A Fast Approximate Spectral Unmixing Algorithm based on Segmentation

Jing Ke[1, 2], Yi Guo[3], Arcot Sowmya[1]
1- School of Computer Science and Engineering, University of New South Wales
2- Commonwealth Scientific and Industrial Research Organization Data 61
3- School of Computing, Engineering and Mathematics, Western Sydney University
Sydney, Australia
crystal.ke@data61.csiro.au, Y.Guo@westernsydney.edu.au

Abstract

Efficient processing of spectral unmixing is a challenging problem in high-resolution satellite data analysis. The decomposition of a pixel into a linear combination of pure spectra and into their corresponding proportions is often very time-consuming. In this paper, a fast unmixing algorithm is proposed based on classifying pixels into a full unmixing group for subset selection requiring intensive computational procedures and a partial unmixing group for proportion estimation with known spectra endmembers. The classification is based on n-band spectral segmentation using the quick-shift algorithm. A subset selection algorithm applied on real satellite data evaluates accuracy and approximation, and experimental results show significant performance acceleration compared with the original algorithm. Parallelization strategies are also presented and verified on NVIDIA GTX TITAN X.

1. Introduction

Hyper-spectral imaging is a very useful imaging technique with wide applications in environment monitoring, natural disaster control and accurate agricultural monitoring, to enumerate only a few. The spectral unmixing process is the well-known inverse problem: given a hyper-spectral image and a spectra library, find the collection of constituent spectra, or endmembers combinations, and their corresponding fractions, or abundances that indicate the proportion of each endmember present in the pixel. The mixing process can be considered to be linear or non-linear. The combination will be basically linear if the endmembers in a pixel appear in spatially segregated patterns [1, 2, 3].

There are several techniques in the literature to estimate land-cover proportions within mixed pixels based on fuzzy set theory [4, 5, 6], neural networks [7] and spectral mixture analysis (SMA) [8]. Although unmixing processes can be non-linear, linear methods are the most used, specifically linear spectral mixture analysis (LSMA). In recent years, many authors have proposed more complex models that provide a higher accuracy and use less computing time [9].

In this research on spectral mixture analysis, a variable subset selection (VSS) based linear regression model is used to estimate the land-cover fraction in remotely sensed spectral data for its high accuracy. The model is adopted by some commercial packages, for example in minerals detection [10, 11]. The problem is as follows: assume G pure spectra in the spectral library, written as $\mathbf{X} \in \mathrm{R}^{\mathrm{D}\times\mathrm{G}}$,and a given spectrum $\boldsymbol{y} \in \mathrm{R}^{\mathrm{D}}$, corresponding to a pixel in a spectral image. Both $\mathbf{X}$ and $\boldsymbol{y}$ contain D wavelengths. If we believe that our spectrum is a mixture of M materials, and $M \ll G$, VSS searches for the best linear combination in terms of the least residual sum of squares (RSS) among all possible combinations:

$$\min_{|\mathcal{S}|=\mathrm{M}} |\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\mathcal{S}}}\boldsymbol{\beta}_{\boldsymbol{\mathcal{S}}}|_2^2 \qquad (1)$$

where $\mathcal{S}$ is a subset of variables (columns) from $\mathbf{X}$, $\mathbf{X}_{\boldsymbol{\mathcal{S}}}$ is a submatrix retaining columns specified by $\mathcal{S}$, $\boldsymbol{\beta}_{\boldsymbol{\mathcal{S}}}$ is the regression coefficient, and $|\mathcal{S}|$ is the cardinality of set $\mathcal{S}$. The solution to Eq. (1) given $\mathbf{X}_{\boldsymbol{\mathcal{S}}}$ is the left product of the Moore–Penrose pseudo inverse of $\mathbf{X}_{\boldsymbol{\mathcal{S}}}$ and $\boldsymbol{y}$:

$$\boldsymbol{\beta}_{\boldsymbol{\mathcal{S}}} = (\mathbf{X}_{\boldsymbol{\mathcal{S}}}^{\boldsymbol{T}}\mathbf{X}_{\boldsymbol{\mathcal{S}}})^{-\mathbf{1}}\mathbf{X}_{\boldsymbol{\mathcal{S}}}^{\boldsymbol{T}}\boldsymbol{y} \qquad (2)$$

and the RSS associated with set $\mathcal{S}$, written as $RSS_{\mathcal{S}}$ is:

$$RSS_{\mathcal{S}} = \boldsymbol{y}^{\mathbf{T}}\boldsymbol{y} - \boldsymbol{y}^{\mathbf{T}}\mathbf{X}_{\boldsymbol{\mathcal{S}}}(\mathbf{X}_{\boldsymbol{\mathcal{S}}}^{\boldsymbol{T}}\mathbf{X}_{\boldsymbol{\mathcal{S}}})^{-\mathbf{1}}\mathbf{X}_{\boldsymbol{\mathcal{S}}}^{\boldsymbol{T}}\boldsymbol{y} \qquad (3)$$

