

Adaptive Superpixel Cuts for Hyperspectral Images

Aleksandar Popovic

February 2024

Abstract

Blind segmentation in hyperspectral images is a challenging problem. Many traditional methods suffer from poor identification of materials and expensive computational costs, which can be partially eased by trading the accuracy with efficiency.

In this paper, we propose a novel graph-based algorithm for segmentation in hyperspectral images. Utilizing the fact that pixels in a local region are likely to have similar spectral features, a pre-clustering algorithm is used to extract the homogenous regions, called superpixels. After extracting the superpixels, a weighted graph is constructed with the weights representing both the spectral similarity and spatial distance between each superpixel and its neighbors. The normalized graph cuts algorithm is then used to perform an initial segmentation of the image. To effectively extract the material information in the superpixels, the mean spectra in each segment is used to estimate the abundance of each endmember in each superpixel using a graph regularized hyperspectral unmixing algorithm. The resulting abundance information is used as a supportive feature, which when combined with the spectral features, form a new spectral feature vector for each superpixel. Using this new feature vector, the weighted graph is once again constructed and the normalized cuts algorithm is applied, resulting in a final segmentation of the image.

Experiments on a real hyperspectral datasets illustrate great potential of the proposed method in terms of accuracy and efficiency.

Contents

1	Introduction	3
1.1	Motivations	3
1.2	Applications to Biomedical Imaging	3
1.3	Hyperspectral Data	3
2	Background	4
2.1	The Hyperspectral Cube	4
2.2	Supapixel Generation	4
2.2.1	Simple Linear Iterative Clustering	4
2.3	Hyperspectral Unmixing	5
2.3.1	Linear Mixing Model	5
2.3.2	Abundance Estimation	6
2.4	Alternating Direction Method of Multipliers	7
2.5	Spectral Clustering	7
2.5.1	Normalized Cuts	8
3	Adaptive Supapixel Cuts	9
3.1	Dataset Preprocessing	10
3.1.1	Singular Value Decomposition	10
3.1.2	Layer Normalization	10
3.2	Hyperspectral Supapixel Generation	11
3.3	Spatial Spectral Clustering	12
3.4	Graph Regularized Abundance Estimation	12
3.5	Feature Vector Creation	12
3.6	Compilation	12
3.7	Parameter Selection	12
4	Experimental Results	13
4.1	Implementation Details	13
4.2	Geospatial Testing Datasets	14
4.2.1	Samson Datasets	14
4.2.2	Salinas Datasets	14
4.2.3	Biomedical Autofluorescence Dataset	14
4.3	Algorithm Evaluation	15
4.3.1	Qualitative Evaluation on Samson	15
4.3.2	Quantitative Evaluation on Salinas	15
4.3.3	Qualitative Evaluation on Biomedical Autofluorescence Data	15
4.4	Algorithm Comparison	16
5	Conclusions	17

1 Introduction

1.1 Motivations

1.2 Applications to Biomedical Imaging

1.3 Hyperspectral Data

2 Background

Analysis in hyperspectral images has traditionally been a computationally expensive and difficult task due to algorithms scaling in both the spatial and spectral resolution of the images. This section will focus on building a relevant background for common preclustering, abundance estimation, and segmentation techniques from the perspective of imaging that will be later be adapted for use in the proposed algorithm.

2.1 The Hyperspectral Cube

In traditional, RGB based imaging systems, an image can be represented by a 3-dimensional tensor of shape $(n_x, n_y, 3)$, where the last dimension corresponds to the color channel the image was captured in. From a mathematical point of view, a hyperspectral image, denoted by \mathbf{X} , is a tensor of shape (n_x, n_y, n_λ) with nonnegative entries. Each pixel in the tensor is represented using a vector $\mathbf{x} \in \mathbb{R}_+^{n_\lambda}$. From a physical point of view, the first two dimensions in \mathbf{X} represent the spatial coordinates of the pixels, while the last dimension represents the specific wavelength band the spectral intensity, reflectance or transmittance measurements were taken at.

2.2 Superpixel Generation

As mentioned in Section (2.1), high spectral resolution is a common constraint in hyperspectral image analysis, with time complexity of the algorithms commonly scaling polynomially with the number of pixels in the image.

