Parallel Implementation of the Multiple Endmember Spectral Mixture Analysis Algorithm for Hyperspectral Unmixing

Sergio Bernabe[1], Francisco D. Igual[1], Guillermo Botella[1], Manuel Prieto-Matias[1] and Antonio Plaza[2]

[1]Complutense University, Madrid, Spain
[2]Hyperspectral Computing Laboratory, University of Extremadura, Caceres, Spain

ABSTRACT

In the last decade, the issue of endmember variability has received considerable attention, particularly when each pixel is modeled as a linear combination of endmembers or pure materials. As a result, several models and algorithms have been developed for considering the effect of endmember variability in spectral unmixing and possibly include multiple endmembers in the spectral unmixing stage. One of the most popular approach for this purpose is the multiple endmember spectral mixture analysis (MESMA) algorithm. The procedure executed by MESMA can be summarized as follows: (i) First, a standard linear spectral unmixing (LSU) or fully constrained linear spectral unmixing (FCLSU) algorithm is run in an iterative fashion; (ii) Then, we use different endmember combinations, randomly selected from a spectral library, to decompose each mixed pixel; (iii) Finally, the model with the best fit, i.e., with the lowest root mean square error (RMSE) in the reconstruction of the original pixel, is adopted. However, this procedure can be computationally very expensive due to the fact that several endmember combinations need to be tested and several abundance estimation steps need to be conducted, a fact that compromises the use of MESMA in applications under real-time constraints. In this paper we develop (for the first time in the literature) an efficient implementation of MESMA on different platforms using OpenCL, an open standard for parallel programing on heterogeneous systems. Our experiments have been conducted using a simulated data set and the clMAGMA mathematical library. This kind of implementations with the same descriptive language on different architectures are very important in order to actually calibrate the possibility of using heterogeneous platforms for efficient hyperspectral imaging processing in real remote sensing missions.

Keywords: Hyperspectral imaging, endmember variability, spectral mixture analysis (SMA), multiple endmember spectral mixture analysis (MESMA), high performance computing (HPC), OpenCL

1. INTRODUCTION

Spectral unmixing[1] is an important technique for remotely sensed hyperspectral data exploitation and an active field of research. It amounts at identifying pure spectral components (called endmembers) and their abundance fractions in each (possibly mixed) pixel of the scene. Popular approaches for this purpose in the literature have been a linear (LMM) or nonlinear mixture model (NLMM).[2] The LMM assumes that the endmembers substances are sitting side-by-side within the field of view of the imaging instrument. On the other hand, the NLMM assumes nonlinear interactions between endmember sustances. In practice, the LMM is more flexible and can be easily adapted to different analysis scenarios. However, the lack of ability to account for temporal and spatial variability between and among endmembers has been acknowledged as a major shortcoming of LMM with fixed endmembers.[3]

In recent years, several models and algorithms have been developed for considering the effect of endmember variability in spectral unmixing,[3] including the multiple endmember spectral mixture analysis (MESMA),[4] Monte Carlo spectral unmixing model (AutoMCU),[5] endmember bundles,[6] or Bayesian spectral mixture analysis (BSMA),[7] among several others. Theses algorithms can be computationally very expensive due to the extremely large volumes of data collected by imaging spectrometers, a fact that compromises their use in applications under real-time constraints, where the acceleration of this algorithms is very important. Until now, there has been no effort to accelerate endmember variability algorithms for hyperspectral images using parallel techniques in the literature.

High-Performance Computing in Remote Sensing V, edited by Bormin Huang, Sebastián López, Zhensen Wu, Jose M. Nascimento, Boris A. Alpatov, Jordi Portell de Mora, Proc. of SPIE Vol. 9646, 96460J
 · doi: 10.1117/12.2195120

In this paper, we focus on this particular problem and develop (for the first time in the literature) an efficient implementation of the MESMA algorithm on different platforms using OpenCL. Our proposed implementation of MESMA significantly reduces its computational cost. The experimental results have been conducted using a simulated data set. This kind of implementations with the same descriptive language on different architectures are very important in order to actually calibrate the possibility of using heterogeneous platforms for efficient hyperspectral imaging processing in real remote sensing missions.

The remainder of this paper is organized as follows. Section 2 describes the MESMA method. Section 3 describes the proposed parallel implementation. Section 4 presents an experimental evaluation of the proposed implementation in terms of both accuracy and parallel performance using a simulated data set on different heterogeneous platforms. Finally, Section 5 presents as few concluding remarks and pointers to future work.