Although this linear regression model has shown its superiority over other existing unmixing algorithms [10, 11], $\mathbf{X}_{\boldsymbol{\mathcal{S}}}$ has to be determined for every pixel $\boldsymbol{y}$ in the spectral image. The computational complexity of minimizing Eq. (1) which is a subset selection problem, is of the order of $O(DM^2\binom{G}{M})$, and it increases dramatically with number M.

Under the linear spectral mixing model, unmixing approaches mainly involve two procedures: endmember acquisition and proportion estimation [12], with both normally performed at pixel or sub-pixel level. The high spatial resolution of modern hyper-spectral sensors and technology poses challenges [13] to real-time processing. This issue has been addressed in unmixing algorithms based on real datasets [11]. Moreover, due to the large size of intermediate data generated in unmixing

computation and the special spectral data layout, parallelization is often performed at basic arithmetic instruction level rather than pixel level, and the performance speedup is limited compared to the massive parallel computing power of GPUs. Fast and efficient approaches are required to resolve these performance issues.

In this paper, an accelerating approach to the unmixing algorithm is proposed based on the assumption that in high spatial satellite images, pixels from homogeneous and neighboring regions are likely to contain a similar endmember combination but different relative proportions of ground cover components, while pixels from heterogeneous and distant regions are likely to contain quite different endmember combinations. Assuming spatial smoothness in remotely sensed images, an optimized and efficient unmixing approach based on a subset selection algorithm is proposed in this work. The main contributions are as follows:

i. We present a fast spectral unmixing framework for endmember selection and proportion estimation. Satellite image pixels are separated into two groups requiring different computational workload using the quick-shift segmentation approach on the n-band spectral image. Only those pixels determined as the root of a segment are unmixed into their constituent classes from a spectral library, and require intensive computation. Using the computed endmembers combinations of the root pixel, all other pixels in the segment dictated by the root pixels for endmembers go through spectra proportion estimation. An unmixing benchmark based on a subset selection algorithm on real satellite spectral images shows significant performance speedup.
ii. We propose a segmentation-based method to improve the "roughness" problem between adjacent pixels due to noise variance in the obtained spectral data and imperfection in the linear model. Experiments using the original brute-force algorithm based on per-pixel level computation show that linear combinations of endmembers for neighboring pixels may be quite different even for highly similar observations. By assuring spatial smoothness for normalized spectral data based on segmentation, the proposed method facilitates subsequent steps after unmixing such as finding a particular mineral or segmentation of one or two combinations of geometrical features [11].
iii. We provide efficient GPU-based parallelization strategies for further acceleration with performance analysis. This framework is also applicable to many other multi-spectral or hyper-spectral unmixing algorithms for high-resolution spatial data.

2. Fast unmixing framework

In the current work, preprocessing of spectral data is performed to generate the covariance matrix. Quick-shift algorithm based segmentation is then used to identify the roots pixel for each pixel in the segment. Only the root pixels are involved in the time-consuming full unmixing procedure to define endmember spectra. For all other pixels, estimation of pure spectra proportions is achieved with a much smaller computation workload. The pipeline of acceleration procedures is displayed in Fig. 1. The steps are discussed in the subsections.

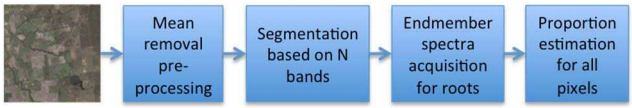


Figure 1: Basic pipeline of spectral unmixing acceleration

2.1. Preprocessing

Rescaling is applied to the spectral matrix dataset to reduce brightness variation for the subsequent segmentation and matrix multiplication:

$$\begin{bmatrix} y_1' \\ y_2' \\ \vdots \\ y_n' \end{bmatrix} = \frac{1}{\sqrt{(y_1^2 + y_2^2 + \cdots y_n^2)}} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad (4)$$