Before the introduction of neural network based solutions in computer vision applications, there was interest in preclustering images into locally homogeneous regions called superpixels [superpixel ref]. Addressing the main motivation of reducing the overall granularity of the data, superpixels are also shown to preserve spectral information, adhere to spatial features in the image, and introduce robustness against noise in subsequent analysis tasks.

2.2.1 Simple Linear Iterative Clustering

In this section, we will introduce the Simple Linear Iterative Clustering (SLIC) algorithm. The algorithm is a special case of the k-means algorithm adapted to the task generating superpixels in a 5-dimensional space, where the first 3 dimensions correspond to the pixel color vector in the CIELAB color space, and last 2 dimensions correspond to the spatial coordinates (i, j) of the pixel in the image.. Formally, we restructure each pixel $\mathbf{x}_{(i,j)} = [\mathbf{x}_l, \mathbf{x}_a, \mathbf{x}_b]$ into the form $\tilde{\mathbf{x}}_{(i,j)} = [\mathbf{x}_l, \mathbf{x}_a, \mathbf{x}_b, i, j]$. With this modified feature vector $\tilde{\mathbf{x}}$, we incorporate both spectral and spatial information into the clustering, however, while the spectral information has bounds on it's values, the spatial information depends on the size of the image.

Taking as an input the desired number of superpixels n_s , for an image with $n_p = n_x n_y$ pixels, each superpixel would be composed of approximately n_s/n_p pixels. Assuming the superpixels lie on a grid, a superpixel centroid would occur at every grid interval $S = \sqrt{n_s/n_p}$. At the onset of the algorithm, a grid of n_s superpixel centers $\mathbf{C}_n = [\mathbf{c}_l, \mathbf{c}_a, \mathbf{c}_b, i, j]$ where $n = 1, \dots, n_s$ are sampled across the image with regular grid intervals S . To avoid sampling noisy pixels, clusters are moved to the lowest gradient position in a 3×3 neighborhood where the image gradient is calculated, using the original spectral vector x in the CIELAB color space as:

$$\mathbb{G}(i, j) = \|\mathbf{x}_{(i+1,j)} - \mathbf{x}_{(i-1,j)}\|^2 + \|\mathbf{x}_{(i,j+1)} - \mathbf{x}_{(i,j-1)}\|^2 \quad (1)$$

After initialization, a modified distance measure is proposed to enforce color similarity and spatial extent within the superpixels. Since the approximate area of each superpixel is S^2 , it is assumed that pixels associated with a superpixel lie within a $2S \times 2S$ neighborhood of the superpixel centroid. Introducing the parameter m to control the compactness and shape of the superpixels, the modified distance is then calculated as

$$\mathbb{D}(x, y) = \|\mathbf{x}_{lab} - \mathbf{y}_{lab}\|^2 + \frac{m}{S} \|\mathbf{x}_{ij} - \mathbf{y}_{ij}\|^2 \quad (2)$$

Each pixel in the image is associated with the nearest cluster whose search area overlaps this pixel. After all pixels are associated with a cluster, a new center is computed as the average feature vector of all the pixels

belonging to the cluster. This is repeated for a set number of iterations. After exhausting all iterations, a final step is performed by relabelling disjoint segments with the labels of the largest neighboring cluster. This step is optional as disjoint segments tend to not occur for larger inputs of n_s and m .

Algorithm 1: SLIC Superpixel Algorithm

Input: \mathbf{X}_f , $m > 0$, $n_s > 0$, $n_{\text{iters}} > 0$

Initialize: $\mathbf{C}_n = [\mathbf{c}_l, \mathbf{c}_a, \mathbf{c}_b, i, j]$ where $n = 1, \dots, n_s$ by sampling pixels at regular grid intervals S .

Perturb cluster centers to lowest gradient position in a 3×3 neighborhood according to (1)

for $k = 1$ **to** n_{iters} **do**

 Assign best matching pixels from a $2S \times 2S$ neighborhood around clusters C_k according to distance measure (2). Compute new cluster centers according to average of all pixels belonging to cluster.

end

Optional: Relabel disjoint segments.

The SLIC algorithm is shown to produce meaningful and noise-robust segments in traditional computer vision applications. This algorithm proves useful in Section 3.2 when adapted as a pre-clustering step in the hyperspectral domain.

2.3 Hyperspectral Unmixing

In hyperspectral imaging applications, there is emphasis on estimating the relative abundance of a given representative material, called an endmember, within each pixel.

