# A Fast Gradient Method for Nonnegative Sparse Regression With Self-Dictionary

Nicolas Gillis and Robert Luce

***Abstract*— A nonnegative matrix factorization (NMF) can be computed efficiently under the *separability assumption*, which asserts that all the columns of the given input data matrix belong to the cone generated by a (small) subset of them. The provably most robust methods to identify these *conic basis columns* are based on nonnegative sparse regression and self-dictionaries, and require the solution of large-scale convex optimization problems. In this paper, we study a particular nonnegative sparse regression model with self-dictionary. As opposed to previously proposed models, this model yields a smooth optimization problem, where the sparsity is enforced through linear constraints. We show that the Euclidean projection on the polyhedron defined by these constraints can be computed efficiently, and propose a fast gradient method to solve our model. We compare our algorithm with several state-of-the-art methods on synthetic data sets and real-world hyperspectral images.**

***Index Terms*— Nonnegative matrix factorization, separability, sparse regression, self dictionary, fast gradient, hyperspectral imaging, pure-pixel assumption.**

## I. Introduction

GIVEN a matrix $M \in \mathbb{R}^{m,n}$ where each column of $M$ represents a point in a data set, we assume in this paper that each data point can be well approximated using a nonnegative linear combination of a small subset of the data points. More precisely, we assume that there exists a small subset $\mathcal{K} \subset \{1, 2, \ldots, n\}$ of $r$ column indices and a nonnegative matrix $H \in \mathbb{R}^{r,n}_+$ such that

$$M \approx M(:, \mathcal{K})H.$$

If $M$ is nonnegative, this problem is closely related to nonnegative matrix factorization (NMF) which aims at decomposing $M$ as the product of two nonnegative matrices $W \in \mathbb{R}^{m,r}_+$ and $H \in \mathbb{R}^{r,n}_+$ with $r \ll \min(m, n)$ such that $M \approx WH$ [1]. In the NMF literature, the assumption above is referred to as the *separability assumption* [2], and the aim is therefore to finding a particular NMF with $W = M(:, \mathcal{K})$.

There are several applications to solving near-separable NMF, e.g., blind hyperspectral unmixing [3], [4], topic modeling and document classification [5], [6], video summarization and image classification [7], and blind source separation [8]–[10].

### A. Self Dictionary and Sparse Regression Based Approaches

Many algorithms have been proposed recently to solve the near-separable NMF problem; see [11] and the references therein. In hyperspectral unmixing, the most widely used algorithms sequentially identify important columns of $M$, such as vertex component analysis (VCA) [12] or the successive projection algorithm (SPA) [13], [14]; see section IV for more details. Another important class of algorithms for identifying a good subset $\mathcal{K}$ of the columns of $M$ is based on sparse regression and self dictionaries. These algorithms are computationally more expensive but have the advantage to consider the selection of the indices in $\mathcal{K}$ at once leading to the most robust algorithms; see the discussion [11].

An exact model for nonnegative sparse regression with self dictionary [7], [15] is

$$\min_{X \in \mathbb{R}^{n,n}_+} \|X\|_{\text{row},0} \quad \text{such that } \|M - MX\| \leq \epsilon, \tag{1}$$

where $\|X\|_{\text{row},0}$ equals to the number of nonzero rows of $X$, and $\epsilon$ denotes the noise level of the data $M$. Here, the norm in which the residual $M - MX$ is measured should be chosen in dependence of the noise model. It can be checked that there is a nonnegative matrix $X$ with $r$ nonzero rows such that $M = MX$ if and only if there exists an index set $\mathcal{K}$ of cardinality $r$ and a nonnegative matrix $H$ such that $M = M(:, \mathcal{K})H$: The index set $\mathcal{K}$ corresponds to the indices of the nonzero rows of $X$, and hence we have $H = X(\mathcal{K}, :)$; see [16, Sec. 3] or [15, Sec. I-B] for details.

In [7] and [15], the difficult problem (1) is relaxed to the convex optimization problem

$$\min_{X \in \mathbb{R}^{n,n}_+} \|X\|_{1,q} \quad \text{s.t. } \|M - MX\| \leq \epsilon \quad \text{and } X \leq 1, \tag{2}$$

where $\|X\|_{1,q} := \sum_{i=1}^{n} \|X(i, :)\|_q$. In [15], $q = +\infty$ is used while, in [7], $q = 2$ is used. The quantity $\|X\|_{1,q}$ is the $\ell_1$-norm of the vector containing the $\ell_q$ norms of the rows of $X$. Because the $\ell_1$ norm promotes sparsity, this model is expected to generate a matrix $X$ with only a few nonzero

Manuscript received October 5, 2016; revised June 8, 2017 and August 16, 2017; accepted September 1, 2017. Date of publication September 18, 2017; date of current version October 17, 2017. The work of N. Gillis was supported in part by the F.R.S.-FNRS Incentive Grant for Scientific Research nº F.4501.16, and in part by the ERC Starting Grant nº 579515. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Christos Bouganis. *(Corresponding author: Robert Luce.)*

N. Gillis is with the Department of Mathematics and Operational Research, Faculté Polytechnique, Université de Mons, 7000 Mons, Belgium (e-mail: nicolas.gillis@umons.ac.be).

R. Luce is with the École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland (e-mail: robert.luce@epfl.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2017.2753400

1057-7149 © 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

rows. The reason is that the $\ell_1$ norm is a good surrogate for the $\ell_0$ norm on the $\ell_\infty$ ball. In fact, the $\ell_1$ norm is the convex envelope of the $\ell_0$ norm on the $\ell_\infty$ ball, that is, the $\ell_1$ norm is the largest convex function smaller than the $\ell_0$ norm on the $\ell_\infty$ ball; see [17]. Hence for $q = +\infty$ and $X \leq 1$, we have $\|X\|_{1,\infty} \leq \|X\|_{\text{row},0}$ so that (2) provides a lower bound for (1). In practice, the constraint $X \leq 1$ is often satisfied; for example, in hyperspectral imaging, the entries of $X$ represent abundances which are smaller than one. If this assumption is not satisfied and the input matrix is nonnegative, it can be normalized so that the entries of the columns of matrix $H$ (hence $X$) are at most one, as suggested for example in [15]. This can be achieved by normalizing each column of $M$ so that its entries sum to one. After such a normalization, we have for all $j$

$$1 = \|M(:,j)\|_1 = \|MX(:,j)\|_1 = \|\sum_k M(:,k)X(k,j)\|_1$$
$$= \sum_{k\in\mathcal{K}} X(k,j)\|M(:,k)\|_1 = \|X(:,j)\|_1,$$

since $M$ and $X$ are nonnegative.

The model (2) was originally proved to be robust to noise, but only at the limit, that is, only for $\epsilon \to 0$, and assuming no columns of $M(:,\mathcal{K})$ are repeated in the data set [15]. If a column of $M(:,\mathcal{K})$ is present twice in the data set, the (convex) models cannot discriminate between them and might assign a weight on both columns. (The situation is worsened in the presence of more (near) duplicates, which is typical in hyperspectral image data, for example). More recently, Fu and Ma [18] improved the robustness analysis of the model for $q = +\infty$ (in the absence of duplicates).

Another sparse regression model proposed in [16] and later improved in [19] is the following:

$$\min_{X\in\mathbb{R}^{n,n}_+} \text{trace}(X) \quad \text{s.t.} \quad \begin{matrix}\|M - MX\| \leq \epsilon \\ X(i,j) \leq X(i,i) \leq 1 \ \forall i,j.\end{matrix} \tag{3}$$

(The model can easily be generalized for non-normalized $M$; see model (4)). Here sparsity is enforced by minimizing the $\ell_1$ norm of the diagonal of $X$ as $\text{trace}(X) = \|\text{diag}(X)\|_1$ for $X \geq 0$, while no off-diagonal entry of $X$ can be larger than the diagonal entry in its row. Hence $\text{diag}(X)$ is sparse if and only if $X$ is row sparse.

The model (3) is, to the best of our knowledge, the provably most robust for near-separable NMF [19]. In particular, as opposed to most near-separable NMF algorithms that require $M(:,\mathcal{K})$ to be full column rank, it only requires the necessary condition that no column of $M(:,\mathcal{K})$ is contained in the convex hull of the other columns. More precisely, let us define the *conical robustness* of a matrix $W \in \mathbb{R}^{m,r}$ as

$$\kappa = \min_{1\leq k\leq r} \min_{x\in\mathbb{R}^{r-1}_+} \|W(:,k) - W(:,\{1,\ldots,r\}\setminus\{k\})x\|_1.$$

We then say that $W$ is *$\kappa$-robustly conical*, and the following recovery result can be obtained:

*Theorem 1 [19, Th. 7]: Let $M = M(:,\mathcal{K})H$ be a separable matrix with $M(:,\mathcal{K})$ being $\kappa$-robustly conical and the entries of each column of $H$ summing to at most one, and let $\tilde{M} = M + N$. If $\epsilon := \max_{1\leq j\leq n}\|N(:,j)\|_1 \leq \mathcal{O}\left(\frac{\kappa}{r}\right)$, then the model* (3) *allows to recover the columns of $M(:,\mathcal{K})$ up to error $\mathcal{O}\left(r\frac{\epsilon}{\kappa}\right)$.*

### B. Contribution and Outline of the Paper

In the work [19] a robustness analysis of the model (3) was given, which we here relate to the robustness of the model (2). We also present a practical and efficient first-order optimization method for (3) (in [19] no such method was given). More precisely, our contribution in this work is threefold:

- In section II, we prove that both sparse regression models (2) and (3) are equivalent. This significantly improves the theoretical robustness analysis of (2), as the results for (3) directly apply to (2).
- In section III, we introduce a new model, very similar to (3) (using the Frobenius norm, and not assuming normalization of the input data), for which we propose an optimal first-order method: the key contribution is a very efficient and non-trivial projection onto the feasible set. Although our approach still requires $\mathcal{O}(mn^2)$ operations per iteration, it can solve larger instances of (3) than commercial solvers, with $n \sim 1000$. We show the effectiveness of our approach on synthetic data sets in section IV-A.
- In section IV-B, we preselect a subset of the columns of the input matrix and scale them appropriately (depending on their importance in the data set) so that we can apply our method meaningfully to real-world hyperspectral images when $n \sim 10^6$. We show that our approach outperforms state-of-the-art pure pixel search algorithms.

## II. Equivalence Between Sparse Regression Models (2) and (3)

We now prove the equivalence between the models (2) and (3). We believe it is an important result because, as far as we know, both models have been treated completely independently in the literature, and, as explained in the Introduction, while model (2) is more popular [7], [15], [18], stronger theoretical guarantees were provided for model (3) [19].