2.2. Constraints

To estimate the best-fitting pure materials, we need to verify that reflectance coefficient vector β_i lies within (0,1] (which means that the logarithms must be non-positive) for every material. The full additive condition requires the fractions to sum to one in Eq. (2), as the non-negative condition requires all abundances to be non-negative [9]. Estimating the number of materials is another constraint in Eq. (1). The obvious way to estimate the number of materials in a mixture is to keep adding more materials to the mixture until an appropriate goodness of fit measure (e.g. the *RSS* in Eq. (1)) is sufficiently small. The RSS values for the best mixtures of one, two, three and four materials cannot be simply compared against each other. However, they can be compared when fitting consecutive "models", for example, models containing M and $M+1$ materials. The best fitting mixture of M materials is called "model M", and RSS_M is denoted the RSS for model M. A model comparison metric is described as:

$$R_{M,M+1} = RSS_M \,/\, RSS_{M+1} \quad (5)$$

M is independent of the scaling used to compute RSS, as used for instance in Eq. (5). Model $M+1$ is preferred to model M if:

$$R_{M,M+1} > r_0 \quad (6)$$

where r_0 is chosen according to some criterion as well as users' requirement [11].

2.3. Optimized quick-shift algorithm for spectral image

Quick shift is a kernelized version of a fast mode seeking segmentation scheme. It initializes a segmentation using a mean shift or medoid shift procedure [14, 15]. It then moves each point in the feature space to the nearest neighbor that increases the Parson Density estimate. It has relatively good boundary adherence and the super-pixels produced are not fixed in approximate size or number. Segmentation with quick shift is controlled by three parameters, namely ratio λ, kernel size σ, and distance τ. For each pixel (x, y), quick shift regards $(x, y, I(x, y))$ as a sample from a $d + 2$ dimensional vector space. It then computes the Parzen density estimate with a Gaussian kernel of standard deviation σ [16]:

$$E(x,y) = P(x,y,I(x,y)) = \sum_{x'y'} \frac{1}{(2\pi\sigma)^{d+2}} \exp\left(-\frac{1}{2\sigma^2} \left\| \begin{matrix} x-x' \\ y-y' \\ I(x,y)-I(x',y') \end{matrix} \right\|_2^2 \right) \quad (7)$$

Then quick-shift constructs a tree connecting each image pixel to its nearest neighbor that has greater density value. Formally, $(x', y') >_p (x, y)$ if and only if $P(x', y', I(x', y')) > P(x, y, I(x, y))$. Each pixel (x, y) is connected to the closest higher density pixel parent (x, y) that achieves the minimum distance in:

$$dist(x,y) = min_{(x',y')>_p(x,y)} (x-x')^2 + (y-y')^2 + \|I(x,y) - I(x',y')\|_2^2 \quad (8)$$

The algorithm computes a forest of pixels whose branches are labeled with a distance value. This specifies a hierarchical segmentation of the image, with segments corresponding to subtrees. When the quick-shift algorithm is applied to color image segmentation [16], the raw RGB values are augmented with the (x, y) positions in the image. So, the feature space is five dimensional. λ is the trade-off between spatial importance (x, y, R, G, B) and color importance (R, G, B), and it balances color-space proximity and image-space proximity, with a value from 0 to 1. A small ratio gives more importance to the spatial component and higher values give more weight to color-space. In the segmentation of multi-spectral and high-spectral images, we consider different weights for different spectral wavelengths in feature space:

$$E(x,y) = \sum_{x'y'} \frac{1}{(2\pi\sigma)^{d+2}} exp\left(-\frac{1}{2\sigma^2} \left\| \begin{matrix} x-x' \\ y-y' \\ W\left(I(x,y)-I(x',y')\right) \end{matrix} \right\|_2^2 \right) \quad (9)$$

This will be useful when we focus on a material of specific frequency in a hyper-spectral image where some bands are reflected well by this material. The distance computation is described by:

$$dist(x,y) = min_{(x',y')>_p(x,y)} (x-x')^2 + (y-y')^2 + \|W(I(x,y) - I(x',y'))\|_2^2 \quad (10)$$

where:

$$W = \begin{bmatrix} w_1 & & & \\ & w_2 & & \\ & & \ddots & \\ & & & w_n \end{bmatrix} \quad (11)$$

The pseudo-code of the quick-shift based segmentation is described in Algorithm 1. The inputs are the segmentation parameters, normalized spectral $\boldsymbol{y_i}$ and their corresponding positions. The outputs are the roots of all input pixels identifying the segments that they are classified to.