Unmixing results often give more detailed information about the overall composition of a hyperspectral scene with respect to simple segmentation. This section will introduce the foundational knowledge behind unmixing and abundance estimation and derive an ADMM based approach to abundance estimation.

2.3.1 Linear Mixing Model

In reality, most pixels in a hyperspectral image capture a mixture of spectra reflected from various materials present within the spatial area, due to constraints with how large a spatial resolution can be achieved.

Figure 1: Image of LMM goes here!

The foundational model behind hyperspectral unmixing is the linear mixing model, which dictates that spectra of every pixel $\mathbf{x} \in \mathbb{R}_+^{n_b}$ in a hyperspectral image is a linear combination of a set of n_e spectra, $\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_{n_e} \in \mathbb{R}_+^{n_b}$, from pure representative materials, called endmembers, with weights $a_1, a_2, \dots, a_{n_e} \in \mathbb{R}$. Denoting $\mathbf{M} = [\mathbf{m}_1 \mid \mathbf{m}_2 \mid \dots \mid \mathbf{m}_{n_e}] \in \mathbb{R}_+^{n_b \times n_e}$ and $\mathbf{a} = [a_1, a_2, \dots, a_{n_e}]^T \in \mathbb{R}^{n_e}$, the linear mixing model is formulated as follows:

$$\mathbf{x} = \mathbf{M}\mathbf{a} + \epsilon. \quad (3)$$

While this model serves useful, there is no direct physical interpretation to the weights in \mathbf{a} , instead, we aim to estimate the physical proportion, called the abundance, of each endmember within each pixel by imposing two constraints on the entries in \mathbf{a} . The abundance nonnegativity constraint (ANC) requires that the entries in \mathbf{a} must be greater than or equal to zero, while the abundance sum-to-one constraint (ASC) requires that the entries in \mathbf{a} sum to 1. Combining the two constraints, we have an extension of the linear mixing model

$$\mathbf{x} = \mathbf{M}\mathbf{a} + \epsilon \quad \text{s.t } \mathbf{a} \in \mathbb{R}_+^{n_e} \text{ and } \|\mathbf{a}\|_1 = 1. \quad (4)$$

The linear mixing model can be additionally be extended from a per pixel basis onto a collection of n_p pixels $\mathbf{X} = [\mathbf{x}_1 \mid \mathbf{x}_2 \mid \dots \mid \mathbf{x}_{n_p}] \in \mathbb{R}_+^{n_b \times n_p}$, with each pixel \mathbf{x}_i having a corresponding abundance vector \mathbf{a}_i . Arranging the abundance vectors into an abundance matrix $\mathbf{A} = [\mathbf{a}_1 \mid \mathbf{a}_2 \mid \dots \mid \mathbf{a}_{n_p}] \in \mathbb{R}^{n_e \times n_p}$, we denote the ANC-ASC constraint using the set $\Delta = \{\mathbf{A} \in \mathbb{R}_+^{n_e \times n_p} \mid \mathbf{1}_{n_e}^T \mathbf{A} = \mathbf{1}_{n_p}\}$. This new extension of the linear mixing model to a collection of pixels that will be used for the following sections

$$\mathbf{X} = \mathbf{M}\mathbf{A} + \epsilon \quad \text{s.t. } \mathbf{A} \in \Delta \quad (5)$$

The linear mixing model is an objectively simple model, which enforces a linear relationship between the spatial mixing of endmembers through assuming that pixels lie on a flat plane. This is almost never the case, however the model remains a efficient and powerful tool for extracting spectral information from a scene.

2.3.2 Abundance Estimation

Given the linear mixing model as formulated in Section 2.3.1, in traditional hyperspectral imaging tasks, both \mathbf{M} and \mathbf{A} are unknown. Often, researchers aim to estimate \mathbf{M} first, as spectral signatures collected from endmembers in same scene under the same conditions will be almost identical. Notably, in the field of remote sensing, effort has been made to create a library of spectral signatures derived from common vegetation and minerals in land cover images, allowing focus to be made solely in estimating \mathbf{A} [REF]. This section will cover the scenario where \mathbf{M} is known and \mathbf{A} is to be estimated.

