A NON-NEGATIVE SPARSE PROMOTING ALGORITHM FOR HIGH RESOLUTION HYPERSPECTRAL IMAGING

Eliot Wycoff[†], *Tsung-Han Chan*[†], *Kui Jia*[†], *Wing-Kin Ma*[‡] *and Yi Ma*[†*]

[†]Advanced Digital Sciences Center, Singapore
[‡]Dept. of Electronic Eng., Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[*]Microsoft Research Asia, Beijing, China
E-mail: {Eliot.wycoff,Th.chan,Chris.jia}@adsc.com.sg; wkma@ieee.org; mayi@microsoft.com

ABSTRACT

Promoting the spatial resolution of off-the-shelf hyperspectral sensors is expected to improve typical computer vision tasks, such as target tracking and image classification. In this paper, we investigate the scenario in which two cameras, one with a conventional RGB sensor and the other with a hyperspectral sensor, capture the same scene, attempting to extract redundant and complementary information. We propose a non-negative sparse promoting framework to integrate the hyperspectral and RGB data into a high resolution hyperspectral set of data. The formulated problem is in the form of a sparse non-negative matrix factorization with prior knowledge on the spectral and spatial transform responses, and it can be handled by alternating optimization where each subproblem is solved by efficient convex optimization solvers; e.g., alternating direction method of multipliers. Experiments on a public database show that our method achieves much lower average reconstruction errors than other state-of-the-art methods.

Index Terms— Hyperspectral images, RGB images, image fusion, non-negativity, sparsity

1. INTRODUCTION

Hyperspectral cameras provide the ability to sample a scene's spectral properties more densely. Whereas a normal RGB camera roughly divides the observation into three components, namely red, green, and blue, a hyperspectral camera can easily obtain thirty or more distinct bands across the visible spectrum. Having detailed spectral information can be very useful in a number of computer vision tasks ranging from object recognition and tracking [1, 2, 3, 4] to geosensing [5], as RGB alone is often insufficient to identify the materials within a scene. With this greater spectral resolution for hyperspectral images, however, comes the tradeoff of decreased spatial resolution [6]; yet in many applications, a high spatial resolution equivalent to that of an RGB camera is desirable. Because of hardware limitations, however, it would be more cost-effective to estimate a high resolution hyperspectral image than to actually obtain it directly.

Hence, we specifically seek a high spatial resolution hyperspectral signal $\mathbf{Z} \in \mathbb{R}^{M_h \times L_c}$ where M_h is the number of hyperspectral bands and L_c is the number of pixels in the desired high resolution image for each band. As noted in previous work [6, 7], $\mathbf{Z}$ can be factorized into two matrices:

$$\mathbf{Z} = \mathbf{A}\mathbf{S}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{M_h \times N}$ and $\mathbf{S} \in \mathbb{R}^{N \times L_c}$ are the hyperspectral material basis matrix and the hyperspectral material coefficients matrix, respectively. Thus, the columns of $\mathbf{A}$ are the hyperspectral basis vectors corresponding to scene materials, with N as the number of scene materials, and the columns of $\mathbf{S}$ are the material coefficients present at each pixel in the high resolution data $\mathbf{Z}$.

Since $\mathbf{Z}$ cannot be directly obtained, we instead attempt to reconstruct it from our low resolution hyperspectral observation $\mathbf{X} \in \mathbb{R}^{M_h \times L_h}$ and our high resolution RGB observation $\mathbf{Y} \in \mathbb{R}^{M_c \times L_c}$. Here we denote the number of pixels in the low resolution data as L_h, where $L_h < L_c$, and the number of spectral bands in the RGB data as $M_c = 3$. We may express $\mathbf{X}$ and $\mathbf{Y}$ as follows:

$$\mathbf{X} = \mathbf{A}\mathbf{S}\mathbf{G} = \mathbf{Z}\mathbf{G}, \tag{2}$$

$$\mathbf{Y} = \mathbf{F}\mathbf{A}\mathbf{S} = \mathbf{F}\mathbf{Z}, \tag{3}$$

Note that $\mathbf{X}$ and $\mathbf{Y}$ each are degraded versions of the desired high resolution hyperspectral data $\mathbf{Z}$, where $\mathbf{X}$ is spatially "downsampled" by $\mathbf{G} \in \mathbb{R}^{L_c \times L_h}$, and where $\mathbf{Y}$ is an RGB-transformed version of $\mathbf{Z}$ by $\mathbf{F} \in \mathbb{R}^{3 \times M_h}$. Here $\mathbf{F}$ and $\mathbf{G}$ are assumed to be known. Since the downsampling matrix $\mathbf{G}$ may not be easily obtained in practice, in Section 3.2 we experiment with the scenario in which $\mathbf{G}$ is imperfectly known.

