Classification of Hyperspectral Images as Tensors Using Nonnegative CP Decomposition

Mohamad Jouni, Mauro Dalla Mura, Pierre Comon

Classification of Hyperspectral Images as Tensors Using Nonnegative CP Decomposition

Mohamad Jouni[1][0000−0002−7258−0935], Mauro Dalla Mura[1][0000−0002−9656−9087], and Pierre Comon[1][0000−0001−9436−9228]

Univ. Grenoble Alpes, CNRS, Grenoble INP, Gipsa-Lab, 38000 Grenoble, France
firstname.lastname@gipsa-lab.fr http://www.gipsa-lab.grenoble-inp.fr

Abstract. A Hyperspectral Image (HSI) is an image that is acquired by means of spatial and spectral acquisitions, over an almost continuous spectrum. Pixelwise classification is an important application in HSI due to the natural spectral diversity that the latter brings. There are many works where spatial information (e.g., contextual relations in a spatial neighborhood) is exploited performing a so-called spectral-spatial classification. In this paper, the problem of spectral-spatial classification is addressed in a different manner. First a transformation based on morphological operators is used with an example on additive morphological decomposition (AMD), resulting in a 4-way block of data. The resulting model is identified using tensor decomposition. We take advantage of the compact form of the tensor decomposition to represent the data in order to finally perform a pixelwise classification. Experimental results show that the proposed method provides better performance in comparison to other state-of-the-art methods.

Keywords: Hyperspectral Imagery · Morphological Profiles · Tensor Decomposition · Scene Classification.

1 Introduction

Hyperspectral Imaging is one of the most important tools in remote sensing. A hyperspectral image (HSI) is typically a three-way[1] block of data acquired when many (two-way) images are taken over hundreds of almost-continuous spectral bands and stacked together forming a so called hypercube. HSI is employed in several fields of applications such as astronomy, Earth and planetary observation, monitoring of natural resources, precision agriculture and biomedical imaging. One of its most common usages is *classification*, i.e., a thematic discrimination of different types of objects present in the image. In other words, the goal of classification is to assign a label to each pixel belonging to the same thematic class according to some of its characteristics (e.g., its spectrum, the homogeneity of its spatial neighborhood or the shape of a region it belongs to). In the case of

[1] The number of *ways* of an array refers to the number of its indices. A HSI is typically a three-way array of dimensions $I_1 \times I_2 \times J$, where I_1 and I_2 are space dimensions (i.e. pixels) and J denotes the number of spectral bands.

HSI, the image can be seen as a mapping from a couple of positive integers (x, y) that correspond to horizontal and vertical positioning, to a vector of positive real values that correspond to the radiance measured by the sensor or the reflectance of the scene. Hence, a direct way to classify a HSI is to take the spectral bands as features and the pixels as samples. However, this approach raises some problems. First, in terms of spectral resolution, the acquisition of HSI induces high dimensionality in the spectra, which means that the samples are put in a high dimensional feature space. Second, by only considering the spectral information can be a limiting factor since spatial information (e.g., contextual relations, size and shape of the objects) is not exploited. For this reason, many works aimed at incorporating spatial information based on the pixels' neighborhood, so that each pixel in the HSI has features that include both spectral and spatial information, and this is known in the literature as *spectral-spatial classification*; such previous works could be found in [1–8] with different approaches.

One way to incorporate spatial information in HSI is by using tools defined in the framework of *Mathematical Morphology* (MM). A well-known example is the morphological profile (MP) [9], which is built by stacking the results of a set of openings and closings based on geodesic reconstruction applied on each band of the HSI (or on each component obtained after dimensionality reduction as in [10]) and has been successfully employed in remote sensing [11, 7]. Attribute Profiles (AP) [4] are another example, in which a scalar image (e.g., a band of a HSI) is filtered with attribute thinnings and thickenings computed on component trees [4] or self-dual attribute filters based on the the tree-of-shapes [12].

Since HSI is by nature a three-way block of data, applying transformations based on MM results in a four-way block of data, which would be of high rank. Since we care about having *low rank* data, we merge the first two ways that correspond to pixel locations. As a result we obtain a three-way block of data where each dimension could be described by pixels, spectral bands, and spatial features (e.g., by means of morphological transformations), respectively.

In this paper we aim at modeling the 3-way hyperspectral data using tensors, and address the problem of pixelwise classification using tensor tools. Our contribution is inspired by [13], where an Additive Morphological Decomposition (AMD) is applied on HSI, resulting in a 4-way block of data, which is then modeled by tensor means. In [13], the corresponding tensor is dealt with using Tensor-PCA (TPCA) in order to apply dimensionality reduction of the data, which is then passed down to classification. In this paper, our motivation comes from the fact that most tensors enjoy a unique Canonical Polyadic (CP) decomposition [14], and we take advantage of how it represents the data for classification purposes. We sum up our contribution as follows:

- CP decomposition is used as a way to represent the pixel data in a compact form, which is more direct and easy to deal with and works thanks to a low dimensionality of the feature space. Moreover, CP decomposition can be adapted to incorporate the nonnegativity of its entries, which leads to a decomposition that can have a physical meaning.