Algorithm 1 Quick-shift based segmentation algorithm

Input: y_i, W, H
Input: segmentation parameters λ, σ, τ
Output: $root(j1, j2)), rootArray$
foreach $i1$ in W **do**
  foreach $i2$ in H **do**
    computation for density $E(i1, i2)$
    foreach $j1$ in τ **do**
      foreach $j2$ in τ **do**
        if $E(j1, j2) > E(i1, i2)$ **then**
          computation for dist$(j1, j2)$
          if dist$(j1, j2) < \tau^2$ **and**
            dist$(j1, j2) < dist_{best}(i1, i2)$ **then**
              $dist_{best}(i1, i2) =$ dist$(j1, j2)$
              $root(j1, j2) = j1 + W * j2$
          end if
        end if
      end loop
    end loop
    $root(i1, i2) = root(j1, j2)$
    if $root(i1, i2) == (i1, i2)$ **then**
      add $root(i1, i2)$ into $rootArray$
    end if *// identify roots pixel*
  end loop
end loop

Given a spectral image of size 512x512 with its pseudo-color image shown in Fig. 2, the segmentation results of the spectral image with $\lambda = 0.5$ and $W_n = I_n$ are shown in Fig. 3.

Figure 2: Pseudo-color image

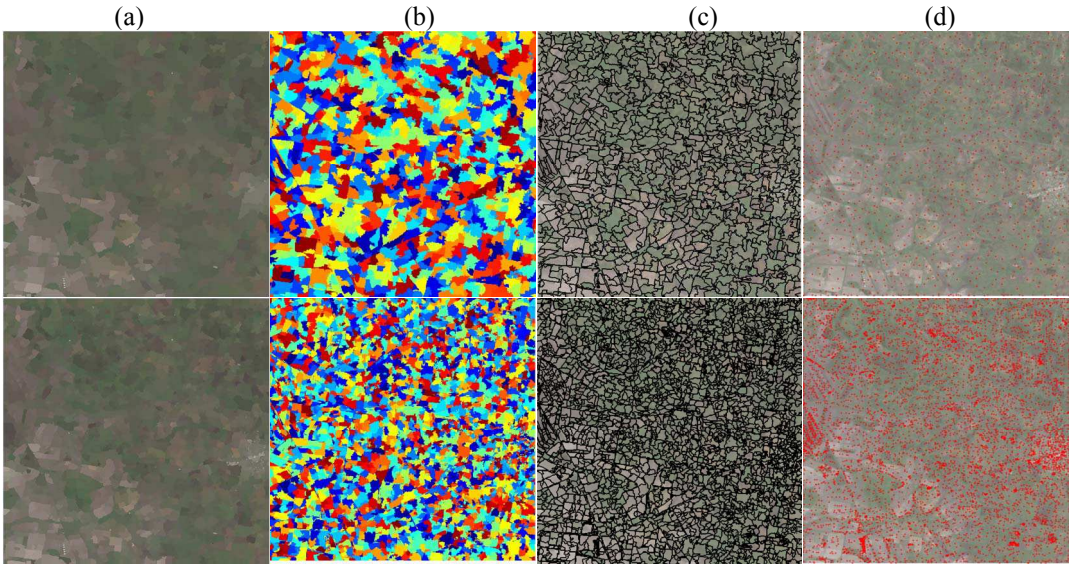


Figure 3: Segmentation for different σ and τ. (a) segmentation on color of pseudo-color image (b) segmentation on normalized image (c) boundaries of segmentation (d) roots pixels of spectra segmentation. σ=1, τ=10 in the first row and σ=3, τ=3 in the second.

2.4. Comparison with original brute-force algorithm

In the original brute-force algorithm for VSS based spectral unmixing procedures [10, 11], for every pixel, combinations of two, three and four materials from the libraries are all tested for smallest RSS. It has not been widely used in real-time unmixing applications due to its significant computational workload. In an image, given i is the index of pixels, $y_i = \{y_{i1}, y_{i2} \dots, y_{iN}\}^T$ represents mixed pixel vectors, G is the number of items of covariance matrix μ_c in the library, M is the number of pure endmembers ($M \ll G$). The output is a covariance matrix of pure endmembers X_i, $X_i = \{\mu_{i1}, \mu_{j2} \dots, \mu_{iM}\}$ for every pixel and their relative proportion vector β_i=$\{\beta_{i1}, \beta_{i2}, \dots \beta_{iM}\}$. The pseudo-code of the Brute-Force algorithm is described in Algorithm 2.

