

APPROVAL SHEET

Title of Dissertation: Growing Simplex Volume Analysis for Finding Endmembers  
in Hyperspectral Imagery

Name of Candidate: Hsiao-Chi Li  
Doctor of Philosophy, 2016

Dissertation and Abstract Approved: Chen-I Chang  
Dr. Chein-I Chang  
Professor  
Department Computer Science and Electrical  
Engineering

Date Approved: 4/27/16

## ABSTRACT

Title of Document: GROWING SIMPLEX VOLUME ANALYSIS  
FOR FINDING ENDMEMBERS IN  
HYPERSPSPECTRAL IMAGERY

Hsiao-Chi Li, Doctor of Philosophy, 2016

Directed By: Dr. Chein-I Chang  
Professor  
Department of Computer Science and Electrical  
Engineering

Finding endmembers is a fundamental task in hyperspectral data exploitation. Many Endmember Finding Algorithms (EFAs) have been proposed over the past years. Among all the algorithms, using maximal Simplex Volume (SV) as an optimal criterion for finding endmembers has been a major approach, which results in a well-known algorithm, N-finder algorithm (N-FINDR). As an alternative to N-FINDR, Simplex Growing Algorithm (SGA) was further proposed to reduce computational complexity to avoid an exhaustive search for endmembers required by N-FINDR. Nevertheless, both N-FINDR and SGA still suffer from an issue of numerical instability when it comes to SV calculation via matrix determinant. The research conducted in this dissertation converts Determinant-based SV calculation to Geometric SV (GSV) calculation by taking advantage of geometric structures of simplexes. As a result, there is no longer a need of using matrix determinant to

calculate SV. Instead, GSV calculates the volume of a simplex by multiplying its base and height of a simplex. Many benefits can be gained from GSV, such as (1) no need of dimensionality reduction; (2) avoidance of numerical instability incurred by finding determinants of a non-square matrices; (3) no matrix inverses required; (4) significantly reduced computational complexity and computing cost; (5) easy implementation in hardware design. By virtue of GSV calculation several GSV-based EFAs can be re-derived to replace original Determinant SGA (DSGA), which include Orthogonal Projection-based SGA (OP-SGA), Geometric SGA (GSGA), and Geometric Convex Cone Volume Analysis (GCCVA). In order to facilitate real-time processing capability, these algorithms are further extended to their respective recursive counterparts which also result in Kalman filter-like EFAs.

GROWING SIMPLEX VOLUME ANALYSIS FOR FINDING ENDMEMBERS IN  
HYPERSPETRAL IMAGERY

By

Hsiao-Chi Li

Dissertation submitted to the Faculty of the Graduate School of the  
University of Maryland, Baltimore County, in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
2016

© Copyright by  
Hsiao-Chi Li  
2016



## Acknowledgement

Firstly, I would like to express my sincere gratitude to my advisor Professor Chein-I Chang for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. His advice on both research as well as in life have been priceless. Without his guidance, it would never be easy to accomplish this dissertation.

Besides my advisor, I would like to thank Professor Kostantinos Kalpakis, Professor Jian Chen, Professor Seung-Jun Kim, Professor Peter Hu, and Dr. Damon Bradley, for their investing time and for providing interesting and valuable feedback. My sincere thanks also goes to Dr. Colin Mackenzie and Professor Peter Hu, who provided me an opportunity to join their research team at University of Maryland Shock Trauma Center. Without the precious support it would not be possible to conduct this research.

I am grateful to Professor Chao-Cheng Wu and Dr. Jiannher Lin for enlightening me the first glance of research. In addition, I thank my fellow labmates, Dr. Meiping Song, Dr. Lin Wang, Cheng Gao, Yao Li, and Li-Chien Lee, for the stimulating discussions and for all the fun we have had in the last three years, as well as my colleagues, Dr. Shiming Yang, Dr. Brandon Bond, Dr. Nehu Parami, Dr. Willam Teeter, George Hagegeorge, in Shock Trauma Center for their advice and encouragement.

Last but not the least, I would like to thank my parents and my sisters for supporting me spiritually throughout writing this thesis and my life in general.

# Table of Contents

Acknowledgement .....	ii
Table of Contents .....	iii
List of Tables .....	vi
List of Figures .....	vii
Chapter 1: INTRODUCTION.....	1
1.1    Hyperspectral Imaging.....	2
1.2    Structure and Organization .....	3
Chapter 2: FINDING ENDMEMBERS: PRELIMINARIES .....	5
2.1    Introduction.....	5
2.2    Endmember Finding Algorithms .....	7
2.3    Simplex Growing Algorithm (SGA).....	10
2.4    Image Datasets .....	13
2.4.1    AVIRIS Cuprite Data.....	13
2.4.2    Hyperspectral Digital Imagery Collection Experiment (HYDICE).....	14
2.4.3    Synthetic Image .....	16
Chapter 3: GROWING SIMPLEX VOLUME ANALYSIS (GSVA) .....	21
3.1    Introduction.....	21
3.2    Criteria for Simplex Volume Calculation .....	22
3.2.1    Geometric Structure Approach .....	22
3.2.2    Eigen-analysis Approach .....	24
3.3    Numerical Results.....	28
3.3.1    Mathematical Examples.....	29
3.3.2    Real Image Experiments .....	30
3.4    Conclusions.....	33
Chapter 4: DESIGN AND DEVELOPMENT OF VARIANTS OF SGA .....	35
4.1    Introduction.....	35
4.2    Gram-Schmidt Orthogonalization Process for Finding Heights .....	37
4.3    Geometric Simplex Growing Algorithm (GSGA).....	38

4.4	Orthogonal Projection Approach to Find Heights .....	39
4.5	Orthogonal Projection-based Simplex Volume Algorithm (OP-SGA) .....	42
4.6	Experiments .....	42
4.6.1	Synthetic Image Experiments .....	43
4.6.2	Real Image Experiments .....	46
4.7	Conclusions.....	58
Chapter 5: RECURSIVE GROWING SIMPLEX VOLUME ANALYSIS (RGSVA)		
.....		59
5.1	Introduction.....	59
5.2	Recursive Equations for GSGA .....	60
5.3	Recursive GSGA.....	60
5.3.1	RGSGA.....	60
5.3.2	KF-OSP-GSGA.....	61
5.3.3	KF-OVP-GSGA .....	63
5.4	Recursive equations for OP-SGA .....	65
5.5	Recursive OP-SGA .....	67
5.6	Determining Number of Endmembers to be generated .....	68
5.7	Real Image Experiments .....	72
5.7.1	VD estimation using real targets.....	72
5.7.2	Computer processing time comparison.....	73
5.8	Conclusions.....	79
Chapter 6: GEOMETRIC CONVEX CONE VOLUME ANALYSIS .....		80
6.1	Introduction.....	80
6.2	Convex Cone Volume Analysis (CCVA).....	81
6.3	Geometric Convex Cone Volume Analysis (GCCVA) .....	85
6.4	Recursive Geometric Convex Cone Volume Algorithm (RGCCVA).....	87
6.5	Experiments .....	88
6.5.1	Synthetic Image Experiments .....	89
6.5.2	Real Image Experiments .....	96
6.6	Conclusions.....	106
Chapter 7: CONCLUSIONS.....		108

7.1	Summary .....	108
7.2	Contributions.....	110
7.3	Future work.....	111
	Bibliography .....	113

## List of Tables

Table 2.1. Mixed pixel panels in 3 <sup>rd</sup> column .....	17
Table 2.2. Subpixel panels in 4 <sup>th</sup> and 5 <sup>th</sup> column.....	17
Table 3.1. A comparative analysis for volume among various methods of simplexes in 3-D space .....	30
Table 3.2. Computational complexity of each SV approach .....	32
Table 4.1. Endmember pixels found by EIA .....	54
Table 4.2. SAM/SID of the closet target pixels with ground-truth by DSGA and OP-SGA on Cuprite reflectance data .....	57
Table 4.3. SAM/SID of the closet target pixels with ground-truth by DSGA and OP-SGA on Cuprite radiance data .....	57
Table 5.1. VD estimated by (5.14) and (5.23) on HYDICE .....	72
Table 5.2. VD estimated for Cuprite reflectance data by (5.14) and (5.23) .....	73
Table 5.3. VD estimated or Cuprite radiance data by (5.14) and (5.23).....	73
Table 5.4. Comparison of computing time in seconds for endmembers found by variants of SGA on HYDICE .....	75
Table 5.5. Comparison of computing time in seconds for endmembers found by variants of SGA on Cuprite data.....	75
Table 6.1. Endmember pixels found by EIA .....	100
Table 6.2. SAM/SID of the closet target pixels with ground-truth by variants of geometric convex cone algorithms on Cuprite reflectance data .....	104
Table 6.3. SAM/SID of the closet target pixels with ground-truth by variants of geometric convex cone algorithms on Cuprite radiance data .....	104

## List of Figures

Figure 1.1. An illustration of a hyperspectral image cube .....	2
Figure 2.1. (a) Cuprite AVIRIS image scene (b) spatial positions of five pure pixels corresponding to minerals: alunite (A), buddingtonite (B), calcite (C), kaolinite (K) and muscovite (M); (c) Five mineral reflectance spectra; (d) Five radiance spectra..	14
Figure 2.2. (a) A HYDICE panel scene which contains 15 panels; (b) Ground truth map of spatial locations of the 15 panels .....	15
Figure 2.3. A set of 25 panels simulated by A, B, C, K, M.....	17
Figure 2.4. Three scenarios designed for target implantation.....	19
Figure 2.5. Three scenarios designed for target embeddedness.....	20
Figure 3.1. (a) 3-dimensional simplex formed by three edge vectors $v_1, v_2, v_3$ (b) 3-dimensional parallelotope (or parallelepiped) formed by the same edge vectors.....	23
Figure 3.2. An interpretation of the volume calculation for 3-dimensional parallelotope.....	24
Figure 3.3. (a) A triangle (b) a regular triangle (c) a 3D-simplex with one vertex in origin; in 3D-space.....	29
Figure 3.4. SV comparison (a) GSV versus PCA-DSV (b) PCA-GSV versus PCA-DSV.....	31
Figure 3.5. Comparisons of accumulated computing time required by GSV and PCA-DSV.....	32
Figure 4.1. Interpretation of finding heights via OP.....	40
Figure 4.2. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TI1 .....	43
Figure 4.3. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TI2.....	44
Figure 4.4. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TI3.....	44
Figure 4.5. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TE1 .....	45

Figure 4.6. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TE2.....	45
Figure 4.7. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TE3.....	46
Figure 4.8. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on HYDICE .....	48
Figure 4.9. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on Cuprite reflectance data .....	51
Figure 4.10. Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on Cuprite radiance data .....	52
Figure 4.11. Endmember pixels found by EIA compared to ground-truth pixels for Cuprite reflectance data .....	53
Figure 4.12. Endmember pixels found by EIA compared to ground-truth pixels for Cuprite radiance data .....	53
Figure 4.13. Comparative plots of spectral signatures found by DSGA and OP-SGA on Cuprite reflectance data .....	55
Figure 4.14. Comparative plots of spectral signatures found by DSGA and OP-SGA on Cuprite radiance data .....	56
Figure 5.1. Accumulative computing time in seconds of DSGA without DR, Dist-SGA, OP-SGA, ROP-SGA, GSGA, RGSGA, KF OSP-GSGA, and KF OVP-GSGA as $p$ increases on HYDICE data.....	74
Figure 5.2. Accumulative computing time in seconds of DSGA without DR, Dist-SGA, OP-SGA, ROP-SGA, GSGA, RGSGA, KF OSP-GSGA, and KF OVP-GSGA as $p$ increases on Cuprite data.....	74
Figure 5.3. Computing times for ATGP, VCA with/without DR, Dist-SGA, OP-SGA, ROP-SGA, GSGA, and RGSGA .....	77
Figure 6.1. A two dimensional bounded convex cone.....	83
Figure 6.2. Illustration of projecting data samples onto selected hyperplane via central projection and OP; where Proj( $\mathbf{r}$ ) indicates the central projection and $P_V^+(\mathbf{r})$ indicates OP, respectively. ....	86

Figure 6.3. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TI1; (f) ATGP (g) DSGA (h) VCA and (i) CCGA .....	90
Figure 6.4. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TI2; (f) ATGP (g) DSGA (h) VCA and (i) CCGA .....	91
Figure 6.5. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TI3; (f) ATGP (g) DSGA (h) VCA and (i) CCGA .....	92
Figure 6.6. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TE1; (f) ATGP (g) DSGA (h) VCA and (i) CCGA .....	93
Figure 6.7. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TE2; (f) ATGP (g) DSGA (h) VCA and (i) CCGA .....	94
Figure 6.8. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TE3; (f) ATGP (g) DSGA (h) VCA and (i) CCGA .....	95
Figure 6.9. Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on HYDICE; (f) ATGP (g) DSGA (h) VCA and (i) CCGA.....	97
Figure 6.10. Endmember pixels found by EIA compared to ground-truth pixels for Cuprite reflectance data; (a-e) GCCVA with data samples orthogonally projected onto different hyperplane, (f) ATGP, (g) DSGA, (h) VCA, and (i) CCGA .....	98
Figure 6.11. Endmember pixels found by EIA compared to ground-truth pixels for Cuprite radiance data; (a-e) GCCVA with data samples orthogonally projected onto different hyperplane, (f) ATGP, (g) DSGA, (h) VCA, and (i) CCGA .....	99
Figure 6.12. Comparative plots of spectral signatures found by GCCVA, ATGP, VCA and CCGA on Cuprite reflectance data .....	101
Figure 6.13. Comparative plots of spectral signatures found by GCCVA, ATGP, VCA and CCGA on Cuprite radiance data .....	102

Figure 6.14. Accumulative computing time in seconds of ATGP, CCGA with SV  
calculated via SVD, VCA, GCCVA, and RGCCVA as  $p$  increases on HYDICE data  
..... 106

Figure 6.15. Accumulative computing time in seconds of ATGP, CCGA with SV  
calculated via SVD, VCA, GCCVA, and RGCCVA as  $p$  increases on Cuprite data 106

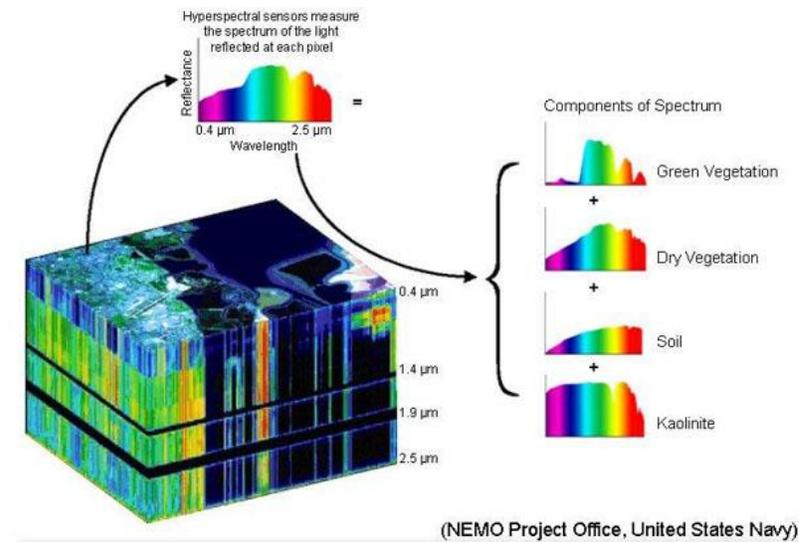
## Chapter 1: INTRODUCTION

Hyperspectral signal and image processing is one of a fast growing area which bridges communities of remote sensing and signal processing through the fact that many problems arising in the former can be now reformatted and solved by the latter. Hyperspectral sensors were originally developed for defense applications but are now widely used in many other areas, including food safety and inspection, agricultural land use and planning, environmental monitoring, as well as intelligence surveillance and medical imaging.

In the early development of remote sensing, multispectral imaging was primarily develop based on spatial information and thus can be considered as a spatial domain-based technique due to low spectral and spatial resolutions. A multispectral image consists of tens of discrete spectral bands, which does not provide as much information as a hyperspectral image does by using hundreds of contiguous spectral bands. In this case, multispectral image processing must rely on spatial information and correlation of image pixels to make up insufficient spectral information. With recent advances of high spectral resolution sensors hyperspectral imaging has become an emerging technique for data exploitation in a wide range of applications in remote sensing, many of which cannot be resolved by multispectral imaging.

## 1.1 Hyperspectral Imaging

A hyperspectral image is an image cube shown in Fig. 1.1 which contains a wealth of data information that provides crucial spectral information which cannot be offered by multispectral imaging.



**Figure 1.1.** An illustration of a hyperspectral image cube

Hyperspectral images offer many great benefits that multispectral images cannot provide. In particular, material substances which cannot be visually identified or inspected by multispectral imagery with prior knowledge, can now be uncovered by hyperspectral imagery. These objects or targets of interest generally appear either as subpixels with targets embedded in a single pixel or as mixed pixels consist of a number of material substances. Specifically, their presence is generally unexpected such as man-made targets, anomalies or rare small targets with distinctive spectral characteristics. Under the circumstances, using only spatial information is not effective.

One of fundamental task in hyperspectral imaging is finding endmembers from hyperspectral data. According to Schowengerdt (1997) an endmember is defined as an idealized pure signature that can be used to specify a particular spectral class. Due to low spectral resolution provided by multispectral imagery the presence of endmembers is very difficult to be justified. However, it is no longer true in hyperspectral imagery due to its very high spectral resolution. As a consequence finding endmembers in hyperspectral imagery has become a major application which cannot be found in multispectral imagery. This dissertation is devoted to this specific topic and further designs and develops algorithms to accomplish this task.

## 1.2 Structure and Organization

This dissertation is organized to consist of seven chapters. Chapter 2 provides preliminaries for finding endmembers in hyperspectral imagery. Chapter 3 first investigates the issue of simplex volume calculation via matrix determinant and discusses simplex volume from geometry structure aspect. Chapter 4 develops two algorithms, Geometry SGA (GSGA) and Orthogonal Projection-based SGA (OP-SGA) by taking advantage of geometry structure of simplex to ease computational complexity for calculating SV via matrix determinant and reduce the computing time of DSGA. In Chapter 5, recursive approaches to implement GSGA and OP-SGA are discussed and they further cut down computational cost. In Chapter 6, an issue encountered in convex cone volume analysis is discussed. And an algorithm, Geometric Convex Cone Volume algorithm (GCCVA), takes advantage of geometric simplex volume calculation to improve endmember finding based on convex cone is

proposed. And the recursive version of GCCVA is also implemented. Chapter 7 summarizes the research work done in this dissertation and discusses the possible future work.

## Chapter 2: FINDING ENDMEMBERS: PRELIMINARIES

### 2.1 Introduction

As mentioned earlier in Chapter 1, finding endmembers has become increasingly important in hyperspectral image analysis due to significantly spectral resolution improved on imaging sensors. Subtle material substances can be uncovered or revealed as a result of the sensor improvement. These substances generally provide vital and crucial information in image analysis, such as rare minerals in geology, toxic waste in environmental monitoring, vehicle in battle field, etc. For example, because of their rare populations endmembers are most likely to appear as anomalies and could be only detected with high spatial resolution information. As a consequence, finding or extracting endmembers becomes an important but challenge issue.

Over the past years, many Endmember Extraction Algorithms (EEAs) or Endmember Finding Algorithms (EFAs) have been developed, such as Minimum Volume Transform (MVT) by Craig (1994), Pixel Purity Index (PPI) by Boardman (1993), N-finder algorithm (N-FINDR) by Winter (1999), Iterative Error Analysis (IEA) proposed by Neville et al. (1999), Convex Cone Analysis (CCA) firstly developed by Ifarraguerri and Chang (1999), Unsupervised Fully Constrained Least Squares (UFCLS) by Heinz and Chang (2001), Automatic Target Generation Process (ATGP) by Ren and Chang (2003), Vertex Component Analysis (VCA) by Nascimento and Dias (2005), Simplex Growing Algorithm (SGA) by Chang et al. (2006), and Independent Component Analysis-based Endmember Extraction

Algorithm (ICA-EEA) by Wang and Chang (2006), and an Alternative N-FINDR (AN-FINDR), statistics-based EEAs which include PCA-EEA and High-Order Statistics-based EEA (Chang, 2013). There are three major criteria used by these algorithms, Simplex Volume (SV), Orthogonal Projection (OP), and Least Square Error (LSE).

1) SV-based Methods:

There are two types of SV-based methods proposed in the literature. One is to use maximum SV as an optimal criterion to find a simplex with the maximal SV embedded in the data space, in which case the vertices of a simplex with maximal volume are assumed to be endmembers. Algorithms developed for this type are called maximum SV-based EFAs where N-FINDR can be considered as its representative. The other type is to use minimal SV as an optimal criterion which finds a simplex with minimal SV embracing all data samples in which case the vertices of the resulting simplex are considered as endmembers. MVT is a representative of this type of algorithms. A major difference between these two types of algorithms is that the endmembers found by maximal SV-based methods are data sample vectors in the data while the minimal SV-found endmembers are not necessarily in the data.

2) OP-based Methods:

There are also two types of algorithms that can be developed using OP criterion, both of which assume that endmembers to be found are most likely the data samples with maximal or minimal OP onto a set of particularly specified vectors. One is to use of random specified vectors to identify

potential endmember candidates. The representative algorithm is PPI. The other is to specify particular vectors of interest to identify endmember candidates. A representative algorithm is ATGP.

### 3) LSE-based Methods:

The idea of this type of algorithms is to take advantage of full abundance constraints, Abundance Sum-to-one Constraint (ASC) and Abundance Non-negativity Constraint (ANC), imposed on a simplex to find potential endmember candidates. In this case, the well-known abundance Fully Constrained Least Squares (FCLS) developed by Heinz and Chang (2001) is generally used to perform linear spectral unmixing so that endmembers to be found are those data sample vectors that produce the minimal unmixing LSE. A representative algorithm is FCLS-based EFA (FCLS-EFA) developed by Gao et al. (2015).

This dissertation is mainly focused on design and development of SV-based algorithms.

## 2.2 Endmember Finding Algorithms

Endmember finding is a fundamental task in hyperspectral data analysis which attempts to search for data samples that can represent spectral classes present in the data. According to Schowengerdt (1997) such data sample vectors are called endmembers whose spectral signatures are pure in the sense of signature purity. With this interpretation many endmember extraction algorithms reported in the literature do not really extract endmembers. This is because endmember extraction assume that

endmembers must in the data. Unfortunately, this assumption is not guaranteed in real world problems. In other words, no endmembers can be extracted if endmembers are not included in real data. Under such circumstances, endmember extraction is no longer applicable. As a result, those algorithms which claimed to extract endmembers actually find endmembers. They should be considered as EFAs which look for potential endmember-like data sample vectors rather than true endmembers. For example, N-FINDR is one of typical EFA that finds endmembers-like data sample vectors as its name implies. More specifically, the found endmembers by N-FINDR are not necessarily pure vectors. It has been misleading in the literature to call it an EEA. Similarly, PPI is also an EFA not EEA. As a matter of fact, it does not extract pure signatures. Instead, PPI finds potential endmember candidates from which true endmembers must be extracted from its found endmember candidates manually by human intervention.

As mentioned earlier, many EFAs were proposed based on three different criteria, SV, OP, and LSE. Among these three types, SV-based algorithm has emerged as a preferred criterion for EFA due to the properties of a simplex which imposes two physical abundance constraints, ASC and ANC (Winter, 1999). These two abundance constraints turn out to be an effective means of determining if a signature is pure due to its convexity. In other words, SV-based algorithms look for an appropriate set of vertices that can be specified by desired endmembers. The major idea of SV-based algorithms is to find a simplex which embraces as many data sample vectors in the data space as possible due to the fact that every data sample in the simplex can be expressed as a linear mixture of its vertices where MVT is one example representing

this approach. As an alternative another approach is to look for a simplex that is embedded in the data space with maximal volume where N-FINDR is one of such algorithms developed for this purpose. But, for a given number of endmembers, N-FINDR must find  $p$  endmembers with maximum SV simultaneously. Consequently, N-FINDR generally requires excessive computing time to conduct such an exhaustive search, which nearly impossible in practice. In addition, the requirement of determining the number of endmembers  $p$  makes N-FINDR impractical since this number needs to be settled by trial-and-error and N-FINDR must be repeatedly implemented for different values of  $p$ . To resolve this issue, Virtual Dimensionality (VD) was introduced to estimate reasonable number of endmembers (Chang, 2003; Chang and Du, 2004; Chang, 2013).

Although the number of endmembers can be determined by VD, the high computational complexity is still an issue in implementing N-FINDR. To address this issue, N-FINDR has been redesigned to be implemented sequentially not simultaneously to ease its complexity. In particular, two sequential versions, called sequential N-FINDR (SQ N-FINDR) and successive N-FINDR (SC N-FINDR) (Wu et al., 2008; Xiong, 2011; Chang, 2013) are of major interest. Unfortunately, such sequential N-FINDR algorithms also suffer from a computational issue that all endmembers must be found sequentially even though they do not find all endmembers simultaneously. To further mitigate this dilemma SGA was developed as an alternative approach. Instead of finding all endmembers sequentially, SGA finds endmembers one after another by growing simplexes with vertices one at a time where each generated new endmember yields the maximal SV during the process of

growing simplexes. As a consequence, SGA significantly reduces computer processing time. Because of its design simplexes found by SGA is not a global optimal solution but rather a local optimal one. Nevertheless, extensive experiments reported in the literature SGA indeed produces very close results to those produced by sequential versions of N-FINDR.

### 2.3 Simplex Growing Algorithm (SGA)

The development of SGA arises from reducing highly computational complexity caused by an exhaustive search for endmembers performed by N-FINDR. Unlike N-FINDR which generates endmembers set simultaneously to form a simplex with maximal volume, SGA grows simplexes vertex by vertex one after another until it reaches the number of endmembers needed to be found. A similar idea was also found in VCA which grows convex hulls one vertex at a time sequentially (Nascimento and Dias, 2005) and CCGA which also grows convex cones with maximal volume one vertex at a time sequentially (Xiong, 2010).

The key to making SGA work hinges on the approach to select new vertices appropriately to enlarge growing simplexes. According to N-FINDR, a simplex formed by  $p$  endmembers is the simplex with maximum volume among all possible simplexes formed by any set of  $p$  data sample vectors for a given value of  $p$ . While using the same criterion, SGA grows the current  $k$ -simplex  $S(\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k)})$  to a  $(k+1)$ -simplex  $S(\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k)}, \mathbf{e}^{(k+1)})$  by adding a vertex  $\mathbf{e}^{(k+1)}$  so that the new  $(k+1)$ -simplex  $S(\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k)}, \mathbf{e}^{(k+1)})$  yields the maximal volumes among all possible  $(k+1)$ -simplexes  $S(\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k)}, \mathbf{r})$  augmented by any other data sample vector  $\mathbf{r}$ . The

detailed implementation of the above simplex growing process is summarized as follows.

### *Simplex Growing Algorithm (SGA)*

#### 1. Initialization:

- (a) Let  $p$  be the number of endmembers to be generated.
- (b) There are two ways to generate random initial endmembers for SGA.
  - (i) Randomly select a data sample vector as an initial endmember  $\mathbf{e}^{(0)}$  and set  $k = 0$ . In this case, SGA is referred to as 1-SGA.
  - (ii) Randomly select a pair of two data sample vectors  $(\mathbf{e}^{(0)}, \mathbf{e}^{(1)})$  to form as a random degenerate 1-dimesnional simplex which is a line segment connecting  $\mathbf{e}^{(0)}$  and  $\mathbf{e}^{(1)}$ . Set  $k = 1$ . In this case, SGA is referred to as 2-SGA.

#### 2. At $k \geq 0$ and for each sample vector $\mathbf{r}$ , we calculate $V(\mathbf{e}^{(0)}, \dots, \mathbf{e}^{(k)}, \mathbf{r})$ defined by

$$V(\mathbf{e}^{(0)}, \dots, \mathbf{e}^{(k)}, \mathbf{r}) = \frac{\left| \det \begin{bmatrix} 1 & 1 & \dots & 1 & 1 \\ \mathbf{e}^{(0)} & \mathbf{e}^{(1)} & \dots & \mathbf{e}^{(k)} & \mathbf{r} \end{bmatrix} \right|}{(k+1)!} \quad (2.1)$$

which is the volume of the  $(k+1)$ -simplex, denoted by  $S(\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k)}, \mathbf{r})$ , consists of vertices  $\{\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k)}, \mathbf{r}\}$ . Since the matrix  $\begin{bmatrix} 1 & 1 & \dots & 1 & 1 \\ \mathbf{e}^{(0)} & \mathbf{e}^{(1)} & \dots & \mathbf{e}^{(k)} & \mathbf{r} \end{bmatrix}$  in (2.1) is not necessarily a square matrix, a dimensionality reduction technique such as PCA or MNF is required to reduce the original data dimensionality  $L$  to the dimension  $k$ .

#### 3. Find $\mathbf{e}^{(k+1)}$ that yields the maximum of (2.1), that is,

$$\mathbf{e}^{(k+1)} = \arg \max_{\mathbf{r}} \{V(\mathbf{e}^{(0)}, \dots, \mathbf{e}^{(k)}, \mathbf{r})\} \quad (2.2)$$

#### 4. Stopping rule:

If  $k \leq p-1$ , then  $k \leftarrow k+1$  and go step 2. Otherwise, the final set of

$\{\mathbf{e}^{(0)}, \mathbf{e}^{(1)}, \dots, \mathbf{e}^{(p-1)}\}$  includes the desired  $p$  endmembers.

Since SGA described above requires finding SV through matrix determinants specified in (2.1), it is referred to as Determinant-based SGA (DSGA) to distinguish various versions of SGA which will be developed later in this dissertation.

Although extremely high computing time for finding endmembers simultaneously can be reduced by introducing DSGA, both N-FINDR and DSGA run into an inevitable numerical instability issue, that is, the matrices of vertices used to calculate SV are generally not of full rank due to the fact that the number of endmembers,  $p$ , is relatively small than the dimensionality of the data  $L$  which is the total number of spectral bands. Under such circumstances, two commonly used approaches are suggested. One is to reduce band dimensionality to the dimensionality of simplex as  $L$  to  $p-1$ . Another one is to make use of Singular Value Decomposition (SVD) without band Dimensionality Reduction (DR). In the former case, with different transforms used for DR it usually results in different sets of endmembers (Schowengerdt, 1997; Wang and Chang, 2006). On the other hand, in the latter case SVD may cause numerical instability which will be discussed in Chapter 3. Most importantly, the found SV by both approaches are not necessarily true SV as shown in Chapter 3. The research work in this dissertation investigates this issue and further develops two new approaches, to be called Geometric Simplex Growing Algorithm (GSGA) and Orthogonal Projection-based Simplex Growing Algorithm (OP-SGA),

as alternatives to DSGA. These two methods take advantage of simplex geometric structures to find SV without performing DR or SVD and will be introduced in Chapter 4.

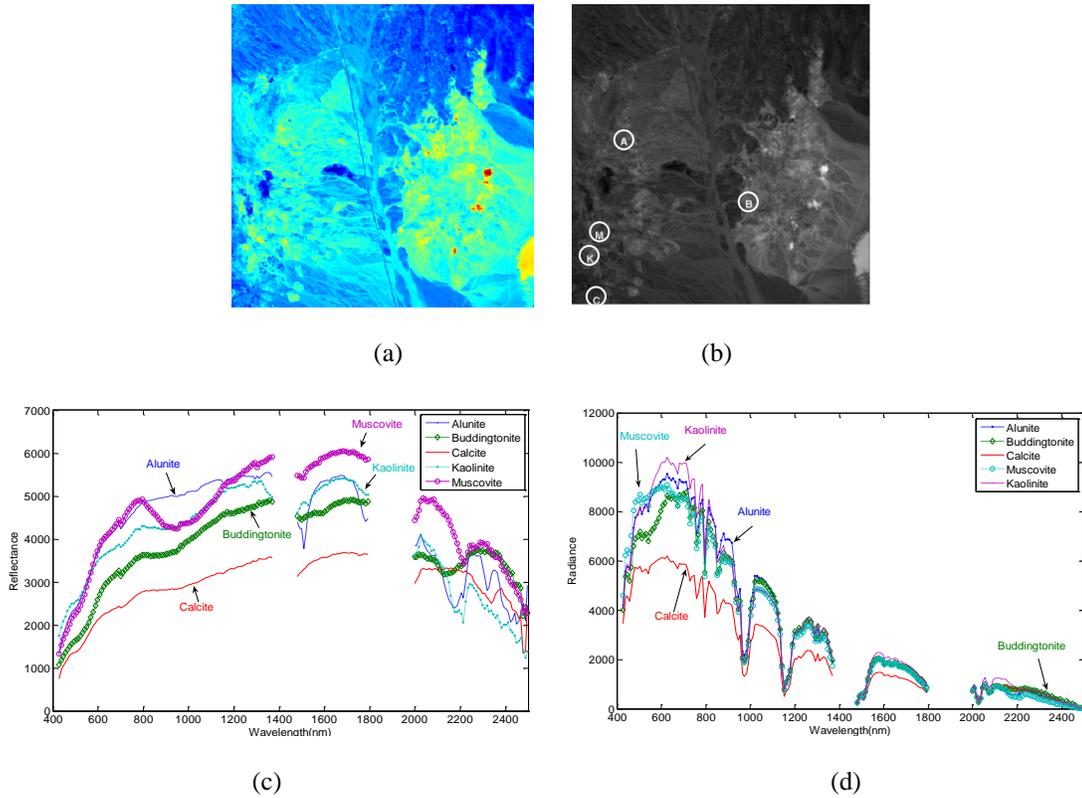
## 2.4 Image Datasets

In this dissertation, several algorithms will be proposed. In order to evaluate these algorithms, a set of synthetic images and two real image data, HYDICE scene and airborne Cuprite data, will be used to conduct experiments. There are six scenarios image designed and simulated based on the Cuprite image data. Based on the providing complete ground-truth information these synthetic images can be used for algorithm validation. With the full knowledge of these images, the strengths and weakness of each algorithm can be easily identified.

### 2.4.1 AVIRIS Cuprite Data

The first image to be used for experiments is a well-known Airborne Visible Infrared Imaging Spectrometer (AVIRIS) image scene, Cuprite data, which is available on the U.S. Geological Survey website <http://aviris.jpl.nasa.gov/>, shown in Fig. 2.1(a). This scene consists of 224 spectral bands with size  $350 \times 350$  pixels and was collected over Cuprite mining site, Nevada, in 1997. Because the surficial geology is well understood and ground-truth in the form of mineral spectra is available in spectra library, Cuprite images are commonly used for analysis in the literature. Due to water absorption and low SNR in bands 1-3, 105-115, and 150-170, those bands were removed prior to analyses which results in a total of 189 bands used for experiments. In Fig. 2.1(b), five minerals, Alunite (A), Buddingtonite (B), Calcite

(C), Kaolinite (K) and Muscovite (M), are specified by pixels with white circle and labeled by A, B, C, K, and M, respectively, along with their reflectance spectra shown in Fig. 2.1(c) and radiance spectra in Fig. 2.1(d). Both reflectance and radiance data are used to validate algorithms in this dissertation.