2. THE MESMA ALGORITHM

Algorithm 1 Pseudocode of MESMA Algorithm

```
INPUTS:
% x is L×ns matrix with the hyperspectral data set, where L is the number of spectral bands and ns is the number of pixels
% Uscene is L×S matrix library containing S spectra
% Ulabelscene contain the endmember class label (i) for each spectrum S in the library
% UnmixingAlg estimate fractions using the least squares unmixing (LSU) algorithm
% nit number of iterations. At each iteration a given number of endmembers (MaxMixtureSize) are randomly selected
% MaxMixtureSize eg. 2 for testing binary mixtures, 3 for binary+ternary, etc. Only the best result (lowest RMSE) for each pixel will be retained
p ← Number of endmembers obtained using Ulabelscene
for nComp = 2 to MaxMixtureSize do
   for i = 1 to nit do
      for e = 1 to p do
         spectra ← random(position)
         U[e] ← Uscene[spectra]
      end for
      x_rec ← 0*x
      % LSU Unmixing
      Abundances ← pinv(U)*x
      x_rec ← U*Abundances
      RMSE ← sqrt( sum_{i=1}^{ns} (x - x_rec)^2 / L )
      for k = 1 to ns do
         if RMSE[k] < RMSE_curr[k] then
            RMSE_curr[k] ← RMSE[k]
            for j = 1 to nComp do
               Fractions[j+k*Comp] ← Abundances[j+k*nComp]
            end for
         end if
      end for
   end for
end for
OUTPUT: Fractions
% p×ns matrix with the estimated fractions
```

The original MESMA method for hyperspectral analysis was developed in.[4] This approach allows endmembers to vary on a per-pixel basis and thereby allows to cope with the effects of intra- and inter-class endmember variability in spectral unmixing. The procedure executed by MESMA can be summarized as follows: (i) First, a standard linear spectral unmixing (LSU) or fully constrained linear spectral unmixing (FCLSU) algorithm is run in an iterative fashion; (ii) Then, we use different endmember combinations, randomly selected from a spectral library, to decompose each mixed pixel; (iii) Finally, the model with the best fit, i.e., with the lowest

root mean square error (RMSE) in the reconstruction of the original pixel, is adopted. MESMA requires as input for each ground component or endmember an extensive library of field, laboratory, and/or image spectra. The pseudocode is represented in Algorithm 1, where the LSU algorithm has been used.

3. OPENCL FRAMEWORK AND IMPLEMENTATION

In this section the OpenCL framework is first discussed, with particular emphasis on how it can be used to effectively speed up the MESMA algorithm on heterogeneous systems. Most importantly, a OpenCL library to calculate matrix multiplications and singular value descompositions (SVD) needs to be found. One of our goals is to use a common code written in OpenCL for the different platforms.

3.1 Framework

OpenCL is an open and royalty-free standard based on C99 that allows the execution of parallel programs on heterogeneous platforms. It is currently supported by several hardware devices, such as CPUs, GPUs, DSPs, FPGAs and other processors. OpenCL is based on a host-device model, where the host is in charge of device memory management, data transfer from/to device and kernel code configuration and invocation.

The kernel is a piece of code which expresses the parallelism of a program. The OpenCL programming model divides a program workload into *work-groups* and *work-items*. *Work-items* are grouped into a *work-group*, which is executed independently with respect to other *work-groups*. Data-level parallelism is regularly exploited in an SIMD way, in which several *work-items* are grouped according to the lane width capabilities of the target device.

The OpenCL memory model distinguishes different memory regions that are characterized by the access type, performance and scope. Global memory is read-write accessible by all *work-items* across all *work-groups*, and it usually corresponds to the DRAM memory device, which carries a high latency memory access. Local memory is a shared read-write memory accessible from all work-items of a single *work-group*, and it habitually involves a low latency memory access. Constant memory is a read-only memory that is visible to all *work-items* across all *work-groups*, and private memory, as the name suggests, is only accessible by a single *work-item*.

3.2 Parallel implementation

By observing the program flow in Algorithm 1, it is possible to identify the main potential bottlenecks in the MESMA algorithm. In this case, we have different matrix multiplications and a SVD operation (Lines 10 and 11). For this purpose, we have selected the clMAGMA library,[8] an OpenCL port of MAGMA. MAGMA* is a project to create a new generation of linear algebra libraries that achieves the fastest possible time to an accurate solution on heterogeneous architectures.

Once the hyperspectral image $\mathbf{x}$ is mapped onto the device global memory, the algorithm starts with an initization step. The clMAGMA library is first initialized and the load image and allocates of memory are included. In this part, all the input parameters are configured. Lines 5 to 8 in Algorithm 1 are performed in the CPU. Then, the singular value decomposition (SVD) is performed for computing the pseudocinverse of $\mathbf{U}$ using the *magma_sgesvd* function. Moreover, two matrix multiplications are performed to obtain the pseudoinverse using the *magma_sgemm* function.