*Theorem 2: Let $\|\cdot\|$ be a column wise matrix norm, that is, $\|A\| = \sum_i \alpha_i \|A(:,i)\|_c$ for some $\alpha_i > 0$ and some vector norm $\|\cdot\|_c$. Then* (2) *is equivalent to* (3) *in the following sense:*

- *At optimality, both objective functions coincide,*
- *any optimal solution of* (3) *is an optimal solution of* (2)*, and*
- *any optimal solution of* (2) *can be trivially transformed into an optimal solution of* (3)*.*

*Proof:* See Appendix A. □

Theorem 2 implies that any robustness result for (2) applies to (3), and vice versa. It is therefore meaningful to compare the robustness results of [18] and [19]. It turns out that the results in [19] are stronger because, as opposed to Fu and Ma [18], it does not require the absence of duplicated columns (which is a rather strong assumption); see Theorem 1. However, it is interesting to note that, in the absence of duplicated columns, both robustness results essentially coincide (the error bounds are the same up to some constant multiplicative factors),

**Algorithm 1** Fast Gradient Method for Nonnegative Sparse Regression With Self Dictionary (FGNSR)

**Require:** A matrix $M \in \mathbb{R}^{m,n}$, number $r$ of columns to extract, a vector $p \in \mathbb{R}^n_{++}$ whose entries are close to 1, a penalty parameter $\mu$, and maximum number of iterations `maxiter`.

**Ensure:** An set $\mathcal{K} \subset \{1,\dots,n\}$ of column indices such that $\min_{H \in \mathbb{R}^{r,n}_+} \|M - M(:,\mathcal{K})H\|_F$ is small.

1: {Initialization}
2: $\alpha_0 \leftarrow 0.05$; $Y \leftarrow 0_{n,n}$; $X \leftarrow Y$; $L \leftarrow \sigma_{\max}(M)^2$;
3: **for** $k = 1 : \texttt{maxiter}$ **do**
4: $\quad Y_p \leftarrow Y$;
5: $\quad \nabla F(X) \leftarrow M^T MX - M^T M + \mu\,\mathrm{diag}(p)$;
6: $\quad$ {Projection on $\Omega$; see Section III-D}
7: $\quad Y \leftarrow \mathcal{P}_\Omega\left(X - \frac{1}{L}\nabla F(X)\right)$;
8: $\quad X \leftarrow Y + \beta_k(Y - Y_p)$, where $\beta_k = \frac{\alpha_{k-1}(1-\alpha_{k-1})}{\alpha^2_{k-1}+\alpha_k}$ such that $\alpha_k \geq 0$ and $\alpha_k^2 = (1-\alpha_k)\alpha^2_{k-1}$;
9: **end for**
10: $\mathcal{K} \leftarrow$ postprocess($X$,$r$); {The simplest way is to pick the $r$ largest entries of $\mathrm{diag}(X)$ as done in [16]. In the presence of (near-)duplicated columns of $M$, one should use more sophisticated strategies [19].}

namely [18, Th. 1] and [19, Th. 2]. We provide this result here for completeness:

*Theorem 3 [19, Th. 2]: Let $M = M(:,\mathcal{K})H$ where $M(:,\mathcal{K})$ is $\kappa$-robustly conical and where the entries of each column of $H$ are at most one. Let also $\tilde{M} = M+N$, and $H(i,j) \leq \beta < 1$ for all $1 \leq i \leq m$ and $j \notin \mathcal{K}$ (this is the condition that there is no duplicates of the columns of $M(:,\mathcal{K})$). If $\epsilon := \max_{1\leq j\leq n}\|N(:,j)\|_1 < \frac{\kappa(1-\beta)}{20}$, then the model (3) allows to recover the columns of $M(:,\mathcal{K})$ up to error $\epsilon$.*

The advantage of the formulation (3) over (2) is that the objective function is smooth. Moreover, as we will show in Section III-D, projecting onto the feasible set can be made efficiently (even when the model is generalized to the case where the columns of the input matrix are not normalized). Hence, we will be able to apply an optimal first-order method of smooth convex optimization.

## III. Convex Model Without Normalization and Fast Gradient Method

In this section, in order to be able to obtain a more practical model that can be optimized using techniques from smooth convex optimization, we derive a new model, namely (6), closely related to (3) but where

- the assumption $X \leq 1$ is not necessary,
- $||.||$ is the Frobenius norm, which is smooth and arguably the most popular choice in practice, and
- Lagrangian duality is used to incorporate the error term $||M - MX||_F^2$ in the objective function.

Then, we apply a fast gradient method on (6) (Algorithm 1), after having listed related algorithmic approaches to tackle similar optimization problems.

### A. Avoiding Column Normalization

The model (3) can be generalized in case $X \not\leq 1$, where $M = MX$, without column normalization of $M$. The advantage is twofold: column normalization (i) is only possible for nonnegative input matrix, and (ii) may introduce distortion in the data set as it would be equivalent to consider that the noise added to each data point (that is, each column of $M$) is proportional to it [5]. If one wants to consider absolute error (the norm of each column of the noise is independent on the norm of the input data), then the input matrix should not be normalized and the following model should be considered [19]:

$$\min_{X\in\Omega} \ \mathrm{trace}(X) \quad \text{such that } ||M - MX||_F \leq \epsilon. \tag{4}$$

The set $\Omega$ is defined as

$$\Omega := \{X \in \mathbb{R}^{n,n}_+ \mid X_{ii} \leq 1, w_i X_{ij} \leq w_j X_{ii} \forall i,j\}, \tag{5}$$

where the vector $w \in \mathbb{R}^n_+$ are the column $\ell_1$ norms of $M$, that is, $w_j = \|M(:,j)\|_1$ for all $j$. The upper bounds $X_{ij} \leq \frac{w_j}{w_i}X_{ii}$ come from the fact that each weight used to reconstruct a data point inside the convex cone generated by some extreme rays cannot exceed the ratio of the $\ell_1$ norm of that data point to each individual extreme ray.

Note that the optimization problem (4) is convex, and it can be solved as a second-order conic program (SOCP) in $n^2$ variables. This large number of variables even for moderate values of $n$ rules out the use of off-the-shelf SOCP optimization software. In this section, we describe an optimal first-order method to solve (4). A main contribution is in the (non-trivial) projection onto the feasible set $\Omega$.

### B. Related Work

To solve a model similar to (3), Bittorf et al. [16] used a stochastic subgradient descent method, with a non-smooth objective function (they were using the component-wise $\ell_1$ norm of $M - MX$). Although the cost per iteration is relatively low with $\mathcal{O}(n^2)$ operations per iterations, the convergence is quite slow.

To solve (2) with $q = +\infty$ in [15] and $q = 2$ in [7], authors propose an alternating direction method of multipliers (ADMM). However, ADMM is not an optimal first-order method as the objective function converges at rate $\mathcal{O}(1/k)$ vs. $\mathcal{O}(1/k^2)$ for optimal first-order methods, where $k$ is the iteration number. Moreover, the cost per iteration of ADMM is larger as it requires the introduction of new variables (one variable $Y$ of the same dimension as $X$, Lagrangian multipliers and a parameter which is not always easy to tune).

Another optimal first-order method was proposed in [20]. However, it solves a rather different optimization problem, namely

$$\min_{X,Q} p^T\,\mathrm{diag}(X) + \beta\|MX - M - Q\|_F^2 + \lambda\|Q\|_1,$$

where two regularization parameters have to be tuned while the objective function is non-smooth, so that authors use a local linear approximation approach to smooth the objective function. They also require column normalization while our

approach does not. Also, they point out that it would be good to incorporate the constraints from (3), which we do in this paper by developing an effective projection on the feasible set.

### C. Fast Gradient Method for (4)

It is possible to solve (4) using commercial solvers that are usually based on interior-point methods, such as Gurobi. However, it is computationally rather expensive as there are $\mathcal{O}(n^2)$ variables, where $n$ is the number of columns of $M$.

Moreover, it has to be noted that in the separable NMF case, it is not crucial to obtain high accuracy solutions: the main information one wants to obtain is which columns of $M$ are the important ones. Hence it is particularly meaningful in this context to use first-order methods (slower convergence but much lower computational cost per iteration).

The additional constraints that allows to take into account the fact that the columns of $M$ are not normalized makes the feasible set more complicated, but we develop an efficient projection method, that allows us to design an optimal first-order method (namely, a fast gradient method).

The problem we propose to solve is

$$\min_{X\in\Omega} F(X) = \frac{1}{2}\|M - MX\|_F^2 + \mu p^T \operatorname{diag}(X), \qquad (6)$$

where $M \in \mathbb{R}^{m,n}$ is the input data matrix, and $X \in \Omega$ are the basis reconstruction coefficients. Note that we have replaced $\operatorname{trace}(X)$ with the more general term $p^T \operatorname{diag}(X)$ (they coincide if $p$ is the vector of all ones). The reason is twofold: (1) it makes the model more general, and (2) it was shown in [16] that using such a vector $p$ (e.g., randomly chosen with its entries close to one) allows to discriminates between (approximate) duplicate basis vectors present in the data. The penalty parameter $\mu \in \mathbb{R}_+$ acts as a Lagrange multiplier. From duality theory, there exists $\mu$ (which depends on the data $M$), such that models (4) and (6) are equivalent [21] (given that $p$ is the vector of all ones).

Algorithm 1 is a first-order method for minimizing $F(X)$ over $\Omega$, based on Nesterov's fast gradient method [22]. Here "fast" refers to the fact that it attains the best possible convergence rate of $\mathcal{O}(1/k^2)$ in the first-order regime. Because $M$ is not necessarily full rank (in particular $\operatorname{rank}(M) \leq r$ when $M$ is a $r$-separable matrix without noise), the objective function of (6) is not necessarily strongly convex. However, its gradient is Lipschitz continuous with constant $L = \lambda_{\max}(M^T M) = \sigma_{\max}(M)^2$, which is sufficent to guarantee the claimed convergence rate.

The requested number of columns $r$ in Algorithm 1 is used only in the postprocessing step (line 10). Hence upon termination the obtained matrix $X$ can be used to extract multiple NMFs, corresponding to different ranks, and to pick the most appropriate one among them for the application at hand.

The penalty parameter $\mu$ in (6) is crucial as it balances the importance between the approximation error $\|M - MX\|_F^2$ and the fact that we want the diagonal of $X$ to be as sparse as possible. On one hand, if $\mu$ is too large, then the term $\|M - MX\|_F^2$ will not have much importance in the objective function leading to a poor approximation. On the other hand, if $\mu$ is too small, then $\|M - MX\|_F^2$ will have to be very small and $X$ will be close to the identity matrix. However, in our experience, it seems that the output of Algorithm 1 is not too sensitive to this scaling. The main a reason is that only the largest entries of (the diagonal of) $X$ will be extracted by the post-processing procedure while the most representative columns of $M$ remains the same independently of the value of $\mu$. In other words, increasing $\mu$ will have the effect of increasing in average the entries of $X$ but the rows corresponding to the important columns of $M$ will continue having larger entries. Therefore the extracted index set $\mathcal{K}$ will remain the same.