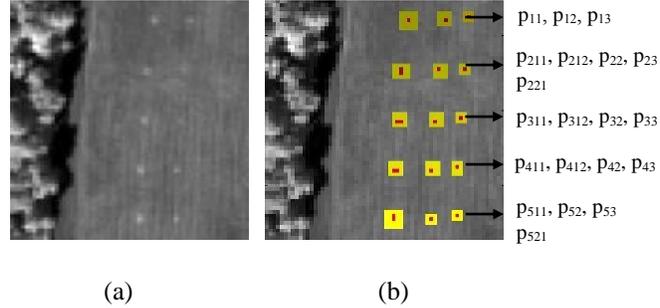


**Figure 2.1.** (a) Cuprite AVIRIS image scene (b) spatial positions of five pure pixels corresponding to minerals: alunite (A), buddingtonite (B), calcite (C), kaolinite (K) and muscovite (M); (c) Five mineral reflectance spectra; (d) Five radiance spectra

#### 2.4.2 Hyperspectral Digital Imagery Collection Experiment (HYDICE)

The image scene shown in Fig. 2.2 was acquired by the airborne Hyperspectral Digital Imagery Collection Experiment (HYDICE) sensor in August 1995 from a flight altitude of 10000 feet with ground sampling distance approximately 1.5 meters. It has 210 spectral channels ranging from 0.4  $\mu\text{m}$  to 2.5  $\mu\text{m}$  with spectral resolution is

10 nm. The low signal/high noise bands, bands 1-3 and bands 202-210, and water vapor absorption bands, bands 101-112 and bands 137-153, were removed. Therefore, a total of 169 bands were used for experiments. It has size of  $64 \times 64$  pixel vectors shown in Fig. 2.2(a) along with its ground-truth provided in Fig. 2.2(b).



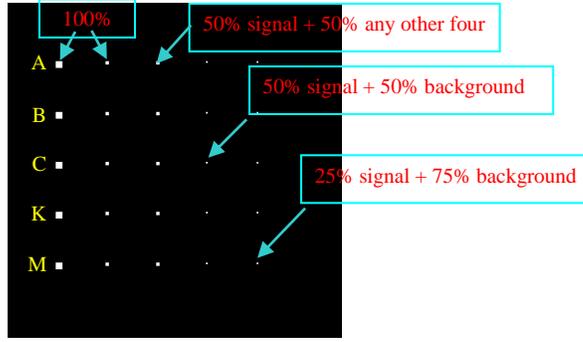
**Figure 2.2.** (a) A HYDICE panel scene which contains 15 panels; (b) Ground truth map of spatial locations of the 15 panels

There are 15 panels located on a grass field and arranged in a  $5 \times 3$  matrix where there is a forest on the left edge of the scene and a road on the right edge of the scene. Each colored element in Fig. 2.2(b) is a square panel and denoted by  $p_{ij}$  with rows indexed by  $i = 1, 2, \dots, 5$  and columns indexed by  $j = 1, 2, 3$ . For each row  $i$ , the panels  $p_{i1}$ ,  $p_{i2}$ ,  $p_{i3}$  were made from the same material but with different sizes that in first, second, and third column are  $3\text{m} \times 3\text{m}$ ,  $2\text{m} \times 2\text{m}$ , and  $1\text{m} \times 1\text{m}$ , respectively. For each column  $j$ , the five panels  $p_{1j}$ ,  $p_{2j}$ ,  $p_{3j}$ ,  $p_{4j}$ ,  $p_{5j}$  have the same size but differ from their making materials or paints. The 1.56m-spatial resolution of the image scene suggests that most of the 15 panels are one pixel in size except that the panels in the 1<sup>st</sup> column with the 2<sup>nd</sup>, 3<sup>rd</sup>, 4<sup>th</sup>, 5<sup>th</sup> rows which are two-pixel panels, denoted by  $p_{211}$ ,  $p_{221}$ ,  $p_{311}$ ,  $p_{312}$ ,  $p_{411}$ ,  $p_{412}$ ,  $p_{511}$ ,  $p_{521}$ . As a result, there are a total 19 panel pixels of interest, where red pixels (R pixels) are the panel center pixels, considered to be pure

pixels, and the pixels in yellow (Y pixels) are boundary panel pixels mixed with the background, considered to be mixed pixels.

#### 2.4.3 Synthetic Image

The five mineral spectra signatures in Cuprite radiance spectra shown in Fig. 2.1(d) were used to simulate synthetic images in this dissertation. The synthetic images for experiments are in size of  $200 \times 200$  pixels image with 25 panels of various sizes which are arranged in a  $5 \times 5$  matrix and located at the center of the scene shown in Fig. 2.3. The 25 panels were simulated according to legends provided in Fig. 2.1 with five panels in each row having the same mineral signature and five panels in each column having the same size. Among the 25 panels are five  $4 \times 4$  pure-pixel panels for each row in the first column and five  $2 \times 2$  pure-pixel panels for each row in the second column, five  $2 \times 2$  mixed pixel panels for each row in the third column, and both the five  $1 \times 1$  subpixel panels for each row in the fourth and fifth columns. The purpose of introducing five mixed pixel panels in the 3<sup>rd</sup> column and subpixel panels in 4<sup>th</sup> and 5<sup>th</sup> columns was designed to conduct a study and analysis on five mineral signatures with different mixing conditions in one pixel and five mineral signatures embedded in single pixels at subpixel scale. And the sample mean of the image in Fig. 2.1(a) was used as a background signature **b** to simulate the image background in Fig. 2.3.



**Figure 2.3.** A set of 25 panels simulated by A, B, C, K, M

Table 2.1 and 2.2 tabulate the details of mineral composition in the 20 mixed pixels in 3<sup>rd</sup> column and 10 subpixels in 4<sup>th</sup> and 5<sup>th</sup> columns, respectively.

**Table 2.1.** Mixed pixel panels in 3<sup>rd</sup> column

Row	Panel pixel composition	
1 <sup>st</sup>	$\mathbf{P^1_{3,11}=0.5A+0.5B}$	$\mathbf{P^1_{3,12}=0.5A+0.5C}$
	$\mathbf{P^1_{3,21}=0.5A+0.5K}$	$\mathbf{P^1_{3,22}=0.5A+0.5M}$
2 <sup>nd</sup>	$\mathbf{P^2_{3,11}=0.5B+0.5A}$	$\mathbf{P^2_{3,12}=0.5B+0.5C}$
	$\mathbf{P^2_{3,21}=0.5B+0.5K}$	$\mathbf{P^2_{3,22}=0.5B+0.5M}$
3 <sup>rd</sup>	$\mathbf{P^3_{3,11}=0.5C+0.5A}$	$\mathbf{P^3_{3,12}=0.5C+0.5B}$
	$\mathbf{P^3_{3,21}=0.5C+0.5K}$	$\mathbf{P^3_{3,22}=0.5C+0.5M}$
4 <sup>th</sup>	$\mathbf{P^4_{3,11}=0.5K+0.5A}$	$\mathbf{P^4_{3,12}=0.5K+0.5B}$
	$\mathbf{P^4_{3,21}=0.5K+0.5C}$	$\mathbf{P^4_{3,22}=0.5K+0.5M}$
5 <sup>th</sup>	$\mathbf{P^5_{3,11}=0.5M+0.5A}$	$\mathbf{P^5_{3,12}=0.5M+0.5B}$
	$\mathbf{P^5_{3,21}=0.5M+0.5C}$	$\mathbf{P^5_{3,22}=0.5M+0.5K}$

**Table 2.2.** Subpixel panels in 4<sup>th</sup> and 5<sup>th</sup> column

Row	50% subpixels in 4 <sup>th</sup> column	25% subpixels in 5 <sup>th</sup> column
1 <sup>st</sup>	$\mathbf{P^1_{4,1}=0.5A+0.5b}$	$\mathbf{P^1_{5,1}=0.25A+0.75b}$
2 <sup>nd</sup>	$\mathbf{P^2_{4,1}=0.5B+0.5b}$	$\mathbf{P^2_{5,1}=0.25B+0.75b}$
3 <sup>rd</sup>	$\mathbf{P^3_{4,1}=0.5C+0.5b}$	$\mathbf{P^3_{5,1}=0.25C+0.75b}$
4 <sup>th</sup>	$\mathbf{P^4_{4,1}=0.5K+0.5b}$	$\mathbf{P^4_{5,1}=0.25K+0.75b}$
5 <sup>th</sup>	$\mathbf{P^5_{4,1}=0.5M+0.5b}$	$\mathbf{P^5_{5,1}=0.25M+0.75b}$

So, there are a total of 130 pixels present in the scene, where 80 pure pixels in 1<sup>st</sup> column, 20 pure pixels in 2<sup>nd</sup> column, 20 mixed pixels in 3<sup>rd</sup> column, five 50%-abundance subpixels in 4<sup>th</sup> column, and 25%-abundance subpixels in 5<sup>th</sup> column.

Once panel pixels are simulated as described above, two types of target insertion can be designed to simulate experiments for various application.

#### 2.4.3.1 Target Implantation (TI)

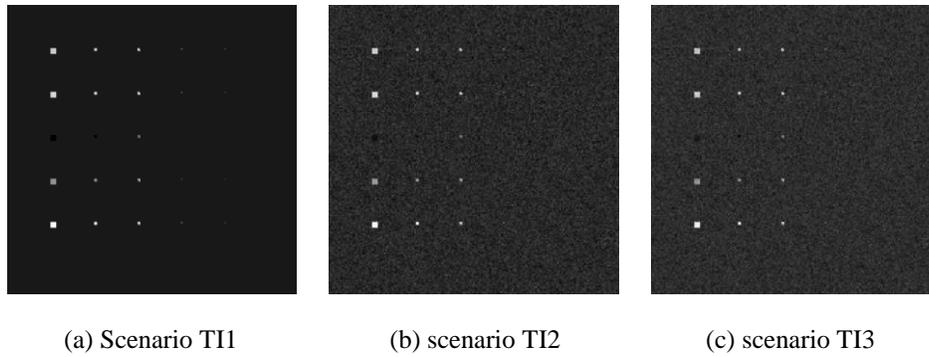
The first type of target insertion is Target Implantation (TI) which can be simulated using three scenarios as it is described in the following.

##### *Scenario T11 (Clean targets implanted into an image with clean background)*

Given a clean background image, clean targets simulated as mentioned earlier can be implanted into the background image by replacing their corresponding background pixels. The resulting image shown in Fig. 2.4(a) is a synthetic image which has clean targets implanted in the image scene with clean background.

##### *Scenario T12 (Clean targets implanted into an image with noisy background)*

In this simulation, a noisy background image was generated by a specific background signature **b** corrupted by an additive Gaussian noise to achieve a specific signal-to-noise ratio (SNR) defined as 50% signature divided by the standard deviation of the noise (Harsanyi and Chang, 1994). The clean targets are implanted into the simulated noisy background image by replacing their corresponding background pixels. The resulting image shown in Fig. 2.4(b) is a synthetic image which has clean targets implanted in the image scene with noisy background.



**Figure 2.4.** Three scenarios designed for target implantation

*Scenario TI3 (Plus additive Gaussian noise into an image with clean targets implanted in clean background)*

The synthetic image was the same image simulated in scenario TI1 but with an additive Gaussian noise to achieve a specific SNR. As a result, the synthetic image has both clean targets and clean background corrupted by an additive Gaussian noise with a specific SNR as shown in Fig. 2.4(c).

#### 2.4.3.2 Target Embeddedness (TE)

A second type of target insertion is Target Embeddedness (TE) which can be simulated as the following three scenarios.

*Scenario TE1 (Clean targets embedded into an image with clean background)*

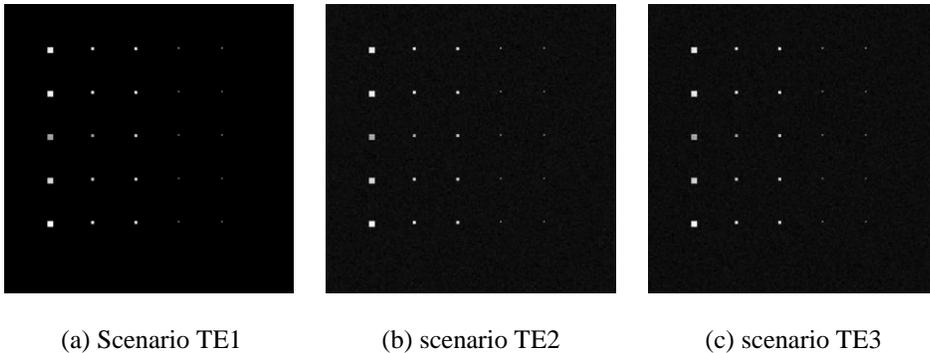
Given a clean background image, clean targets simulated as mentioned earlier can be superimposed into the background image while keeping their corresponding background pixels. The resulting image shown in Fig. 2.5(a) is a synthetic image which has clean targets embedded in the image scene with clean background.

*Scenario TE2 (Clean targets embedded into an image with noisy background)*

In this simulation, the background image of this scenario is the same one used in scenario TI2, that is, noisy background image simulated by a signature  $\mathbf{b}$  with an additive Gaussian noise to achieve a specific SNR. Instead of replacing corresponding background pixels as it is done for TI2, the targets are actually embedded into background pixels. In other words, the background pixels are not removed to accommodate the implanted targets. In this case, the resulting image shown in Fig. 2.5(b) is a synthetic image which has clean targets embedded in the image scene with noisy background.

*Scenario TE3 (Plus additive Gaussian noise into an image with clean targets embedded in clean background)*

The synthetic image was the same image simulated in scenario TE1 but with an additive Gaussian noise to achieve a specific SNR. As a result, the synthetic image has both embedded clean targets and clean background corrupted by an additive Gaussian noise with a specific SNR as shown in Fig. 2.5(c).



**Figure 2.5.** Three scenarios designed for target embeddedness

## Chapter 3: GROWING SIMPLEX VOLUME ANALYSIS (GSVA)

### 3.1 Introduction

As mentioned in Chapter 2, maximal SV is commonly used as an optimal criterion for finding endmembers. There are several well-known simplex-volume-based EFA such as N-FINDR, sequential N-FINDR (SQ N-FINDR), successive N-FINDR (SC N-FINDR), and Simplex Growing Algorithm (SGA) which were derived from N-FINDR to ease computational complexity (Chang et al., 2006). In the literature, little work has been reported on how SV is calculated. Intuitively, it seems that determinant-based volume calculation could work in any situation. As a matter of fact, it turns out that the issue of calculating SV is much more complicated and involved than we thought. In this chapter we investigate this issue and focus on methods of finding SV from two different aspects, geometry structure and eigen-analysis.

There is one major issue in eigen-analysis-based SV calculation which requires finding determinants of matrices formed by vertices. Due to the fact that the number of vertices is generally smaller than its data dimensionality, SV calculation suffers from ill-rank issues in finding matrix determinant. In this case, a simplex must be calculated by an ill-ranked matrix. There are two different approaches. One is to find the matrix determinant via Singular Value Decomposition (SVD) without issuing data dimensionality. The other is to pre-process data by data Dimensionality Reduction (DR) prior to matrix determinant calculation. These two methods can be categorized

as eigen-analysis approach. From the geometric structure of a simplex viewpoint, SV can be derived by multiplying its base with its height. As a result, the volume of an  $(n+1)$ -simplex can be calculated by the distance or height to its base, which is an  $n$ -simplex. This approach is the one will be explored in this dissertation.

### 3.2 Criteria for Simplex Volume Calculation

#### 3.2.1 Geometric Structure Approach

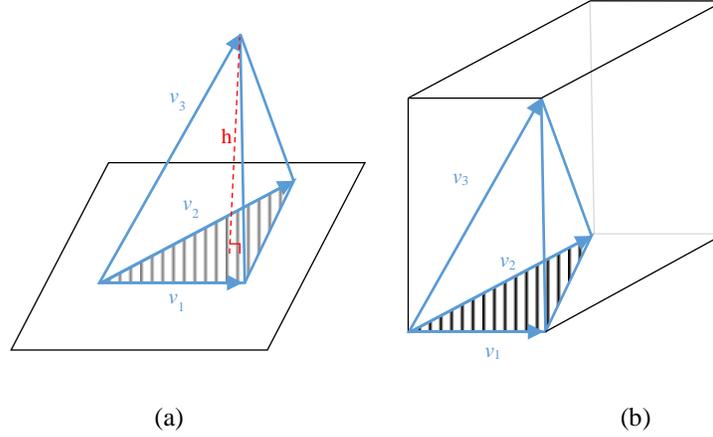
In geometry, a simplex, also called a hyper-tetrahedron, is the generalization of a tetrahedral region of space to arbitrary  $n$  dimensions. A  $k$ -dimensional simplex or  $(k+1)$ -vertex simplex  $S_k$  is a  $k$ -dimensional convex hull with  $k+1$  vertices (Stein, 1996; Wong, 2003; Friedberg et al., 2003; Berger, 2010). Thus, a single point could be considered as 0-simplex and 1-simplex is the line segment between two specified points. Moreover, 2-simplex is a triangle, 3-simplex is a tetrahedron, and 4-simplex is a pentachoron.

Suppose  $k+1$  points  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k \in \mathfrak{R}^n$  in an  $n$ -dimensional space are affinely independent, which means  $\mathbf{m}_1 - \mathbf{m}_0, \dots, \mathbf{m}_k - \mathbf{m}_0$  are linearly independent, a  $k$ -simplex  $S_k$  can be expressed by the set of points as

$$S_k = \left\{ \alpha_0 \mathbf{m}_0 + \alpha_1 \mathbf{m}_1 + \dots + \alpha_k \mathbf{m}_k \mid \alpha_i \geq 0, 0 \leq i \leq k, \sum_{i=0}^k \alpha_i = 1 \right\}. \quad (3.1)$$

The content or hyper-volume (hereinafter referred to briefly as volume) of a simplex is denoted as  $V(S_k)$ , which can be the length of 1-dimensional simplex, the area of 2-dimensional simplex, the volume of 3-dimensional simplex, and so on. It is

noted that any vertex of a  $k$ -dimensional simplex can be regarded as the apex of a pyramid on a  $(k-1)$ -dimensional base formed by  $k$  vertices (Wong, 2003).



**Figure 3.1.** (a) 3-dimensional simplex formed by three edge vectors  $v_1, v_2, v_3$  (b) 3-dimensional parallelepiped (or parallelepiped) formed by the same edge vectors

**Theorem:** Volume  $V(S_k)$  of a  $k$ -simplex is  $\frac{1}{k!}$  of the volume  $V(P_k)$  of the corresponding parallelotope, i.e.  $V(S_k) = \frac{1}{k!}V(P_k)$ .

*Proof.* Let  $V(S_{k-1})$  denotes the volume of the base and  $h$  the perpendicular distance (altitude or height) of the apex from the subspace containing the base. The volume  $V(S_k)$  of the pyramid is given by

$$V(S_k) = \int_{h=0}^{h_k} V(S_{k-1}) \left( \frac{h}{h_k} \right)^{k-1} dh = V(S_{k-1}) \cdot \frac{h_k}{k} \quad (3.2)$$

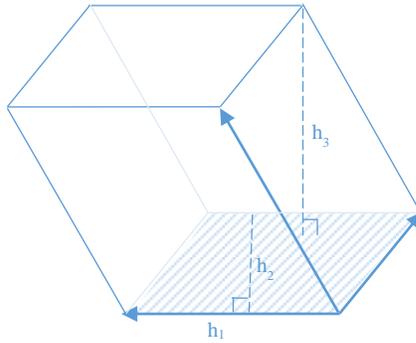
through integral calculus as in Fig. 3.1(a). Therefore,  $V(S_k)$  can be renewed by the previous  $V(S_{i-1})$  for  $1 \leq i \leq k$  recursively as

$$V(S_k) = V(S_{k-1}) \cdot \frac{h_k}{k} = \frac{1}{k!} h_k h_{k-1} \cdots h_1, \quad (3.3)$$

where  $h_1$  is the distance between the first two vertices. Considering a  $k$ -dimensional parallelotope  $P_k$  decided by  $k$  edge vectors which has the first base  $h_1$  and  $k-1$  heights  $h_2, \dots, h_k$  perpendicular to base, then the volume of  $P_k$  is  $\prod_{i=1}^k h_i$ . Therefore  $V(S_k)$  in (3.3) can be rewrite as

$$V(S_k) = \frac{1}{k!} \prod_{i=1}^k h_i = \frac{1}{k!} V(P_k). \quad (3.4)$$

By (3.3) and (3.4), it is proved that volume of a  $k$ -simplex  $V(S_k)$  is  $\frac{1}{k!}$  of the volume of the corresponding parallelotope as shown in Fig. 3.1(b). The interpretation of volume calculation for 3-dimensional parallelotope is in Fig. 3.2.



**Figure 3.2.** An interpretation of the volume calculation for 3-dimensional parallelotope

### 3.2.2 Eigen-analysis Approach

Now considering an  $n$ -simplex  $S_n$  in an  $n$ -dimensional space with  $n+1$  vertices, i.e.  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_n \in \mathfrak{R}^n$ .

**Corollary:** The volume of the parallelotope  $P_n$ , which is determined by  $n$  edge vectors  $\mathbf{m}_1 - \mathbf{m}_0, \mathbf{m}_2 - \mathbf{m}_0, \dots, \mathbf{m}_n - \mathbf{m}_0$ , corresponding to  $S_n$ , is the absolute value of  $\det(\mathbf{M}_E)$  with its edge matrix  $\mathbf{M}_E = [\mathbf{m}_1 - \mathbf{m}_0, \mathbf{m}_2 - \mathbf{m}_0, \dots, \mathbf{m}_n - \mathbf{m}_0]$  as

$$V(P_n) = |\det(\mathbf{M}_E)| = |\det([\mathbf{m}_1 - \mathbf{m}_0, \dots, \mathbf{m}_n - \mathbf{m}_0])|. \quad (3.5)$$

Thus, by (3.4) and (3.5) the volume  $V(S_n)$  of  $n$ -simplex  $S_n$  can be expressed as

$$V(S_n) = \frac{1}{n!} V(P_n) = \frac{1}{n!} |\det(\mathbf{M}_E)|. \quad (3.6)$$

By elementary column operation of matrix and properties of the determinant for block matrix, it can be obtained that

$$\begin{aligned} & \det \left( \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{m}_0 & \mathbf{m}_1 & \cdots & \mathbf{m}_n \end{bmatrix} \right) \\ &= \det \left( \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \mathbf{m}_0 & \mathbf{m}_1 - \mathbf{m}_0 & \cdots & \mathbf{m}_n - \mathbf{m}_0 \end{bmatrix} \right) \\ &= \det([\mathbf{m}_1 - \mathbf{m}_0, \mathbf{m}_2 - \mathbf{m}_0, \dots, \mathbf{m}_n - \mathbf{m}_0]) \end{aligned} \quad (3.7)$$

According to (3.7),  $V(S_n)$  is equivalent to

$$\begin{aligned} V(S_n) &= \frac{1}{n!} |\det(\mathbf{M}_E)| \\ &= \frac{1}{n!} |\det([\mathbf{m}_1 - \mathbf{m}_0, \mathbf{m}_2 - \mathbf{m}_0, \dots, \mathbf{m}_n - \mathbf{m}_0])|, \\ &= \frac{1}{n!} \left| \det \left( \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{m}_0 & \mathbf{m}_1 & \cdots & \mathbf{m}_n \end{bmatrix} \right) \right| \end{aligned} \quad (3.8)$$

which is a well-known approach to calculate volume of simplex via determinant of its vertex matrix in linear algebra. It is worth noting that volume derived by (3.8) is only available while  $\mathbf{M}_E$  is a square matrix since determinant is defined as a value associated with a square matrix. More details of (3.8) can be found in (Stein, 1996).

In order to obtain determinant easily and efficiently, some properties of determinant are recalled.

**Properties:** Let  $\mathbf{A}$  and  $\mathbf{B}$  be an  $n \times n$  square matrix, there are some properties of the determinant  $\det(\mathbf{A})$ :

1.  $\det(\mathbf{AB}) = \det(\mathbf{A})\det(\mathbf{B})$
2.  $\det(c\mathbf{A}) = c^n \det(\mathbf{A})$  where  $c$  is a scalar
3.  $\det(\mathbf{A}) = \prod_{i=1}^n \lambda_i$  where  $\lambda_i$  is eigenvalue of  $\mathbf{A}$
4.  $\det\left(\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{B} & \mathbf{C} \end{bmatrix}\right) = \det\left(\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{C} \end{bmatrix}\right) = \det(\mathbf{A})\det(\mathbf{C})$
5. By LU factorization,

$$\det(\mathbf{A}) = |\mathbf{A}| = |\mathbf{PLU}| = |\mathbf{P}| |\mathbf{L}| |\mathbf{U}| = \pm \prod_{i=1}^n u_{ii}$$

where  $\mathbf{L}$  is a lower triangular matrix and  $\mathbf{U}$  is an upper triangular matrix, both of which given by

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ l_{21} & 1 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & u_{(n-1)n} & u_{(n-1)n} \\ 0 & 0 & 0 & \cdots & u_{nn} \end{bmatrix},$$

and  $\mathbf{P}$  is a permutation matrix to be used for pivoting so as to achieve numerical stability during the LU factorization.

In theoretical view, a simplex  $S_k$  is often defined in a  $k$ -dimensional space with  $k+1$  vertices and volume  $V(S_k)$  can be easily calculated via (3.6). However, it stays not true in reality. To find a simplex  $S_k$  in  $n$ -dimensional space with  $n$  larger than  $k$  is more common instead. As a result, several comments on calculating SV via (3.6) are worthwhile.

1. Since a simplex  $S_k$  with  $k+1$  vertices in  $n$ -dimensional space has only rank of  $k$  at most which is generally much smaller than full spectral dimensionality,  $n$ . Therefore, calculating SV via (3.6) generally requires dimensionality reduction to reduce  $n$  to  $k$  to find the DR-determinant of a non-square matrix. However, it is not necessary while using (3.1) to derive  $V(S_k)$ .
2. Once a simplex  $S_k$  in  $n$ -dimensional space is reduced to  $k$ -dimensional space, the matrix  $\mathbf{M}_E$  used to calculate  $V(S_k)$  in (3.6) becomes a  $k \times k$  square matrix. And  $|\det(\mathbf{M}_E)|$  can be calculated via its properties. Nevertheless, it should be noted that the found volume after DR does not guarantee to be the real volume of  $S_k$  but rather an approximation smaller than its original volume.
3. When no DR is applied we must deal with the issue of undefined determinant for a non-square matrix in (3.6). In this case, SVD can be used for accomplishing the purpose. Briefly, the SVD can be applied to any  $m \times n$  matrix whereas eigenvalue decomposition can only be used to non-singular square matrix. Applying an SVD of  $\mathbf{M}_E$ , two relations are held.

$$(1) \mathbf{M}_E^* \mathbf{M}_E = \mathbf{V} \Sigma^* \mathbf{U}^* \mathbf{U} \Sigma \mathbf{V}^* = \mathbf{V} (\Sigma^* \Sigma) \mathbf{V}^*$$

$$(2) \mathbf{M}_E \mathbf{M}_E^* = \mathbf{U} \Sigma^* \mathbf{V}^* \mathbf{V} \Sigma \mathbf{U}^* = \mathbf{U} (\Sigma^* \Sigma) \mathbf{U}^*$$

Consequently, the columns of  $\mathbf{V}$  are eigenvectors of  $\mathbf{M}_E^* \mathbf{M}_E$ , the columns of  $\mathbf{U}$  are eigenvectors of  $\mathbf{M}_E \mathbf{M}_E^*$ , and the non-zero elements or singular value of  $\Sigma$  are the square roots of the non-zero eigenvalues of  $\mathbf{M}_E \mathbf{M}_E^*$  or  $\mathbf{M}_E^* \mathbf{M}_E$  (Kwizera, 2010). Thus, let us define a pseudo-determinant of  $\mathbf{M}_E$ ,  $\det(\mathbf{M}_E^+)$ , as

$$\det(\mathbf{M}_E^+) = \left( \det(\mathbf{M}_E^* \mathbf{M}_E) \right)^{1/2} = \left( \det(\mathbf{M}_E \mathbf{M}_E^*) \right)^{1/2} = \left( \prod_{\forall \lambda_i \neq 0} \lambda_i \right)^{1/2} = \prod_{\forall \sigma_i \neq 0} \sigma_i \quad (3.9)$$

where  $\lambda_i$  is the non-zero eigenvalues of  $\mathbf{M}_E^* \mathbf{M}_E$  or  $\mathbf{M}_E \mathbf{M}_E^*$  and  $\sigma_i$  is the corresponding singular value of  $\Sigma$ . Unfortunately,  $V(S_k)$  calculated by (3.9) does not provide real volume of simplex  $S_k$ . There is no such problem by using (3.3).

Theoretically, both (3.3) and (3.8) yield the same volume while finding volume of  $n$ -simplex in  $n$ -dimensional space. When it comes to practical implementation they do not produce identical answers due to the strategies used to solve the issue of nonexistent determinant of non-square matrix as mentioned above. However, using (3.3) does not have such problem.

### 3.3 Numerical Results

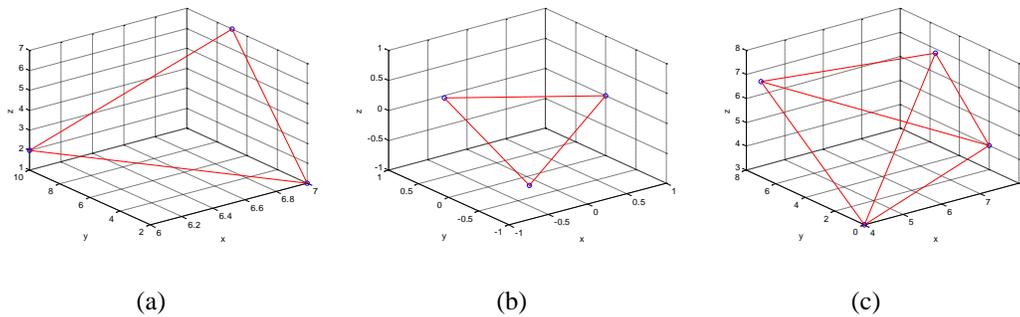
In this section, a comparison of volume calculation using various approaches described in section 3.2 is conducted. Let us define the volume calculated by (3.3) as Geometric Simplex Volume (GSV), the volume calculated by determinant as Determinant Simplex Volume (DSV), and the DSV with DR or SVD applied as DR-DSV volume or pseudo-DSV. Theoretically, GSV and DSV can be shown to be identical. DR-GSV is also computed for comparison. The DR-GSV is derived by reducing the dimensionality of  $S_k$  in  $N$ -dimensional space to  $k$ -dimensional space as  $S_k^\#$  and then apply (3.3) to  $S_k^\#$ .

To compare the volume value of simplex among these methods, we first use two simple 2-simplexes and a 3-simplex all lied in 3-dimensional space to illustrate their

differences and then a real hyperspectral image scene is also considered for comparative analysis.

### 3.3.1 Mathematical Examples

First of all, three different simplexes in a 3-dimensional space are generated. A triangle is formed by three points randomly generated in a 3-dimensional space as shown in Fig. 3.3(a). The three vertices are specified by coordinates,  $(7,7,7)$ ,  $(6,10,2)$ ,  $(7,2,1)$ . And a special case of a 2-simplex embedded in a 3-dimensional data space known as a regular 2-simplex or a regular triangle with equal edge length 1.633 is also generated for comparison in Fig. 3.3(b) where the three vertices are specified by coordinates,  $(1,0,0)$ ,  $(-0.3333,0.9428,0)$ , and  $(-0.3333,-0.4714,-0.8165)$ . Finally, an arbitrary 3-simplex is plotted as a pyramid in Fig. 3.3(c) with vertices specified by coordinates,  $(8,2,4)$ ,  $(7,3,8)$ ,  $(4,7,7)$ ,  $(4,0,3)$ . Unlike the first two simplexes in Fig. 3(a-b) which are more realistic where the dimensionality of simplex is less than the dimensionality of its ambient space, this 3D-simplex has the same dimensionality with its ambient space. Comparisons for volume calculated by various volume calculation methods for the three examples are tabulated in Table 3.1.



**Figure 3.3.** (a) A triangle (b) a regular triangle (c) a 3D-simplex with one vertex in origin; in 3D-space

As we can see in Table 3.1, volumes calculated via three methods, geometric method, determinant-based method, and pseudo-determinant method, all yield the same value while the dimensionality of simplex equals to the dimensionality of its ambient space. In this case, no DR is needed and SVD could be used as an alternative method to obtain the volume. But this result does not support the usage of SVD for calculating volume of simplex in a different dimensional space. The GSV still yields the same volume as DSV while the dimensionalities of simplex and ambient space are different where the Pseudo-DSV is not consistent.

**Table 3.1.** A comparative analysis for volume among various methods of simplexes in 3-D space

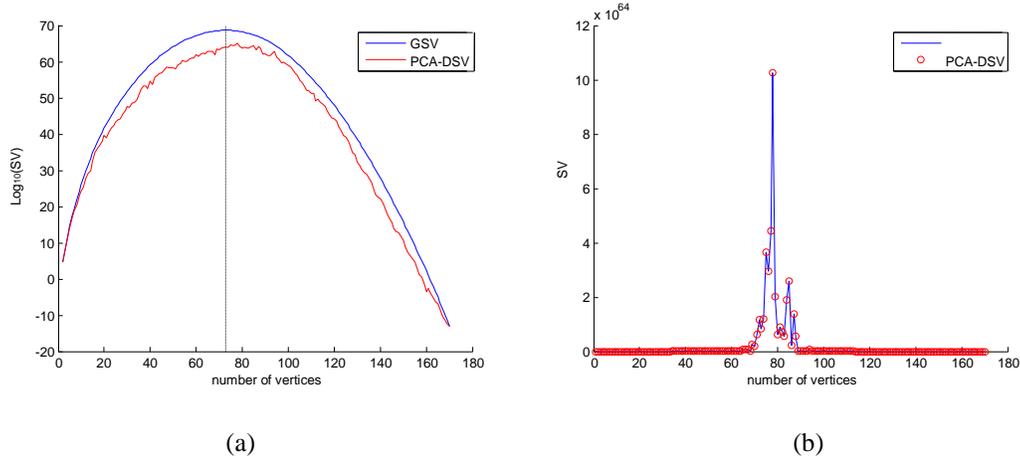
S \ Method	GSV	DSV	Pseudo-DSV	PCA-GSV	PCA-DSV
Arbitrary 2-simplex (Fig. 3.3(a))	21.8518	X	155.5426	21.8518	21.8518
Regular 2-simplex (Fig. 3.3(b))	1.1547	X	1.2172	1.1547	1.1547
Arbitrary 3-simplex (Fig. 3.3(c))	15.8333	15.8333	15.8333	X	X

Another experiment is studied on a real image scene which is HYDICE data shown in Fig. 2.1.