– The proposed method shows promising results, when comparing it to TPCA approach.

The rest of the paper is organized as follows. In Section 2 we briefly talk about adding a new dimension to the data set based on MM. In Section 3 we describe CP decomposition, and explain how it can be a suitable way to directly represent the pixel data in a low dimensional feature space ready for classification. Experiments and results are reported in Section 4. Finally, a conclusion is drawn in Section 5 and some future work is outlined.

2 Mathematical Morphology

Let us consider $\{\bar{\Phi}^i, \underline{\Phi}^i\}_{i=1\ldots m}$, a set of extensive and anti-extensive operators, respectively, where m refers to the size of the filter and the greater m the coarser the filter effect. The MP of a grayscale image (i.e., with scalar pixels value) $\boldsymbol{I}$ using the latter set of operators is defined as:

$$\mathrm{MP}(\boldsymbol{I}) := \{\underline{\Phi}^m(\boldsymbol{I}), \ldots, \underline{\Phi}^1(\boldsymbol{I}), \boldsymbol{I}, \bar{\Phi}^1(\boldsymbol{I}), \ldots, \bar{\Phi}^m(\boldsymbol{I})\}, \tag{1}$$

in which the following ordering relation holds $\mathrm{MP}(\boldsymbol{I})_i \leq \mathrm{MP}(\boldsymbol{I})_{i+1}$ for $i = 1, \ldots, 2m$ and with $\mathrm{MP}(\boldsymbol{I})_i$ denoting the i-th component of the MP. A MP using Opening and Closing by reconstruction is shown in Figure 1.

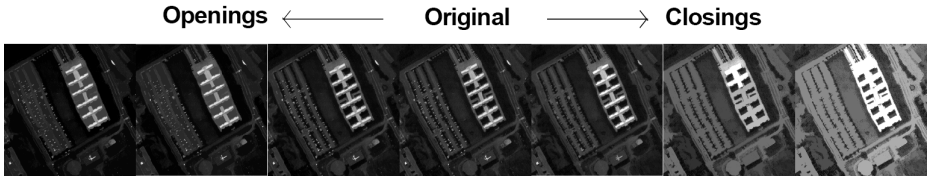

Fig. 1. Morphological profile of a portion of the 8th spectral band of the HSI of University of Pavia using 3 different SE; disks with sizes [1,6,11].

In the case of HSI, or more generally in the case of multivariate images, a direct extension to MP is the Extended Morphological Profiles (EMP) [10]. The concept is the same: one simply concatenates the MP of each image along a fourth way and the result is provided as an input to the classifier. Along the same lines, works in the literature extended this concept to other possible rearrangements or derivations, and an example that we adopt in our experiments is the AMD [13].

As explained in [13], first we suppose that we have any image $\boldsymbol{I}$ decomposed through AMD such that:

$$\boldsymbol{I} = \frac{\bar{\Phi}^m(\bar{\Phi}^{m-1}(\boldsymbol{I})) + \underline{\Phi}^m(\underline{\Phi}^{m-1}(\boldsymbol{I}))}{2} + \sum_{i=1}^{m} \frac{\boldsymbol{R}_i^- - \boldsymbol{R}_i^+}{2} = \boldsymbol{S} + \sum_{i=1}^{m} \boldsymbol{R}_i, \tag{2}$$

The term $\boldsymbol{S}$ is the structure and contains the unfiltered objects of the decomposition, while the terms $\{\boldsymbol{R}_i\}_{i=1\ldots m}$ are the residuals and they contain elements that are filtered out according to the parameters used. Each element of the set

$\{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{R}}_i\}_{i=1\ldots m}$ is a 3-way data set, and at the end they are stacked to form a 4-way data set.

Unlike [13], we prefer to build the morphological way using the set $\{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{R}}_i^-, \boldsymbol{\mathcal{R}}_i^+\}_{i=1\ldots m}$ in order to preserve the nonnegativity of the data, as a result we have a 4-way hyperspectral data set $\boldsymbol{\mathcal{D}} = [\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{R}}_1^-, \boldsymbol{\mathcal{R}}_1^+, \ldots, \boldsymbol{\mathcal{R}}_m^-, \boldsymbol{\mathcal{R}}_m^+]$, which is represented by an $I_1 \times I_2 \times J \times (2m+1)$ array, $I_1 \times I_2$ is the number of pixels, J is the number of spectral bands, and $2m+1 = K$ is the number of terms in the morphological decomposition.

We note that the case study is not restricted to this kind of application, one could also build a 4th way using attribute thinning and thickenings as done in attribute profiles [4] for example.

3 Tensor Decomposition

In this section, we explain how to process by tensor decomposition the data obtained with AMD. $\boldsymbol{\mathcal{D}}$ is of high rank, this can be explained starting with the fact that usually a 2D image is full rank which is high, and a tensor built upon this image will have a rank that is at least as much as that of the image. So we group the first two ways so that the first dimension becomes $I_1 \times I_2 = I$, and $\boldsymbol{\mathcal{D}}$ becomes an $I \times J \times K$ three-way array.