Line 10 in Algorithm 1 is carried out with a matrix multiplication using the *magma_sgemm* function between the pseudoinverse obtained and the data set, where the map abundances are achieved. Then, another matrix multiplication between the abundance maps and the matrix $\mathbf{U}$ is needed to obtain the reconstructed data set. Finally, lines between 12 and 20 are performed in the CPU. Experimental tests have shown it using the synthetic data set for this work.

*https://icl.cs.utk.edu/projectsfiles/magma/doxygen

Table 1. OpenCL feature

Device Type	Model	#Cores	Clock Freq.	Global Mem. Size	Local Mem. Size	Max *work-items*
CPU	two Intel Xeon E5-2670	16	2.60 GHz	64 GB	32 KB	8192×8192×8192
GPU	NVIDIA-K20c	2496	706 MHz	5 GB	48 KB	1024×1024×64

4. EXPERIMENTAL RESULTS

4.1 Data Set Description

The synthetic data set used in this experiment was generated randomly with MATLAB Tool. The simulated image consists of 100 × 100 pixels and 3000 spectral bands for a total size of 114 MB. On the other hand, the Uscene and Ulabelscene parameters were generated randomly with MATLAB using p=3, $MaxMixtureSize$=3, S=30, L=3000 and ns=10000.

4.2 Work Environment

To carry out the tests on OpenCL, we have used a multicore heterogeneous system equipped with: two Intel Xeon E5-2670 processor with 8 cores each, at 2.60 GHz and 64 GB of DDR3 RAM memory and an NVIDIA K-20c GPU with features 2496 cores operating at 706 MHz and dedicated memory of 5 GB. The Table 1 summarized the main features of the system used for the comparison.

4.3 Accuracy Evaluation

The accuracy of the proposed parallel MESMA provides exactly the same results as the serial approach of the algorithm, implemented using C/C++ and the gcc (gnu compiler default) with optimization flags `-O3` to exploit data locality and avoid redundant computations. The results of spectral unmixing can also be evaluated in terms of the quality of the reconstruction of the original data set using the extracted endmembers, the estimated fractional abundances, and the linear mixture model. In this case, the metric in employed to evaluate the goodness of the reconstruction is the root mean square error (RMSE) obtained after comparing the original scene with the reconstructed one. This metric showed in Algorithm 1 (Line 12) is based on the assumption that a set of high-quality endmembers (and their corresponding estimated abundance fractions) may allow reconstruction of the original scene with higher precision compared to a set of low-quality endmembers. In this case, the original scene is used as a reference to measure the fidelity of the reconstructed version on a per-pixel basis. For illustrative purposes, Fig. 1 represents the per-pixel RMSE obtained in the reconstruction process depending on the number of iterations applied on the MESMA algorithm. In any event, the per-pixel RMSE values are quite low, indicating a good overall compromise in the reconstruction of the scene.

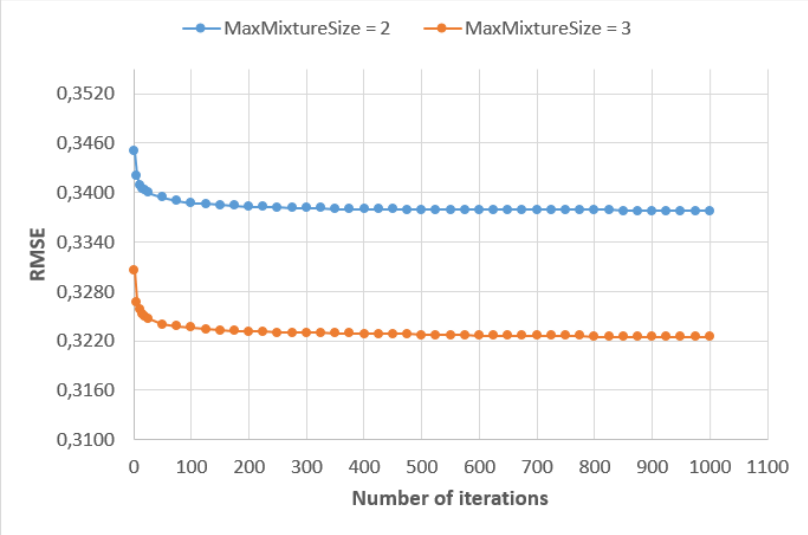

Figure 1. RMSE values between the original and the reconstructed synthetic data set for different number of iterations.

Table 2. Processing times (in seconds) and speedups achieved for the proposed OpenCL implementation in different platforms and tested with synthetic data set during 500 iterations.