### 3.3.2 Real Image Experiments

To compare the volumes calculated by these three methods, we first applied SGA to find a set of endmembers on this image. The simplex formed by the set of endmembers has 170 vertices in 169-dimensional space. Since the impact which occurred by a simplex with lower dimensionality than its ambient space is desired, we reformed the simplex using these set of data sample vectors as  $S_k, k = 0, 1, \dots, 169$ , by

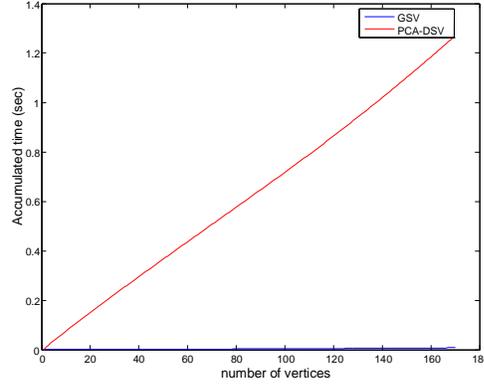
the order of the found endmembers. In other words,  $S_0$  is formed by a single vertex  $v_0$ ,  $S_1$  is a line segment formed by two vertices  $v_0, v_1$ , and  $S_k$  consists of  $k+1$  vertices  $v_0, v_1, \dots, v_k$ . And the comparison for volume of simplex  $S_k$  is shown in Fig. 3.4.



**Figure 3.4.** SV comparison (a) GSV versus PCA-DSV (b) PCA-GSV versus PCA-DSV

By examining the results in Fig. 3.4, some interesting observations are worth being mentioned. First, the volume calculated by DR-DSV and GSV would be different, especially when the number of vertices is high. Although in the mathematical examples as shown in Table 3.1 volumes calculated by DR-DSV in a 3D space were identical to GSV. However, there is a numerical issue needed to be addressed as the number of vertices and number of dimensions are increased. Second, the volume using both methods shows a trend that the volume was increased until it reached a peak at a specific number of vertices and then it began to decrease. This specific number occurs when the height, which is the distance from the newly generated vertex  $v_k$  to the previous formed simplex  $S_{k-1}$  denoted as  $h_k$ , is less than the

dimensionality of simplex  $S_k$ , i.e.  $\frac{h_k}{k} < 1$ . The peak occurs when the 73<sup>th</sup> and 78<sup>th</sup> endmember was found without and with DR performed respectively.



**Figure 3.5.** Comparisons of accumulated computing time required by GSV and PCA-DSV

A major advantage of GSV over DSV is that GSV calculates SV which only uses mathematical products. This cannot be done by using DSV since the volume of  $S_j$  can only be calculated one at a time. Fig. 3.5 shows the accumulated time required by GSV to calculate SV by multiplying height  $h(S_{j+1})$  with the SV of the previous  $j$ -vertex simplex  $S_j$  compared to the accumulated time required by PCA-DSV where the computing time was obtained by an average of 100 trials. The experimental results show that GSV can significantly reduce the computing time of SV calculation.

To further compare the performance for each method, a comparative of the computational complexity is in Table 3.2. Let us denote that  $p$  is the number of vertices and  $L$  is dimensionality of the ambient space.

**Table 3.2.** Computational complexity of each SV approach

Method	GSV	DSV	PCA-DSV	Pseudo-DSV
Computational complexity	$O(pL)$	$O(p^3)$	$O(pL^2) + O(p^3)$	$O(p^2L)$

In order to calculate the volume of a simplex by GSV, only inner product operators are needed, which has complexity  $O(pL)$ ; compared to using DSV which has complexity with  $O(p^3)$ . In real world problems, the high data dimensionality is always an issue. To overcome this issue, a DR technique or SVD must be applied to the calculation. In this case the computational complexity becomes  $O(pL^2) + O(p^3)$  if PCA is performed for DR, and  $O(p^2L)$  is requested to perform SVD process.

A major advantage of GSV over DSV is that GSV provides ability in finding volumes from its previous found simplex in (2.2). This cannot be accomplished by the determinant-based method. Considering a desired simplex  $S_k$  to be found with  $k+1$  vertices  $v_0, v_1, \dots, v_k$ , where  $S_{k-1}$  with  $v_i, i = 0, 1, \dots, k-1$ , is known or found in previous process, an inner product could be applied to find the distances from all other data samples to  $S_{k-1}$  simultaneously. This cannot be done by using DSV since the volume of  $S_k$  can only be calculated one at a time.

### 3.4 Conclusions

This chapter investigates an issue arising from SV calculation caused by the inconsistent dimensionality of simplex and its ambient space. A geometric method is applied to address the issue. It has several benefits that determinant-based method cannot provide. One is no requirement of DR. Another is that geometric approach gives the true SV without suffering from the numerical issues when it calculates the volume of a simplex in a high dimensional space. Third, the computational complexity is significantly reduced. Last but not least, when geometric approach is

applied to endmember finding algorithms it can avoid finding incorrect endmembers caused by the numerical errors resulting from using determinant-based approach to calculate SVs.

## Chapter 4: DESIGN AND DEVELOPMENT OF VARIANTS OF SGA

### 4.1 Introduction

This chapter develops two new endmember finding algorithms, to be called Geometric Simplex Growing Algorithm (GSGA) and Orthogonal Projection-based Simplex Growing Algorithm (OP-SGA) as alternatives to SGA, which is referred hereinafter as Determinant-based SGA (DSGA) to distinguish GSGA and OP-SGA from DSGA in the sense that the former takes advantage of geometric structure of simplex to find SV without DR or SVD required by the latter. Although there are methods with faster computing time proposed in (Xiong et al., 2011) to relieve computational complexity for DSGA, calculating matrix determinant still remains very challenging. The idea of GSGA and OP-SGA comes naturally from geometric structures of simplexes where SV can be calculated by multiplying the base of a simplex with its height as described in Chapter 3.

Suppose that  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k$  are previously found endmembers and  $\mathbf{m}_{k+1}$  is the next endmember to be generated. Assume that  $S(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k)$  is the  $k$ -simplex formed by  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k$  with the volume given by  $V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k)$ . Let  $\mathbf{t}_{k+1}$  be a new vertex to be added to form  $(k+1)$ -simplex,  $S(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k, \mathbf{t}_{k+1})$ . Then its SV  $V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k, \mathbf{t}_{k+1})$  can be calculated in proportional to the value of multiplying  $V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k, \mathbf{t}_{k+1})$  as its base and  $\mathbf{t}_{k+1}^h$  as its height which can be obtained by finding the distance from  $\mathbf{t}_{k+1}$  perpendicular to the base, i.e.,

$V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k, \mathbf{t}_{k+1}) \propto V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k) \cdot \mathbf{t}_{k+1}^h$  where the notation “ $\propto$ ” means “proportional” scaled by a constant. Then the desired endmember  $\mathbf{m}_{k+1}$  is the one that maximizes  $V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k, \mathbf{t}_{k+1})$  over  $\mathbf{t}_{k+1}$ , i.e.,

$$\mathbf{m}_{k+1} = \arg \left\{ \max_{\mathbf{t}_{k+1}} V(\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k, \mathbf{t}_{k+1}) \right\}. \quad (4.1)$$

Calculating SV via height is not a new concept. Most recently, Wang et al. proposed a distance measure to calculate distance from vertex candidate to base and developed an algorithm so called distance-based SGA (Dist-SGA) (Wang et al., 2013). Although this distance measure seems to be a new approach, it is nothing more than finding height of simplex. In (Geng et al., 2014) a similar approach using  $\mathbf{t}_{k+1}^h$  was also proposed for band selection but not for finding endmembers.

One major contribution derived from GSGA and OP-SGA is to take advantage of geometry structure of a simplex to derive various versions of SGA through recursive processes of volume calculations which cannot be found in the literature. More specifically, let us start with two endmembers which is a two-vertex 1-simplex with maximal SV. In this case, two endmembers,  $\mathbf{m}_0$  and  $\mathbf{m}_1$  are found to be two data sample vectors with largest Euclidean distance,  $\mathbf{m}_1 - \mathbf{m}_0$ . Then the third endmembers will be a data sample vector,  $\mathbf{m}_2$  that yields the maximal distance perpendicular to the segment  $\overline{\mathbf{m}_0 \mathbf{m}_1}$ . These three endmembers,  $\mathbf{m}_0$ ,  $\mathbf{m}_1$  and  $\mathbf{m}_2$  are then used to form a 2-simplex as a triangle  $\Delta(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2)$  to find the 4<sup>th</sup> endmembers  $\mathbf{m}_3$  via (4.1). The same procedure is repeatedly over and over again until it reaches the desired number of endmembers,  $p$  and finds the  $p^{\text{th}}$  endmember,  $\mathbf{m}_{p-1}$ . In this chapter, two different approaches are used in SV calculation to find the height  $\mathbf{t}_{k+1}^h$ . One is to apply Gram-

Schmidt Orthogonalization Process (GSOP) whereas the other is to apply Orthogonal Projection (OP) to accomplish the goal.

#### 4.2 Gram-Schmidt Orthogonalization Process for Finding Heights

Suppose that for each  $0 \leq k \leq p-1$  a simplex,  $S_k$  with the  $k+1$  vertices is specified by  $k+1$  data sample vectors,  $\{\mathbf{m}_i\}_{i=0}^k$ . One of key elements in implementing GSGA is to find the maximal height,  $h_k^{\text{GSGA}}$  and its corresponding data sample vector  $\mathbf{m}_k^{\text{GSGA}}$  so that the  $S_k(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}, \mathbf{m}_k^{\text{GSGA}})$  formed by  $S_{k-1}(\mathbf{m}_0, \dots, \mathbf{m}_{k-1})$  with adding  $\mathbf{m}_k^{\text{GSGA}}$  as the  $k^{\text{th}}$  vertex yields the maximal SV. According to (3.4) and (4.1),

$$\begin{aligned} & V\left(S_k\left(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}, \mathbf{m}_k^{\text{GSGA}}\right)\right) \\ &= \max_{\mathbf{m}_k} V\left(S_k\left(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}, \mathbf{m}_k\right)\right). \quad (4.2) \\ &= \frac{h_k^{\text{GSGA}}}{k} \cdot V\left(S_{k-1}\left(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}\right)\right) \end{aligned}$$

The optimal solution to (4.2) can be found by the well-known Gram-Schmidt Orthogonalization Process (GSOP) in the following procedure.

*GSOP for Finding  $h_j^{\text{GSGA}}$*

1. Initial condition:

$$\tilde{\mathbf{m}}_1 = \mathbf{m}_1 - \mathbf{m}_0 \text{ and } \bar{\mathbf{u}}_1 = \frac{\tilde{\mathbf{m}}_1}{\left(\tilde{\mathbf{m}}_1^T \tilde{\mathbf{m}}_1\right)^{1/2}} = \frac{\tilde{\mathbf{m}}_1}{\|\tilde{\mathbf{m}}_1\|}.$$

2. For the given target  $\mathbf{m}_k$  for  $k \geq 1$  find

$$\tilde{\mathbf{m}}_k = \mathbf{m}_k - \mathbf{m}_0 \quad (4.3)$$

and the orthonormal vector  $\bar{\mathbf{u}}_k$  to the space linearly spanned by  $\bar{\mathbf{u}}_1, \bar{\mathbf{u}}_2, \dots, \bar{\mathbf{u}}_{k-1}$  by

$$\begin{aligned}
\bar{\mathbf{m}}_k &= \tilde{\mathbf{m}}_k - \sum_{i=1}^{k-1} \frac{\langle \tilde{\mathbf{m}}_k, \tilde{\mathbf{m}}_i \rangle}{\langle \tilde{\mathbf{m}}_i, \tilde{\mathbf{m}}_i \rangle} \tilde{\mathbf{m}}_i \\
&= \tilde{\mathbf{m}}_k - \sum_{i=1}^{k-1} \langle \tilde{\mathbf{m}}_k, \bar{\mathbf{u}}_i \rangle \bar{\mathbf{u}}_i \\
&= \tilde{\mathbf{m}}_k - \sum_{i=1}^{k-1} (\tilde{\mathbf{m}}_k^T \bar{\mathbf{u}}_i) \bar{\mathbf{u}}_i \\
&= \tilde{\mathbf{m}}_k - \sum_{i=1}^{k-1} \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T \tilde{\mathbf{m}}_k
\end{aligned} \tag{4.4}$$

where

$$\bar{\mathbf{u}}_k = \frac{\bar{\mathbf{m}}_k}{(\bar{\mathbf{m}}_k^T \bar{\mathbf{m}}_k)^{1/2}} = \frac{\bar{\mathbf{m}}_k}{\|\bar{\mathbf{m}}_k\|}. \tag{4.5}$$

3. The maximal height of GSGA,  $h_k^{\text{GSGA}}$  can be found and computed by

$$\tilde{\mathbf{m}}_k^{\text{GSGA}} = \arg \left\{ \max_{\tilde{\mathbf{r}}} \tilde{\mathbf{r}}^T \bar{\mathbf{u}}_k \right\} \tag{4.6}$$

$$h_k^{\text{GSGA}} = \max_{\tilde{\mathbf{r}}} \tilde{\mathbf{r}}^T \bar{\mathbf{u}}_k = \tilde{\mathbf{m}}_k^{\text{GSGA}} \bar{\mathbf{u}}_k \tag{4.7}$$

$$\mathbf{m}_k^{\text{GSGA}} = \tilde{\mathbf{m}}_k^{\text{GSGA}} + \mathbf{m}_0 \tag{4.8}$$

where  $\tilde{\mathbf{r}} = \mathbf{r} - \mathbf{m}_0$ .

Check if  $k = p-1$ . If yes, the algorithm is terminated. Otherwise, let  $k \leftarrow k+1$  and go to step 2.

### 4.3 Geometric Simplex Growing Algorithm (GSGA)

#### GSGA

1. Initial Conditions:

Find two data sample vectors with the maximal segment with two endpoints specified by  $\mathbf{m}_0$  and  $\mathbf{m}_1$ , denoted by  $\mathbf{m}_0^{\text{GSGA}}$  and  $\mathbf{m}_1^{\text{GSGA}}$  respectively. Set  $k = 1$  and define  $\tilde{\mathbf{m}}_1^{\text{GSGA}} = \mathbf{m}_1^{\text{GSGA}} - \mathbf{m}_0^{\text{GSGA}}$ .

2. For each  $1 < k \leq p-1$  implementing GSOP to find  $h_k^{\text{GSGA}}$  by (4.7) via

$$\tilde{\mathbf{m}}_k^{\text{GSGA}} = \arg \left\{ \max_{\tilde{\mathbf{r}}} \tilde{\mathbf{r}}^T \bar{\mathbf{u}}_k \right\} \text{ using (4.6).}$$

3. Find  $\mathbf{m}_k^{\text{GSGA}} = \tilde{\mathbf{m}}_k^{\text{GSGA}} + \mathbf{m}_0$ .

4. The set of  $\left\{ \mathbf{m}_k^{\text{GSGA}} \right\}_{k=1}^p$  obtained in step 3 is the desired set of endmembers

generated by GSGA. In addition,  $V \left( S_k \left( \mathbf{m}_0^{\text{GSGA}}, \dots, \mathbf{m}_{k-1}^{\text{GSGA}}, \mathbf{m}_k^{\text{GSGA}} \right) \right)$  calculated by

(4.2) is the maximal SV produced by the simplex with its vertices specified by

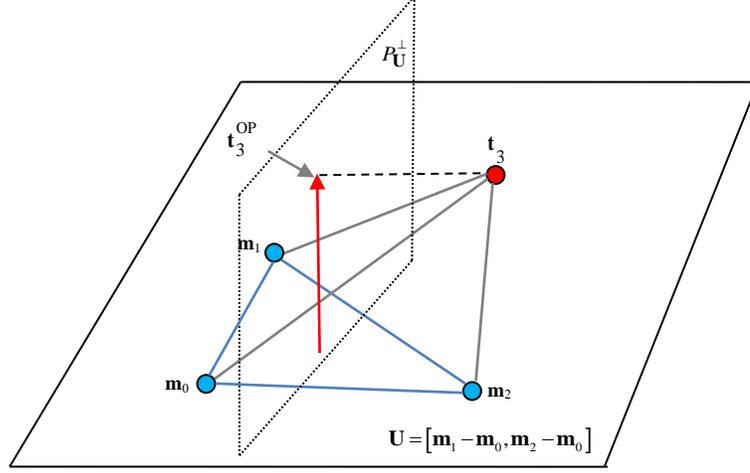
$$\left\{ \mathbf{m}_k^{\text{GSGA}} \right\}_{k=0}^{p-1}.$$

According to the above algorithm GSGA is actually a sequence of the following processes.

$$\begin{aligned}
 & \mathbf{m}_{k-1}^{\text{GSGA}} \text{ (} k^{\text{th}} \text{ vertex/endmember)} \\
 & \xrightarrow{(14)} \tilde{\mathbf{m}}_{k-1}^{\text{GSGA}} = \mathbf{m}_{k-1}^{\text{GSGA}} - \mathbf{m}_0 \text{ ((} k-1 \text{)th edge vector)} \\
 & \xrightarrow{(15)} \bar{\mathbf{m}}_{k-1}^{\text{GSGA}} \text{ (orthogonalized vector)} \\
 & \xrightarrow{(16)} \bar{\mathbf{u}}_{k-1}^{\text{GSGA}} \text{ (orthonormalized vector)} \\
 & \xrightarrow{(17)} \tilde{\mathbf{m}}_k^{\text{GSGA}} \xrightarrow{(18)} h_k^{\text{GSGA}} \text{ (} k^{\text{th}} \text{ height)} \\
 & \xrightarrow{(19)} \mathbf{m}_k^{\text{GSGA}} = \tilde{\mathbf{m}}_k^{\text{GSGA}} + \mathbf{m}_0 \text{ ((} k+1 \text{)th vertex/endmember)}
 \end{aligned} \tag{4.9}$$

#### 4.4 Orthogonal Projection Approach to Find Heights

As it is shown in Fig. 4.1, finding the height of a simplex is equivalent to finding the Orthogonal Projection (OP) onto the hyperplane linearly spanned by its base.



**Figure 4.1.** Interpretation of finding heights via OP

By virtue of (3.8) and (4.1) a general expression can be further represented by

$$V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}, \mathbf{t}_k) = \frac{1}{k!} \left| \det \left[ \tilde{\mathbf{m}}_1 \tilde{\mathbf{m}}_2 \cdots \tilde{\mathbf{m}}_{k-1} \tilde{\mathbf{t}}_k \right] \right| = \frac{1}{k} \mathbf{t}_k^{\text{OP}} \cdot V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}) \quad (4.10)$$

for  $0 \leq k \leq p-1$  where  $\mathbf{t}_k$  is the  $k^{\text{th}}$  endmember to be determined,  $\tilde{\mathbf{t}}_k = \mathbf{t}_k - \mathbf{m}_0$  and  $\mathbf{t}_k^{\text{OP}}$  is the OP of  $\mathbf{t}_k$  orthogonally projected on the hyperplane spanned by points  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_{k-1}$  in geometric space. It is worth noting that  $\mathbf{t}_k^{\text{OP}}$  is the same as the OP of  $\tilde{\mathbf{t}}_k = \mathbf{t}_k - \mathbf{m}_0$  orthogonally projected on the hyperplane  $\langle \tilde{\mathbf{U}}_{k-1} \rangle$  with  $\tilde{\mathbf{U}}_{k-1} = [\tilde{\mathbf{m}}_1 \tilde{\mathbf{m}}_2 \cdots \tilde{\mathbf{m}}_{k-1}]$  and  $\tilde{\mathbf{m}}_k = \mathbf{m}_k - \mathbf{m}_0$  for all  $1 \leq k \leq p-1$ . Also since  $\mathbf{m}_0$  is a constant vector,  $\|\tilde{\mathbf{t}}_k^{\text{OP}}\| = \|(\mathbf{t}_k - \mathbf{m}_0)^{\text{OP}}\|$ .

In order to find the  $k^{\text{th}}$  endmember  $\mathbf{m}_k$  we use (4.10) to find a data sample vector  $\tilde{\mathbf{m}}_k = \mathbf{m}_k - \mathbf{m}_0$  that yields the maximal SV of (3.2), i.e.

$$\tilde{\mathbf{m}}_k = \arg \left\{ \max_{\mathbf{t}_k} V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}, \mathbf{t}_k) \right\} = \arg \left\{ \max_{\tilde{\mathbf{t}}_k} \text{OP}(\tilde{\mathbf{t}}_k) \right\} \cdot V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}) \quad (4.11)$$

which is equivalent to find a target data sample vector  $\mathbf{t}_k$  producing the maximal OP

in the space,  $\langle \tilde{\mathbf{U}}_{k-1} \rangle^\perp$  perpendicular to the linear subspace  $\langle \tilde{\mathbf{U}}_{k-1} \rangle$  spanned by the edge vectors of previously found  $k$  endmembers,  $\mathbf{m}_0, \dots, \mathbf{m}_{k-1}$ . Interestingly, solving (4.11) is equivalent to solving

$$\tilde{\mathbf{m}}_k = \arg \left\{ \max_{\tilde{\mathbf{t}}_k} \|\mathbf{t}_k^{\text{OP}}\| \right\} = \arg \left\{ \max_{\tilde{\mathbf{r}} \in \langle \tilde{\mathbf{U}}_{k-1} \rangle^\perp} \tilde{\mathbf{r}}^T \tilde{\mathbf{r}} \right\} \quad (4.12)$$

which is in turn to find

$$\tilde{\mathbf{m}}_k = \arg \left\{ \max_{\tilde{\mathbf{t}}_k} \|\mathbf{t}_k^{\text{OP}}\| \right\} = \arg \left\{ \max_{\tilde{\mathbf{r}}} \left\| P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right\| \right\} \quad (4.13)$$

with  $P_{\tilde{\mathbf{U}}_{k-1}}^\perp = \mathbf{I} - \tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^\#$  and  $\tilde{\mathbf{U}}_{k-1}^\# = \left( \tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1} \right)^{-1} \tilde{\mathbf{U}}_{k-1}^T$  being the pseudo-inverse of  $\tilde{\mathbf{U}}_{k-1}$ ,  $\left\| P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right\|^2 = \left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right)^T P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} = \tilde{\mathbf{r}}^T P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}}$  and  $\left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \right)^T = P_{\tilde{\mathbf{U}}_{k-1}}^\perp = \left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \right)^2$  being idempotent. In other words, (4.1) can be solved via (4.12-4.13) by

$$\begin{aligned} \tilde{\mathbf{m}}_k &= \arg \max_{\tilde{\mathbf{r}}} \left\{ V(\tilde{\mathbf{m}}_1, \dots, \tilde{\mathbf{m}}_{k-1}, \tilde{\mathbf{r}}) \right\} \\ &= \arg \max_{\tilde{\mathbf{r}}} \left\{ (1/k) \|\tilde{\mathbf{r}}^{\text{OP}}\| \cdot V(\tilde{\mathbf{m}}_1, \dots, \tilde{\mathbf{m}}_{k-1}) \right\} \\ &= \arg \max_{\tilde{\mathbf{r}} \in \langle \tilde{\mathbf{U}}_{k-1} \rangle^\perp} \left\{ \tilde{\mathbf{r}}^T \tilde{\mathbf{r}} \right\} = \arg \max_{\tilde{\mathbf{r}}} \left\{ \left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right)^T \left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right) \right\} \end{aligned} \quad (4.14)$$

While we can find SV via geometric structure as (3.3) we can further derive a recursive equation that reduces one dimension at a time for (21) recursively as

$$\begin{aligned} V(\mathbf{m}_0, \dots, \mathbf{m}_k) &= (1/k) \|\tilde{\mathbf{m}}_k^{\text{OP}}\| \cdot V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}) \\ &= ((k-j)!/k!) \left[ \prod_{i=0}^{j-1} \|\tilde{\mathbf{m}}_{k-i}^{\text{OP}}\| \right] \cdot V(\mathbf{m}_0, \dots, \mathbf{m}_{k-j}) \end{aligned} \quad (4.15)$$

for  $0 \leq j \leq k-1$  until  $j = k-1$  (4.12) becomes

$$\begin{aligned} V(\mathbf{m}_0, \dots, \mathbf{m}_k) &= (1/k) \|\tilde{\mathbf{m}}_k^{\text{OP}}\| \cdot V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}) \\ &= (1/k!) \left[ \prod_{i=0}^{k-1} \|\tilde{\mathbf{m}}_{k-i}^{\text{OP}}\| \right] \cdot V(\mathbf{m}_0, \mathbf{m}_1) \end{aligned} \quad (4.16)$$

in which case  $V(\mathbf{m}_0, \mathbf{m}_1)$  is 1-simplex with two vertices whose volume is the length of

the line segment connecting  $\mathbf{m}_0$  and  $\mathbf{m}_1$ ,  $\tilde{\mathbf{m}}_1 = \mathbf{m}_1 - \mathbf{m}_0$ . It should be noted that OP-SGA does not directly use (2.1) to calculate SV. Instead it makes use of (3.3) and (4.16) without applying DR.

#### 4.5 Orthogonal Projection-based Simplex Volume Algorithm (OP-SGA)

##### *OP-SGA*

##### 1. Initial Conditions:

Find two data sample vectors with the maximal segment with two endpoints specified by  $\mathbf{m}_0$  and  $\mathbf{m}_1$ , denoted by  $\mathbf{m}_0^{\text{OP-SGA}}$  and  $\mathbf{m}_1^{\text{OP-SGA}}$  respectively. Set  $k = 1$  and define  $\tilde{\mathbf{m}}_1^{\text{OP-SGA}} = \mathbf{m}_1^{\text{OP-SGA}} - \mathbf{m}_0^{\text{OP-SGA}}$ .

##### 2. At $k > 1$ find $\mathbf{m}_k^{\text{OP-SGA}}$ that solves (4.14).

##### 3. Stopping rule:

If  $k < p - 1$ , then  $k \leftarrow k + 1$  and go to step 2. Otherwise, the final set of

$\{\mathbf{m}_0^{\text{OP-SGA}}, \mathbf{m}_1^{\text{OP-SGA}}, \dots, \mathbf{m}_{p-1}^{\text{OP-SGA}}\}$  is the desired  $p$  endmembers.

#### 4.6 Experiments

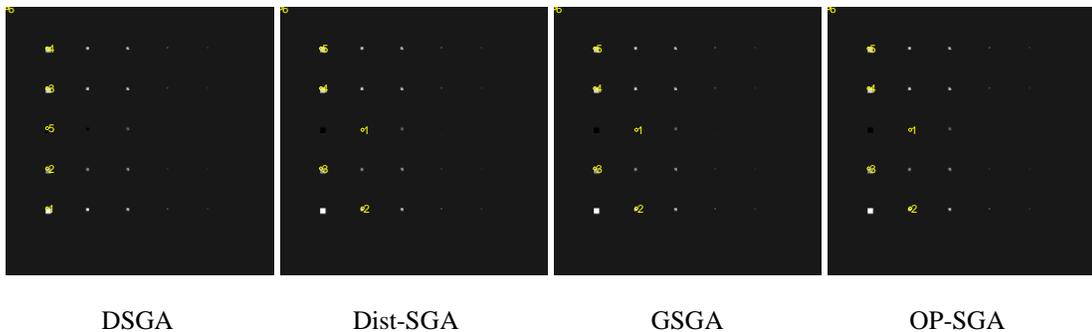
Three different sets of hyperspectral images mentioned in Chapter 2 are used to conduct experiments in this section. In the experiments, endmember pixels found by the four EFAs, GSGA, OP-SGA, DSGA and Dist-SGA are shown in Figs 4.2-4.10 to conduct a comparative study. If we look more closely into the development of Dist-SGA, the used distance is actually the orthogonal distance to existing endmember subspace which is exactly orthogonal projection (OP). Accordingly, it is not a surprise to see from the results that GSGA, OP-SGA and Dist-SGA produced

identical endmember pixels whereas DSGA produced completely different results. The reason that DSGA produced different endmembers from the other algorithms is mainly caused by the fact that DSGA computed SV via determinant with SVD to find endmember pixels which is a different strategy from OP-SGA, GSGA and Dist-SGA using OP.

#### 4.6.1 Synthetic Image Experiments

In this subsection, the experimental results on six scenarios image data are conducted. Firstly, the synthetic data images were constructed using Cuprite radiance spectra as it is described in Chapter 2. For scenarios TI2, TI3, TE2, and TE3, an additive Gaussian noise with  $SNR = 20$  is added to target panel pixels or background pixels.

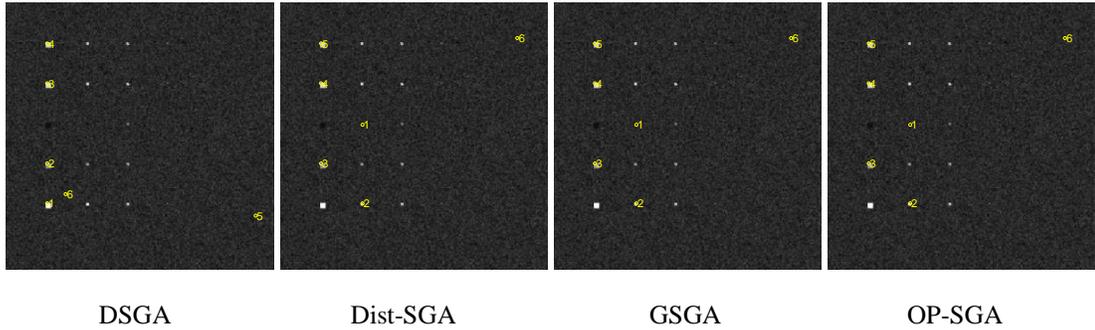
As a result of having full knowledge for the dataset, the performances of variants of SGA can be evaluated based on the provided ground-truth. Due to the effect caused by background, the experiments were conducted by assuming the number of endmembers  $p = 6$  present in the scene with one endmember used to account for background signature. In the following results, open yellow circles represent found endmembers and the number used to label each circled pixels denotes its order.



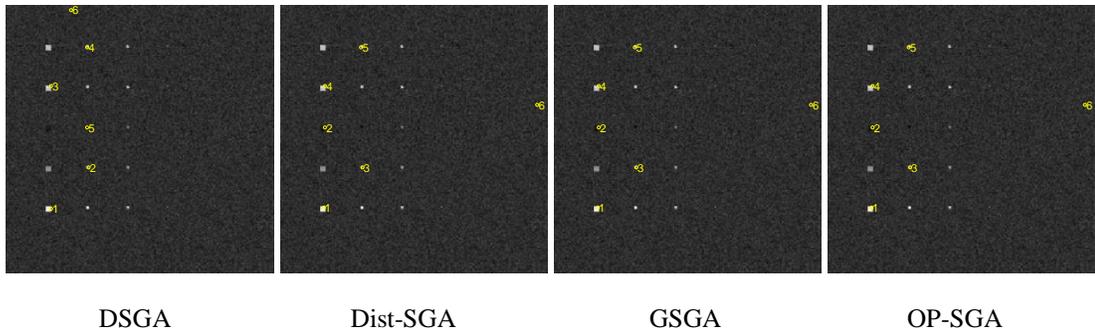
**Figure 4.2.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TI1

Figs. 4.2-4.4 show the result of proposed GSGA and OP-SGA comparing to DSGA and Dist-SGA on first type of synthetic data. As shown in Fig. 4.2, all four EFAs can extract all five panel pixels successively.

With introducing of noise in background to achieve SNR = 20 on scenario TI2, only DSGA missed target panel pixel C which has very similar spectrum with background signature. It would be interesting to know that DSGA is failed to find this target panel pixel C although the desired number was set to be three times of the total number of distinct signatures in synthetic image, i.e.  $p = 18$ . This result demonstrated that noisy background increases the interference of background signature to mineral spectral signature C. Interestingly, if noise is also introduced into panel pixels the contaminated background is no longer dominant.



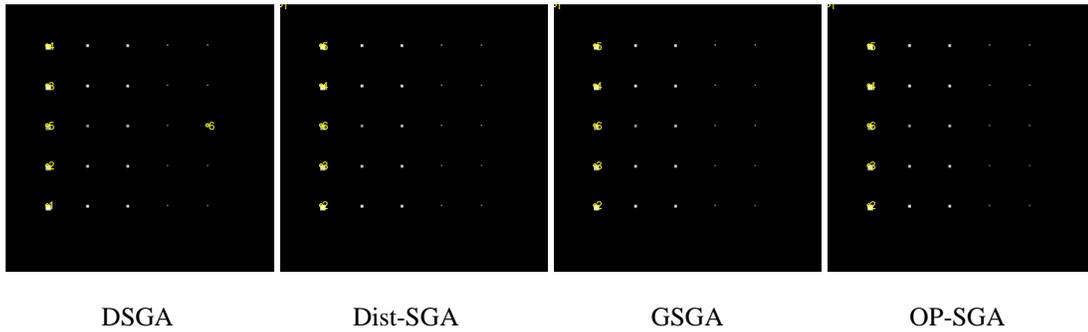
**Figure 4.3.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TI2



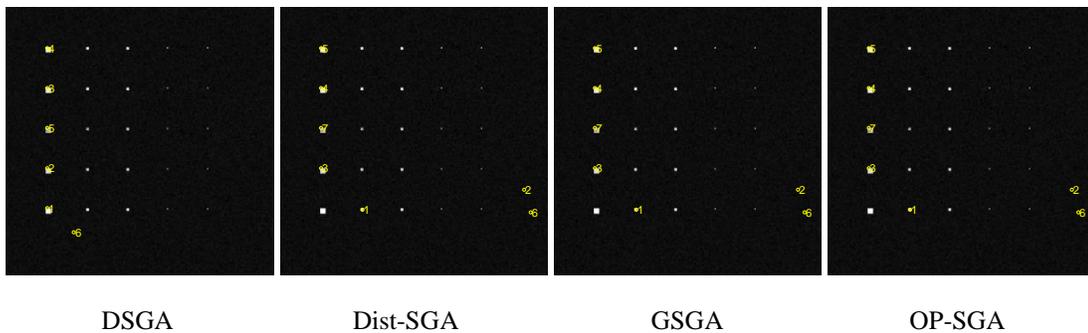
**Figure 4.4.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TI3

Unlike the performance on scenario TI2, all four variants of SGA successfully extracted all panel pixels corresponding to five mineral spectral signatures on scenario TI3 as shown in Fig. 4.4. Similar results can be seen in Figs. 4.5-4.7 which shows the experimental results on Target Embededness scenarios.

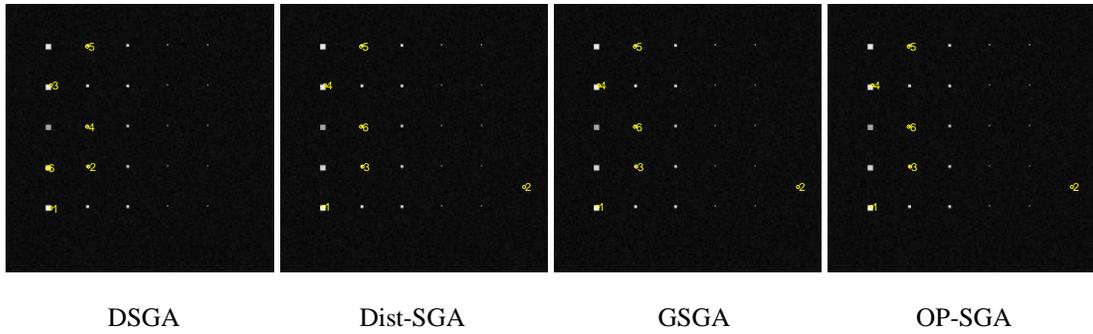
Following the same experiments conducted for scenario TI1, all variants of SGA extracted all panel pixels corresponding to five mineral signatures on scenario TE1. Different from results on scenario TI2, DSGA successfully finds all five panel pixels while noise was introduced to background pixels on TE2. Interestingly, the other three distance-based SGAs, GSGA, OP-SGA, and Dist-SGA, cannot extract panel pixel C while the total number of desired endmembers to be found was set to be six, i.e.  $p = 6$ . The panel pixel C then was found as 7<sup>th</sup> endmember.



