Unsupervised Classification in Hyperspectral Imagery with Nonlocal Total Variation and Primal-Dual Hybrid Gradient Algorithm

Wei Zhu, Victoria Chayes, Alexandre Tiard, Stephanie Sanchez, Devin Dahlberg, Da Kuang, Andrea L. Bertozzi, Stanley Osher, and Dominique Zosso

arXiv:1604.08182v1 [cs.CV] 27 Apr 2016

***Abstract*—We propose a graph-based nonlocal total variation method (NLTV) for unsupervised classification of hyperspectral images (HSI). The variation problem is solved by the primal-dual hybrid gradient (PDHG) algorithm. By squaring the labeling function and using a stable simplex clustering routine, we can implement an unsupervised clustering method with random initialization. Finally, we speed up the calculation using a k-d tree and approximate nearest neighbor search algorithm for calculation of the weight matrix for distances between pixel signatures. The effectiveness of this proposed algorithm is illustrated on both synthetic and real-world HSI, and numerical results show that our algorithm outperform other standard unsupervised clustering methods such as spherical K-means, nonnegative matrix factorization (NMF), and the graph-based Merriman-Bence-Osher (MBO) scheme.**

***Index Terms*—Hyperspectral images (HSI), nonlocal total variation (NLTV), primal-dual hybrid gradient (PDHG) algorithm, unsupervised classification, stable simplex clustering**

I. Introduction

HYPERSPECTRAL imagery (HSI) is an important domain in the field of remote sensing with numerous applications in agriculture, environmental science, mineralogy, medical imaging, and surveillance [1]. Hyperspectral sensors capture information of intensity of reflection at different wavelengths, from the infrared to ultraviolet. They typically take measurements approximately 10-30nm apart, and as many as 200 sample layers for a single image. Thus each pixel has a unique spectral signature. These spectral signatures can be used to differentiate objects that cannot be distinguished based on visible spectra, for example: invisible gas plumes, oil or chemical spills over water, or healthy from unhealthy crops. Hence an important task in HSI analysis is to classify the pixels in the image.

There are two types of methods in HSI classification: *unmixing* methods and *clustering* methods. Unmixing methods attempt to extract the information of the constitutive materials (the *endmembers*) and the abundance map, which can be further used to create a segmentation of the image [2]–[5]. Clustering methods do not extract endmembers; instead, they return the spectral signatures of the centroids of the clusters. These centroids are merely the mean of the signatures of all the pixels of the cluster, and do not necessarily represent the true signatures of the original materials. However, when we assume that most of the pixels are dominated mostly by one endmember, i.e. in the absence of partial volume effects, which is usually the case for high-resolution HSI, these two types of methods are expected to give very similar results [5]. Our nonlocal total variation (NLTV) method for HSI classification in this paper is a clustering method.

Much work has been carried out in the literature in both the unmixing and the clustering categories. HSI unmixing models can be characterized as linear or nonlinear. In a linear mixing model (LMM), each pixel is approximated by a linear combination of the endmembers. When the linear coefficients are constrained to be nonnegative, it is equivalent to nonnegative matrix factorization (NMF), and good unsupervised clustering results have been achieved in [3]–[5] using either NMF or hierarchical rank-2 NMF (H2NMF). Despite the simplicity of LMM, the linear mixing assumption has been shown to be physically inaccurate [6]. Researchers are starting to expand aggressively into the much more complicated nonlinear unmixing realm [7], where nonlinear effects such as atmospheric scattering are explicitly modeled. However, most of the work that has been done for nonlinear unmixing so far is supervised in the sense that a prior knowledge of the endmember signatures is required [2]. Discriminative machine learning methods such as support vector machine (SVM) [8]–[10] and relevance vector machine (RVM) [11]–[13] based approaches have also been applied to hyperspectral images, generating pixel classes without modeling the endmembers, but they are also supervised methods since a training set is needed to learn the classifiers.

On the contrary, graph-based clustering methods implicitly model the nonlinear mixing effects between the endmembers. This type of methods does not take the original spectral signatures as input, but is built upon a weight matrix that encodes the similarity between the pixels, which is typically a sparse matrix constructed using the distances between the spectral signatures. Graph cuts problems for graph segmentation have been well studied in the literature [14]–[17]. In 2012, Bertozzi and Flenner proposed a diffuse interface model on graphs

This work was supported by NSF grants DMS-1118971, DMS-1045536, DMS-0914856, and ONR grant N00014-16-1-2119

Wei Zhu, Da Kuang, Andrea L. Bertozzi, Stanley Osher, Dominique Zosso, and Stephanie Sanchez are with the Department of Mathematics at University of California, Los Angeles. Email:{weizhu731, dakuang, bertozzi, sjo, zosso}@math.ucla.edu, stephanie2000@g.ucla.edu

Victoria Chayes is with Bard College. Email: vminervachayes@gmail.com

Alexandre Tiard is with ENSE3, Grenoble Institute of Technology. Email: alexandretiard@gmail.com

Devin Dahlberg is with University of California, San Diego. Email: dahlbergdevin@gmail.com

with applications to classification of high dimensional data [18]. This idea has been combined with Merriman-Bence-Osher (MBO) scheme [19] and applied to multi-class graph segmentation [20], [21] and HSI classification [22], [23]. The method in [18] minimizes a graph version of the Ginzburg-Landau (GL) functional, which consists of the Dirichlet energy of the labeling function and a double well potential, and uses Nyström extension to speed up the calculation of the eigenvectors for inverting the graph Laplacian. This graph-based method performed excellently compared to other algorithms in the detection of chemical plumes in hyperspectral video sequences, which is a difficult problem because of the diffusive nature of the gas [22], [23]. However, the GL functional is non-convex due to its double-well term, which may cause the algorithm to get stuck in local minima. This issue can be circumvented by running the algorithm multiple times with different initial conditions and hand-picking the best result.

The two methods proposed in this paper are totally unsupervised graph-based clustering techniques. Instead of minimizing the GL functional, which has been proved to converge to the total variation (TV) semi-norm, we propose to minimize the NLTV semi-norm of the labeling functions directly:

$$E_1(u) = \sum_{l=1}^{k} \|\nabla_w u_l\|_{L^1} + \sum_{l=1}^{k} \int u_l(x) f_l(x) dx, \qquad (1)$$

where ∇_w is the nonlocal gradient, $u = (u_1, u_2, \ldots, u_k)$ is the labeling function mapping the image into the unit simplex, and f_l is the error function with respect to the l-th centroid. The L^1 regularized problem above is a convex optimization problem and can be solved by the primal-dual hybrid gradient (PDHG) algorithm, which avoids the need to invert the graph Laplacian. We also introduce the novel idea of the quadratic model by squaring the labeling function u in the fidelity term:

$$E_2(u) = \sum_{l=1}^{k} \|\nabla_w u_l\|_{L^1} + \sum_{l=1}^{k} \int u_l^2(x) f_l(x) dx. \qquad (2)$$

This new energy ensures that anomalies converge to their own clusters and makes random endmember initialization possible in our algorithm. Our direct usage of the NLTV semi-norm makes our clustering methods more accurate than other methods when evaluated quantitatively on HSI with ground-truth labels, and our quadratic model for random initialization with stable simplex clustering is a completely new addition to the field of hyperspectral image classification.

This paper is organized as follows: in Section II background is provided on total variation, nonlocal operators, and the primal-dual hybrid gradient algorithm. In Section III we introduce our work: first we apply PDHG to the linear NLTV model, then we apply PDHG to our new innovation of the quadratic NLTV model. We also introduce a stable simplex clustering scheme, which allows the algorithm to converge in fewer iterations. Section IV outlines a computer science technique for speeding up the calculation of the weight matrix, namely, the formation of a k-d tree then the application of an approximate nearest neighbor search, which vastly reduces the number of pixels that need to be compared. Section V presents our results. We run our algorithm on both synthetic and real-world datasets such as Urban, San Diego Airport, and Chemical Plume, and Section VI presents our conclusions.

II. Total Variation and Primal-Dual Hybrid Gradient Algorithm

A. Total Variation and Nonlocal Operators

Total variation (TV) method was introduced by Rudin et al in 1992 [24] and has been applied to various image processing tasks, such as image denoising, deconvolution, inpainting, and segmentation [25]. Its advantage is that one can preserve the edges in the image when minimizing $\|\nabla u\|_{L^1}$ (TV semi-norm). The total variation model is:

$$\min_u E(u) = \|\nabla u\|_{L^1} + \frac{\lambda}{2} S(u). \qquad (3)$$

The parameter λ can be adjusted to give higher priority to the TV-regularizing term, or the data fidelity term $S(u)$.

Despite its huge success in image processing, the total variation method is still a local method. More specifically, when the gradient of a pixel is being calculated, this is done using its immediate adjacent pixels. It is known that local image processing techniques fail to produce satisfactory results when the image has repetitive structures, or intrinsically related objects in the image are not spatially connected. To address this problem, Buades et al proposed a nonlocal mean method based on patch distances for image denoising [26]. Gilboa and Osher [27] later formalized a systematic framework for nonlocal image processing. Nonlocal image processing produces much better results because theoretically any pixel in the image can interact with any other, which better preserves texture and fine detail.

In HSI classification, clusters can have elements that are not spatially connected. Thus to use total variation as a segmenting technique, it is necessary to develop a nonlocal method of calculating the gradient. We provide a review of nonlocal operators for readers unfamiliar with the prior work in the rest of this section. Note that in our utilization, the model is continuous and the weights are not necessarily symmetric [28].