Many methods have been explored for reconstructing $\mathbf{Z}$ from $\mathbf{X}$ and $\mathbf{Y}$. Most recently, coupled nonnegative matrix factorization (CNMF) [7] uses a conventional NMF framework to estimate $\mathbf{A}$ and $\mathbf{S}$ from $\mathbf{X}$ and $\mathbf{Y}$, and it employs $\mathbf{F}$ and $\mathbf{G}$ to couple the two estimations (2) and (3) in an iterative manner. The non-negativity constraint in CNMF is motivated by the simple observation that in real-world scenarios both the material basis $\mathbf{A}$ and the material coefficients $\mathbf{S}$ should take non-negative values. One drawback, however, is that NMF algorithms do not always yield a unique factorization [8, 9]. On the other hand, the work by Kawakami *et al.* [6] estimates the spectral basis $\mathbf{A}$ by a sparse-based dictionary learning method [10], and it enforces the coefficients in $\mathbf{S}$ to also be sparse. This sparsity constraint relies on the assumption that there are few materials present at any particular pixel location in $\mathbf{Z}$. Moreover, this is useful because the sparsity of $\mathbf{S}$ may be helpful in finding a unique reconstruction of $\mathbf{Z}$, as the linear system produced by (2) and (3) is underdetermined. The results yielded by [6], however, may not be physically explainable because in practice $\mathbf{A}$ and $\mathbf{S}$ should be non-negative.

This study is supported by the research grant for the Human Sixth Sense Programme at the Advanced Digital Sciences Center from Singapore's Agency for Science, Technology and Research (A*STAR).

In this paper we propose a non-negative sparse promoting framework that takes into consideration not only the non-negativity of the material basis $\mathbf{A}$ and the material coefficients matrix $\mathbf{S}$, but also the sparsity in $\mathbf{S}$. The formulated problem is in the form of a sparse non-negative matrix factorization with prior knowledge on the spectral and spatial transform responses $\mathbf{F}$ and $\mathbf{G}$, and it can be handled by alternating optimizations where each subproblem is solved by convex optimization solvers. To be computationally efficient, we implement alternating direction method of multipliers (ADMM) methods for each subproblem. Experiments on 20 different scenes from a public database show that our proposed method outperforms the state-of-the-art methods [7, 6] on average, regardless of whether the downsampling matrix $\mathbf{G}$ is known perfectly or imperfectly.

2. PROBLEM FORMULATION AND ALGORITHM

We consider the problem of estimating $\mathbf{Z}$ or, equivalently, $\mathbf{A}$ and $\mathbf{S}$, from the hyperspectral data $\mathbf{X}$ and the corresponding RGB image $\mathbf{Y}$. To encourage non-negativity on $\mathbf{A}$ and both non-negativity and sparsity on $\mathbf{S}$, we can formulate by (2) and (3) the optimization problem as follows:

$$\min_{\substack{\mathbf{A}\in\mathbb{R}_+^{M_h\times N}\\ \mathbf{S}\in\mathbb{R}_+^{N\times L_c}}} \frac{1}{2}\|\mathbf{X}-\mathbf{ASG}\|_F^2+\frac{1}{2}\|\mathbf{Y}-\mathbf{FAS}\|_F^2+\lambda\|\mathbf{S}\|_1, \quad (4)$$

where $\mathbb{R}_+^{M\times N}$ denotes the set of non-negative $M \times N$ matrices, $\|\cdot\|_F$ denotes the Frobenius norm, $\|\mathbf{S}\|_1$ denotes the ℓ_1 norm of the elements of $\mathbf{S}$, and $\lambda > 0$ leverages the fitting errors of the first two cost functions against the sparsity of $\mathbf{S}$. Problem (4) is biconvex. Thus a natural way to handle this problem is to apply alternating optimization; that is, with an initial $\mathbf{A}^0$ we solve the following two subproblems iteratively,

$$\mathbf{S}^{k+1} = \arg\min_{\mathbf{S}\in\mathbb{R}_+^{N\times L_c}} \mathcal{J}(\mathbf{A}^k,\mathbf{S}), \quad (5)$$

$$\mathbf{A}^{k+1} = \arg\min_{\mathbf{A}\in\mathbb{R}_+^{M_h\times N}} \mathcal{J}(\mathbf{A},\mathbf{S}^{k+1}), \quad (6)$$

where $\mathcal{J}(\mathbf{A},\mathbf{S})$ denotes the objective function of problem (4), and $k = 0, 1, 2, ...$ is the iteration number. It has been shown in [11] that the iterates (5) and (6) above will converge to a stationary point of problem (4). Additionally, both subproblems are convex and can be solved by any convex optimization solver. In our case, however, the size of $\mathbf{X}$ and $\mathbf{Y}$ are very large with hundreds of thousands of entries, thus we need a computationally cheap algorithm to solve these two problems. Alternating direction method of multipliers [12, 13] is well-suited for this purpose because it uses variable splitting and dual decomposition which, together, make the computational complexity in each ADMM iteration relatively low. Further, the linear convergence of ADMM algorithms for convex problems has been established [14] given certain error bound conditions.