**Figure 4.5.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TE1



**Figure 4.6.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TE2



**Figure 4.7.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on TE3

While noise is also introduced into panel pixels, all four variants of SGAs can extract the five panel pixels successfully. But it shall be noticed that DSGA finds two pixels in 4<sup>th</sup> row that are corresponding to the same mineral spectral signature K. This endmember result of DSGA demonstrated the numerical instability issue encountered in volume calculation via SVD that is mentioned in Chapter 3.

#### 4.6.2 Real Image Experiments

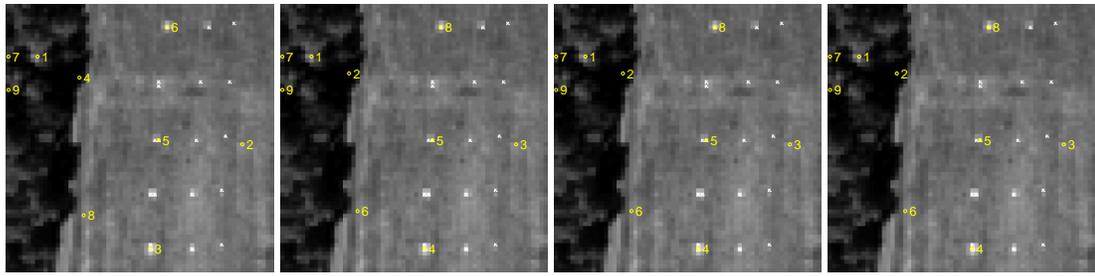
In this subsection, two real image data, HYDICE and Cuprite data, are used to conduct experiments. The performances of variants of SGAs are evaluated by the number of extracted panels for HYDICE image. More performance evaluation on Cuprite data will be discussed later in Chapter 5.

##### 4.6.2.1 HYDICE Image

Unlike the synthetic dataset with complete knowledge, the number of endmembers, which is unknown in HYDICE image, must be determined prior to analysis for real image data processing. According to previous study in the literature, Virtual Dimensionality (VD), coined by Chang (2003) and later published by Chang

and Du (2004), was addressed this issue. The VD estimated for this HYDICE scene by using Harsanyi-Farrand-Chang (HFC) method in (Harsanyi et al., 1994) was 9. Recently, an approach using real target signal sources for VD estimation was proposed (Chang et al., 2014; Chang et al., 2015) and VD was shown to be in the range between 39 and 45. However, when unsupervised fully constrained least squares (UFCLS) in (Heinz and Chang, 2001) was used to find targets, the number of targets was determined as 34. As a result, the values of VD for HYDICE,  $n_{VD}$  was set to be from 9 to 45.

Interestingly, all the four algorithms cannot find all five pure panel pixels corresponding to the five panel signatures when  $n_{VD} = 9$  but can do so when  $n_{VD} = 18$  as shown in Fig. 4.8(b). This number is twice value of  $n_{VD} = 9$  which was also noted in the literature (Chang et al., 2010; Chang et al., 2011). Since these four algorithms are sequential, the endmembers generated by a smaller value of VD are always part of endmember pixels generated by a larger value of VD. Accordingly, the endmember pixels found in Fig. 4.2 are separated into four ranges: (a)  $n_{VD} = 9$ ; (b)  $n_{VD} = 18$ ; (c)  $n_{VD} = 34$  and (d)  $n_{VD} = 45$ .



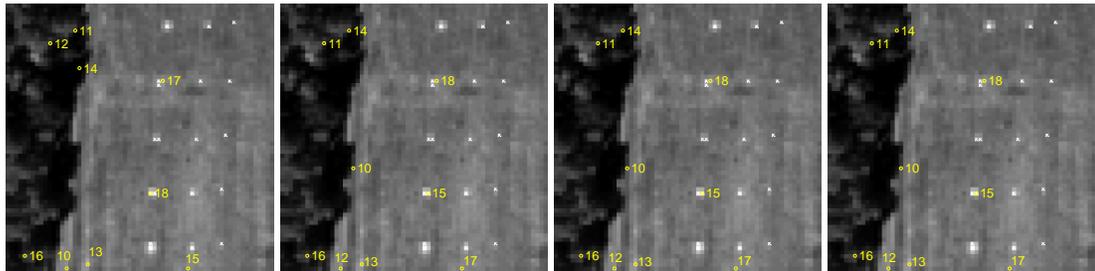
DSGA

Dist-SGA

GSGA

OP-SGA

(a) 9 endmember pixels



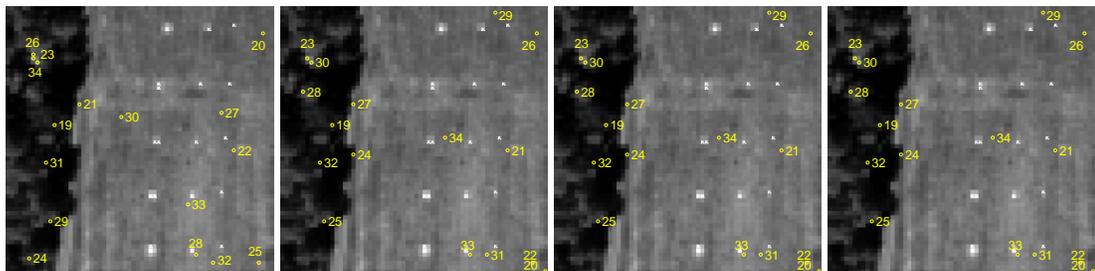
DSGA

Dist-SGA

GSGA

OP-SGA

(b) 10-18 endmember pixels



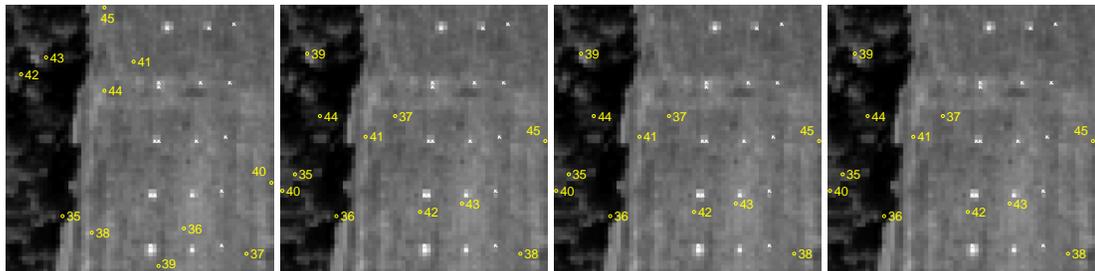
DSGA

Dist-SGA

GSGA

OP-SGA

(c) 19-34 endmember pixels



DSGA

Dist-SGA

GSGA

OP-SGA

(d) 35-45 endmember pixels

**Figure 4.8.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on HYDICE

As expected, GSGA, OP-SGA, and Dist-SGA found identical sets of endmembers as a matter of fact that the maximal OP of a vertex perpendicular to a simplex base is indeed its height. Moreover, while  $n_{VD} = 18$  all the four variants of SGA extracted five target panels successfully.

#### 4.6.2.2 AVIRIS Cuprite data

In this sub-subsection, another real image scene to be used for experiments is a well-known Airborne Visible Infrared Spectrometer (AVIRIS) image scene mentioned earlier in Chapter 2. There are two types of Cuprite datasets, reflectance data and radiance data, were used for validate algorithms. Since there is no available prior knowledge about spatial locations of endmembers we must rely on an unsupervised means of identifying if an extracted target pixel is an endmember. To address this issue the following Endmember Identification Algorithm (EIA) developed in (Chang et al., 2014) was used for this purpose.

##### Endmember Identification Algorithm (EIA)

Assume that  $\{\mathbf{t}_j\}_{j=1}^J$  are  $J$  extracted target pixels and  $\{\mathbf{m}_i\}_{i=1}^p$  are known  $p$  ground-truth endmembers.

1. Cluster all extracted pixels,  $\{\mathbf{t}_j\}_{j=1}^J$  into  $p$  endmember classes  $\{C_j\}_{j=1}^p$  according to

the following rule

$$\mathbf{t}_j \in C_{j^*} \Leftrightarrow j^* = \arg \left\{ \min_{1 \leq i \leq p} \text{SAM}(\mathbf{t}_j, \mathbf{m}_i) \right\} \quad (4.17)$$

where the Spectral Angle Mapper (SAM) is a spectral similarity measure (Chang et al., 2003).

2. For each of endmembers,  $\mathbf{m}_i$  find the target pixel  $\mathbf{t}_{i^*}$  among all the extracted pixels  $\{\mathbf{t}_j\}_{j=1}^J$  which is closest to  $\mathbf{m}_i$  by

$$i^* = \arg \left\{ \min_{1 \leq j \leq J} \text{SAM}(\mathbf{t}_j, \mathbf{m}_i) \right\} \quad (4.18)$$

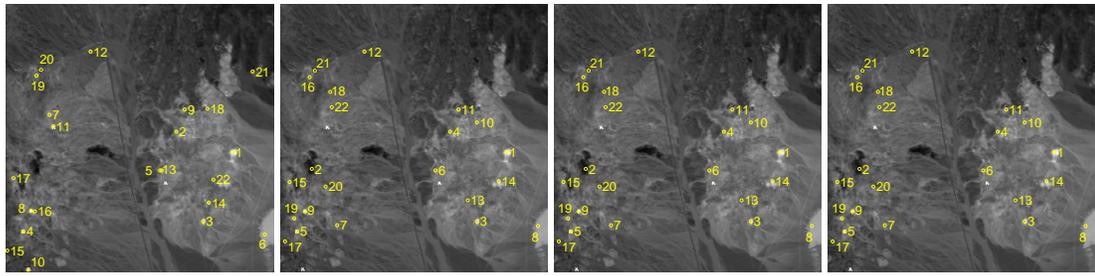
3. Find all target pixels which satisfy

$$i^* = \arg \left\{ \min_{1 \leq j \leq J} \text{SAM}(\mathbf{t}_j, \mathbf{m}_i) \right\} \Leftrightarrow \mathbf{t}_{i^*} \in C_i \quad (4.19)$$

All those target pixels found in step 3 are extracted as endmembers.

According to (Chang et al., 2014), the VD estimated by the HFC method was  $n_{\text{VD}} = 22$  for reflectance data with twice of VD values  $2n_{\text{VD}} = 44$ , and  $n_{\text{VD}} = 15$  for radiance data with twice of VD values  $2n_{\text{VD}} = 30$ . Using these values along with the values that will be discussed later in Chapter 5, the experiments were conducted for three ranges, (a)  $n_{\text{VD}} = 22$ ; (b)  $n_{\text{VD}} = 46$ ; (c)  $n_{\text{VD}} = 75$ ; for both reflectance and radiance data. The reason that we could do so is because DSGA, Dist-SGA, GSGA, and OP-SGA are sequential algorithms and the endmember pixels generated by a smaller value of  $n_{\text{VD}}$  are always part of endmember pixels generated by a larger value of  $n_{\text{VD}}$ .

Figs. 4.9-4.10 show the endmembers found by DSGA, Dist-SGA, GSGA, and OP-SGA for Cuprite reflectance data and Cuprite radiance data, respectively. The pixels marked by yellow circles represent the endmembers found by the algorithms along with numerals used to indicate their found orders.



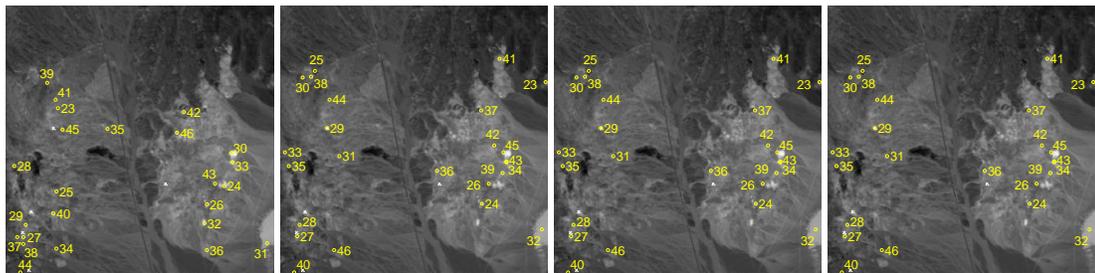
DSGA

Dist-SGA

GSGA

OP-SGA

(a) 22 endmember pixels



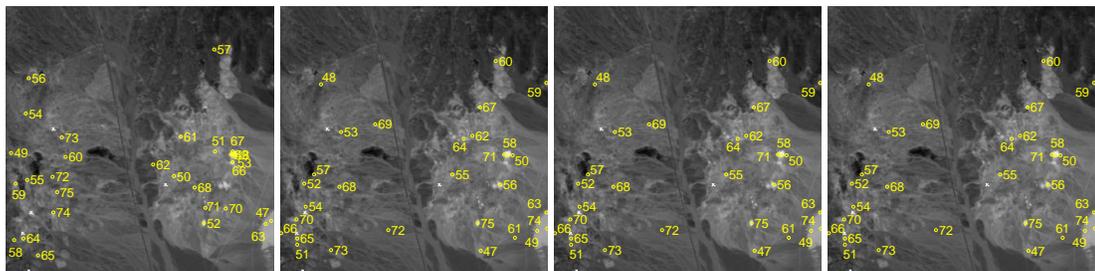
DSGA

Dist-SGA

GSGA

OP-SGA

(b) 23-46 endmember pixels



DSGA

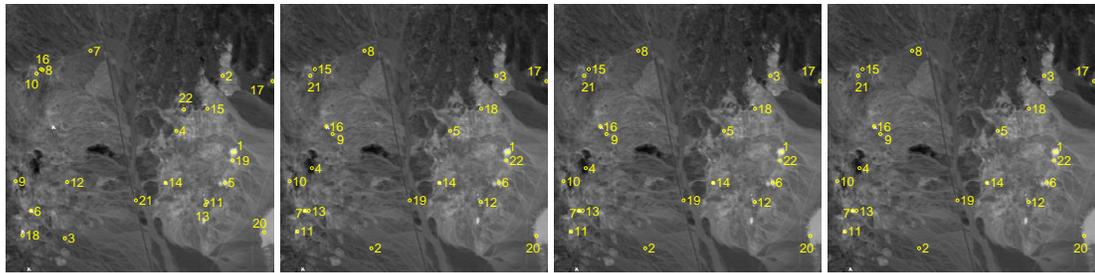
Dist-SGA

GSGA

OP-SGA

(c) 47-75 endmember pixels

**Figure 4.9.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on Cuprite reflectance data



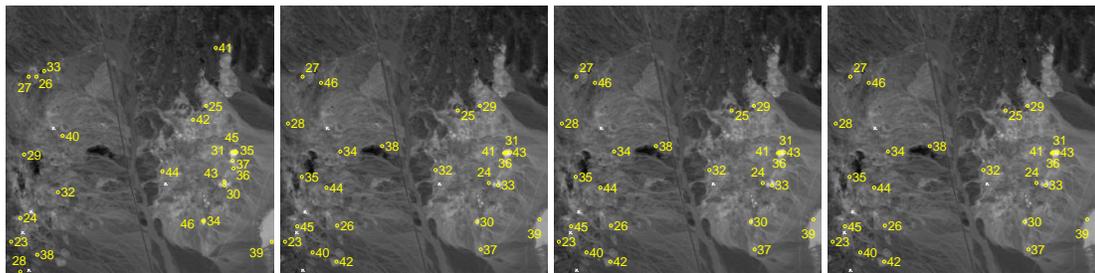
DSGA

Dist-SGA

GSGA

OP-SGA

(a) 22 endmember pixels



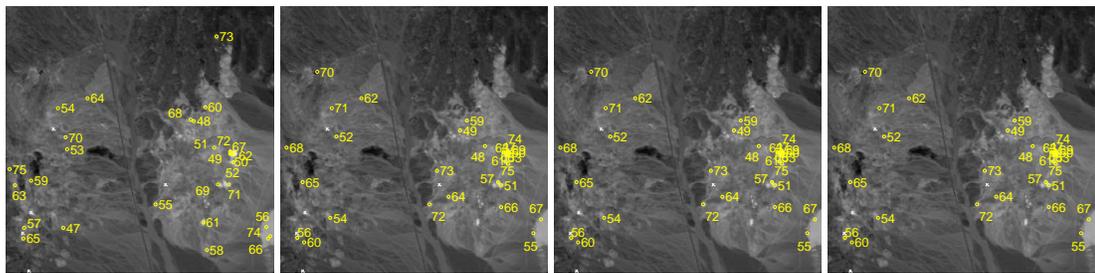
DSGA

Dist-SGA

GSGA

OP-SGA

(b) 23-46 endmember pixels



DSGA

Dist-SGA

GSGA

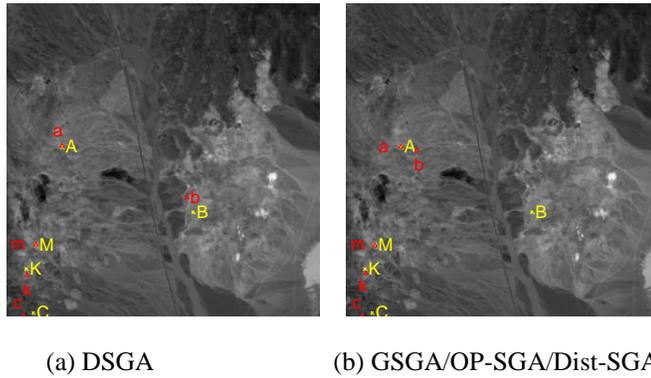
OP-SGA

(c) 47-75 endmember pixels

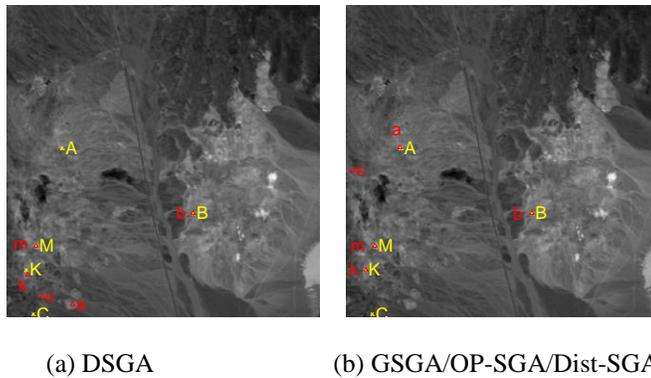
**Figure 4.10.** Endmember pixels found by DSGA, Dist-SGA, GSGA and OP-SGA on Cuprite radiance data

In order to identify which pixels among the found endmembers in Figs. 4.9-4.10 are desired endmember pixels, the EIA described above was used for this purpose and the spatial locations of these desired endmember pixels are shown in Figs. 4.11-4.12 where the pixels marked by the lower case of “a”, “b”, “c”, “k”, “m” with red

triangles are the desired endmember pixels found by EIA that correspond to the five ground-truth mineral endmembers marked by the upper cases of “A”, “B”, “C”, “K”, “M” with yellow crosses symbols in the sense of spectral similarity measured by SAM and spectral information divergence (SID) (Chang, 2003).



**Figure 4.11.** Endmember pixels found by EIA compared to ground-truth pixels for Cuprite reflectance data



**Figure 4.12.** Endmember pixels found by EIA compared to ground-truth pixels for Cuprite radiance data

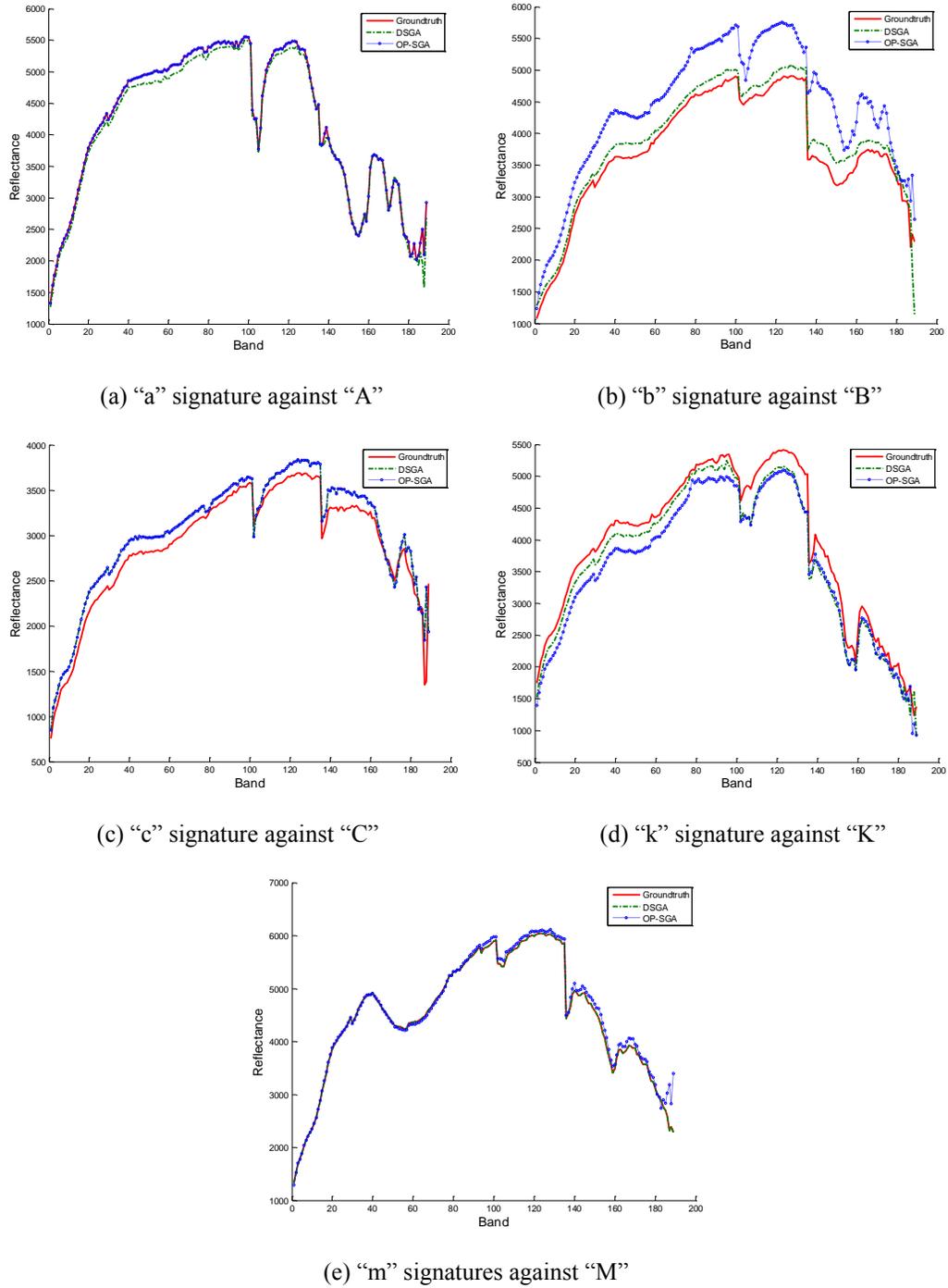
Table 4.1 further tabulates the five desired endmember pixels found by EIA among DSGA-found target pixels,  $\{t_j^{DSGA}\}$  and all three distance-based algorithms,

hereinafter represented by OP-SGA, -found target pixels,  $\{t_j^{OP-SGA}\}$  where the subscript  $j$  indicates the order of a particular target pixel was found.

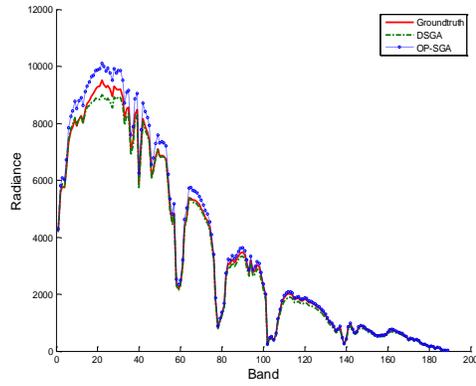
**Table 4.1.** Endmember pixels found by EIA

	Cuprite	A	B	C	K	M
DSGA	reflectance	$t_{11}^{DSGA}$	$t_{13}^{DSGA}$	$t_{44}^{DSGA}$	$t_{64}^{DSGA}$	$t_8^{DSGA}$
	radiance	$t_{12}^{DSGA}$	$t_{14}^{DSGA}$	$t_{28}^{DSGA}$	$t_{103}^{DSGA}$	$t_6^{DSGA}$
OP-SGA	reflectance	$t_{29}^{OP-SGA}$	$t_{53}^{OP-SGA}$	$t_{40}^{OP-SGA}$	$t_{27}^{OP-SGA}$	$t_9^{OP-SGA}$
	radiance	$t_{16}^{OP-SGA}$	$t_{14}^{OP-SGA}$	$t_{68}^{OP-SGA}$	$t_{11}^{OP-SGA}$	$t_7^{OP-SGA}$

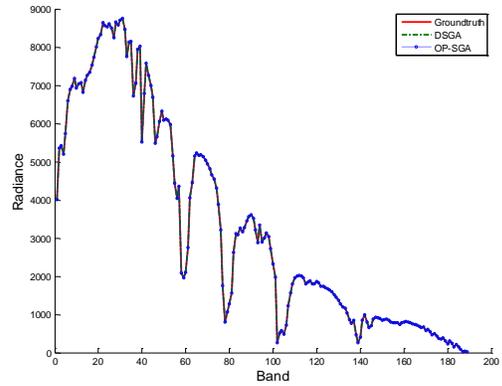
As we can see from Table 4.1 for the reflectance data, it required by DSGA at least 64 target pixels to find the last mineral signature K to complete all the five mineral signatures, while 53 target pixels by OP-SGA to find the last mineral signature B. For radiance data it took DSGA at least 103 target pixels to find the last mineral signature K to complete all the five mineral signatures, while 68 target pixels by OP-SGA to complete all the five mineral signatures with the mineral signature C being the last one to be found. Nonetheless, both DSGA and OP-SGA found the same first mineral signature which was M because the spectrum of M is probably the most distinct among the five mineral signatures which can be seen in Figs. 2.1(c)-2.1(d). Once the spatial locations of the five desired endmember pixels were found in Figs. 4.11-4.12 we can further perform a comparative spectral analysis between EIA-identified endmember pixels in Table 4.1 and the ground truth pixels in Fig. 2.1(b). For each of spectral signatures of the five mineral signatures, A, B, C, K, M both in reflectance data and radiance data, Figs. 4.13(a-e)-4.13(a-e) plot 3 spectra, ground-truth pixel spectrum, spectrum of  $t_j^{DSGA}$ , and spectrum of  $t_j^{OP-SGA}$  identified in Table 4.1, for comparison.



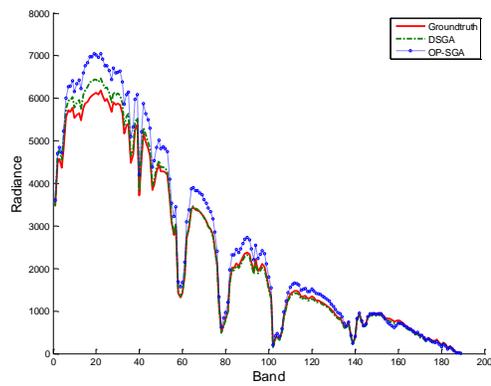
**Figure 4.13.** Comparative plots of spectral signatures found by DSGA and OP-SGA on Cuprite reflectance data



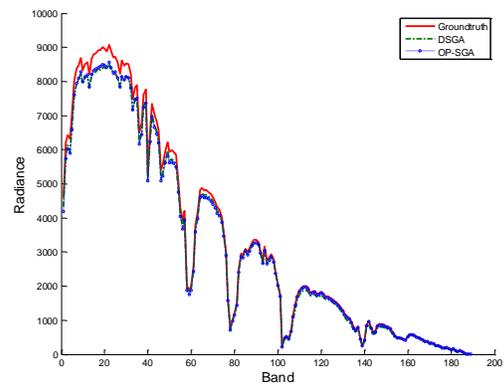
(a) "a" signatures against "A"



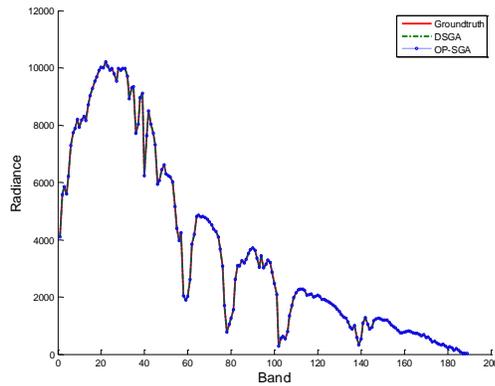
(b) "b" signatures against "B"



(c) "c" signatures against "C"



(d) "k" signatures against "K"



(e) "m" signatures against "M"

**Figure 4.14.** Comparative plots of spectral signatures found by DSGA and OP-SGA on Cuprite radiance data

While the plots of Figs. 4.13 and 4.14 offer an advantage of visual assessment about how close a found endmember pixel to a ground truth pixel it does not provide quantitative measurements on their spectral similarity. Tables 4.2 and 4.3 calculate spectral similarity values of the plots among identified target pixels against the ground truth pixels in Figs. 4.13 and 4.14 where SAM and SID were used as spectral measure.

As we can see from Tables 4.2 and 4.3 the spectral similarity values between DSGA-found and OP-SGA-found pixels compared to the ground-truth pixels were indeed very close even the found pixels by DSGA and OP-SGA were identified by EIA in different locations.

**Table 4.2.** SAM/SID of the closet target pixels with ground-truth by DSGA and OP-SGA on Cuprite reflectance data

SAM SID	(A,a)	(B,b)	(C,c)	(K,k)	(M,m)
DSGA	0.0167 0.0002	0.0334 0.0009	0.0379 0.0009	0.0341 0.0007	0 0
OP-SGA	0 0	0.0497 0.0012	0.0379 0.0009	0.0304 0.0006	0.0264 0.0005

**Table 4.3.** SAM/SID of the closet target pixels with ground-truth by DSGA and OP-SGA on Cuprite radiance data

SAM SID	(A,a)	(B,b)	(C,c)	(K,k)	(M,m)
DSGA	0.0205 0.0003	0 0	0.0247 0.0006	0.0123 0.0001	0 0
OP-SGA	0.0098 0.0001	0 0	0.0253 0.0010	0.0100 0.0001	0 0

#### 4.7 Conclusions

Due to impracticality of exhaustively search procedure, determinant-based simplex growing algorithm (DSGA) has become a good alternative to N-FINDR in the sense that it finds endmembers sequentially through growing simplexes so as to reduce computing time. However, calculating simplex volume (SV) remains a challenging issue for DSGA in computation of matrix determinants. This chapter proposed two algorithms called geometric simplex growing algorithm (GSGA) and orthogonal projection-based simplex growing algorithm (OP-SGA), both of which calculate SV by taking advantage of geometric structure of a simplex via Gram-Schmidt Orthogonalization process (GSOP) and Orthogonal Projection (OP) operators respectively. With appropriate interpretations GSGA and OP-SGA resolve several inherent issues in DSGA and N-FINDR. First of all, it requires no Dimensionality Reduction (DR). Second, true SV could be derived without suffering from the numerical issues when it comes to calculating the volume of a simplex in a high dimensional space. Last but not least, the computing time is significantly reduced since finding heights of all data sample vectors can be done through recursive processes without actually inverting matrices so that there is no need of computing matrix determinants. This great advantage is not applicable to DSGA.

# Chapter 5: RECURSIVE GROWING SIMPLEX VOLUME ANALYSIS (RGSVA)

## 5.1 Introduction

Chapter 4 introduces two new EFAs, GSGA and OP-SGA, as alternatives to DSGA in order to reduce computational complexity. Moreover, they can be considered as a new approach to SV calculation via heights without need of DR. However, both of them have a major drawback that they require finding heights repeatedly in each epoch. More specifically, since in OP-SGA a new endmember is generated in each epoch and  $\mathbf{U}$  is augmented by adding this new endmember,  $P_{\mathbf{U}}^{\perp}$  must be re-implemented over and over again by varying  $\mathbf{U}$  to generate next endmembers. Suppose that in  $k-2$  iteration  $\mathbf{U}_{k-1}$  is a matrix consists of first  $k-1$  endmembers,  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_{k-2}$ , and  $\mathbf{m}_{k-1}$  is newly found endmember. In order to generate endmember  $\mathbf{m}_k$ ,  $P_{\mathbf{U}}^{\perp}$  must be re-calculated by  $P_{\mathbf{U}_k}^{\perp}$  with a new matrix  $\mathbf{U}_k$  which includes  $\mathbf{U}_{k-1}$  and  $\mathbf{m}_{k-1}$ . A similar issue is also encountered in GSGA while performing orthogonalization process on all data samples. To deal with this issue recursive versions of GSGA and OP-SGA are further developed in this chapter, referred to as Recursive GSGA (RGSGA) and Recursive OP-SGA (ROP-SGA). They can be considered as Kalman-like filters which make use of recursive equations to update projections of all data samples via the newly found endmember and  $P_{\mathbf{U}_k}^{\perp}$  via  $P_{\mathbf{U}_{k-1}}^{\perp}$ , respectively, in GSGA and OP-SGA without the needs of re-processing all previous found  $k-1$  endmembers,  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_{k-1}$ . Furthermore, the recursive

equations also provide an effective means of calculating GSV recursively. This advantage would be more significant in hardware design and implementation.

## 5.2 Recursive Equations for GSGA

For  $k > 1$  with the processed data  $\bar{\mathbf{u}}_i$  and a new input data sample vector

$$\tilde{\mathbf{r}}_k = \mathbf{r}_k - \mathbf{m}_0, \quad \text{i.e.} \quad \bar{\mathbf{r}}_k = \tilde{\mathbf{r}}_k - \sum_{i=1}^{k-1} \frac{\langle \tilde{\mathbf{r}}_k, \tilde{\mathbf{m}}_i \rangle}{\langle \tilde{\mathbf{m}}_i, \tilde{\mathbf{m}}_i \rangle} \tilde{\mathbf{m}}_i \quad \text{and} \quad \bar{\mathbf{u}}_i = \frac{\bar{\mathbf{m}}_i}{(\bar{\mathbf{m}}_i^T \bar{\mathbf{m}}_i)^{1/2}} = \frac{\bar{\mathbf{m}}_i}{\|\bar{\mathbf{m}}_i\|},$$

for  $i = 1, \dots, k-1$ . Now, let  $\bar{\mathbf{U}}_{k-1} = [\bar{\mathbf{u}}_1 \bar{\mathbf{u}}_2 \cdots \bar{\mathbf{u}}_{k-1}]$ . Then the orthonormalized vector  $\bar{\mathbf{r}}_k$

can be derived by

$$\begin{aligned} \bar{\mathbf{r}}_k &= \tilde{\mathbf{r}}_k - \sum_{i=1}^{k-1} (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T) \tilde{\mathbf{r}}_k = \tilde{\mathbf{r}}_k - \bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T \tilde{\mathbf{r}}_k \\ &= \tilde{\mathbf{r}}_k - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T \tilde{\mathbf{r}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}}_k \\ &= \bar{\mathbf{r}}_{k-1} - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}}_k \end{aligned} \quad (5.1)$$

where

$$\bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T = \sum_{i=1}^{k-1} (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T) = \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T + \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T. \quad (5.2)$$

## 5.3 Recursive GSGA

### 5.3.1 RGSGA

#### RGSGA

#### 1. Initial Conditions:

Find two data sample vectors with the maximal segment with two endpoints specified by  $\mathbf{m}_0$  and  $\mathbf{m}_1$ , denoted by  $\mathbf{m}_0^{\text{GSGA}}$  and  $\mathbf{m}_1^{\text{GSGA}}$  respectively.