To set the value of $\mu$, we propose the following heuristic which appears to work very well in practice:

- Extract a subset $\mathcal{K}$ of $r$ columns of $M$ with the fast algorithm proposed in [14] (other fast separable NMF algorithms would also be possible);
- Compute the corresponding optimal weight $H$,

$$H = \operatorname{argmin}_{Z\in\mathbb{R}_+^{r,n}} ||M - M(:,\mathcal{K})Z||_F^2,$$

  using a few iterations of coordinate descent; see [23].
- Define $X_0(\mathcal{K},:) = H$ and $X_0(i,:) = 0$ for all $i \notin \mathcal{K}$.
- Set $\mu = \frac{\|M-MX_0\|_F^2}{p^T \operatorname{diag}(X_0)}$, to balance the importance of both terms in the objective function.

Note that if the noise level $\epsilon$, or an estimate thereof, is given as an input, $\mu$ can be easily updated in the course of the gradient iteration so that $\|M-MX\|_F \approx \epsilon$: If $\|M-MX\|_F$ is too small (large) relative to $\epsilon$ in the course of the gradient iteration, $\mu$ is simply increased (decreased), and the method is restarted. Of course these adjustments should be carried out in a convergent scheme, say, geometrically decreasing, in order to maintain convergence of Algorithm 1.

*Remark 1: Algorithm 1 can be directly generalized to any other smooth norm for which the gradient is Lipschitz continuous and can be computed efficiently.*

### D. Euclidean Projection on $\Omega$

In Algorithm 1 we need to compute the Euclidean projection of a point $X \in \mathbb{R}^{n,n}$ on the set $\Omega$ from Equation (5), denoted $\mathcal{P}_\Omega$. Recall that for a convex subset $C \in \mathbb{R}^n$ of an Euclidean vector space, a function $\phi : \mathbb{R}^n \to C$ is an Euclidean projection on $C$ if for all $x \in \mathbb{R}^n$

$$\|x - \phi(x)\| = \min_{z\in C} \|x - z\|.$$

We describe in Appendix B how to compute this projection efficiently. More precisely, we show how to solve the problem $\min_{Z\in\Omega}\|X - F\|_F$ in $\mathcal{O}(n^2 \log n)$ operations. In the unweighted case, that is, $w_j \equiv 1$ for all $1 \leq j \leq n$, our algorithm is similar to the one described in [16], but the inclusion of non-unit weights makes the details very much different. The worst case bound of $\mathcal{O}(n^2 \log n)$ operations will typically overestimate the computational cost if appropriate data structures are used. This is explained in Remark 3, Appendix B.

TABLE I
COMPLEXITY COMPARISON FOR A $m$-BY-$n$ INPUT MATRIX

| | FLOPs | Memory | Parameters | Normalization | Run time in sec. IV-A |
|---|---|---|---|---|---|
| SPA | $2mnr + \mathcal{O}(mr^2)$ | $\mathcal{O}(mn)$ | $r$ | Yes | $< 0.01$s |
| XRAY | $\mathcal{O}(mnr)$ | $\mathcal{O}(mn)$ | $r$ | No | 0.03s |
| SNPA | $\mathcal{O}(mnr)$ | $\mathcal{O}(mn)$ | $r$ | Yes | 0.10s |
| SOCP (Gurobi, IPM) | $\mathcal{O}(n^6)$ | $\mathcal{O}(n^4)$ | $\|N\|_F$ | No | 2.78s |
| FGNSR | $\mathcal{O}(mn^2)$ | $\mathcal{O}(mn + n^2)$ | $r$ or $\|N\|_F$ | No | 0.09s |

### E. Computational Cost

In order to find the asymptotic computational cost of Algorithm 1, we analyze the three main steps as follows.

Line 2: The maximum singular value of an $m$-by-$n$ matrix can be well approximated with a few steps of the power method, requiring $\mathcal{O}(mn)$ operations.

Line 5: The matrix $M^TM$ should be computed only once at a cost of $\mathcal{O}\left(mn^2\right)$ operations. If $m \geq 2n$, then computing $(M^TM)X$ requires $\mathcal{O}\left(n^3\right)$ operations. Otherwise, one should first compute $MX$ at a cost of $\mathcal{O}\left(mn^2\right)$ operations and then $M^T(MX)$ at a cost of $\mathcal{O}\left(mn^2\right)$ operations (the total being smaller than $n^3$ if $m \leq 2n$).

Line 7: The projection onto $\Omega$ of an $n$-by-$n$ matrix $X$ requires $\mathcal{O}\left(n^2 \log n\right)$ operations (the $\log n$ factor comes from the fact that we need to sort the entries of each row of $X$); see Section III-D for the details about the projection step. Note that each row of $X$ can be projected independently hence this step is easily parallelizable. Moreover, many rows of $X$ are expected to be all-zeros and their projection is trivial.

Hence the number of operations is in $\mathcal{O}(mn^2+n^2 \log n)$, and since we typically have $m \geq \log n$, this reduces to $\mathcal{O}\left(mn^2\right)$ operations.

The computational cost could potentially be decreased using random projections of the data points to reduce the dimension $m$ of the input matrix; see, e.g., [24], [25]. It would be interesting to combine these techniques with Algorithm 1 in future work.

## IV. NUMERICAL EXPERIMENTS

We now study the noise robustness of Algorithm 1 numerically, and compare it to several other state-of-the-art methods for near-separable NMF problems on a number of hyperspectral image data sets. We briefly summarize the different algorithms under consideration as follows.

1) *Successive projection algorithm (SPA).* SPA extracts recursively $r$ columns of the input matrix $M$. At each step, it selects the column with the largest $\ell_2$ norm, and projects all the columns of $M$ on the orthogonal complement of the extracted column [13]. SPA was shown to be robust to noise [14]. SPA can also be interpreted as a greedy method to solve the sparse regression model with self dictionary [26].
2) *XRAY.* It recursively extracts columns of the input unnormalized matrix $M$ corresponding to an extreme ray of the cone generated by the columns of $M$, and then projects all the columns of $M$ on the cone generated by the extracted columns. We used the variant referred to as "max" [5].
3) *Successive nonnegative projection algorithm (SNPA).* A variant of SPA using the nonnegativity constraints in the projection step [27]. To the best of our knowledge, it is the provably most robust sequential algorithm for separable NMF (in particular, it does not need $M(:\mathcal{K})$ to be full rank).
4) *Exact SOCP solution.* We solve the exact model (4) using the SOCP solver of Gurobi,[1] an interior point method. The obtained solution will serve as a "reference solution".
5) *FGNSR.* A Matlab/C implementation of Algorithm 1, which is publicly available.[2]

Our comparison does not include algorithms using linear functions to identify vertices (such as the pure pixel index algorithm [28] and vertex component analysis [12]) because they are not robust to noise and do not perform well for the challenging synthetic data sets described hereafter; see [14].

In all our experiments with FGNSR and the exact SOCP solution to (4) we use the simplest postprocessing to extract the sought for index set $\mathcal{K}$ from the solution matrix $X$ in (4): We always pick the indices of the $r$ largest diagonal values of $X$ (see final step in Algorithm 1).

Table I summarizes the following information for the different algorithms: computational cost, memory requirement, parameters, and whether the $H$ is required to be column normalized. The FLOP count and memory requirement for the exact solution of the SOCP via an interior-point method depends on the actual SOCP formulation used, as well as on the sparsity of the resulting problem. In any case, they are orders of magnitudes greater than for the other algorithms.

We complement these information with the average wall clock run times for the small "middlepoint" matrices from Section IV-A ($m = 50$, $n = 55$) in the last column. More run time results are shown in Section IV-B.

In the following section IV-A we study numerically the noise robustness of the model (6) on an artificial dataset, and in section IV-B we compare the methods from above to real-world hyperspectral image data sets.

### A. Robustness Study on Synthetic Datasets

The data set we consider is specifically designed to test algorithms for their robustness against noise. We set $m = 50$,

[1] https://www.gurobi.com
[2] https://github.com/rluce/FGNSR

$n = 55$ and $r = 10$. Given the noise level $\epsilon$, a noisy $r$-separable matrix

$$M = WH + N \in \mathbb{R}^{m,n} \tag{7}$$

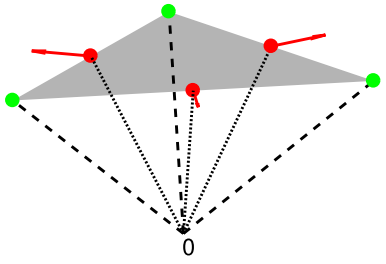
is generated as follows:

- Each entry of the matrix $W$ is generated uniformly at random in the interval [0, 1] (using the `rand` function of MATLAB), and each column of $W$ is then normalized so that it sums to one.
- The first $r$ columns of $H$ are always taken as the identity matrix to satisfy the separability assumption. The remaining $\frac{r(r-1)}{2} = 45$ columns of $H$ contain all possible combinations of two nonzero entries equal to 0.5 at different positions. Geometrically, this means that these 45 columns of $M$ are the *middle points* of all the pairs from the columns of $W$.
- No noise is added to the first $r$ columns of $M$, that is, $N(:, j) = 0$ for all $1 \leq j \leq r$, while all the other columns corresponding to the middle points are moved towards the exterior of the convex hull of the columns of $W$. Specifically, we set

$$N(:, j) = M(:, j) - \bar{w}, \quad \text{for } r + 1 \leq j \leq n,$$

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 $W$). Finally, the noise matrix $N$ is scaled so that it matches the given noise level $\|N\|_F = \epsilon$.

Finally, in order to prevent an artificial bias due to the ordering in which $H$ is constructed, the columns of $M$ are randomly permuted. We give the following illustration of this type of data set (with $m = r = 3$):




The shaded area shows the convex hull of $W$, and the arrow attached to the middle points indicate the direction of the noise added to them. With increasing noise level $\epsilon$, any algorithm for recovering the conic basis $W$ will eventually be forced to select some displaced middle points, and hence will fail to identify $W$, which makes this data set useful for studying the noise robustness of such algorithms.

In order to compare the algorithms listed at the beginning of the section, two measures between zero and one will be used, one being the best possible value and zero the worst: given a set of indices $\mathcal{K}$ extracted by an algorithm, the measures are as follows:

- *MRSA*. We compute the mean-removed spectral angle between a selected basis column $w$ and the true basis column $w_*$, according to

$$\arccos\left(\frac{\langle w - \bar{w}, w_* - \bar{w}_* \rangle}{\|w - \bar{w}\| \|w_* - \bar{w}_*\|}\right),$$

and normalizing the result to the interval [0, 100] (a value of zero is a perfect match). In order to obtain a single number for a given computed basis matrix $W$ we take the mean of all individual MRSAs.
- *Relative approximation error.* It is defined as

$$1.0 - \frac{\min_{H \geq 0} \|M - M(:, \mathcal{K})H\|_F}{\|M\|_F}.$$

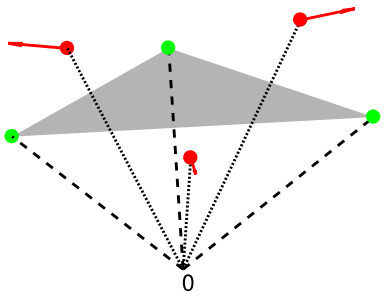
(Taking $H = 0$ gives a measure of zero).

