Fully Constrained Least Squares Spectral Unmixing by Simplex Projection

Rob Heylen, *Member, IEEE*, Dževdet Burazerović, *Member, IEEE*, and Paul Scheunders, *Member, IEEE*

***Abstract*—We present a new algorithm for linear spectral mixture analysis, which is capable of supervised unmixing of hyperspectral data while respecting the constraints on the abundance coefficients. This simplex-projection unmixing algorithm is based upon the equivalence of the fully constrained least squares problem and the problem of projecting a point onto a simplex. We introduce several geometrical properties of high-dimensional simplices and combine them to yield a recursive algorithm for solving the simplex-projection problem. A concrete implementation of the algorithm for large data sets is provided, and the algorithm is benchmarked against well-known fully constrained least squares unmixing (FCLSU) techniques, on both artificial data sets and real hyperspectral data collected over the Cuprite mining region. Unlike previous algorithms for FCLSU, the presented algorithm possesses no optimization steps and is completely analytical, severely reducing the required processing power.**

***Index Terms*—Hyperspectral imaging, multidimensional signal processing, spectral analysis.**

I. Introduction

SPECTRAL UNMIXING of hyperspectral imagery is an important subject in the field of remote sensing and image processing and has received significant attention in the past decades. Spectral unmixing concerns the decomposition of a pixel spectrum into its constituent spectra, with the motivation being that the surface area of a single pixel contains a low number of "pure" materials, each with a given fraction [1]. Usually, both the spectra of the pure materials, called endmembers, as well as their abundances in each pixel, are considered unknown. Most algorithms for spectral unmixing therefore need to perform at least two tasks, one task concerned with finding the unknown endmember spectra and a second one of determining the corresponding abundances of each endmember in each pixel. The former task is labeled as endmember extraction, while the latter is labeled as unmixing, the inversion problem, or spectral mixture analysis. By combining an endmember extraction algorithm (EEA) with a spectral mixture analysis algorithm, one yields a holistic approach to the spectral unmixing problem. Some approaches (e.g., [2]–[4]) integrate both tasks in a single algorithm (unsupervised unmixing), while most algorithms first look for the endmembers, after which these endmembers are used in a supervised unmixing algorithm for determining the abundances.

Most spectral unmixing algorithms start from the linear mixing assumption [1], or the assumption that an observed spectrum is a linear combination of a limited number of endmember spectra, where the linear coefficient of each endmember is its abundance. These abundances have to obey two constraints, derived from their physical interpretation: They have to be positive, since no endmember material can have a negative presence in a pixel's field of view, and they have to sum to one, indicating that the spectrum of every pixel can be completely decomposed into the contributions of the endmembers, and nothing else. This linear model, together with the constraints on the abundances, has proven to be very successful in practice, and many algorithms for spectral unmixing that start from these assumptions exist. See [5] for a detailed taxonomy of linear spectral unmixing algorithms and [6] and [7] for recent surveys of unmixing algorithms.

Most of the linear unmixing algorithms exploit the fact that, due to the linear mixing equation, the mixed pixels have to lie inside the convex hull of the endmembers [8]. This convex hull forms a simplex in the spectral space, with the endmembers as spanning vertices. Some algorithms then try to look for the largest volume embedded simplex (NfindR [9] and simplex growing algorithm [10]) or try to identify extreme points in the data cloud (e.g., pixel purity index [8], orthogonal subspace projection [11], convex cone analysis [12], and vertex component analysis [13]). Under the assumption that the endmembers are present in the data as pure pixels, these strategies are able to locate these pure pixels, hence identifying the endmembers. Many other algorithms exist, and many of them do not require the endmembers to be present in the data [6].

Most of these algorithms are mainly concerned with endmember extraction. Once the endmembers have been found, the estimation of the abundances requires the inversion of the linear mixing equation, since one only has the data points and the endmembers available, and one would like to estimate the linear coefficients (the abundances). Due to the presence of noise in the data and possibly small errors in the estimated endmember spectra, this problem has no straightforward analytical solution. Furthermore, one would like to obey the restrictions imposed on the abundance coefficients (positivity and sum to one), further complicating the matter.

Over time, many algorithms have been proposed for inverting the linear mixing equation, effectively unmixing the data points

Manuscript received September 23, 2010; revised January 31, 2011 and April 28, 2011; accepted May 8, 2011. Date of publication June 16, 2011; date of current version October 28, 2011.

The authors are with the Interdisciplinary Institute for Broadband Technology (IBBT), Visionlab, Department of Physics, University of Antwerp, 2610 Antwerp, Belgium (e-mail: rob.heylen@ua.ac.be; dzevdet.burazerovic@ua.ac.be; paul.scheunders@ua.ac.be).

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/TGRS.2011.2155070

into their abundances. Originally, one used simple least squares linear regression for solving the overdetermined system of linear equations [14]. Unfortunately, these solutions did not obey any of the constraints on the abundances. However, it proved easy to also incorporate the sum-to-one constraint, leaving the problem to be solved as a nonnegatively constrained least squares (NNLS) problem [2]. Further improvements could then be reached via a zeroing procedure, where negative abundances were replaced with zero, and normalization of the resulting positive abundance vectors. Orthogonal subspace projection leads to unmixing algorithms via subspace projection methods [15]. In [16], a weighted least squares method was proposed, and here, it was also observed that solving the unmixing problem with all constraints obeyed would require quadratic programming. None of these techniques, however, is able to find the optimal solution.

Then, in [2], a technique that first turns the unmixing problem into an NNLS problem by introduction of a Lagrangian was proposed, after which this problem is solved using the techniques developed by Lawson and Hanson (the NNLS problem [17]). This was the first algorithm that finds the least squares solution that obeys both the positivity and sum-to-one constraints, yielding the fully constrained least squares unmixing (LSU) (FCLSU) algorithm. Soon after, other algorithms that also allowed finding this solution followed, however through some other approach. In [18] for instance, one first notes the equivalence between solving the FCLSU problem and the simplex-projection problem, after which a quadratic programming algorithm is used for solving the latter problem in the context of spectral unmixing. Quadratic programming has also been successfully used in other unmixing algorithms, e.g., [19] and [20]. In [21], one transforms the formulation of the least squares problem into a least distance problem, which subsequently can be solved by the Lawson–Hanson NNLS algorithm. In [2] and [3], a combined algorithm is proposed, which is capable of simultaneous FCLSU and endmember extraction by extremizing reconstruction errors. More recently, a hierarchical Bayesian model has been proposed [4], [22] for simultaneous endmember extraction and solving the inversion problem, again allowing one to find the optimal solution to the FCLSU problem. Another approach treats the problem as a blind source separation problem and uses Bayesian techniques [23]–[25] or nonnegative matrix factorization [26], [27] to find the endmembers and abundances. In [28], it has been shown that also fuzzy set theory and fuzzy logic can be successfully used for solving the inversion problem.

Most of the approaches that provide the exact FCLSU solution use an optimization or maximization procedure, gradient or steepest descent methods, or require a random sampler from some distribution. For example, the Lawson–Hanson NNLS algorithm and quadratic programming techniques are both optimization techniques, requiring appropriate starting vectors and step sizes. Bayesian techniques depend on random samplers. Although the results obtained with these algorithms can be made arbitrarily accurate by modifying the algorithmic parameters, they do not provide an analytically correct solution based upon some mathematical principles but resort to optimization strategies.

In this paper, we present a new supervised unmixing algorithm, the simplex-projection unmixing (SPU) algorithm, based upon the equivalence between the FCLSU problem and the simplex-projection problem [18], [29]. We present an algorithm that is capable of finding the projection of a point onto a generic simplex without resorting to quadratic programming or NNLS techniques. To our knowledge, the algorithms for simplex projection present in the literature either employ quadratic programming [18], [30] or project only on canonical [31] simplices. The solution obtained via this simplex-projection algorithm almost always minimizes the least squares error while strictly obeying the constraints on the abundances. The algorithm is a recursive algorithm that reduces the dimensionality of the problem by one with each recursion until an appropriate abundance vector is found. At each step of the recursion, we identify an endmember with zero abundance and perform an orthogonal projection onto a hyperplane of one dimension less than the previous recursion. Because it does not resort to some optimization or maximization procedure, it has a very high computational performance, and in real spectral unmixing situations, the produced results are comparable to the exact FCLSU solution.