Let Ω be a region in $\mathbb{R}^n$, and $u : \Omega \to \mathbb{R}$ be a real function. We define the nonlocal derivative:

$$\frac{\partial u}{\partial y}(x) := \frac{u(y) - u(x)}{d(x, y)}, \quad \text{for all } x, y \in \Omega,$$

where d is a positive distance between x and y. With the following nonlocal weight defined as (4), we can rewrite the nonlocal derivative as (5).

$$w(x, y) = d^{-2}(x, y), \qquad (4)$$

$$\frac{\partial u}{\partial y}(x) = \sqrt{w(x, y)}(u(y) - u(x)). \qquad (5)$$

With this nonlocal derivative, we can define the nonlocal gradient $\nabla_w u$ for $u \in L^2(\Omega)$ as a function from Ω to $L^2(\Omega)$; we therefore use the notation $\nabla_w u \in L^2(\Omega, L^2(\Omega))$. Then $\nabla_w u \in L^2(\Omega, L^2(\Omega))$ is the collection of all partial derivatives:

$$\nabla_w u(x)(y) = \frac{\partial u}{\partial y}(x) = \sqrt{w(x, y)}(u(y) - u(x)).$$

We use the standard L^2 inner product on Hilbert spaces $L^2(\Omega)$ and $L^2(\Omega, L^2(\Omega))$. More specifically, for $u_1, u_2 \in L^2(\Omega)$ and $v_1, v_2 \in L^2(\Omega, L^2(\Omega))$

$$\langle u_1, u_2 \rangle := \int_\Omega u_1(x) u_2(x) dx,$$
$$\langle v_1, v_2 \rangle := \int_\Omega \int_\Omega v_1(x)(y) v_2(x)(y) dy dx.$$

With the above definition of inner products and nonlocal gradient, the nonlocal divergence div_w is defined as the negative adjoint of the nonlocal gradient:

$$\text{div}_w v(x) := \int_\Omega \sqrt{w(x,y)} v(x)(y) - \sqrt{w(y,x)} v(y)(x) dy.$$

At last, we can define a standard L^1 and L^∞ norm on the space $L^2(\Omega, L^2(\Omega))$:

$$\|v\|_{L^1} := \int_\Omega \|v(x)\|_{L^2} dx = \int_\Omega \left| \int_\Omega |v(x)(y)|^2 dy \right|^{\frac{1}{2}} dx,$$
$$\|v\|_{L^\infty} := \sup_x \|v(x)\|_{L^2}.$$

B. Primal-Dual Hybrid Gradient Algorithm

A lot of energy functionals involving NLTV semi-norms in image processing are convex, and there is a huge collection of convex optimization techniques that can be applied to such problems. First-order primal-dual algorithms have recently been successfully used in image processing with L^1 type regularizers [29]–[32]. We introduce the framework here to contextualize our extension to nonlocal model for hyperspectral imagery.

Let X and Y be two finite-dimensional real vector spaces. Let F and G be proper convex lower semi-continuous functions $F : Y \to [0, \infty), G : X \to [0, \infty)$, and F^* the convex conjugate of F, and $K : X \to Y$ a continuous linear operator with the operator norm,

$$\|K\| = \sup\{\|Kx\| : x \in X, \|x\| \leq 1\}.$$

For the total variation energy functional (3), we can define the gradient ∇ as K, $\|\cdot\|_{L^1}$ as F, and the fidelity term $\frac{\lambda}{2} S(u)$ as G. Then minimizing (3) is equivalent to minimizing the primal problem:

$$\min_{x \in X} \{F(Kx) + G(x)\}. \tag{6}$$

Then the primal-dual formulation of (6) is the saddle-point problem:

$$\min_{x \in X} \max_{y \in Y} \{\langle Kx, y \rangle - F^*(y) + G(x)\}. \tag{7}$$

The saddle-point problem is then solved using the iterations of Algorithm 1 in [29].

In Algorithm 1, $(I + \lambda \partial f)^{-1}(x)$ is the proximal operator of f, which is defined as:

$$(I + \lambda \partial f)^{-1}(x) = \text{prox}_{\lambda f}(x) = \arg\min_y f(y) + \frac{1}{2\lambda} \|y - x\|_2^2.$$

It has been shown in [29] that $O(1/N)$ (where N is the number of iterations) convergence can be achieved as long as σ, τ satisfy

$$\sigma \tau \|K\|^2 \leq 1.$$

Algorithm 1 Primal-Dual Hybrid Gradient (PDHG) Algorithm

1: Initialization: Choose $\tau, \sigma > 0$, $\theta \in [0, 1]$, $(x^0, y^0) \in X \times Y$, and set $\bar{x}^0 = x^0$
2: **while** not converge **do**
3: $\quad y^{n+1} = (I + \sigma \partial F^*)^{-1}(y^n + \sigma K \bar{x}^n)$
4: $\quad x^{n+1} = (I + \tau \partial G)^{-1}(x^n - \tau K^* y^{n+1})$
5: $\quad \bar{x}^{n+1} = x^{n+1} + \theta(x^{n+1} - x^n)$
6: $\quad n = n + 1$
7: **end while**

III. Two NLTV Models for Unsupervised Classification of HSI

In this section, we propose two NLTV models for unsupervised classification of HSI. The linear model runs faster in each iteration, but it requires an accurate centroid initialization. The quadratic model runs slower in each iteration, but it makes random initialization of the centroids possible. And the quadratic model converges faster if the initialization is not very accurate.

A. Linear Model

We extend the idea from [29] to formulate a linear model for classification on HSI. The linear model seeks to minimize the following energy

$$\begin{aligned} E_1(u) &= \|\nabla_w u\|_{L^1} + \langle u, f \rangle \\ &= \sum_{l=1}^k \|\nabla_w u_l\|_{L^1} + \sum_{l=1}^k \int u_l(x) f_l(x) dx, \end{aligned} \tag{8}$$

where $\mathbb{K}^k = \{(x_1, x_2, \ldots, x_k) | \sum_{i=1}^k x_i = 1, x_i \geq 0\}$ is the unit simplex in $\mathbb{R}^k$, $u = (u_1, u_2, \ldots, u_k) : \Omega \to \mathbb{K}^k$ is the labeling function, k is the number of clusters, $\nabla_w u = (\nabla_w u_1, \ldots, \nabla_w u_k)$ such that $\|\nabla_w u\|_{L^1} = \sum_{l=1}^k \|\nabla_w u_l\|_{L^1}$, and $f_l(x)$ is the error function defined as

$$f_l(x) = \frac{\lambda}{2} |g(x) - c_l|_\mu^2, \tag{9}$$

where $g(x)$ is the spectral signature at pixel x and c_l is the spectral signature of the l-th centroid, which is either picked randomly from the HSI or generated by any fast unsupervised centroid extraction algorithm (e.g. H2NMF, spherical K-means.) The distance in the above definition of $f_l(x)$ is a linear combination of cosine distance and Euclidean distance:

$$|g(x) - c_l|_\mu = 1 - \frac{\langle g(x), c_l \rangle}{\|g(x)\|_2 \|c_l\|_2} + \mu \|g(x) - c_l\|_2, \quad \mu \geq 0.$$

In HSI processing, the cosine distance is generally used because it is more robust in the face of atmospheric interference and topographical features [33]. The reason why we also use the Euclidean distance is that sometimes different classes have very similar spectral angles, but vastly different spectral amplitudes (e.g. "dirt" and "road" in the Urban dataset, which is illustrated in Section V.) We call this the linear model since the power of the labeling function u_l in (8) is 1. Now we discuss how to discretize (8) for numerical implementation.

1) Weight Matrix: First, we address the creation of a weight matrix measuring similarities between spectral signatures. Following the idea from [27], we define the patch distance as:

$$d_\sigma(x,y) = \int_\Omega G_\sigma(t) \left| g(x+t) - g(y+t) \right|^2 dt,$$

where G_σ is a Gaussian of standard deviation σ. The Patch distance in RGB image processing addresses the situations when the image is repetitive in space. Then we discretize and binarize the weights in the following way: we take a patch and a search window around every pixel i, and compute the patch distance $(d_\sigma)_{i,j}$ to all the points within the search window and select the m closest ones. For the m closest pixels, the $w_{i,j}$ are set to be 1, while all the other weights are set to be 0. In our experiments, we have used 3×3 and 1×1 patches, and 21×21 search windows, and m is set to be 10. There is one more thing to mention about the patch size: unlike RGB image processing, the patch size for HSI does not have to be very large, and 3×3 patches, or in some extreme cases 1×1 patches, can generate similar results to that of 11×11 patches. The reason is that while low dimensional RGB images require spatial context to identify pixels, high dimensional hyperspectral images already encode enough information for each pixel in the spectral dimension. Of course, a larger patch size that is consistent with the spatial resolution of the HSI will still be preferable when significant noise is present.