The task is referred to as abundance estimation and continues to be an active area of research, where the aim is to find \mathbf{A} such that an error function \mathcal{L} is minimized with respect to the reconstructed collections of pixels $\tilde{\mathbf{X}} = \mathbf{M}\mathbf{A}$ and the original collection of pixels \mathbf{X} . Traditionally, we aim to minimize the least-square reconstruction error between the entries in $\tilde{\mathbf{X}}$ and \mathbf{X}

$$\mathcal{L}(\mathbf{X}, \tilde{\mathbf{X}}) = \sum_{i=1}^{n_b} \sum_{j=1}^{n_p} (\mathbf{x}_{(i,j)} - \tilde{\mathbf{x}}_{(i,j)})^2 = \|\tilde{\mathbf{X}} - \mathbf{X}\|_F^2. \quad (6)$$

The least-squares reconstruction error can alternatively be written as the squared Frobenius norm, denoted as $\|\cdot\|_F^2$, of the difference between $\tilde{\mathbf{X}}$ and \mathbf{X} . This choice of \mathcal{L} is the straightforward approach as \mathcal{L} is both convex and differentiable, with the additional properties that $\mathcal{L}(\tilde{\mathbf{X}}, \mathbf{X}) = \mathcal{L}(\mathbf{X}, \tilde{\mathbf{X}})$ and $\mathcal{L}(\mathbf{X}, \tilde{\mathbf{X}}) = \mathcal{L}(\mathbf{X}^T, \tilde{\mathbf{X}}^T)$ [REF].

To incorporate the ANC-ASC constraint into the overall formulation of \mathcal{L} , the set Δ from Section 2.3.1 proves useful. It is important to note that Δ is a convex set, meaning that for matrices $A, B \in \Delta$, for all $0 \leq \alpha \leq 1$, the matrix $C = \alpha A + (1 - \alpha)B$ is also an element of Δ . The inclusion of the constraints on \mathbf{A} is facilitated using the piecewise function χ_S defined as follows

$$\chi_S(x) = \begin{cases} 0 & \text{if } x \in S \\ \infty & \text{if } x \notin S. \end{cases} \quad (7)$$

Adding χ_Δ in the formulation of \mathcal{L} restricts the values \mathbf{A} can take on to the set Δ while ensuring that the overall formulation still has a global minimum within Δ . Additionally, a regularization term J can be added to impose other constraints on the values in \mathbf{A} . Later sections operate on the assumption that J is also convex. Formally, abundance estimation can be formulated as a convex optimization problem of the form

$$\mathbf{A} = \arg \min_{\mathbf{A} \in \mathbb{R}^{n_e \times n_p}} \frac{1}{2} \|\mathbf{M}\mathbf{A} - \mathbf{X}\|_F^2 + \chi_\Delta(\mathbf{A}) + J(\mathbf{A}). \quad (8)$$

This problem has no closed form solution, relying on iterative methods or applying solvers like GUROBI [REF] or CVXOPT [REF] to solve the problem for individual pixels given the problem can be split pixelwise. The formulation of the abundance estimation allows for flexibility in choice of \mathcal{L} . There is particular interest in choosing \mathcal{L} such that it is convex and differentiable, in order to apply gradient-based methods to estimate the abundance matrix, however there exist approaches that choose rely on divergence-based or non differentiable metrics [REF].

2.4 Alternating Direction Method of Multipliers

Alternating Direction Method of Multipliers, or ADMM, introduced by Boyd et al. [REF] is a framework for solving convex optimization problems of the form:

$$\begin{aligned} & \text{minimize} && f(x) + g(z) \\ & \text{subject to} && Ax + Bz = c \end{aligned} \quad (9)$$

with variables $x \in \mathbb{R}^n$ and $z \in \mathbb{R}^m$, where $A \in \mathbb{R}^{p \times n}$, $B \in \mathbb{R}^{p \times m}$ and $c \in \mathbb{R}^p$. f and g are assumed to be convex. The aim of ADMM is to incorporate the decomposability of the dual ascent method into the superior convergence properties of method of multipliers. To allow for this, ADMM introduces the corresponding augmented Lagrangian \mathcal{L}_μ defined as:

$$\mathcal{L}_\mu(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) + \frac{\mu}{2}\|Ax + Bz - c\|_2^2 \quad (10)$$

where $\mu > 0$ is augmented lagrangian convergence parameter and $y \in \mathbb{R}^p$ is the corresponding dual variable. Scaling with $u = \frac{1}{\mu}y$ gives the following equivalent definition:

$$\mathcal{L}_\mu(x, z, u) = f(x) + g(z) + \frac{\mu}{2}\|Ax + Bz - c + u\|_2^2. \quad (11)$$

