ABSTRACT
A novel algorithm for endmember extraction is presented. The approach follows the linear mixture model for hyperspectral data. Endmembers are identified based on sparsity considerations. Theoretical and experimental results suggest the potential of the method.

1. OVERVIEW
Multispectral measurements are used in various scientific fields to characterize the chemical composition of remote surfaces. Each pixel of a multispectral image set often represents a mixture of materials. There is a demand to develop automated tools in order to specify constituent materials, so-called endmembers. The linear mixture model of hyperspectral data assumes that each pixel is a linear combination of endmembers and the constituent abundances are nonnegative and sum to one. There is a large variety of endmember extraction algorithms that are based on this model [1, 2], see [3] for a parallel implementation. Utilizing sparsity constraints for computing abundance maps is proposed in [4], claiming that a more realistic linear mixture model should take into account that only a few of the identified endmembers contribute to one single pixel. Endmember extraction and computing abundance maps can be joined into one optimization problem by means of dictionary learning [5]. This leads, however, to non-convex optimization. A potential solution is usually computed by alternated minimizations over the dictionary and the coefficients. We develop a complementary approach by splitting the optimization into two compatible parts. First, we identify endmembers based on sparsity considerations. Second, we compute abundance maps utilizing sparsity constraints.

Our novel method provides several advantages. First, we seem to follow the more realistic linear mixture model as proposed in [4] when enforcing sparsity. Second, our algorithm is fully parallelizable. Third, the number of endmembers is an input and increasing this number preserves the smaller endmember set and just results in additional endmembers rather than a completely new set. Therefore, our proposed scheme provides features that offer advantages in fields that obey a linear mixture model with sparsity constraints.

2. GENERAL SETTING
Let \( \mathcal{X} = \{x_i\}_{i=1}^n \subset \mathbb{R}^d \) be a high dimension data set, and assume that \( n \gg d \). According to the Linear Mixture Model (LMM), an endmember set \( \mathcal{E} = \{e_i\}_{i=1}^s \) for \( \mathcal{X} \), \( s \ll n \), is a collection of constituent vectors such that

\[
x_i = \sum_{j=1}^s \alpha_{i,j} e_j + N_{x_i}, \quad \forall x_i \in \mathcal{X},
\]

where \( N_{x_i} \in \mathbb{R}^d \) is a noise vector and the coefficient set \( \{\alpha_{i,j}\}_{i,j=1}^{n,s} \) satisfies \( \alpha_{i,j} \geq 0 \) and \( \sum_{j=1}^s \alpha_{i,j} = 1 \). For now, we suppress the noise vector \( N_{x_i} \). We assume that \( \mathcal{E} \) is a subset of \( \mathcal{X} \) consistent with common assumptions in satellite imaging. Each \( x_i \in \mathcal{X} \setminus \mathcal{E} \) can then be written as a linear combination of elements in \( \mathcal{X} \setminus \{x_i\} \). First, since \( s \ll n \), the Equation (1) means that \( x_i \) has a sparse expansion in the dictionary \( \mathcal{X} \setminus \{x_i\} \). Second, each \( x_i \) is likely to be a mixture of not \( s \) endmembers but only few of them, and hence the coefficient vector \( \{\alpha_{i,1}, \ldots, \alpha_{i,s}\} \) has only few nonzero entries. Both observations motivate the application of tools from sparse approximation to recover \( \mathcal{E} \) from \( \mathcal{X} \).

The material presented in this paper is based upon work supported in part by the Intramural Research Program of NICHD/NIH, by NSF (CBET0854233), by NGA (HM15820810009), by ONR (N000140910144), as well as by NIH/DFG Research Career Transition Awards Program (EH 405/1-1/575910). Copyright 2012 IEEE. Published in the IEEE 2012 International Geoscience & Remote Sensing Symposium (IGARSS 2012), scheduled for July 22-27, 2012 in Munich, Germany. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE. Contact: Manager, Copyrights and Permissions / IEEE Service Center / 445 Hoes Lane / P.O. Box 1331 / Piscataway, NJ 08855-1331, USA. Telephone: + Intl. 908-562-3966.
3. ENDMEMBER EXTRACTION

The algorithm consists of two main steps, a weighting and an ordering, detailed below. After the endmembers are determined, the abundances can be computed by utilizing sparsity constraints again.