2) The Labeling Function and the Nonlocal Operators: We discretize the labeling function u and nonlocal operators ∇_w in the following way: $u = (u_1, u_2, \ldots, u_k)$ is discretized as a matrix of size $r \times k$, where r is the number of pixels in the hyperspectral image, and $(u_l)_j$ is the l-th labeling function at j-th pixel; $(\nabla_w u_l)_{i,j} = \sqrt{w_{i,j}}((u_l)_j - (u_l)_i)$ is the nonlocal gradient of u_l; $(\mathrm{div}_w v)_i = \sum_j \sqrt{w_{i,j}} v_{i,j} - \sqrt{w_{j,i}} v_{j,i}$ is the divergence of v at i-th pixel; and the discrete L^1 and L^∞ norm of $\nabla_w u_l$ are defined as:

$$\|\nabla_w u_l\|_{L^1} = \sum_i \Big(\sum_j (\nabla_w u_l)_{i,j}^2 \Big)^{\frac{1}{2}},$$

$$\|\nabla_w u_l\|_{L^\infty} = \max_i \Big(\sum_j (\nabla_w u_l)_{i,j}^2 \Big)^{\frac{1}{2}}.$$

Next, we will use similar idea from [29] to derive an algorithm for minimizing the energy E_1 in (8). By adding an indicator function δ_U, minimizing E_1 is equivalent to solving the following primal problem:

$$\min_u \|\nabla_w u\|_{L^1} + \langle u, f \rangle + \delta_U(u), \tag{10}$$

where $U = \{u = (u_1, u_2, \ldots, u_k) \in \mathbb{R}^{r \times k} : \sum_{l=1}^k (u_l)_i = 1, \forall i = 1, \ldots, r, (u_l)_i \geq 0\}$, and δ_U is the indicator function on U. More specifically:

$$\delta_U(u) = \begin{cases} 0 & \text{if } u \in U, \\ \infty & \text{otherwise.} \end{cases}$$

The primal-dual saddle-point formulation of (10) is:

$$\min_{u=(u_l)_{l=1}^k} \max_{p=(p_l)_{l=1}^k} \langle \nabla_w u, p \rangle + \langle u, f \rangle + \delta_U(u) - \delta_P(p), \tag{11}$$

where δ_P is the convex conjugate of $\|\cdot\|_{L^1}$, and the set $P = \{p \in \mathbb{R}^{(r \times r) \times k} : \|p_l\|_\infty \leq 1\}$.

Algorithm 2 Primal-Dual Iterations for Linear Model

1: **while** not converge **do**
2: $p^{n+1} = \mathrm{proj}_P(p^n + \sigma \nabla_w \bar{u}^n)$
3: $u^{n+1} = \mathrm{proj}_U(u^n + \tau \mathrm{div}_w p^{n+1} - \tau f)$
4: $\bar{u}^{n+1} = u^{n+1} + \theta(u^{n+1} - u^n)$
5: $n = n + 1$
6: **end while**

We can then use the PDHG algorithm (see Algorithm 2) to solve the above saddle point problem (11). We specify the two orthogonal projections in Algorithm 2: let $\tilde{p} = \mathrm{proj}_P(p)$, where $p = (p_l)_{l=1}^k \in \mathbb{R}^{(r \times r) \times k}$. Then for every $i \in \{1, 2, \ldots, r\}$ and every $l \in \{1, 2, \ldots, k\}$, the i-th row of $\tilde{p}_l$ is the projection of the i-th row of p_l on to the unit ball in $\mathbb{R}^r$. Similarly, if $\tilde{u} = \mathrm{proj}_U(u)$, then for every $i \in \{1, 2, \ldots, r\}$, $((\tilde{u}_1)_i, (\tilde{u}_2)_i, \ldots, (\tilde{u}_k)_i)$ is the projection of $((u_1)_i, (u_2)_i, \ldots, (u_k)_i)$ onto the unit simplex $\mathbb{K}^k$ in $\mathbb{R}^k$. And from [34], we know at most k steps are needed to project an arbitrary vector in $\mathbb{R}^k$ onto $\mathbb{K}^k$.

Notice that no matrix inversion is involved in the above primal-dual iteration, as opposed to general graph Laplacian methods. The most expensive part in the computation comes from sparse matrix multiplications, which is still very cheap due to the fact that only 10 nonzero elements are kept in each row of the nonlocal weight matrix.

Next, we address centroid updates and stopping criteria for our linear model. The concept of centroid updates in the linear model is not a new one; in fact, the standard K-means algorithm consists of two steps: first, it assigns each point to a cluster whose mean yields the least within-cluster sum of squares, then it re-calculates the means from the centroids, and terminates when assignments no longer change [35]. Especially for data-based methods, re-calculating the centroid is essential for making the algorithm less sensitive to initial conditions and more likely to find the "true" clusters.

After every few steps of primal-dual iterations (we used five steps in our experiments), the output u will be thresholded to u_{hard}. More specifically, for every $i \in \{1, 2, \ldots, r\}$, we pick the largest element among $((u_1)_i, (u_2)_i, \cdots, (u_k)_i)$ and set it to 1, while leaving the others at 0, and we say the i-th pixel belongs to that particular cluster. Then we update the l-th centroid by taking the mean of all the pixels in that cluster. We repeat the process until the difference between two consecutive u_{hard} drops below a certain threshold. The pseudocode for our linear model on HSI is listed in Algorithm 3.

Before ending the discussion of our proposed linear model, we would like to point out its connection to the piecewise constant Mumford-Shah model for multi-class graph segmentation [36]. Assume that the domain Ω of the HSI is segmented by a contour Φ into k disjoint regions, $\Omega = \cup_{l=1}^k \Omega_l$. The piecewise constant Mumford-Shah energy is defined as:

$$E_{MS}(\Phi, \{c_l\}_{l=1}^k) = |\Phi| + \lambda \sum_{l=1}^k \int_{\Omega_l} |g(x) - c_l|^2 \, dx, \tag{12}$$

where $|\Phi|$ is the length of the contour. To illustrate the connection between (8) and (12), let us first look at the "local"

Algorithm 3 Linear Model

1: Initialization of centroids: Choose $(c_l)_{l=1}^k$ (randomized or generated by fast unsupervised centroid extraction algorithm).
2: Initialization of parameters: Choose $\tau, \sigma > 0$ satisfying $\sigma\tau\|\nabla_w\|^2 \leq 1$, $\theta = 1$
3: Initial iterate: Set $u^0 \in \mathbb{R}^{r\times k}$ and $p^0 \in \mathbb{R}^{(r\times r)\times k}$ randomly, set $\bar{u}^0 = u^0$, $u_{hard} = threshold(u^0)$
4: **while** not converge **do**
5: N steps of primal-dual iterations for linear model
6: $u_{hard} = threshold(u)$
7: update $(c_l)_{l=1}^k$
8: **end while**

version of (8), which essentially replaces the NLTV regularizer $\|\nabla_w u_l\|_{L^1}$ with its local conterpart :

$$E_1^{\text{loc}}(u) = \sum_{l=1}^{k} \|\nabla u_l\|_{L^1} + \sum_{l=1}^{k} \int u_l(x) f_l(x) dx. \tag{13}$$

Assume that the labeling function u_l is the characteristic function of Ω_l. Then $\int u_l(x) f_l(x) dx$ is equal to $\int_{\Omega_l} |g(x) - c_l|^2 dx$ up to a multiplicative constant. Moreover, the total variation of a characteristic function of a region equals the length of its boundary, and hence $|\Phi| = \sum_{l=1}^{k} \|\nabla u_l\|_{L^1}$. So the linear model (8) can be viewed as a nonlocal convex-relaxed version of Mumford-Shah model. We would also like to point out that the linear energy (8) has been studied in previous work [22]. But in their work, the authors used a graph-based MBO method to minimize (8) instead of the PDHG algorithm, and the difference of the numerical performances can be seen in Section V.

B. Quadratic Model

1) Intuition: Our proposed linear model performs very well when the centroids are initialized by accurate centroid extraction algorithms. As we will show in Section V, the linear model can have a significant boost to the accuracy of other algorithms if the centroid extraction algorithm is accurate, without sacrificing speed. However, if endmembers are not extracted accurately, or if random pixel initialization is used, the segmenting results are no longer accurate, and the algorithm takes far more iterations to converge to a stable classification.

To reduce the times of centroid updates and merge similar clusters automatically and simultaneously, we propose the following quadratic model:

$$E_2(u) = \sum_{l=1}^{k} \|\nabla_w u_l\|_{L^1} + \sum_{l=1}^{k} \int u_l^2(x) f_l(x) dx. \tag{14}$$

Similarly to before, $u = (u_1, u_2, \ldots, u_k) : \Omega \to \mathbb{K}^k$ is the labeling function, k is the number of clusters, $\mathbb{K}^k$ is the unit simplex in $\mathbb{R}^k$, and $f_l(x)$ is the error function defined in (9).

Note that the only difference between (8) and (14) is that the power of the labeling function u_l here is 2. The intuition for this is as follows:

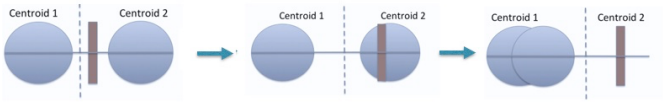

Fig. 1. "Pushing" mechanism of the quadratic model. The horizontal line represents the unit simplex in $\mathbb{R}^2$. Signatures from cluster A_1 are colored blue, and signatures from cluster A_2 are colored brown. The vertical dashed bar is generated by a stable simplex clustering method, and it thresholds the points on the simplex into two categories.

Consider for simplicity we have a hyperspectral image with a ground truth of only two clusters, A_1 and A_2. Suppose our randomized initial centroids are chosen such that $c_1 \approx c_2 \in A_1$; or, that the two random initial pixels are of very similar spectral signatures and belong to the same ground truth cluster.

Let x be a pixel from A_2. Then $0 \ll |g(x) - c_1|^2 \approx |g(x) - c_2|^2$. When we apply (8), the fidelity term $\langle u, f\rangle$ does not change when $u(x)$ moves on the simplex in $\mathbb{R}^2$, and thus pixels of A_2 will be scattered randomly on the simplex. After thresholding, an approximately equal number of pixels from cluster A_2 will belong to clusters C_1 and C_2, so the new centroids $\tilde{c}_1$ and $\tilde{c}_2$ that are the mean of the spectral signatures of the current clusters will once again be approximately equal.