ADMM aims to minimize scaled form of \mathcal{L}_μ by alternating minimizations with respect to x , y , and u by performing the following updates:

$$\begin{aligned} x^{(k+1)} &= \arg \min_x \mathcal{L}_\mu(x, z^{(k)}, u^{(k)}) \\ z^{(k+1)} &= \arg \min_z \mathcal{L}_\mu(x^{(k+1)}, z, u^{(k)}) \\ u^{(k+1)} &= u^{(k)} + Ax^{(k+1)} + Bz^{(k+1)} - c. \end{aligned} \quad (12)$$

Under mild conditions on f and g , ADMM can be shown to provide guaranteed objective and residual convergence, independent on choice of μ . For lax choices of μ , the algorithm provides modest accuracy solutions in a relatively low number of iterations, favorable to tasks in statistical learning where parameter estimation often yields little improvement to results. The algorithm allows practioners to put focus on efficient implementations to the minimization problems for x and z , which proves useful in the following section.

2.5 Spectral Clustering

Clustering aims to partition unlabelled data into a set of groupings called clusters such that a predefined similarity metric is minimized within the data points in the cluster and maximized between clusters. Traditional clustering methods such as k-means, and tree based methods suffer in situations where both spatial and spectral information must be taken into account for computing clusters and often are sensitive to initialization and outliers in data. The focus of this section is to introduce the concept of spectral clustering, which aims to partition a set of data points into cluster by storing similarity between data points in a graph structure then using spectral analysis techniques to calculate globally optimal partitions.

The particular focus will be in the context of imaging, and particularly hyperspectral imaging, the final product of a clustering algorithm should be perceptually meaningful groupings that respect both the spectral and spatial features in the image. Considering a collection of pixels $\mathbf{X} = [\mathbf{x}_1 \mid \mathbf{x}_2 \mid \cdots \mid \mathbf{x}_{n_p}] \in \mathbb{R}_+^{n_b \times n_p}$ and a symmetric similarity measure d , the affinity matrix $\mathbf{W} \in \mathbb{R}_+^{n_p \times n_p}$ is constructed as

$$\mathbf{W}_{(i,j)} = d(\mathbf{x}_i, \mathbf{x}_j). \quad (13)$$

Typical choices for d in imaging applications include calculating the euclidean norm and the cosine angle between the spectral features of \mathbf{x}_i and \mathbf{x}_j . The euclidean distance calculates the difference in magnitude between the two pixels, leading to sensitivity under different lighting conditions. Cosine angle calculates the relative angle between the two pixel vectors, with 0 indicating that the pixels are exactly identical or one of

them is a scaled version of the other. Cosine angle is often the metric of choice due to its scale invariance property, allowing for better distinction of materials in different lighting conditions.

$$\begin{aligned} d_{L_2}(\mathbf{x}_i, \mathbf{x}_j) &= \|\mathbf{x}_i - \mathbf{x}_j\|_2 \\ d_\theta(\mathbf{x}_i, \mathbf{x}_j) &= \arccos \left(\frac{\mathbf{x}_i \mathbf{x}_j^T}{\|\mathbf{x}_i\|_2 \|\mathbf{x}_j\|_2} \right) \end{aligned} \quad (14)$$

As $d_{L_2} \in [0, \infty)$ and $d_\theta \in [0, \pi]$, the affinity matrix \mathbf{W} can alternatively be constructed using the heat kernel matrix with $0 < \sigma < 1$, this pushes similar pixels to have $\mathbf{W}(i, j) = 0$ and similar pixels to have $\mathbf{W}(i, j) = 1$.

$$\mathbf{W}_{(i,j)} = \exp \left(-\frac{d(\mathbf{x}_i, \mathbf{x}_j)^2}{\sigma^2} \right). \quad (15)$$

A graph $G = (V, E)$ is a set of vertices V and edges E that connect them. Considering the set of pixels $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n_p}\}$ as the set of vertices and $d(\mathbf{x}_i, \mathbf{x}_j)$ as the edges between them, \mathbf{W} is the matrix representation of a graph G_W . Spectral Analysis is then the study of \mathbf{W} using linear algebra techniques to determine insights on the structure of G_W using the eigenvalues and eigenvectors of \mathbf{W} . A fundamental matrix in spectral analysis is the graph Laplacian matrix \mathbf{L} , calculated as the difference between the diagonal matrix \mathbf{D} where

$$\mathbf{D}_{(i,j)} = \begin{cases} \sum_j \mathbf{W}_{(i,j)} & \text{if } i = j, \\ 0 & \text{if } i \neq j \end{cases} \quad (16)$$

and \mathbf{W} . Formally,

$$\mathbf{L} = \mathbf{D} - \mathbf{W}. \quad (17)$$

As \mathbf{L} is known to be positive semidefinite, the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$ are strictly non-negative and provide key insights into the structure of G .