3.1 CP Decomposition

A third order tensor $\boldsymbol{\mathcal{D}}_r$ is referred to as *decomposable* if it can be written as

$$\boldsymbol{\mathcal{D}}_r \overset{\text{def}}{=} \boldsymbol{a}_r \otimes \boldsymbol{b}_r \otimes \boldsymbol{c}_r, \tag{3}$$

A tensor $\boldsymbol{\mathcal{Y}}$ can always be decomposed into a sum of decomposable tensors $\boldsymbol{\mathcal{D}}_r$:

$$\boldsymbol{\mathcal{Y}} = \sum_{r=1}^{R} \lambda_r \, \boldsymbol{\mathcal{D}}_r, \tag{4}$$

The *tensor rank*, usually denoted by R, is the least number of terms required such that the *CP decomposition* (4) is exact. Suppose we have a 3-way tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I \times J \times K}$, each decomposable term $\boldsymbol{\mathcal{D}}_r$ in the CP decomposition can be expressed as in (3), where the set of vectors $\{\boldsymbol{a}_r\}_{r=1\ldots R} \in \mathbb{R}^I$, $\{\boldsymbol{b}_r\}_{r=1\ldots R} \in \mathbb{R}^J$, and $\{\boldsymbol{c}_r\}_{r=1\ldots R} \in \mathbb{R}^K$ form the columns of the so-called factor matrices, $\boldsymbol{A} \in \mathbb{R}^{I \times R}$, $\boldsymbol{B} \in \mathbb{R}^{J \times R}$, and $\boldsymbol{C} \in \mathbb{R}^{K \times R}$. Hence another way to write equation 4 is, with an obvious notation:

$$\boldsymbol{\mathcal{Y}} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}) \cdot \boldsymbol{\Lambda} \tag{5}$$

The diagonal tensor $\boldsymbol{\Lambda}$ governs the interaction between the components (columns) of the factor matrices such that components of different r indices do not interact. R can then be seen as directly related to the degrees of freedom in the decomposition.

When $\boldsymbol{\mathcal{Y}}$ is nonnegative, it is often desirable to constrain factor matrices to be also nonnegative. In this case, the rank is called *nonnegative rank* and sometimes denoted R_+. Having nonnegative factors is essential for the sake of physical interpretation of the results where negative values have no meaning. Consequently, the goal is to minimize the cost function:

$$\begin{aligned} &\frac{1}{2}\|\boldsymbol{\mathcal{Y}} - (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}).\boldsymbol{\Lambda}\|^2 \\ &\text{s.t. } \boldsymbol{A} \succeq 0, \boldsymbol{B} \succeq 0, \boldsymbol{C} \succeq 0 \end{aligned} \tag{6}$$

An interesting aspect is that each of the factor matrices represents one way of the tensor. For example, an element $y_{i,j,k}$ is decomposed into R_+ components and the decomposed information is mapped to the i-th row of $\boldsymbol{A}$, the j-th row of $\boldsymbol{B}$, and the k-th row of $\boldsymbol{C}$.

In the case of HSI, $\boldsymbol{A}$ corresponds to the way of pixels, $\boldsymbol{B}$ corresponds to the way of spectral bands, and $\boldsymbol{C}$ corresponds to the way based on MM. Accordingly each row of $\boldsymbol{A}$ represents a pixel, that of $\boldsymbol{B}$ represents a spectral band, and that of $\boldsymbol{C}$ is defined by a scale or a structure. Thanks to the nonnegativity constraints and especially after normalizing the columns of the factor matrices, one could assume that columns of $\boldsymbol{B}$ represent spectral signatures and those of $\boldsymbol{C}$ represent a combination of scale domination along the set of structuring elements being chosen, and finally each column of $\boldsymbol{A}$ could be seen as an abundance map, that could be reshaped to show a grayscale image, with respect to information received at $\boldsymbol{B}$ and $\boldsymbol{C}$, very similar to the case of spectral unmixing (but for two dimensions only) – except that unmixing is presently not our concern.

In this paper, we take advantage of the low-rank aspect of our CP decomposition, i.e. how it represents the data in a more compact form with almost no loss of information (depending on how exact the decomposition is). First, instead of dealing with a multidimensional data set of large dimensions, we are able to deal with matrices of much smaller size, as explained earlier (since a small rank permits compression). Second, since our approach is pixelwise classification, we mainly think of $\boldsymbol{A}$ as the matrix to deal with. Theoretically speaking, similar pixels should have the same composition of coefficients with respect to the components in $\boldsymbol{A}$, so we take the components (columns) as features and the rows as samples in the classification.