Algorithm 2 Original brute-force algorithm

Input: y_i, μ_j, G, M, W, H
Output: $X_i, \beta_i,\ RSS_i$
foreach i in $W * H$ **do**
  foreach c in $\binom{G}{M}$ **do**
    composite $M * D\ matirx\ X_c$ from μ_j in library
    if meetConstrains($\beta_{ic} = (X_c^T X_c)^{-1} X_c^T y_{ic}$) **then**
      $RSS_{ic} = y_i{}^T y_i - y_i{}^T X_c (X_c^T X_c)^{-1} X_c^T y_i$
      if $RSS_{ic} < RSS_{imin}$ **then**
        $RSS_{imin} = RSS_{ic}$
        $\beta_i = (X_c^T X_c)^{-1} X_c^T y_i$
        $X_i = X_c$
      end if
    end if
  end loop
  output $X_i, \beta_i,\ RSS_i$
end loop

In the optimized approach, the M combinations of materials from the library are computed only once and read by subsequent matrix operations. After spectral segmentation, only the roots pixels are involved in full unmixing processing for smallest RSS in Eq. (3) and the non-roots pixels undergo estimation of pure spectral proportions only in Eq. (2). The pseudo-code of the optimized fast unmixing algorithm is described in Algorithm 3. Constraints in Section 2.2 for β_i are used in selection of the smallest RSS.

Algorithm 3 Fast unmixing algorithm

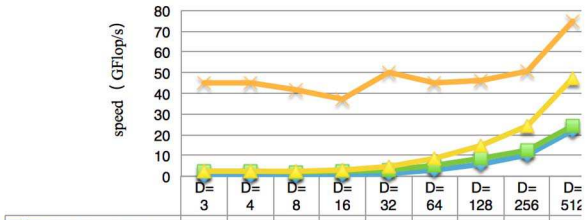
Input: $y_i, \mu_j, G, M, W, H, rootArray$
Output: $X_i, \beta_i,\ RSS_i$
foreach c in $\binom{G}{M}$ **do**
  $A_c = (X_c^T X_c)^{-1} X_c^T$
  $M_c = X_c A_c$ *// compute coefficient matrix*
foreach r in rootsArray **do**
  foreach c in $\binom{G}{M}$ **do**
    if meetConstrains($\beta_{ic} = (X_c^T X_c)^{-1} X_c^T y_{ic}$) **then**
      $RSS_{rc} = y_r{}^T y_r - y_r{}^T M_c y_r$
      if $RSS_{rc} < RSS_{rmin}$ **then**
        $RSS_{rmin} = RSS_{rc}$
        $\beta_r = \beta_{rc}$
        $X_r = X_{rc}$ *// best combination for roots*
      end if
    end if
  end loop
end loop
foreach i in $W * H$ **do**
  $X_i = X_{ri}$
  $\beta_i = (X_{ri}^T X_{ri})^{-1} X_{ri}^T y_i$
  $RSS_i = y_i{}^T y_i - y_i{}^T M_{ri} y_i$ *// estimate proportions*
  output $X_i, \beta_i,\ RSS_i$
end loop

3. Parallelization Strategies on GPU

Parallelization strategies are the key to performance speedup in unmixing processing on GPUs. In the preprocessing stage of segmentation, quick shift operates on each pixel of an image, and the computations of each pixel are independent of its surrounding pixels. This is a good candidate for implementation on the parallel architecture of GPU [14]. For distance computation within a $\tau \times \tau$ window per pixel, one approach is the sequential computing of a two-layer traversal of $i1$, $i2$ for the whole window in Algorithm 1. As a result, pixel-level parallelization is performed. Another solution with higher parallelism is achieved by computing distance in one GPU thread and the intermediated data generated by threads for best distance is communicated through shared memory in thread blocks [17]. In this case, the loop of $j1$ and $j2$ is also parallelized with a higher degree of parallelism compared with the first parallelization strategy. However, these two solutions at segmentation stage do not make much difference to the overall performance, as the performance bottleneck of the whole pipeline lies in selection of the best endmember combination of root pixels, with massive matrix multiplication workload.

It has been shown in many experimental tests that parallelism at matrix multiplication level is much more efficient than single thread computing on CPUs [17, 18, 19]. Typically, when a kernel is launched, two-dimensional grids of two-dimensional thread blocks are created and queued for multiplication and addition from row entries and column entries simultaneously, which can be executed either by accessing global memory alone or shared memory as well [17].