2.5.1 Normalized Cuts

A graph $G = (V, E)$ with affinity matrix \mathbf{W} can be partitioned into two subgraphs $G_A = (V_A, E_A)$ and $G_B = (V_B, E_B)$ such that $V_A \cup V_B = V$ and $V_A \cap V_B = \emptyset$ by removing the edges between the vertices in G_A and G_B . The dissimilarity between these two graphs can be calculated as the sum of the edges cut to form the partitions

$$\text{cut}(G_A, G_B) = \sum_{i \in V_A, j \in V_B} \mathbf{W}_{(i,j)} \quad (18)$$

The optimal bipartitioning of G_W is given as the graphs G_A and G_B that minimize (18). However, in the case of image segmentation, this criteria will heavily prioritize partitioning single pixels from the image. Instead, the normalized cuts criteria [REF] is proposed, focusing on balancing the ratio between the edges cut and the sum of the internal edge nodes within G_A and G_B defined as

$$\text{ncut}(G_A, G_B) = \frac{\text{cut}(G_A, G_B)}{\text{assoc}(G_A, G)} + \frac{\text{cut}(G_A, G_B)}{\text{assoc}(G_B, G)} \quad (19)$$

where

$$\text{assoc}(G_A, G) = \sum_{i \in V_A, j \in V} \mathbf{W}_{(i,j)}. \quad (20)$$

The normalized cuts criteria, in general terms, aims to minimize the disassociation between the subgraphs and maximize the association within them. While (19) is NP-Complete, Shi et. al [REF] show that solving for the eigenvector \mathbf{u}_2 corresponding to the second smallest eigenvalue λ_2 in the system

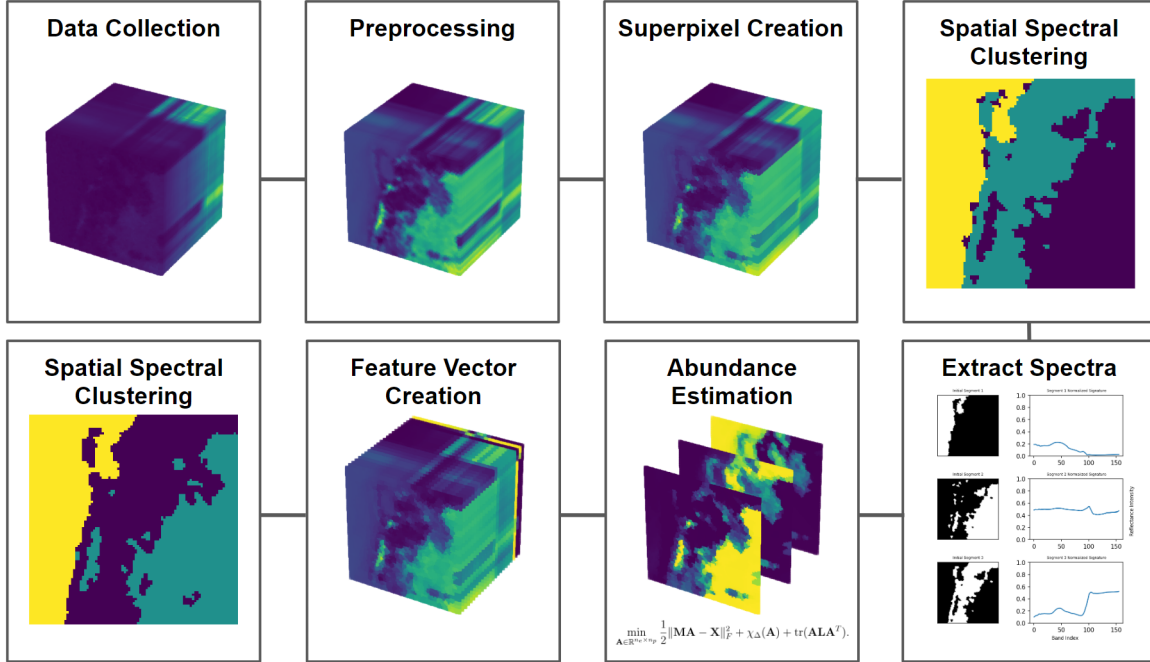
$$\mathbf{D}^{-\frac{1}{2}}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-\frac{1}{2}}\mathbf{z} = \lambda\mathbf{z} \quad (21)$$