3.2 High-Order Singular Value Decomposition (HOSVD)

Because of the huge dimensions of our tensor, directly applying CP decomposition would be computationally very demanding. One solution to make the decomposition possible is to pass by a pre-processing step by compressing the original tensor into one with smaller dimensions without losing any information. This can be done by providing the minimal compression size for each way such that the information is preserved. In that sense, let's suppose a tensor $\boldsymbol{\mathcal{Y}}$ with dimensions $I \times J \times K$, where $I \gg JK$ (which is our case), the compressed tensor can have a size of $JK \times J \times K$ whenever its rank satisfies $R \leq JK$.

Algorithm 1 COMPRESS

Require: $\boldsymbol{\mathcal{Y}}$, $\boldsymbol{way_c}$, $\boldsymbol{dim_c}$
  for $d \in \boldsymbol{way_c}$ **do**
    Unfold $\boldsymbol{\mathcal{Y}}$ into $\boldsymbol{Y}_{(d)}$ such that the mode d takes the second way of the matrix;
    Compute the right singular matrix from the SVD of $\boldsymbol{Y}_{(d)}$, denoted by $\boldsymbol{V}_d$;
    Truncate the columns of $\boldsymbol{V}_d$ by $\boldsymbol{dim_c}(d)$;
  end for
  for $d \notin \boldsymbol{way_c}$ **do**
    $\boldsymbol{V}_d = \boldsymbol{I}_{\boldsymbol{dim_c(d)}}$; (Identity matrix)
  end for
  $\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{Y}} \bullet_1 \boldsymbol{V}_1^T \bullet_2 \ldots \bullet_N \boldsymbol{V}_N^T$;
  return $\boldsymbol{\mathcal{G}}$ and $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_N$

This can be explained as follows. First let us denote the d-mode unfolding of $\boldsymbol{\mathcal{Y}}$ by $\boldsymbol{Y}_{(d)}$, which is the matrix flattening of the tensor obtained by reordering the elements with respect to the d-th index. Each of these unfoldings has a matrix rank that essentially appears in its economic SVD. More precisely, after applying the SVD on each unfolding, we get three right singular (orthogonal) matrices that form the basis of the HOSVD factor matrices:

$$\boldsymbol{Y}_{(d)} = \boldsymbol{U}_d \boldsymbol{\Sigma}_d \boldsymbol{V}_d^T \quad \forall d = \{1, 2, 3\} \tag{7}$$

These matrices, denoted by $\boldsymbol{V}_1$, $\boldsymbol{V}_2$, and $\boldsymbol{V}_3$ have dimensions $I \times R_1$, $J \times R_2$ and $K \times R_3$ respectively, with $R_1 \leq JK$, $R_2 \leq J$, and $R_3 \leq K$. Hence they can be used to compress the original tensor into smaller dimensions without any loss of information. In fact, this is how HOSVD is computed by finding orthogonal matrices as the basis of the decomposition, and the compressed tensor $\boldsymbol{\mathcal{G}}$ is found by projecting the original one onto the basis of factor matrices. HOSVD of $\boldsymbol{\mathcal{Y}}$ can be written as follows:

$$\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{G}} \underset{1}{\bullet} \boldsymbol{V}_1 \underset{2}{\bullet} \boldsymbol{V}_2 \underset{3}{\bullet} \boldsymbol{V}_3 = (\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3) \cdot \boldsymbol{\mathcal{G}} \tag{8}$$

where $\bullet_d$ indicates a summation over the dth tensor index. Subsequently, we shall compress only the first mode, so that $R_1 < JK$, but $R_2 = J$ and $R_3 = K$. As a consequence, we may choose $\boldsymbol{V}_2$ and $\boldsymbol{V}_3$ both as Identity matrices. The algorithm that we implement to compress an N-way tensor using HOSVD is described in Algorithm 1; $\boldsymbol{way_c}$ is a vector containing the ways that we want to compress and $\boldsymbol{dim_c}$ is a vector containing the compressed dimension size.

3.3 Alternating Optimization - Alternating Direction Method of Multipliers (AO-ADMM)

In order to compute the CP decomposition and cope with the constraints of non-negativitity and compression together, the algorithm AO-ADMM [15] is adopted for its efficiency and flexibility especially with the different kinds of constraints.

The CP decomposition can be computed via the Alternating Least Squares algorithm (ALS), where one factor matrix is updated alternatingly at a time by fixing the others and minimizing the convex least squares cost function:

$$\boldsymbol{H}_d = \underset{\boldsymbol{H}_d}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{Y}_{(d)} - \boldsymbol{W}\boldsymbol{H}_d^T\|^2 \quad \forall d = 1 \dots N, \tag{9}$$

where $\boldsymbol{H}_d$ is the factor matrix corresponding to the d-th way of the tensor, and $\boldsymbol{W}$ is the Khatri-Rao product of all the factor matrices excluding $\boldsymbol{H}_d$. The optimization problem (9) needs to be modified when constraints on the factor matrices are imposed, say through regularization functions $r(\boldsymbol{H}_d)$:

$$\boldsymbol{H}_d = \underset{\boldsymbol{H}_d}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{Y}_{(d)} - \boldsymbol{W}\boldsymbol{H}_d^T\|^2 + r(\boldsymbol{H}_d) \quad \forall d = 1 \dots N, \tag{10}$$

As the name suggests, AO-ADMM works by alternating between the optimization problems of each factor matrix alone while fixing the others. For example in our case, the three factor matrices $(\boldsymbol{H}_1, \boldsymbol{H}_2, \boldsymbol{H}_3)$ are recommended to have nonnegative entries, while only the first one is concerned with compression. AO-ADMM allows to pass different kinds of constraints and parameters for each mode-decomposition and alternates accordingly.