Define  $\tilde{\mathbf{m}}_1^{\text{GSGA}} = \mathbf{m}_1^{\text{GSGA}} - \mathbf{m}_0^{\text{GSGA}}$  and set  $k = 2$ .

2. At  $k^{\text{th}}$  recursion:

a. For each  $1 < k \leq p-1$  use (14) to find  $\tilde{\mathbf{r}}_k = \mathbf{r}_k - \mathbf{m}_0$  and (5.1) to find

$$\begin{aligned}\bar{\mathbf{r}}_k^{\text{GSGA}} &= \left( \mathbf{I} - \bar{\mathbf{U}}_{k-1} \left( \bar{\mathbf{U}}_{k-1} \right)^T \right) \tilde{\mathbf{r}}_k \\ &= \tilde{\mathbf{r}}_k - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T \tilde{\mathbf{r}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}}_k \\ &= \bar{\mathbf{r}}_{k-1}^{\text{GSGA}} - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}}_k\end{aligned}$$

where  $\bar{\mathbf{U}}_{k-2} = [\bar{\mathbf{u}}_1 \bar{\mathbf{u}}_2 \cdots \bar{\mathbf{u}}_{k-2}]$ .

b. Use (4.6) to find  $\bar{\mathbf{m}}_k^{\text{GSGA}} = \arg \left\{ \max_{\tilde{\mathbf{r}}_k} \tilde{\mathbf{r}}_k^T \tilde{\mathbf{r}}_k \right\}$ .

c. Calculate  $h_k^{\text{GSGA}} = \|\bar{\mathbf{m}}_k^{\text{GSGA}}\|$ .

d. Use (4.2) to find the simplex volume that

$$V \left( S_k \left( \bar{\mathbf{m}}_1^{\text{GSGA}}, \dots, \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}, \bar{\mathbf{m}}_k^{\text{GSGA}} \right) \right) = \frac{h_k^{\text{GSGA}}}{k} \cdot V \left( S_{k-1} \left( \bar{\mathbf{m}}_1^{\text{GSGA}}, \dots, \bar{\mathbf{m}}_{k-1}^{\text{GSGA}} \right) \right).$$

e. Update  $\bar{\mathbf{U}}$  with  $\bar{\mathbf{U}}_{k-1} = [\bar{\mathbf{u}}_1 \bar{\mathbf{u}}_2 \cdots \bar{\mathbf{u}}_{k-1}]$  and  $\bar{\mathbf{u}}_k = \frac{\bar{\mathbf{m}}_k}{\|\bar{\mathbf{m}}_k\|}$  by (5.2)

3. Check if  $k = p - 1$ , terminating the algorithm. Otherwise, Let  $k \leftarrow k + 1$  and go back to step 2.

It should be noted that the major difference of RGSGA from GSGA is the use of recursive equations (5.2) in step 6.

### 5.3.2 KF-OSP-GSGA

From (5.1) we can further derive Kalman filter-like equations for GSGA using OSP as follows.

For  $k > 1$  the new endmember  $\bar{\mathbf{m}}_k$  can be found by

$$\begin{aligned}
\bar{\mathbf{m}}_k &= \tilde{\mathbf{m}}_k - \sum_{i=1}^{k-1} (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T) \tilde{\mathbf{m}}_k = \tilde{\mathbf{m}}_k - \bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T \tilde{\mathbf{m}}_k \\
&= \tilde{\mathbf{m}}_k - \tilde{\mathbf{m}}_{k-1} + \tilde{\mathbf{m}}_{k-1} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T \tilde{\mathbf{m}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{m}}_k \\
&= \tilde{\mathbf{m}}_k - \tilde{\mathbf{m}}_{k-1} + \tilde{\mathbf{m}}_{k-1} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T (\tilde{\mathbf{m}}_k - \tilde{\mathbf{m}}_{k-1} + \tilde{\mathbf{m}}_{k-1}) - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{m}}_k \\
&= \nabla \tilde{\mathbf{m}}_k + \bar{\mathbf{m}}_{k-1} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T \nabla \tilde{\mathbf{m}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{m}}_k \\
&= \bar{\mathbf{m}}_{k-1} + (\mathbf{I} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T) \nabla \tilde{\mathbf{m}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{m}}_k
\end{aligned} \tag{5.3}$$

where

$$\nabla \tilde{\mathbf{m}}_k = \tilde{\mathbf{m}}_k - \tilde{\mathbf{m}}_{k-1} = \mathbf{m}_k - \mathbf{m}_{k-1} \tag{5.4}$$

is the innovations information obtained from  $\bar{\mathbf{m}}_k$  but not in  $\bar{\mathbf{m}}_{k-1}$  and with

$$\bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T = \sum_{i=1}^{k-1} (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T) = \bar{\mathbf{U}}_{k-2} + \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tag{5.5}$$

updated by  $\bar{\mathbf{U}}_{k-2}$  and  $\bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T$ .

### *KF-OSP-GSGA*

#### 1. Initial Conditions:

Find two data sample vectors with the maximal segment with two endpoints specified by  $\mathbf{m}_0$  and  $\mathbf{m}_1$ , denoted by  $\mathbf{m}_0^{\text{GSGA}}$  and  $\mathbf{m}_1^{\text{GSGA}}$  respectively. Set  $k=1$  and define  $\tilde{\mathbf{m}}_1^{\text{GSGA}} = \mathbf{m}_1^{\text{GSGA}} - \mathbf{m}_0^{\text{GSGA}}$ .

2. For each  $1 < k \leq p-1$  use (4.3) and (4.13) to find  $\tilde{\mathbf{m}}_k = \mathbf{m}_k - \mathbf{m}_0$ . Update

$$\nabla \bar{\mathbf{m}}_k^{\text{GSGA}} = \bar{\mathbf{m}}_k^{\text{GSGA}} - \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}$$

and

$$\bar{\mathbf{U}}_{k-1}^{\text{GSGA}} (\bar{\mathbf{U}}_{k-1}^{\text{GSGA}})^T = \bar{\mathbf{U}}_{k-2}^{\text{GSGA}} (\bar{\mathbf{U}}_{k-2}^{\text{GSGA}})^T + \bar{\mathbf{u}}_{k-1}^{\text{GSGA}} (\bar{\mathbf{u}}_{k-1}^{\text{GSGA}})^T.$$

3. Now  $\bar{\mathbf{m}}_k^{\text{GSGA}}$  and  $\bar{\mathbf{u}}_k^{\text{GSGA}}$  can be calculated by (4.5) and (5.3) with  $h_k^{\text{GSGA}} = \|\bar{\mathbf{m}}_k^{\text{GSGA}}\|$ .

4. Use (4.2) to find the simplex volume that

$$V\left(S_k\left(\bar{\mathbf{m}}_1^{\text{GSGA}}, \dots, \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}, \bar{\mathbf{m}}_k^{\text{GSGA}}\right)\right) = \frac{h_k^{\text{GSGA}}}{k} \cdot V\left(S_{k-1}\left(\bar{\mathbf{m}}_1^{\text{GSGA}}, \dots, \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}\right)\right).$$

5. Check if  $k = p-1$ , then the algorithm is terminated. Otherwise, let  $k \leftarrow k+1$  and go to step 2.

To perform (5.3) recursively there are three pieces of information:

1. Processed information:

- $\bar{\mathbf{U}}_{k-1} = [\bar{\mathbf{u}}_1 \bar{\mathbf{u}}_2 \dots \bar{\mathbf{u}}_{k-1}]$  and  $\bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T = \sum_{i=1}^{k-1} (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T)$
- $\bar{\mathbf{m}}_k = (\mathbf{I} - \bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T) \tilde{\mathbf{m}}_k$
- $\bar{\mathbf{u}}_k = \frac{\bar{\mathbf{m}}_k}{\|\bar{\mathbf{m}}_k\|}$ ,
- $(\mathbf{I} - \bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T)$

2. New information:  $\tilde{\mathbf{m}}_{k+1} = \mathbf{m}_{k+1} - \mathbf{m}_0$ .

3. Innovations information:  $\nabla \tilde{\mathbf{m}}_{k+1} = \tilde{\mathbf{m}}_{k+1} - \tilde{\mathbf{m}}_k$ .

### 5.3.3 KF-OVP-GSGA

Alternatively, (5.3) can be re-derived as

$$\begin{aligned} \bar{\mathbf{r}}_k &= \tilde{\mathbf{r}} - \sum_{i=1}^{k-1} (\bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^T) \tilde{\mathbf{r}} = \tilde{\mathbf{r}} - \bar{\mathbf{U}}_{k-1} \bar{\mathbf{U}}_{k-1}^T \tilde{\mathbf{r}} \\ &= \tilde{\mathbf{r}} - \tilde{\mathbf{m}}_{k-1} + \tilde{\mathbf{m}}_{k-1} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T \tilde{\mathbf{r}} - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}} \\ &= \nabla \tilde{\mathbf{r}}_k + \tilde{\mathbf{m}}_{k-1} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T \nabla \tilde{\mathbf{r}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}} \\ &= \tilde{\mathbf{m}}_{k-1} + (\mathbf{I} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T) \nabla \tilde{\mathbf{r}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}} \end{aligned} \tag{5.6}$$

where  $\tilde{\mathbf{r}} = \mathbf{r} - \mathbf{m}_0$  and  $\nabla \tilde{\mathbf{r}}_k = \tilde{\mathbf{r}} - \tilde{\mathbf{m}}_{k-1} = \mathbf{r} - \mathbf{m}_{k-1}$  for all data sample.

By virtue of (5.6) equations (4.6) becomes

$$\bar{\mathbf{m}}_k^{\text{GSGA}} = \arg \left\{ \max_{\bar{\mathbf{r}}_k} \|\bar{\mathbf{r}}_k\| \right\} \quad (5.7)$$

Using (5.7) the RGSGA can be re-derived as a Kalman filter–based orthogonal vector projection GSGA (KF OVP-GSGA) as follows.

### *KF OVP-GSGA*

#### 1. Initial Conditions:

Find two data sample vectors with the maximal segment with two endpoints specified by  $\mathbf{m}_0$  and  $\mathbf{m}_1$ , denoted by  $\mathbf{m}_0^{\text{GSGA}}$  and  $\mathbf{m}_1^{\text{GSGA}}$  respectively. Set  $k = 2$

and define  $\tilde{\mathbf{m}}_1^{\text{GSGA}} = \mathbf{m}_1^{\text{GSGA}} - \mathbf{m}_0^{\text{GSGA}}$ . Then  $\bar{\mathbf{u}}_1 = \frac{\bar{\mathbf{m}}_1^{\text{GSGA}}}{\|\mathbf{m}_1^{\text{GSGA}}\|}$ .

#### 2. At $k^{\text{th}}$ recursion:

a. For each  $1 < k \leq p-1$  find  $\tilde{\mathbf{r}} = \mathbf{r} - \mathbf{m}_0$  and use (5.6) to calculate

$$\bar{\mathbf{r}}_k = \tilde{\mathbf{m}}_{k-1} + (\mathbf{I} - \bar{\mathbf{U}}_{k-2} \bar{\mathbf{U}}_{k-2}^T) \nabla \tilde{\mathbf{r}}_k - \bar{\mathbf{u}}_{k-1} \bar{\mathbf{u}}_{k-1}^T \tilde{\mathbf{r}}.$$

Then  $\bar{\mathbf{m}}_k^{\text{GSGA}}$  and  $h_k^{\text{GSGA}}$  could be found by (5.7) and (4.8) respectively.

b. Use (4.2) to find the simplex volume

$$V\left(S_k\left(\bar{\mathbf{m}}_1^{\text{GSGA}}, \dots, \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}, \bar{\mathbf{m}}_k^{\text{GSGA}}\right)\right) = \frac{h_k^{\text{GSGA}}}{k} \cdot V\left(S_{k-1}\left(\bar{\mathbf{m}}_1^{\text{GSGA}}, \dots, \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}\right)\right).$$

c. Update  $\nabla \bar{\mathbf{r}}_k^{\text{GSGA}} = \bar{\mathbf{r}} - \bar{\mathbf{m}}_{k-1}^{\text{GSGA}}$  and

$$\bar{\mathbf{U}}_{k-1}^{\text{GSGA}} \left(\bar{\mathbf{U}}_{k-1}^{\text{GSGA}}\right)^T = \bar{\mathbf{U}}_{k-2}^{\text{GSGA}} \left(\bar{\mathbf{U}}_{k-2}^{\text{GSGA}}\right)^T + \bar{\mathbf{u}}_{k-1}^{\text{GSGA}} \left(\bar{\mathbf{u}}_{k-1}^{\text{GSGA}}\right)^T.$$

3. Check if  $k = p-1$ . If yes, the algorithm is terminated. Otherwise, let  $k \leftarrow k+1$  and go to step 2.

In order to perform (5.6) recursively, three pieces of information are required.

1. Processed information:

- $\bar{\mathbf{m}}_{k-1}^{\text{GSGA}}$ .
- $\bar{\mathbf{u}}_{k-1}^{\text{GSGA}} = \frac{\bar{\mathbf{m}}_{k-1}^{\text{GSGA}}}{\|\bar{\mathbf{m}}_{k-1}^{\text{GSGA}}\|}$ .
- $\bar{\mathbf{U}}_{k-2} = [\bar{\mathbf{u}}_1 \bar{\mathbf{u}}_2 \cdots \bar{\mathbf{u}}_{k-2}]$ .
- $\mathbf{I} - \bar{\mathbf{U}}_{k-2} (\bar{\mathbf{U}}_{k-2})^T$ .

2. New information:

- $\mathbf{m}_k$
- $\tilde{\mathbf{r}}$

3. Innovations information:

- $\nabla \tilde{\mathbf{m}}_k$ .
- $\nabla \tilde{\mathbf{r}}_k = \tilde{\mathbf{r}} - \tilde{\mathbf{m}}_{k-1}$ .

5.4 Recursive equations for OP-SGA

As shown in (4.16) the key element in calculating GSV is the product of maximal OPs,

$$\prod_{i=0}^{k-1} \|\tilde{\mathbf{m}}_{k-i}^{\text{OP}}\|, \quad (5.8)$$

which is produced by endmembers where  $\mathbf{m}_k$  solves (4.12-4.13). This section develops a recursive equation to solve (4.14). According to (4.14)

$\tilde{\mathbf{m}}_k = \arg \max_{\tilde{\mathbf{r}}} \left\{ \left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right)^T \left( P_{\tilde{\mathbf{U}}_{k-1}}^\perp \tilde{\mathbf{r}} \right) \right\}$ , which corresponds to finding the maximal OP in the

space  $\langle \tilde{\mathbf{U}}_{k-1} \rangle^\perp$ . In this case, an recursive equation is needed to renew  $P_{\tilde{\mathbf{U}}_j}^\perp$  via  $P_{\tilde{\mathbf{U}}_{j-1}}^\perp$ .

Let  $\tilde{\mathbf{U}}_{k-1} = [(\mathbf{m}_1 - \mathbf{m}_0)(\mathbf{m}_2 - \mathbf{m}_0) \cdots (\mathbf{m}_{k-1} - \mathbf{m}_0)] = [\tilde{\mathbf{m}}_1 \tilde{\mathbf{m}}_2 \cdots \tilde{\mathbf{m}}_{k-1}]$  and

$\tilde{\mathbf{U}}_k = [\tilde{\mathbf{m}}_1 \tilde{\mathbf{m}}_2 \cdots \tilde{\mathbf{m}}_{k-1} \tilde{\mathbf{m}}_k] = [\tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{m}}_k]$ . Then we can obtain the following recursive equations for implementing OP-SGA.

$$[\tilde{\mathbf{U}}_k^T \tilde{\mathbf{U}}_k]^{-1} = \begin{bmatrix} \tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1} & \tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{m}}_k \\ \tilde{\mathbf{m}}_k^T \tilde{\mathbf{U}}_{k-1} & \tilde{\mathbf{m}}_k^T \tilde{\mathbf{m}}_k \end{bmatrix}^{-1} = \begin{bmatrix} (\tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1})^{-1} + \beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T & -\beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \\ -\beta \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T & \beta \end{bmatrix} \quad (5.9)$$

where  $\tilde{\mathbf{U}}_{k-1}^\# = (\tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1})^{-1} \tilde{\mathbf{U}}_{k-1}^T$  and

$$\beta = \left\{ \tilde{\mathbf{m}}_k^T \left[ \mathbf{I} - \tilde{\mathbf{U}}_{k-1} (\tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1})^{-1} \tilde{\mathbf{U}}_{k-1}^T \right] \tilde{\mathbf{m}}_k \right\}^{-1} = \left\{ \tilde{\mathbf{m}}_k^T \left[ P_{\tilde{\mathbf{U}}_{k-1}}^\perp \right] \tilde{\mathbf{m}}_k \right\}^{-1}. \quad (5.10)$$

$$\begin{aligned} \tilde{\mathbf{U}}_k^\# &= (\tilde{\mathbf{U}}_k^T \tilde{\mathbf{U}}_k)^{-1} \tilde{\mathbf{U}}_k^T = \begin{bmatrix} (\tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1})^{-1} + \beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T & -\beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \\ -\beta \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T & \beta \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{U}}_{k-1}^T \\ \tilde{\mathbf{m}}_k^T \end{bmatrix} \\ &= \begin{bmatrix} \tilde{\mathbf{U}}_{k-1}^\# + \beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T \tilde{\mathbf{U}}_{k-1}^T - \beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T \\ -\beta \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T \tilde{\mathbf{U}}_{k-1}^T + \beta \tilde{\mathbf{m}}_k^T \end{bmatrix} \end{aligned} \quad (5.11)$$

$$\begin{aligned} \tilde{\mathbf{U}}_k \tilde{\mathbf{U}}_k^\# &= [\tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{m}}_k] (\tilde{\mathbf{U}}_k^T \tilde{\mathbf{U}}_k)^{-1} \tilde{\mathbf{U}}_k^T \\ &= [\tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{m}}_k] \begin{bmatrix} \tilde{\mathbf{U}}_{k-1}^\# + \beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T \tilde{\mathbf{U}}_{k-1}^T - \beta \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T \\ -\beta \tilde{\mathbf{m}}_k^T (\tilde{\mathbf{U}}_{k-1}^\#)^T \tilde{\mathbf{U}}_{k-1}^T + \beta \tilde{\mathbf{m}}_k^T \end{bmatrix} \\ &= \tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^\# + \beta \tilde{\mathbf{u}}_{k-1} \tilde{\mathbf{u}}_{k-1}^T - \beta \tilde{\mathbf{u}}_{k-1} \tilde{\mathbf{m}}_k^T - \beta (\tilde{\mathbf{u}}_{k-1} \tilde{\mathbf{m}}_k^T)^T + \beta \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T \end{aligned} \quad (5.12)$$

where  $\tilde{\mathbf{u}}_{k-1} = \tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k$ .

$$\begin{aligned} P_{\tilde{\mathbf{U}}_k}^\perp &= \mathbf{I} - \tilde{\mathbf{U}}_k \tilde{\mathbf{U}}_k^\# = \mathbf{I} - \tilde{\mathbf{U}}_k (\tilde{\mathbf{U}}_k^T \tilde{\mathbf{U}}_k)^{-1} \tilde{\mathbf{U}}_k^T \\ &= P_{\tilde{\mathbf{U}}_{k-1}}^\perp - \beta \tilde{\mathbf{u}}_{k-1} \tilde{\mathbf{u}}_{k-1}^T + \beta \tilde{\mathbf{u}}_{k-1} \tilde{\mathbf{m}}_k^T + \beta \tilde{\mathbf{m}}_k \tilde{\mathbf{u}}_{k-1}^T - \beta \tilde{\mathbf{m}}_k \tilde{\mathbf{m}}_k^T \\ &= P_{\tilde{\mathbf{U}}_{k-1}}^\perp - \beta (\tilde{\mathbf{u}}_{k-1} - \tilde{\mathbf{m}}_k) (\tilde{\mathbf{u}}_{k-1} - \tilde{\mathbf{m}}_k)^T \end{aligned} \quad (5.13)$$

where  $P_{\tilde{\mathbf{U}}_{k-1}}^\perp = \mathbf{I} - \tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^\# = \mathbf{I} - \tilde{\mathbf{U}}_{k-1} (\tilde{\mathbf{U}}_{k-1}^T \tilde{\mathbf{U}}_{k-1})^{-1} \tilde{\mathbf{U}}_{k-1}^T$ . It should be also noted that all  $\mathbf{m}_k^T \tilde{\mathbf{u}}_{k-1} = \tilde{\mathbf{u}}_{k-1}^T \tilde{\mathbf{m}}_k$ ,  $\tilde{\mathbf{m}}_k^T (\tilde{\mathbf{u}}_{k-1} - \tilde{\mathbf{m}}_k) = (\tilde{\mathbf{u}}_{k-1} - \tilde{\mathbf{m}}_k)^T \tilde{\mathbf{m}}_k$ ,  $\beta$  in (4.11)-(4.15) are scalars. If  $\tilde{\mathbf{U}}_{k-1}$  is invertible, then  $\tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^\# = \mathbf{I}$ . It should be noted that  $\tilde{\mathbf{u}}_{k-1} = \tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^\# \tilde{\mathbf{m}}_k$ ,  $\mathbf{m}_k^T \tilde{\mathbf{u}}_{k-1} = \tilde{\mathbf{u}}_{k-1}^T \tilde{\mathbf{m}}_k$  and  $\beta = \left\{ \tilde{\mathbf{m}}_k^T \left[ P_{\tilde{\mathbf{U}}_{k-1}}^\perp \right] \tilde{\mathbf{m}}_k \right\}^{-1}$  are used to account for correlation between  $\tilde{\mathbf{U}}_{k-1}$  and  $\tilde{\mathbf{m}}_k$ .

### 5.5 Recursive OP-SGA

According to (4.10-4.11) the  $k^{\text{th}}$  endmember  $\mathbf{m}_k$  can be found by maximizing the residuals of  $P_{\tilde{\mathbf{U}}_{k-1}}^\perp \mathbf{r}$  over all the data sample vectors in the space  $\langle \tilde{\mathbf{U}}_{k-1} \rangle^\perp$  and (5.13) is a recursive equation that can be used to compute  $P_{\tilde{\mathbf{U}}_k}^\perp$  via  $P_{\tilde{\mathbf{U}}_{k-1}}^\perp$  recursively. The detailed implementation of recursive OP-SGA (ROP-SGA) is summarized as follows.

#### *Recursive OP-SGA (ROP-SGA)*

##### 1. Initial conditions:

- a. Find two data sample vectors with the maximal segment with two endpoints specified by  $\mathbf{m}_0$  and  $\mathbf{m}_1$ , denoted by  $\mathbf{m}_0^{\text{OPSGA}}$  and  $\mathbf{m}_1^{\text{OPSGA}}$  respectively. Define

$$\tilde{\mathbf{m}}_1^{\text{OPSGA}} = \mathbf{m}_1^{\text{OPSGA}} - \mathbf{m}_0^{\text{OPSGA}}.$$

- b. Find  $P_{\tilde{\mathbf{U}}_1}^\perp = \mathbf{I} - \tilde{\mathbf{U}}_1 \tilde{\mathbf{U}}_1^\# = \mathbf{I} - \tilde{\mathbf{U}}_1 (\tilde{\mathbf{U}}_1^T \tilde{\mathbf{U}}_1)^{-1} \tilde{\mathbf{U}}_1^T$  with  $\tilde{\mathbf{U}}_1 = [\mathbf{m}_1 - \mathbf{m}_0]$  and

$$V(\mathbf{m}_0, \mathbf{m}_1) = d(\mathbf{m}_0, \mathbf{m}_1)$$

- c. Set  $k = 2$ .

2. At the  $k^{\text{th}}$  recursion:

a. Find  $\tilde{\mathbf{m}}_k = \arg \left\{ \max_{\tilde{\mathbf{r}}} \left\| P_{\tilde{\mathbf{U}}_{k-1}}^{\perp} \tilde{\mathbf{r}} \right\| \right\}$  via (4.11).

b. Calculate  $\| \tilde{\mathbf{m}}_k^{\text{OP}} \| = \left\| P_{\tilde{\mathbf{U}}_{k-1}}^{\perp} \tilde{\mathbf{m}}_k \right\|$

c. Compute volume by using  $\| \tilde{\mathbf{m}}_k^{\text{OP}} \|$  obtained in step 2(b) as

$$V(\mathbf{m}_0, \dots, \mathbf{m}_k) = \frac{1}{k} \| \tilde{\mathbf{m}}_k^{\text{OP}} \| \cdot V(\mathbf{m}_0, \dots, \mathbf{m}_{k-1}).$$

d. Use the following recursive equation to calculate  $P_{\tilde{\mathbf{U}}_j}^{\perp}$  via  $P_{\tilde{\mathbf{U}}_{j-1}}^{\perp}$ .

$$P_{\tilde{\mathbf{U}}_k}^{\perp} = P_{\tilde{\mathbf{U}}_{k-1}}^{\perp} - \beta (\tilde{\mathbf{u}}_{k-1} - \tilde{\mathbf{m}}_k) (\tilde{\mathbf{u}}_{k-1} - \tilde{\mathbf{m}}_k)^T$$

$$\text{where } \tilde{\mathbf{u}}_{k-1} = \tilde{\mathbf{U}}_{k-1} \tilde{\mathbf{U}}_{k-1}^{\#} \tilde{\mathbf{m}}_k.$$

3. If  $k < p-1$ ,  $k \leftarrow k+1$  and go to step 2. Otherwise, all  $p$  endmembers,  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_{p-1}$  with  $\mathbf{m}_k = \tilde{\mathbf{m}}_k + \mathbf{m}_0$  for  $1 \leq k < p$ , have been generated by ROP-SGA.

### 5.6 Determining Number of Endmembers to be generated

One issue encountered in implementing most of EFAs, such as NFINDR, VCA, and SGA, is that how to determine the value of  $p$ , i.e. the number of endmembers that present in the data. When its value is unknown, it must be determined in an unsupervised fashion. As it was mentioned in previous chapters, VD was used to address this issue. In this section, ROP-SGA offers a rather different approach without appealing for VD. While it generates endmembers, it also makes use of its found endmembers to determine the number of endmembers needed to generate.

The idea is originated from  $\beta = \left\{ \tilde{\mathbf{m}}_j^T \left[ P_{\tilde{\mathbf{U}}_{j-1}}^\perp \right] \tilde{\mathbf{m}}_j \right\}^{-1}$  in (5.10) which accounts for the accuracy in prediction of  $P_{\tilde{\mathbf{U}}_j}^\perp$  via  $P_{\tilde{\mathbf{U}}_{j-1}}^\perp$  where the smaller  $\beta$  is, the better prediction of  $P_{\tilde{\mathbf{U}}_j}^\perp$  by  $P_{\tilde{\mathbf{U}}_{j-1}}^\perp$  is. In this case minimizing  $\beta$  is equivalent to maximizing  $\tilde{\mathbf{m}}_j^T \left[ P_{\tilde{\mathbf{U}}_{j-1}}^\perp \right] \tilde{\mathbf{m}}_j$ . Suppose that  $p$  is the number of endmembers needed to be determined. Then the quantity

$$\eta_p = \tilde{\mathbf{m}}_p^T P_{\tilde{\mathbf{U}}_{p-1}}^\perp \tilde{\mathbf{m}}_p \quad (5.14)$$

can be used to determine how many endmembers for RGSGA and ROP-SGA should be generated. In other words, both the  $p^{\text{th}}$  endmember,  $\tilde{\mathbf{m}}_p$  found by (5.14). More interestingly, the  $\eta_p$ , turned to be exactly the same signal sources under the binary hypothesis testing problem considered in the maximal orthogonal subspace projection (MOSP) in (Chang et al., 2011) to determine VD or maximal orthogonal complement algorithm (MOCA) in (Kuybeda et al., 2007) to determine the rank of rare signal dimensionality as shown as follows.

Following the same approach as in MOSP and MOCA, we can formulate a binary hypothesis testing problem using  $\eta_p$  as signal sources as follows.

$$\begin{aligned} H_0 : \eta_p &\approx p(\eta_p | H_0) = p_0(\eta_p) \\ \text{versus} & & \text{for } p = 1, 2, \dots, L \quad (5.15) \\ H_1 : \eta_p &\approx p(\eta_p | H_1) = p_1(\eta_p) \end{aligned}$$

where the alternative hypothesis  $H_1$  and the null hypothesis  $H_0$  represent two cases of  $\tilde{\mathbf{m}}_p$  being an endmember signal source under  $H_1$  and not an endmember signal source under  $H_0$  respectively in the sense that  $H_0$  represents the maximum residual resulting from the background signal sources, while  $H_1$  represents the maximum residual

leaked from the endmember signal sources. By virtue of extreme value theory (Leadbetter, 1987),  $\eta_p$  can be modeled as a Gumbel distribution, i.e.,  $F_{v_p}(\eta_p)$  is the cumulative distribution function (cdf) of  $v_p$  given by

$$F_{v_p}(x) \approx \exp \left\{ -e^{-\left[ \frac{x - \sigma^2(L-p)}{\sigma^2 \sqrt{2(L-p)}} - (2 \log N)^{1/2} + \frac{1}{2} (2 \log N)^{-1/2} (\log \log N + \log 4\pi) \right]} \right\} \quad (5.16)$$

Since there is no prior knowledge available about distribution of signal sources, assuming  $\eta_p$  under  $H_1$  uniformly distributed seems most reasonable according to the maximum entropy principle in Shannon's information theory.

Under these two assumptions, we can use the same approaches as MOSP and MOCA to obtain

$$p(H_0, \eta_p) = p_{v_p}(\eta_p) F_{\xi_p}(\eta_p) = p_{v_p}(\eta_p) (\eta_p / \eta_{p-1}) \quad (5.17)$$

$$p(H_1, \eta_p) = F_{v_p}(\eta_p) p_{\xi_i}(\eta_p) = F_{v_p}(\eta_p) (1 / \eta_{p-1}) \quad (5.18)$$

where

$$p_0(\eta_p) = p(H_0 | \eta_p) = \frac{\eta_p p_{v_p}(\eta_p)}{\eta_p p_{v_p}(\eta_p) + F_{v_p}(\eta_p)} \quad (5.19)$$

and

$$p_1(\eta_p) = p(H_1 | \eta_p) = \frac{F_{v_p}(\eta_p)}{\eta_p p_{v_p}(\eta_p) + F_{v_p}(\eta_p)}. \quad (5.20)$$

By virtue of (5.19) and (5.20) a Neyman-Pearson detector (NPD), denoted by  $\delta^{\text{NP}}(\eta_p)$  for the binary composite hypothesis testing problem specified by (5.14) can be obtained by maximizing the detection power  $P_D$  while the false alarm probability  $P_F$  being fixed at a specific given value,  $\alpha$ , which determines the threshold value  $\tau_p$  in the following randomized decision rule

$$\delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) = \begin{cases} 1; & \text{if } \Lambda(\eta_p) > \tau_p \\ 1 \text{ with probability } \kappa; & \text{if } \Lambda(\eta_p) = \tau_p \\ 0; & \text{if } \Lambda(\eta_p) < \tau_p \end{cases} \quad (5.21)$$

where the likelihood ratio test  $\Lambda(\eta_p)$  is given by  $\Lambda(\eta_p) = p_1(\eta_p) / p_0(\eta_p)$  with  $p_0(\eta_p)$  and  $p_1(\eta_p)$  given by (5.19) and (5.20). So, according to (5.14) a case of  $\eta_p > \tau_p$  indicates that  $\delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p)$  in (5.21) fails the test, in which case  $\tilde{\mathbf{m}}_p = \tilde{\mathbf{m}}_p^{\text{ROP-SGA}}$  is assumed to be an endmember. It should be noted that the test for (5.21) must be performed for each of  $L$  potential endmember candidates. Therefore, for  $l$  the threshold  $\eta$  varies. Using (5.21) the VD can be determined by calculating

$$\text{VD}_{\text{ROP-SGA}}^{\text{NP}}(\text{P}_F) = \sum_{p=2}^L \left[ \delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) \right] \quad (5.22)$$

where  $\text{P}_F$  is a pre-determined false alarm probability,  $\left[ \delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) \right] = 1$  only if  $\delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) = 1$  and  $\left[ \delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) \right] = 0$  if  $\delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) < 1$ . It should be noted that since  $p$  starts with 2, the actual value of VD estimated for ROP-SGA in (5.22) should be  $\text{VD}_{\text{ROP-SGA}}^{\text{NP}}(\text{P}_F) + 1$ .

Interestingly, it has been shown in (Chang et al., 2015) that the residual energy of the signal source  $\tilde{\mathbf{m}}_p$  in (5.14) into  $P_{\tilde{\mathbf{U}}_{p-1}}^\perp$  can be further modified by the residual strength of the signal source  $\tilde{\mathbf{m}}_p$  in (5.14) into  $P_{\tilde{\mathbf{U}}_{p-1}}^\perp$  and given by

$$\rho_p = \sqrt{\eta_p} = \left( \tilde{\mathbf{m}}_p^T P_{\tilde{\mathbf{U}}_{p-1}}^\perp \tilde{\mathbf{m}}_p \right)^{1/2}. \quad (5.23)$$

The detector (5.21) is then performed on the signal residual strength  $\eta_p$  in (5.23) under the binary hypothesis testing problem (5.20). In this case, the VD in (5.22) becomes

$$VD_{\text{ROP-SGA}}^{\text{NP}}(P_F) = \sum_{p=2}^L \left[ \delta_{\text{ROP-SGA}}^{\text{NP}}(\eta_p) \right]. \quad (5.24)$$

## 5.7 Real Image Experiments

### 5.7.1 VD estimation using real targets

Despite that ROP-SGA can be used to find endmember candidates up to the number of total bands plus one,  $L + 1$ , there is no necessity for doing so. Using (5.14) and (5.23), the number of endmembers required for ROP-SGA to generate can be estimated while the process of ROP-SGA is ongoing. Table 5.1 tabulates the values of VD for HYDICE estimated by MOCA as well as NPD for various false alarm probabilities where MOCA is actually a maximum-likelihood detector (Chang et al., 2014).