However, in this research, we found that peak performance is not always reached by parallelism at matrix multiplication level, due to the specialty of matrix size of hyper-spectral data and unmixing algorithms. In a DxM matrix involved in unmixing computation, M is the number of endmembers and limited to four, as the case of more than four pure endmembers in one pixel is seldom considered. D is the spectral band, valued from three to hundreds for multi-spectral and hyper-spectral imagery. The benchmark was carried out on NVIDIA GeForce GT750M with 384 CUDA cores, NVIDIA GTX TITAN X with 3072 CUDA cores, and the CPU of 2.5 GHz Intel Core i7. The performance results of computing $\boldsymbol{X_S^T X_S}$ in Eq. (2) or Eq. (3) at pixel-level parallelism and matrix-level parallelism are shown in Fig. 4. They show that matrix-level parallelism at smaller bands is not as efficient as pixel-level parallelism, as the result of low occupancy of GPU hardware resources caused by insufficient number of threads and expensive synchronization. However, with increasing number of bands, the performance of matrix-level parallelism increases dramatically and significantly exceeds that of



	D= 3	D= 4	D= 8	D= 16	D= 32	D= 64	D= 128	D= 256	D= 512
Titan X pixel-level parallelism	42.46	42.52	39.61	34.14	45.56	36.9	31.21	26.37	27.3
Titan X matrix-level parallelism	0.15	0.19	0.39	0.78	1.55	3.1	6.08	11.7	23.3
GT750M pixel-level parallelism	2.18	2.2	1.62	1.5	1.79	2.57	2.96	2.52	2.1
GT750M matrix-level parallelism	0.13	0.17	0.33	0.68	1.38	2.75	5.71	10.14	22

Figure 4: Performance of matrix multiplication on bands D for combination number M=4 on GeForce GT750M and GTX TITAN X

pixel-level parallelism. Therefore, the best parallelization strategy is dependent on the spectral data layout and GPU hardware features.

All the covariance matrices X_S for $O(DM^2\binom{G}{M})$ combinations are loaded in global memory, and each thread is launched to perform the numerical solution to Eq. (3), with GPU shared memory used for intermediate data communication for the smallest RSS selection. Root pixels are processed sequentially due to the large number of linear combinations X_S. For the last stage of proportions estimation in Eq. (2), parallelization is performed at per-pixel level, taking full advantage of the massive parallel computing power of GPUs.

4. Evaluations

The benchmark tests were carried out on a set of satellite images on GPU devices. Given the uncertainty of real endmember combinations and relative proportions of real spectral data, we propose two approaches to evaluate the fast unmixing algorithm.

The first approach is to compare the estimated RSS to all the RSS of M combinations of library templates to find the combinations that are better than the estimated combination by minimizing Eq. (3). Experimental test results in Fig. 5 are the empirical cumulative probability function (CDF) of RSS ratios and the log likelihood of the number of endmember combinations that are better than the approximate one, with typical parameters $M = 2, 3$, $\tau = 3, 4,$ and σ is sampled from 0.4 to 5.

An alternative approach to evaluate precision is to compare the hit rate of endmembers by the fast approximating algorithm and the original best-fitting algorithm, and the results are shown in Table 1.

Normally more segments will result in the lower performance speedup but better hit ratio. However, with more homogeneous regions and higher resolution of satellite imagery, the approach works well for both performance speedup and accuracy estimation.