The complete alternating optimization procedure for problem (4) is outlined in Algorithm 1. We use the alternating volume maximization (AVMAX) algorithm [15] to initialize $\mathbf{A}^0$, as it is an efficient and effective algorithm that produces a spectral basis with maximal convex cone volume. Since one of the assumptions made in [15] does not hold in our problem, we deviate slightly from the exact method in [15] by rescaling each column of the hyperspectral data $\mathbf{X}$, shown below, before providing it as the input of the AVMAX algorithm:

$$[\bar{\mathbf{X}}]_i = [\mathbf{X}]_i/\mathbf{1}^T[\mathbf{X}]_i,\ i = 1, ..., L_h. \quad (7)$$

Algorithm 1: Alternating optimization for problem (4).

```
input : $\mathbf{X}, \mathbf{Y}, \mathbf{G}, \mathbf{F}, \lambda > 0$ and $N$.
initialize $\mathbf{A}_0$ via AVMAX with $\bar{\mathbf{X}}$ given by (7), and set
$k = 0$.
while not converged do
    estimate $\mathbf{S}^{k+1}$ via ADMM in (5).
    estimate $\mathbf{A}^{k+1}$ via ADMM in (6).
    update $k := k + 1$.
end
compute $\mathbf{Z} = \mathbf{A}^k\mathbf{S}^k$.
output: reconstructed high resolution hyperspectral data $\mathbf{Z}$.
```

Here $[\bar{\mathbf{X}}]_i$ denotes the ith column of $\bar{\mathbf{X}}$, and $\mathbf{1}$ is an appropriately sized vector with each entry equal to 1.

2.1. ADMM for Problems (5) and (6)

We now develop the ADMM method for problem (5). By using $\text{vec}(\mathbf{PQR}) = (\mathbf{R}^T \otimes \mathbf{P})\,\text{vec}(\mathbf{Q})$, where $\otimes$ denotes the Kronecker product and $\text{vec}(\cdot)$ denotes the vectorization operator, and by letting $\boldsymbol{s} = \text{vec}(\mathbf{S})$, we can easily rewrite (5) as

$$\min_{\boldsymbol{s}\in\mathbb{R}_+^{NL_c}} \frac{1}{2}\|\boldsymbol{z}-\mathbf{B}\boldsymbol{s}\|_2^2+\lambda\|\boldsymbol{s}\|_1 \quad (8)$$

where

$$\boldsymbol{z} = \left[\text{vec}(\mathbf{X})^T\ \text{vec}(\mathbf{Y})^T\right]^T \in \mathbb{R}^{M_hL_h+3L_c}, \quad (9a)$$

$$\mathbf{B} = \left[(\mathbf{G}^T\otimes\mathbf{A}^k)^T\ (\mathbf{I}\otimes\mathbf{FA}^k)^T\right]^T \in \mathbb{R}^{(M_hL_h+3L_c)\times NL_c}. \quad (9b)$$

Problem (8) turns out to be a non-negative, ℓ_1 norm regularized linear least squares problem. To apply ADMM to this problem, we reformulate problem (8) as

$$\min_{\boldsymbol{x},\boldsymbol{s},\boldsymbol{y}\in\mathbb{R}^{NL_c}} \frac{1}{2}\|\boldsymbol{z}-\mathbf{B}\boldsymbol{y}\|_2^2+\lambda\|\boldsymbol{s}\|_1+\mathcal{I}_+(\boldsymbol{x}),\ \text{s.t. } \boldsymbol{x}=\boldsymbol{s}, \boldsymbol{x}=\boldsymbol{y}, \quad (10)$$

where $\mathcal{I}_+(\boldsymbol{x})$ is the indicator function of the non-negative orthant of $\boldsymbol{x}$. The ADMM framework operates on the augmented Lagrange multiplier of problem (10),