This situation changes dramatically when we minimize (14). Observe that the fidelity term is minimized for a pixel $x \in A_2$ when $u_1 \approx u_2 \approx \frac{1}{2}$. Therefore, the pixels of cluster A_2 will be "pushed" toward the center of the simplex. With a stable simplex clustering method (explained in Section III-B4), we divide the clusters such that all of these pixels in the center belong to either C_1 or C_2; without loss of generality suppose they belong to C_2. Then the updated centroid $\tilde{c}_1$ is essentially c_1, while the updated centroid $\tilde{c}_2$ is a linear combination of the spectral signature of members belonging to A_1 and A_2, and thus quite different from the original c_2. After a few iterations of primal-dual algorithm, pixels from A_1 will be clustered in C_1, and pixels from A_2 will be pushed to C_2. Therefore, we will finish the clustering in just two steps. See Fig.1 for a graphical illustration.

The quadratic model not only reduces the number of iterations needed to find the "true" clustering because of its anomaly distinction, but it allows for random initialization as well, making it a more robust technique.

2) primal-dual Iterations for the Quadratic Model: We use the PDHG algorithm to minimize the energy E_2 in (14). As in the linear model, we get an unconstrained primal problem by adding an indicator function δ_U:

$$\min_u \|\nabla_w u\|_{L^1} + \sum_{l=1}^{k} \int u_l^2(x) f_l(x) dx + \delta_U(u), \tag{15}$$

where $U = \{u = (u_1, u_2, \ldots, u_k) \in \mathbb{R}^{r\times k} : \sum_{l=1}^{k} (u_l)_i = 1, \forall i = 1, \ldots, r, (u_l)_i \geq 0\}$. The primal-dual formulation of the above problem is:

$$\min_{u=(u_l)_{l=1}^k} \max_{p=(p_l)_{l=1}^k} \langle \nabla_w u, p\rangle + \sum_{l=1}^{k} \int u_l^2(x) f_l(x) dx + \delta_U(u) - \delta_P(p), \tag{16}$$

Algorithm 4 Primal-Dual Iterations for the Quadratic Model

1: **while** not converge **do**
2: $p^{n+1} = \text{proj}_P(p^n + \sigma \nabla_w \bar{u}^n)$
3: Update u^{n+1} as in (18)
4: $\bar{u}^{n+1} = u^{n+1} + \theta(u^{n+1} - u^n)$
5: $n = n + 1$
6: **end while**

where again δ_P is the convex conjugate of $\|\cdot\|_{L^1}$, and the set $P = \{p \in \mathbb{R}^{(r\times r)\times k} : \|p_l\|_\infty \leq 1\}$. To solve the saddle point problem (16), we alternate the directions and optimize p and u sequentially. Assume that we have already obtained the n-th iterates u^n, $\bar{u}^n$ and p^n.

First we solve the optimization problem with respect to p when $\bar{u}^n$ is fixed. More specifically,

$$p^{n+1} = \arg\max_p \langle \nabla_w \bar{u}^n, p\rangle - \delta_P(p) - \frac{1}{2\sigma}\|p - p^n\|^2.$$

This problem is essentially the same as that in the linear model, and p^{n+1} is updated as,

$$p^{n+1} = \text{proj}_P(p^n + \sigma \nabla_w \bar{u}^n). \tag{17}$$

Next, we alternate the direction and update u while fixing p^{n+1}:

$$u^{n+1} = \arg\min_u \langle \nabla_w u, p^{n+1}\rangle + \langle u, f \odot u\rangle + \delta_U(u) + \frac{1}{2\tau}\|u - u^n\|^2,$$

where $f \odot u$ denotes the pointwise product between f and u. We define a linear operator $A : \mathbb{R}^{r\times k} \to \mathbb{R}^{r\times k}$ such that $\frac{1}{2}Au = f \odot u$, then A is a positive semidefinite diagonal matrix of size $rk \times rk$. Taking the subgradient of the above optimization problem, we have:

$$\begin{aligned} 0 &\in -\tau \text{div}_w p^{n+1} + \tau A u^{n+1} + \tau \partial \delta_U(u^{n+1}) + (u^{n+1} - u^n),\\ 0 &\in (I + \tau A)u^{n+1} + \tau \partial \delta_U(u^{n+1}) - (u^n + \tau \text{div}_w p^{n+1}). \end{aligned}$$

Or equivalently,

$$u^{n+1} = \arg\min_u \delta_U(u) + \frac{1}{2}\|B^{\frac{1}{2}}u - B^{-\frac{1}{2}}Y\|^2, \tag{18}$$

where $B = (I + \tau A)$ and $Y = (u^n + \tau \text{div}_w p^{n+1})$. It is worth mentioning that the matrix B is diagonal and positive definite, and hence it is easy to compute the inverse and square root of B. Problem (18) is like a preconditioned projection on to a unit simplex $\mathbb{K}^k$, and we will discuss how to solve it exactly in at most k steps in the next section.

Combining (17) and (18), we have the primal-dual iterations (Algorithm 4) for our quadratic model.

3) Preconditioned Projection onto the Unit Simplex: Now we address how to solve problem (18). It is easy to see that the rows of u in (18) are decoupled, and we only need to know how to solve the following problem:

$$\min_{u\in\mathbb{R}^k} \delta_U(u) + \frac{1}{2}\|Au - y\|^2, \tag{19}$$

where A is positive definite diagonal matrix of size $k \times k$, $U = \mathbb{K}^k$ is the unit simplex in $\mathbb{R}^k$, and $y \in \mathbb{R}^k$. Our method to solve the problem above is a direct generalization of [34].

Taking the subgradient, we have:

$$\begin{aligned} & 0 \in \partial\delta_U(u) + A(Au - y)\\ \Longrightarrow\quad & A(y - Au) \in \partial\delta_U(u)\\ \Longrightarrow\quad & Au \in A\partial\delta_U^*(A(y - Au))\\ \Longrightarrow\quad & Au \in \partial(\delta_U^* \circ A)(y - Au)\\ \Longrightarrow\quad & y - Au = \text{prox}_{\delta_U^*\circ A}(y)\\ \Longrightarrow\quad & y - Au = \arg\min_z\{\delta_U^*(Az) + \frac{1}{2}\|z - y\|^2\}\\ \Longrightarrow\quad & u = A^{-1}(y - \arg\min_z\{\delta_U^*(Az) + \frac{1}{2}\|z - y\|^2\}). \end{aligned}$$

Therefore we must compute $\arg\min_z\{\delta_U^*(Az) + \frac{1}{2}\|z - y\|^2\}$. If $A = \text{diag}(a_1, a_2, \ldots, a_k)$, then:

$$\begin{aligned} &\arg\min_z\{\delta_U^*(Az) + \frac{1}{2}\|z - y\|^2\}\\ &= \arg\min_z\{\max_{1\leq l\leq k}\{a_l z_l\} + \frac{1}{2}\|z - y\|^2\}. \end{aligned} \tag{20}$$

It is not hard to prove the following theorem, which shows that it suffices to solve the problem:

$$\min_t \min_z \{t + \frac{1}{2}\|z - y\|^2, z_l \leq \frac{t}{a_l}\}. \tag{21}$$

Theorem 1: If t^* is a solution of (21), then $z^{t^*} := \arg\min_z\{t^* + \frac{1}{2}\|z - y\|^2, \text{s.t. } z_l \leq \frac{t^*}{a_l}\}$ satisfies $\max_l\{z_l^{t^*} a_l\} = t^*$, and z^{t^*} is a solution of (20).

For any given t, if z^t is the solution of the inner minimization of (21):

$$z^t := \arg\min_z\{t + \frac{1}{2}\|z - y\|^2, \text{s.t. } z_l \leq \frac{t}{a_l}\}, \tag{22}$$

then z^t can be solved exactly:

$$(z^t)_l = \min\left(y_l, \frac{t}{a_l}\right). \tag{23}$$

For any given $y = (y_1, y_2, \ldots, y_k) \in \mathbb{R}^k$, we sort the components of a and y in the ascending order $a_{(1)}y_{(1)} \leq a_{(2)}y_{(2)} \leq \ldots \leq a_{(k)}y_{(k)}$. Let:

$$f(t) = \min_z\{t + \frac{1}{2}\|z - y\|^2, z_l \leq \frac{t}{a_l}\} = t + \frac{1}{2}\|z^t - y\|^2, \tag{24}$$

then:

$$f(t) = \begin{cases} t + \frac{1}{2}\sum_{l=1}^k \left|y_{(l)} - \frac{t}{a_{(l)}}\right|^2 & t \leq a_{(1)}y_{(1)}\\ t + \frac{1}{2}\sum_{l=i}^k \left|y_{(l)} - \frac{t}{a_{(l)}}\right|^2 & a_{(i-1)}y_{(i-1)} \leq t \leq a_{(i)}y_{(i)}\\ t & t \geq a_{(k)}y_{(k)}. \end{cases}$$

In [34], it is shown that f is a piecewise quadratic function and $f \in C^1(\mathbb{R})$. The derivative f' of f is:

$$f'(t) = \begin{cases} 1 + \sum_{l=1}^k \frac{1}{a_{(l)}}\left(\frac{t}{a_{(l)}} - y_{(l)}\right) & t \in I_1 = (-\infty, a_{(1)}y_{(1)}]\\ 1 + \sum_{l=i}^k \frac{1}{a_{(l)}}\left(\frac{t}{a_{(l)}} - y_{(l)}\right) & t \in I_i, 2 \leq i \leq k\\ 1 & t \in I_{k+1} = [a_{(k)}y_{(k)}, \infty), \end{cases}$$

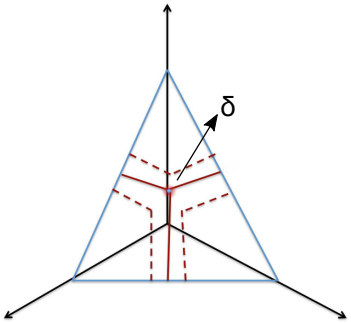

Fig. 2. Stable simplex clustering. Every grid point δ on the simplex generates a simplex clustering. We want to choose a δ such that there are very few data points falling into the "Y-shaped region".