Algorithm 2 Alternating Optimization (Least Squares loss)

Require: $\boldsymbol{\mathcal{Y}}$, $\boldsymbol{H}_1, \dots, \boldsymbol{H}_N$, $\boldsymbol{U}_1, \dots, \boldsymbol{U}_N$
  $\boldsymbol{H}_1, \dots, \boldsymbol{H}_N$ are initialized; $\boldsymbol{U}_1, \dots, \boldsymbol{U}_N$ are initialized to zero matrices;
  for $d = 1, \dots, N$ **do**
    Store the different unfoldings $\boldsymbol{Y}_d$;
  end for
  repeat
    for $d = 1, \dots, N$ **do**
      $\boldsymbol{W} = \odot_{j \neq d} \boldsymbol{H}_j$;
      update $\boldsymbol{H}_d$ and $\boldsymbol{U}_d$ using Algorithm 3;
    end for
    update μ if necessary; (refer to [15] for the update of μ)
  until some termination criterion is reached (number of iterations)
  Normalize the columns of the factor matrices and store the weights in $\boldsymbol{\Lambda}$;
  return $\boldsymbol{H}_1, \dots, \boldsymbol{H}_N$, $\boldsymbol{\Lambda}$

It is important to note that the tensor fed in AO-ADMM is the compressed one that is computed as a result of Section 3.2, which allows to have negative entries, and this presents a certain inconsistency with the fact of having nonnegativity constraints on the decomposition. For this reason we present a solution with ADMM inspired by [16]. So our minimization problem goes as follows:

$$\begin{aligned} \arg \min_{\boldsymbol{H}_1, \boldsymbol{H}_2, \boldsymbol{H}_3, \boldsymbol{\Lambda}} \ & \frac{1}{2} \|\boldsymbol{\mathcal{G}} - (\boldsymbol{H}_1, \boldsymbol{H}_2, \boldsymbol{H}_3).\boldsymbol{\Lambda}\|^2 \\ \text{s.t. } & \boldsymbol{V}_1\boldsymbol{H}_1 \succeq 0, \boldsymbol{H}_2 \succeq 0, \boldsymbol{H}_3 \succeq 0, \boldsymbol{\Lambda} \succeq 0 \end{aligned} \tag{11}$$

Algorithm 3 ADMM of a mode d

Require: $\boldsymbol{Y}$, $\boldsymbol{W}$, $\boldsymbol{H}$, $\boldsymbol{U}$, k, μ, ϵ, constraint, $imax$, $\boldsymbol{V}$
  $\boldsymbol{H}$ and $\boldsymbol{U}$ are already initialized;
  $\boldsymbol{G} = \boldsymbol{W}^T\boldsymbol{W}$;
  $\rho = \text{trace}(\boldsymbol{G})/k$;
  Calculate $\boldsymbol{L}$ from Cholesky decomposition such that $\boldsymbol{G} + (\rho + \mu)\boldsymbol{I_k} = \boldsymbol{L}\boldsymbol{L}^T$;
  $\boldsymbol{F} = \boldsymbol{W}^T\boldsymbol{Y}$;
  $\boldsymbol{H_f} = \boldsymbol{H}$;
  repeat
    $\tilde{\boldsymbol{H}} \leftarrow (\boldsymbol{L}^T)^{-1}\boldsymbol{L}^{-1}(\boldsymbol{F} + \rho(\boldsymbol{H} + \boldsymbol{U})^T + \mu\boldsymbol{H_f}^T)$; (See [15] for μ and $\boldsymbol{H_f}$)
    $\boldsymbol{H} \leftarrow$ proximity(constraint,$\tilde{\boldsymbol{H}}^T$,$\boldsymbol{U}$,$\boldsymbol{V}$); (refer to Algorithm 4)
    $\boldsymbol{U} \leftarrow \boldsymbol{U} + \boldsymbol{H} - \tilde{\boldsymbol{H}}^T$;
    Update r and s; (refer to [15] for the updates of r and s)
  until ($r < \epsilon$ and (s is undefined or $s < \epsilon$)) or ($imax > 0$ and $i \geq imax$)
  return $\boldsymbol{H}$ and $\boldsymbol{U}$