**Table 5.1. VD estimated by (5.14) and (5.23) on HYDICE**

$\eta_p$ in (5.14)	MOCA	$P_F=10^{-1}$	$P_F=10^{-2}$	$P_F=10^{-3}$	$P_F=10^{-4}$	$P_F=10^{-5}$
DSGA	98	98	98	98	98	98
ROP-SGA	45	46	44	43	40	39

$\eta_p$ in (5.23)	MOCA	$P_F=10^{-1}$	$P_F=10^{-2}$	$P_F=10^{-3}$	$P_F=10^{-4}$	$P_F=10^{-5}$
DSGA	98	98	98	98	98	98
ROP-SGA	43	45	43	40	39	37

For Cuprite image data, (5.14) and (5.23) were also used to estimate VD for both reflectance and radiance data and the results are tabulated in Tables 5.2 and 5.3 where an “F” indicates that the NPD test passed all the signal sources, in which case  $n_{\text{VD}} = 169$  for HYDICE,  $n_{\text{VD}} = 120$  for Cuprite reflectance data and  $n_{\text{VD}} = 139$  for Cuprite radiance data. It should be noted that for Cuprite data DSGA could not produce 189 target signal sources due to its use of SVD to calculate SV. From Tables 5.2 and 5.3, it can be seen that VD values estimated by (5.14) and (5.23) with endmembers found

by ROP-SGA will not over 75 while false alarm probability was set to be less than  $10^{-2}$ . Therefore,  $n_{VD} = 75$  was set to be the total number of endmembers for variants of SGA to generate in Chapter 4.

**Table 5.2.** VD estimated for Cuprite reflectance data by (5.14) and (5.23)

$\eta_p$ in (5.14)	MOCA	$P_F=10^{-1}$	$P_F=10^{-2}$	$P_F=10^{-3}$	$P_F=10^{-4}$	$P_F=10^{-5}$
DSGA	F	F	F	119	119	117
ROP-SGA	75	82	74	69	64	62

$\eta_p$ in (5.23)	MOCA	$P_F=10^{-1}$	$P_F=10^{-2}$	$P_F=10^{-3}$	$P_F=10^{-4}$	$P_F=10^{-5}$
DSGA	119	119	119	117	103	95
ROP-SGA	75	82	74	69	64	62

**Table 5.3.** VD estimated or Cuprite radiance data by (5.14) and (5.23)

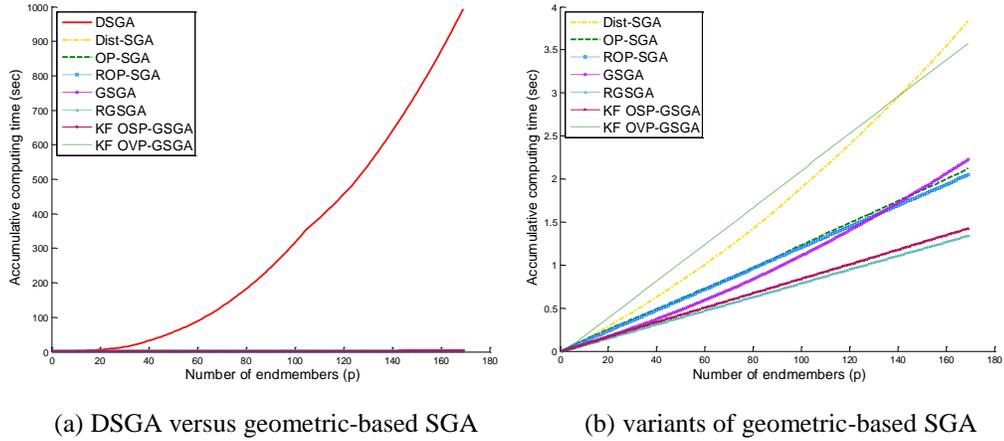
$\eta_p$ in (5.14)	MOCA	$P_F=10^{-1}$	$P_F=10^{-2}$	$P_F=10^{-3}$	$P_F=10^{-4}$	$P_F=10^{-5}$
DSGA	F	F	109	106	104	95
ROP-SGA	72	79	72	69	65	62

$\eta_p$ in (5.23)	MOCA	$P_F=10^{-1}$	$P_F=10^{-2}$	$P_F=10^{-3}$	$P_F=10^{-4}$	$P_F=10^{-5}$
DSGA	103	106	103	95	95	85
ROP-SGA	67	71	67	62	58	56

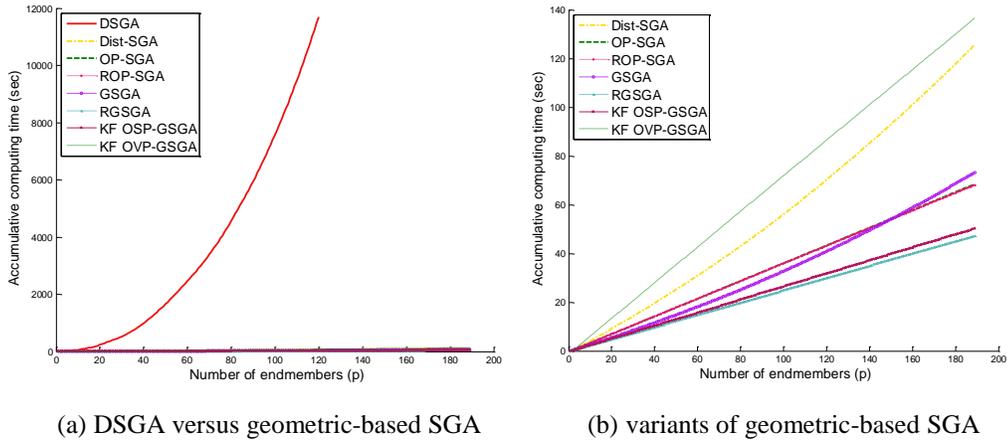
### 5.7.2 Computer processing time comparison

Despite that Dist-SGA, GSGA, RGSGA, OP-SGA, ROP-SGA, KF OSP-SGA and KF OVP-SGA produced identical sets of target pixels, their required computer processing times were quite different. In order to conduct a fair comparative analysis on computing time, all the algorithms, Dist-SGA, GSGA, RGSGA, OP-SGA, ROP-SGA, KF OSP-SGA and KF OVP-SGA, were run to find simplexes from 1-simplex up to  $L$ -simplex,  $L = 169$  and  $L = 189$  for HYDICE and Cuprite, respectively, and further calculated their corresponding processing times. Figs. 5.1 and 5.2 plot their computer processing time for HYDICE scene and Cuprite data where the RGSGA

was the best and DSGA was the worst. It is worthy noted that the computing time for DSGA on Cuprite data can only be recoded up to  $L = 139$  since only 139 endmembers can be found in Cuprite reflectance data with DSGA using SVD to calculate SV.



**Figure 5.1.** Accumulative computing time in seconds of DSGA without DR, Dist-SGA, OP-SGA, ROP-SGA, GSGA, RGSGA, KF OSP-GSGA, and KF OVP-GSGA as  $p$  increases on HYDICE data



**Figure 5.2.** Accumulative computing time in seconds of DSGA without DR, Dist-SGA, OP-SGA, ROP-SGA, GSGA, RGSGA, KF OSP-GSGA, and KF OVP-GSGA as  $p$  increases on Cuprite data

Tables 5.4 and 5.5 also tabulate computer processing time required for DSGA, Dist-SGA, OP-SGA, ROP-SGA, GSGA, RGSGA, KF OSP-GSGA, and KF-OVP for both HYDICE scene and Cuprite data. As we can see from Tables 5.4 and 5.5, among all algorithms RGSGA was the best requiring least computing time, while DSGA was the worst requiring much more time. Additionally, while more endmembers are needed to be found more time was saved via RGSGA. It is worth noting that Dist-SGA was shown to be the best in (Wang et al., 2013) among all variants of SGA. Figs. 5.1-5.2 and Tables 5.4-5.5 further show that our developed GSGA, RGSGA, KF OSP-GSGA, OP-SGA and ROP-SGA outperformed Dist-SGA. And KF OVP-GSGA had less computing time while the number of endmember is less than 140 for HYDICE.

**Table 5.4.** Comparison of computing time in seconds for endmembers found by variants of SGA on HYDICE

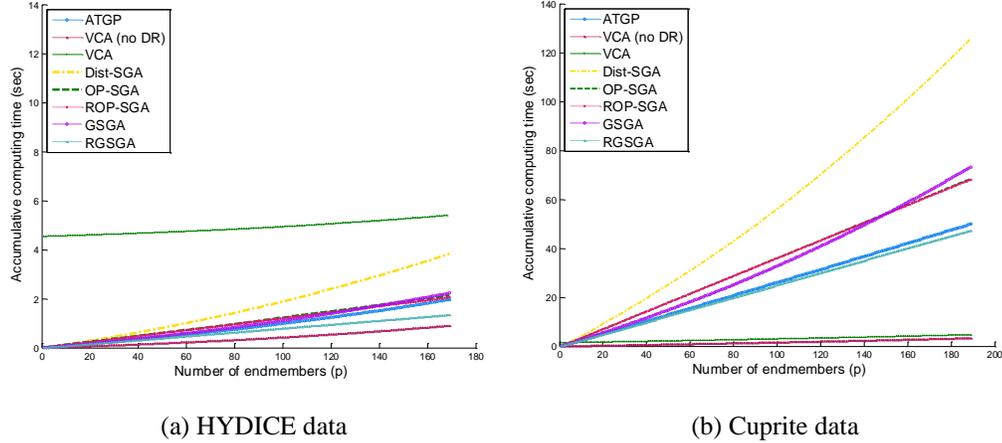
methods $p$	DSGA	Dist-SGA	OP-SGA	ROP-SGA	GSGA	RGSGA	KF OSP-GSGA	KF OVP-GSGA
9	1.7086	0.1134	0.1136	0.1026	0.0697	0.0640	0.0673	0.1519
18	5.2592	0.2542	0.2292	0.2114	0.1521	0.1359	0.1457	0.3433
34	20.1801	0.5202	0.4177	0.4014	0.3087	0.2639	0.2842	0.6843
45	43.7703	0.7192	0.5482	0.5344	0.4236	0.3532	0.3763	0.9230

**Table 5.5.** Comparison of computing time in seconds for endmembers found by variants of SGA on Cuprite data

methods $p$	DSGA	Dist-SGA	OP-SGA	ROP-SGA	GSGA	RGSGA	KF OSP-GSGA	KF OVP-GSGA
22	291.91	10.15	7.68	7.65	6.06	5.34	5.54	14.83
46	1372.89	22.97	16.40	16.35	13.51	11.36	11.90	32.43
75	3945.55	39.89	26.93	26.87	23.34	18.66	19.64	53.63

As a final remark, some comments are noteworthy. It is known that VCA is also an OP-based algorithm and has been widely used and become popular for finding endmembers in recent years. However, it suffers from several implementation issues. One is its use of random initial conditions which result in finding inconsistent sets of endmembers. Another is that it generally requires pre-processing prior to finding endmembers such as DR transform to reduce data volumes. As shown in (Chang, 2013), different DR transforms generally produce different sets of endmembers. Most importantly, VCA is not a fully constrained algorithm which generally finds sub-optimal solutions. As shown in (Du et al., 2008; Chang et al., 2013), VCA cannot compete against ATGP in the sense of maximal OP. It is also shown in (Du, 2012) that SGA outperforms VCA in the sense of maximal SV. Compared to VCA, Dist-SGA proposed by Wang et al. and the proposed algorithms using geometric volume to calculate SV, such as GSGA and OP-SGA along with their recursive versions, have no such issues. The only major advantage that VCA can offer is its simple computational complexity which requires only inner products. But this advantage can be also gained by GSGA and its recursive versions which also require only inner products.

Fig. 5.3 plots computing times required to run the HYDICE scene and Cuprite data for six algorithms, ATGP which is an unconstrained OP-based algorithm, VCA with/without dimensionality reduction and the fully abundance sum-to-one constraint (ASC)-imposed OP-based endmember finding algorithms, Dist-SGA, GSGA, OP-SGA along with their recursive versions, RGSGA, ROP-SGA, respectively.



**Figure 5.3.** Computing times for ATGP, VCA with/without DR, Dist-SGA, OP-SGA, ROP-SGA, GSGA, and RGSGA

As shown in Fig. 5.3, the computing time of VCA varies with size of data and is not stable compared to other algorithms because VCA with no DR applied required the least computing time for both HYDICE scene and Cuprite data whereas VCA with DR applied produced different results with regard to computing time performance. VCA took the longest computing time for HYDICE, but it was the second best algorithm in computing time for Cuprite data. Overall, Dist-SGA was the worst algorithm with worst computing time performance among all the eight algorithms except VCA, which is with DR applied prior to processing, for HYDICE, while ATGP and RGSGA are ranked as 2<sup>nd</sup> and 3<sup>rd</sup> best in computing time, respectively, next to VCA with no DR applied. However, since VCA is not designed to find maximal SV, it did not produce good endmember results unless VCA used ATGP-generated target pixels as initial conditions regardless of its savings in computing time which was pointed out in (Chang, 2013; Chen, 2014). It is also interesting to see that VCA with DR applied took the longest time to find

endmembers for HYDICE as shown in Fig. 5.3(a) but the 2<sup>nd</sup> best in time performance for Cuprite data as it is in Fig. 5.3(b) which is resulting from the performing of DR prior to projection step in VCA. When the size of data is very large such as Cuprite having 350×350 pixels, VCA with DR applied actually takes advantage of DR-reduced data to save computing time where DR is only one-time operation in VCA. As a result, VCA with/without DR applied achieved the two best methods in time performance.

Another comment is noteworthy. Since VCA makes use of random initial conditions for each endmember it generates, it is interesting to see how random initial conditions have impact on its final endmember results. In (Chang, 2013; Chen, 2014) three initial conditions were investigated for VCA with no DR applied prior to projection step, (1) initial conditions generated by Gaussian random generators originally used by VCA, (2) the unity vector with all components specified by 1's, i.e.,  $(\underbrace{1, 1, \dots, 1}_L)^T$ , and (3) ATGP-generated target pixels. It has been shown that the best performance VCA could produce was the one which used the ATGP-generated target pixels as initial conditions. Interestingly, in this case, VCA is simply reduced to ATGP. Moreover, it is also shown that if VCA is implemented with DR or using random initial conditions, VCA generally did not perform as well as ATGP did.

Finally, it is faithfully to point out that as long as maximal OP and maximal SV are used as criteria for finding endmembers VCA may not be effective even though VCA with/without DR applied may take the least computing time as shown in Fig. 5.3. Unfortunately, this must be traded for its suboptimal performance in finding endmembers as a compromise, for which we consider it not worthwhile. With all

things considered RGSGA and ROP-SGA is generally preferred among all currently available SV-based endmember finding algorithms in terms of performances in finding endmember and computing time.

## 5.8 Conclusions

This chapter develops recursive versions of geometry simplex growing algorithm (GSGA) and orthogonal projection-based SGA (OP-SGA). In the previous chapters, SV calculation via geometric method is proved to be a reliable approach to finding endmembers and also significantly reducing computational cost. The proposed versions of GSGA and OP-SGA make it easy to update the key operators in a recursive manner. In particular, ROP-SGA is derived to allow OP-SGA to update orthogonal subspace  $P_{U_k}^\perp$  with  $P_{U_{k-1}}^\perp$  and new endmember  $\mathbf{m}_k$  without re-processing previous endmembers  $\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_k$ , whereas RGSGA is derived by updating orthogonalized vectors of all data sample vectors via  $\bar{\mathbf{m}}_k$  corresponded to new endmember  $\mathbf{m}_k$ . The recursive versions of GSGA and OP-SGA have two significant benefits. One is tremendous reduction of computational cost. They replace matrix inverse calculations with calculating inner and outer products of vectors. Another is that the recursive equations developed to implement RGSGA and ROP-SGA are Kalman filter-like filters which can be easily implemented in real time and hardware design. Besides, ROP-SGA is able to provide an effective means of determining how many endmembers needed to be generated in an unsupervised fashion compared to SGA which must know this number in advance prior to its implementation.

# Chapter 6: GEOMETRIC CONVEX CONE VOLUME ANALYSIS

## 6.1 Introduction

The previous chapters introduce two geometric approaches to implementing SGA, which are GSGA and OP-SGA. In order to provide their real-time capability, these algorithms are further extended to their corresponding recursive versions, RGSGA, KF OSP-GSGA, KF OVP-GSGA, and ROP-SGA, to reduce exceedingly high computational complexity and also avoid numerical issue caused by calculating SV via matrix determinant. Recently, an alternative to SGA, called Convex Cone Volume Analysis (CCVA), was developed by Chang et al. (2016) This chapter applies a similar treatment to develop a theory for CCVA, to be called Geometric Convex Cone Volume Algorithm (GCCVA).

Convex Cone Analysis (Ifarraguerri and Chang, 1999) was aimed at looking for boundaries of a convex cone with convex region containing all data sample vectors in the data. The boundary vectors of a convex cone are specified as desired endmembers. In comparison with simplex-based approaches which finds a simplex with as many data samples lying on/inside the simplex as possible to satisfy fully-abundance constraints, where abundances must be sum-to-one and non-negative, convex-cone-based methods express each data sample vectors as a linear combination of the boundary vectors with only abundances non-negativity constraint satisfied.

Convex Cone-based Growing Algorithm (CCGA) (Xiong et al., 2010) was proposed to improve computational complexity of SGA by replacing maximal

simplex volume with maximal convex cone volume as a criterion where the vertices of a simplex are the projection points of convex cone boundary vectors  $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_k$  on the hyperplane  $\mathbf{e}^T \mathbf{u} = 1$  orthogonal to the sample mean vector  $\mathbf{u}$ . However, the issue of direct simplex volume calculation in CCGA also arises from calculating SV via matrix determinant as it is described in Chapter 3.

In this chapter, GCCVA takes advantage of the idea of geometric simplex volume calculation presented in Chapter 3 to reduce computing time of CCGA and improve the performance.

## 6.2 Convex Cone Volume Analysis (CCVA)

The idea of Convex Cone Analysis (CCA) is based on the observation that some physical quantities, such as radiance or mass spectra, are strictly nonnegative (Ifarraguerri and Chang, 1999). The data sample vectors formed by such spectra thus lie inside a convex region consisting of the nonnegative spectra. The objective of CCA is to find the boundary vectors of this region as defined by its corner vertices. With introducing a hyperplane to convex cone, a bounded convex cone can be found as shown in Fig. 6.1. This bounded convex cone is indeed a simplex formed by a set of projection vectors and origin. As a result, finding a convex cone with all the data sample vectors lying in the convex region becomes finding a bounded convex cone with maximal SV. The volume of this bounded convex cone can be used as a criterion for finding endmembers.

Assume  $k$  points  $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_k \in \mathfrak{R}^n$  in an  $n$ -dimensional space are linearly independent. A convex cone  $C_k$  can be expressed by the set of points that satisfies

$$C_k = \{\alpha_1 \mathbf{e}_1 + \alpha_2 \mathbf{e}_2 + \cdots + \alpha_k \mathbf{e}_k \mid \alpha_i \geq 0, 1 \leq i \leq k\}. \quad (6.1)$$

A hyperplane  $\mathbf{e}^T \mathbf{u} = 1$  thereafter is used to produce a bounded convex cone  $\tilde{C}_k$  with

corner vectors  $\frac{\mathbf{e}_1}{\mathbf{e}_1^T \mathbf{u}}, \frac{\mathbf{e}_2}{\mathbf{e}_2^T \mathbf{u}}, \dots, \frac{\mathbf{e}_k}{\mathbf{e}_k^T \mathbf{u}}$ , which are projections of  $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_k$  onto

hyperplane  $\mathbf{e}^T \mathbf{u} = 1$ , where the content of  $\tilde{C}_k$  is denoted as  $V(\tilde{C}_k)$ . Therefore, the

volume  $V(\tilde{C}_k)$  of a bounded convex cone can be considered as  $\frac{h}{k}$  of the volume of

simplex  $S_{k-1}$  formed by  $k$  projection vertices,  $\frac{\mathbf{e}_1}{\mathbf{e}_1^T \mathbf{u}}, \frac{\mathbf{e}_2}{\mathbf{e}_2^T \mathbf{u}}, \dots, \frac{\mathbf{e}_k}{\mathbf{e}_k^T \mathbf{u}}$  lying on hyperplane

$\mathbf{e}^T \mathbf{u} = 1$ , i.e.

$$V(\tilde{C}_k) = \frac{h}{k} V(S_{k-1}) \quad (6.2)$$

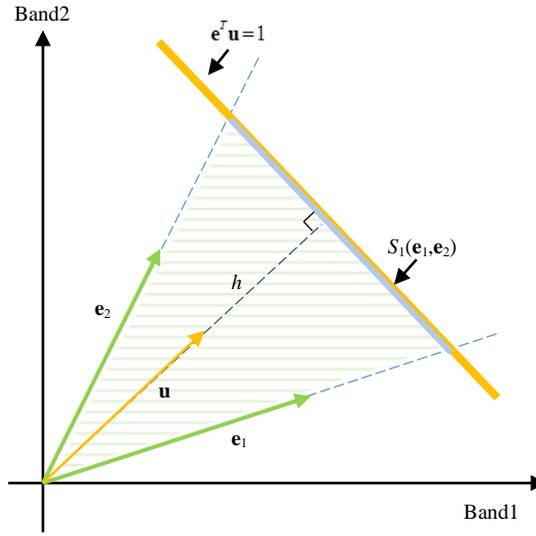
where  $h$  is the height from origin to hyperplane  $\mathbf{e}^T \mathbf{u} = 1$ .

While this bounded convex cone  $\tilde{C}_k$  lies in an  $n$ -dimensional space with  $n = k$ , the

volume  $V(\tilde{C}_k)$  can be calculated via (3.5), so that

$$V(\tilde{C}_k) = \frac{h}{k} V(S_{k-1}) = \frac{h}{k} \cdot \frac{1}{(k-1)!} |\det(\mathbf{M}_E)| = \frac{h}{k!} |\det(\mathbf{M}_E)|, \quad (6.3)$$

where  $\mathbf{M}_E = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_k \\ \mathbf{e}_1^T \mathbf{u} & \mathbf{e}_2^T \mathbf{u} & \cdots & \mathbf{e}_k^T \mathbf{u} \end{bmatrix}$ .



**Figure 6.1.** A two dimensional bounded convex cone

Now, finding boundary vectors of a convex cone  $C_k$  with maximal volume can be achieved by maximizing the volume of a bounded convex cone  $\tilde{C}_k$ . Since the maximization of  $V(\tilde{C}_k)$  is equivalent to the maximization of  $V(S_{k-1})$ , i.e.

$$\begin{aligned} \max \{V(\tilde{C}_k)\} &= \max \left\{ \frac{h}{k} \cdot V(S_{k-1}) \right\} \\ &= \frac{h}{k} \cdot \max \{V(S_{k-1})\} = \frac{h}{k!} \cdot \max |\det(\mathbf{M}_E)| \end{aligned} \quad (6.4)$$

it turns out that among all the data, convex cone volume analysis aims at finding a set of sample vectors  $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_k\}$  which yields the maximal SV of a simplex consisting of  $\{\tilde{\mathbf{e}}_1, \tilde{\mathbf{e}}_2, \dots, \tilde{\mathbf{e}}_k\}$  lying on hyperplane  $\mathbf{e}^T \mathbf{u} = 1$ , where  $\{\tilde{\mathbf{e}}_1, \tilde{\mathbf{e}}_2, \dots, \tilde{\mathbf{e}}_k\}$  is a projection set corresponding to  $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_k\}$ . CCGA was proposed based on this idea and its procedures can be described as following.

## CCGA

### 1. Pre-processing:

- a) Let  $p$  be the number of endmembers to be found.
- b) Apply a DR transform such as PCA to reduce the data dimensionality from  $L$  to  $p$  where  $L$  is the total number of spectral bands.

### 2. Initialization:

- a) Project all data samples  $\mathbf{r}_j$  onto hyperplane  $\mathbf{e}^T \mathbf{u} = 1$  as  $\tilde{\mathbf{r}}_j, j = 1, \dots, N$  where  $\mathbf{u}$  is the sample mean vector of all the data and  $N$  is total number of data samples in the image.
- b) Let  $\mathbf{e}_0$  be an initial vector randomly selected from the data and  $\tilde{\mathbf{e}}_0$  be the corresponding projection. Search all data samples to find  $\tilde{\mathbf{e}}_1$  corresponding to  $\mathbf{e}_1$  that yields maximum distance to  $\tilde{\mathbf{e}}_0$  by  $\tilde{\mathbf{e}}_1 = \arg \{ \max_{\tilde{\mathbf{r}}} S_1(\tilde{\mathbf{e}}_0, \tilde{\mathbf{r}}) \}$ .

3. At  $k \geq 2$ , for each sample vector  $\tilde{\mathbf{r}}$ , calculate  $V(\tilde{C}_k) \approx V(S_{k-1}) = V(S_{k-1}(\tilde{\mathbf{e}}_1, \tilde{\mathbf{e}}_2, \dots, \tilde{\mathbf{e}}_{k-1}, \tilde{\mathbf{r}}))$  defined by (6.3).

4. Find  $\tilde{\mathbf{e}}_k$  corresponding to  $\mathbf{e}_k$  that maximize (6.4) as

$$\tilde{\mathbf{e}}_k = \arg \{ \max_{\tilde{\mathbf{r}}} S_{k-1}(\tilde{\mathbf{e}}_1, \tilde{\mathbf{e}}_2, \dots, \tilde{\mathbf{e}}_{k-1}, \tilde{\mathbf{r}}) \}.$$

### 5. Stopping rule:

If  $k < p$ , then  $k \leftarrow k + 1$  and go to step 3. Otherwise, the final set of  $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_p\}$  is the desired  $p$  endmembers.

### 6.3 Geometric Convex Cone Volume Analysis (GCCVA)

As it is mentioned in Chapter 3, calculating SV via matrix determinant can only be valid when the simplex and its ambient space are in the same dimensionality. Otherwise, a pre-processing procedure, such as SVD or DR transform, is necessary to be applied prior to SV calculation. To avoid an issue of numerical instability and a cumbersome process, Geometric Convex Cone Volume Algorithm is developed in this chapter.

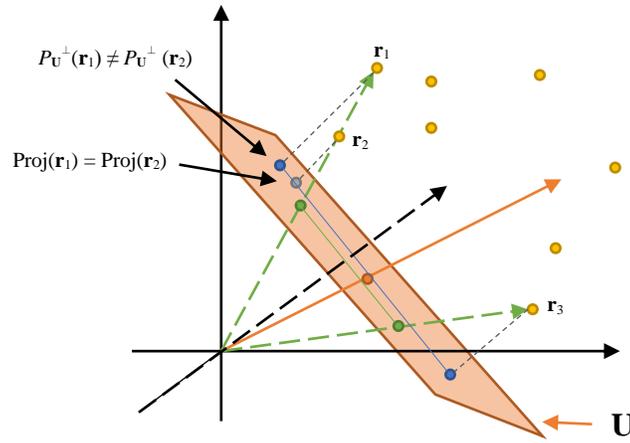
Since it is shown that the volume of a  $k$ -simplex is proportional to its corresponding parallelotope, by (3.3) and (6.2) the volume  $V(\tilde{C}_k)$  of a bounded convex cone  $\tilde{C}_k$  can be expressed as

$$V(\tilde{C}_k) = \frac{h}{k} V(S_{k-1}) = \frac{h}{k} \cdot \frac{1}{(k-1)!} h_{k-1} \cdots h_2 h_1 = \frac{h}{k!} h_{k-1} \cdots h_2 h_1, \quad (6.5)$$

where  $h_j$  is height from  $\tilde{\mathbf{e}}_j$  to  $S_{j-2}(\tilde{\mathbf{e}}_1, \tilde{\mathbf{e}}_2, \dots, \tilde{\mathbf{e}}_{j-1})$ ,  $j = 1, \dots, k$ .

Prior to calculating volume of the bounded convex cone  $\tilde{C}_k$ , all data samples must be projected onto a determined hyperplane. In CCGA, the sample mean vector  $\mathbf{u}$  is used to determine the hyperplane, and the central projection of center  $\mathbf{0}$  is used to map data samples onto the hyperplane (Gallier, 2011). It should be noted that the central projection always keep the boundary vectors of a convex cone invariant. In other words, the boundary vectors of a convex cone will still be boundaries of the bounded convex cone no matter which hyperplane was used. An OP approach is suggested in this section to perform the projection process. Fig. 6.2 shows the difference of projecting data sample vectors onto hyperplane via central projection and OP. Consequently, Geometric Convex Cone Volume Analysis (GCCVA) can be

implemented in two different ways by central projection and OP. By using OP, it can not only guarantee to find the boundary points but also the farthest vectors from the origin along the direction of boundary vectors. A example of this concept is shown in Fig. 6.2 that two possible boundaries,  $\{\mathbf{r}_1, \mathbf{r}_3\}$  or  $\{\mathbf{r}_2, \mathbf{r}_3\}$ , could be found using central projection, but  $\{\mathbf{r}_1, \mathbf{r}_3\}$  will be the boundaries once OP is used to map the data. In this section, GCCVA is implemented with sample vectors mapping onto the hyperplane via OP.



**Figure 6.2.** Illustration of projecting data samples onto selected hyperplane via central projection and OP; where  $\text{Proj}(\mathbf{r})$  indicates the central projection and  $P_{\mathbf{U}}^{\perp}(\mathbf{r})$  indicates OP, respectively.

### GCCVA

#### 1. Initial Conditions:

- a) Determine a hyperplane perpendicular to  $\mathbf{u}$ .
- b) Project all data sample vectors  $\mathbf{r}_j$  orthogonally onto the hyperplane  $\mathbf{e}^T \mathbf{u} = 1$  as  $\tilde{\mathbf{r}}_j, j = 1, \dots, N$ , where  $N$  is the total number of data samples in the image.
- c) Let  $\mathbf{e}_0$  be an initial vector randomly selected from the data and  $\tilde{\mathbf{e}}_0$  be the

corresponding projection. Search all data sample vectors to find  $\tilde{\mathbf{e}}_1$  corresponding to  $\mathbf{e}_1$  that yields maximum distance to  $\tilde{\mathbf{e}}_0$  by  $\tilde{\mathbf{e}}_1 = \arg\{\max_{\tilde{\mathbf{r}}} S_1(\tilde{\mathbf{e}}_0, \tilde{\mathbf{r}})\}$ . Let  $\mathbf{U} = \{\tilde{\mathbf{e}}_1\}$ .

2. At  $k \geq 2$ , find  $h_{k-1}$  that solves

$$\tilde{\mathbf{e}}_k = \arg \max_{\tilde{\mathbf{r}} \in \langle \mathbf{U} \rangle^\perp} \{\tilde{\mathbf{r}}^T \tilde{\mathbf{r}}\} = \arg \max_{\tilde{\mathbf{r}} \in \langle \mathbf{U} \rangle^\perp} \left\{ \left( P_{\mathbf{U}}^\perp \tilde{\mathbf{r}} \right)^T \left( P_{\mathbf{U}}^\perp \tilde{\mathbf{r}} \right) \right\}. \quad (6.6)$$

3. Find  $\mathbf{e}_k$  corresponding to  $\tilde{\mathbf{e}}_k$ .

4. Stopping rule:

If  $k < p$ , then  $k \leftarrow k+1$  and go to step 3. Otherwise, the final set of  $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_p\}$  is the desired  $p$  endmembers.

#### 6.4 Recursive Geometric Convex Cone Volume Algorithm (RGCCVA)

Since the maximum volume of a bounded convex cone problem can be considered as the maximum volume of a simplex lying inside the determined hyperplane, the key of making GCCVA work more efficiently is to derive the volume of the augmented simplex without re-calculating from the previous vertices. Using the recursive equation (4.11) derived in Chapter 5 for OP-SGA, GCCVA can also be implemented in a recursive manner by updating  $P_{\mathbf{U}^k}^\perp$  from  $P_{\mathbf{U}^{k-1}}^\perp$ . The recursive version of GCCVA, so called Recursive Geometric Convex Cone Volume Algorithm (RGCCVA), can be described as follows.

## RGCCVA

### 1. Initial Conditions:

- a) Determine a hyperplane perpendicular to  $\mathbf{u}$ .
- b) Project all data samples  $\mathbf{r}_j$  orthogonally onto the hyperplane  $\mathbf{e}^T \mathbf{u} = 1$  as  $\tilde{\mathbf{r}}_j, j = 1, \dots, N$ , where  $N$  is the total number of data sample vectors in the image.
- c) Let  $\mathbf{e}_0$  be an initial vector randomly selected from the data and  $\tilde{\mathbf{e}}_0$  be the corresponding projection. Search all data sample vectors to find  $\tilde{\mathbf{e}}_1$  corresponding to  $\mathbf{e}_1$  that yields maximum distance to  $\tilde{\mathbf{e}}_0$  by  $\tilde{\mathbf{e}}_1 = \arg \left\{ \max_{\tilde{\mathbf{r}}} S_1(\tilde{\mathbf{e}}_0, \tilde{\mathbf{r}}) \right\}$ . Let  $\mathbf{U} = \{\tilde{\mathbf{e}}_1\}$ .

2. At  $k \geq 2$ , find  $h_{k-1}$  that solves  $\tilde{\mathbf{e}}_k = \arg \left\{ \max_{\tilde{\mathbf{r}} \in (\mathbf{U})^\perp} \left\| P_{\mathbf{U}}^\perp \tilde{\mathbf{r}} \right\| \right\}$  via (4.11)

3. Find  $\mathbf{e}_k$  corresponding to  $\tilde{\mathbf{e}}_k$ .

4. Stopping rule:

If  $k < p$ , then  $k \leftarrow k + 1$  and go to step 3. Otherwise, the final set of  $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_p\}$  is the desired  $p$  endmembers.

## 6.5 Experiments

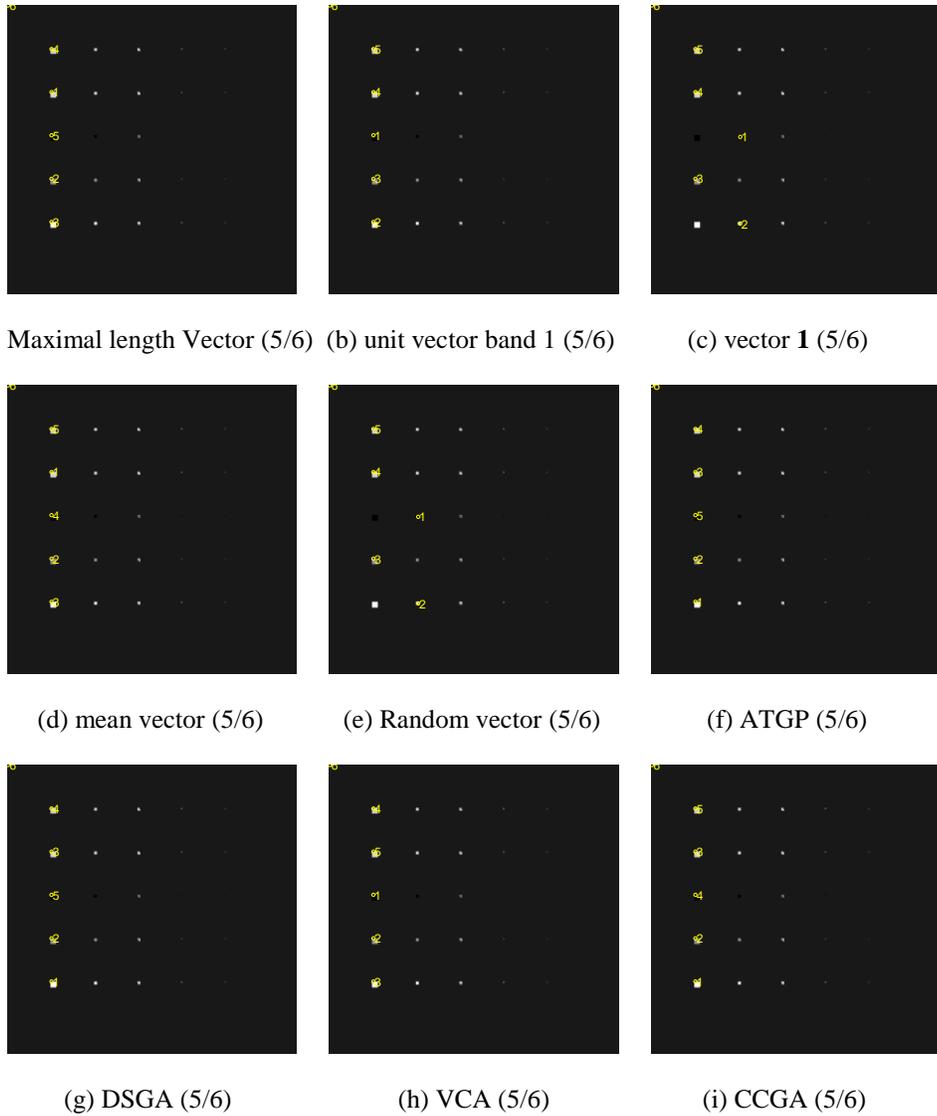
In the following experiments, GCCVA and RGCCVA were implemented with data samples orthogonal projected onto five different hyperplanes determined by vector  $\mathbf{u}$ : 1)  $\mathbf{u}$  is the sample vector with maximal vector length, 2)  $\mathbf{u}$  is a unit vector along the direction of the first band, i.e.,  $\mathbf{u} = [1, 0, 0, \dots, 0]^T$ , 3)  $\mathbf{u}$  is an all one vector  $[1, 1, \dots, 1]$ , 4)  $\mathbf{u}$  is the sample mean vector, and 5)  $\mathbf{u}$  is randomly generated, and

analyses of their found endmember are conducted to compare with Automatic Target Generation Process (ATGP) (Ren and Chang, 2003), DSGA referred in previous chapters originated from Simplex Growing Algorithm (SGA) (Chang et al., 2006), Vertex component Analysis (VCA) (Nascimento and Dias, 2005), and CCGA. To be noticed that RGCCVA finds endmembers identical to GCCVA so that only results of GCCVA were shown in Figs. 6.3-6.9. Two sets of image are used to conduct experiments. One is a set of synthetic images of six different scenarios and another is a real image data, HYDICE, as described in Chapter 2.