The outline of this paper is as follows. In Section II, we define the problem of spectral unmixing and explain the equivalence between the fully constrained least squares problem and the simplex-projection problem. In Section III, we first present the reasoning that leads to the algorithm, along with the various mathematical concepts needed. We then formally present the algorithm, followed by an explanation of every step, and a concrete implementation for large data sets. In Section IV, we present the results obtained by the algorithm and compare the computational performance of this algorithm with that of the well-known FCLSU algorithm presented in [2], both on a synthetic data set and on the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) Cuprite data set. Section V contains the conclusions and future work.

II. FCLSU and Simplex Projection

Consider a pixel $\boldsymbol{x}$ in a hyperspectral image with d spectral bands: $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$. In the linear mixing model, one assumes that every pixel consists of a linear combination of p endmembers $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$ and added noise. We assume that $p < d$

$$\boldsymbol{x} = \sum_{i=1}^{p} a_i \boldsymbol{e}_i + \boldsymbol{\epsilon}. \tag{1}$$

The coefficients a_i represent the abundances of each endmember in pixel $\boldsymbol{x}$. The two constraints on these abundance coefficients, positivity and sum to one, then become

$$\forall i : a_i \geq 0 \tag{2}$$

$$\sum_{i=1}^{p} a_i = 1. \tag{3}$$

The sum-to-one constraint is often relaxed to $\sum_{i=1}^{p} a_i \leq 1$, but the same effect can be obtained by artificially including an

additional endmember that represents shade: $\boldsymbol{e}_{p+1} = \overline{0}$. Hence, in this paper, we assume that the sum-to-one constraint (3) has to be strictly obeyed.

The spectral unmixing problem tries to solve the following problem: If the endmembers $\{\boldsymbol{e}_i\}_i$ are known, find the abundances $\{a_i\}_i$ so that (1) holds. When the noise equals zero ($\boldsymbol{\epsilon} = \overline{0}$), this problem becomes trivial; since then, we have an overdetermined system of d linear equations in p variables for every pixel $\boldsymbol{x}$, with $d > p$. However, when the noise is nonzero, one usually cannot satisfy (1) for any set of abundances $\{a_i\}_i$.

Since the noise $\boldsymbol{\epsilon}$ is usually assumed to be independent and identically distributed Gaussian noise with mean zero and a covariance matrix proportional to the unity matrix, the simplest method for approximating the solution of an overdetermined linear system that cannot be solved exactly is by simple least squares linear regression, yielding LSU. This method minimizes the sum of squared differences between the observed spectral values x_i and the reconstructed values predicted by the linear approximation, without any constraints on the abundance coefficients. With $\hat{a}_i$, the estimate for a_i, we find

$$\{\hat{a}_i\}_{\mathrm{LSU}} = \underset{\{\hat{a}_i\}_i}{\arg\min} \|\boldsymbol{x}'(\hat{a}_1, \ldots, \hat{a}_p) - \boldsymbol{x}\|_2 \tag{4}$$

where we introduced the reconstructed data point

$$\boldsymbol{x}' = \sum_{i=1}^{p} \hat{a}_i \boldsymbol{e}_i. \tag{5}$$

Stated otherwise, the LSU solution provides us with the unique linear combination of endmembers with the smallest $\mathcal{L}_2$-distance to the data point $\boldsymbol{x}$. If we define the reconstruction error as

$$\Delta = \|\boldsymbol{x}' - \boldsymbol{x}\|_2 \tag{6}$$

one can also state that the LSU solution minimizes the reconstruction error. The solution to the LSU problem can be easily found via the pseudoinverse of the endmember matrix. With $\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_p]$

$$\hat{\boldsymbol{a}} = (\boldsymbol{E}^{\mathrm{T}}\boldsymbol{E})^{-1}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{x}. \tag{7}$$

With this, we can write the reconstructed data point $\boldsymbol{x}'$ as

$$\boldsymbol{x}' = \boldsymbol{E}\hat{\boldsymbol{a}} = \boldsymbol{E}(\boldsymbol{E}^{\mathrm{T}}\boldsymbol{E})^{-1}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{x}. \tag{8}$$

Hence, the solution is provided by a projection operator $P_{\mathrm{LSU}} = \boldsymbol{E}(\boldsymbol{E}^{\mathrm{T}}\boldsymbol{E})^{-1}\boldsymbol{E}^{\mathrm{T}}$, which orthogonally projects onto a subspace spanned by the columns of $\boldsymbol{E}$, which are the endmembers: $\boldsymbol{x}' = P_{\mathrm{LSU}}(\boldsymbol{x})$.

The main problem with the LSU solution is that the constraints on the abundances (2) and (3) are not necessarily obeyed. Adding some constraints on the allowed values for the abundances corresponds to looking for a reconstructed point in only a subspace of the linear space spanned by the endmembers. If both constraints have to be obeyed exactly, this subspace will be the convex hull of the p endmembers, which is the simplex spanned by these endmembers. Indeed, the positivity and sum-to-one constraints effectively turn the abundances into normalized barycentric coordinates with respect to the endmembers, yielding a simplex lying on a $(p-1)$-dimensional plane in the d-dimensional spectral space.

The reconstructed point $\boldsymbol{x}'$ corresponding to the FCLSU solution now equals the point inside this endmember simplex that minimizes the reconstruction error Δ, or the $\mathcal{L}_2$-distance, with the actual data point $\boldsymbol{x}$. If we define the simplex S as

$$\boldsymbol{y} \in S \iff \exists a_1, \ldots, a_p \in [0,1] : \begin{cases} \boldsymbol{y} = \sum_{i=1}^{p} a_i \boldsymbol{e}_i \\ \sum_{i=1}^{p} a_i = 1 \end{cases} \tag{9}$$

then we can introduce an operator P, projecting a point $\boldsymbol{x}$ onto the closest point inside the simplex S, as

$$\boldsymbol{x}' \in S : \boldsymbol{x}' = P(\boldsymbol{x}) \iff \forall \boldsymbol{y} \in S : \|\boldsymbol{x} - \boldsymbol{y}\|_2 \geq \|\boldsymbol{x} - \boldsymbol{x}'\|_2. \tag{10}$$

The operator P is idempotent [$\forall \boldsymbol{x} : P(P(\boldsymbol{x})) = P(\boldsymbol{x})$], making it a projection operator. Due to its construction, the projected point $\boldsymbol{x}'$ lies in the simplex S, and by solving $\boldsymbol{x}' = \sum_i \hat{a}_i \boldsymbol{e}_i$, which now is an exactly solvable overdetermined linear system, we can find the corresponding abundance coefficients $\hat{\boldsymbol{a}}$. Hence, there exists a clear equivalence between the FCLSU problem and projecting a point onto a simplex: The reconstructed point from the FCLSU solution equals the projected point onto the simplex. Similarly, the abundance coefficients of the projected point, which are normalized barycentric coordinates in the endmember coordinate system, are the same as those of the FCLSU solution. In the next section, we present an algorithm for performing the projection operation P.

III. SPU Algorithm

A. Prerequisites

The presented method for projecting a point $\boldsymbol{x}$ onto a simplex S and finding the corresponding abundance coefficients rests on several observations. We list these observations here, together with a proof, and present the reasoning leading to the algorithm.

Lemma 3.1: Orthogonal projection onto the simplex plane (the hyperplane through the endmembers) leaves the simplex projection invariant.

Proof: Consider a simplex S in $\mathbb{R}^d$ spanned by p endmembers $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$. Let $\boldsymbol{y}$ be the orthogonal projection of a point $\boldsymbol{x}$ onto the plane through the endmember points $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$, and let $\boldsymbol{s}$ be any point inside the simplex S. We then have that $\|\boldsymbol{x} - \boldsymbol{s}\|_2^2 = \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \|\boldsymbol{y} - \boldsymbol{s}\|_2^2$ because the vectors $\boldsymbol{x} - \boldsymbol{y}$ and $\boldsymbol{y} - \boldsymbol{s}$ are orthogonal. Since $\|\boldsymbol{x} - \boldsymbol{y}\|_2$ is constant, the $\boldsymbol{s}$ that minimizes $\|\boldsymbol{x} - \boldsymbol{s}\|_2$ will also minimize $\|\boldsymbol{y} - \boldsymbol{s}\|_2$. Hence, $P(\boldsymbol{x}) = P(\boldsymbol{y})$, and orthogonal projection onto the simplex plane is an invariant operation for the simplex-projection operator P. ■

A consequence of this lemma is that we can always restrict ourselves to points lying in the $(p-1)$-dimensional simplex plane, instead of in the full d-dimensional space, by replacing $\boldsymbol{x}$ by its orthogonal projection onto this simplex plane. The coordinates of the points, however, are still d-dimensional.