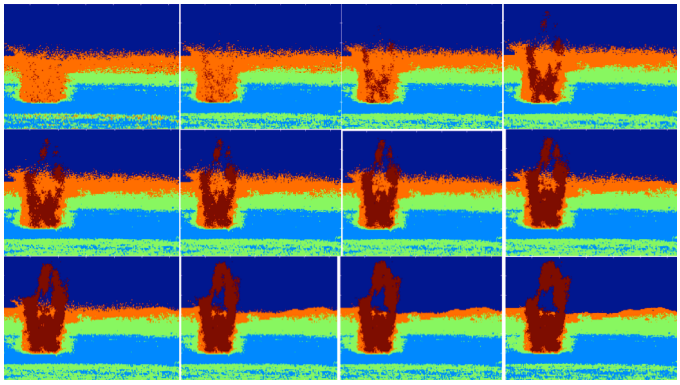

Fig. 3. Quadratic model and stable simplex clustering on the plume dataset. The chemical plume (brown) is perfectly detected in 12 iterations.

where $I_i = [a_{(i-1)}y_{(i-1)}, a_{(i)}y_{(i)}]$, for all $2 \leq i \leq k$. Therefore, the solution t^* of (21) satisfies $f'(t^*) = 0$, and it is the unique t_i among $\{t_1, t_2, \ldots, t_k\}$ that falls into the corresponding interval I_i, where $t_i = (\sum_{l=i}^{k} \frac{y_{(l)}}{a(l)} - 1)/(\sum_{l=i}^{k} \frac{1}{a_{(l)}^2})$. So t^* can be found in at most k steps. By Theorem 1, we know the solution z^* of (20) is $z^* = z^{t^*}$, i.e. $(z^*)_l = \min\,(y_l, \frac{t^*}{a_l})$. Therefore, the solution u^* of (19) is $u^* = A^{-1}(y - z^*)$.

4) Stable Simplex Clustering: As we have mentioned in the previous section, our quadratic model pushes anomalies into the middle of the unit simplex. Therefore it would be ill-conceived to simply classify the pixels based on the largest component of the labeling function $u(x) = (u_1(x), u_2(x), \ldots, u_k(x))$. Instead, we need a stable simplex clustering method.

The concept behind stable simplex clustering is to choose a division that puts all the data points in the "middle" into a single cluster. Fig. 1 demonstrates this in the simple two-cluster case. Also refer to section III-B1 for explanation of the "pushing" process.

Our idea to accomplish this goal comes from [5]. Mathematically speaking, we first create a grid on a k-dimensional simplex, where k is the number of clusters, and each grid point δ generates a simplex clustering. Then we find δ to minimize the energy $g(\delta)$:

$$g(\delta) = -\log(\prod_{l=1}^{k} F_l(\delta)) + \eta \exp(G(\delta)), \tag{25}$$

where $F_l(\delta)$ is the percentage of data points in cluster l, and $G(\delta)$ is the percentage of data points on the edges near the division, i.e. the "Y-shaped region" in Figure 2. The first term in $g(\delta)$ rewards keeping clusters approximately the same size, ensuring no skewed data from clusters far too small. And the second term rewards sparsity of points in the intermediate region. We choose the constant η large enough such that stability has a bigger weight in the energy.

The quadratic model combined with stable simplex clustering (see Algorithm 5) produced remarkable results; Figure 3 demonstrates how this detected the chemical plumes in a frame with background centroids pre-calculated and random initialization for the final centroid. Notice that no plume is detected in the first iteration. But by the twelfth iteration, the gas plume is nearly perfectly segmented.

Algorithm 5 Qinear Model with Stable Simplex Clustering

1: Initialization of centroids: Choose $(c_l)_{l=1}^k$ (randomized or generated by fast unsupervised centroid extraction algorithm).
2: Initialization of parameters: Choose $\tau, \sigma > 0$ satisfying $\sigma\tau\|\nabla_w\|^2 \leq 1$, $\theta = 1$
3: Initial iterate: Set $u^0 \in \mathbb{R}^{r\times k}$ and $p^0 \in \mathbb{R}^{(r\times r)\times k}$ randomly, set $\bar{u}^0 = u^0$,
4: **while** not converge **do**
5: N steps of primal-dual iterations for quadratic model
6: $u_{hard} = threshold(u)$ with stable simplex clustering
7: update $(c_l)_{l=1}^k$
8: **end while**

Finally, we present the results of the linear model and the quadratic model on the Urban dataset with identical random pixel initialization in Figure 4. The stopping criteria for the number of iterations was visual confirmation of all six clusters appearing. The linear model took about 50 iterations and 273 seconds to converge, and the quadratic model took 4 iterations and 75 seconds to converge.

IV. ANN Scheme For Weight Matrix Calculation

One of the largest time-sinks in our algorithm was the calculation of the sparse weight matrix. Using the straightforward local window approach described in Section III-A, for the Urban dataset on a Linux machine with Intel core i5, 3.3hz with 2GB of DDR3 ram, it took 111 seconds to compute the weight matrix. Speeding up this calculation would thus improve the usability of our algorithm.

When building the weight matrix, we determine the m nearest neighbors of a pixel by calculating the distances from all the pixels in a $w \times w$ local window ($m = 10$ and $w = 21$

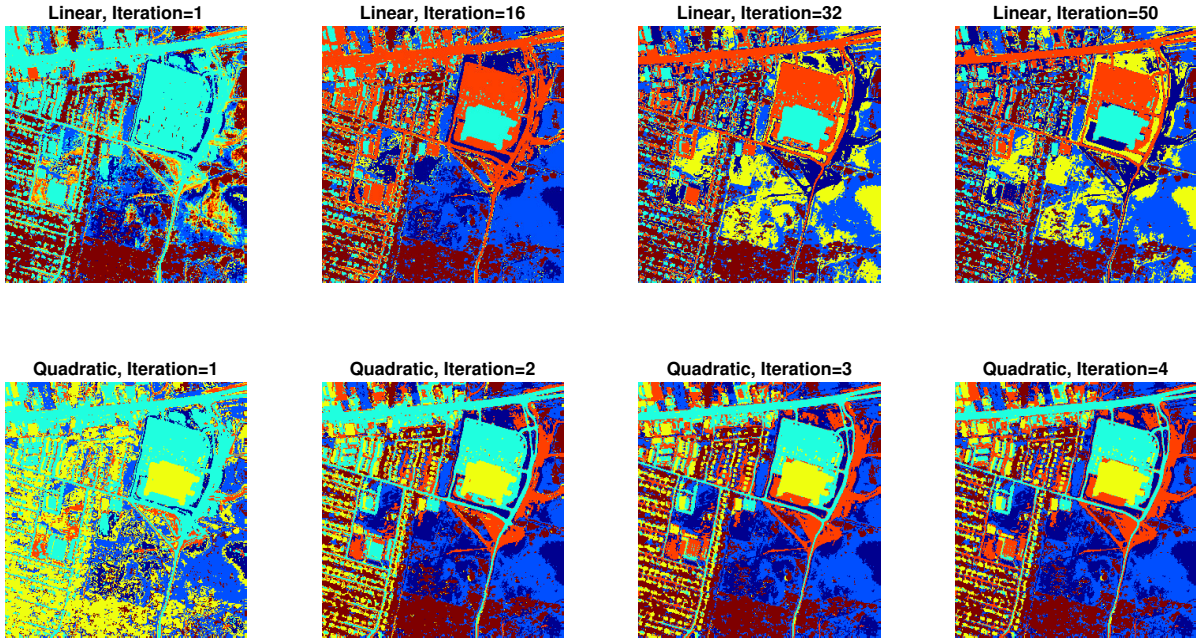

Fig. 4. Linear vs Quadratic Model on Urban dataset with the same centroid initialization. To produce essentially identical results, the Linear model (first row) took 50 iterations of centroid updates, and the Quadratic model (second row) took just 4 iterations.

in our experiments). In light of this, using an approximate nearest neighbor (ANN) search algorithm can be far faster than calculating the distance between all the pixels. We use a k-d tree approach as well as an ANN search algorithm to reduce the run-time.

A *k-d tree*, or k-dimensional tree, is a binary tree that recursively partitions a k-dimensional space [37] (k corresponds to the number of bands sampled in our case). The tree is organized as follows: the root node corresponds to the entire space, and each node in the tree corresponds to a partition in space. At each non-leaf node, we choose a dimension and a hyperplane perpendicular to that dimension's axis which divides the partition into two sub-partitions. Data points on one side of the hyperplane belong to the left sub-tree, and data points on the other side belong to the right sub-tree. Modern algorithms to build a balanced k-d tree generally at worst converge in $O(kn \log n)$ time, where k is the number of dimensions and n is the number of points [38].

The space partitioning scheme offered by the k-d tree significantly reduces the time cost of nearest neighbor search. Let us consider the case of 1-nearest-neighbor search for simplicity. Intuitively, given a query data point q, if we have found a candidate data point x that lies on one side of a hyperplane corresponding to a non-leaf node, and the hypersphere centered at q with radius $\|q - x\|_2$ does not intersect with the hyperplane, then all the data points lying on the other side of the hyperplane can be excluded from consideration. This way, branches can be eliminated from the search space quickly. This algorithm converges in $O(\log n)$ time [37]. We employ a randomized and approximate version of this algorithm [39] implemented in the open source VLFeat package `http://www.vlfeat.org`. We terminate the algorithm by placing an upper bound (say, 256) on the number of distance comparisons. It takes only 6 seconds to compute the sparse weight matrix for Urban dataset using ANN scheme as opposed to 111 seconds using the local search window.