3.1. Weighting the elements of $\mathcal{X}$

For each $x_i \in \mathcal{X}$ we seek to represent $x_i$ in terms of the remaining elements of $\mathcal{X}$. That is, in principle at least, we wish to solve for weights $\{w_{i,j}\}_{i,j=1}^n$ such that

$$x_i = \sum_{j=1}^n w_{i,j} x_j + N_{x_i}, \quad w_{i,j} \geq 0, \quad w_{i,i} = 0. \quad (2)$$

We then choose the $s$ elements of $\mathcal{X}$ with the most 'significant' weights as our endmembers. Later, we shall define precisely what it means for the weights to be 'significant'. Since $n \gg d$, it is most likely that the weights satisfying (2) are not uniquely determined. To make the weights match the endmembers, we shall find weights with minimal support, i.e., we aim to solve

$$w_{i,*} = \arg \min_{\hat{w}} \|\hat{w}\|_{\ell_0}$$

subject to the constraints:

$$x_i = \sum_{j=1}^n \hat{w}_j x_j + N_{x_i}, \quad \hat{w}_j \geq 0, \quad \hat{w}_i = 0,$$

which yields the sparse weight matrix $W = (w_{i,j})$. In practice, we have a heuristic analysis of $W$ in order to extract $\mathcal{E}$ from $W$.

We still need to accommodate the noise vector $N_{x_i}$. To compensate we change the minimization problem and allow the representation to be imperfect. For $\tilde{w} = (\tilde{w}_1, \ldots, \tilde{w}_n)$, we could then consider

$$w_{i,*} = \arg \min_{\tilde{w}} \|x_i - \sum_{j=1}^n \tilde{w}_j x_j\|_{\ell_2}^2 + \tau_i \|\tilde{w}\|_{\ell_0} \quad (3)$$

subject to the constraints:

$$\tilde{w}_j \geq 0, \quad \tilde{w}_i = 0.$$

The parameter $\tau_i \geq 0$ balances the reconstruction error and the penalty term $\|\tilde{w}\|_{\ell_0}$. The bigger $\tau_i$ the more importance is placed on the $\ell_0$ norm and thus the sparser the minimizer. For best results $\tau_i$ has to be chosen in accordance to the expected noise level. The problem (3), however, is a non-convex optimization and not tractable for large data sets. Replacing the $\ell_0$ norm with the $\ell_1$ norm is known to provide a sparse minimizer as well. Moreover, the $\ell_1$ norm is more robust against low level noise than the $\ell_0$ norm. We thus replace (3) with

$$w_{i,*} = \arg \min_{\tilde{w}} \|x_i - \sum_{j=1}^n \tilde{w}_j x_j\|_{\ell_2} + \tau_i \|\tilde{w}\|_{\ell_1} \quad (4)$$

subject to the constraints:

$$\tilde{w}_j \geq 0, \quad \tilde{w}_i = 0.$$

To reduce the complexity, we apply a spatially uniform subsampling of the dataset. Dimension reduction would further reduce the computational cost [6].

3.2. Ordering the elements of $\mathcal{X}$ and demixing

After computing the weights via the optimization problem in (4), we order the elements $x_i \in \mathcal{X}$ according to the significance of their weight columns $w_{i,j}$. As expected, the vast majority of columns of $W$ are zeros, see Fig. 1, and we remove the corresponding $x_i$ for which $w_{i,i}$ is the zero vector. From the remaining elements of $\mathcal{X}$, we create two ordered lists according to the relations $\preceq_0$ and $\preceq_2$, respectively, given by $x_i \preceq_0 x_j$ if and only if $\|w_{i,j}\|_{\ell_0} \leq \|w_{j,i}\|_{\ell_0}$ and $x_i \preceq_2 x_j$ if and only if $\|w_{i,j}\|_{\ell_2} \leq \|w_{j,i}\|_{\ell_2}$. Therefore, each vector $x_i \in \mathcal{X}$ now has two additional indexes, $r_{i,0}$ and $r_{i,2}$, one for each list, respectively. The $r_{i,0}$ list ranks vectors in $\mathcal{X}$ according to how many representations they are used in; the $r_{i,2}$ list, on the other hand, ranks the vector $x_i$, highly if it plays a significant role in the representations it is in, regardless of the number of representations it is in used. A final ordered list is then created by combining the orderings of $r_{i,0}$ and $\ell_2 - \ell_0$ lists, i.e., $x_i \preceq x_j$ if and only if $r_{i,0}^0 + r_{i,2}^0 \geq r_{j,0}^0 + r_{j,2}^0$. The index of $x_i$ in this final list is denoted as $r_i$. The endmember set $\mathcal{E} = \{e_{ij}\}_{i,j=1}^s$ is then given by the highest rankings.

Next, we compute the coefficient matrix $\alpha$ of $\mathcal{X}$ with respect to $\mathcal{E}$, i.e., for $\alpha = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_s)$, we solve

$$\alpha_{i,*} = \arg \min_{\tilde{\alpha}} \|x_i - \sum_{j=1}^s \tilde{\alpha}_j e_j\|_{\ell_2}^2 + \beta_i \|\tilde{\alpha}\|_{\ell_1}$$

subject to the constraints:

$$\tilde{\alpha}_j \geq 0, \quad \sum_{j=1}^s \alpha_{i,j} \approx 1.$$

4. NUMERICAL RESULTS

4.1. Artificial Data Sets

Select i.i.d. vectors $e_1, \ldots, e_s$ according to the uniform surface measure on the $d$-dimensional unit sphere. We then create a random mixing matrix whose entries are i.i.d. according to the uniform distribution on $(0,1)$. For its $j$-th column, we randomly set $s - k_j$ entries to zero, where $k_j$ is randomly selected from $\{1, \ldots, k\}$. After $\ell_1$ normalizing its columns, we obtain a mixing matrix $A$ that satisfies $A_{ij} \geq 0, \sum_{i=1}^s A_{ij} = 1$, and $|\text{supp}(A_{ij})| = k_j \leq k$. Let $\mathcal{X}'$ be the union between $\mathcal{E}$ and their mixtures. Finally, $\mathcal{X}$ results from
adding an i.i.d. gaussian noise with zero mean and variance \( \varepsilon \) to \( \mathcal{X}' \) and then normalize its columns.

We compare the exact and computed endmember sets by first counting the exact matches and then computing the average inner product \( \sigma \) between all inexact matches. Thus, \( \sigma \approx 1 \) would mean that we reasonable well-recover \( \mathcal{E} \). We repeated the experiments 100 times and computed the average for \( \sigma \) and the number of exact matches, cf. Fig. 2.

<table>
<thead>
<tr>
<th>( d )</th>
<th>( s )</th>
<th>( k )</th>
<th>( n )</th>
<th>( \varepsilon )</th>
<th># matches</th>
<th>( \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>500</td>
<td>0.00</td>
<td>8.55</td>
<td>0.9995</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>500</td>
<td>0.01</td>
<td>7.35</td>
<td>0.9997</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>500</td>
<td>0.05</td>
<td>7.70</td>
<td>0.9974</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>500</td>
<td>0.10</td>
<td>6.68</td>
<td>0.8638</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>7</td>
<td>500</td>
<td>0.00</td>
<td>8.24</td>
<td>0.9830</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>7</td>
<td>500</td>
<td>0.01</td>
<td>7.46</td>
<td>0.9842</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>7</td>
<td>500</td>
<td>0.05</td>
<td>7.91</td>
<td>0.9735</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>7</td>
<td>500</td>
<td>0.10</td>
<td>7.48</td>
<td>0.8375</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>10</td>
<td>500</td>
<td>0.00</td>
<td>8.88</td>
<td>0.9192</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>10</td>
<td>500</td>
<td>0.01</td>
<td>5.74</td>
<td>0.9199</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>10</td>
<td>500</td>
<td>0.05</td>
<td>6.35</td>
<td>0.9109</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>10</td>
<td>500</td>
<td>0.10</td>
<td>7.53</td>
<td>0.8245</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>10000</td>
<td>0.00</td>
<td>2.44</td>
<td>0.9986</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>10000</td>
<td>0.01</td>
<td>0.70</td>
<td>0.9975</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>10000</td>
<td>0.05</td>
<td>0.36</td>
<td>0.9949</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4</td>
<td>10000</td>
<td>0.10</td>
<td>0.54</td>
<td>0.9401</td>
</tr>
</tbody>
</table>

**Fig. 2.** Artificial data; endmembers are almost perfectly recovered in the noiseless setting. Higher noise levels naturally decrease the recovery rate. Note that for the data sets consisting of 10000 data points, the endmembers were computed from 500 randomly chosen samples. Hence the low number of matches; however, the inner product remains high, indicating that the algorithm seems to choose the closest match from the samples.

### 4.2. Satellite Data

We applied the proposed scheme to extract endmembers and compute their abundances for 16 bit reflectance Smith Island PROBE2 HSI data published in [7]. Abundance maps of endmembers (E1-E11) are ordered from left to right and top to bottom, see Fig. 3. For comparison, we applied vector component analysis (VCA) [8] in Fig. 4. The abundance maps for the VCA endmembers were computed as the least squares solution subject to the constraints that the coefficients are positive and sum to one.

Both VCA and our method separate materials well. Our method seems to split materials into further subclasses while VCA maps more “globally”. For example, VCA yields two maps that get most of the ocean (no. 1 gets the vast majority of it, and then no. 11 which appears as ocean mixed with sand). We obtain 3 maps (E8, E9, and E10) which break the ocean into finer parts, and also E2 for the southern shore water. Similarly, VCA has two main maps for the island vegetation and land (no. 3 and no. 4), while we derive E1, E3, E4, E5, E7. The nets in the water are picked up by VCA but put in the same endmember class as the inland water (no. 2), whereas sparse endmembers do a reasonable job of splitting that information between E6 and E11.

### 5. REFERENCES


Fig. 3. Pseudo-color image of the Smith Island scene that emphasizes as many features as possible and abundance maps of 11 endmembers. The first two endmembers divide the shore into complementary areas. Notice that while some of the abundance maps overlap, the highest intensity areas do not.

Fig. 4. Abundance maps of VCA computed from 11 endmembers. We omit 6 of them here since they had very little information.