Algorithm 4 Proximity Update of $\boldsymbol{H}$ in ADMM

Require: constraint, $\boldsymbol{H}_t$, $\boldsymbol{U}$, $\boldsymbol{V}$
  switch (constraint)
  case Nonnegativity**:**
    $\boldsymbol{H} \leftarrow (\boldsymbol{H}_t - \boldsymbol{U})^+$;
  case Compression and Nonnegativity**:**
    $\boldsymbol{H}_u \leftarrow \boldsymbol{V}(\boldsymbol{H}_t - \boldsymbol{U})$; $\boldsymbol{H}_u \leftarrow \boldsymbol{H}_u^+$; $\boldsymbol{H} \leftarrow \boldsymbol{V}^T\boldsymbol{H}_u$;
  end switch
  return $\boldsymbol{H}$

This minimization is executed by ADMM as follows, for $d \in \{2, 3\}$:

$$\left.\begin{array}{l} \tilde{\boldsymbol{H}}_1 \leftarrow (\boldsymbol{W}^T\boldsymbol{W} + \rho\boldsymbol{I})^{-1}(\boldsymbol{W}^T\boldsymbol{G}_{(1)} + \rho(\boldsymbol{U} + \boldsymbol{H}_1)^T) \\ \boldsymbol{H}_1 \leftarrow \boldsymbol{V}_1^T \max(0, \boldsymbol{V}_1(\tilde{\boldsymbol{H}}_1^T - \boldsymbol{U})) \\ \boldsymbol{U} \quad \leftarrow \boldsymbol{U} + \boldsymbol{H}_1 - \tilde{\boldsymbol{H}}_1^T \end{array}\right\} \tag{12}$$

$$\left.\begin{array}{l} \tilde{\boldsymbol{H}}_d \leftarrow (\boldsymbol{W}^T\boldsymbol{W} + \rho\boldsymbol{I})^{-1}(\boldsymbol{W}^T\boldsymbol{G}_{(d)} + \rho(\boldsymbol{U} + \boldsymbol{H}_d)^T) \\ \boldsymbol{H}_d \leftarrow \max(0, \tilde{\boldsymbol{H}}_d^T - \boldsymbol{U}) \\ \boldsymbol{U} \quad \leftarrow \boldsymbol{U} + \boldsymbol{H}_d - \tilde{\boldsymbol{H}}_d^T \end{array}\right\} \tag{13}$$

Of course, the calculation of $(\boldsymbol{W}^T\boldsymbol{W} + \rho\boldsymbol{I})^{-1}$ should be done once for all outside the loops, as explained in the pseudo-codes Algs. 2, 3, and 4.

4 Results

4.1 Description of the dataset

Our dataset is described by a real hyperspectral image, University of Pavia, as shown in Fig. 2 in true colors (by choosing the bands 53, 31, and 8 as Red, Green, and Blue channels respectively). It is an image acquired by the ROSIS sensor with geometric resolution of 1.3 meters. The image has 610×340 pixels and 103 spectral bands. The dataset contains a groundtruth image that consists

of 9 classes: trees, asphalt, bitumen, gravel, metal sheets, shadows, self-blocking bricks, meadows, and bare soil. There are also 42776 labeled pixels available as test set and 3921 pixels available as training set, the latter is fixed at that.

4.2 Classification

For the classification part we use Support Vector Machines (SVM), which has proved useful in the application of pixelwise classification of HSI. Practically SVM is carried out through the open source machine learning library provided by Libsvm [17]. After decomposing the tensor using CP decomposition, we use the first factor matrix as the set of data to be classified by SVM.

4.3 Results and discussion

In the following experiments, we fix the morphological transformations on structuring elements whose neighborhoods are defined arbitrarily by disk shapes with varying sizes of radii: 1, 6, 11, 16, 21, and 26 pixels, this accounts to $m = 6$ structuring elements, thus a morphological decomposition of $K = 13$ components. We note that considering the whole set of $\{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{R}}_i^-, \boldsymbol{\mathcal{R}}_i^+\}_{i=1\ldots6}$ without dimensionality reduction yielded a classification with better accuracy than that of the set of $\{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{R}}_i\}_{i=1\ldots6}$ (96.60% to 93.06% respectively) under the same AMD parameters.

AMD. The HSI of University of Pavia is loaded in its 3rd-order form, $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{610\times340\times103}$. $\boldsymbol{\mathcal{I}}$ is then decomposed into a morphological structure, going from a dataset of one tensor to a set of tensors ($\boldsymbol{\mathcal{S}}$, $\boldsymbol{\mathcal{R}}^-$'s and $\boldsymbol{\mathcal{R}}^+$'s, forming a 4th-order dataset) representing information of the scene with different scales:

$$\boldsymbol{\mathcal{I}} \to \boldsymbol{\mathcal{D}} = \{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{R}}_1^-, \boldsymbol{\mathcal{R}}_1^+, \ldots, \boldsymbol{\mathcal{R}}_6^-, \boldsymbol{\mathcal{R}}_6^+\}$$

such that $\boldsymbol{\mathcal{I}} = \boldsymbol{\mathcal{S}} + \sum_{i=1}^{6} \frac{\boldsymbol{\mathcal{R}}_i^- - \boldsymbol{\mathcal{R}}_i^+}{2}$. As we mentioned in Section 3, $\boldsymbol{\mathcal{D}}$ is vectorized by merging the first two modes resulting in a 3-way tensor of dimensions $207400 \times 103 \times 13$.