The next lemma states that the projection of a point outside a simplex always has at least one abundance coefficient zero.

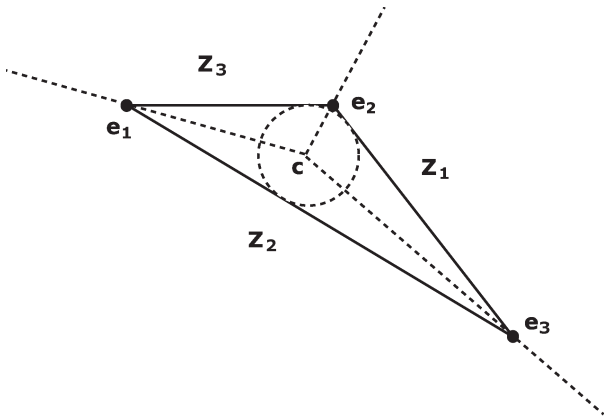

Fig. 1. Generic 2-D simplex, with the incenter and the three bissective cones indicated. A point in a cone Z_i will have a simplex projection with $\hat{a}_i = 0$.

Lemma 3.2: Consider a simplex S in $\mathbb{R}^{p-1}$ spanned by p endmembers $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$ and a point $\boldsymbol{x} \notin S$. The projection $\boldsymbol{x}' = P(\boldsymbol{x})$ lies on the boundary of the simplex S and hence has at least one abundance coefficient $\hat{a}_i = 0$.

Proof: Suppose that $\boldsymbol{x}'$ has only nonzero abundance coefficients. Then, $\boldsymbol{x}'$ will lie in the interior of S. The line segment connecting $\boldsymbol{x}'$ with $\boldsymbol{x}$ will intersect the boundary of the simplex S in a point $\boldsymbol{y}$ between $\boldsymbol{x}'$ and $\boldsymbol{x}$, and $\|\boldsymbol{x} - \boldsymbol{y}\|_2 < \|\boldsymbol{x} - \boldsymbol{x}'\|_2$, proving that $\boldsymbol{x}'$ cannot be the projection of $\boldsymbol{x}$ onto S. ■

The aim next is to be able to identify the abundance coefficient that has to be zero. For this, we first introduce the incenter of a simplex.

Definition: The incenter $\boldsymbol{c}$ of a simplex S spanned by $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$ is defined as the intersection of all $(p-2)$-dimensional planes that bisect the dihedral angles between the faces of the simplex. This is also the center of the largest possible hypersphere that one can inscribe in the simplex. See Fig. 1.

The coordinates of the incenter $\boldsymbol{c}$ can be found via the $(p-2)$-dimensional volumes of the p faces of the simplex. Let V_i be the volume of the subsimplex spanned by $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{i-1}, \boldsymbol{e}_{i+1}, \ldots, \boldsymbol{e}_p\}$. Then, we have [32]

$$a_i^c = \frac{V_i}{\sum_{i=1}^{p} V_i} \tag{11}$$

yielding a vector $\boldsymbol{a}^c$ containing the barycentric coordinates of the incenter $\boldsymbol{c}$. The Euclidean coordinates of the incenter are then given by

$$\boldsymbol{c} = \boldsymbol{E}\boldsymbol{a}^c. \tag{12}$$

Consider a point $\boldsymbol{x}$ lying in the simplex plane. The ray starting from the incenter $\boldsymbol{c}$ through $\boldsymbol{x}$ will intersect a certain face of the simplex. We define the set of all points intersecting a given face as the bissective cone of that face.

Definition: The bissective cone Z_i of a simplex S in $\mathbb{R}^{p-1}$ spanned by $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$ is the set of points defined by

$$\boldsymbol{x} \in Z_i \Leftrightarrow \exists b_1, \ldots, b_p \geq 0 : \begin{cases} \boldsymbol{x} = \boldsymbol{c} + \sum_{j=1}^{p} b_j(\boldsymbol{e}_j - \boldsymbol{c}) \\ b_i = 0 \end{cases} \tag{13}$$

with $\boldsymbol{c}$ being the incenter of S.

In Fig. 1, the concept of bissective cones is illustrated for a simplex in the 2-D plane. It is clear that these cones form a partitioning of the $(p-1)$-dimensional simplex plane

$$\bigcup_i Z_i = \mathbb{R}^{p-1}. \tag{14}$$

These bissective cones allow one to estimate the abundance coefficient that has to be zero. We first present the following theorem, of which we will use an extension to generic simplices later.

Theorem 3.3: Consider the canonical simplex S_c in $\mathbb{R}^p$: $\boldsymbol{v} \in S_c \iff (\sum_i v_i = 1 \text{ and } \forall i : v_i \geq 0)$ and a point $\boldsymbol{x}$ in the simplex plane, with $\boldsymbol{x} \notin S$. The projection of $\boldsymbol{x}$ on the simplex S_c is $\boldsymbol{x}' = P(\boldsymbol{x})$, with barycentric coordinates $\hat{a}_1, \ldots, \hat{a}_p$ (abundances). We then have

$$\boldsymbol{x} \in Z_i \Rightarrow \hat{a}_i = 0. \tag{15}$$

Proof: First of all, remark that barycentric coordinates equal Euclidean coordinates in the canonical simplex. To prove the theorem, we use the following lemma proven in [33] and also used in [34].

Consider the canonical simplex S_c in $\mathbb{R}^p$: $\boldsymbol{v} \in S_c \iff (\sum_i v_i = 1 \text{ and } \forall i : v_i \geq 0)$. Let $\boldsymbol{y} \in S_c$ be the vector that minimizes $\|\boldsymbol{y} - \boldsymbol{x}\|_2$ for a given $\boldsymbol{x}$: $\boldsymbol{y} = P(\boldsymbol{x})$. Let i and j be two indices such that $x_i > x_j$. We then have that $y_i = 0 \Rightarrow y_j = 0$.

In the context of simplex projection, we know (see Lemma 3.1) that any point $\boldsymbol{x}$ can be replaced with its orthogonal projection on the endmember plane without affecting the projection onto the simplex. Hence, we can restrict ourselves to points $\boldsymbol{x}$ lying in this simplex plane. We also know that, if a point $\boldsymbol{x}$ lies in the simplex plane but outside the simplex S_c, there has to be at least one barycentric coordinate (or abundance coefficient) that is zero (Lemma 3.2). A corollary of the above lemma is then that the smallest x_i certainly corresponds to a zero barycentric coordinate

$$\boldsymbol{x} \notin S_c, x_i = \min(x_1, \ldots, x_p) \Rightarrow y_i = 0. \tag{16}$$

See Figs. 2 and 3 for an illustration of this. Now, the bissective cones Z_i correspond exactly to those subsets where a given coordinate x_i becomes minimal. Hence, a consequence of this corollary is that, at least for the canonical simplex, every point $\boldsymbol{x} \notin S_c$, $\boldsymbol{x} \in Z_i$ will have its corresponding barycentric coordinate $\hat{a}_i = 0$. ■

In the proposed algorithm, we extend the previous theorem to generic simplices, where we replace the origin with the incenter. Hence, consider a simplex S in $\mathbb{R}^{p-1}$ spanned by p endmembers $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$ and a point $\boldsymbol{x} \notin S$. The projection of $\boldsymbol{x}$ is $\boldsymbol{x}' = P(\boldsymbol{x})$, with barycentric coordinates $\hat{a}_1, \ldots, \hat{a}_p$ (abundances). We then assume that

$$\boldsymbol{x} \in Z_i \Rightarrow \hat{a}_i = 0. \tag{17}$$

This assumption is not true in general: Although, for $p = 3$, the assumption seems to hold, one can generate counterexamples for any $p > 3$. However, these counterexamples usually involve simplices that are not well shaped: Their volume is very low

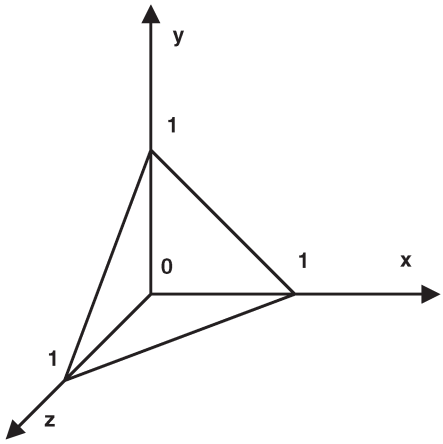

Fig. 2. Canonical simplex S_c in $\mathbb{R}^3$, spanned by the three endmembers $\{(1,0,0),(0,1,0),(0,0,1)\}$. All points $\boldsymbol{y} \in S_c$ lie on the 2-D triangle spanned by the endmembers.