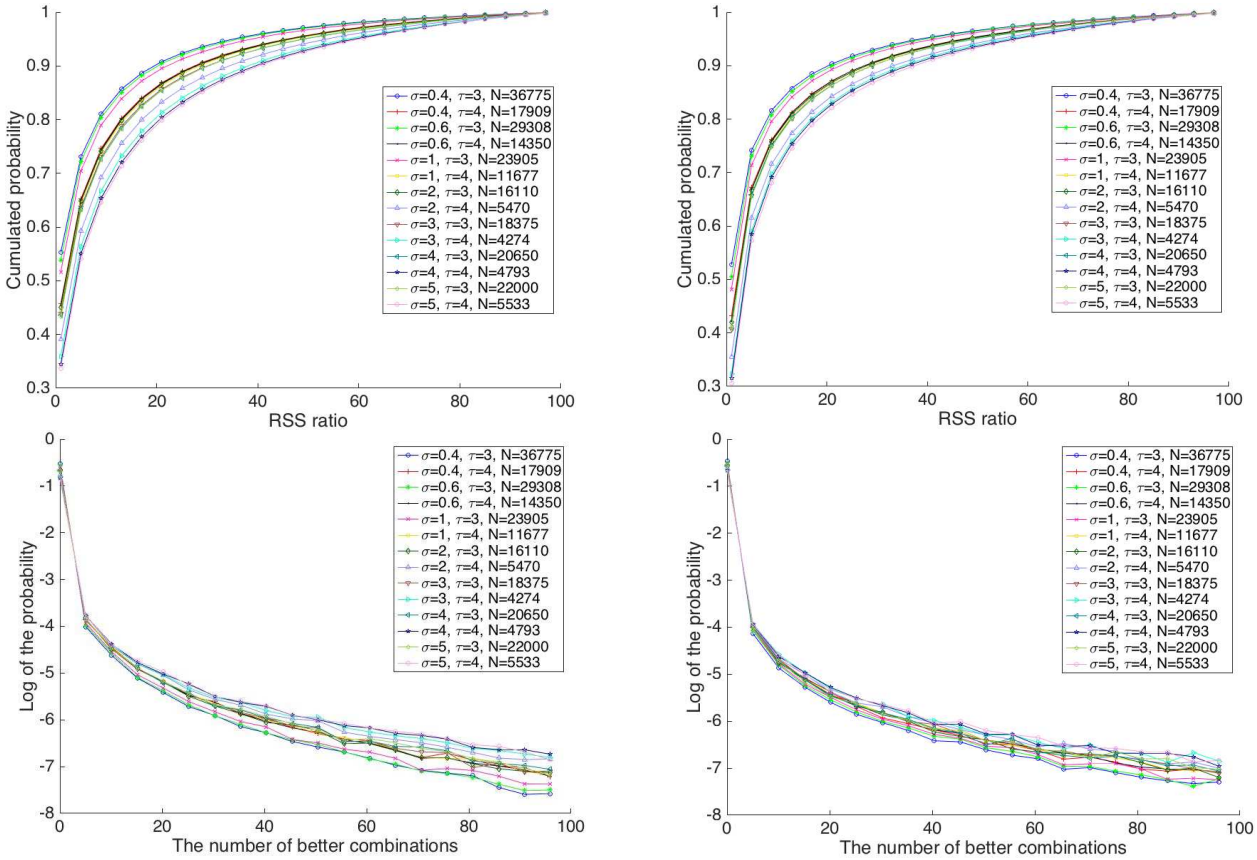


Figure 5: Empirical CDF of RSS ratio (upper row) and the log likelihood of the number of endmember combinations better than the estimated combination (bottom row) of *M*=2 (left column) and *M*=3(right column).

Table 1: Materials hit ratios and speedup with fast approximate unmixing algorithm

	hit ratio	τ=3, σ=0.4	τ=3, σ=0.6	τ=3, σ=1.0	τ=3, σ=2.0	τ=3, σ=3	τ=3, σ=4	τ=3, σ=5	τ=4, σ=0.4	τ=4, σ=0.6	τ=4, σ=1.0	τ=4, σ=2.0	τ=4, σ=3	τ=4, σ=4	τ=4, σ=5
M=2	hit 0	25.5%	31.2%	26.3%	31.5%	27.6%	32.0%	31.4%	35.4%	32.5%	37.5%	32.5%	38.7%	32.3%	38.8%
	hit 1	31.6%	35.6%	32.5%	35.7%	33.6%	36.3%	36.3%	38.1%	36.3%	38.9%	36.6%	39.3%	37.0%	39.7%
	hit 2	42.9%	33.2%	41.1%	32.8%	38.9%	31.8%	32.3%	26.4%	31.2%	23.5%	30.9%	22.0%	30.7%	21.5%
M=3	hit 0	13.1%	16.1%	13.6%	16.4%	14.3%	16.8%	16.2%	18.7%	17.0%	20.3%	16.9%	20.7%	17.4%	21.5%
	hit 1	30.4%	34.9%	31.5%	35.3%	32.5%	35.7%	35.4%	38.1%	36.0%	39.3%	35.9%	39.8%	35.9%	40.2%
	hit 2	26.7%	28.2%	27.6%	28.5%	28.2%	28.8%	28.8%	29.1%	28.0%	28.4%	27.9%	27.9%	27.4%	27.3%
	hit 3	29.7%	20.8%	27.4%	19.8%	25.1%	18.7%	19.6%	14.2%	19.0%	12.0%	19.2%	11.6%	19.4%	11.0%
speedup		7.1	8.9	11.0	16.3	14.3	12.7	11.9	14.6	18.3	22.4	47.9	61.3	54.7	47.4

Nevertheless, the linear models do not easily lead to exact accurate solutions, along with noise variance in spectral data. As a result, the segmentation approach provides a good solution for spatial smoothness as well.