The benefit of the ANN scheme is two-fold:

1) We compute the distances and similarity values only after the m (approximate) nearest neighbors are identified. Thus the cost of computing distances becomes $O(mnd)$ (excluding the cost of building and querying the k-d tree), as opposed to $O(w^2nd)$ in the local window approach.
2) Without the presence of local windows, the ANN scheme enables a "global search", that is, any pixel in the entire image could be a candidate of the m nearest neighbors.

Overall, this approximate way of nearest neighbor search and weight computation contributes to significantly reducing the run time of computing the weight matrix.

V. Numerical Results

A. Comparison Methods

We ran the following unsupervised algorithms on both synthetic and real-world hyperspectral datasets:

1) **(Spherical) K-means**: built in MatLab Code.
2) **NMF**: Non-negative Matrix Factorization, proposed in [40].

3) **H2NMF**: Hierarchical Non-negative Matrix Factorization, proposed in [5].
4) **MBO**: Graph Merriman-Bence-Osher scheme, proposed in [23]. The code runs 10 times on each dataset, and the best results is chosen.
5) **NLTV**: Nonlocal Total Variation, Cosine-Euclidean distance, Quadratic Model with random pixel initialization.
6) **NLTV, H2NMF/Kmeans init**: Nonlocal Total Variation, Cosine-Euclidean distance, with endmembers/centroids extracted from H2NMF/Kmeans. Linear Model unless otherwise specified as NLTV-2.

All experiments were run on Intel core i5, 3.3hz with 2GB of DDR3 ram.

B. Synthetic Dataset

We first test our algorithm on a synthetic dataset and compare it to other classical unsupervised classification methods. In the synthetic HSI, the five endmembers were randomly extracted from a real scene with 162 bands in ranges 400-2500 nm, and the 40000 abundance vectors were generated as a sum of Gaussian fields, with constraints so as to respect the abundance-nonnegative-constraint (ANC) and abundance-sum-to-one-constraint (ASC). The dataset was generated using a Generalized Bilinear Mixing Model (GBM):

$$y = \sum_{i=1}^{p} a_i e_i + \sum_{i=1}^{p-1} \sum_{j=i+1}^{p} \gamma_{ij} a_i a_j e_i \odot e_j + n, \qquad (26)$$

where γ_{ij} are chosen uniformly and randomly in the interval $[0, 1]$, n is the Gaussian noise, with an SNR of 30 dB, and a_i satisfies: $a_i \geq 0$ (ANC), and $\sum_{i=1}^{p} a_i = 1$ (ASC). The classification results are shown in Table I and Fig. 5. As we can see, both our NLTV algorithms (linear model with accurate initialization K-means, and quadratic model with less accurate initialization H2NMF) have much better overall accuracy than all of the other methods, although they take a longer time to converge. There are two things to mention. First, from Table I we can notice that the NLTV linear model can boost the accuracy of K-means from 90.98% to 97.44%, but the linear model does not perform well with the less accurate H2NMF initialization; however, the NLTV quadratic model can fix this problem (raising the accuracy from 72.02% to 99.03%). Second, we observe in our numerical experiments that the NLTV quadratic model is not extremely robust with respect to completely random initialization of the centroids. But this can be remedied by using a somewhat accurate initialization (such as H2NMF in this dataset), or if we use the initializing procedure like that in "K-means++" [41]. After all, NLTV performs with very high accuracy on this synthetic dataset with comparable speed to other unsupervised methods.

TABLE I
COMPARISON OF NUMERICAL RESULTS ON SYNTHETIC DATASET

Algorithm	Run-Time	Accuracy
K-means	2s	90.98%
NMF	9s	80.99%
H2NMF	2s	72.02%
MBO	21s	84.49%
NLTV-2 H2NMF init	51s	99.03%
NLTV Kmeans init	42s	97.44%

TABLE II
COMPARISON OF NUMERICAL RESULTS ON URBAN DATASET

Algorithm	Run-Time	Accuracy
K-means	7s	75.20%
NMF	87s	55.70%
H2NMF	7s	85.96%
MBO	92s	78.86%
NLTV	292s	92.25%
NLTV-2 H2NMF init	100s	92.10%
NLTV H2NMF init	47s	91.56%

C. Urban DataSet

The first real-world dataset we examined was the Urban dataset from HYperspectral Digital Imagery Collection Experiment (HYDICE), which has 307 x 307 pixels and contains 162 clean spectral bands. This dataset has the advantage of only having six classes of material: road, dirt, house, metal, tree, and grass. The RGB image of the urban dataset is shown in Fig 6.

There was no ground-truth provided for the data. We used a structured sparse algorithm [42] (which is different from all of the algorithms listed above) to initialize a ground truth division, then corrected areas pixel by pixel to provide a framework for numerical analysis of accuracy. As this "ground truth" was hand-corrected, it does not necessarily represent the most accurate segmentation of the image; however, it provides a basis for quantitative comparison. Values for λ and μ were 10^6 and 10^{-5} respectively; λ represents the weight on the fidelity term, and μ represents the balance of the cosine and Euclidean distance. As the magnitude of the squared cosine distance is generally very small, a small μ and a large λ are required to balance the scale.

It is worth mentioning that after running every algorithm that are compared to create six clusters, we noticed that all the algorithms split "grass" into two different clusters (one of them actually corresponds to a mixture of grass and dirt), while treating "road" and "metal" as the same. To obtain a reliable overall accuracy of the classification results, we combined the two "grass" clusters in every algorithm, hence obtaining the classification results for 5 clusters, which are "grass", "dirt", "road+metal", "roof", and "tree". The "ground truth" is modified in the same fashion.

The overall classification accuracies are displayed in Table II, as well as run-times. As can be seen, although our method took longer to run than the competing algorithms again, it performed consistently at higher accuracy. It is easier to see visually in Fig. 7 that the NLTV algorithm performs best of the five algorithms tested; specifically, the NLTV algorithm alone distinguished all of the dirt beneath the parking lot and the intricacies of the road around the parking lot. The total variation process also gives the segmented image smoother and more distinct edges, allowing for easier human identification

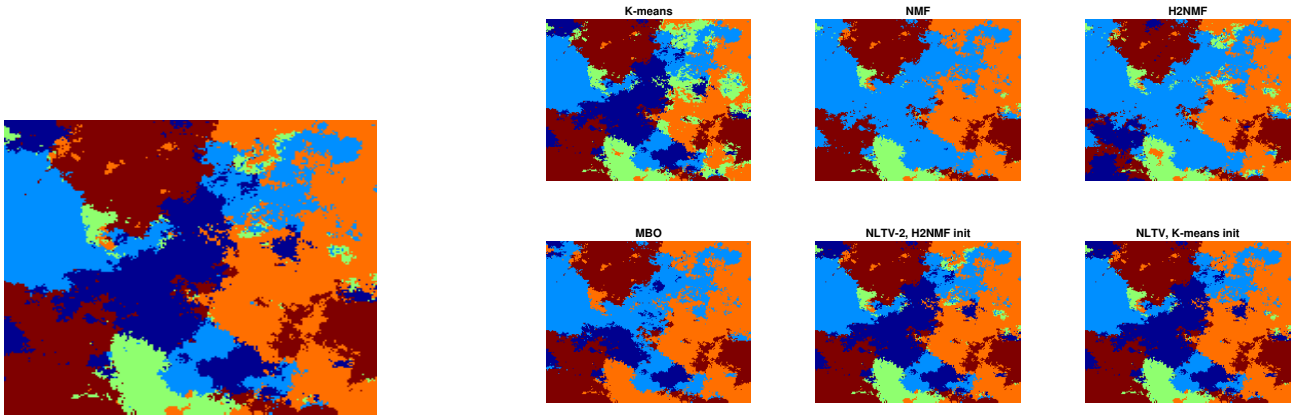

Fig. 5. Clustering results for the synthetic dataset generated by 5 endmembers. The large image on the left is the ground truth, and the six smaller images on the right are the clustering results of the corresponding algorithms.

Fig. 6. RGB Image of the Urban dataset

TABLE III
RUN-TIMES FOR SAN DIEGO AIRPORT AND PLUME DATASETS

Dataset	San Diego Airport	Plume
K-means	9s	2s
NMF	4s	2s
H2NMF	13s	2s
MBO	329s	18s
NLTV	92s	69s
NLTV H2NMFinit	68s	54s

of the clusters.

D. San Diego Airport Dataset

We examined the San Diego Airport dataset, provided by the HYDICE sensor, which is 400 x 400 pixels and contains 158 clean spectral bands. There are seven types of materials: trees, grass, three types of road surfaces (boarding and landing zones, parking lots, and streets), and two types of rooftops [5]. The RGB image with cluster labels are shown in Fig. 8.

After examining the spectral signatures of various pixels in the scene, we managed to pinpoint some errors that were common for each algorithm. We will not go into detail about the NMF and H2NMF algorithms, which clearly do not perform well on this dataset. K-means obtains some decent results, but splits the rooftops of the four buildings on the bottom right of the image into two distinct clusters, and fails to separate two different buildings at the top. The MBO scheme fails on two accounts: it does not properly segment two different road surfaces (cluster 6 and 7), and does not account for the different rooftop types (cluster 3 and 4). Our linear model with H2NMF initialization is significantly more accurate than H2NMF and MBO. It successfully picks out two different types of roof (cluster 3, light blue and cluster 4, yellow), two different types of road (cluster 6, medium blue and 7, green), although the other type of road (cluster 5, dark blue) is mixed with one type of roof (cluster 3, light blue). The best result was obtained by using our quadratic model with random initialization, with the only problem that tree and grass (clusters 1 and 2, red) are mixed up. However, this mixing of grass and tree is actually the case for all the other algorithms. This means that our algorithm alone was able to identify six of the seven clusters correctly.