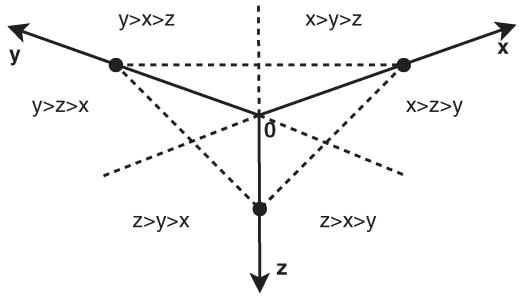

Fig. 3. Two-dimensional plane of the canonical simplex S_c in $\mathbb{R}^3$, where the origin equals the incenter. The six different possibilities for the coordinates of a point $\boldsymbol{x}$ in this plane, with $\boldsymbol{x} \notin S_c$, are indicated. The barycentric coordinate of $P(\boldsymbol{x})$ corresponding to the smallest coordinate has to be zero.

compared to the smallest intervertex distance, and some dihedral angles are very obtuse. Such situations will not occur often in real hyperspectral data. Furthermore, for higher values of p, it becomes very hard to find counterexamples, and this assumption can be used in practice.

Since this assumption now allows us to estimate which abundance has to be zero, we can construct a recursive algorithm: Suppose that $\boldsymbol{x}$ lies outside a simplex S but in the simplex plane. Since a certain known endmember $\boldsymbol{e}_i$ will not contribute to the projection $\boldsymbol{x}'$ of $\boldsymbol{x}$, we know that $\boldsymbol{x}'$ has to lie on the subsimplex S_i spanned by all endmembers except $\boldsymbol{e}_i$. Hence, projecting onto this subsimplex S_i instead of the full simplex S will yield the same abundance coefficients with respect to the endmembers spanning S_i. Orthogonal projection onto the plane of S_i reduces the implicit dimensionality of the problem by one, and one can recursively continue until either one finds a projected point lying inside a simplex or one finishes with a trivial 1-D problem. In the next section, we formally present this algorithm.

B. Algorithm

The observations in the previous section lead to the following algorithm for projecting a point onto a simplex and finding the corresponding abundance coefficients. Suppose that one has a point $\boldsymbol{x} \in \mathbb{R}^d$ that one wants to project onto a simplex S_I spanned by the p endmembers in the set $I = \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$.

1) Orthogonally project the point $\boldsymbol{x}$ onto the simplex plane, determined by the endmembers in I, yielding $\boldsymbol{y}$.
2) If $\boldsymbol{y}$ lies in the simplex S_I, go to step 4. Else, go to the next step.
3) Determine the cone Z_i that contains $\boldsymbol{y}$. Remove $\boldsymbol{e}_i$ from the set I, yielding $I' = I/\{\boldsymbol{e}_i\}$ and a new subsimplex $S_{I'}$ of one dimension less. Put the abundance coefficient $\hat{a}_i = 0$. Set $I = I'$ and $\boldsymbol{x} = \boldsymbol{y}$, and go to step 1.
4) Determine the abundance coefficients by writing $\boldsymbol{y}$ as a linear combination of the remaining endmembers in I. This is now an overdetermined and exactly solvable system of linear equations, respecting the constraints on the abundances. All abundances that correspond to an endmember not in I are zero.

The first step of the algorithm is to find the orthogonal projection $\boldsymbol{y}$ of a point $\boldsymbol{x}$ onto the simplex plane of S_I. This projection can be obtained by adapting (8), describing projection onto the subspace spanned by the endmember vectors

$$\hat{\boldsymbol{E}} = [\boldsymbol{e}_2 - \boldsymbol{e}_1, \boldsymbol{e}_3 - \boldsymbol{e}_1, \ldots, \boldsymbol{e}_p - \boldsymbol{e}_1] \tag{18}$$

$$\boldsymbol{y} = \hat{\boldsymbol{E}}(\hat{\boldsymbol{E}}^{\mathrm{T}}\hat{\boldsymbol{E}})^{-1}\hat{\boldsymbol{E}}^{\mathrm{T}}(\boldsymbol{x} - \boldsymbol{e}_1) + \boldsymbol{e}_1. \tag{19}$$

The above equations are also valid for any permutation of the p endmembers. This projection corresponds to a translation of $-\boldsymbol{e}_1$, which puts the endmember $\boldsymbol{e}_1$ at the origin, then projecting the translated point $\boldsymbol{x} - \boldsymbol{e}_1$ onto the space spanned by the $(p-1)$ other translated endmembers $\boldsymbol{e}_2 - \boldsymbol{e}_1, \ldots, \boldsymbol{e}_p - \boldsymbol{e}_1$ (this is the plane through these translated endmembers and the origin) and performing the inverse translation from the origin to $\boldsymbol{e}_1$, yielding the projected point $\boldsymbol{y}$.

Equation (19) also allows us to determine whether a point lies inside a simplex or not, which is needed in the second step of the algorithm. Furthermore, it allows us to find the abundances of a point inside a simplex as well, as needed in the fourth step of the algorithm. First, we define the partial abundance coefficients $\boldsymbol{v}$ as

$$\boldsymbol{v} = (\hat{\boldsymbol{E}}^{\mathrm{T}}\hat{\boldsymbol{E}})^{-1}\hat{\boldsymbol{E}}^{\mathrm{T}}(\boldsymbol{x} - \boldsymbol{e}_1). \tag{20}$$

If all v_i's, with $i = 1, \ldots, p-1$, are positive and $\sum_i v_i \leq 1$, the orthogonal projection $\boldsymbol{y}$ of the point $\boldsymbol{x}$ lies in the simplex, and the corresponding abundance coefficients $\hat{\boldsymbol{a}}$ are given by the concatenation of $\hat{a}_1 = 1 - \sum_i v_i$ and the vector $\boldsymbol{v}$

$$\hat{\boldsymbol{a}} = \left[1 - \sum_i v_i, \boldsymbol{v}\right]. \tag{21}$$

In the third step, we need to determine the cone Z_i that contains $\boldsymbol{x}$. To determine this cone, we must first calculate the location of the incenter $\boldsymbol{c}$, which can be done via (11) and (12). We must still calculate the volumes of the p subsimplices however, where the vertices are given as d-dimensional coordinate vectors. A straightforward method for calculating these

volumes that is independent of the dimensionality d of the data is by using the Cayley–Menger determinant. Let d_{ij} be the Euclidean distance between points $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$. The volume V of a simplex spanned by endmembers $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p$ can then be found via

$$(-1)^p 2^{p-1} \left((p-1)!\right)^2 V^2 = \det(\boldsymbol{C}_{1,2,\ldots,p}) \tag{22}$$

$$\boldsymbol{C}_{1,2,\ldots,p} = \begin{pmatrix} \boldsymbol{D}_{1,2,\ldots,p} & \mathbf{1} \\ \mathbf{1} & 0 \end{pmatrix} \tag{23}$$

with $\boldsymbol{D}_{1,2,\ldots,p} = [d_{ij}^2]_{i,j=1,2,\ldots,p}$. This equation is valid for any number of endmembers p and has no explicit dependence on the dimensionality of the data d.

