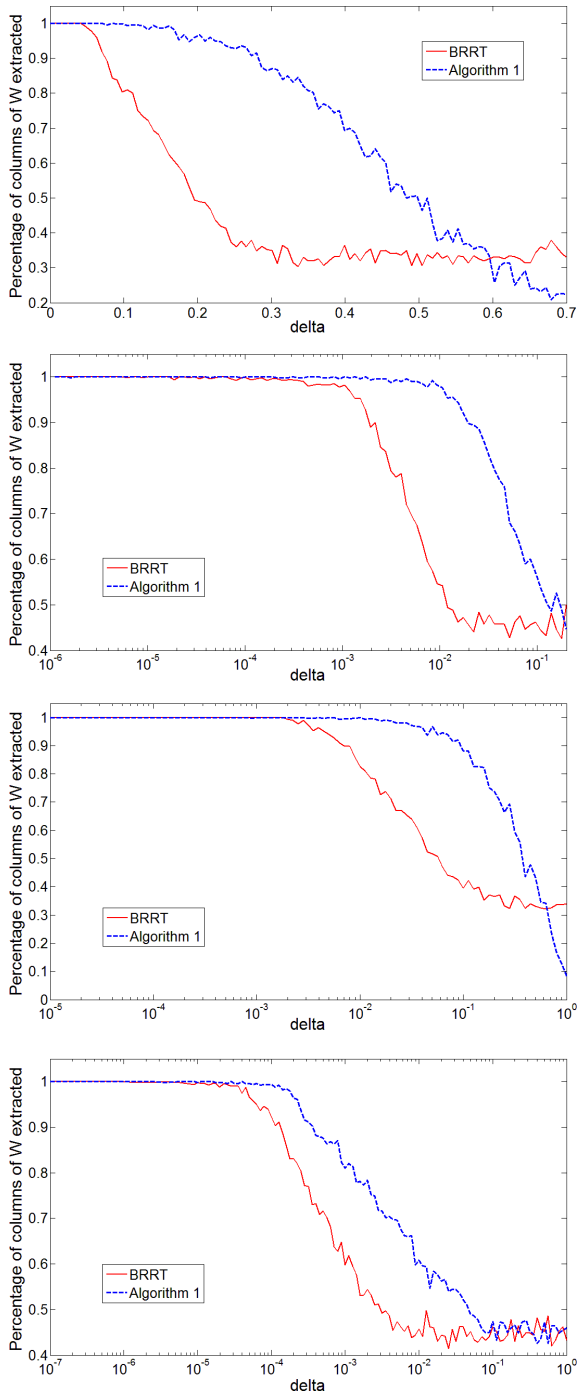

Figure 2: Comparison of Algorithm 1 and BRRT [6]. From top to bottom: Exp. 1, Exp. 2, Exp. 3, and Exp. 4.

- Is it possible to provide better error bounds for Algorithm 1 than the ones of Theorem 3? In other words, is our analysis tight? Also, can we improve the bounds if we assume specific generative and/or noise models?
- How can we choose appropriate functions $f(x)$ for Algorithm 1 depending on the input data matrix?

- Can we design other robust and fast algorithms for the separable NMF problem leading to better error bounds?

Acknowledgments

The authors would like to thank the reviewers for their feedback which helped improve the paper significantly.

References

[1] Ambikapathi, A., Chan, T.H., Chi, C.Y., Keizer, K.: Two effective and computationally efficient pure-pixel based algorithms for hyperspectral endmember extraction. In: IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 1369–1372 (2011)

[2] Araújo, U., Saldanha, B., Galvão, R., Yoneyama, T., Chame, H., Visani, V.: The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. Chemometrics and Intelligent Laboratory Systems **57**(2), 65–73 (2001)

[3] Arora, S., Ge, R., Kannan, R., Moitra, A.: Computing a nonnegative matrix factorization – provably. In: Proceedings of the 44th symposium on Theory of Computing, STOC '12, pp. 145–162. ACM, New York, NY, USA (2012)

[4] Arora, S., Ge, R., Moitra, A.: Learning topic models - going beyond SVD. In: Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS '12, pp. 1–10 (2012)

[5] Bioucas-Dias, J., Plaza, A., Dobigeon, N., Parente, M., Du, Q., Gader, P., Chanussot, J.: Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **5**(2), 354–379 (2012)

[6] Bittorf, V., Recht, B., Ré, E., Tropp, J.: Factoring nonnegative matrices with linear programs. In: Advances in Neural Information Processing Systems, NIPS '12, pp. 1223–1231 (2012)