Tensors. Now that the data is modeled as a 3rd-order tensor, we seek to decompose it using CP decomposition. The latter is carried out by taking the compressed tensor as input, the factor matrices are initialized randomly (absolute value of i.i.d. standard Gaussian distribution), and only the first factor matrix is compressed after that. The rank is set to different values followed by different initializations of factor matrices.

Classification. The accuracy of the classification is related to the reconstruction error of the CP decomposition since it is important that factor matrix $\boldsymbol{H}_1$ represents the data in a shape as good as possible. Two factors play major roles in the reconstruction error: the number of iterations and the input rank of the decomposition. Hence we try different values of the two variables and record both the reconstruction error and the result of the overall accuracy as seen in Table 1. Per-class and average accuracy is also recorded for two methods in Table 2. As discussed in Section 3.1, $\boldsymbol{H}_1$ is passed to the classifier since it is considered as the representative of pixel data in compact low-dimensional form.

Method	Rec. Error	OA
AMD + TPCA + SVM (PC=(10,5))	-	91.09
AMD + CPD + SVM (itr=50,R=10)	18.45	87.87
AMD + CPD + SVM (itr=20,R=20)	11.10	88.85
AMD + CPD + SVM (itr=50,R=20)	10.94	**91.17**
AMD + CPD + SVM (itr=20,R=30)	8.86	**93.76**
AMD + CPD + SVM (itr=50,R=30)	8.72	**94.08**
AMD + CPD + SVM (itr=100,R=30)	8.70	**93.94**
AMD + CPD + SVM (itr=100,R=40)	7.16	**94.03**

Table 1. Records of various tests in terms of Reconstruction Error (Rec. Error) (in case of CPD) and Overall Accuracy (OA).

Class	AMD + TPCA (PC=(10,5))	AMD + CPD (itr=50,R=30)
Asphalt	95.54	**98.46**
Meadow	94.08	**96.42**
Gravel	**72.12**	59.83
Tree	97.59	**98.00**
Metal Sheet	**100**	99.64
Bare Soil	81.56	**92.71**
Bitumen	**99.79**	99.69
Brick	76.66	**88.49**
Shadow	**99.87**	86.66
Overall	91.09	**94.08**
Average	90.80	**91.10**

Table 2. Records of Per-class, Overall and Average Accuracies.

Results. We compare our results to the method of [13]. Looking at Table 1, we can see that the better the reconstruction error in the CP decomposition, the better the overall accuracy of the classification. The accuracy shows promising results for our proposed method. It is worth to stress the influence of rank and/or number of iterations on the reconstruction error. But at some point the error decreases very slowly as more iterations are run, towards what seems to be a limit, which is mainly influenced by the value of the input rank. Figure 2 shows images of the training set, the test set, and the results of various classifications; parameters are mentioned in the caption.