$$\mathcal{L}_\mu(\boldsymbol{x},\boldsymbol{s},\boldsymbol{y},\boldsymbol{h}_1,\boldsymbol{h}_2) = \frac{1}{2}\|\boldsymbol{z}-\mathbf{B}\boldsymbol{y}\|_2^2+\lambda\|\boldsymbol{s}\|_1+\boldsymbol{h}_1^T(\boldsymbol{x}-\boldsymbol{s}) + \mathcal{I}_+(\boldsymbol{x})+\boldsymbol{h}_2^T(\boldsymbol{x}-\boldsymbol{y})+\frac{\mu}{2}\|\boldsymbol{x}-\boldsymbol{s}\|_2^2+\frac{\mu}{2}\|\boldsymbol{x}-\boldsymbol{y}\|_2^2, \quad (11)$$

where $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ are dual variables, and $\mu > 0$. Further, ADMM optimizes in steps the primal variables of $\mathcal{L}_\mu$ by

$$\boldsymbol{x}^{j+1} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^{NL_c}} \mathcal{L}_\mu(\boldsymbol{x},\boldsymbol{s}^j,\boldsymbol{y}^j,\boldsymbol{h}_1^j,\boldsymbol{h}_2^j), \quad (12a)$$

$$\boldsymbol{s}^{j+1} = \arg\min_{\boldsymbol{s}\in\mathbb{R}^{NL_c}} \mathcal{L}_\mu(\boldsymbol{x}^{j+1},\boldsymbol{s},\boldsymbol{y}^j,\boldsymbol{h}_1^j,\boldsymbol{h}_2^j), \quad (12b)$$

$$\boldsymbol{y}^{j+1} = \arg\min_{\boldsymbol{y}\in\mathbb{R}^{NL_c}} \mathcal{L}_\mu(\boldsymbol{x}^{j+1},\boldsymbol{s}^{j+1},\boldsymbol{y},\boldsymbol{h}_1^j,\boldsymbol{h}_2^j), \quad (12c)$$

and the dual variables by

$$\boldsymbol{h}_1^{j+1} = \boldsymbol{h}_1^j + \mu(\boldsymbol{x}^{j+1}-\boldsymbol{s}^{j+1}), \quad (13)$$

$$\boldsymbol{h}_2^{j+1} = \boldsymbol{h}_2^j + \mu(\boldsymbol{x}^{j+1}-\boldsymbol{y}^{j+1}), \quad (14)$$

where j is the ADMM iteration number. The problems in (12) can be further expressed as

$$\min_{\boldsymbol{x}\in\mathbb{R}^{NL_c}} \mathcal{I}_{+}(\boldsymbol{x}) + \frac{1}{2}\left\|\boldsymbol{x} - \frac{1}{2}\left(\boldsymbol{s}^j + \boldsymbol{y}^j - \mu^{-1}\left(\boldsymbol{h}_1^j + \boldsymbol{h}_2^j\right)\right)\right\|_2^2, \tag{15a}$$

$$\min_{\boldsymbol{s}\in\mathbb{R}^{NL_c}} \frac{\lambda}{\mu}\|\boldsymbol{s}\|_1 + \frac{1}{2}\left\|\boldsymbol{x}^{j+1} - \boldsymbol{s} + \mu^{-1}\boldsymbol{h}_1^j\right\|_2^2, \tag{15b}$$

$$\min_{\boldsymbol{y}\in\mathbb{R}^{NL_c}} \|\boldsymbol{z} - \mathbf{B}\boldsymbol{y}\|_2^2 + \mu\left\|\boldsymbol{x}^{j+1} - \boldsymbol{y} + \mu^{-1}\boldsymbol{h}_2^j\right\|_2^2, \tag{15c}$$

Problems (15a) and (15b) are well known to be proximal operators associated with $\mathcal{I}_{+}(\boldsymbol{x})$ and $\lambda\mu^{-1}\|\boldsymbol{s}\|_1$, respectively [13], and thus they can be easily verified to have the closed-form solutions

$$\boldsymbol{x}^{j+1} = \left[\boldsymbol{s}^j + \boldsymbol{y}^j - \mu^{-1}\left(\boldsymbol{h}_1^j + \boldsymbol{h}_2^j\right)\right]_{+}/2, \tag{16}$$

$$\boldsymbol{s}^{j+1} = \operatorname{sgn}\left(\boldsymbol{g}^j\right)\max\left\{\left|\boldsymbol{g}^j\right| - \lambda\mu^{-1},\, 0\right\}, \tag{17}$$

where $[\cdot]_{+}$, $\operatorname{sgn}(\cdot)$, $|\cdot|$, and $\max\{\boldsymbol{a},\boldsymbol{b}\}$ denote the non-negative projection, algebraic sign function, absolute value function, and maximum function of $\boldsymbol{a}$ and $\boldsymbol{b}$, respectively, where each operates element-wise, and $\boldsymbol{g}^j = \boldsymbol{x}^{j+1} + \mu^{-1}\boldsymbol{h}_1^j$. Problem (15c) is an unconstrained least-squares problem, and it has the closed-form solution

$$\boldsymbol{y}^{j+1} = \left(\mathbf{B}^T\mathbf{B} + \mu\mathbf{I}\right)^{-1}\left(\mathbf{B}^T\boldsymbol{z} + \mu\left(\boldsymbol{x}^{j+1} + \mu^{-1}\boldsymbol{h}_2^j\right)\right), \tag{18}$$

where $\mathbf{I}$ is the identity matrix of proper dimension.