Once we have determined the incenter $\boldsymbol{c}$, we can find the cone Z_i that contains $\boldsymbol{x}$. If we define

$$\boldsymbol{x}^c \equiv \boldsymbol{x} - \boldsymbol{c} \tag{24}$$

$$\boldsymbol{e}_i^c \equiv \boldsymbol{e}_i - \boldsymbol{c} \tag{25}$$

$$\boldsymbol{E}_i^c \equiv \left[\boldsymbol{e}_1^c, \ldots, \boldsymbol{e}_{i-1}^c, \boldsymbol{e}_{i+1}^c, \ldots, \boldsymbol{e}_p^c\right] \tag{26}$$

then the linear system of equations

$$\boldsymbol{x}^c = \boldsymbol{E}_i^c \boldsymbol{b}^i \tag{27}$$

will have a solution for every index $i \in \{1, \ldots, p\}$. Because of the definition of bissective cone (13), we find

$$\boldsymbol{x} \in Z_i \iff \forall j : b_j^i \geq 0. \tag{28}$$

These observations result in the following algorithm for determining the cone Z_i that contains $\boldsymbol{x}$.

1) Calculate the incenter $\boldsymbol{c}$ and the entities (24)–(26).
2) Solve (27) for every i, yielding a set of vectors $\{\boldsymbol{b}^i\}_{i=1,\ldots,p}$.
3) The set $\boldsymbol{b}^i$ with only positive entries identifies the cone that contains $\boldsymbol{x}$.

To solve (27) for $\boldsymbol{b}^i$, it suffices to consider only a $(p-1) \times (p-1)$ linearly independent submatrix of $\boldsymbol{E}_i^c$ instead of the full $d \times (p-1)$ matrix and to use matrix inversion.

In the case of $p = 2$, the algorithm could be simplified by observing that the incenter is located in the middle between the two endmembers making up the simplex.

C. Implementation for Large Data Sets

In the previous section, we have described how the algorithm functions for projecting a single point onto a simplex. In practice however, one usually needs to perform this operation for a large number of points. Since many steps of the algorithm contain operations involving only the endmembers, a large performance boost can be gained by processing all data points at the same time. We provide a concrete implementation here for the simplex-projection algorithm executed on large data sets. Hence, consider a data set of N points $\boldsymbol{x}_n \in \mathbb{R}^d$, $n = 1, \ldots, N$, and p endmembers $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p\}$. We want to project all points $\boldsymbol{x}_n$ onto the simplex spanned by these p endmembers, yielding the projected points $\boldsymbol{x}'_n$ and their corresponding abundance vectors $\hat{\boldsymbol{a}}_n = (a_{n1}, \ldots, a_{np})$. The projected or reconstructed points $\boldsymbol{x}'_n$ are found via $\boldsymbol{x}'_n = E\hat{\boldsymbol{a}}_n$.

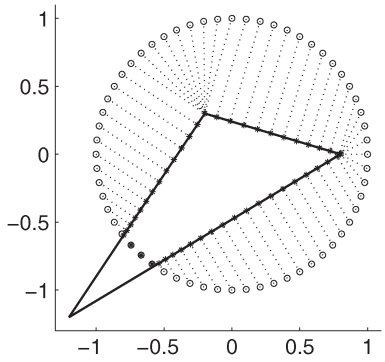

Fig. 4. (Circles) Points on the unit circle are projected onto a (triangle) randomly chosen simplex. The points are connected to their (stars) projections with dotted lines for clarity.

Algorithm 1: simplex_project

input: $\boldsymbol{x}$, E
$\boldsymbol{x}$: $d \times N$ matrix, the N points to unmix
E: $d \times p$ matrix, the p endmembers
output: $\boldsymbol{a}$
$\boldsymbol{a}$: $p \times N$ vector, the abundances
begin
  if $p = 1$ **then**
    $\boldsymbol{a} = \overline{1}$;
    return;
  $I = \{\}$;
  for $\forall n \in [1, \ldots, N]$ **do**
    Calculate partial abundance $\boldsymbol{v}^n$ with (20);
    if $\forall j : v_j^n \geq 0$ *and* $\sum_j v_j^n \leq 1$ **then**
      $\boldsymbol{a}_n = [1 - \sum_j v_j^n, \boldsymbol{v}^n]$;
    else
      Project $\boldsymbol{x}_n$ onto simplex plane with (19);
      Add n to I;
  Calculate endmember distance matrix D;
  for $\forall i \in [1, \ldots, p]$ **do**
    Calculate volume V_i of subsimplex
    $E_i = [\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{i-1}, \boldsymbol{e}_{i+1}, \ldots, \boldsymbol{e}_p]$ with (22);
  Calculate the incenter $\boldsymbol{c}$ with (11) and (12);
  for $\forall i \in [1, \ldots, p]$ **do**
    $I_i = \{\}$;
    for $\forall j \in I$ **do**
      Solve $\boldsymbol{x}_j = \boldsymbol{E}_i^c \boldsymbol{b}^i$ for $\boldsymbol{b}^i$;
      if $\forall k : \boldsymbol{b}_k^i \geq 0$ **then**
        Add j to I_i;
    if $I_i \neq \{\}$ **then**
      $\boldsymbol{a}^r =$ simplex_project$(\boldsymbol{x}_{I_i}, E_i)$;
      $\boldsymbol{a}([1, \ldots, i-1, i+1, \ldots, p], I_i) = \boldsymbol{a}^r$;
      $\boldsymbol{a}(i, I_i) = \overline{0}$;
end

As a simple illustration, we have projected a number of points on the unit circle onto a random simplex with this algorithm in Fig. 4.

IV. Performance of the Algorithm

A. Complexity

The algorithm always finishes in a finite number of recursions, where every recursion contains a finite number of steps. The number of recursions needed however is highly dependent on the data. In the worst case scenario, the number of recursions needed equals $\min(N(p-1), 2^p - 2)$, where N is the number of data points and $2^p - 2$ is the number of subsimplices in a simplex spanned by p endmembers. In the spectral unmixing problem however, most pixels lie inside or close to the endmember simplex, significantly reducing the number of iterations required.

Within a single recursion of the algorithm, the main computational costs of the different steps of the algorithm are as follows.

1) Simplex plane projection requires matrix inversion and multiplication of a $p \times p$ matrix and matrix multiplication of a $p \times d$ matrix with a $d \times N$ matrix.
2) Determining the location of the incenter involves the calculation of p determinants of symmetric and positive definite matrices of size $(p-1) \times (p-1)$.
3) Determining the vectors $\boldsymbol{b}^i$ for a given cone Z_i involves inversion of a $(p-1) \times (p-1)$ matrix and matrix multiplication of a $(p-1) \times (p-1)$ matrix and a $(p-1) \times N$ matrix. This has to be executed p times.
4) All other steps are steps involving index management or have a negligible computational complexity.

We did not list the step where the intervertex distances are calculated, since this can be executed once at the beginning of the algorithm, and can be passed between the recursive calls, since this matrix does not change. The total complexity of the algorithm depends strongly on the implementation details and optimizations used and is hard to determine explicitly. We rather quantify the dependence of runtime on the dimensionality of the problem in a practical way by explicitly running the algorithm for several (simulated and real) data sets and determining the runtime and the number of recursive calls needed for completion.

B. Execution on an Artificial Data Set

As a first experiment, we test the performance of the algorithm on an artificial data set representing simulated hyperspectral data that are linearly mixed with known endmembers and abundances. The data have been generated in the following way. Choose the number of points N, the number of endmembers p, the dimensionality of the data d, and a signal-to-noise ratio (SNR). Generate the $d \times p$ matrix E of random numbers in the interval [0, 1], containing columnwise the p endmembers. Then, generate a $p \times N$ abundance matrix $\boldsymbol{A}$ where every column is a random sampling from the canonical simplex, hence respecting the positivity and sum-to-one constraints. For each column $\boldsymbol{a}$ of $\boldsymbol{A}$, this can be done by sampling p variables a_i independently and uniformly in [0, 1], returning $-\log(a_i)$ (yielding an exponential distribution), and normalizing. Finally, we generate a $d \times N$ noise matrix $\boldsymbol{\eta}$ of random Gaussian numbers with a given SNR. According to Harsanyi and Chang [11], the SNR is defined as half the reflectance divided by the standard deviation of the noise. The data set $\boldsymbol{x}$ is then given by

$$\boldsymbol{x} = \boldsymbol{E}\boldsymbol{A} + \boldsymbol{\eta}. \tag{29}$$

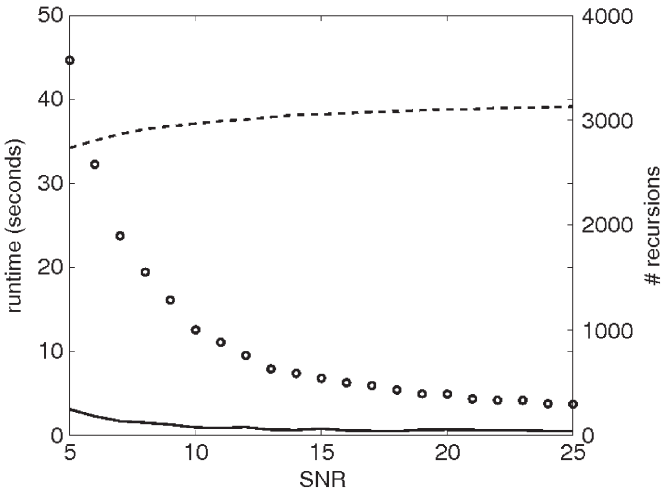

Fig. 5. (Left axis) Runtimes for the (solid line) SPU algorithm and the (dashed line) FCLSU algorithm as a function of SNR. (Right axis) (Circles) Number of recursions in the SPU algorithm. The parameters are $p = 15$, $d = 50$, and $N = 10\,000$.