Figure 1 shows these two measures over a series of data sets over increasing noise level $\epsilon$. For each noise level, a random middle point data set as described above was generated 25 times, and the average of the respective measure over this sample yields one data point at noise level $\epsilon$.

Algorithm 1 is clearly superior to all other algorithms, and recovers the true conic basis even at quite large noise levels. With the heuristic choice for the multiplier $\mu$ (see Section III-C), the robustness is still slightly inferior to the exact SOCP solution. The results labelled "FGNSR, dynamic" refers to a variant of Algorithm 1 where $\mu$ is heuristically adjusted in the course of the gradient iteration so that $\|M - MX\|_F \approx \epsilon$. (Similarly, one could steer $\mu$ towards a prescribed value of trace$(X)$).

Note that by construction the $\ell_1$ norm of all the columns in $H$ in (7) is 1.0, which is in fact a requirement by for some the algorithms considered here (see Table I). It is an important feature of Algorithm 1 that it is also applicable if the columns of $H$ are not normalized. We now study this case in more detail.

Consider the following slight variation of the middle point data from above: Instead of placing the middle points by means of a convex combination of two vertices, we now allow for conic combination of these pairs, i.e., the middle point will be randomly scaled by some scalar in $[\alpha^{-1}, \alpha]$ (we take $\alpha = 4$). The following picture illustrates these *scaled middle point data*:




We compare the algorithms listed at the beginning of this section on this data set exactly as described above. the results are shown in Figure 2. The results labeled "normalize, SPA" and "normalize, FGNSR" refer to $\ell_1$-normalizing the columns of the input matrix $M$ prior to applying SPA and FGNSR, respectively. From the results it is clear that FGNSR is by far the most robust algorithm in this setting.

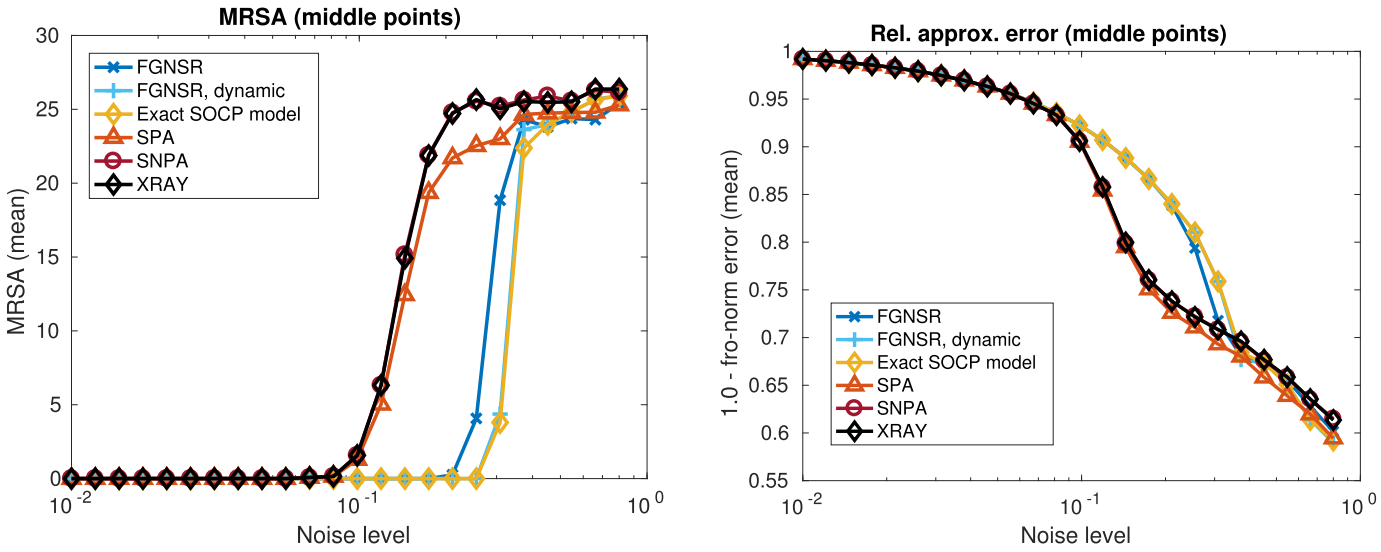



Fig. 1. Robustness study of various near-separable algorithms on the middle point set (see Sec. IV-A). *Left:* Index recovery measure. Note that the results of "XRAY" and "SNPA" are visually almost indistinguishable, as are those of "Exact SOCP model" and "FGNSR, dynamic". *Right:* Relative approximation error.

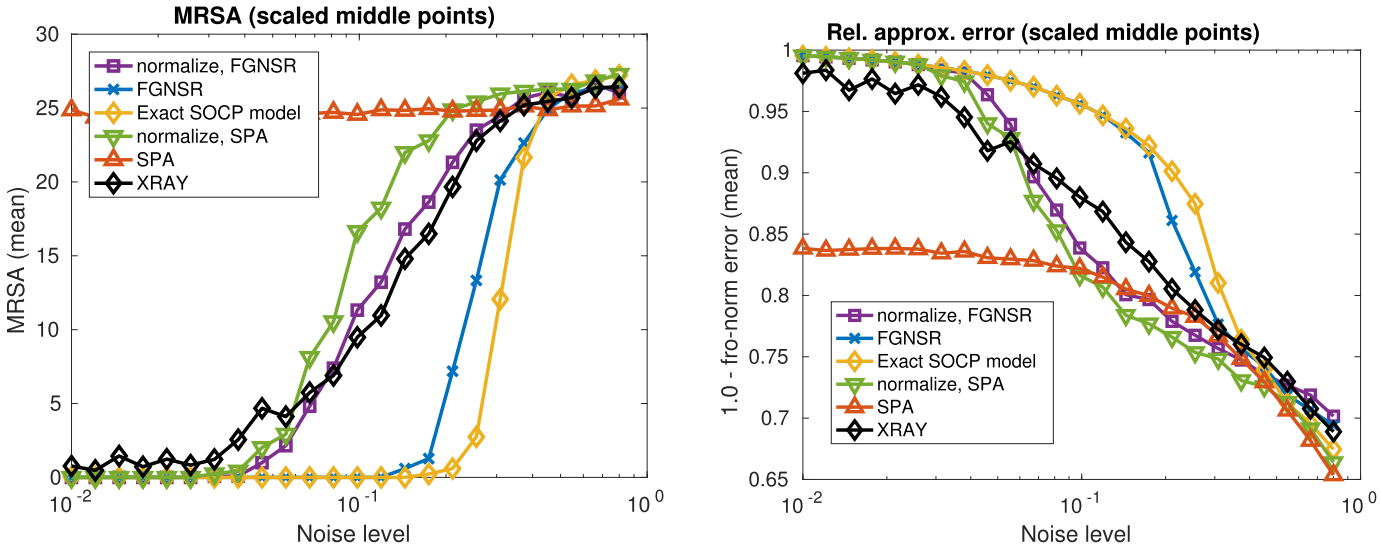



Fig. 2. Robustness study of various near-separable algorithms on the scaled middle point set (see Sec. IV-A). The show data is analogous to the data in Figure 1. The results for SNPA are not shown here to allow for a cleaner presentation; they are very similar to the ones for SPA.

### *B. Blind Hyperspectral Unmixing*

A hyperspectral image (HSI) measures the fraction of light reflected (the reflectance) by the pixels at many different wavelengths, usually between 100 and 200. For example, most airborne hyperspectral systems measure reflectance for wavelengths between 400nm and 2500nm, while regular RGB images contain the reflectance for three visible wavelengths: red at 650nm, green at 550nm and blue at 450nm. Hence, HSI provide much more detailed images with information invisible to our naked eyes. A HSI can be represented as a nonnegative $m$-by-$n$ matrix where the entry $(i, j)$ of matrix $M$ is the reflectance of the $j$th pixel at the $i$th wavelength, so that each column of $M$ is the so-called spectral signature of a given pixel. Assuming the linear mixing model, the spectral signature of each pixel equals the linear combination of the spectral signatures of the constitutive materials it contains, referred to as endmembers, where the weights correspond to the abundance of each endmember in that pixel. This is a simple but natural model widely used in the literature. For example, if a pixel contains 60% of grass and 40% of water, its spectral signature will be 0.6 times the spectral signature of the grass plus 0.4 times the spectral signature of water, as 60% is reflected by the grass and 40% by the water. Therefore, we have

$$M(:, j) = \sum_{k=1}^{r} W(:, k)H(k, j) + N(:, j),$$

where $M(:, j)$ is the spectral signature of the $j$th pixel, $r$ is the number of endmembers, $W(:, k)$ is the spectral signature of the $k$th endmember, $H(k, j)$ is the abundance of the $k$th endmember in the $j$th pixel, and $N$ represents the noise (and modeling errors). In this context, the separability assumption is

equivalent to the so-called pure-pixel assumption that requires that for each endmember there exists a pixel containing only that endmember, that is, for all $k$, there exists $j$ such that $M(:, j) \approx W(:, k)$.

The theoretical robustness results of near-separable NMF algorithms do not apply in most cases, the reasons being that (i) the noise level is usually rather high, (ii) images contain outliers, (iii) the linear mixing model itself is incorrect (in particular because of multiple interactions of the light with the surface, or because of its interaction with the atmosphere), (iv) the pure-pixel assumption is only approximately satisfied (or only some endmembers have pure pixels), and (v) the number of endmembers is unknown (and usually endmembers in small proportion are considered as noise); see [3] and the references therein.

However, near-separable NMF algorithms (a.k.a. pure-pixel search algorithms) usually allow to extract pure (or almost pure) pixels and are very popular in the community; for example NFIND-R [29] or vertex component analysis (VCA) [12]. They can also be particularly useful to initialize more sophisticated method not based on the pure-pixel assumption; see [4].

For most HSI, $n$ is of the order of millions, and it is impractical to solve (6) with either our fast gradient method or even the interior point solver of Gurobi, as we did in Section IV-A. In the next section, we adopt a strategy similar to that in [15] where a subset of pixels is first selected as a preprocessing step. Then, we apply our model (6) on that subset using an appropriate strategy which is described in the next section.

*1) Subsampling and Scaling of HSI's:* A natural way to handle the situation when $n$ is large is to preselect a subset of the columns of $M$ that are representative of the data set. In [15], authors used $k$-means to identify that subset. However, $k$-means has several drawbacks: it cannot handle different scaling of the data points, and scales badly as the number of clusters increases, running in $\mathcal{O}(mnC)$ where $C$ is the number of clusters to generate. A much better alternative, that was specifically designed to deal with HSI, is the hierarchical clustering procedure developed in [30]. The computational cost is $\mathcal{O}(mn \log_2 C)$ (given that it generates well-balanced clusters).