The details of the above ADMM procedure for problem (10) are summarized in Algorithm 2. The initializations are $\mu = 1.0$, $\beta = 1.01$, and $\lambda = 0.001$. The ADMM iteration will stop when the relative change in $\mathcal{L}_\mu$ is smaller than a preset threshold.

From an optimization perspective, problem (6) is a special case of problem (5) without the sparsity pursuit, and it can therefore be executed in a similar fashion; we omit the details due to space limitations.

Algorithm 2: ADMM for problem (10)

```
input : $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{G}$, $\mathbf{F}$, $\mathbf{A}^k$, $\lambda > 0$, $\mu > 0$.
initialize $\boldsymbol{s}^0 = \boldsymbol{y}^0 = \boldsymbol{h}_1^0 = \boldsymbol{h}_2^0 = \mathbf{0}$ and $j = 0$.
compute $\boldsymbol{z}$ and $\mathbf{B}$ by (9).
while not converged do
    compute $\boldsymbol{x}^{j+1}$ by (16).
    compute $\boldsymbol{s}^{j+1}$ by (17).
    compute $\boldsymbol{y}^{j+1}$ by (18).
    compute $\boldsymbol{h}_1^{j+1}$ and $\boldsymbol{h}_2^{j+1}$ by (13).
    update $j := j + 1$ and $\mu := \min\{\beta\mu, 10^6\}$ for some
    $\beta > 1$.
end
output: estimated $\mathbf{S}^k$ from the reversed vectorization of
        $\boldsymbol{x}^{j+1}$.
```

3. EXPERIMENTS

We test our algorithm on twenty scenes of a public database of hyperspectral images [17] that consists of everyday objects such as paintings, toys, and food. For each scene there are $M_h = 31$ spectral bands ranging from 400nm to 700nm in increments of 10nm, each band taking the form of a 512 by 512 pixel image; we use this data as our ground-truth $\mathbf{Z}$, and we simulate the RGB observation $\mathbf{Y}$ by applying equation (3) using a spectral transform matrix $\mathbf{F}$ based on the response of a Nikon D700 camera[1]. To simulate the observed $\mathbf{X}$, we naively downsample the ground-truth $\mathbf{Z}$ by uniformly averaging over each disjoint 8 by 8 pixel block in the original high resolution hyperspectral data.

An example of the input data $\mathbf{X}$ and $\mathbf{Y}$ can be seen in Figure 1, where the blown-up image in row (a) is simulated from $\mathbf{Z}$ via equation (3). The blown-up images in rows (b) and (c) allow the reader to compare the simulated low resolution $\mathbf{X}$ to its ground truth $\mathbf{Z}$.