We compare the proposed SPU algorithm with the FCLSU algorithm presented in [2] for testing the accuracy and benchmarking the required computational time. Both algorithms are implemented in MATLAB. The implementation of the FCLSU algorithm turns the problem into an equivalent problem where only the positivity constraint needs to be obeyed, and it uses the built-in MATLAB function "lsqnonneg" for solving this NNLS problem with the technique proposed by Lawson and Hanson [17]. This method has become a *de facto* standard in linear spectral unmixing for determining the abundances after the endmember extraction phase. We must point out that the SPU algorithm can benefit from the simultaneous treatment of the entire data set, while the FCLSU algorithm treats every data point sequentially and independently from previous runs. These tests were done on a regular desktop machine with an Intel Core 2 Quad 3.0-GHz processor and 4 GB of RAM.

First of all, we tested the numerical accuracy by calculating the abundance coefficients for several parameter sets with both algorithms and comparing them one by one. We found that, in a typical situation ($p=5$, $N=10^5$, $d=4$, and $SNR=10$), about 99.7% of the retrieved abundances will differ by less than 10^{-7} from those obtained by the FCLSU algorithm (averaged over 100 runs). In the other 0.3%, some errors can be attributed to the breakdown of the assumption used in the algorithm, but even then, the typical errors usually stay very low. Similar results can be obtained for other values of p, although it must be noted that, for any value of $p>3$, examples which have significant errors can be artificially created. Because the results obtained by both algorithms are hence comparable in quality in practical use, we mainly focus on the computational performance.

In Fig. 5, we have plotted the runtime and the number of recursions needed to obtain the solution as a function of the SNR, for $p = 15$ endmembers, dimensionality $d = 50$, and

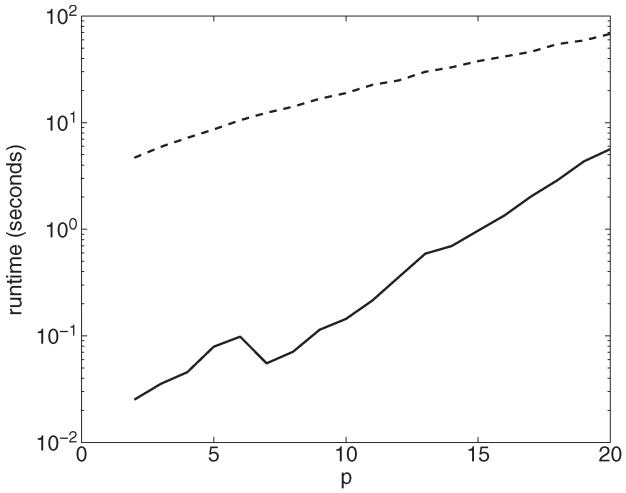

Fig. 6. Runtimes for the (solid line) SPU algorithm and the (dashed line) FCLSU algorithm as a function of the number of endmembers p (logarithmic scale). The parameters are $N = 10\,000$, $SNR = 10$, and $d = 50$.

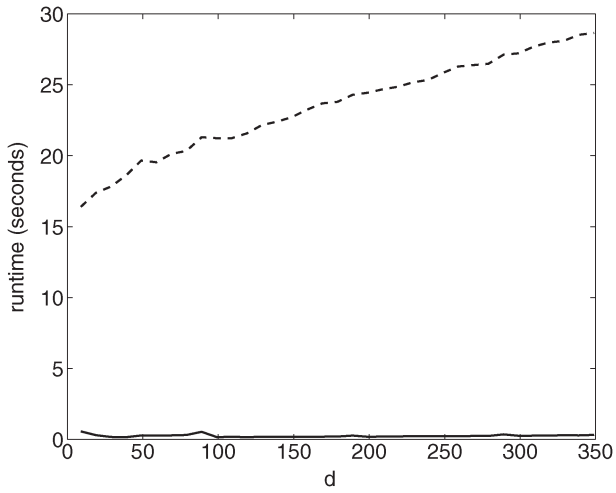

Fig. 7. Runtimes for the (solid line) SPU algorithm and the (dashed line) FCLSU algorithm as a function of the dimensionality d. The parameters are $N = 10\,000$, $SNR = 10$, and $p = 10$.

$N = 10\,000$ data points. We notice that, as can be expected, the runtime of the SPU algorithm scales linearly with the number of recursions. Second, the SPU algorithm has a significantly shorter runtime for all values of SNR. For very low SNR, the number of recursions needed strongly increases, which causes longer runtimes of the SPU algorithm. For larger values of SNR however, which are typical of most practical situations, the SPU algorithm will outperform the FCLSU algorithm by a factor of order 10–50 and often higher.

Next, we have plotted the runtimes as a function of the number of endmembers p in Fig. 6 for both algorithms, with $N = 10\,000$, SNR $= 10$, and $d = 50$. The required processing time increases exponentially as a function of p for both algorithms. Again, the SPU algorithm performs significantly better than the FCLSU algorithm. The slightly higher slope of the SPU result in Fig. 6 however indicates that, for very high values of p, the FCLSU algorithm can have a better performance than the SPU algorithm. Such high values of p are not often encountered in practice however.

The dependence of runtime on the dimensionality of the data d is shown in Fig. 7. It can be seen that the runtime of the FCLSU algorithm increases significantly for increasing d. The runtime of the SPU algorithm, however, is mostly insensitive to the dimensionality d, as the same value is found for the entire range of d. This is mostly due to the fact that the most computationally intensive steps of the SPU algorithm are the calculations of matrix inverses, where the size of each matrix depends only on the number of endmembers p, and not the dimensionality d. Again, we notice that the runtime of the SPU algorithm is drastically smaller than the runtime of the FCLSU algorithm.

Finally, the dependence of runtime on the number of data points N is shown in Fig. 8. The runtime of the FCLSU algorithm is linear in the number of data points N, as can be expected from the fact that the FCLSU algorithm processes point per point. The runtime of the SPU algorithm appears to scale linearly in N as well, although this relation is not so clear from Fig. 8. The runtimes of the SPU algorithm are, again, much lower than those of the FCLSU algorithm. Furthermore, the slope of the SPU curve is smaller than that of the FCLSU curve,

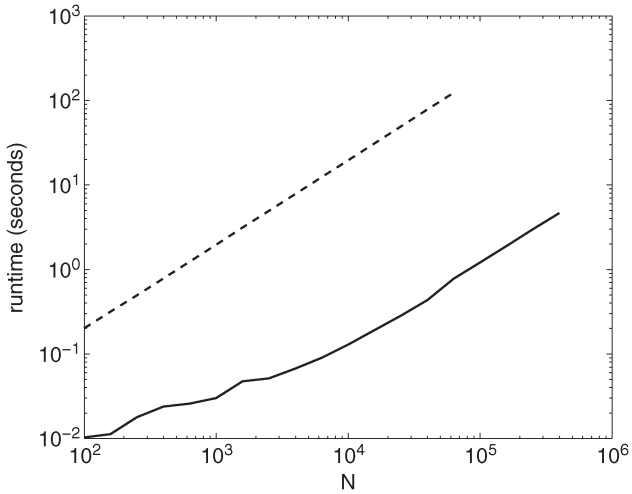

Fig. 8. Runtimes for the (solid line) SPU algorithm and the (dashed line) FCLSU algorithm as a function of the number of data points N (logarithmic scale). The parameters are $SNR = 10$, $p = 10$, and $d = 50$.

indicating that the difference in computational performance between both algorithms will only increase with increasing N.