E. Chemical Plume Dataset

Next we examined the chemical plume dataset, which consists of frames taken from a hyperspectral video of the release of chemical plumes provided by the John Hopkins University Applied Physics Laboratory. These images were taken by long wave infrared spectrometers placed 2km from the release of the plume at an elevation of approximately 1300 feet. The image is 128 x 320 pixels, with 129 clean spectral bands. There was no ground truth provided for this data, so we assumed segmentation into four classes: chemical plume, sky,

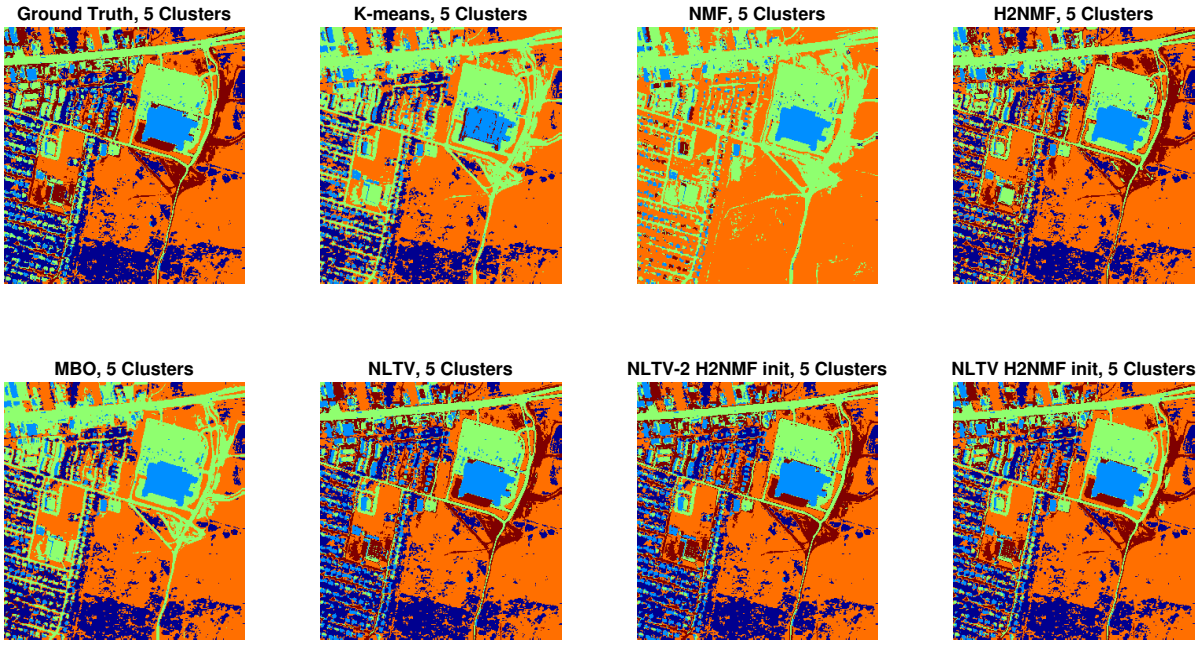

Fig. 7. Clustering results for Urban dataset. Five clusters including rooftops, grass, trees, dirt, and "road+metal" are generated by the algorithms.

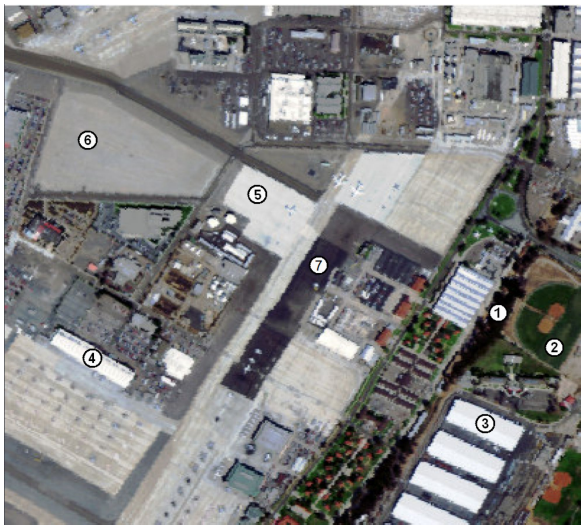

Fig. 8. RGB Image of San Diego Airport

foreground, and mountain. A fifth cluster is added so that the noise pixels would not interfere with the segmentation [22].

Analyzing images for chemical plumes is a far more difficult problem than merely segmenting images, because the gas in the plumes is generally diffuse and semi-transparent and thus very difficult to detect. Chemical plume detection generally faces the challenge of a high presence of noise in the picture that requires intensive preprocessing before segmentation schemes can be implemented. We ran all the algorithms on the image before it was denoised and the results are shown in Figure 10. Parameters for NLTV were λ and μ were 10^9 and 0 respectively. And due to the diffusion of the plume, we kept 20 nearest neighbor for every pixel, instead of 10 in the previous cases.

We figured out that unmixing methods such as NMF and H2NMF do not perform very well on this dataset. MBO, spherical K-means, NLTV can all properly identify the chemical plume. Notice that NLTV with H2NMF as centroid initialization can again raise the accuracy of the initializing algorithm. We have to point out that the result of our quadratic model with completely randomized centroid is again not very robust on this dataset, but we can still use the initializing procedure in "K-means++" [41] to overcome this problem. Also, as long as at least one random centroid is chosen from the "plume", the entire plume cluster can be detected by quadratic NLTV.

VI. Conclusion

In this paper we present the framework for a nonlocal total variation method for unsupervised HSI classification, which we solve with the primal-dual hybrid gradient algorithm. We develop a linear and a quadratic version of this model; the linear version updates more quickly and can refine results produced by an centroid extraction algorithm like hierarchical non-negative matrix factorization, and the quadratic model provides a robust means of classifying hyperspectral images with randomized pixel initialization. To reduce computational time, we introduce a stable simplex clustering scheme, and utilize a k-d tree and approximate nearest neighbor search to compute the weight matrix.

We test this algorithm on both synthetic and three real-world datasets, with very promising results. Our algorithm

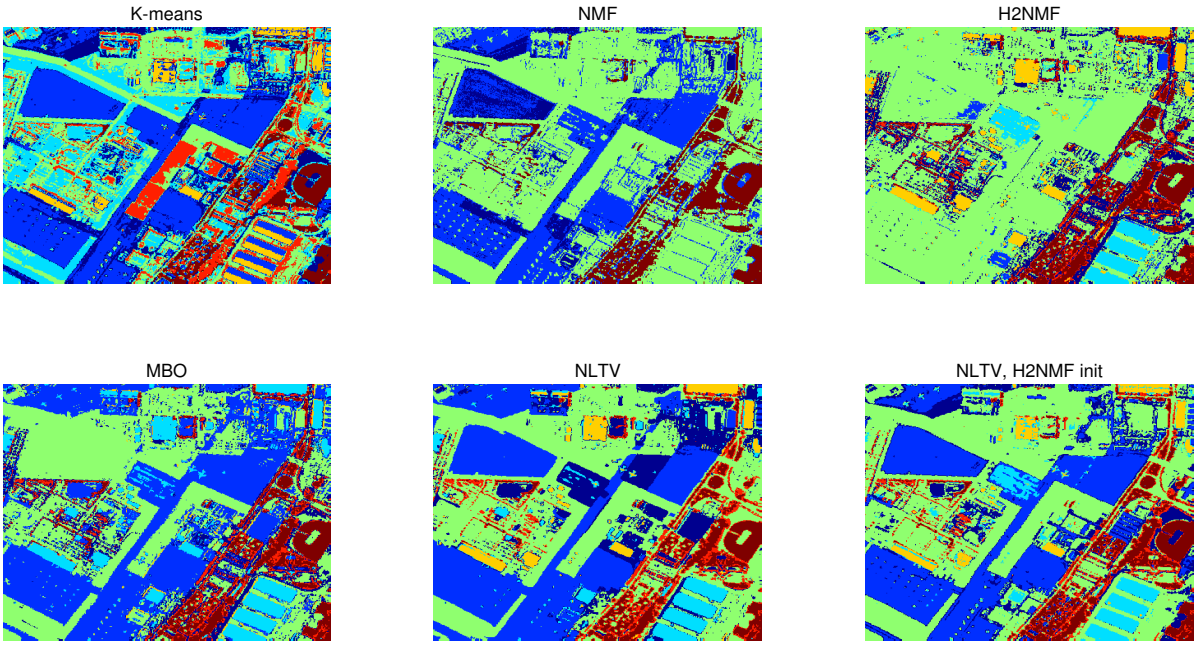

Fig. 9. Clustering results for San Diego Airport dataset. Seven clusters including trees, grass, three types of road surfaces, and two types of rooftops are generated by the algorithms. Only NLTV quadratic model can accurately detect 6 out of 7 clusters.