[7] Boardman, J.: Geometric mixture analysis of imaging spectrometry data. In: IEEE Int. Geoscience and Remote Sensing Symposium, vol. 4, pp. 2369–2371 (1994)

[8] Businger, P., Golub, G.: Linear least squares solutions by Householder transformations. Numerische Mathematik **7**, 269–276 (1965)

[9] Çivril, A., Magdon-Ismail, M.: On selecting a maximum volume sub-matrix of a matrix and related problems. Theoretical Computer Science **410**(47-49), 4801–4811 (2009)

[10] Çivril, A., Magdon-Ismail, M.: Exponential inapproximability of selecting a maximum volume sub-matrix. Algorithmica (2011). Doi:10.1007/s00453-011-9582-6

[11] Chan, T.H., Ma, W.K., Ambikapathi, A., Chi, C.Y.: A simplex volume maximization framework for hyperspectral endmember extraction. IEEE Trans. on Geoscience and Remote Sensing **49**(11), 4177–4193 (2011)

[12] Chang, C.I., Wu, C.C., Liu, W.M., Ouyang, Y.C.: A new growing method for simplex-based endmember extraction algorithm. IEEE Trans. on Geoscience and Remote Sensing **44**(10), 2804–2819 (2006)

[13] Craig, M.: Minimum-volume tranforms for remotely sensed data. IEEE Trans. on Geoscience and Remote Sensing **32**(3), 542–552 (1994)

[14] Donoho, D., Stodden, V.: When does non-negative matrix factorization give a correct decomposition into parts? In: Advances in Neural Information Processing 16 (2003)

[15] Elhamifar, E., Sapiro, G., Vidal, R.: See all by looking at a few: Sparse modeling for finding representative objects. In: IEEE Conference on Computer Vision and Pattern Recognition (2012)

[16] Esser, E., Moller, M., Osher, S., Sapiro, G., Xin, J.: A convex model for nonnegative matrix factorization and dimensionality reduction on physical space. IEEE Transactions on Image Processing **21**(7), 3239–3252 (2012)

[17] Gillis, N.: Sparse and unique nonnegative matrix factorization through data preprocessing. Journal of Machine Learning Research **13**(Nov), 3349–3386 (2012)

[18] Gillis, N.: Robustness analysis of hottopixx, a linear programming model for factoring nonnegative matrices. SIAM J. Mat. Anal. Appl. **34**(3), 1189–1212 (2013)

[19] Golub, G., Van Loan, C.: Matrix Computation, 3rd Edition. The Johns Hopkins University Press Baltimore (1996)

[20] Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx/` (2011)

[21] Hiriart-Urruty, J.B., Lemaréchal, C.: Fundamentals of Convex Analysis. Springer, Berlin (2001)

[22] Juditsky, A., Nemirovski, A.: Large Deviations of Vector-valued Martingales in 2-Smooth Normed Spaces (2008). ArXiv:0809.0813v1

[23] Nascimento, J., Bioucas-Dias, J.: Vertex component analysis: a fast algorithm to unmix hyperspectral data. IEEE Trans. on Geoscience and Remote Sensing **43**(4), 898–910 (2005)

[24] Ren, H., Chang, C.I.: Automatic spectral target recognition in hyperspectral imagery. IEEE Trans. on Aerospace and Electronic Systems **39**(4), 1232–1249 (2003)

[25] Sun, Y., Xin, J.: Underdetermined sparse blind source separation of nonnegative and partially overlapped data. SIAM Journal on Scientific Computing **33**(4), 2063–2094 (2011)

[26] Thurau, C., Kersting, K., Wahabzada, M., Bauckhage, C.: Descriptive matrix factorization for sustainability adopting the principle of opposites. Data Mining and Knowledge Discovery **24**, 325–354 (2012)

[27] Vavasis, S.: On the complexity of nonnegative matrix factorization. SIAM J. on Optimization **20**(3), 1364–1377 (2009)

[28] Winter, M.: N-findr: an algorithm for fast autonomous spectral end-member determination in hyperspectral data. In: Proc. SPIE Conference on Imaging Spectrometry V (1999)

[29] Wu, C.C., Liu, W., Ren, H., Chang, C.I.: A comparative study and analysis between vertex component analysis and orthogonal subspace projection for endmember extraction. p. 656523. SPIE (2007)

[30] Zhang, J., Rivard, B., Rogge, D.: The successive projection algorithm (SPA), an algorithm with a spatial constraint for the automatic search of endmembers in hyperspectral data. Sensors **8**(2), 1321–1342 (2008)