Execution of both tests with a profiler shows that, for the FCLSU algorithm, 75% of the runtime is used for calculating matrix pseudoinverses, while the other 25% of the time is mainly used for memory management and other matrix manipulations. For the SPU algorithm, virtually all time is dedicated to matrix manipulations. A careful optimization based on studying the profiling output and using a compiled programming language can yield further significant increases in performance.

C. Execution on the AVIRIS Cuprite Data Set

As we have seen in the previous section, the SPU algorithm will, in most cases, have a significantly lower runtime than the FCLSU algorithm when executed on synthetic data. Next, we evaluate the performance of both algorithms when executed on real hyperspectral data sets. The data set under consideration is the well-known AVIRIS Cuprite data set, obtained over the Cuprite mining region in Nevada, USA (see, e.g., [35] for a

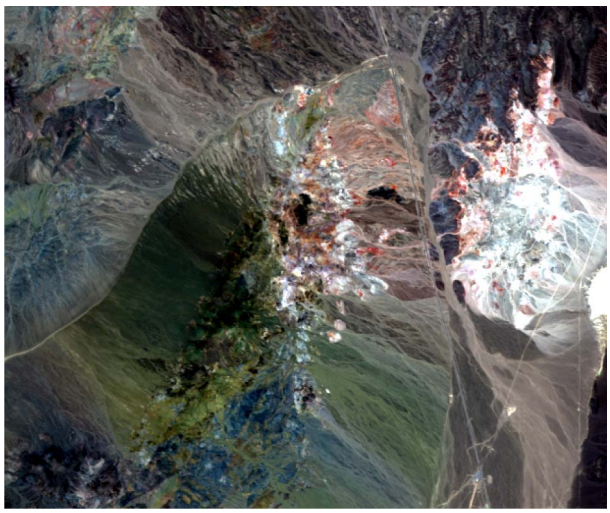

Fig. 9. Approximate color image of the AVIRIS Cuprite data set used. The bands used as RGB channels are bands (40, 20, 10) of the original 224-band image, corresponding to (731, 557, 458) nm.

detailed description of this data set). Specifically, we used the entire "f970619t01p02_r02_sc04.a.rfl" file, resulting in a 512×614 image. After elimination of several bands with low SNR, we kept 188 of the 224 spectral bands [36]. An approximate color image of this data set is shown in Fig. 9.

Since the unmixing algorithms are supervised and hence assume that the endmembers are known, we first have to run an EEA to locate these endmembers. The algorithm chosen is the NfindR algorithm [9] for its proven performance on this data set. Furthermore, we must determine the number of endmembers p to extract, which is not a trivial task. A too low number will not yield good unmixing results since the retrieved endmembers themselves will be mixtures of several actual pure materials. A too high number of endmembers will result in many unidentifiable endmembers, mostly corresponding to spectral noise, and multiple endmembers corresponding to the same pure material [37]. Ideally, the intrinsic dimensionality of the data set in spectral space should be determined, from which the necessary number of endmembers can be derived, e.g., via the virtual dimensionality technique [38]. Another method for estimating p is by using ground data, e.g., the number of known different minerals in the region under consideration. Such methods lead to an estimate of the number of endmembers of order $\mathcal{O}(10)$. Therefore, we have extracted the endmembers and unmixed the Cuprite data set for all values of p in the range [8, 15]. For every value of p, the found endmembers correspond to those expected in this data set, e.g., alunite, kaolinite, buddingtonite, calcite, or muscovite. Therefore, we do not focus on the quality of the endmember extraction but on the runtimes needed for the subsequent unmixing algorithm. Hence, we use the NfindR algorithm to extract different sets of endmembers, depending on the value of p, and use the SPU and FCLSU algorithms to unmix the data with respect to the found endmembers.

A typical unmixing result is shown in Fig. 10 for the alunite endmember. This abundance map can also be found in the literature (e.g., [9]). As can be seen in Fig. 10, both the FCLSU and the SPU algorithms generate qualitatively identical abundance maps. The runtimes of both algorithms as a function of the number of endmembers p are shown in Fig. 11. It is clear that the SPU algorithm has superior performance compared to the FCLSU algorithm on this data set, at least for $p \in [8, 15]$.

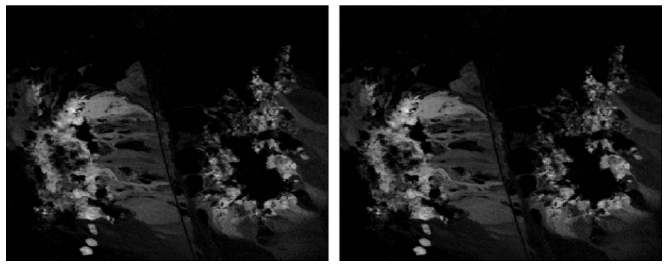

Fig. 10. Cropped selection of the abundance map for the alunite endmember, obtained with the (left) FCLSU algorithm and the (right) SPU algorithm. The image is cropped for clarity (this endmember has almost zero abundance in the rest of the image). Both abundance maps are indiscernible.

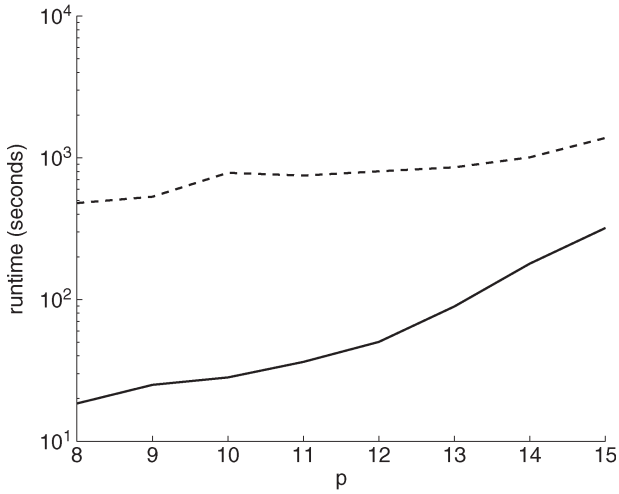

Fig. 11. Runtimes for the (solid line) SPU algorithm and the (dashed line) FCLSU algorithm as a function of the number of endmembers p, for the AVIRIS Cuprite data set. The total number of pixels is $N = 314\,368$, and the number of bands is $d = 188$.

V. Conclusion

We have introduced a new method for supervised linear spectral unmixing of hyperspectral data based upon the equivalence between solving the fully constrained least squares problem and the projection of a point onto a simplex. This SPU algorithm uses a sequence of orthogonal projections on subsimplices, decreasing the dimensionality of the problem by one with each step. The algorithm can be implemented recursively and is well fit to deal with large amounts of data because many operations that depend solely on the endmembers need to be executed only once for all data points. A concrete implementation of the algorithm in pseudocode is given, along with an explanation of all the steps.

We have tested the performance of the algorithm on simulated data and on the AVIRIS Cuprite hyperspectral data set and compared the results and the runtimes with those obtained with the well-known FCLSU algorithm from [2]. We have observed that the SPU algorithm and the FCLSU algorithm both yield almost identical results on real data sets. The runtimes of the SPU algorithm, however, are drastically lower for most practical purposes, and only for very high numbers of endmembers, the

SPU algorithm can perform worse than the FCLSU algorithm. Also, on the AVIRIS data, the SPU algorithm has a superior computational performance for several values of the number of endmembers.

Future work includes a detailed profiling of the different steps of the algorithm, seeking further ways to improve the computational performance. Also, the assumption used concerning the bissective cones in the derivation of the algorithm should be replaced with an alternative that always yields the correct results and is preferably provably correct. We are also working on a distance-geometry-based version of the SPU algorithm, allowing, among others, the use of geodesic distance matrices in the unmixing algorithm. Furthermore, the simplex-projection problem and the constrained unmixing problem are recurrent problems occurring in many scientific fields. This same algorithm can be used to solve such problems outside the scope of spectral unmixing, and identifying such fields is another task.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for providing the counterexample to the assumption used and for the interesting suggestions for improving the first version of this paper.

REFERENCES

[1] N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 44–57, Jan. 2002.

[2] D. C. Heinz and C.-I. Chang, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 3, pp. 529–545, Mar. 2001.