#### 6.5.1 Synthetic Image Experiments

In this subsection, the experimental results on six scenarios image data are conducted. As a result of having full knowledge for the dataset, the performances of various EFAs can be evaluated based on the provided ground-truth. Due to the effect caused by background, the experiments conducted by assuming the number of endmembers  $p = 6$  present in the scene with one endmember used to account for background signature. In the following results, open yellow circles represent found endmembers and the number used to label each circled pixels denotes its order.

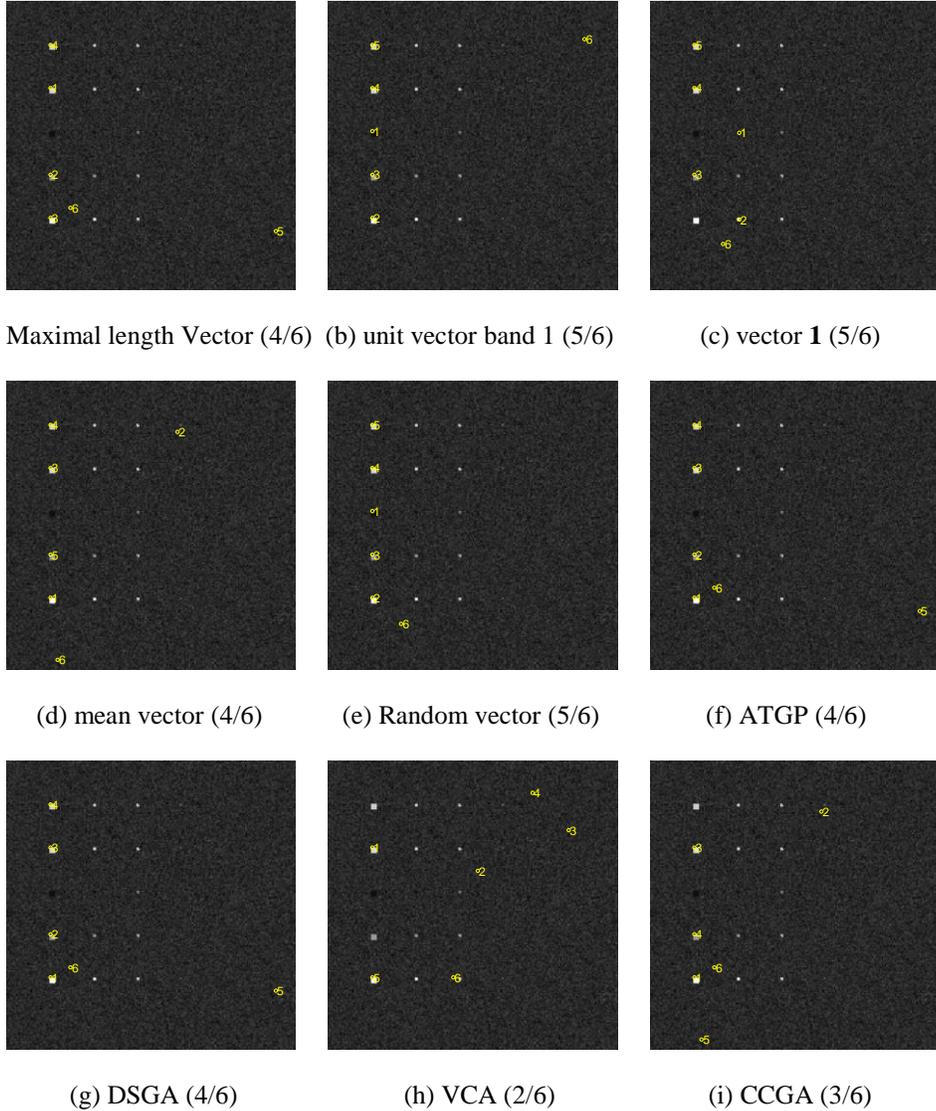
Figs. 6.3-6.5 show the result of proposed GCCVA with data samples projected onto five different hyperplanes comparing to four EFAs, ATGP, DSGA, VCA, and CCGA on three Target Implantation scenarios. As the results on TI1 shown in Fig. 6.3, no matter onto which hyperplane data samples projected, all five target panels can be extracted using GCCVA in different orders. And the other four EFAs can also find all five target panels successively.



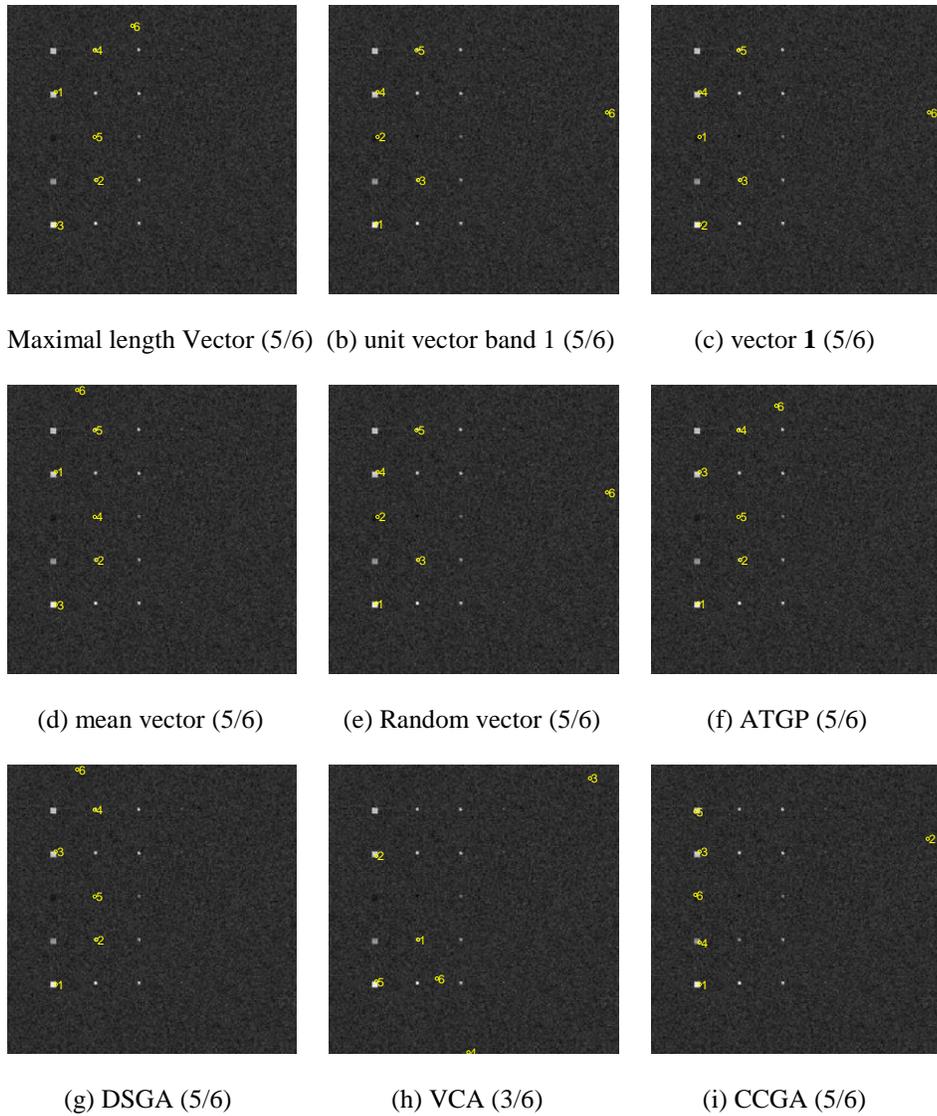
**Figure 6.3.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TI1; (f) ATGP (g) DSGA (h) VCA and (i) CCGA

With introducing of noise into background as scenario TI2, all four EFAs other than GCCVA missed panel pixel C, which spectrum is very similar to background. It should be noticed that panel pixel C can be extracted by GCCVA with samples projected onto other three hyperplanes but hyperplanes orthogonal to sample mean vector and the vector with maximal vector length as shown in Fig. 6.4. Another

convex-cone-based algorithm, VCA which initialization process is to transform data samples into a selected space, failed to extract three panel pixels terribly. These results demonstrated that noisy background have a high influence on endmember finding. Interestingly, if noise is also introduced to panel pixels the contaminated background is no longer dominant.

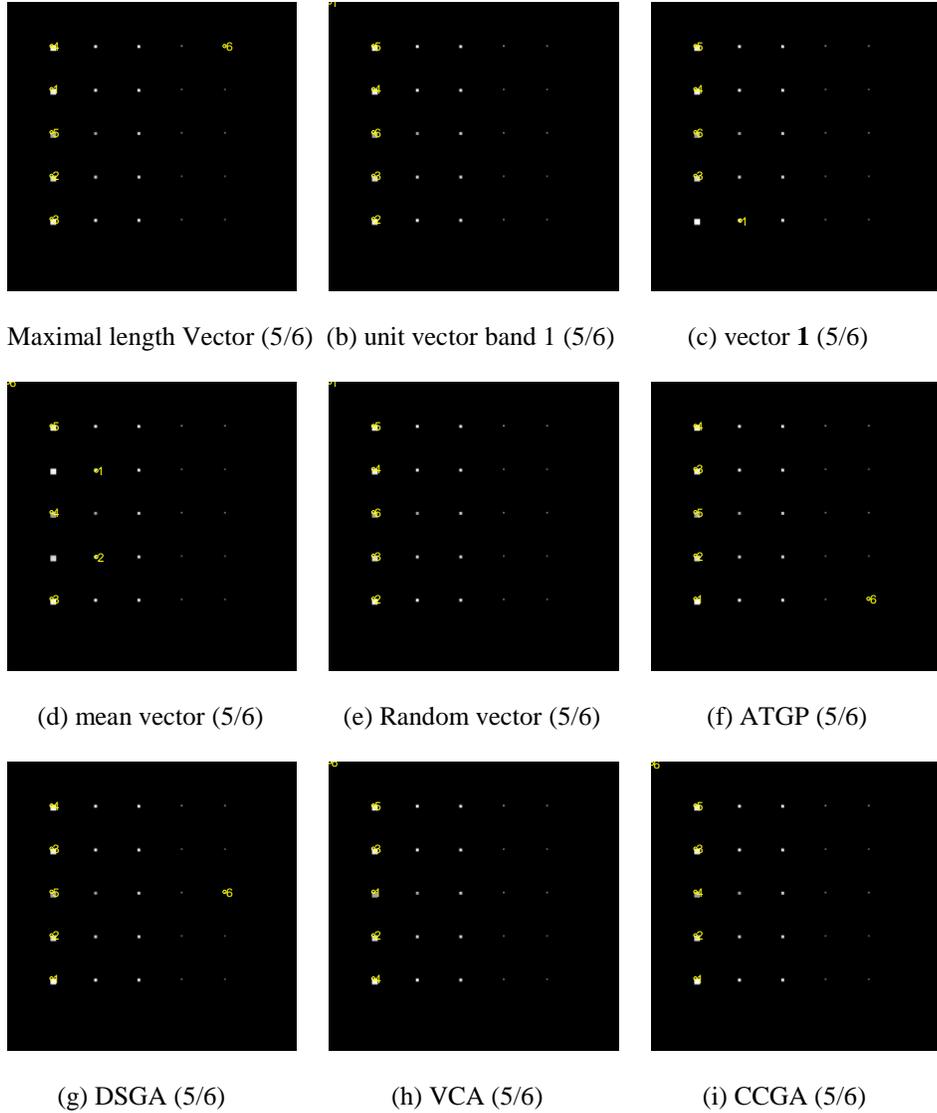


**Figure 6.4.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TI2; (f) ATGP (g) DSGA (h) VCA and (i) CCGA



**Figure 6.5.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TI3; (f) ATGP (g) DSGA (h) VCA and (i) CCGA

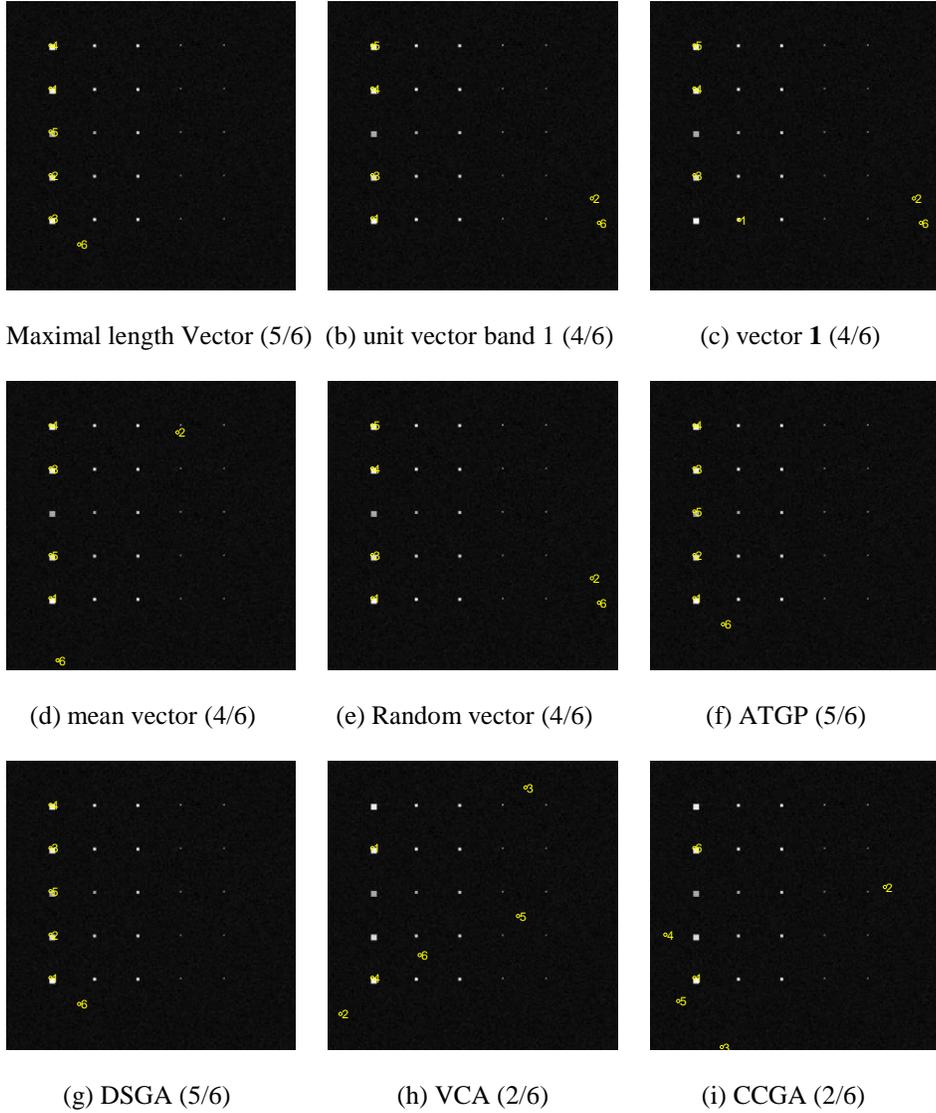
As shown in Fig. 6.5, unlike the performance on scenario TI2, GCCVA successfully extracted all panel pixels on scenario TI3 as it did on TI1. Also, ATGP, DSGA, and CCGA performed well in this scene whereas VCA still failed to find two panel pixels. Similar results can be seen as the experiments of scenarios TE images shown in Figs. 6.6-6.8.



**Figure 6.6.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TE1; (f) ATGP (g) DSGA (h) VCA and (i) CCGA

Following the same experiments conducted for scenario TI1, all EFAs including the proposed GCCVA and VCA extracted all panel pixels successfully. Different from results on scenario TI2, all five panel pixels on scenario TE2 can be extracted via ATGP, DSGA and GCCVA with data samples projected onto the hyperplane determined by vector with maximum vector length. GCCVA with other hyperplanes

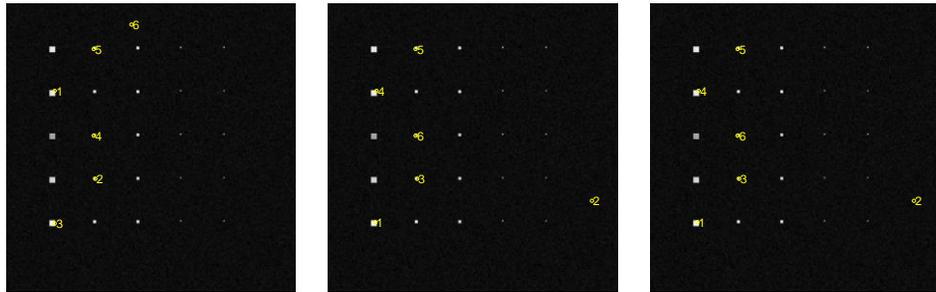
applied missed panel pixel C whereas VCA and CCGA can only extract two panels out of five.



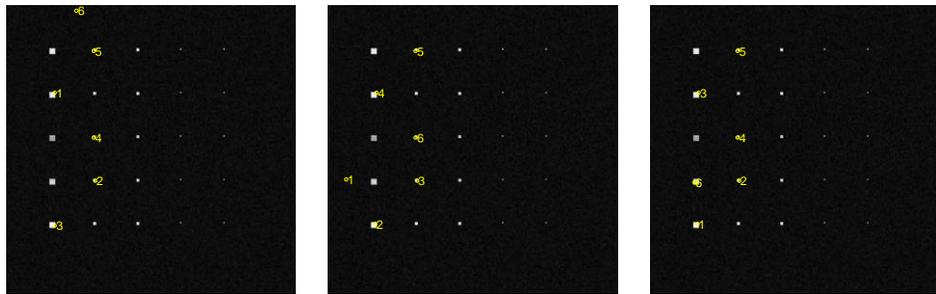
**Figure 6.7.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TE2; (f) ATGP (g) DSGA (h) VCA and (i) CCGA

Similar to scenario TI3, GCCVA, ATGP and DSGA successfully extracted all five panel pixels on scenario TE3. But interestingly, CCGA can only extract two panels out of five as VCA did. These results demonstrated an idea mentioned in

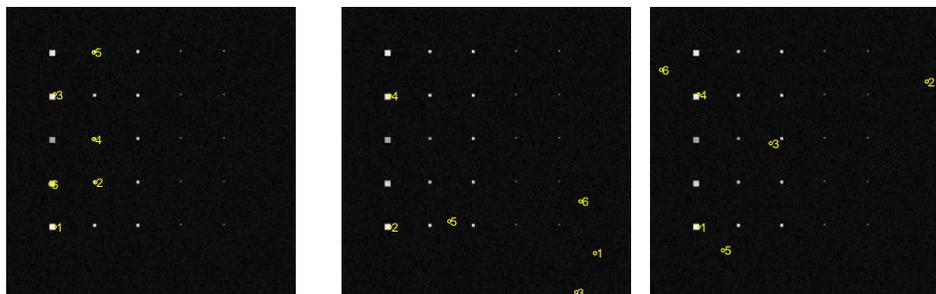
previous section that the effect of background to an embedded target can be removed by taking advantage of orthogonal projection. Unlike CCGA using central projection to project data samples onto the determined hyperplane, GCCVA uses orthogonal projection approach instead.



(a) Maximal length Vector (5/6) (b) unit vector band 1 (5/6) (c) vector 1 (5/6)



(d) mean vector (5/6) (e) Random vector (5/6) (f) ATGP (5/6)



(g) DSGA (5/6) (h) VCA (2/6) (i) CCGA (2/6)

**Figure 6.8.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on TE3; (f) ATGP (g) DSGA (h) VCA and (i) CCGA

## 6.5.2 Real Image Experiments

In this subsection, two real image data, HYDICE and Cuprite data, are used to conduct experiments. The performances of various EFAs are evaluated by the number of extracted panels for HYDICE image. And performance evaluation on Cuprite data will also be discussed later.

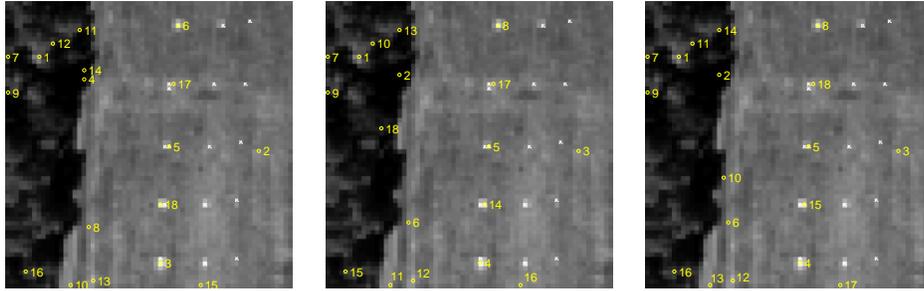
### 6.5.2.1 HYDICE Image

The first real image to be used for experiment is HYDICE dataset. Unlike the synthetic dataset with complete knowledge, the number of endmembers is unknown in this image. To address the issue, VD, coined in (Chang, 2003) and later published in (Chang and Du, 2004), is used to estimate the number of endmembers as previous chapters.

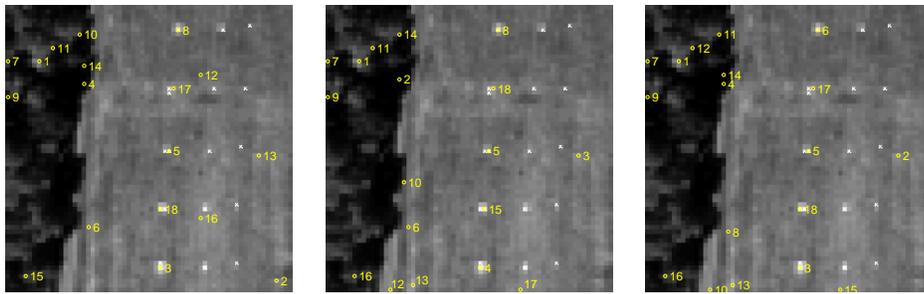
The VD estimated for HYDICE scene by Harsanyi-Farrand-Chang (HFC) method (Harsanyi et al., 1994) was  $n_{VD} = 9$ . Recently, an approach using real target signal sources was proposed for VD estimation (Chang et al., 2015), and VD was shown to be in the range between 39 and 45. Interestingly, all EFAs other than VCA and CCGA can find pure panel pixels corresponding to five panel signatures when  $n_{VD} = 18$  which is twice the value of  $n_{VD} = 9$ , which is also noted in other literatures.

Fig. 6.9 presents the endmember results of GCCVA with sample vectors projected onto different hyperplanes and other four EFAs on HYDICE scene. All of these algorithms except VCA and CCGA find five pure panel pixels corresponding to five target panels within 18 endmembers whereas CCGA finds all five panel pixels with  $n_{VD} = 35$  and VCA finds four panels out of five with  $n_{VD} = 45$ . It is worthy noted that via GCCVA with various hyperplanes to perform the projection process onto, the

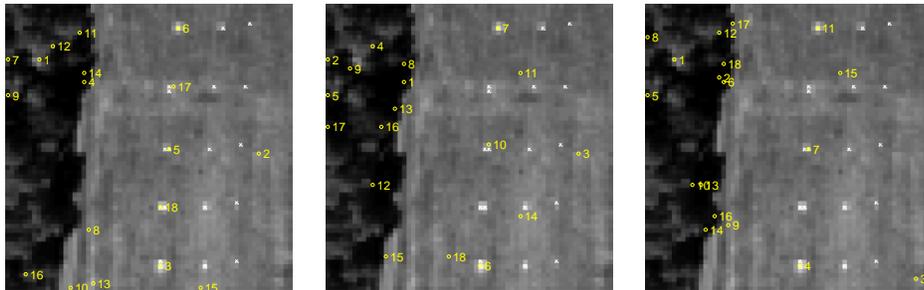
found endmembers are slightly different in the background and varied in panel pixels with different orders.



(a) Maximal length Vector (5/18) (b) unit vector band 1 (5/18) (c) vector 1 (5/18)



(d) mean vector (5/18) (e) Random vector (5/18) (f) ATGP (5/18)



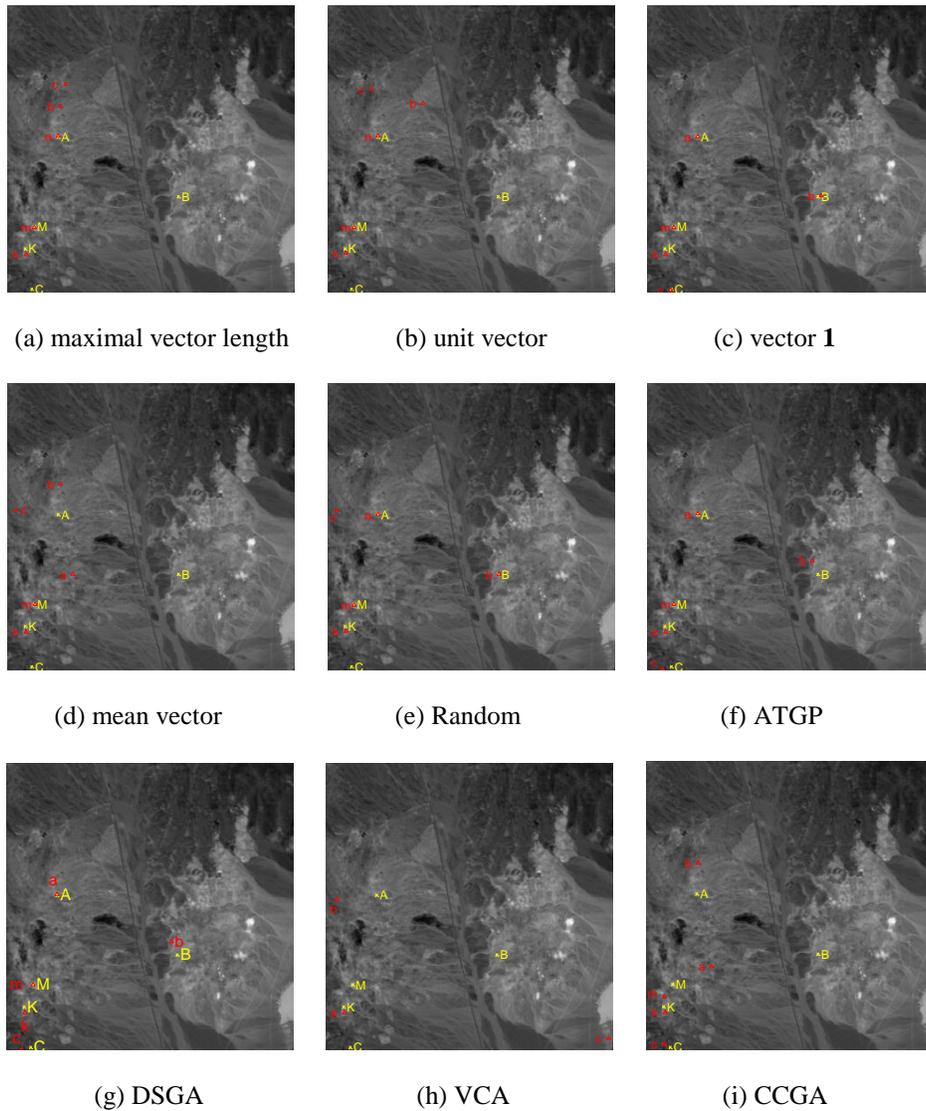
(g) DSGA (5/18) (h) VCA (3/18) (i) CCGA (3/18)

**Figure 6.9.** Endmember pixels found by (a-e) GCCVA with data samples orthogonally projected onto different hyperplane on HYDICE; (f) ATGP (g) DSGA (h) VCA and (i) CCGA

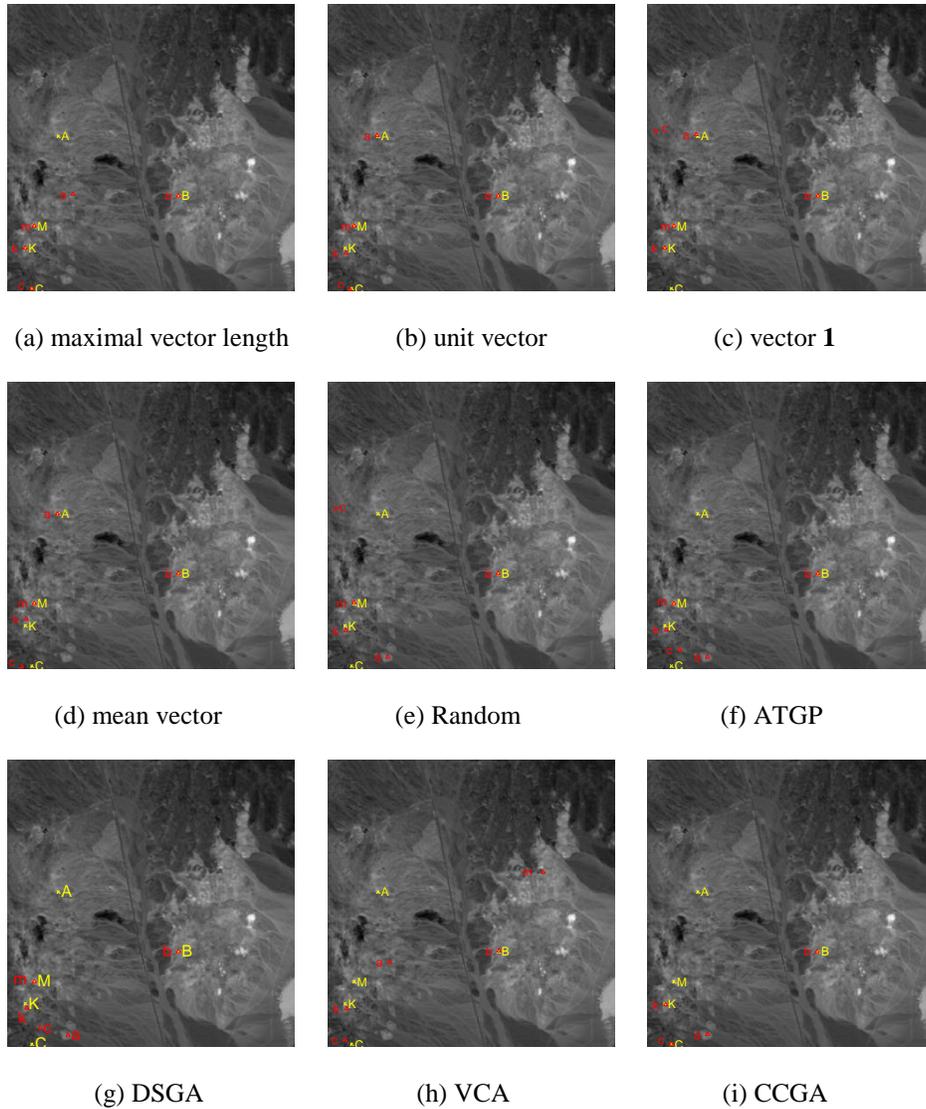
### 6.5.2.2 AVIRIS Cuprite data

In this sub-subsection, a well-known Airborne Visible Infrared Spectrometer (AVIRIS) image scene, Cuprite dataset, was used to conduct experiments. Two types of Cuprite datasets, reflectance data and radiance data, were used to validate

algorithms. Since there is no available prior knowledge about spatial locations of endmembers we must rely on an unsupervised means of identifying if an extracted target pixel is an endmember. Endmember Identification Algorithm (EIA) developed in (Chang et al., 2014) and described in Chapter 4 was used to address the issue.