Keeping only the centroids generated by a clustering algorithm is a natural way to subsample HSI. However, it is important to take into account the importance of each centroid, that is, the number of data points attached to it. Let $\mathcal{C}$ be the index set corresponding to the centroids $M(:, \mathcal{C})$ of the extracted clusters. For $|\mathcal{C}|$ sufficiently large, each pixel will be relatively close to its centroid: mathematically, for all $j$, there exists $k \in \mathcal{C}$ such that $M(:, j) \approx M(:, k)$. If we only allow the centroids in the dictionary and denote $X \in \mathbb{R}^{|\mathcal{C}|,|\mathcal{C}|}$ the corresponding weights, the error term can be approximated by

$$\begin{aligned}\|M - M(:, K)X\|_F^2 &= \sum_{j=1}^{n} \|M(:, j) - M(:, K)X(:, j)\|_F^2 \\ &\approx \sum_{k \in \mathcal{C}} n_k \|M(:, k) - M(:, K)X(:, k)\|_F^2 \\ &= \sum_{k \in \mathcal{C}} \|\sqrt{n_k} M(:, k) - \sqrt{n_k} M(:, K)X\|_F^2,\end{aligned}$$

where $n_k$ the number of pixels in the $k$th cluster.

Therefore, in this section, we apply our model only to the matrix $M(:, \mathcal{C})$ where each centroid is scaled according to the square root of the number of points belonging to its cluster. This allows us to take into account the importance of the different clusters. For example, an outlier will correspond to a cluster with a single data points (provided that $|\mathcal{C}|$ sufficiently large) hence its influence in the objective function will be negligible in comparison with large clusters.

*Postprocessing of $X$:* In the synthetic data sets, we identified the subset $\mathcal{K}$ using the $r$ largest entries of $X$. It worked well because (i) the data sets did not contain any outlier, and (ii) there were no (near-)duplicated columns in the data sets. In real data sets, these two conditions are usually not met. Note however that the preprocessing clustering procedure aggregates (near-)duplicated columns. However, if a material is present in very large proportion of the image (e.g., the grass in the Urban data sets; see below), several clusters will be made mostly of that material.

Therefore, in order to extract a set of column indices from the solution matrix $X$ (see Algorithm 1, line 10), we will use a more sophisticated strategy.

The $i$th row of matrix $X$ provides the weights necessary to reconstruct each column of $M$ using the $i$th column of $M$ (since $M \approx MX$), while these entries are bounded by the diagonal entry $X_{ii}$. From this, we note that

(i) If the $i$th row corresponds to an outlier, it will in general have its corresponding diagonal entry $X_{ii}$ non-zero but the other entries will be small (that is, $X_{ij}$ $j \neq i$). Therefore, it is important to also take into account off-diagonal entries of $X$ in the postprocessing: a row with a large norm will correspond to an endmember present in many pixels. (A similar idea was already proposed in [14, Sec. 3].)

(ii) Two rows of $X$ that are close to one another (up to a scaling factor) correspond to two endmembers that are present in the same pixels in the same proportions. Therefore, it is likely that these two rows correspond to the same endmember. Since we would like to identify columns of $M$ that allow to reconstruct as many pixels as possible, we should try to identify rows of $X$ that are as different as possible. This will in particular allow us to avoid extracting near-duplicated columns.

Finally, we need to identify rows (i) with large norms (ii) that are as different as one another as possible. This can be done using SPA on $X^T$: at each step, identify the row of $X$ with the largest norm and project the other rows on its orthogonal complement (this is nothing but a QR-factorization with column pivoting). We observe in practice that this postprocessing is particularly effective at avoiding outliers and near-duplicated columns (moreover, it is extremely fast).

*2) Experimental Setup:* In the following sections, we combine the hierarchical clustering procedure with our near-separable NMF algorithm and compare it with state-of-the-art pure-pixel search algorithms (namely SPA, VCA, SNPA, H2NMF and XRAY) on several HSI's. We have included

TABLE II

NUMERICAL RESULTS FOR THE URBAN HSI
(THE BEST RESULT IS HIGHLIGHTED IN BOLD)

| | $r = 6$ | | $r = 8$ | |
|---|---|---|---|---|
| | Time (s.) | Rel. error | Time (s.) | Rel. error |
| VCA | 1.02 | 18.05 | 1.05 | 22.68 |
| VCA-100 | 0.05 | 6.67 | 0.07 | 4.76 |
| VCA-500 | 0.03 | 7.19 | 0.09 | 7.25 |
| SPA | 0.26 | 9.58 | 0.32 | 9.45 |
| SPA-100 | <0.01 | 9.49 | <0.01 | 5.01 |
| SPA-500 | <0.01 | 10.05 | <0.01 | 8.86 |
| SNPA | 13.60 | 9.63 | 23.02 | 5.64 |
| SNPA-100 | 0.10 | 11.03 | 0.15 | 6.17 |
| SNPA-500 | 0.15 | 10.05 | 0.25 | 8.86 |
| XRAY | 28.17 | 7.50 | 95.34 | 6.82 |
| XRAY-100 | 0.11 | 6.78 | 0.17 | 6.57 |
| XRAY-500 | 0.15 | 8.07 | 0.28 | 7.36 |
| H2NMF | 12.20 | 5.81 | 14.92 | 5.47 |
| H2NMF-100 | 0.16 | 7.11 | 0.23 | 6.14 |
| H2NMF-500 | 0.27 | 5.87 | 0.37 | 5.68 |
| FGNSR-100 | 2.73 | 5.58 | 2.55 | 4.62 |
| FGNSR-500 | 40.11 | **5.07** | 39.49 | **4.08** |

TABLE III

NUMERICAL RESULTS FOR THE SAN DIEGO HSI
(THE BEST RESULT IS HIGHLIGHTED IN BOLD)

| | $r = 8$ | | $r = 10$ | |
|---|---|---|---|---|
| | Time (s.) | Rel. error | Time (s.) | Rel. error |
| VCA | 1.71 | 7.46 | 1.79 | 9.46 |
| VCA-100 | 0.07 | 8.49 | 0.12 | 6.08 |
| VCA-500 | 0.06 | 9.19 | 0.13 | 6.29 |
| SPA | 0.53 | 12.62 | 0.61 | 7.01 |
| SPA-100 | 0.03 | 8.49 | 0.01 | 5.83 |
| SPA-500 | <0.01 | 12.64 | <0.01 | 6.61 |
| SNPA | 38.95 | 12.84 | 58.45 | 7.67 |
| SNPA-100 | 0.22 | 8.49 | 0.20 | 6.90 |
| SNPA-500 | 0.25 | 12.64 | 0.48 | 6.47 |
| XRAY | 93.29 | 13.06 | 243.40 | 12.62 |
| XRAY-100 | 0.14 | 8.68 | 0.21 | 5.12 |
| XRAY-500 | 0.19 | 13.17 | 0.35 | 6.82 |
| H2NMF | 21.51 | 4.75 | 24.42 | 4.28 |
| H2NMF-100 | 0.30 | 6.85 | 0.22 | 5.61 |
| H2NMF-500 | 0.33 | 6.78 | 0.38 | 5.75 |
| FGNSR-100 | 2.55 | **3.73** | 2.47 | **3.40** |
| FGNSR-500 | 38.70 | 4.05 | 38.28 | **3.40** |

vertex component analysis (VCA) [12] because it is extremely popular in the hyperspectral unmixing community, although it is not robust to noise [14]. VCA is similar to SPA except that (i) it first performs dimensionality reduction of the data using PCA to reduce the ambient space to dimension $r$, and (ii) selects the column maximizing a randomly generated linear function.

Because the clustering procedure already does some work to identify candidate pure pixels, it could be argued that the comparison between our hybrid approach and plain pure-pixel search algorithms is unfair. Therefore, we will also apply SPA, VCA, XRAY, H2NMF and SNPA on the subsampled data set. We subsample the data set by selecting 100 (resp. 500) pixels using H2NMF, and denote the corresponding algorithms SPA-100 (resp. SPA-500), VCA-100 (resp. VCA-500), etc.

Because it is difficult to assess the quality of a solution on a real-world HSI, we use the relative error in percent: given the index set $\mathcal{K}$ extracted by an algorithm, we report

$$100\frac{\min_{H\geq 0}\|M - M(:,\mathcal{K})H\|_F}{\|M\|_F},$$

where $M$ is always the full data set.

The MATLAB code used in this study is available,[3] and all computations were carried out with MATLAB-R2015b on a standard Linux/Intel box.

*3) Data Sets and Results:* We will compare the different algorithms on the following data sets:

- The Urban HSI[4] is taken from HYper-spectral Digital Imagery Collection Experiment (HYDICE) air-borne sensors, and contains 162 clean spectral bands where each image has dimension 307 × 307. The corresponding near-separable nonnegative data matrix $M$ therefore has dimension 162 by 94249. The Urban data is mainly composed of 6 types of materials: road, dirt, trees, roofs, grass and metal (as reported in [31]).

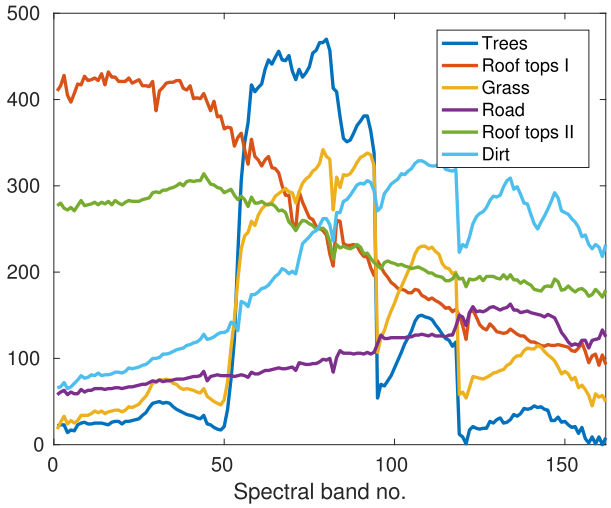



Fig. 3. Endmembers of the Urban dataset obtained from FGNSR after preprocessing with 500 clusters.

- The San Diego airport HSI is also from the HYDICE air-borne sensors. It contains 158 clean bands, with 400 × 400 pixels for each spectral image hence $M \in \mathbb{R}_+^{160000\times 158}$. There are about eight types of materials: three road surfaces, two roof tops, trees, grass and dirt; see, e.g., [30].
- The Terrain HSI data set is constituted of 166 clean bands, each having 500 × 307 pixels, and is composed of about 5 different materials: road, tree, bare soil, thin and tick grass.[5]

Tables II–IV show the relative error attained by the different algorithms on the data sets "Urban" ($r = 6$ and ($r = 8$), "San Diego" ($r = 8$ and $r = 10$), and "Terrain" ($r = 5$ and $r = 6$). The reported time refers to the run time of the algorithms, without the preprocessing step.