	Serial CPU	**CPU Xeon**	**GPU K20c**
Initialization	0.1621	0.4652	0.6313
Lines 5-8	0.0025 (0.00%)	0.0022 (0.00%)	0.0016 (0.00%)
Line 10	121.9969 (55.58%)	89.7973 (56.54%)	66.2758 (49.95%)
Line 11	66.5126 (30.31%)	33.6112 (21.19%)	35.4218 (26.69%)
Lines 12-20	31.0608 (14.16%)	35.2841 (22.17%)	30.9815 (23.36%)
Total Time	219.5728	158.6948	132.6807
Speedup	–	1.38x	1.65x

4.4 Performance Evaluation

We conduct next an experimental evaluation of the computational performance of our proposed parallel MESMA implementation. For this purpose, we evaluate whether a single code written in OpenCL allows us to achieve acceptable performance across all of them. Before describing the parallel algorithm performance results, it is important to emphasize that the parallel version provide exactly the same results as the serial versions of the implemented algorithms, using the g++-4.4.7 (gnu compiler default) with optimization flags `-O3` (for the single-core version) to exploit data locality and avoid redundant computations. Note that for the serial implementation, we used the BLAS and LAPACK implementations from Netlib, version 3.5.0. Hence, the only difference between the serial and parallel versions is the time they need to complete their calculations. For each experiment, ten runs were performed and the mean values were reported (these times were always very similar, with differences on the order of a few milliseconds).

Table 2 shows partial times, where *Initialization* includes the load image and allocates of memory, *lines 5-8* correspond to the selection of endmembers randomly in CPU, *line 10* denotes the pseudoinverse of U matrix and the matrix multiplication between U and the data set using the clMAGMA library in device. *Line 11* includes the matrix multiplication and the transfer memory of *Abundances* matrix between device and host. Finally, *lines 12-20* correspond to obtain the RMSE value performed in CPU.

In our experiments, the scene could be processed with significant speedup factor using the GPU device, up to 1.65 times, including both the processing time and the time spent on host/device memory transfers and compared with a optimized serial version using mathematical libraries. As can be seen, clMAGMA library is a good choice when OpenCL is used.

5. CONCLUSION AND FUTURE LINES

In this work, a parallel implementation of the multiple endmember spectral mixture analysis (MESMA) algorithm is presented for the first time in the literature. The use of OpenCL introduces a level of freedom in the adoption of different high-performance computing solutions on heterogeneous systems. Our experimental results, demonstrate that our single code written in OpenCL and using the clMAGMA library allows us to achieve acceptable speedup rates across all the heterogeneous systems tested improving performance against the BLAS and LAPACK libraries. As future work, we will study the implementation on FPGA using additional real hyperspectral scenes and we will also continue working on the method presented in this paper in order to reduce processing times. Another future remark will be to assess the gap between a portable OpenCL code and the hand-tuned FPGA implementation, in order to have the possiblity to be applied onboard from a control station on Earth.

6. ACKNOWLEDGEMENT

This work has been supported by the Spanish Ministry of Economy and Competitiveness (MINECO) through the research contract TIN 2012-32180 and the Formación Posdoctoral programme (FPDI-2013-16280).

REFERENCES

1. J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing: geometrical, statistical, and sparse regression-based approaches," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **5**(2), pp. 354–379, 2012.
2. N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE Signal Processing Magazine* **19**(1), pp. 44–57, 2002.
3. B. Somers, G. P. Asner, L. Tits, and P. Coppin, "Endmember variability in spectral mixture analysis: A review," *Remote Sensing Environment* **115**(7), pp. 1603–1616, 2011.
4. D. Roberts, M. Gardner, R. Church, S. Ustin, G. Scheer, and R. Green, "Mapping chaparral in the santa monica mountains using multiple endmember spectral mixture models," *Remote Sensing Environment* **65**(3), pp. 267–279, 1998.
5. G. P. Asner and D. B. Lobell, "A biogeophysical approach for automated swir unmixing of soils and vegetation," *Remote Sensing Environment* **74**(1), pp. 99–112, 2000.
6. B. Somers, M. Zortea, A. Plaza, and G. P. Asner, "Automated extraction of image-based endmember bundles for improved spectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **5**(2), pp. 396–408, 2012.
7. C. Song, "Spectral mixture analysis for subpixel vegetation fractions in the urban environment: How to incorporate endmember variability?," *Remote Sensing Environment* **95**(2), pp. 248–263, 2005.
8. C. Cao, J. Dongarra, P. Du, M. Gates, L. Piotr, and S. Tomov, "clmagma: high performance dense linear algebra with opencl," *Proceedings of the International Workshop on OpenCL (IWOCL)* , pp. 1–9, 2014.