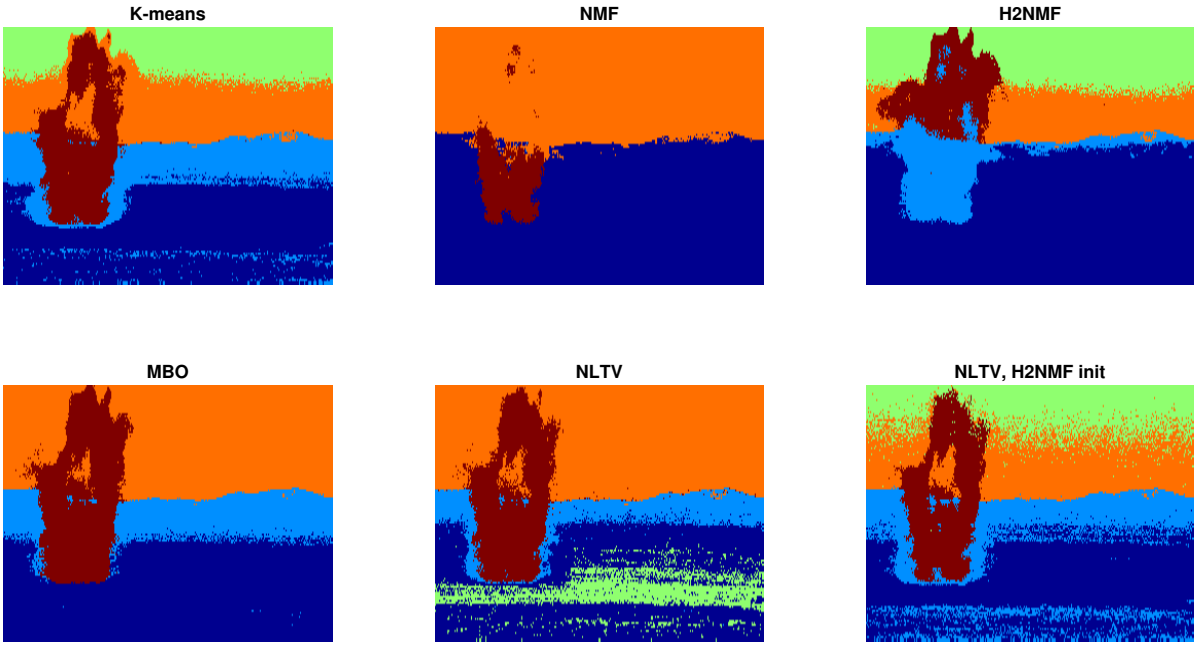

Fig. 10. Clustering results for Chemical Plume dataset. Number of clusters k=5.

consistently performed with highest accuracy on synthetic and urbanized datasets (Urban, San Diego Airport), both producing smoother results with easier visual identification of segmentation, and distinguishing classes of material that other algorithms were not able to differentiate. Our algorithm also performed well on anomaly detection scenarios like the

Chemical Plume datasets; with proper initialization on the plume dataset, it performed on par with the Merriman-Bence-Osher scheme developed specifically for this dataset. While the run-times were not yet idealized, they were still comparable to the other methods, and the consistent higher accuracy on different types of datasets suggests that this technique is a more robust and precise means of classifying hyperspectral images.

ACKNOWLEDGMENT

The authors would like to thank Zhaoyi Meng and Justin Sunu, for providing and helping with the MBO code.

REFERENCES

[1] C.-I. Chang, *Hyperspectral imaging: techniques for spectral detection and classification*. Springer Science & Business Media, 2003, vol. 1.

[2] J. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. 5, no. 2, pp. 354–379, 2012.

[3] N. Gillis and S. Vavasis, "Fast and robust recursive algorithmsfor separable nonnegative matrix factorization," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 4, pp. 698–714, 2014.

[4] S. Jia and Y. Qian, "Constrained nonnegative matrix factorization for hyperspectral unmixing," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 47, no. 1, pp. 161–173, 2009.

[5] N. Gillis, D. Kuang, and H. Park, "Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 53, no. 4, pp. 2066–2078, 2015.

[6] N. Dobigeon, J.-Y. Tourneret, C. Richard, J. Bermudez, S. McLaughlin, and A. Hero, "Nonlinear unmixing of hyperspectral images: Models and algorithms," *Signal Processing Magazine, IEEE*, vol. 31, no. 1, pp. 82–94, 2014.

[7] R. Heylen, M. Parente, and P. Gader, "A review of nonlinear hyperspectral unmixing methods," *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. 7, no. 6, pp. 1844–1868, 2014.

[8] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 42, no. 8, pp. 1778–1790, 2004.

[9] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 43, no. 6, pp. 1351–1362, 2005.

[10] M. Fauvel, J. Benediktsson, J. Chanussot, and J. Sveinsson, "Spectral and spatial classification of hyperspectral data using svms and morphological profiles," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 46, no. 11, pp. 3804–3814, 2008.

[11] B. Demir and S. Erturk, "Hyperspectral image classification using relevance vector machines," *Geoscience and Remote Sensing Letters, IEEE*, vol. 4, no. 4, pp. 586–590, 2007.

[12] G. M. Foody, "Rvmbased multiclass classification of remotely sensed data," *International Journal of Remote Sensing*, vol. 29, no. 6, pp. 1817–1823, 2008.

[13] F. Mianji and Y. Zhang, "Robust hyperspectral classification using relevance vector machine," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 49, no. 6, pp. 2100–2112, 2011.

[14] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, Aug 2000.

[15] M. Stoer and F. Wagner, "A simple min-cut algorithm," *J. ACM*, vol. 44, no. 4, pp. 585–591, Jul. 1997. [Online]. Available: http://doi.acm.org/10.1145/263867.263872

[16] A. Szlam and X. Bresson, "A Total Variation-based Graph Clustering Algorithm for Cheeger Ratio Cuts," Tech. Rep. UCLA CAM Report 09-68, 2009.

[17] X. Bresson and A. D. Szlam, "Total variation, cheeger cuts," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 1039–1046.

[18] A. L. Bertozzi and A. Flenner, "Diffuse interface models on graphs for classification of high dimensional data," *Multiscale Modeling & Simulation*, vol. 10, no. 3, pp. 1090–1118, 2012.

[19] B. Merriman, J. K. Bence, and S. J. Osher, "Motion of multiple junctions: A level set approach," *Journal of Computational Physics*, vol. 112, no. 2, pp. 334 – 363, 1994.

[20] C. Garcia-Cardona, E. Merkurjev, A. Bertozzi, A. Flenner, and A. Percus, "Multiclass data segmentation using diffuse interface methods on graphs," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 8, pp. 1600–1613, 2014.

[21] "Diffuse interface methods for multiclass segmentation of high-dimensional data," *Applied Mathematics Letters*, vol. 33, pp. 29 – 34, 2014.

[22] H. Hu, J. Sunu, and A. L. Bertozzi, *Energy Minimization Methods in Computer Vision and Pattern Recognition: 10th International Conference, EMMCVPR 2015, Hong Kong, China, January 13-16, 2015. Proceedings*. Cham: Springer International Publishing, 2015, ch. Multiclass Graph Mumford-Shah Model for Plume Detection Using the MBO scheme, pp. 209–222.

[23] E. Merkurjev, J. Sunu, and A. Bertozzi, "Graph MBO method for multiclass segmentation of hyperspectral stand-off detection video," in *Image Processing (ICIP), 2014 IEEE International Conference on*, 2014, pp. 689–693.

[24] "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 14, pp. 259 – 268, 1992.

[25] T. Chan, S. Esedoglu, F. Park, and A. Yip, "Recent developments in total variation image restoration," *Mathematical Models of Computer Vision*, vol. 17, 2005.

[26] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[27] G. Gilboa and S. Osher, "Nonlocal operators with applications to image processing," *Multiscale Modeling & Simulation*, vol. 7, no. 3, pp. 1005–1028, 2009.

[28] D. Zosso, G. Tran, and S. J. Osher, "Non-local retinex—a unifying framework and beyond," *SIAM Journal on Imaging Sciences*, vol. 8, no. 2, pp. 787–826, 2015.

[29] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems withapplications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2010.

[30] M. Zhu and T. Chan, "An efficient primal-dual hybrid gradient algorithm for total variation image restoration," Tech. Rep. UCLA CAM Report 08-34, 2008.

[31] M. Zhu, S. J. Wright, and T. F. Chan, "Duality-based algorithms fortotal-variation-regularized image restoration," *Computational Optimization and Applications*, vol. 47, no. 3, pp. 377–400, 2008.

[32] E. Esser, X. Zhang, and T. F. Chan, "A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science," *SIAM Journal on Imaging Sciences*, vol. 3, no. 4, pp. 1015–1046, 2010.

[33] J. Zhang, W. Zhu, L. Wang, and N. Jiang, "Evaluation of similarity measure methods for hyperspectral remote sensing data," in *Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International*, 2012, pp. 4138–4141.

[34] Y. Chen and X. Ye, "Projection onto a simplex," *arXiv preprint arXiv:1101.6081*, 2011.

[35] D. MacKay, "An example inference task: Clustering," in *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003, ch. 20, pp. 284–292.

[36] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems," *Communications on pure and applied mathematics*, vol. 42, no. 5, pp. 577–685, 1989.

[37] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209–226, Sep. 1977.

[38] R. A. Brown, "Building a balanced kd tree in o (kn log n) time," *arXiv preprint arXiv:1410.5420*, 2014.

[39] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration." *VISAPP (1)*, vol. 2, pp. 331–340, 2009.

[40] J. Kim and H. Park, "Fast nonnegative matrix factorization: An active-set-like method and comparisons," *SIAM Journal on Scientific Computing*, vol. 33, no. 6, pp. 3261–3281, 2011.

[41] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '07. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007,

pp. 1027–1035. [Online]. Available: http://dl.acm.org/citation.cfm?id=1283383.1283494

[42] "Structured sparse method for hyperspectral unmixing," {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, vol. 88, pp. 101 – 118, 2014.