We observe that FGNSR-100 and FGNSR-500 perform consistently better than all the other algorithms, although, as expected, at a higher computational cost than SPA and VCA.

[3] https://sites.google.com/site/nicolasgillis/

[4] http://www.erdc.usace.army.mil/

[5] http://www.way2c.com/rs2.php

Fig. 4. Abundance maps corresponding to the endmembers extracted by FGNSR-500 for the Urban HSI ($r = 6$). From left to right, top to bottom: (i) trees, (ii) roof tops I, (iii) grass, (iv) road, (v) roof tops II, (vi) dirt.

TABLE IV
NUMERICAL RESULTS FOR THE TERRAIN HSI
(THE BEST RESULT IS HIGHLIGHTED IN BOLD)

| | $r = 5$ | | $r = 6$ | |
|---|---|---|---|---|
| | Time (s.) | Rel. error | Time (s.) | Rel. error |
| VCA | 1.65 | 10.92 | 1.67 | 6.22 |
| VCA-100 | 0.02 | 5.59 | 0.03 | 7.33 |
| VCA-500 | 0.03 | 5.77 | 0.03 | 5.57 |
| SPA | 0.38 | 5.89 | 0.43 | 4.81 |
| SPA-100 | <0.01 | 4.74 | 0.01 | 3.95 |
| SPA-500 | 0.01 | 4.83 | 0.01 | 4.63 |
| SNPA | 17.54 | 5.76 | 24.28 | 4.60 |
| SNPA-100 | 0.10 | 5.75 | 0.11 | 5.65 |
| SNPA-500 | 0.10 | 4.83 | 0.13 | 4.78 |
| XRAY | 33.63 | 5.39 | 73.91 | 5.17 |
| XRAY-100 | 0.07 | 4.15 | 0.12 | 4.13 |
| XRAY-500 | 0.09 | 5.21 | 0.19 | 4.97 |
| H2NMF | 18.23 | 5.09 | 20.92 | 4.85 |
| H2NMF-100 | 0.15 | 4.72 | 0.17 | 4.39 |
| H2NMF-500 | 0.23 | 5.43 | 0.29 | 5.35 |
| FGNSR-100 | 4.23 | **3.34** | 2.63 | **3.21** |
| FGNSR-500 | 40.29 | 3.68 | 40.13 | 3.39 |

We summarize the results as follows.

- For the Urban HSI with $r = 6$ (resp. $r = 8$), FGNSR-100 provides a solution with relative error 5.58% (resp. 4.62%) and FGNSR-500 with relative error 5.07% (resp. 4.08%), the third best being VCA-100 with 5.94% (resp. SPA-100 with 5.01%).
- For the San Diego airport HSI with $r = 8$ (resp. $r = 10$), FGNSR-100 provides a solution with relative error 3.73% (resp. 3.40%) and FGNSR-500 with relative error 4.05% (resp. 3.40%), the third best being H2NMF with 4.75% (resp. XRAY-100 with 5.12%).
- For the Terrain HSI with $r = 5$ (resp. $r = 6$), FGNSR-100 provides a solution with relative error 3.34% (resp. 3.21%) and FGNSR-500 with relative error 3.68% (resp. 3.39%), the third best being XRAY-100 with 4.15% (resp. SPA-100 with 3.95%).

It is interesting to note that, in most cases, near-separable algorithms applied on the subset of columns identified by H2NMF perform much better than when applied on the full data set. The reason is that these algorithms tend to extract outlying pixels which are filtered out by the subsampling procedure (especially when the number of clusters is small).

We show the computed endmembers for the "Urban" data set in Figure 3, and the corresponding abundance maps for each of the species is shown in Figures 4.

To conclude, we have observed on three data sets that FGNSR-100 and FGNSR-500 are able to identify the best subset of columns to reconstruct the original input image in all cases, while its computational cost is reasonable.

*Remark 2: Note that we also compared the algorithms on the widely used Cuprite data set but the results are not very*

*interesting as most algorithms find very similar solutions in terms of relative error. The reason is that the data is not contaminated with outliers and spectral signatures are rather similar in that data set. For example, for $r = 15$, all algorithms have relative error in the interval [1.39,1.99]%, and, for $r = 20$, in the interval [1.35,1.84]%.*

## V. Conclusion and Further Work

In this paper, we analyzed a robust convex optimization model for dealing with nonnegative sparse regression with self dictionary; in particular showing its close connection with the model proposed in [15]. We then developed an optimal first-order method to solve the problem. We showed that this approach outperforms standard near-separable NMF algorithms on synthetic data sets, and on real-world HSI's when combined with a hierarchical clustering strategy. Moreover, we observed that preselecting a small number of good candidate allows all near-separable NMF algorithms to perform much better.

The model (6) (and the corresponding Algorithm 1) can be easily generalized to handle any dictionary $D$, changing the model to

$$\min_{X\in\Omega'} \ \text{trace}(X) \quad \text{such that } \|M - DX\|_F \leq \epsilon,$$

where

$$\Omega' = \{X \in [0,1]^{n,n} \mid X_{ij}\|D(:,i)\|_1 \leq X_{ii}\|M(:,j)\|_1 \forall i, j\}.$$

It would therefore be an interesting direction of further research to analyze and apply this model in other contexts.

Further it would be of interest to study the robustness of Algorithm 1 if combined with random projections (see [24], [25]), and to consider non-additive noise models, e.g., spectral variability in the context of HSI [32].

## Appendix A
## Proof of Theorem 2

Observe that

- Any feasible solution of (3) is a feasible solution of (2): in fact, the only difference between the feasible domains are the additional constraints $X_{ij} \leq X_{ii}$ for all $i, j$.
- The objective function of (2) is larger than the one of (3): in fact, by definition, $\text{trace}(X) \leq ||X||_{1,\infty}$. Moreover, the two objective functions coincide if and only if $X_{ii} = \max_j X_{ij}$ for all $i$.

These observations imply that the optimal objective function value of (3) is larger than the one of (2) (since any feasible solution $X$ of (3) is feasible for (2) and satisfies $\text{trace}(X) = \|X\|_{1,\infty}$).

Therefore, if we can transform any optimal solution $X^*$ of (2) into a feasible solution $X^\dagger$ of (3) with the same objective function value, $X^\dagger$ will be an optimal solution of (3) and the proof will be complete.

Let $X^*$ be any optimal solution of (2). If $X^* = 0$, then $X^*$ is trivially feasible for (3) and the proof is complete.

So assume $X^* \neq 0$. We will show by contradiction that $\|M - MX^*\| = \epsilon$, so assume that $\|M - MX^*\| < \epsilon$. By continuity of norms there exists $0 < \delta < 1$ so that $\|M - M(\delta X^*)\| < \epsilon$. The matrix $\delta X^*$ is a feasible solution for (2) since $0 \leq \delta X^* \leq X^* \leq 1$ while $\|\delta X^*\|_{1,\infty} = \delta\|X^*\|_{1,\infty} < \|X^*\|_{1,\infty}$, a contradiction to the optimality of $X^*$.

Assume that $X^*_{ii} < X^*_{ij} \leq 1$ for some $j$. Let us show this is only possible if $M(:,i) = MX^*(:,i)$: assume $M(:,i) \neq MX^*(:,i)$, we have

$$M(:,i) - MX^*(:,i) = (1 - X^*_{ii})M(:,i) - MX^*(\mathcal{I},i),$$

where $\mathcal{I} = \{1,\ldots,n\} \setminus \{i\}$. Increasing $X^*_{ii}$ to $X^*_{ij}$ while decreasing the entries of $X^*(\mathcal{I},i)$ by the factor $\beta = \frac{X^*_{ii}}{X^*_{ij}} < 1$ decreases $\|M(:,i) - MX(:,i)\|_c$ by a factor $(1 - X^*_{ii})$ which would be a contradiction since $\|M - MX\|$ would be reduced (see above).

Finally, let us construct another optimal solution $X^\dagger$: we take $X^\dagger = X^*$, and for all $j$ such that $X^*_{ii} < X^*_{ij} \leq 1$, $X^\dagger_{ii}$ is replaced with $X^\dagger_{ij}$ and $X^\dagger(\mathcal{I},i)$ is multiplied by the factor $\beta = \frac{X^*_{ii}}{X^*_{ij}} < 1$. The error $\|M - MX\|$ remains unchanged while the objective function might have only decreased: $X^\dagger$ is an optimal solution of (2) satisfying $X^\dagger_{ii} = \max_j X^\dagger_{ij}$ hence is also an optimal solution of (3).

## Appendix B
## Projection Onto $\Omega$

We now give the details for evalutating the Euclidean projection onto the set $\Omega$, see Section III-D. Note first that it is sufficient to consider the problem of projecting a single row of $X$, say, the first one, on the set

$$\Omega_1 := \{z \in \mathbb{R}^n_+ \mid z_1 \leq 1, w_1 x_j \leq w_j z_1\},$$

since the the rows of $X$ can be projected individually on $\Omega$ as they do not depend on each other in $\Omega$. Further we may assume that $w > 0$ because $w_1 = 0$ removes all constraints on $x_j$ for $2 \leq j \leq n$, and $w_j$ for $j \neq 1$ fixes $x_j = 0$ for any point in $\Omega_1$. Similarly we can assume w.l.o.g. that $x_j > 0$ for $2 \leq j \leq n$, since $x_j \leq 0$ fixes $z_j = 0$ for any point in $\Omega_1$ and does not affect the choice of the first coordinate. Note that the norm we wish to minimize now is given by the standard Euclidean norm $\|x\| = (\sum_j x_j^2)^{\frac{1}{2}}$.

Let $t \in \mathbb{R}$ be a parameter and denote by $\phi_t : \mathbb{R}^n \to \mathbb{R}^n$ the mapping

$$\phi_t(x)_j := \begin{cases} t & \text{if } j = 1, \\ \frac{w_j}{w_1} t & \text{if } j \neq 1 \text{ and } \frac{w_1}{w_j} x_j \geq t, \\ x_j & \text{else.} \end{cases} \tag{8}$$

From the definition we see that for all $x \in \mathbb{R}^n$ and all $t \in [0,1]$ we have that $\phi_t(x) \in \Omega_1$.

Unfortunately we cannot simply assume that $0 \leq x_1 \leq 1$. Treating the cases $x_1 < 0$, $0 \leq x_1 \leq 1$ and $1 < x_1$ homogeneously puts a burden on the notation and slightly obfuscates the arguments used in the following. We set $x_1^+ := \min\{1, \max\{0, x_1\}\}$ and $x_1^- := \min\{0, x_1\}$.