5. Conclusion

We have presented a fast approximate unmixing framework for endmember selection and proportion estimation for multi-spectral and hyper-spectral satellite imagery. The approach provides significant performance acceleration with the quick-shift based segmentation algorithm. It also improves spatial smoothness and eliminates the potential "roughness" problem among adjacent pixels.

The GPU-based parallelization strategies are also analyzed with implementations for further performance acceleration.

For evaluation, the linear regression model of subset selection was used as a base comparison with good approximation and performance speedup. This framework can be implemented for other unmixing algorithms as well.

References

[1] P. Ksieniewicz, D. Jankowski, B. Ayerdi, K. Jackowski, M. Grana, and M. Wozniak. A novel hyperspectral segmentation algorithm—concept and evaluation. Logic Journal Of The Igpl, 2015 Feb, Vol.23(1), pp.105-120.

[2] N. Keshava and J. F. Mustard, Spectral unmixing. IEEE Signal Processing Magazine, vol. 19, no. 1, pp. 44-57, Jan 2002.

[3] N. Keshava, Performance comparisons for spectral unmixing algorithms. Proceedings of SPIE. The International Society for Optical Engineering, 2002, Vol.4480, pp.40-48.

[4] G. M. Foody. Cross-entropy for the evaluation of the accuracy of a fuzzy land cover classification with fuzzy ground data. ISPRS Journal of Photogrammetry and Remote Sensing, 1995, Vol.50(5), pp.2-12.

[5] G. M. Foody. 1996, Approaches for the production and evaluation of fuzzy land cover classifications from remotely sensed data. International Journal of Remote Sensing. 1996, Vol.17(7), pp.1317-1340.

[6] G. M. Foody and D. P. Cox. Sub-Pixel land cover composition estimation using a linear mixture model and fuzzy membership model and fuzzy membership functions. International Journal of Remote Sensing. 1994, Vol.15(3), p.619-631.

[7] G. M. Foody, R. M. Lucas, P. J. Curran and M. Honzak. Non-linear mixture modeling without end-members using an artificial neural network. International Journal of Remote Sensing, 1997, Vol.18(4), p.937-953.

[8] B. TSO, B. and P. M Mather. Classification Methods for Remotely Sensed Data (London and New York: Taylor and Francis), 2001.

[9] C. Quintano, A. Fernández-Manso, Y. E. Shimabukuro and G. Pereira. Spectral unmixing. International Journal of Remote Sensing, Vol.33(17), p.5307-5340

[10] Y. Guo, M. Berman and J. Gao. Group subset selection for linear regression. Computational Statistics and Data Analysis. July 2014, Vol.75, pp.39-52

[11] M. Berman, L. Bischof, R. Lagerstron, Y. Guo, J. Huntington and P. Mason. An unmixing algorithm based on a large library of shortwave infrared spectra, in: Technical Report EP117468, CSIRO Mathematics, Informatics and Statistics, 2011.

[12] B. Luo and J. Chanussot. Supervised Hyperspectral Image Classification Based on Spectral Unmixing and Geometrical Features. Journal of Signal Processing Systems, 65, 457–468, 2011.

[13] T. Akgun, Y. Altunbasak, and R. M. Mersereau. Super-resolution reconstruction of hyperspectral images. IEEE Trans. Image Process. vol. 14, no. 11, pp. 1860–1875, Nov. 2005.

[14] B. Fulkerson and S. Soatto. Really quick shift: Image segmentation on a GPU. Lecture Notes in Computer Science. 2012, Vol.6554 (2), pp.350-358.

[15] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. Lecture Notes in Computer Science. 2008, Vol.5305(4), pp.705-718.

[16] A. Ibrahim, A, M. Salem and H. A. Ali, (2014). Automatic quick-shift segmentation for color images. International Journal of Computer Science Issues (IJCSI), 11(3), 122-127.

[17] NVIDIA. Programming guide :: CUDA Toolkit Documentation, 2017.

[18] S. Sanchez, A. Plaza, B. Huang, A. J. Plaza. A Comparative Analysis of GPU Implementations of Spectral Unmixing Algorithms. High-Performance Computing in Remote Sensing, 2011, Vol.8183(1), pp.81830E-81830E-10.

[19] R. Hochberg. Matrix Multiplication with CUDA — A basic introduction to the CUDA programming model. SHODOR technical document. Aug 2012.