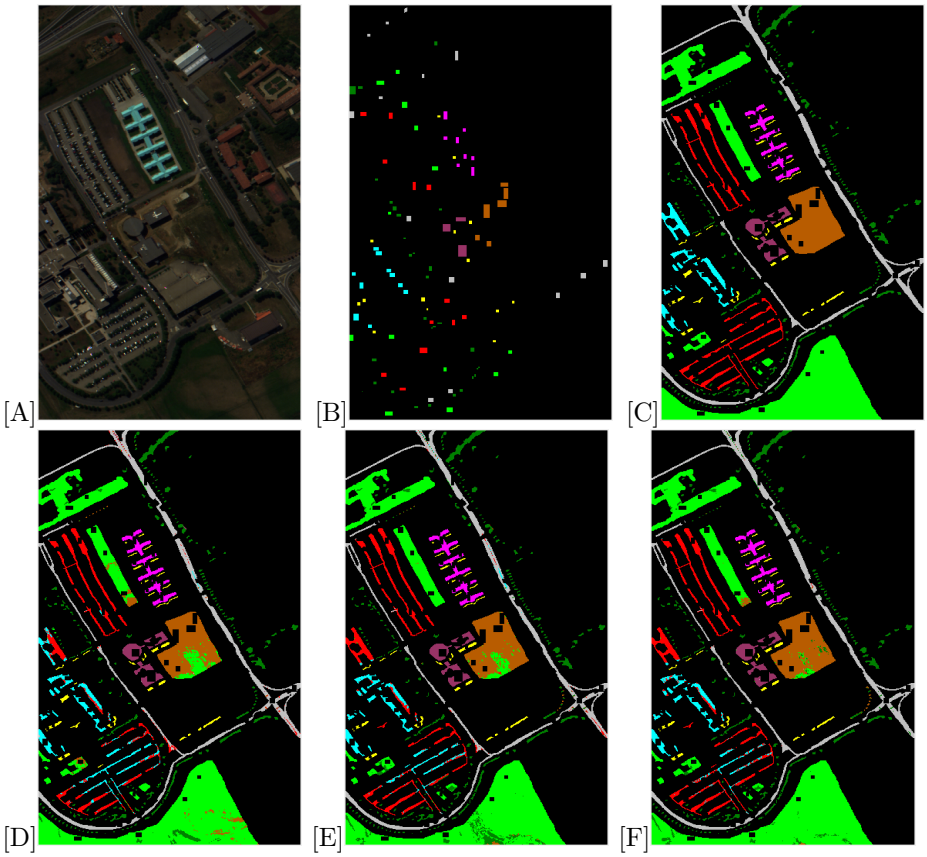

Fig. 2 [A] image in true colors, [B] available training set, [C] test set, [D-F] classification results: [D] AMD+TPCA(15)+SVM (91.09%), [E] AMD+CPD (itr=50,R=20)+SVM (91.17%), [F] AMD+CPD (itr=50,R=30)+SVM (94.08%)

5 Conclusion

In the framework of pixelwise classification of hyperspectral images, an extraneous diversity is built from hyperspectral data thanks to Mathematical Morphology. Then the use of tensor decomposition allows to take into account both spectral and spatial information of the scene. Experimental results using a real image show that the CP decomposition is a promising way to represent the data in a low dimensional feature space and improve on classification accuracy. In future works, we plan to investigate other potential interests of the CP decomposition, like spectral unmixing of HSI.

References

1. Lorenzo Bruzzone and Lorenzo Carlin. A multilevel context-based system for classification of very high spatial resolution images. *IEEE transactions on Geoscience and Remote Sensing*, 44(9):2587–2600, 2006.

2. Julio M Duarte-Carvajalino, Guillermo Sapiro, Miguel Vélez-Reyes, and Paul E Castillo. Multiscale representation and segmentation of hyperspectral imagery using geometric partial differential equations and algebraic multigrid methods. *IEEE Transactions on Geoscience and Remote Sensing*, 46(8):2418–2434, 2008.
3. Jun Li, José M Bioucas-Dias, and Antonio Plaza. Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):809–823, 2012.
4. Mauro Dalla Mura, Jón Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, 2010.
5. Mathieu Fauvel, Jocelyn Chanussot, and Jon Atli Benediktsson. A spatial–spectral kernel-based approach for the classification of remote-sensing images. *Pattern Recognition*, 45(1):381–392, 2012.
6. Mathieu Fauvel, Yuliya Tarabalka, Jon Atli Benediktsson, Jocelyn Chanussot, and James C Tilton. Advances in spectral-spatial classification of hyperspectral images. *Proceedings of the IEEE*, 101(3):652–675, 2013.
7. Pedram Ghamisi, Mauro Dalla Mura, and Jon Atli Benediktsson. A survey on spectral–spatial classification techniques based on attribute profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 53(5):2335–2353, 2015.
8. Pedram Ghamisi, Emmanuel Maggiori, Shutao Li, Roberto Souza, Yuliya Tarabalka, Gabriele Moser, Andrea De Giorgi, Leyuan Fang, Yushi Chen, Mingmin Chi, et al. Frontiers in spectral-spatial classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Magazine*, 2018.
9. Jon Atli Benediktsson, Martino Pesaresi, and Kolbeinn Amason. Classification and feature extraction for remote sensing images from urban areas based on morphological transformations. *IEEE Transactions on Geoscience and Remote Sensing*, 41(9):1940–1949, 2003.
10. Jón Atli Benediktsson, Jón Aevar Palmason, and Johannes R Sveinsson. Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):480–491, 2005.
11. Mauro Dalla Mura, Jon Atli Benediktsson, Jocelyn Chanussot, and Lorenzo Bruzzone. The evolution of the morphological profile: From panchromatic to hyperspectral images. In *Optical Remote Sensing*, pages 123–146. Springer, Berlin, Heidelberg, 2011.
12. Gabriele Cavallaro, Mauro Dalla Mura, Jon Atli Benediktsson, and Antonio Plaza. Remote sensing image classification using attribute filters defined over the tree of shapes. *IEEE Transactions on Geoscience and Remote Sensing*, 54(7):3899–3911, 2016.
13. Santiago Velasco-Forero and Jesus Angulo. Classification of hyperspectral images by tensor modeling and additive morphological decomposition. *Pattern Recognition*, 46(2):566–577, 2013.
14. Pierre Comon. Tensors: a brief introduction. *IEEE Sig. Proc. Magazine*, 31(3):44–53, May 2014. hal-00923279.
15. Kejun Huang, Nicholas D Sidiropoulos, and Athanasios P Liavas. A flexible and efficient algorithmic framework for constrained matrix and tensor factorization. *IEEE Transactions on Signal Processing*, 64(19):5052–5065, 2016.
16. Jeremy Cohen, Rodrigo Cabral Farias, and Pierre Comon. Fast decomposition of large nonnegative tensors. *IEEE Signal Processing Letters*, 22(7):862–866, 2015.
17. Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.