*Lemma 1: Let $x \in \mathbb{R}^n$, then*

$$\min_{z\in\Omega_1} \|x - z\| = \min_{t\in[x_1^+,1]} \|x - \phi_t(x)\|.$$

*In particular, if* $z^* = \operatorname{argmin}_{z\in\Omega_1}\|x - z\|$, *we have for* $2 \le j \le n$ *that*

$$w_1 z_j^* < w_j z_1^* \quad \text{if } w_1 x_j < w_j z_1^*$$

*and*

$$w_1 z_j^* = w_j z_1^* \quad \text{if } w_1 x_j \ge w_j z_1^*.$$

*Proof:* Let $z^* = \operatorname{argmin}_{z\in\Omega_1}$. We will show first that

$$\min_{z\in\Omega_1}\|x - z\| = \min_{t\in[0,1]}\|x - \phi_t(x)\|.$$

If $w_1 x_j < w_j z_1^*$, it follows $z_j^* = x_j$, because only in this case the cost contribution of the $j$-th coordinate is minimum (zero). Otherwise, if $w_1 x_j \ge w_j z_1^*$, the projection cost from the $j$-th coordinate is minimized only if $x_j = \frac{w_j}{w_1} z_1^*$, because all the quantities involved are positive. It follows that $z_j^* = \phi_{z_1^*}(x)_j$ for $2 \le j \le n$. And since $z_1^* \in [0, 1]$, it follows that $z^* = \operatorname{argmin}_{t\in[0,1]}\|x - \phi_t(x)\|$.

To finish the proof, we will now show that $z_1^* \ge x_1^+$, which is trivial for the case $x_1 < 0$. In the case $x_1 \in [0, 1]$, we show $z_1^* \ge x_1$. In order to obtain a contradiction, assume $z_1^* < x_1$, and define $\tilde{z} \in \Omega_1$ by

$$\tilde{z}_j = \begin{cases} x_1 & \text{if } j = 1 \\ z_j^* & \text{if } j \ne 1. \end{cases}$$

We compute

$$\begin{aligned}\|x - \tilde{z}\|_2 &= \sum_{j=1}^{n}(x_j - \tilde{z}_j)^2 = \sum_{j=2}^{n}(x_j - z_j^*)^2 < \sum_{j=1}^{n}(x_j - z_j^*)^2 \\ &= \|x - z^*\|_2,\end{aligned}$$

which contradicts the optimality of $z^*$.

In the case $x_1 > 1$, a similar reasoning shows that $z_1^* \ge 1$ (in fact $z_1^* = 1$). □

The previous lemma shows that the optimal projection can be computed by minimization of a univariate function. We will show next that this can be done quite efficient. For a given $x \in \mathbb{R}^n$ we define the function

$$c_x : [x_1^-, \infty[ \to \mathbb{R}, \quad t \mapsto \|x - \phi_t(x)\|^2.$$

By Lemma 1, the (squared) minimum projection cost is given by the minimum of $c_x|_{[x_1^+,1]}$, and our next step is to understand the behavior of $c_x$ (see Fig. 5).

In order to simplify the notation in the following, we abbreviate $b_j := \frac{w_1}{w_j} x_j$ and assume that the components of $x$ are ordered so that $b_2 \le b_3 \le \cdots \le b_n$, and since $x_j > 0$ for $2 \le j \le n$, we have in fact that

$$x_1^- =: b_1 \le 0 < b_2 \le b_3 \le \cdots \le b_n. \tag{9}$$

*Lemma 2: Let* $x \in \mathbb{R}^n$.

*(i) The function* $c_x$ *is a piecewise* $C^\infty$ *function with* $C^1$ *break points* $b_j$, *and each piece* $c_x|_{[b_k,b_{k+1}]}$ *is strongly convex. In particular,* $c_x$ *is strongly convex and attains its minimum.*

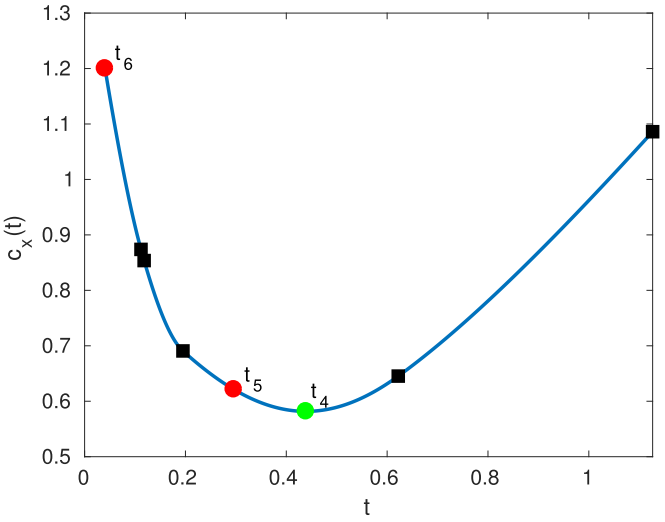



Fig. 5. Example for the minimization of $c_x$ ($n = 6$). Black squares indicate the break points and the discs show the location of the trial values $t_k$ from (12). The optimal projection is realized by $t_4$ and the set of optimal active indices are the two break points to the right of $t_4$.

*(ii) Let* $t^* = \operatorname{argmin}_{t\in[x_1^-,\infty[} c_x(t)$, *the optimal projection of* $x$ *on* $\Omega_1$ *is*

$$\operatorname{argmin}_{z\in\Omega_1}\|x - z\| = \begin{cases} \phi_{x_1^+}(x) & \text{if } t^* < x_1^+, \\ \phi_{t^*}(x) & \text{if } t^* \in [x_1^+, 1] \text{ or} \\ \phi_1(x) & \text{if } 1 < t^*. \end{cases} \tag{10}$$

*Proof:* We show (i) first. Using the definition (8), we compute for $x \in \mathbb{R}^n$ and $t \in [b_1, \infty[$

$$c_x(t) = \|x - \phi_t(x)\|^2 = (x_1 - t)^2 + \sum_{j\in B(t)}(x_j - \tfrac{w_j}{w_1}t)^2,$$

where

$$B(t) := \{1 \le j \le n \mid t \le b_j\},$$

from which we see that $c_x$ is piecewise smooth and continuously differentiable at the points $b_j$. For a point $t \in ]b_k, b_{k+1}[$, we see that $c_x''(t) > 0$ which shows strong convexity on each piece. But $c_x'$ is continuous at each $b_j$, so $c_x$ is strongly convex on all of $[b_1, \infty[$. Finally, since $\lim_{t\to\infty} c_x(t) = \lim_{t\to\infty}(x_1 - t)^2 = \infty$, we see see that $c_x$ attains its minimum.

The assertions in (ii) follow directly from Lemma 1 and the convexity of $c_x$. □

Since $c_x$ attains its minimum at $t^* \in [b_1, \infty[$, we have that either $t^* \in [b_1, b_2]$, $t^* \in ]b_k, b_{k+1}]$, for some $2 \le k < n$, or $t^* \in ]b_n, \infty[$. By the definition of $\phi_t$, the constraints $w_1 x_j \le w_j x_1$ corresponding to break points $b_j \ge t^*$ are "active" at the point $\phi_{t^*}(x)$, while all other constraints are not active. More formally, in each of the cases in (11), we can uniquely associate a set of optimal active indices $B^* \subseteq \{2, \ldots, n\}$ as follows (here $2 \le k \le n - 1$):

$$B^* = \begin{cases} B_1 := \{2, \ldots, n\} & \text{iff } t^* \in [b_1, b_2] =: T_1 \\ B_k := \{k + 1, \ldots, n\} & \text{iff } t^* \in ]b_k, b_{k+1}] =: T_k \\ B_n := \emptyset & \text{iff } t^* \in ]b_n, \infty[=: T_n. \end{cases} \tag{11}$$

**Algorithm 2** Euclidean Projection on $\Omega_1$

**Require:** $x \in \mathbb{R}^n$ with $x_2, \ldots, x_n > 0$, $0 < w \in \mathbb{R}^n$, heap data structure $h$
**Ensure:** $z \in \Omega_1$ with $\|x - z\|_2$ minimal
$x_1^+ \leftarrow \max\{0, x_1\}$
$\mathcal{B} \leftarrow \{2 \le j \le n \mid x_1^+ \le \frac{w_1}{w_j} x_j \le 1\}$
$\mathcal{B}^* \leftarrow \{2 \le j \le n \mid \frac{w_1}{w_j} x_j > 1\}$
$p \leftarrow w_1 x_1 + \sum_{j \in \mathcal{B}^*} w_j x_j$
$q \leftarrow w_1^2 + \sum_{j \in \mathcal{B}^*} w_j^2$
$t \leftarrow w_1 \frac{p}{q}$
initheap($h$, $\{\frac{w_1}{w_j} x_j \mid j \in \mathcal{B}\}$) % *Operation linear in* $|\mathcal{B}|$.
**while** $h \neq \emptyset$ and $t <$ findmin($h$) **do**
    $M \leftarrow$ extractmin($h$)
    Let $2 \le j \le n$ such that $M$ corresponds to $\frac{w_1}{w_j} x_j$
    $\mathcal{B}^* \leftarrow \mathcal{B}^* \cup \{j\}$
    $p \leftarrow p + w_j x_j$
    $q \leftarrow q + w_j^2$
    $t \leftarrow w_1 \frac{p}{q}$
**end while**
$z \leftarrow x$
$z_1 \leftarrow \min\{1, \max\{0, t\}\}$ {See (10)}
**for all** $j \in \mathcal{B}^*$ **do**
    $z_j \leftarrow z_1 \frac{w_j}{w_1}$ {See (8)}
**end for**

The preceding observation yields an efficient "dual" algorithm for minimizing $c_x$ over $[b_1, \infty[$, which we develop in the following lemma.

*Lemma 3: Let $x \in \mathbb{R}^n$, and $t^* = \operatorname{argmin}_{t \in [b1, \infty[} \|x - \phi_t(x)\|$, denote the set of active indices corresponding to $t^*$ by $B^* = \{1 \le j \le n \mid b_j \ge t^*\}$, and set*

$$t_k = w_1 \frac{w_1 x_1 + \sum_{j \in B_k} w_j x_j}{w_1^2 + \sum_{j \in B_k} w_j^2}, \quad 1 \le k \le n. \tag{12}$$

*If $t_k \in T_k$, then $t^* = t_k$ (and $B^* = B_k$).*

*Proof:* Let $1 \le k \le n$. In order to minimize the strongly convex function

$$c_x(t) = (x_1 - t)^2 + \sum_{j \in B_k} (x_j - \tfrac{w_j}{w_1} t)^2,$$

one simply solves $c_x'(t) = 0$ for $t$, and obtains expression (12). So $t_k$ is the minimum of $c_x$ under the hypothesis that $B_k$ is the correct guess for $B^*$. But by (11) we have that $B_k = B^*$ if, and only if, $t_k \in T_k$, which gives a trivially verifiable criterion for deciding whether the guess for $B^*$ is correct. □

The algorithmic implication of this lemma is as follows. Since we know that one of the sets $B_1, \ldots, B_n$ must be the optimal active set $B^*$, we simply compute for each such set $B_k$ the corresponding optimal point from (12) until we encounter a point $t_k \in T_k$, which then is the sought optimum.