provides a approximate real valued solution to (19) through assigning subgraph membership of the vertices according to the sign of the entries in \mathbf{u}_2 .

Spectral graph techniques like the normalized cuts algorithm provide the advantage of flexible and deterministic results based on the initialization of the adjacency matrix \mathbf{W} , however begin to fall behind iterative methods when the graph is large, due to the time complexity of solving (21) scaling cubically with the number of vertices.

3 Adaptive Superpixel Cuts

This section focuses on introducing the proposed unsupervised hyperspectral image segmentation algorithm Adaptive Superpixel Cuts for Hyperspectral Images (ASC-HSI). This algorithm leverages the algorithms proposed in Section 2 to perform image segmentation on a superpixel-basis rather than a pixel-basis, reducing computational and memory costs while allowing for the use of a graph based approach to segmentation and unmixing. The steps of the algorithm can be depicted as follows:



3.1 Dataset Preprocessing

As mentioned in Section 2.1, the input to the algorithm is a hyperspectral image represented by a nonnegative tensor $\hat{\mathbf{X}}$ of shape (n_x, n_y, n_λ) . Raw hyperspectral images are susceptible to outliers and scale invariance across wavelengths, as such, to ensure algorithms perform reliably along a wide range of domains, preprocessing is a crucial step. Typically, the first main goal of practitioners is to deal with noise across the spectral dimension, then work to deal with spatial artifacts in the image. This section will cover the Singular Value Decomposition and Layer Normalization methods that form the basis of the spectral preprocessing methods for the algorithm.

3.1.1 Singular Value Decomposition

\mathbf{X} can be alternatively represented using a nonnegative matrix \mathbf{X}_f of shape (n_λ, n_p) , where $n_p = n_x n_y$ is the total number of pixels in the image, with each column representing the pixel at index (i, j) in \mathbf{X} , arranged across the first spectral dimension, then the second: $\mathbf{X}_f = [\mathbf{x}_{(0,0)}, \dots, \mathbf{x}_{(n_x,0)}, \dots, \mathbf{x}_{(0,n_y)}, \dots, \mathbf{x}_{(n_x,n_y)}]$.

do i even need this to be honest?

3.1.2 Layer Normalization

Hyperspectral images often have bands with significantly higher intensity values compared to others. It is important to consider the full range of wavelengths rather than allow the overprioritization of higher intensity wavelengths, especially in situations where direct scale comparison is made between pixels. By normalizing each spectral band to a similar scale, all spectral bands contribute more equally to the segmentation process. This ensures that results capture the underlying spectral information more effectively. A common approach to normalization in hyperspectral images is band normalization, where intensity values are independently scaled to $[0, 1]$ for each spectral band:

$$\tilde{\mathbf{X}}_{(:,\cdot,k)} = \frac{\mathbf{X}_{(:,\cdot,k)} - \min \mathbf{X}_{(:,\cdot,k)}}{\max \mathbf{X}_{(:,\cdot,k)} - \min \mathbf{X}_{(:,\cdot,k)}} \quad (22)$$

3.2 Hyperspectral Superpixel Generation

- 3.3 Spatial Spectral Clustering**
- 3.4 Graph Regularized Abundance Estimation**
- 3.5 Feature Vector Creation**
- 3.6 Compilation**
- 3.7 Parameter Selection**

4 Experimental Results

4.1 Implementation Details

4.2 Geospatial Testing Datasets

4.2.1 Samson Datasets

4.2.2 Salinas Datasets

4.2.3 Biomedical Autofluorescence Dataset

4.3 Algorithm Evaluation

4.3.1 Qualitative Evaluation on Samson

4.3.2 Quantitative Evaluation on Salinas

4.3.3 Qualitative Evaluation on Biomedical Autofluorescence Data

4.4 Algorithm Comparison

5 Conclusions