	G Known Perfectly								G Known Imperfectly							
	RMSE				PSNR				RMSE				PSNR			
Image	Ours	MF	CNMF	PCA	Ours	MF	CNMF	PCA	Ours	MF	CNMF	PCA	Ours	MF	CNMF	PCA
Balloons	2.1	3.2	2.8	3.8	43.5	41.2	42.9	40.9	2.1	2.9	3.0	3.8	43.1	41.7	41.4	40.9
Balls	1.9	3.7	3.6	4.2	46.3	44.0	44.5	42.4	2.0	3.9	3.8	4.2	46.3	44.0	42.5	42.4
Beads	6.1	7.2	6.6	8.0	37.0	34.8	35.7	33.8	7.2	5.0	7.2	8.0	36.1	36.4	33.5	33.8
Beans	4.2	7.0	6.2	7.7	40.1	35.6	37.6	36.5	4.0	8.5	6.2	7.7	39.6	35.6	37.4	36.5
CD	6.5	10	11	12	35.3	32.0	31.4	30.5	5.8	11	10	12	35.5	32.2	31.8	30.5
Clay	3.1	2.9	8.9	3.9	37.4	40.3	30.9	37.5	2.8	2.8	8.1	3.9	38.5	40.7	31.6	37.5
Cloth	9.5	6.2	20	7.5	32.4	34.1	24.3	36.7	9.8	6.1	20	7.5	31.5	34.5	24.2	36.7
Face	3.4	2.6	11	2.3	37.6	39.8	26.9	41.2	3.8	3.0	10	2.3	36.6	38.4	27.6	41.2
Feathers	2.9	5.6	5.3	7.7	42.0	37.4	37.7	35.6	3.1	4.6	5.4	7.7	41.4	40.3	40.0	35.6
Flowers	4.0	5.2	7.6	4.8	36.2	36.7	30.2	40.5	3.4	4.8	7.3	4.8	37.5	37.2	30.6	40.6
Hairs	2.3	3.5	3.4	3.0	43.4	42.0	40.7	45.6	2.0	3.6	2.8	3.1	46.4	41.9	43.3	45.6
Painting	6.7	4.2	26	4.3	34.7	37.9	22.8	40.6	5.9	4.7	21	4.3	35.3	36.6	23.8	40.5
Paints	4.5	7.1	9.8	5.9	38.1	35.6	30.0	37.7	4.8	6.8	8.9	5.9	38.0	35.6	30.8	37.7
Peppers	2.1	4.4	5.5	4.3	44.2	41.3	37.3	41.3	2.5	5.8	5.7	4.3	42.8	39.0	36.9	41.3
Sponges	2.0	2.9	4.0	8.2	42.0	40.9	36.7	32.8	1.8	2.9	3.5	8.2	43.2	40.7	39.1	32.8
Spools	5.3	5.8	15	5.6	37.8	38.5	28.2	40.4	5.5	9.3	15	5.6	36.0	36.4	28.2	40.4
Statue	4.3	2.1	16	1.4	39.0	45.4	28.1	49.4	4.3	1.7	14	1.4	39.8	45.7	28.8	49.5
Tiles	7.4	4.3	23	5.8	35.7	40.9	25.1	37.4	7.1	4.9	20	5.8	35.8	39.0	25.7	37.4
Toys	3.0	6.1	5.0	5.2	40.1	35.0	38.3	37.5	2.9	5.8	5.0	5.2	40.7	34.5	35.1	37.5
Waterclr.	3.6	5.7	7.1	5.2	38.6	35.3	30.9	38.1	3.3	5.9	4.4	5.2	39.5	34.3	36.4	38.1
Average	4.2	5.0	9.8	5.5	39.1	38.4	33.0	38.8	4.2	5.2	9.1	5.5	39.2	38.2	33.4	38.8

Table 1. Results of the proposed algorithm against Matrix Factorization (MF) [6], Coupled Nonnegative Matrix Factorization (CNMF) [7] and Principal Component Analysis (PCA) [16] with RMSE and PSNR error measures under two scenarios: $\mathbf{G}$ known perfectly and imperfectly. Results highlighted in green to red are the best to worst, respectively.

3.1. Baseline Comparisons

For comparison, we implement both the Matrix Factorization (MF) method proposed by Kawakami *et al.* in [6] and the Coupled Nonnegative Matrix Factorization (CNMF) method proposed by Yokoya *et al.* in [7]. For all methods, including ours, we set $N = 10$. For CNMF, we initialize $\mathbf{A}$ with AVMAX [15] and allow the algorithm to converge over as many iterations as needed, with each set of inner iterations capped at 300.

Additionally, we also test a simple Principal Component Analysis (PCA) [16] to compute the material basis $\mathbf{A}$ and then uses equation (3) to directly solve for $\mathbf{S}$. Here we specifically change $N = 3$ as do Kawakami *et al.* because we find that PCA performs poorly with $N = 10$.

3.2. The Effect of G

Because we may not know the spatial relationship $\mathbf{G}$ in a real life scenario, we test all algorithms under two separate conditions. In the first condition we assume that we know $\mathbf{G}$ perfectly. The simulated $\mathbf{X}$ that we use is produced by simply applying equation (2) with a $\mathbf{G}$ that uniformly downsamples each 8 by 8 pixel block of the high resolution hyperspectral data. Subsequently, when we reconstruct $\mathbf{Z}$ from $\mathbf{X}$ and $\mathbf{Y}$ we use the same $\mathbf{G}$ matrix.

[1]Camera response figures can be found at `www.maxmax.com`.

In the second condition we assume that we do not know $\mathbf{G}$ perfectly. Thus to simulate this we use a different downsampling method to produce $\mathbf{X}$. For convenience, we choose Matlab's bicubic downsampling function with a downsampling factor of 8. For reconstruction we use the naive $\mathbf{G}$ used in the previous scenario.