*Theorem 4: The Euclidean projection of $X \in R^{n,n}$ on $\Omega$ can be computed in $\mathcal{O}\left(n^2 \log n\right)$.*

*Proof:* The projection cost for each row of $X$ amounts to evaluating (12) for each of the sets $B_k$ until the optimal set of active indices is identified. If the sets $B_k$ are processed in the ordering $B_n, B_{n-1}, \ldots, B_1$, the nominator and denominator in (12) can be updated from one set to the next, resulting in a computation linear in $n$. The only non-linear cost per row is induced by sorting the break points of $c_x$ as in (9), which can be done in $\mathcal{O}(n \log n)$, and results in the stated worst-case complexity bound. □

The function $c_x$ is shown in Fig. 5 for a randomly chosen vector $x$ and weights $w$. In this example, the sets $B_6, B_5$ and $B_4$ are tested for optimality; the corresponding values $t_k$ are indicated in the plot.

*Remark 3: In an implementation of the outlined algorithm it is not necessary to sort* all *the break points of $c_x$ for a row $x$ of $X$ as in* (9)*. Only the break points $b_j \in [x_1^+, 1]$ need to be considered, as all other break points are either never (if $b_j < x_1^+$) or always (if $b_j > 1$) in the optimal set $B^*$ of active indices. Denote $k_1$ the number of break points in $[x_1^+, 1]$ and $k_2 := |B^*|$. If the indices are not sorted but maintained on a heap (see, e.g., [33]), the cost overhead for sorting is reduced to $\mathcal{O}(k_2 \log k_1)$. Hence the overall complexity for projecting a single row is $\mathcal{O}(n + k_2 \log k_1)$. The resulting algorithm is sketched in Algorithm 2. Asymptotically it still has the worst case complexity of $\mathcal{O}(n \log n)$ as we may need to extract all n possible elements from h, but it should run much faster in practice.*

## REFERENCES


[1] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999. [Online]. Available: http://dx.doi.org/10.1038/44565

[2] S. Arora, R. Ge, R. Kannan, and A. Moitra, "Computing a nonnegative matrix factorization–provably," in *Proc. 44th Symp. Theory Comput. (STOC)*, 2012, pp. 145–162. [Online]. Available: http://dx.doi.org/10.1145/2213977.2213994

[3] J. M. Bioucas-Dias *et al.*, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE J. Sel. Topics Appl. Earth Observat. Remote Sens.*, vol. 5, no. 2, pp. 354–379, Apr. 2012.

[4] W.-K. Ma *et al.*, "A signal processing perspective on hyperspectral unmixing: Insights from remote sensing," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 67–81, Jan. 2014.

[5] A. Kumar, V. Sindhwani, and P. Kambadur, "Fast conical hull algorithms for near-separable non-negative matrix factorization," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013, vol. 28. no. 1, pp. 231–239.

[6] S. Arora *et al.*, "A practical algorithm for topic modeling with provable guarantees," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013, vol. 28. no. 2, pp. 280–288.

[7] E. Elhamifar, G. Sapiro, and R. Vidal, "See all by looking at a few: Sparse modeling for finding representative objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1600–1607. [Online]. Available: http://dx.doi.org/10.1109/cvpr.2012.6247852

[8] T. H. Chan, W. K. Ma, C. Y. Chi, and Y. Wang, "A convex analysis framework for blind separation of non-negative sources," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 5120–5134, Oct. 2008. [Online]. Available: http://dx.doi.org/10.1109/tsp.2008.928937

[9] L. Chen, P. L. Choyke, T.-H. Chan, C.-Y. Chi, G. Wang, and Y. Wang, "Tissue-specific compartmental analysis for dynamic contrast-enhanced MR imaging of complex tumors," *IEEE Trans. Med. Imag.*, vol. 30, no. 12, pp. 2044–2058, Dec. 2011. [Online]. Available: http://dx.doi.org/10.1109/tmi.2011.2160276

[10] R. Luce, P. Hildebrandt, U. Kuhlmann, and J. Liesen, "Using separable nonnegative matrix factorization techniques for the analysis of time-resolved Raman spectra," *Appl. Spectrosc.*, vol. 70, no. 9, pp. 1464–1475, 2016. [Online]. Available: http://dx.doi.org/10.1177/0003702816662600

[11] N. Gillis, "The why and how of nonnegative matrix factorization," in *Regularization, Optimization, Kernels, and Support Vector Machines* (Machine Learning and Pattern Recognition Series), J. Suykens, M. Signoretto, and A. Argyriou, Eds. London, U.K.: Chapman & Hall, 2014, pp. 257–291. [Online]. Available: http://www.crcpress.com/product/isbn/9781482241396

[12] J. M. P. Nascimento and J. M. Bioucas-Dias, "Vertex component analysis: A fast algorithm to unmix hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 4, pp. 898–910, Apr. 2005. [Online]. Available: http://dx.doi.org/10.1109/tgrs.2005.844293

[13] M. C. U. Araújo, T. C. B. Saldanha, R. K. H. Galvão, T. Yoneyama, H. C. Chame, and V. Visani, "The successive projections algorithm for variable selection in spectroscopic multicomponent analysis," *Chem. Intell. Lab. Syst.*, vol. 57, no. 2, pp. 65–73, 2001. [Online]. Available: http://dx.doi.org/10.1016/s0169-7439(01)00119-8

[14] N. Gillis and S. A. Vavasis, "Fast and robust recursive algorithms for separable nonnegative matrix factorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 698–714, Apr. 2014. [Online]. Available: http://dx.doi.org/10.1109/tpami.2013.226

[15] E. Esser, M. Moller, S. Osher, G. Sapiro, and J. Xin, "A convex model for nonnegative matrix factorization and dimensionality reduction on physical space," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3239–3252, Jul. 2012. [Online]. Available: http://dx.doi.org/10.1109/tip.2012.2190081

[16] V. Bittorf, B. Recht, E. Ré, and J. Tropp, "Factoring nonnegative matrices with linear programs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1223–1231.

[17] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Rev.*, vol. 52, no. 3, pp. 471–501, 2010. [Online]. Available: http://dx.doi.org/10.1137/070697835

[18] X. Fu and W.-K. Ma, "Robustness analysis of structured matrix factorization via self-dictionary mixed-norm optimization," *IEEE Signal Process. Lett.*, vol. 23, no. 1, pp. 60–64, Jan. 2016. [Online]. Available: http://dx.doi.org/10.1109/lsp.2015.2498523

[19] N. Gillis and R. Luce, "Robust near-separable nonnegative matrix factorization using linear optimization," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1249–1280, 2014. [Online]. Available: http://jmlr.org/papers/v15/gillis14a.html

[20] J. G. Liu and S. Aeron, "Robust large-scale non-negative matrix factorization using proximal point algorithm," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Dec. 2013, pp. 1127–1130. [Online]. Available: http://dx.doi.org/10.1109/globalsip.2013.6737093

[21] S. J. Wright and J. Nocedal, *Numerical Optimization*. New York, NY USA: Springer, 1999.

[22] Y. Nesterov, *Introductory Lectures on Convex Optimization* (Applied Optimization), vol. 87. Boston, MA, USA: Kluwer, 2004. [Online]. Available: http://dx.doi.org/10.1007/978-1-4419-8853-9

[23] N. Gillis and F. Glineur, "Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization," *Neural Comput.*, vol. 24, no. 4, pp. 1085–1105, 2012. [Online]. Available: http://dx.doi.org/10.1162/NECO_a_00256

[24] W. Ding, M. H. Rohban, P. Ishwar, and V. Saligrama, "Topic discovery through data dependent and random projections," in *Proc. ICML*, Feb. 2013, pp. 1202–1210.

[25] A. Benson, J. Lee, B. Rajwa, and D. Gleich, "Scalable methods for nonnegative matrix factorizations of near-separable tall-and-skinny matrices," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 945–953.

[26] X. Fu, W.-K. Ma, T.-H. Chan, and J. M. Bioucas-Dias, "Self-dictionary sparse regression for hyperspectral unmixing: Greedy pursuit and pure pixel search are related," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 6, pp. 1128–1141, Sep. 2015. [Online]. Available: http://dx.doi.org/10.1109/jstsp.2015.2410763

[27] N. Gillis, "Successive nonnegative projection algorithm for robust nonnegative blind source separation," *SIAM J. Imag. Sci.*, vol. 7, no. 2, pp. 1420–1450, 2014. [Online]. Available: http://dx.doi.org/10.1137/130946782

[28] J. W. Boardman, "Geometric mixture analysis of imaging spectrometry data," in *Proc. Surf. Atmos. Remote Sens., Technol., Data Anal. Interpretation Int. Geosci. Remote Sens. Symp. (IGARSS)*, vol. 4. Aug. 1994, pp. 2369–2371.

[29] M. E. Winter, "N-FINDR: An algorithm for fast autonomous spectral end-member determination in hyperspectral data," *Proc. SPIE*, vol. 3753, p. 5, Oct. 1999. [Online]. Available: http://dx.doi.org/10.1117/12.366289

[30] N. Gillis, D. Kuang, and H. Park, "Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 4, pp. 2066–2078, Apr. 2015. [Online]. Available: http://dx.doi.org/10.1109/tgrs.2014.2352857

[31] Z. Guo, T. Wittman, and S. Osher, "L1 unmixing and its application to hyperspectral image enhancement," *Proc. SPIE*, vol. 7334, p. 73341M, Apr. 2009. [Online]. Available: http://dx.doi.org/10.1117/12.818245

[32] A. Zare and K. Ho, "Endmember variability in hyperspectral analysis: Addressing spectral variability during spectral unmixing," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 95–104, Jan. 2014.

[33] R. E. Tarjan, *Data Structures and Network Algorithms* (CBMS-NSF Regional Conference Series in Applied Mathematics), vol. 44. Philadelphia, PA, USA: SIAM, 1983. [Online]. Available: http://dx.doi.org/10.1137/1.9781611970265

**Nicolas Gillis** received the master's and the Ph.D. degrees in applied mathematics from the Université catholique de Louvain, Belgium, in 2007 and 2011, respectively. He is currently an Associate Professor with the Department of Mathematics and Operational Research, Faculté polytechnique, Université de Mons, Belgium. His research interests lie in optimization, numerical linear algebra, machine learning, and data mining.

**Robert Luce** received the Ph.D. degree from the Technische Universität Berlin, Germany, in 2014. Since 2014, he has been a Scientist with École Polytechnique Fédérale de Lausanne, Switzerland, where his research is focused on numerical linear algebra, numerical optimization, and efficient implementations.