**Figure 6.10.** Endmember pixels found by EIA compared to ground-truth pixels for Cuprite reflectance data; (a-e) GCCVA with data samples orthogonally projected onto different hyperplane, (f) ATGP, (g) DSGA, (h) VCA, and (i) CCGA



**Figure 6.11.** Endmember pixels found by EIA compared to ground-truth pixels for Cuprite radiance data; (a-e) GCCVA with data samples orthogonally projected onto different hyperplane, (f) ATGP, (g) DSGA, (h) VCA, and (i) CCGA

Figs. 6.10-6.11 show the spatial locations of various EFAs found endmembers identified by EIA. The pixels marked by the lower case of “a”, “b”, “c”, “k”, “m” with red triangles are the desired endmember pixels found by EIA that correspond to the five ground-truth mineral endmembers marked by the upper cases of “A”, “B”,

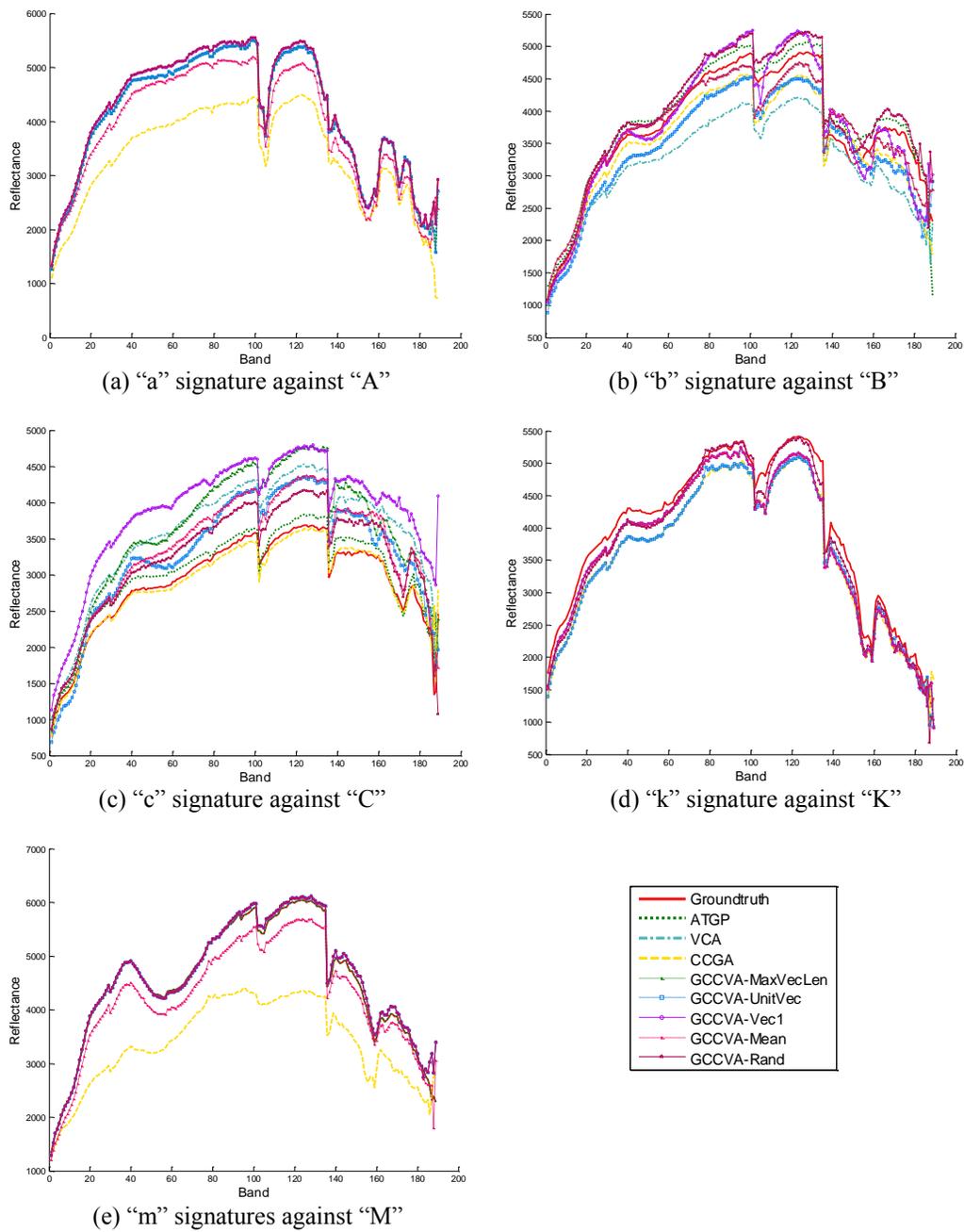
“C”, “K”, “M” with yellow crosses symbols in the sense of spectral similarity measured by SAM and spectral information divergence (SID) (Chang, 2003).

Table 6.1 tabulates the five desired endmember pixels found by EIA among ATGP-found target pixels,  $\mathbf{t}_j^{\text{ATGP}}$ , VCA-found target pixels,  $\mathbf{t}_j^{\text{VCA}}$ , CCGA-found target pixels,  $\mathbf{t}_j^{\text{CCGA}}$ , and GCCVA-found target pixels with all five different initial hyperplanes,  $\mathbf{t}_j^{\text{GCCVA}}$ , where the subscript  $j$  indicates the order of a particular target pixel was found.

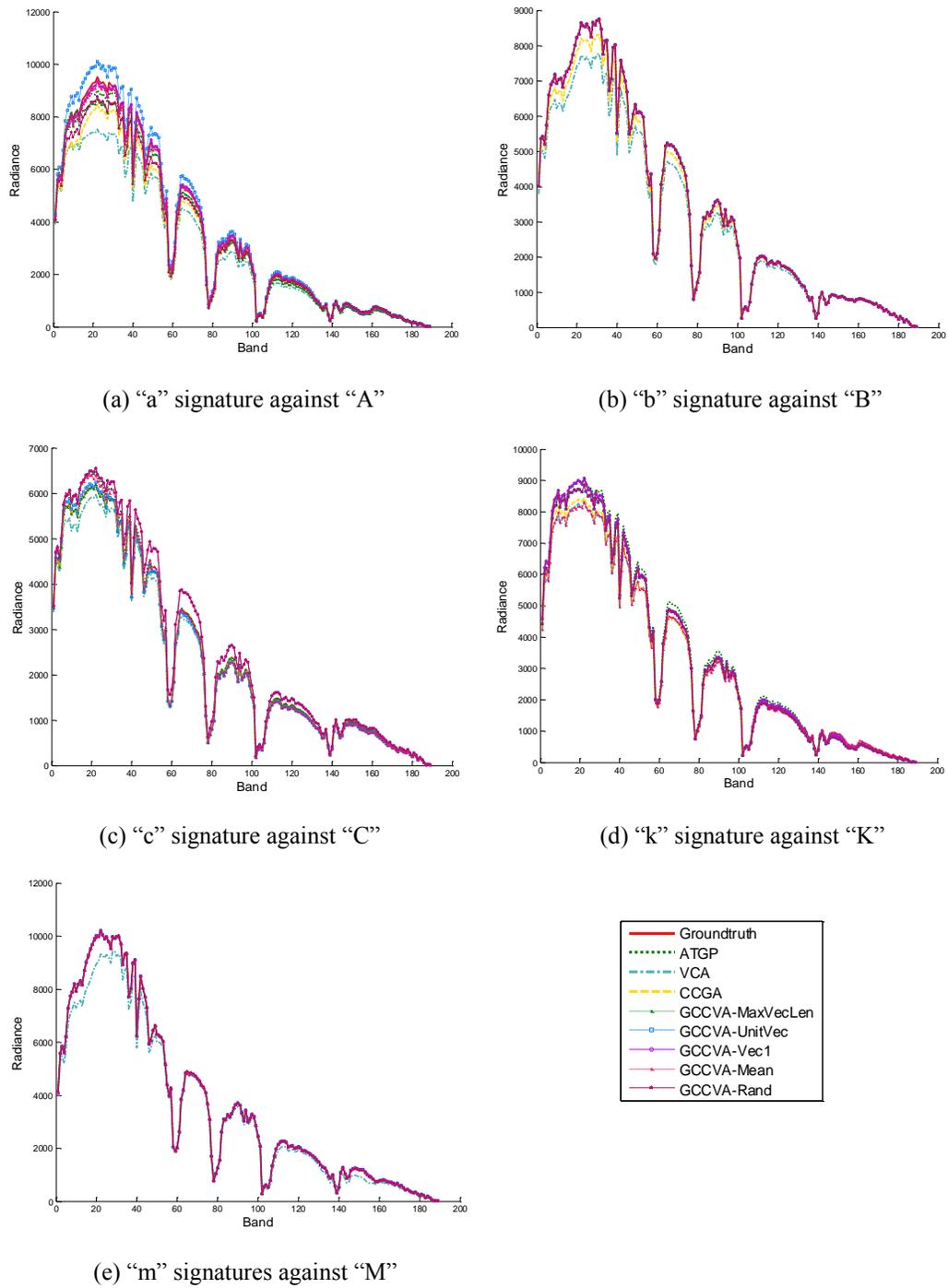
**Table 6.1.** Endmember pixels found by EIA

	Cuprite	A	B	C	K	M
ATGP	Reflectance	$\mathbf{t}_{11}^{\text{ATGP}}$	$\mathbf{t}_{13}^{\text{ATGP}}$	$\mathbf{t}_{44}^{\text{ATGP}}$	$\mathbf{t}_{64}^{\text{ATGP}}$	$\mathbf{t}_8^{\text{ATGP}}$
	Radiance	$\mathbf{t}_{12}^{\text{ATGP}}$	$\mathbf{t}_{14}^{\text{ATGP}}$	$\mathbf{t}_{28}^{\text{ATGP}}$	$\mathbf{t}_{18}^{\text{ATGP}}$	$\mathbf{t}_6^{\text{ATGP}}$
VCA	Reflectance	F	$\mathbf{t}_{150}^{\text{VCA}}$	$\mathbf{t}_{140}^{\text{VCA}}$	$\mathbf{t}_{85}^{\text{VCA}}$	F
	Radiance	$\mathbf{t}_{18}^{\text{VCA}}$	$\mathbf{t}_{28}^{\text{VCA}}$	$\mathbf{t}_{45}^{\text{VCA}}$	$\mathbf{t}_{61}^{\text{VCA}}$	$\mathbf{t}_{23}^{\text{VCA}}$
CCGA	Reflectance	$\mathbf{t}_{42}^{\text{CCGA}}$	$\mathbf{t}_{71}^{\text{CCGA}}$	$\mathbf{t}_{16}^{\text{CCGA}}$	$\mathbf{t}_{23}^{\text{CCGA}}$	$\mathbf{t}_{74}^{\text{CCGA}}$
	Radiance	$\mathbf{t}_{29}^{\text{CCGA}}$	$\mathbf{t}_{19}^{\text{CCGA}}$	$\mathbf{t}_{42}^{\text{CCGA}}$	$\mathbf{t}_{21}^{\text{CCGA}}$	F
GCCVA- MaxVecLen	Reflectance	$\mathbf{t}_{12}$	$\mathbf{t}_{46}$	$\mathbf{t}_{63}$	$\mathbf{t}_{72}$	$\mathbf{t}_{10}$
	Radiance	$\mathbf{t}_{23}$	$\mathbf{t}_{18}$	$\mathbf{t}_{29}$	$\mathbf{t}_{12}$	$\mathbf{t}_6$
GCCVA- unit vector	Reflectance	$\mathbf{t}_{18}$	$\mathbf{t}_{48}$	$\mathbf{t}_{80}$	$\mathbf{t}_{26}$	$\mathbf{t}_8$
	Radiance	$\mathbf{t}_{20}$	$\mathbf{t}_{14}$	$\mathbf{t}_{53}$	$\mathbf{t}_{45}$	$\mathbf{t}_7$
GCCVA- vector 1	Reflectance	$\mathbf{t}_{52}$	$\mathbf{t}_{77}$	$\mathbf{t}_{87}$	$\mathbf{t}_{27}$	$\mathbf{t}_{10}$
	Radiance	$\mathbf{t}_{25}$	$\mathbf{t}_{17}$	$\mathbf{t}_{27}$	$\mathbf{t}_{16}$	$\mathbf{t}_6$
GCCVA- mean vector	Reflectance	$\mathbf{t}_{25}$	$\mathbf{t}_{45}$	$\mathbf{t}_{15}$	$\mathbf{t}_{71}$	$\mathbf{t}_{13}$
	Radiance	$\mathbf{t}_{53}$	$\mathbf{t}_{15}$	$\mathbf{t}_{27}$	$\mathbf{t}_{70}$	$\mathbf{t}_6$
GCCVA- rand vec	Reflectance	$\mathbf{t}_{12}$	$\mathbf{t}_{38}$	$\mathbf{t}_{55}$	$\mathbf{t}_{28}$	$\mathbf{t}_9$
	Radiance	$\mathbf{t}_{43}$	$\mathbf{t}_{17}$	$\mathbf{t}_{29}$	$\mathbf{t}_{46}$	$\mathbf{t}_{10}$

As we can see from Table 6.1 for the reflectance data, VCA failed to find mineral signatures A and M while the other EFAs found the last mineral signature within 80 target pixels to complete all the five mineral signatures. For radiance data, CCGA failed to find mineral signature M while the other EFAs found the last mineral signature within 70 target pixels to complete all the five mineral signatures.



**Figure 6.12.** Comparative plots of spectral signatures found by GCCVA, ATGP, VCA and CCGA on Cuprite reflectance data



**Figure 6.13.** Comparative plots of spectral signatures found by GCCVA, ATGP, VCA and CCGA on Cuprite radiance data

Nonetheless, all EFAs besides VCA found the same first mineral signature which was M because the spectrum of M is probably the most distinct among the five mineral signatures which can be seen in Figs. 2.1(c)-2.1(d). Once the spatial locations of the five desired endmember pixels were found in Figs. 6.10-6.11 we can further perform a comparative spectral analysis between EIA-identified endmember pixels in Table 6.1 and the ground-truth pixels in Fig. 2.1(b). For each of spectral signatures of the five mineral signatures, A, B, C, K, M both in reflectance data and radiance data, Figs. 6.12-6.13 plot nine spectra including spectrum of ground-truth pixel, and spectrum of EIA identified pixel spectra of endmember results using different EFAs listed in Table 6.1 for comparison.

While the plots of Figs. 6.12 and 6.13 offer an advantage of visual assessment about how close a found endmember pixel to a ground truth pixel, it does not provide quantitative measurements on their spectral similarity. Tables 6.2 and 6.3 calculate spectral similarity values of the plots among identified target pixels against the ground-truth pixels in Figs. 6.12 and 6.13 where SAM and SID were used as spectral measure.

As we can see from Tables 6.2 and 6.3, the spectral similarity values among all EFAs-found pixels compared to the ground-truth pixels were indeed very close even the found pixels by these EFAs were identified by EIA in different locations. GCCVA with the initial hyperplane determined by the vector with maximal vector length outperformed the other EFAs on Cuprite radiance data since the average of spectral similarity between identified endmember pixels and the ground-truth pixels is the smallest compared to the others.

**Table 6.2.** SAM/SID of the closet target pixels with ground-truth by variants of geometric convex cone algorithms on Cuprite reflectance data

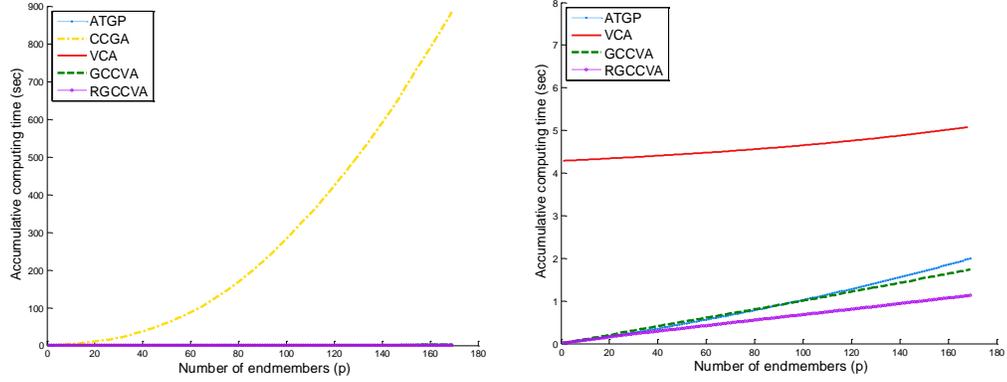
SAM SID	(A,a)	(B,b)	(C,c)	(K,k)	(M,m)
ATGP	0.0167 0.0002	0.0334 0.0009	0.0379 0.0009	0.0341 0.0007	0 0
VCA	F F	0.0613 0.0020	0.0411 0.0011	0.0341 0.0007	F F
CCGA	0.0580 0.0030	0.0589 0.0018	0.0374 0.0009	0.0314 0.0007	0.0700 0.0027
GCCVA- MaxVecLen	0.0167 0.0002	0.0623 0.0020	0.0436 0.0011	0.0341 0.0007	0.0264 0.0005
GCCVA- unit vector	0.0167 0.0002	0.0573 0.0017	0.0506 0.0018	0.0304 0.0006	0.0264 0.0005
GCCVA- vector 1	0 0	0.0670 0.0024	0.0493 0.0013	0.0341 0.0007	0.0264 0.0005
GCCVA- mean vector	0.0235 0.0004	0.0623 0.0020	0.0350 0.0008	0.0341 0.0007	0.0249 0.0004
GCCVA- rand vec	0 0	0.0218 0.0003	0.0418 0.0013	0.0348 0.0008	0.0264 0.0005

**Table 6.3.** SAM/SID of the closet target pixels with ground-truth by variants of geometric convex cone algorithms on Cuprite radiance data

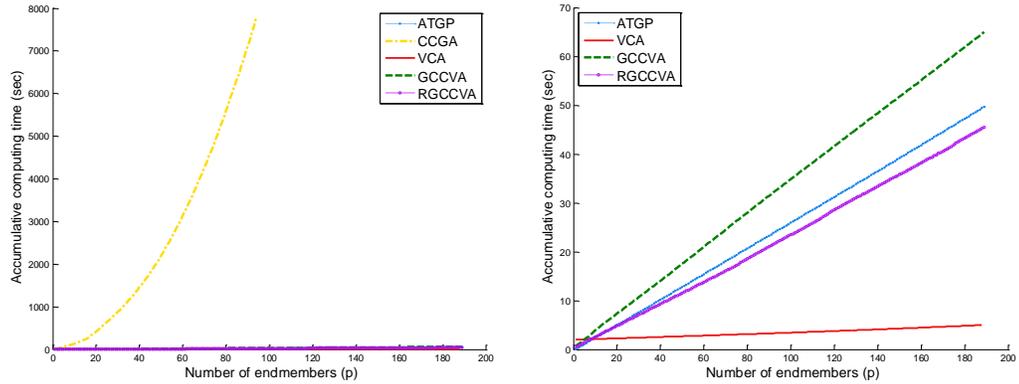
SAM SID	(A,a)	(B,b)	(C,c)	(K,k)	(M,m)
ATGP	0.0205 0.0003	0 0	0.0247 0.0006	0.0219 0.0003	0 0
VCA	0.0344 0.0007	0.0151 0.0003	0.0108 0	0.0128 0.0001	0.0328 0.0010
CCGA	0.0223 0.0006	0.0082 0.0001	0 0	0.0081 0.0001	F F
GCCVA- MaxVecLen	0.0230 0.0004	0 0	0 0	0 0	0 0
GCCVA- unit vector	0.0098 0.0001	0 0	0.0181 0.0003	0.0122 0.0002	0 0
GCCVA- vector 1	0.0086 0.0001	0 0	0.0279 0.0004	0 0	0 0
GCCVA- mean vector	0.0048 0	0 0	0.0261 0.0006	0.0202 0.0009	0 0
GCCVA- rand vec	0.0147 0.0003	0 0	0.0279 0.0004	0.0122 0.0002	0.0058 0

### 6.5.2.3 Computer Processing Time

In order to count a fair comparative analysis for convex cone based algorithms on computing time, ATGP, CCGA with SV calculated via SVD, VCA, GCCVA, and RGCCVA were run to find the bounded convex cone from 1-simplex up to  $L$ -simplex,  $L = 169$  and  $L = 189$  for HYDICE and Cuprite data, respectively, and further calculated their corresponding processing time. Figs. 6.14 and 6.15 plot their computer processing time for HYDICE where RGCCVA was the best and CCGA with SVD calculated via SVD was the worst regarding in computing time. It is worth noted that the computing time of VCA varies with size of data and is not stable compared to other algorithms. As it can be seen in Figs. 6.14 and 6.15, VCA took the longest computing time for HYDICE, but it was the best algorithm in computing time for Cuprite data. Overall, CCGA was the worst algorithm with worst computing time performance among all the other algorithms while R-GCCVA is ranked as 1<sup>st</sup> and 2<sup>nd</sup> best in computing time for HYDICE and Cuprite data, respectively. Although VCA performs better in terms of computing time, it did not produce good endmember results unless VCA used ATGP-generated target pixels as initial conditions regardless of its savings in computing time which was pointed out in (Chang, 2013; Chen, 2014).



**Figure 6.14.** Accumulative computing time in seconds of ATGP, CCGA with SV calculated via SVD, VCA, GCCVA, and RGCCVA as  $p$  increases on HYDICE data



**Figure 6.15.** Accumulative computing time in seconds of ATGP, CCGA with SV calculated via SVD, VCA, GCCVA, and RGCCVA as  $p$  increases on Cuprite data

## 6.6 Conclusions

This chapter introduces an approach which converts convex cone analysis to convex cone volume analysis. It projects a convex cone onto a hyperplane so that the volume of the projected convex cone can be realized by a simplex on the hyperplane. In this case, a new convex cone growing algorithm can be developed to ease computational complexity of SGA. As noted, SV calculation by the matrix determinant causes numerical instability in SGA. Geometric simplex volume (GSV)

calculation is applied to address this issue. As a result, a Geometric Convex Cone Volume Algorithm (GCCVA) is further developed. However, how to determine a desired hyperplane remains an issue. Fortunately, according to our extensive experiments conducted in this chapter, the hyperplane actually does not play an important role since the use of the central projection with center  $\mathbf{0}$  keeps the convex cone invariant no matter which hyperplane is chosen for projection. Furthermore, it is also shown by experiments that using OP instead of the central projection with center  $\mathbf{0}$ , GCCVA can successfully reduce the background effect to find endmembers more effectively. Moreover, a recursive version of GCCVA, called RGCCVA, can be derived to update convex cone volumes via OP in a recursive manner. The experimental results demonstrated that GCCVA developed in this chapter outperformed other convex cone-based algorithms in terms of finding endmembers. Also, RGCCVA not only provides the same results identical to the results found by GCCVA but also significantly reduces computational cost via updating volumes of bounded convex cones recursively.

## Chapter 7: CONCLUSIONS

### 7.1 Summary

The maximal Simplex Volume (SV) has been used as a major criterion to design Endmember Finding Algorithms (EFAs), especially Simplex Growing Algorithm (SGA), which was developed to resolve several issues arising from well-known N-finder algorithm (N-FINDR). Although SGA successfully reduces high computational complexity resulting from exhaustive search for finding maximal-SV simplexes, its use of matrix determinant to calculate SV turns out not to produce true SV. In addition, calculating matrix determinants suffers from numerical instability. In particular since the number of endmembers is generally much smaller than the number of bands, computing the matrix determinant becomes more complicated in which case Dimensionality Reduction (DR) usually implemented prior to SV calculation. The research performed in this dissertation takes up these issue and focused on design and development of maximal SV-based EFAs. It converts direct finding SV via the matrix determinant to finding the height of simplexes by interpreting SV calculation as the multiplication of the height and base of a simplex from a geometric point of view.

In Chapter 3, a concept of using geometric structure was proposed to address the issue encountered in SV calculation via the matrix determinant. The proposed geometric volume calculation approach is very simple and easy to be implemented. It offers several benefits that determinant-based SV approaches cannot provide. One is no requirement of DR. Another is that the geometric approach gives the true SV

without running into the numerical issues when a simplex to be calculated has very high dimensionality. Third, the computational complexity is significantly reduced by its geometric structure since it only needs to find the height of a simplex. Finally, it can avoid finding incorrect endmembers caused by the numerical errors resulting from using determinant-based SV approaches.

The concept of Geometric SV (GSV) calculation is first proposed in Chapter 4 to replace SGA using matrix determinant to calculate SV, referred to as determinant-based SGA (DSGA). Two such GSV-based EFAs are proposed, to be called Geometric SGA (GSGA) and Orthogonal Projection-based SGA (OP-SGA). By taking advantage of Gram-Schmidt Orthogonalization process (GSOP) to find heights of simplexes and Orthogonal Subspace Projection (OSP) to find Orthogonal Projection (OP), SV can be calculated and updated by new generated endmembers without re-computing volumes by matrix determinants. As a result, computational complexity significantly reduced.

In order to realize real-time processing ability, Chapter 5 derives recursive equations to be used to further develop recursive versions of GSGA and OP-SGA called Recursive GSGA (RGSGA) and Recursive OP-SGA (ROP-SGA) so that both of them can be implemented as Kalman filter-like algorithms which can be easily implemented in hardware design.

Finally, Chapter 6 presents another alternative geometric concept to simplex. By taking advantage of GSV calculation, it extends a recent work, Convex Cone Growing Analysis (CCGA) developed by Xiong et al. (2010), to Geometric Convex

Cone Volume Analysis (GCCVA) as well as its recursive version, Recursive GCCVA (R-GCCVA).

## 7.2 Contributions

There are several contributions made in this dissertation, each of which can be described as follows.

1. Simplex volume analysis (Chapter 3)
  - a. Investigates and analyzes the issue in calculating volume of simplexes from two different aspects, geometric structure and eigen-analysis.
  - b. Proposes a simplex volume calculation method via the geometric structure of simplexes.
2. Design and development of variants of SGA (Chapter 4)
  - a. Develops two new EFAs, OP-SGA and GSGA, with SV calculation in SGA replaced by GSV calculation.
  - b. Conducts a comparative analysis and study among various SV-based SGA and GSV-based SGA.
3. Recursive growing simplex volume analysis (Chapter 5)
  - a. Develops recursive versions of OP-SGA and GSGA, called ROP-SGA and RGSGA.
  - b. Derives Kalman filter-like algorithms for ROP-SGA and RGSGA which can be easily implemented in real-time and hardware design.
  - c. Provides an effective means of determining VD in an unsupervised fashion.

4. Geometric convex cone volume analysis (Chapter 6)
  - a. Addresses the issue in volume calculation of bounded convex cones, and applies the GSV approach proposed in Chapter 3 to resolve the issue.
  - b. Explores the issue of determination of hyperplanes that generate different bounded convex cone.
  - c. Develops a new EFA using convex cone and its recursive version.

### 7.3 Future work

This dissertation develops several GSV-based EFAs and demonstrates their effectiveness and efficiency by experiments. One of the best benefits is recursive versions of GSV-based SGA which can be used to design real-time process algorithms as well as their hardware implementations. Another is applications in real-time band processing according to a hyperspectral data acquisition format, Band SeQuential (BSQ). Since the number of endmembers,  $p$ , is relatively small compared to band dimensionality, it generally requires DR to reduce data dimensionality to  $p$ . As an alternative to DR, Band Selection (BS) can also be used for this purpose. However, BS must be performed prior to endmember finding where the number of bands must be determined in advance and appropriate bands also must be selected beforehand. Therefore, there is no way for BS to provide information about how each band has effect on endmember finding. Interestingly, BSQ allows EFAs to process data band by band progressively so that various endmembers can be found through different stages of band processing. Consequently, one potential application is to extend OP-

SGA and GSGA to their progressive band processing versions in such a manner that OP-SGA and GSGA can be carried out according to BSQ band by band progressively to provide progressive endmember finding maps from which we can see different levels of difficulty with finding various endmembers.

## Bibliography

Berger, Marcel. *Geometry Revealed: A Jacob's Ladder to Modern Higher Geometry*.

Heidelberg: Springer, 2010. Print.

Bioucas-Dias, J. M., Plaza, A., Dobigeon, N., Parente, M., Du, Q., Gader, P., and Chanussot, J, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5.2 (2012): 354-379.

Boardman, J.W., "Automated spectral unmixing of AVIRIS data using convex geometry concepts," *Summaries, Fourth JPL Airborne Geoscience Workshop*, JPL Publication 93-26, Vol. 1 (1993): 11-14.

Boardman, J.W., "Geometric mixture analysis of imaging spectrometry data," *International Geoscience Remote Sense Symposium*, vol. 4 (1994): 2369-2371.

Chang, Chein-I, *Hyperspectral Imaging: Techniques for Spectral detection and Classification*, Kluwer Academic/Plenum Publishers, 2003.

Chang, C.-I and Du, Q., "Estimation of number of spectrally distinct spectral signal sources in hyperspectral imagery," *IEEE Trans. on Geoscience and Remote Sensing*, vol. 42 (2004): 608-619.

Chang, C.-I, Wu, C., Liu, W., and Ouyang, Y.C., “A growing method for simplex-based endmember extraction algorithms,” *IEEE Trans. on Geoscience and Remote Sensing*, vol. 44 (2006): 2804-2819.

Chang, C.-I, Jiao, X., Du, Y., and Chang, M.-L., “A review of unsupervised hyperspectral target analysis,” *EURASIP Journal on Advanced in Signal Processing*, 2010.1 (2010): 1-26.

Chang, C.-I, Jiao, X., Du, Y., and Chen H.-M., “Component analysis-based unsupervised linear spectral mixture analysis for hyperspectral imagery,” *IEEE Trans. on Geoscience and Remote Sensing*, 49.11 (2011): 4123-4137.

Chang, C.-I, Xiong, W., Chen, H.M., and Tsai, J.W., “Maximum orthogonal subspace projection to estimating number of spectral signal sources for hyperspectral images,” *IEEE Journal of Selected Topics in Signal Processing*, 5.3 (2011): 504-520.

Chang, Chein-I, *Hyperspectral Data Processing: Signal Processing Algorithm Design and Analysis*, Hoboken, NJ: Wiley-Interscience, 2013.

Chang, C.-I, Wen, C.H., and Wu, C.C., “Relationship exploration among PPI, ATGP and VCA via theoretical analysis,” *Int. J. of Computational Science and Engineering*, 8.4 (2013): 361-367.

Chang, C.-I, Xiong, W., and Wen, C.H., “A theory of high order statistics-based virtual dimensionality for hyperspectral imagery,” *IEEE Trans. on Geoscience and Remote Sensing*, 52.1 (2014): 188-208.

Chang, C.-I, Paylor, D., Lee, L.-C., “Virtual dimensionality analysis for hyperspectral imagery,” *SPIE Sensing Technology+ Applications*. International Society for Optics and Photonics, 2015.

Chen, S.Y., *Algorithm Design and Analysis for Hyperspectral Endmember Finding*, Ph.D. dissertation, Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD, May 2014.

Craig, M.D., “Minimum-volume transforms for remotely sensed data,” *IEEE Trans. Geoscience Remote Sensing*, 32.3 (1994): 542-552.

Du, Q., Raksuntorn, N., Younan, N.H., and King, K.L., “Endmember extraction in hyperspectral images,” *Applied Optics*, 47.28 (2008): F77-F84.

Du, Q., “A new sequential algorithm for hyperspectral endmember extraction,” *IEEE Geoscience and Remote Sensing Letters*, 9.4 (2012): 695-699.

Friedberg, Stephan H., Insel, Arnold J., and Spence, Lawrence E., *Linear Algebra*, 4<sup>th</sup> edition, Prentice Hall, 2003.

Gallier, Jean, *Basics of Projective Geometry. In Geometric Methods and Applications (pp. 103-175)*, New York: Springer, 2011.

Gao, C., Chen, S.-Y., Chen, H.-M., Wu, C.-C., Wen, C.-H., and Chang, C.-I, "Fully abundance-constrained endmember finding for hyperspectral images," *7<sup>th</sup> Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, (WHISPERS)*, Tokyo, Japan, 2-5 June, 2015.

Geng, X., Sun, K., Ji, L., Zhao, Y., "A fast volume-gradient-based band selection method for hyperspectral image," *IEEE Transactions on Geoscience and Remote Sensing*, 52.11 (2014): 7111-7119.

Harsanyi, J.C., Farrand W., and Chang, C.-I, "Detection of subpixel spectral signatures in hyperspectral image sequences," *Annual Meeting, Proceedings of American Society of Photogrammetry & Remote Sensing*, Reno, pp. 236-247, 1994.

Heinz, D. and Chang, C.-I, "Fully constrained least squares linear mixture analysis for material quantification in hyperspectral imagery," *IEEE Trans. on Geoscience and Remote Sensing*, 39.3 (2001): 529-545.

Ifarragaerri, A. and Chang C.-I, "Hyperspectral image segmentation with convex cones," *IEEE Trans. on Geoscience and Remote Sensing*, 37.2 (1999): 756-770.

Kuybeda, O., Malah D., and Barzohar, M., "Rank estimation and redundancy reduction of high-dimensional noisy signals with preservation of rare vectors," *IEEE Trans. on Signal Processing*, 55.12 (2007): 5579-5592.

Kwizera, P., "Matrix Singular Value Decomposition," UNF Theses and Dissertations Paper 381, 2010.

Leadbetter, M.R., Lindgren, Georg, and Rootzen, Holger, *Extremes and Related Properties of Random Sequences and Processes*, New York: Springer-Verlag, 1983.

Ma, W. K., Bioucas-Dias, J. M., Chan, T. H., Gillis, N., Gader, P., Plaza, A. J., Ambikapathi, A., and Chi, C. Y., "A signal processing perspective on hyperspectral unmixing," *IEEE Signal Processing Magazine*, 31.1 (2014): 67-81.

Nascimento, J. M. and Dias, J. M. B., "Vertex component analysis: a fast algorithm to unmix hyperspectral data," *IEEE Trans. Geoscience and Remote Sensing*, 43.4 (2005): 898-910.

Neville, R.A., Staenz, K., Szeredi, T., Lefebvre J., and Hauff, P., "Automatic endmember extraction from hyperspectral data for mineral exploration," In: *Proc. 4th*

*International Airborne Remote Sensing Conference and Exhibition/21st Canadian Symposium on remote Sensing*, Ottawa, Ontario, Canada, pp. 21-24, June 1999.

Poor, H. Vincent, *An Introduction to Signal Detection and Estimation*, New York: Springer, 1994.

Ren, H., and Chang, C.-I, "Automatic spectral target recognition in hyperspectral imagery," *IEEE Trans. on Aerospace and Electronic Systems*, 39.4 (2003): 1232-1249.

Schowengerdt, Robert A., *Remote Sensing: Models and Methods for Image Processing*, 2nd Ed., New York: Academic Press, 1997.

Stein, P., "A note on the volume of a simplex", *Amer. Math. Monthly*, 73.3 (1996): 299–301.

Wang, J. and Chang, C.-I, "Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis," *IEEE Trans. on Geoscience and Remote Sensing*, 44.6 (2006): 1586-1600.

Wang, L., Liu D., and Wang, Q. "Geometric method of fully constrained least squares linear spectral mixture analysis," *IEEE Trans. on Geoscience and Remote Sensing*, 51.6 (2013): 3558-3566.

Wang, L., Wei, F., Liu, D., and Wang, Q., "Fast implementation of maximum simplex volume-based endmember extraction in original hyperspectral data space," *IEEE Journal of Selected Topics in Applied Earth Observation and Remote Sensing*, 6.2 (2013): 516-521.

Winter, M.E., "N-finder: an algorithm for fast autonomous spectral endmember determination in hyperspectral data," *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, 1999.

Winter, M.E., "Fast autonomous spectral endmember determination in hyperspectral data," *Proc. Of 13<sup>th</sup> International Conference on Applied Geologic Remote Sensing*, Vancouver, B.C., Canada, vol. II, pp. 337-344, 1999.

Wong, W.W., "Application of Linear Algebra," (2003).

Wu, C.C., Chu, S., and Chang, C.-I., "Sequential N-FINDR algorithm," SPIE Conference on *Imaging Spectrometry XIII*, August 10-14, San Diego, 2008.

Xiong, W., Tsai, C.T., Yang, C.W., and Chang, C.-I., "Convex cone-based endmember extraction for hyperspectral imagery," SPIE Conference on *Imaging Spectrometry XV*, vol. 7812, San Diego, CA, August 2-5, 2010.

Xiong, W., Wu, C.-C., Chang, C.I, Kapalkis, K., and Chen, H.M., “Fast algorithms to implement N-FINDR for hyperspectral endmember extraction,” *IEEE Journal of Selected Topics in Applied Earth Observation and Remote Sensing*, 4.3 (2011): 545-564.

“Simplex Volumes and the Cayley-Menger Determinant,” Dec. 2014, 2 Mar. 2016,  
<<http://mathpages.com/home/kmath664/kmath664.htm>>

Jet Propulsion Laboratory, “Airborne Visible/Infrared Spectrometer,” 2 Mar. 2016,  
<<http://aviris.jpl.nasa.gov/>>