3.3. Error Measures

To test the accuracy of our proposed algorithm and the comparisons, we use both the root-mean-square errors (RMSE) between the estimated $\hat{\mathbf{Z}}$ and the ground-truth $\mathbf{Z}$ (across all spectral bands)

$$\text{RMSE} = \sqrt{\frac{\|\mathbf{Z} - \hat{\mathbf{Z}}\|_F^2}{M_h L_c}}, \tag{19}$$

and the average peak signal-to-noise ratios (PSNR) defined as:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{1}{M_h}\sum_{i=1}^{M_h}\frac{\text{MAX}_i}{\text{MSE}_i}\right) \tag{20}$$

where MAX_i and MSE_i are the maximum pixel value and the mean-square error of $\hat{\mathbf{Z}}$ in the ith band, respectively. With these metrics, the best algorithm is the one that produces the smallest RMSE and the largest PSNR on average.

4. RESULTS AND CONCLUSION

Full results are reported in Table 1. The proposed method clearly outperforms all others when taken on average over our data set. Even when we remove the assumption that we know $\mathbf{G}$ perfectly, our method performs the best. These trends hold true in both error measures, RMSE and PSNR. Notably, CNMF shows the worst performance while the performances of PCA and Matrix Factorization are comparable.

A closer examination of the data suggests that images with many materials (or many apparent colors) that vary rapidly are the most difficult to accurately reconstruct. Table 1 shows that our algorithm has the most difficulty on these scenes (such as “Cloth,” “Painting,” and “Statue”). In Figure 1, for instance, it is apparent that errors occur along borders between colors and in regions where many colors blend together or change from one to the other. All of the methods we tested seem susceptible to this challenge, and evidence of this can be seen in the 420-430nm band of “Watercolors” (first column from the left of Figure 1) in which errors spike sharply at the borders between colors. In addition, however, both MF and CNMF to varying degrees appear to have difficulty in accurately reconstructing entire material spectra, suggesting that the algorithms do not lead $\mathbf{A}$ to converge well. Indeed, PCA also seems to suffer from this difficulty. Our method, however, is unique in that its framework allows for repeated updates of $\mathbf{A}$ from $\mathbf{S}$, and vice versa, which may explain its superior performance on images like “Watercolors” shown in Figure 1. In addition, our method makes direct use of $\mathbf{G}$ whereas CNMF only uses this information during initializations and MF does not use it at all.

It is conceivable that the relative performance of these algorithms may differ under other conditions: higher or lower downsampling rates, for instance (Kawakami *et al.* use a downsampling factor of 32, Yokoya *et al.* use 6, and we use 8), or different numbers of spectral bands in both $\mathbf{X}$ and $\mathbf{Y}$. A separate consideration is the computation time. For the 512 by 512 images used in our experiments both our method and CNMF take many hours to complete one reconstruction, whereas Matrix Factorization usually takes less than one hour, and PCA takes only a few minutes.