[3] C.-I. Chang, H. Ren, C.-C. Chang, F. D'Amico, and J. O. Jensen, "Estimation of subpixel target size for remotely sensed imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 6, pp. 1309–1320, Jun. 2004.

[4] N. Dobigeon, S. Moussaoui, M. Coulon, J.-Y. Tourneret, and A. O. Hero, "Joint Bayesian endmember extraction and linear unmixing for hyperspectral imagery," *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4355–4368, Nov. 2009.

[5] A. Plaza, P. Martinez, R. Perez, and J. Plaza, "A quantitative and comparative analysis of endmember extraction algorithms from hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 3, pp. 650–663, Mar. 2004.

[6] M. Parente and A. Plaza, "Survey of geometric and statistical unmixing algorithms for hyperspectral images," in *Proc. 2nd WHISPERS*, 2010, pp. 1–4.

[7] M. Veganzones and M. Graña, *Endmember Extraction Methods: A Short Review*, vol. 5179, *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2010.

[8] J. W. Boardman, "Geometric mixture analysis of imaging spectrometry data," in *Proc. Int. GRS Symp.*, 1994, vol. 4, pp. 2369–2371.

[9] E. M. Winter, "N-FINDR: An algorithm for fast autonomous spectral end-member determination in hyperspectral data," in *Proc. SPIE*, 1999, vol. 3753, pp. 266–275.

[10] C.-I. Chang, C.-C. Wu, W.-M. Liu, and Y.-C. Ouyang, "A new growing method for simplex-based endmember extraction algorithm," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 10, pp. 2804–2819, Oct. 2006.

[11] J. C. Harsanyi and C.-I. Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 779–785, Jul. 1994.

[12] A. Ifarraguerri and C.-I. Chang, "Multispectral and hyperspectral image analysis with convex cones," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 2, pp. 756–770, Mar. 1999.

[13] J. M. P. Nascimento and J. M. B. 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.

[14] J. J. Settle and N. A. Drake, "Linear mixing and the estimation of ground cover proportions," *Int. J. Remote Sens.*, vol. 14, no. 6, pp. 1159–1177, Apr. 1993.

[15] C.-I. Chang, X.-L. Zhao, M. L. G. Althouse, and J. J. Pan, "Least squares subspace projection approach to mixed pixel classification for hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 36, no. 3, pp. 898–912, May 1998.

[16] Y. E. Shimabukuro and J. A. Smith, "The least-squares mixing models to generate fraction images derived from remote sensing multispectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 29, no. 1, pp. 16–20, Jan. 1991.

[17] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*. Englewood Cliffs, NJ: Prentice-Hall, 1974.

[18] P. Bajorski, "Simplex projection methods for selection of endmembers in hyperspectral imagery," in *Proc. IEEE IGARSS*, 2004, vol. 5, pp. 3207–3210.

[19] A. Zare and P. D. Gader, "Sparsity promoting iterated endmember detection in hyperspectral imagery," *IEEE Geosci. Remote Sens. Lett.*, vol. 4, no. 3, pp. 446–451, Jul. 2007.

[20] J. Chen, X. Jia, W. Yang, and B. Matsushita, "Generalization of subpixel analysis for hyperspectral data with flexibility in spectral similarity measures," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 7, pp. 2165–2171, Jul. 2009.

[21] M. Velez-Reyes and S. Rosario, "Solving abundance estimation in hyperspectral unmixing as a least distance problem," in *Proc. IEEE IGARSS*, 2004, vol. 5, pp. 3276–3278.

[22] N. Dobigeon, J.-Y. Tourneret, and C.-I. Chang, "Semi-supervised linear spectral unmixing using a hierarchical Bayesian model for hyperspectral imagery," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 2684–2695, Jul. 2008.

[23] S. Moussaoui, D. Brie, A. Mohammad-Djafari, and C. Carteret, "Separation of non-negative mixture of non-negative sources using a Bayesian approach and MCMC sampling," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4133–4145, Nov. 2006.

[24] S. Moussaoui, H. Hauksdottir, F. Schmidt, C. Jutten, J. Chanussot, D. Brie, S. Douté, and J. A. Benediktsson, "On the decomposition of Mars hyperspectral data by ICA and Bayesian positive source separation," *Neurocomputing*, vol. 71, no. 10–12, pp. 2194–2208, Jun. 2008.

[25] F. Schmidt, A. Schmidt, E. Tréguier, M. Guiheneuf, S. Moussaoui, and N. Dobigeon, "Implementation strategies for hyperspectral unmixing using Bayesian source separation," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 11, pp. 4003–4013, Nov. 2010.

[26] S. Jia and Y. Qian, "Constrained nonnegative matrix factorization for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 1, pp. 161–173, Jan. 2009.

[27] S. A. Robila and L. G. Maciak, "Considerations on parallelizing nonnegative matrix factorization for hyperspectral data unmixing," *IEEE Geosci. Remote Sens. Lett.*, vol. 6, no. 1, pp. 57–61, Jan. 2009.

[28] J. L. Silván-Cárdenas and L. Wang, "Fully constrained linear spectral unmixing: Analytic solution using fuzzy sets," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 11, pp. 3992–4002, Nov. 2010.

[29] D. Gillis, J. Bowles, and M. E. Winter, "Using endmembers as a coordinate system in hyperspectral imagery," in *Proc. SPIE, Imaging Spectrometry VIII*, 2002, vol. 4816, pp. 346–354.

[30] C. Michelot, "A finite algorithm for finding the projection of a point onto the canonical simplex of α^n," *J. Optim. Theory Appl.*, vol. 50, no. 1, pp. 195–200, Jul. 1986.

[31] H. J. H. Tuenter, "The minimum L2-distance projection onto the canonical simplex: A simple algorithm," *Algo Research Quarterly*, vol. 4, pp. 53–55, Dec. 2001.

[32] I.-H. Lin, *Geometric Linear Algebra*, vol. 2. Singapore: World Scientific, 2008.

[33] S. S. Shwartz, Y. Singer, P. Bennett, and E. Parrado-Hernandez, "Efficient learning of label ranking by soft projections onto polyhedra," *J. Mach. Learn. Res.*, vol. 7, pp. 1567–1599, Dec. 2006.

[34] J. Duchi, S. S. Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the $l1$-ball for learning in high dimensions," in *Proc. ICML*, 2008, pp. 272–279.

[35] F. Kruse, J. Boardman, and J. Huntington, "Comparison of airborne hyperspectral data and EO-1 Hyperion for mineral mapping," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 6, pp. 1388–1400, Jun. 2003.

[36] C.-I. Chang and S. Wang, "Constrained band selection for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 6, pp. 1575–1585, Jun. 2006.

[37] D. Stein, "Application of the normal compositional model to the analysis of hyperspectral imagery," in *Proc. IEEE Workshop Adv. Tech. Anal. Remotely Sens. Data*, 2003, pp. 44–51.

[38] C.-I. Chang and Q. Du, "Estimation of the number of spectrally distinct signal sources in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 3, pp. 608–619, Mar. 2004.

Rob Heylen (M'10) received the B.S. and M.S. degrees and the Ph.D. degree in physics, with work in the field of statistical mechanics, from Katholieke Universiteit Leuven, Leuven, Belgium, in 2001, 2003, and 2008, respectively.

Since 2009, he has been a Postdoctoral Researcher with the Visionlab, Department of Physics, University of Antwerp, Antwerp, Belgium. His main areas of research interest are hyperspectral image processing and computational physics.

Dževdet Burazerović (M'00) received the M.S. degree in electrical engineering from Eindhoven University of Technology, Eindhoven, The Netherlands, in 2000. He is currently working toward the Ph.D. degree in the Visionlab, Department of Physics, University of Antwerp, Antwerp, Belgium.

Since 2009, he has been with the Visionlab, Department of Physics, University of Antwerp, where he is working as a Researcher. His current research interests include hyperspectral image processing and data/pattern analysis.

Paul Scheunders (M'98) received the B.S. degree and the Ph.D. degree in physics, with work in the field of statistical mechanics, from the University of Antwerp, Antwerp, Belgium, in 1983 and 1990, respectively.

In 1991, he became a Research Associate with the Visionlab, Department of Physics, University of Antwerp, where he is currently a Professor. He has published over 120 papers in international journals and proceedings in the field of image processing and pattern recognition. His research interests include wavelets and multispectral image processing.