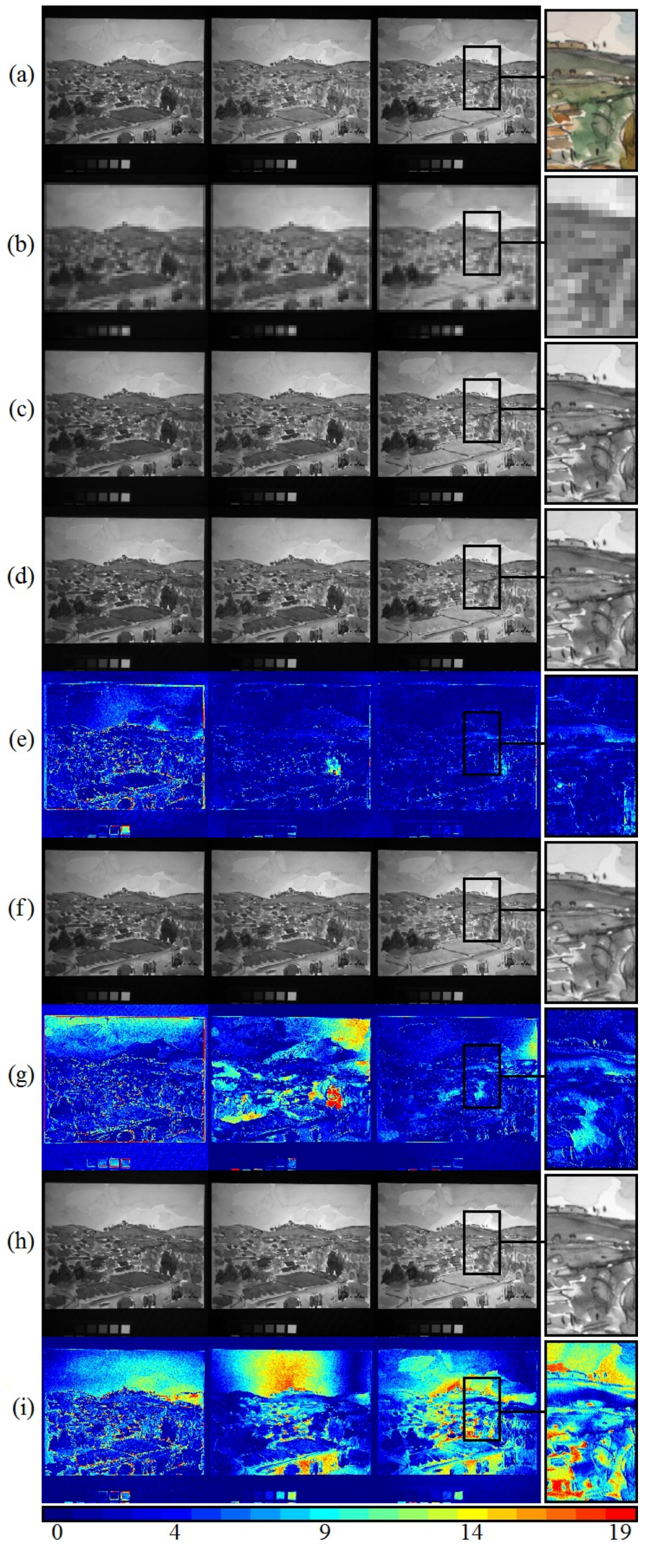

Fig. 1. Results for “Watercolors” with G known, from left to right: (a) RGB bands of $\mathbf{Y}$; remaining rows are the 420-430nm, 490-500nm and 600-610nm bands of: (b) $\mathbf{X}$, (c) ground truth $\mathbf{Z}$, (d) our reconstruction, (e) our error, (f) MF reconstruction, (g) MF error, (h) CNMF reconstruction, (i) CNMF error. Errors are on an 8-bit scale.

5. REFERENCES

[1] A. C. Rice, J. R. Vasquez, J. Kerekes, and M. J. Mendenhall, "Persistent hyperspectral adaptive multi-modal feature-aided tracking," in *SPIE Defense, Security, and Sensing*, 2009, p. 73340M.

[2] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson, "Segmentation and classification of hyperspectral images using watershed transformation," *Pattern Recognition*, vol. 43, no. 7, pp. 2367–2379, 2010.

[3] L. Varsano, I. Yatskaer, and S. R. Rotman, "Temporal target tracking in hyperspectral images," *Optical Engineering*, vol. 45, no. 12, pp. 126201, 2006.

[4] J. Blackburn, M. Mendenhall, A. Rice, P. Shelnutt, N. Soliman, and J. Vasquez, "Feature aided tracking with hyperspectral imagery," *Proc. of SPIE*, p. 66990S, 2007.

[5] A. Greiwe and M. Ehlers, "Combined analysis of hyperspectral and high resolution image data in an object oriented classification approach," in *Proc. International Symposium on Remote Sensing and Data Fusion over Urban Areas*, 2005, pp. 13–15.

[6] R. Kawakami, J. Wright, Y.W. Tai, Y. Matsushita, M. Ben-Ezra, and K. Ikeuchi, "High-resolution hyperspectral imaging via matrix factorization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 2329–2336.

[7] N. Yokoya, T. Yairi, and A. Iwasaki, "Coupled nonnegative matrix factorization unmixing for hyperspectral and multispectral data fusion," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 2, pp. 528–537, 2012.

[8] Daniel D. Lee and H. Sebastian Seung, "Algorithms for non-negative matrix factorization," in *NIPS*. 2001, pp. 556–562, MIT Press.

[9] Daniel D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, Oct. 1999.

[10] John Wright Quan Geng, Huan Wang, "On the local correctness of ℓ_1 minimization for dictionary learning," ArXiv preprint arXiv:1101.5672, 2011.

[11] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, pp. 475–494, 2001.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and trends in machine learning*, vol. 3, pp. 1–122, 2011.

[13] D. P. Bertsekas, *Nonlinear Programming*, MA: Athena Scientific, 1999.

[14] M. Hong and Z.-Q. Luo, "On the linear convergence of the alternating direction method of multipliers," ArXiv preprint arXiv:1208.3922, 2012.

[15] T.-H. Chan, W.-K. Ma, A. Ambikapathi, and C.-Y. Chi, "A simplex volume maximization framework for hyperspectral endmember extraction," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4177–4193, 2011.

[16] I. T. Jolliffe, *Principal Component Analysis*, Springer, New York, 2002.

[17] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar, "Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum," *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2241–2253, 2010.